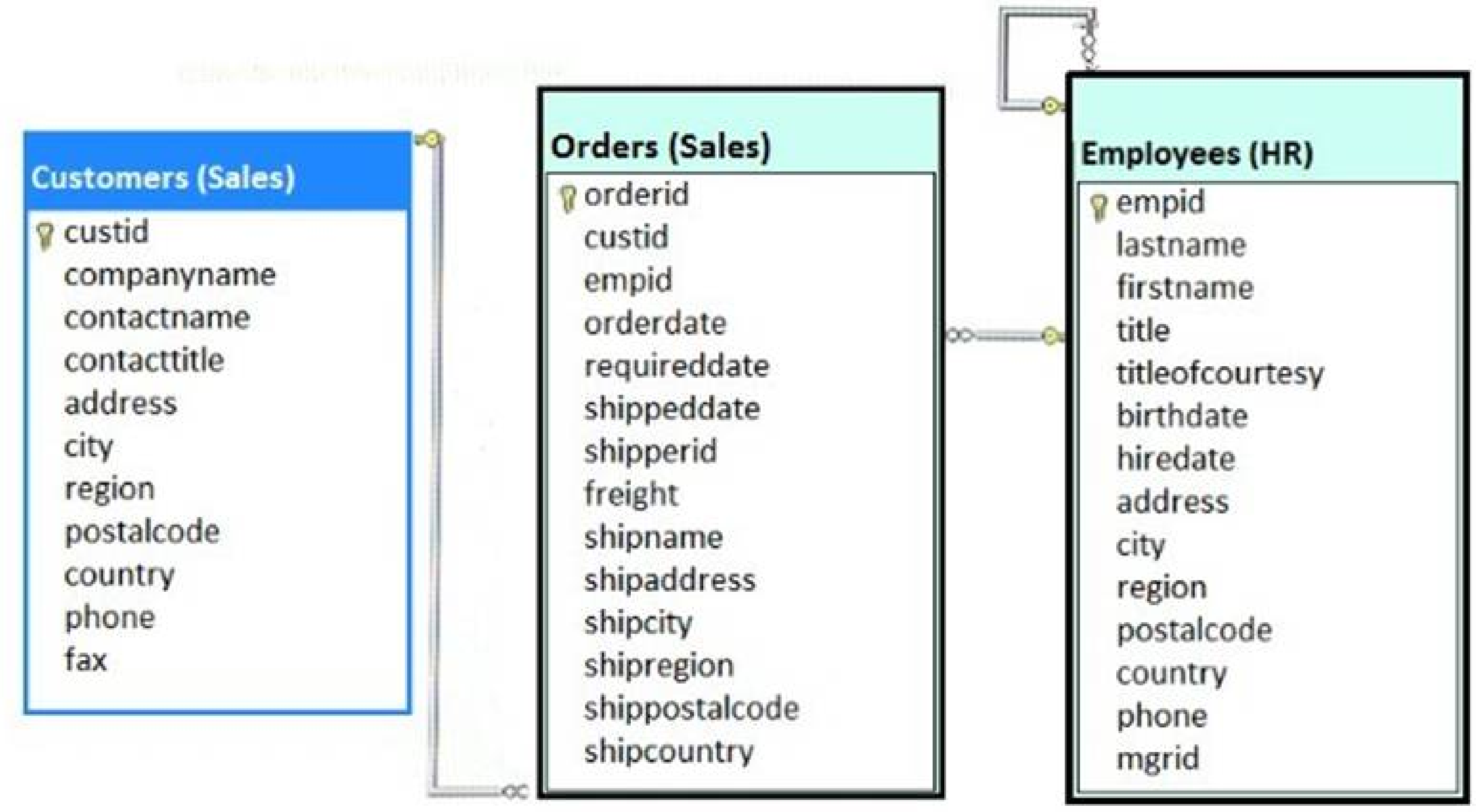
Exam Questions 70-761

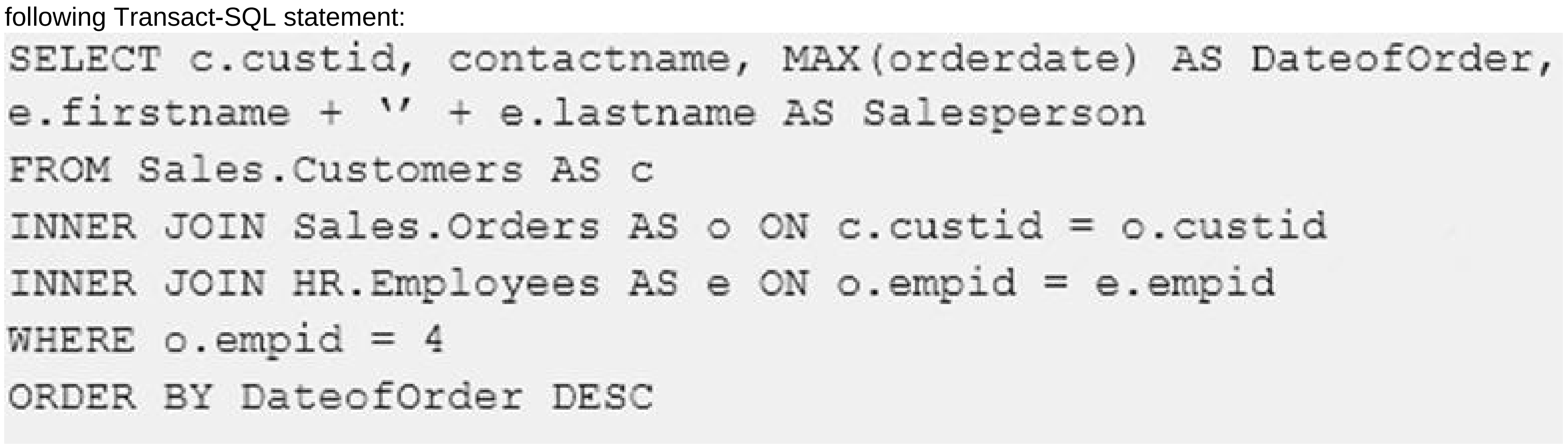
Querying Data with Transact-SQL (beta)

https://www.2passeasy.com/dumps/70-761/

NEW QUESTION 1
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a database that includes the tables shown in the exhibit (Click the Exhibit button.)



You need to create a Transact-SQL query that returns the following information:
- the customer number
- the customer contact name
- the date the order was placed, with a name of DateofOrder
- a column named Salesperson, formatted with the employee first name, a space, and the employee last name
- orders for customers where the employee identifier equals 4
The output must be sorted by order date, with the newest orders first. The solution must return only the most recent order for each customer. Solution: You run the following Transact-SQL statement:



```
SELECT c.custid, contactname, MAX(orderdate) AS DateofOrder,
e.firstname + ' ' + e.lastname AS Salesperson
FROM Sales.Customers AS c
INNER JOIN Sales.Orders AS o ON c.custid = o.custid
INNER JOIN HR.Employees AS e ON o.empid = e.empid
WHERE o.empid = 4
ORDER BY DateofOrder DESC
```

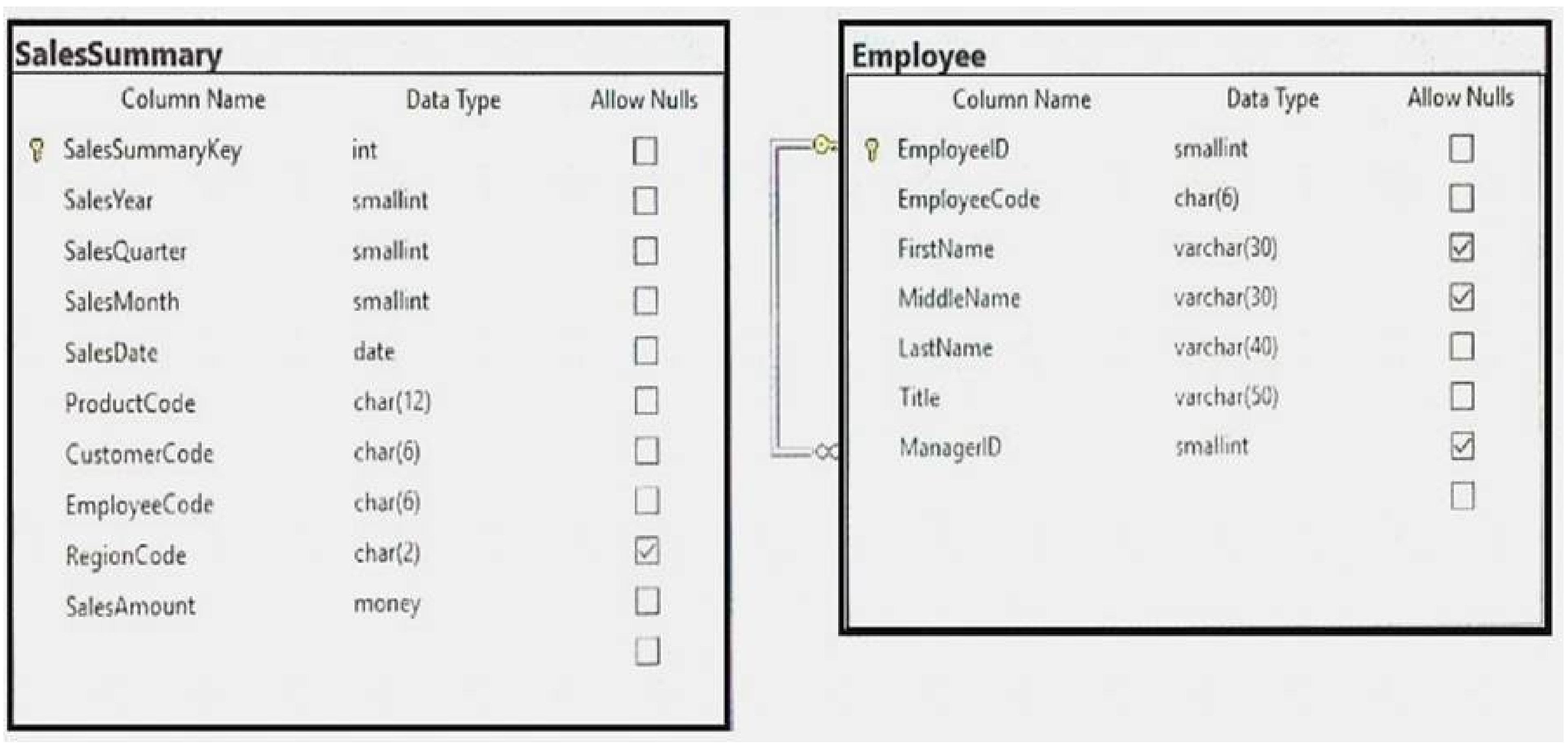
Does the solution meet the goal?

A. Yes
B. No

Answer: B

Explanation:
We need a GROUP BY statement as we want to return an order for each customer.

NEW QUESTION 2
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.)



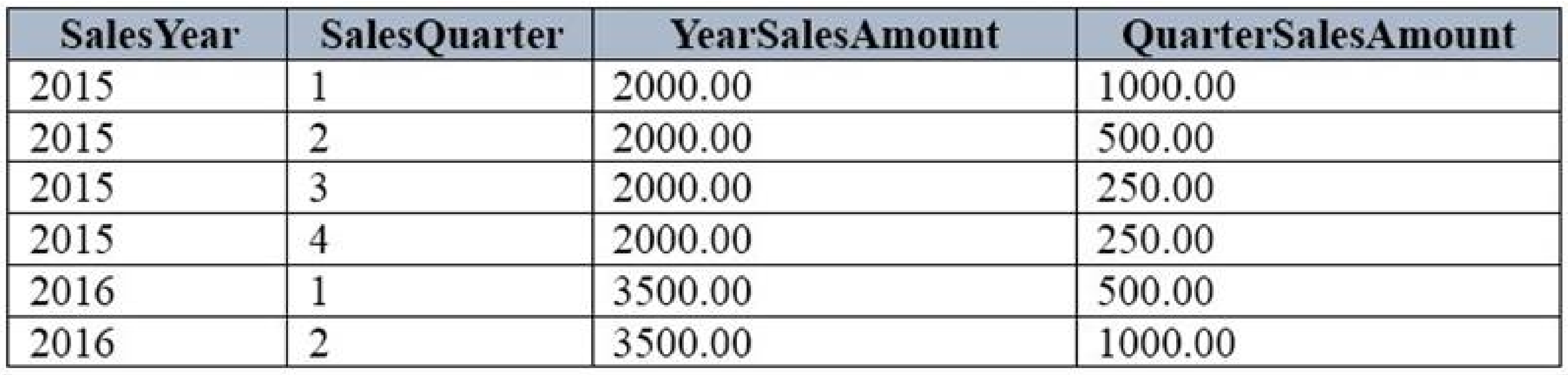
SalesSummary

Column Name	Data Type	Allow Nulls
SalesSummaryKey	int	☐
SalesYear	smallint	☐
SalesQuarter	smallint	☐
SalesMonth	smallint	☐
SalesDate	date	☐
ProductCode	char(12)	☐
CustomerCode	char(6)	☐
EmployeeCode	char(6)	☐
RegionCode	char(2)	✓
SalesAmount	money	☐
		☐

Employee

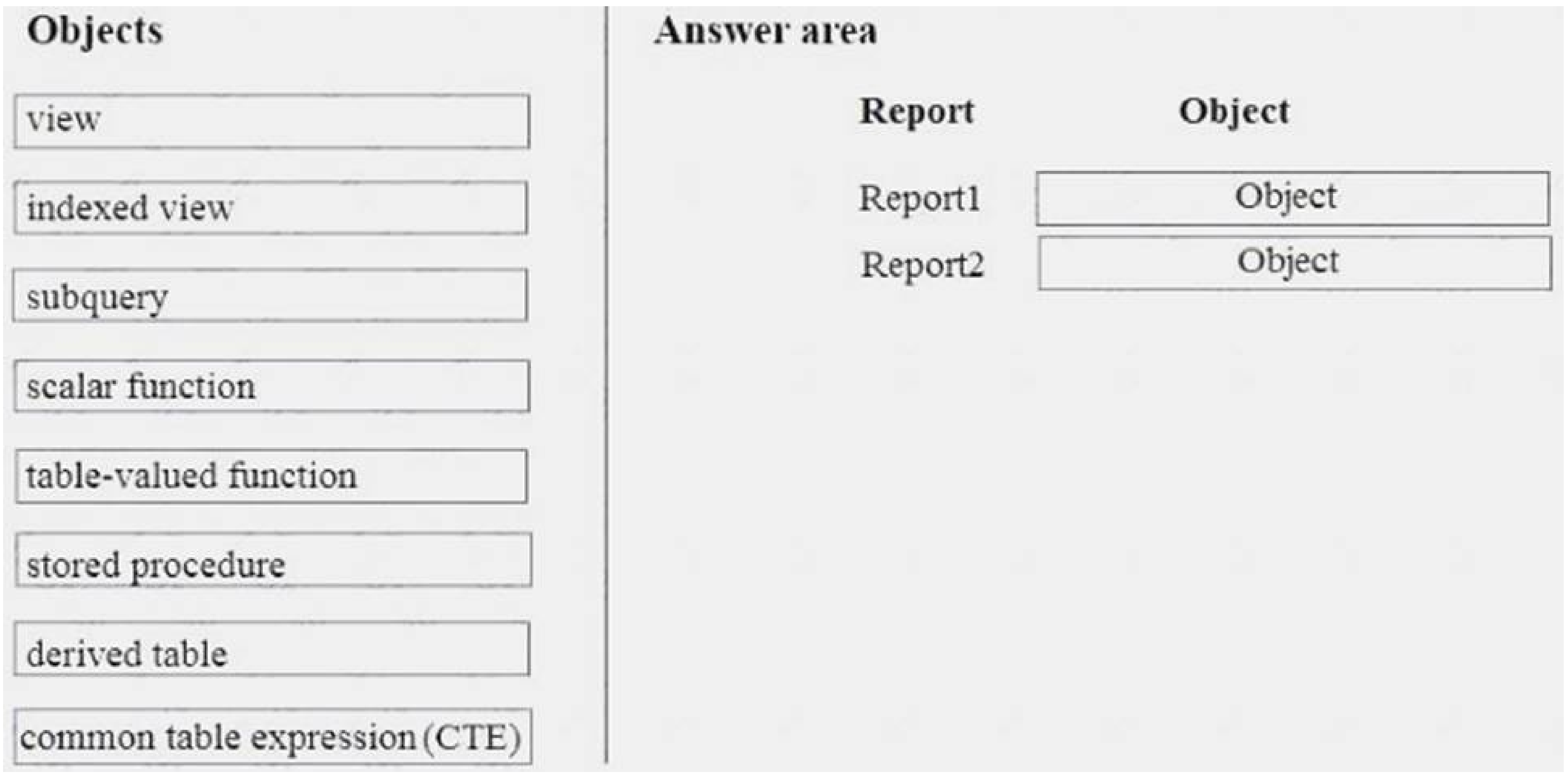
Column Name	Data Type	Allow Nulls
EmployeeID	smallint	☐
EmployeeCode	char(6)	☐
FirstName	varchar(30)	✓
MiddleName	varchar(30)	✓
LastName	varchar(40)	☐
Title	varchar(50)	☐
ManagerID	smallint	✓
		☐

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:
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 are creating the queries for Report1 and Report2.
You need to create the objects necessary to support the queries.
Which object should you use to join the SalesSummary table with the other tables that each report uses? To answer, drag the appropriate objects to the correct reports. each object 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.


Objects

view

indexed view

subquery

scalar function

table-valued function

stored procedure

derived table

common table expression (CTE)

Answer area

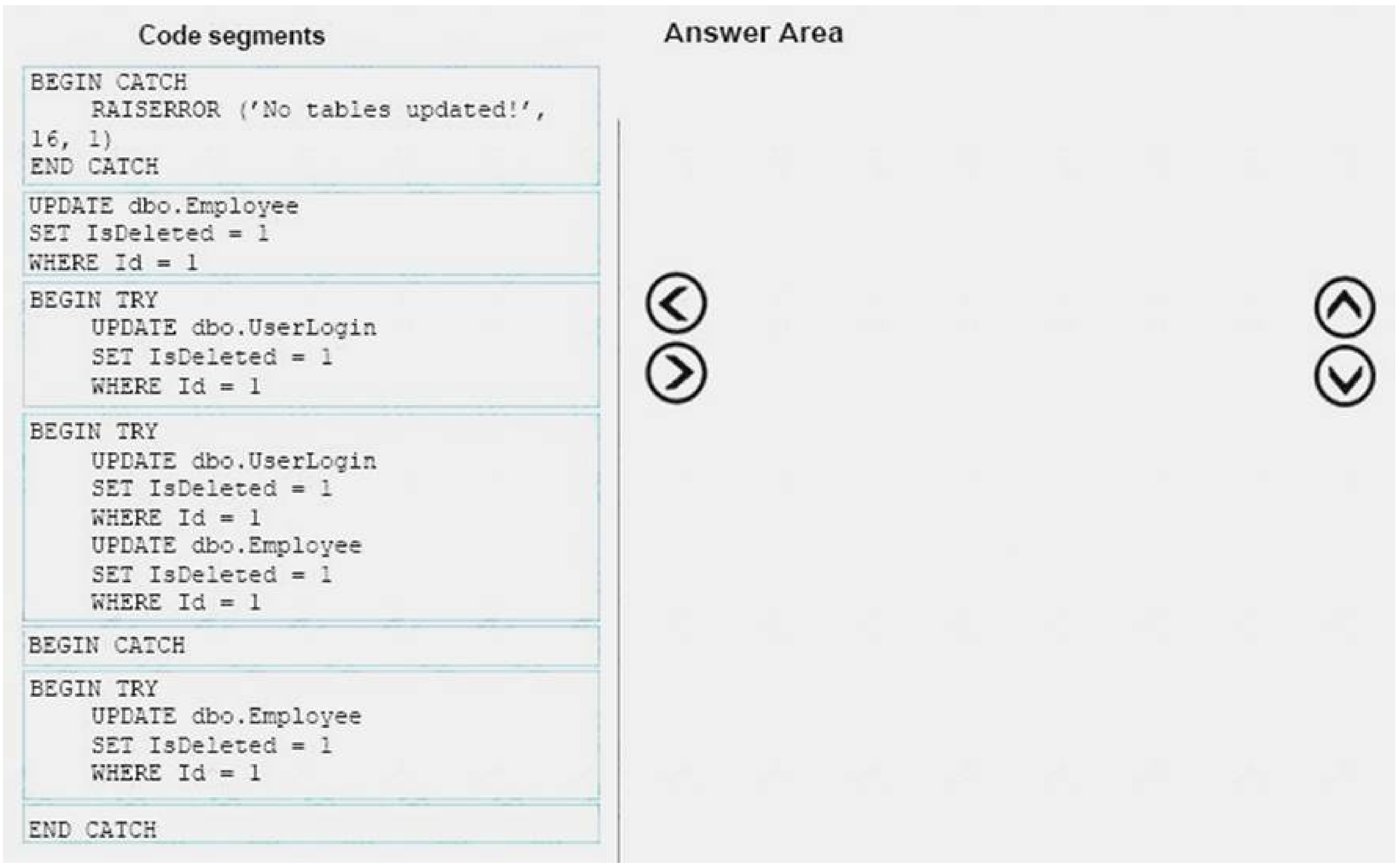
Report	Object
Report1	Object
Report2	Object

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Box 1: common table expression (CTE)
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.
A CTE can be used to:
From Scenario: 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:
Box 2: view
From scenario: 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:
References: https://technet.microsoft.com/en-us/library/ms190766(v=sql.105).aspx

NEW QUESTION 3
You have two tables named UserLogin and Employee respectively.
You need to create a Transact-SQL script that meets the following requirements:
* The script must update the value of the IsDeleted column for the UserLogin table to 1 if the value of the Id column for the UserLogin table is equal to 1.
* The script must update the value of the IsDeleted column of the Employee table to 1 if the value of the Id column is equal to 1 for the Employee table when an update to the UserLogin table throws an error.
* The error message “No tables updated!” must be produced when an update to the Employee table throws an error.
Which five 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.


Code segments

```
BEGIN CATCH
    RAISERROR ('No tables updated!',
16, 1)
END CATCH
```

```
UPDATE dbo.Employee
SET IsDeleted = 1
WHERE Id = 1
```

```
BEGIN TRY
    UPDATE dbo.UserLogin
    SET IsDeleted = 1
    WHERE Id = 1
```

```
BEGIN TRY
    UPDATE dbo.UserLogin
    SET IsDeleted = 1
    WHERE Id = 1
    UPDATE dbo.Employee
    SET IsDeleted = 1
    WHERE Id = 1
```

```
BEGIN CATCH
```

```
BEGIN TRY
    UPDATE dbo.Employee
    SET IsDeleted = 1
    WHERE Id = 1
```

```
END CATCH
```

Answer Area

A. Mastered
B. Not Mastered

Answer: A

Explanation:
A TRY block must be immediately followed by an associated CATCH block. Including any other statements between the END TRY and BEGIN CATCH statements generates a syntax error.
References: https://msdn.microsoft.com/en-us/library/ms175976.aspx

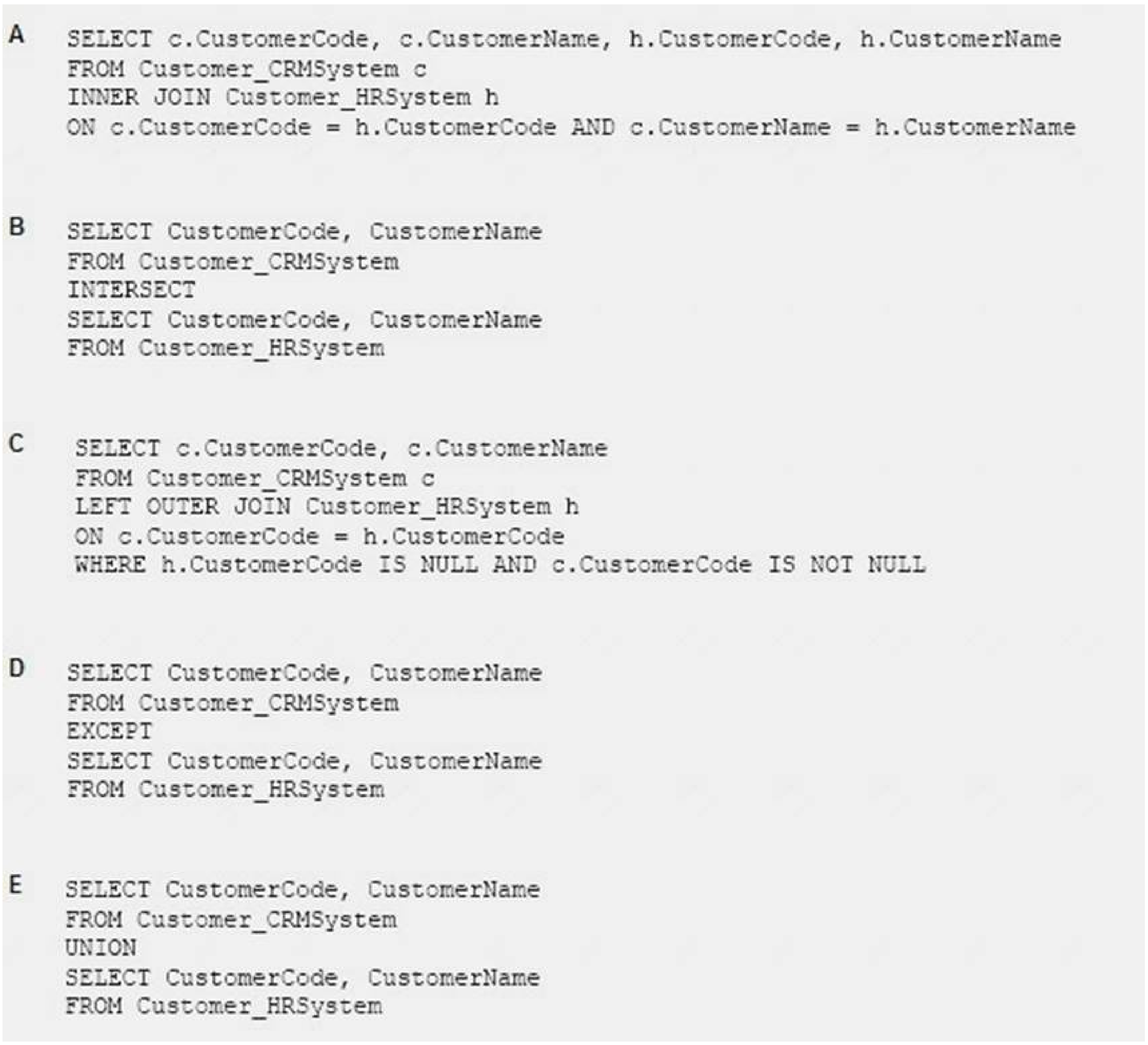
NEW QUESTION 4
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You have a database that contains tables named Customer_CRMSystem and Customer_HRSystem. Both tables use the following structure:
The tables include the following records: Customer_CRMSystem

CustomerID	CustomerCode	CustomerName
1	CUS1	Roya
2	CUS9	Almudena
3	CUS4	Jack
4	NULL	Jane
5	NULL	Francisco

Customer_HRSystem

CustomerID	CustomerCode	CustomerName
1	CUS1	Roya
2	CUS2	Jose
3	CUS9	Almudena
4	NULL	Jane

Records that contain null values for CustomerCode can be uniquely identified by CustomerName.
You need to create a list of all unique customers that appear in either table. Which Transact-SQL statement should you run?



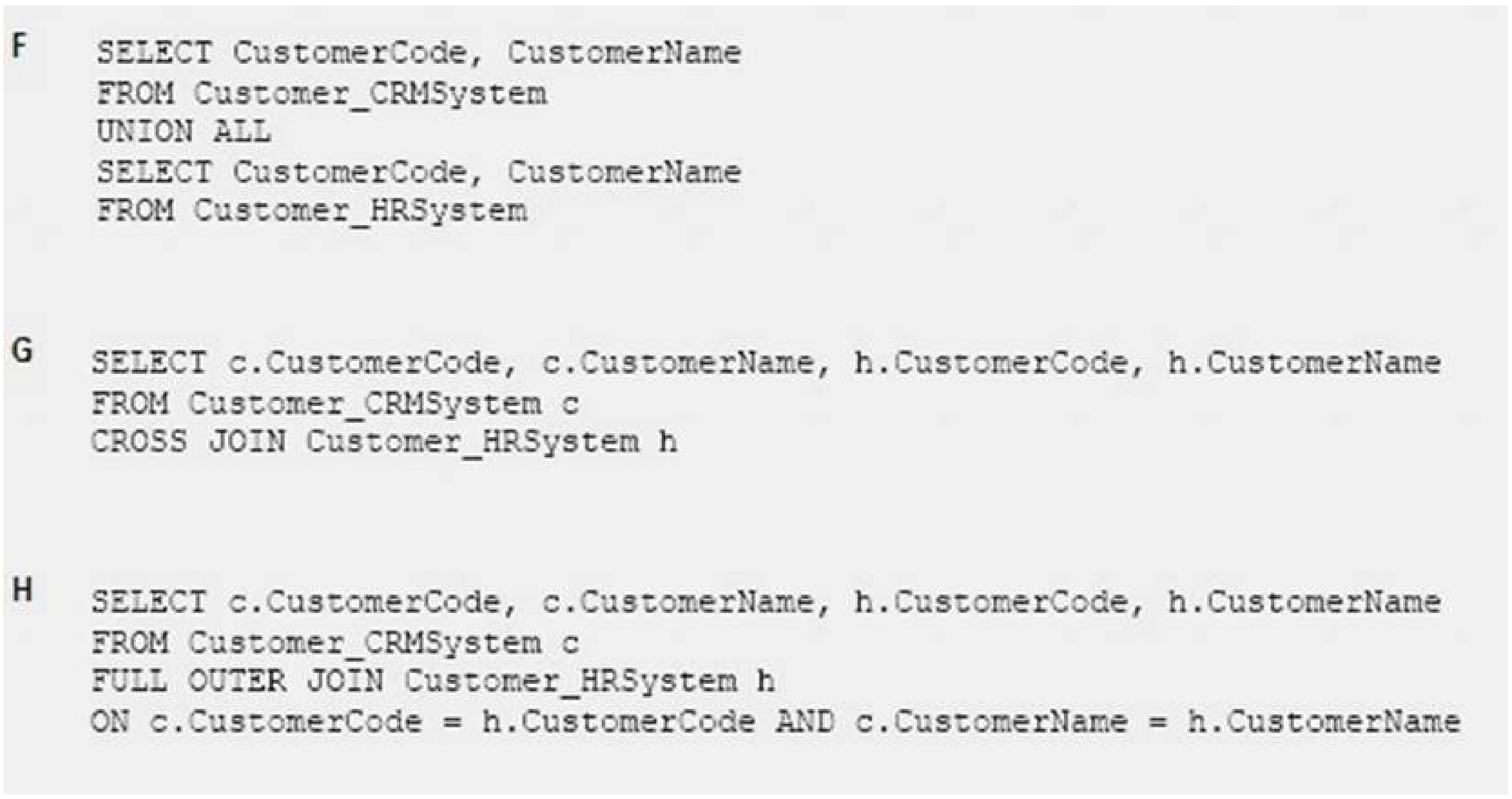
A
```
SELECT c.CustomerCode, c.CustomerName, h.CustomerCode, h.CustomerName
FROM Customer_CRMSystem c
INNER JOIN Customer_HRSystem h
ON c.CustomerCode = h.CustomerCode AND c.CustomerName = h.CustomerName
```

B
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
INTERSECT
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```

C
```
SELECT c.CustomerCode, c.CustomerName
FROM Customer_CRMSystem c
LEFT OUTER JOIN Customer_HRSystem h
ON c.CustomerCode = h.CustomerCode
WHERE h.CustomerCode IS NULL AND c.CustomerCode IS NOT NULL
```

D
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
EXCEPT
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```

E
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
UNION
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```



F
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
UNION ALL
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```

G
```
SELECT c.CustomerCode, c.CustomerName, h.CustomerCode, h.CustomerName
FROM Customer_CRMSystem c
CROSS JOIN Customer_HRSystem h
```

H
```
SELECT c.CustomerCode, c.CustomerName, h.CustomerCode, h.CustomerName
FROM Customer_CRMSystem c
FULL OUTER JOIN Customer_HRSystem h
ON c.CustomerCode = h.CustomerCode AND c.CustomerName = h.CustomerName
```

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E
F. Option F
G. Option G

H. Option H

Answer: E

Explanation:
UNION combines the results of two or more queries into a single result set that includes all the rows that belong to all queries in the union. The UNION operation is different from using joins that combine columns from two tables.

NEW QUESTION 5
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You are building a stored procedure that will be used by hundreds of users concurrently.
You need to store rows that will be processed later by the stored procedure. The object that stores the rows must meet the following requirements:
 Be indexable
 Contain up-to-date statistics
 Be able to scale between 10 and 100,000 rows
The solution must prevent users from accessing one another's data. Solution: You create a table variable in the stored procedure.
Does this meet the goal?

A. Yes
B. No

Answer: B

NEW QUESTION 6
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a table that was created by running the following Transact-SQL statement:

```
CREATE TABLE Products (
     ProductID int NOT NULL PRIMARY KEY,
     ProductName nvarchar(100) NULL,
     UnitPrice decimal(18, 2) NOT NULL,
     UnitsInStock int NOT NULL,
     UnitsOnOrder int NULL
)
```

The Products table includes the data shown in the following table:

ProductID	ProductName	UnitPrice	UnitsInStock	UnitsOnOrder
1	ProductA	10.00	10	15
2	ProductB	30.00	20	Null
3	ProductC	15.00	5	20

TotalUnitPrice is calculated by using the following formula: TotalUnitPrice = UnitPrice * (UnitsInStock + UnitsOnOrder)
You need to ensure that the value returned for TotalUnitPrice for ProductB is equal to 600.00. Solution: You run the following Transact-SQL statement:

```
SELECT ProductName, UnitPrice*(UnitsInStock+UnitsOnOrder) AS
TotalUnitPrice FROM Products
```

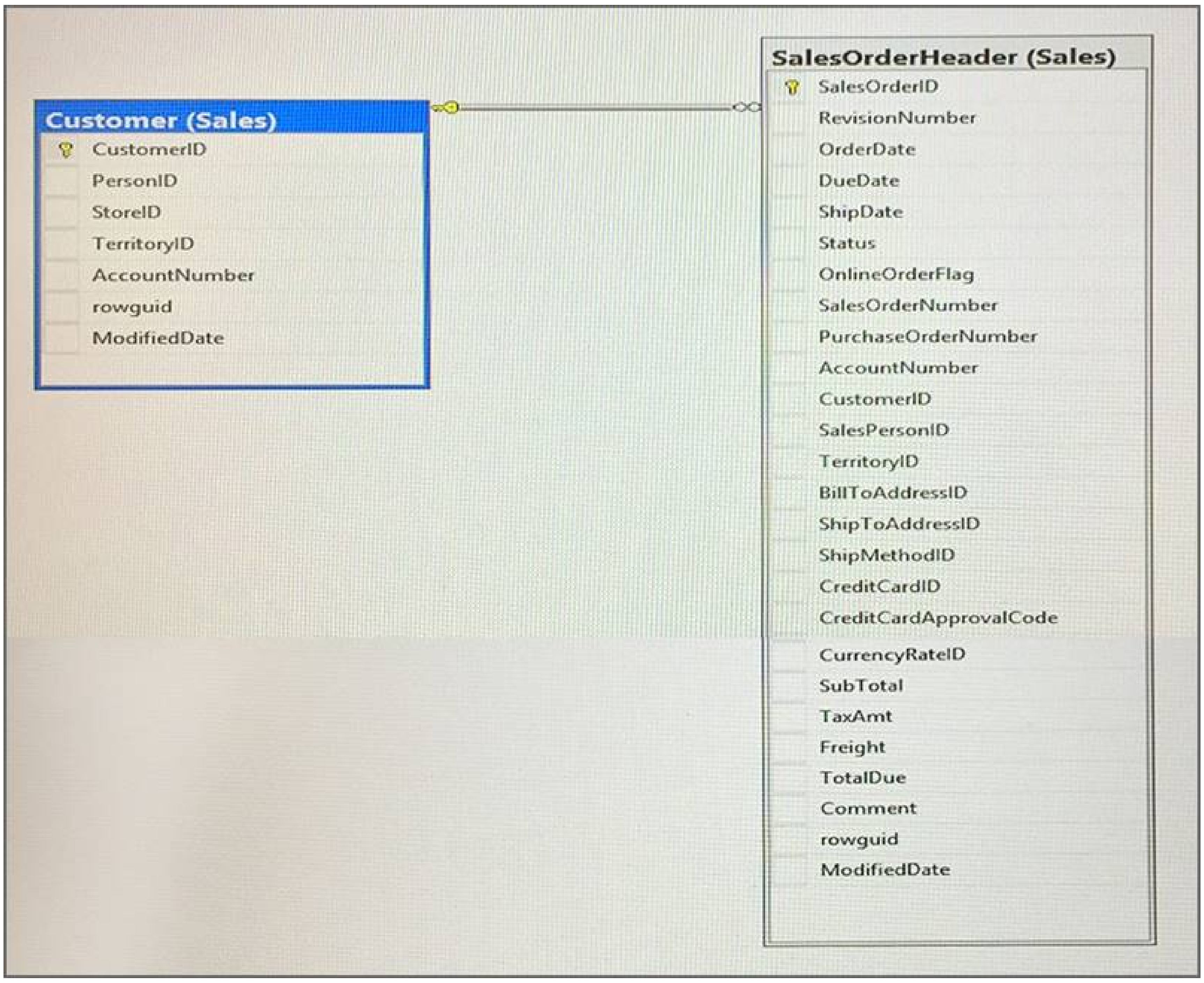
Does the solution meet the goal?

A. Yes
B. No

Answer: B

Explanation:
The NULL value in the UnitsOnOrder field would cause a runtime error.

NEW QUESTION 7
You have a database that includes the tables shown in the exhibit. (Click the exhibit button.)



You need to create a list of all customers, the order ID for the last order that the customer placed, and the date that the order was placed. 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?

A

```
SELECT C.CustomerID, ISNULL(SOH.SalesOrderID, 0) AS OrderID,ISNULL(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
```

B

```
SELECT C.CustomerID, SOH.SalesOrderID, MAX(OrderDate)
FROM Sales.Customer C INNER JOIN Sales.SalesOrderHeader SOH
ON C.CustomerID = SOH.CustomerID
GROUP BY C.CustomerID, SOH.SalesOrderID
ORDER BY C.CustomerID
```

C

```
SELECT C.CustomerID, SOH.SalesOrderID, MAX(OrderDate)
FROM Sales.Customer C CROSS JOIN Sales.SalesOrderHeader SOH
ON C.CustomerID = SOH.CustomerID
GROUP BY C.CustomerID, SOH.SalesOrderID
ORDER BY C.CustomerID
```

D

```
SELECT C.CustomerID, SOH.SalesOrderID, MAX(OrderDate)
FROM Sales.Customer C RIGHT OUTER JOIN Sales.SalesOrderHeader SOH
ON C.CustomerID = SOH.CustomerID
GROUP BY C.CustomerID, SOH.SalesOrderID
ORDER BY C.CustomerID
```

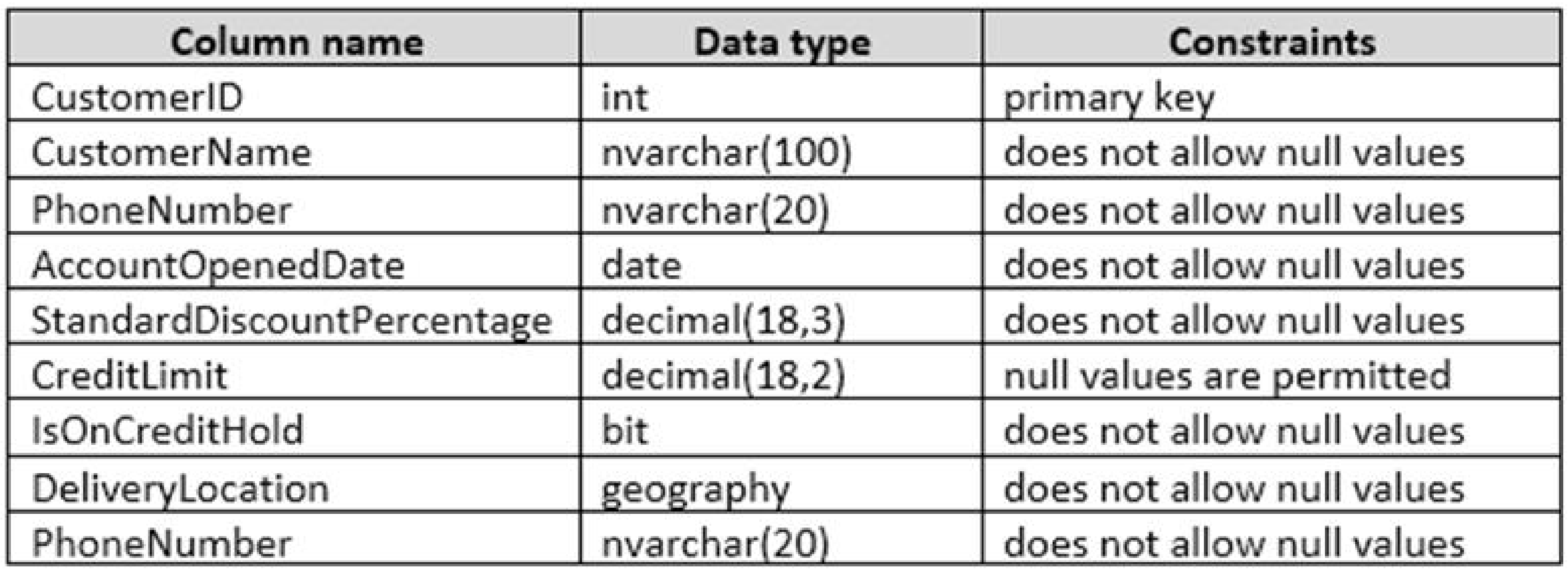
A. Option A

B. Option B
C. Option C
D. Option D

Answer: A

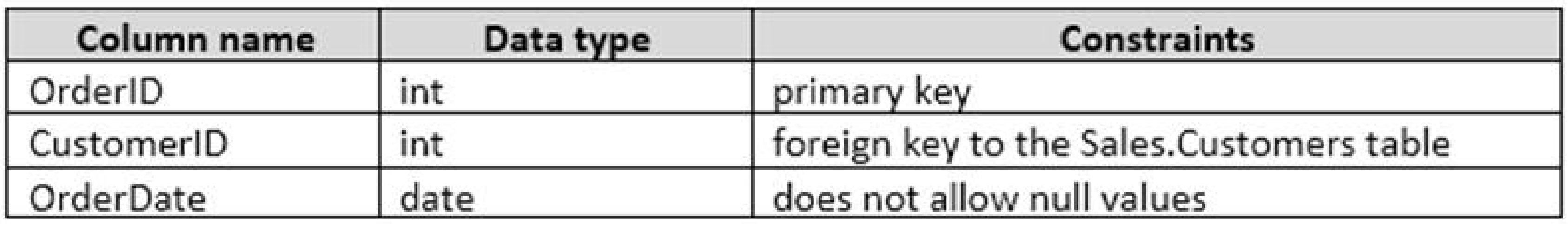
Explanation:
ISNULL Syntax: ISNULL (check_expression , replacement_value) author:"Luxemburg, Rosa"
The ISNULL function replaces NULL with the specified replacement value. The value of check_expression is returned if it is not NULL; otherwise, replacement_value is returned after it is implicitly converted to the type of check_expression.
References: https://msdn.microsoft.com/en-us/library/ms184325.aspx

NEW QUESTION 8
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.



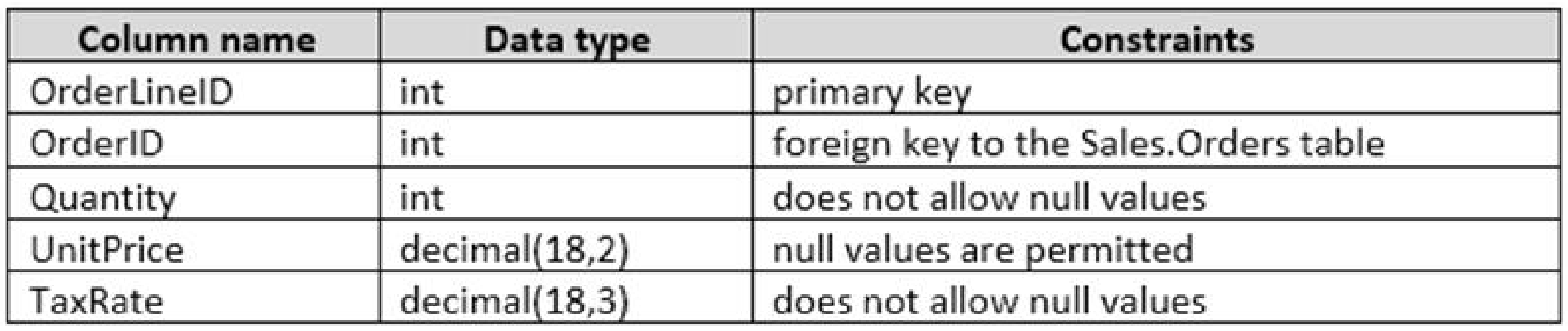
Column name	Data type	Constraints
CustomerID	int	primary key
CustomerName	nvarchar(100)	does not allow null values
PhoneNumber	nvarchar(20)	does not allow null values
AccountOpenedDate	date	does not allow null values
StandardDiscountPercentage	decimal(18,3)	does not allow null values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	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.



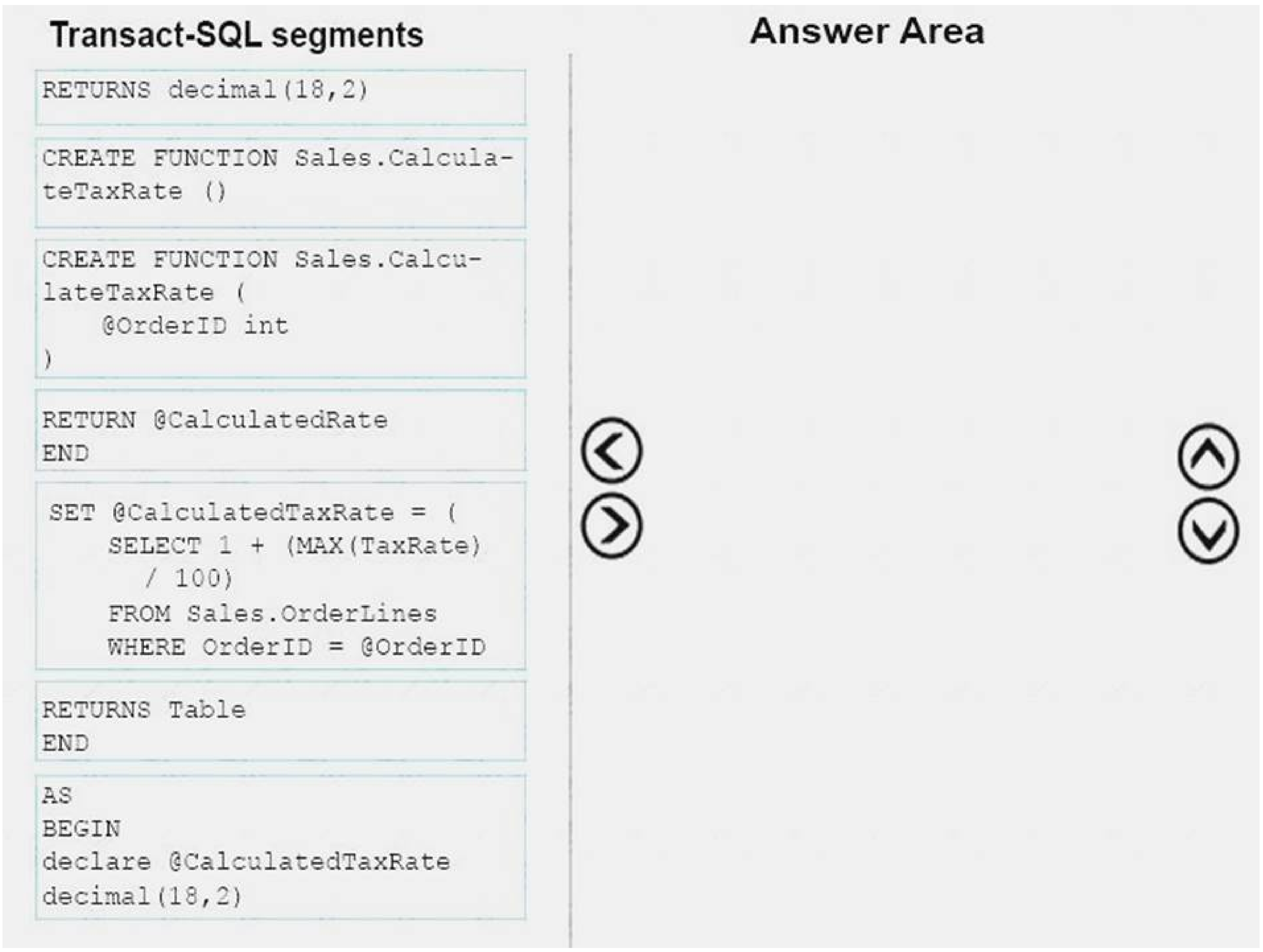
Column name	Data type	Constraints
OrderID	int	primary key
CustomerID	int	foreign key to the Sales.Customers table
OrderDate	date	does not allow null values

The following table describes the columns in Sales.OrderLines.



Column name	Data type	Constraints
OrderLineID	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 calculates the highest tax rate charged for an item in a specific order. Which five 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.



```
RETURNS decimal(18,2)
```

```
CREATE FUNCTION Sales.Calcula-
teTaxRate ()
```

```
CREATE FUNCTION Sales.Calcu-
lateTaxRate (
    @OrderID int
)
```

```
RETURN @CalculatedRate
END
```

```
 SET @CalculatedTaxRate = (
     SELECT 1 + (MAX(TaxRate)
       / 100)
     FROM Sales.OrderLines
     WHERE OrderID = @OrderID
```

```
RETURNS Table
END
```

```
AS
BEGIN
declare @CalculatedTaxRate
decimal(18,2)
```

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Box 1: CREATE FUNCTION…@OrderID
Include definition for the …@OrderID parameter. Box 2: RETURNS decimal(18,2)
The function is defined to return a scalar value. Box 3: AS BEGIN …
Declare the local variables of the function. Box 4: SET @CalculatedTaxRate = (.. Calculate the tax rate.
Box 5: RETURN @CalculatedRate END Return a scalar value.
References: https://msdn.microsoft.com/en-us/library/ms186755.aspx

NEW QUESTION 9
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You are creating indexes in a data warehouse.
You have a dimension table named Table1 that has 10,000 rows. The rows are used to generate several reports. The reports join a column that is the primary key.
The execution plan contains bookmark lookups for Table1. You discover that the reports run slower than expected.
You need to reduce the amount of time it takes to run the reports.
Solution: You create a nonclustered index on the primary key column that includes the bookmark lookup columns.
Does this meet the goal?

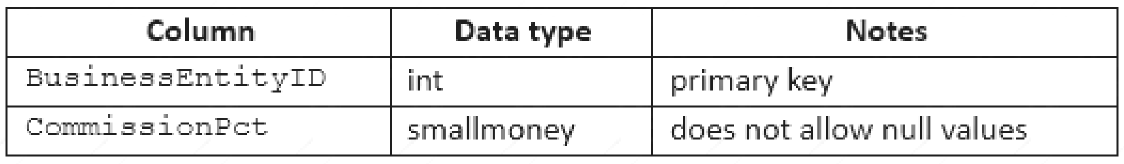
A. Yes
B. No

Answer: B

NEW QUESTION 10
You have a database that includes the following tables. HumanResources.Employee

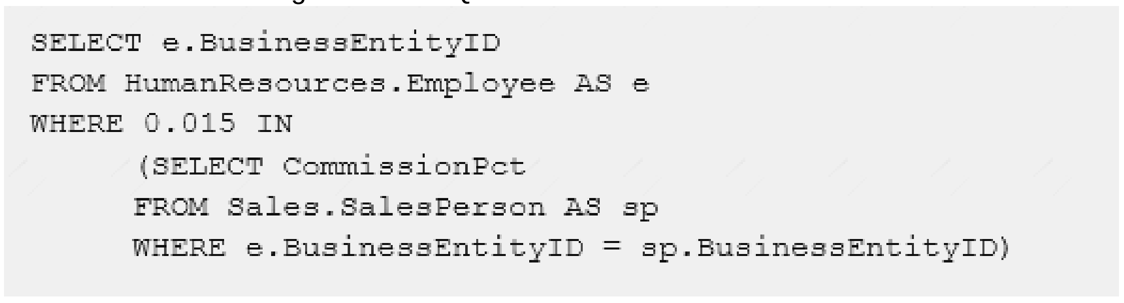
Column	Data type	Notes
BusinessEntityID	int	primary key

Sales.SalesPerson



Column	Data type	Notes
BusinessEntityID	int	primary key
CommissionPct	smallmoney	does not allow null values

The HumanResources.Employee table has 2,500 rows, and the Sales.SalesPerson table has 2,000 rows.
You review the following Transact-SQL statement:



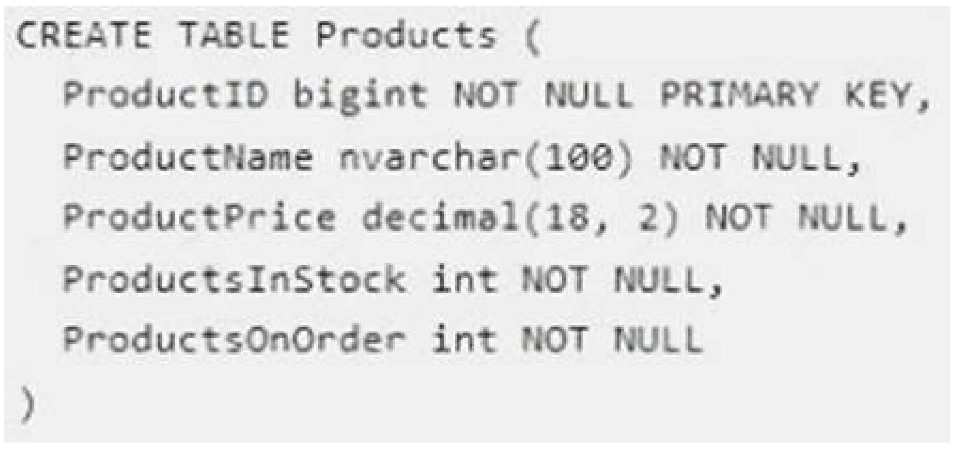
```
SELECT e.BusinessEntityID
FROM HumanResources.Employee AS e
WHERE 0.015 IN
      (SELECT CommissionPct
      FROM Sales.SalesPerson AS sp
      WHERE e.BusinessEntityID = sp.BusinessEntityID)
```

You need to determine the performance impact of the query.
How many times will a lookup occur on the primary key index on the Sales.SalesPerson table?

A. 200
B. 2,000
C. 2,500
D. 5,500

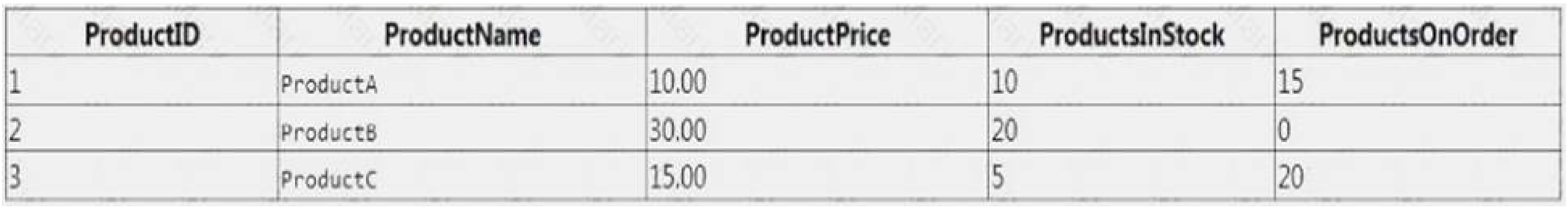
Answer: C

NEW QUESTION 10
You have a database that contains a table named Products in the sales schema. The table was created by running the following Transact-SQL statement:



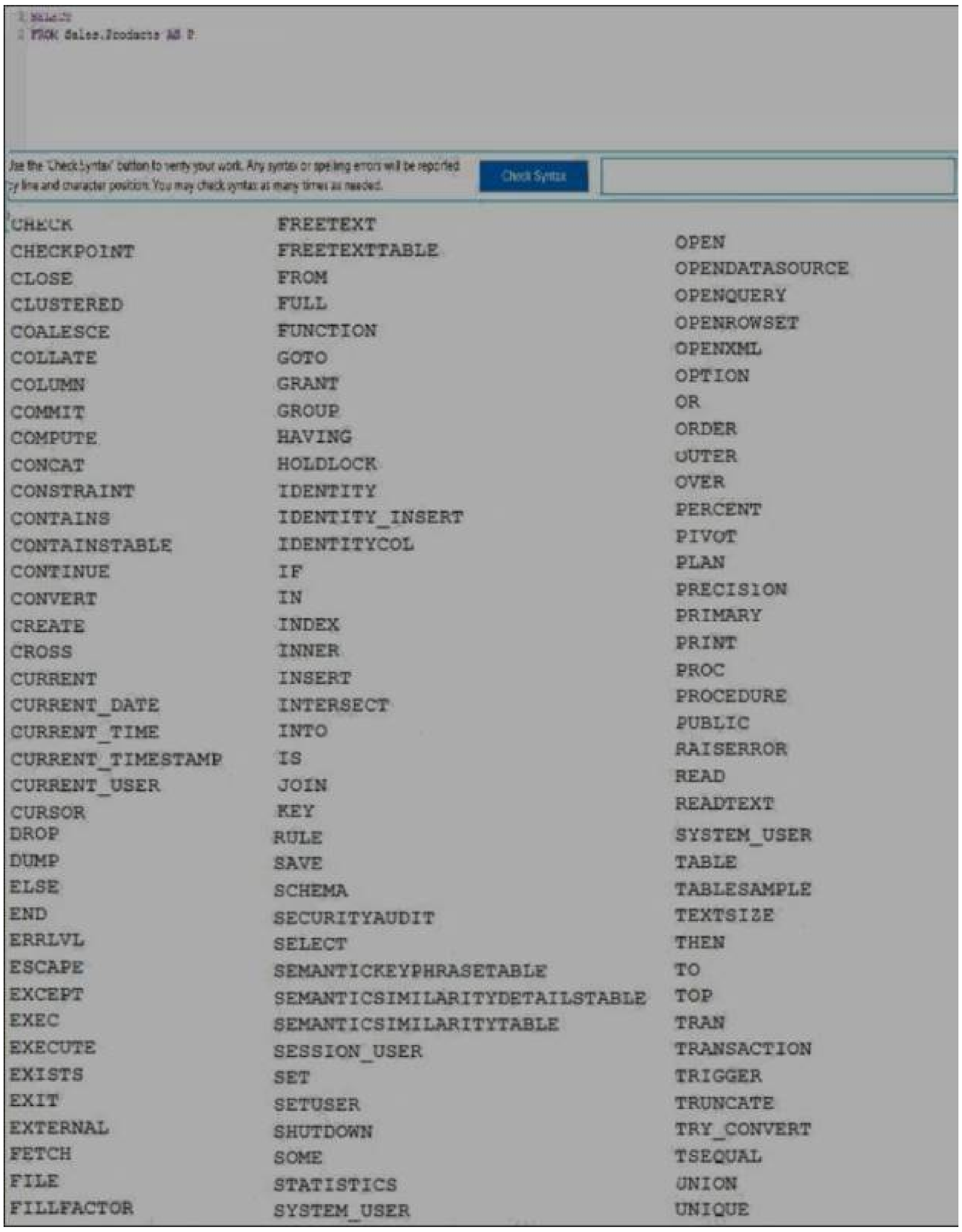
```
CREATE TABLE Products (
  ProductID bigint NOT NULL PRIMARY KEY,
  ProductName nvarchar(100) NOT NULL,
  ProductPrice decimal(18, 2) NOT NULL,
  ProductsInStock int NOT NULL,
  ProductsOnOrder int NOT NULL
)
```

The table includes the data shown below:



ProductID	ProductName	ProductPrice	ProductsInStock	ProductsOnOrder
1	ProductA	10.00	10	15
2	ProductB	30.00	20	0
3	ProductC	15.00	5	20

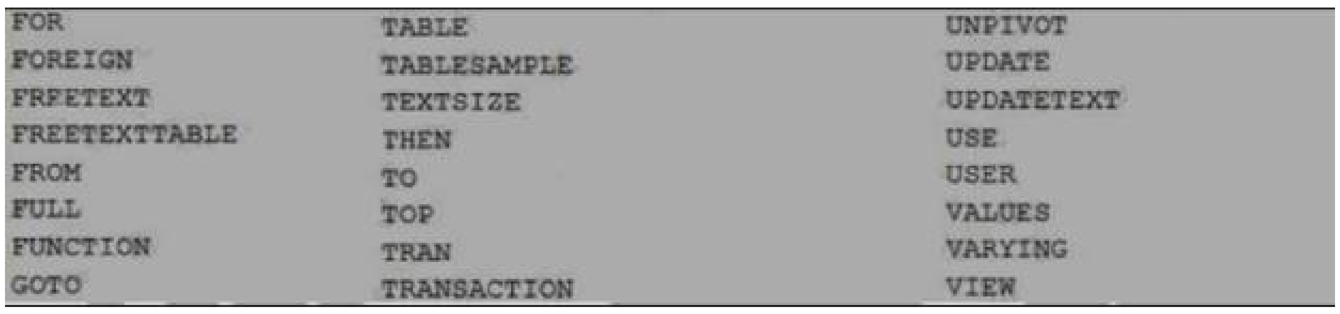
You are developing a report that displays the following values and column headers in the order listed below:
• average price of a product named Average
• the smallest number of products in stocK-named LowestNumber
• the highest product price named HighestPrice
You need to write a query to return the results for the report The query must meet the following requirements: You need to write a query to return the results for the report. The query must meet the following requirements:
• Use built-in, aggregate anfa*mathematical functions.
• Use two-part names and tables.
• Use the table alias to qualify column names.
• Define the alias for all fields by using the AS keyword.
• Use the first letter of the table name as the table alias.
• Do not use the Row_number function.
• Do not surround object names with square brackets.
• Do not use variables.
Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the code that has been provided as well as below it.



```
SELECT
FROM Sales.Products AS P
```

Jse the 'Check Syntax' button to verify your work. Any syntax or spelling errors will be reported
y line and character position. You may check syntax as many times as needed.

CHECK	FREETEXT	
CHECKPOINT	FREETEXTTABLE	OPEN
CLOSE	FROM	OPENDATASOURCE
CLUSTERED	FULL	OPENQUERY
COALESCE	FUNCTION	OPENROWSET
COLLATE	GOTO	OPENXML
COLUMN	GRANT	OPTION
COMMIT	GROUP	OR
COMPUTE	HAVING	ORDER
CONCAT	HOLDLOCK	OUTER
CONSTRAINT	IDENTITY	OVER
CONTAINS	IDENTITY_INSERT	PERCENT
CONTAINSTABLE	IDENTITYCOL	PIVOT
CONTINUE	IF	PLAN
CONVERT	IN	PRECISION
CREATE	INDEX	PRIMARY
CROSS	INNER	PRINT
CURRENT	INSERT	PROC
CURRENT_DATE	INTERSECT	PROCEDURE
CURRENT_TIME	INTO	PUBLIC
CURRENT_TIMESTAMP	IS	RAISERROR
CURRENT_USER	JOIN	READ
CURSOR	KEY	READTEXT
DROP	RULE	SYSTEM_USER
DUMP	SAVE	TABLE
ELSE	SCHEMA	TABLESAMPLE
END	SECURITYAUDIT	TEXTSIZE
ERRLVL	SELECT	THEN
ESCAPE	SEMANTICKEYPHRASETABLE	TO
EXCEPT	SEMANTICSIMILARITYDETAILSTABLE	TOP
EXEC	SEMANTICSIMILARITYTABLE	TRAN
EXECUTE	SESSION_USER	TRANSACTION
EXISTS	SET	TRIGGER
EXIT	SETUSER	TRUNCATE
EXTERNAL	SHUTDOWN	TRY_CONVERT
FETCH	SOME	TSEQUAL
FILE	STATISTICS	UNION
FILLFACTOR	SYSTEM_USER	UNIQUE



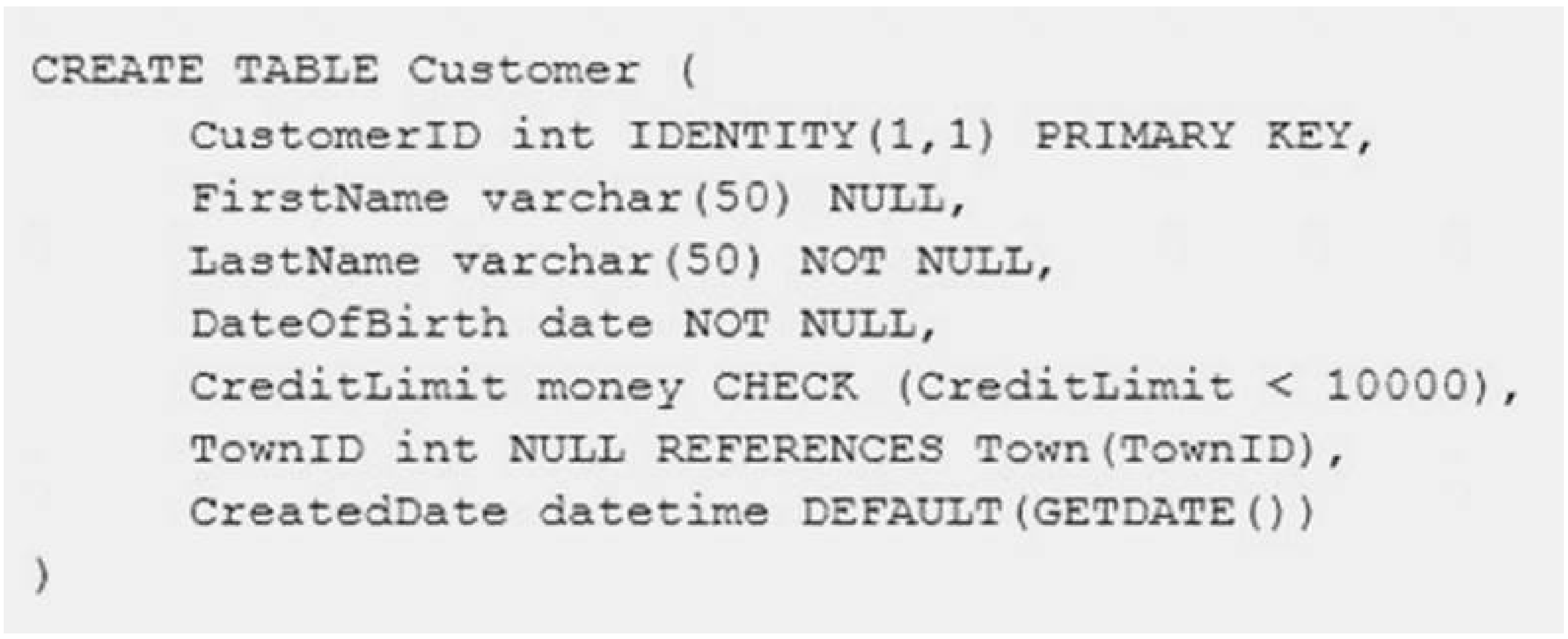
FOR	TABLE	UNPIVOT
FOREIGN	TABLESAMPLE	UPDATE
FREETEXT	TEXTSIZE	UPDATETEXT
FREETEXTTABLE	THEN	USE
FROM	TO	USER
FULL	TOP	VALUES
FUNCTION	TRAN	VARYING
GOTO	TRANSACTION	VIEW

A. Mastered
B. Not Mastered

Answer: A

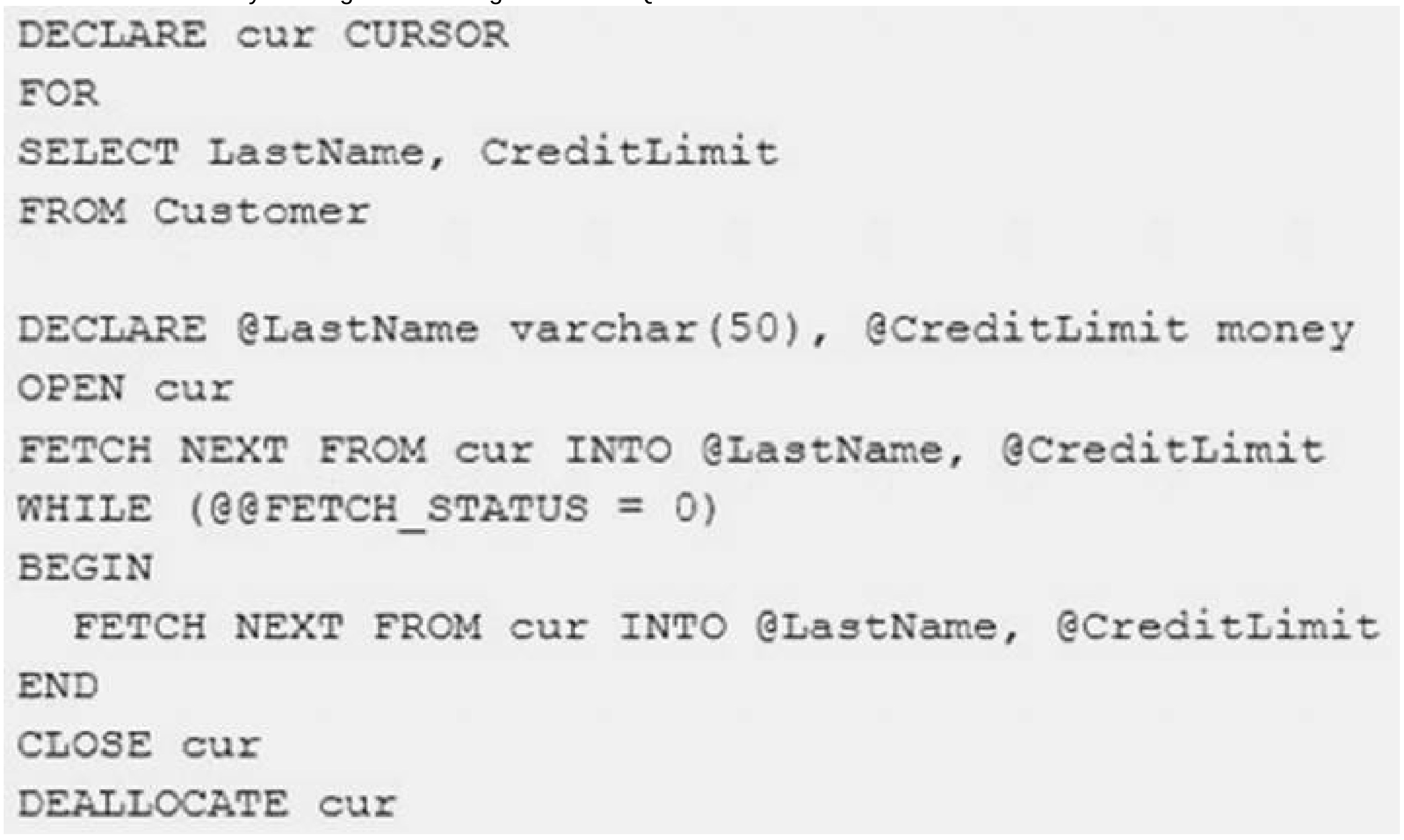
Explanation:
TRY_Convert

NEW QUESTION 15
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You create a table named Customer by running the following Transact-SQL statement:



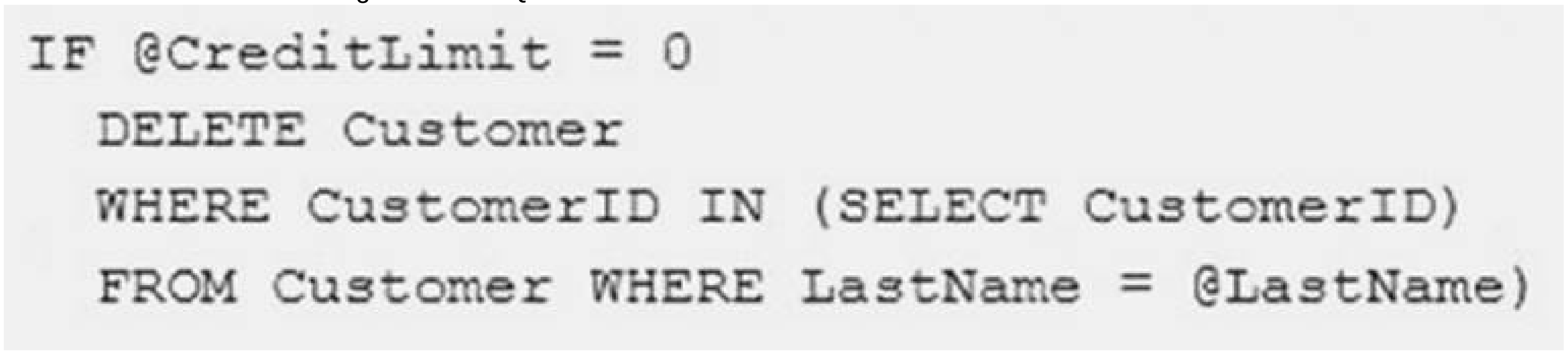
```
CREATE TABLE Customer (
      CustomerID int IDENTITY(1,1) PRIMARY KEY,
      FirstName varchar(50) NULL,
      LastName varchar(50) NOT NULL,
      DateOfBirth date NOT NULL,
      CreditLimit money CHECK (CreditLimit < 10000),
      TownID int NULL REFERENCES Town(TownID),
      CreatedDate datetime DEFAULT(GETDATE())
)
```

You create a cursor by running the following Transact-SQL statement:



```
DECLARE cur CURSOR
FOR
SELECT LastName, CreditLimit
FROM Customer

DECLARE @LastName varchar(50), @CreditLimit money
OPEN cur
FETCH NEXT FROM cur INTO @LastName, @CreditLimit
WHILE (@@FETCH_STATUS = 0)
BEGIN
  FETCH NEXT FROM cur INTO @LastName, @CreditLimit
END
CLOSE cur
DEALLOCATE cur
```

If the credit limit is zero, you must delete the customer record while fetching data. You need to add the DELETE statement.
Solution: You add the following Transact-SQL statement:



```
IF @CreditLimit = 0
  DELETE Customer
  WHERE CustomerID IN (SELECT CustomerID)
  FROM Customer WHERE LastName = @LastName)
```

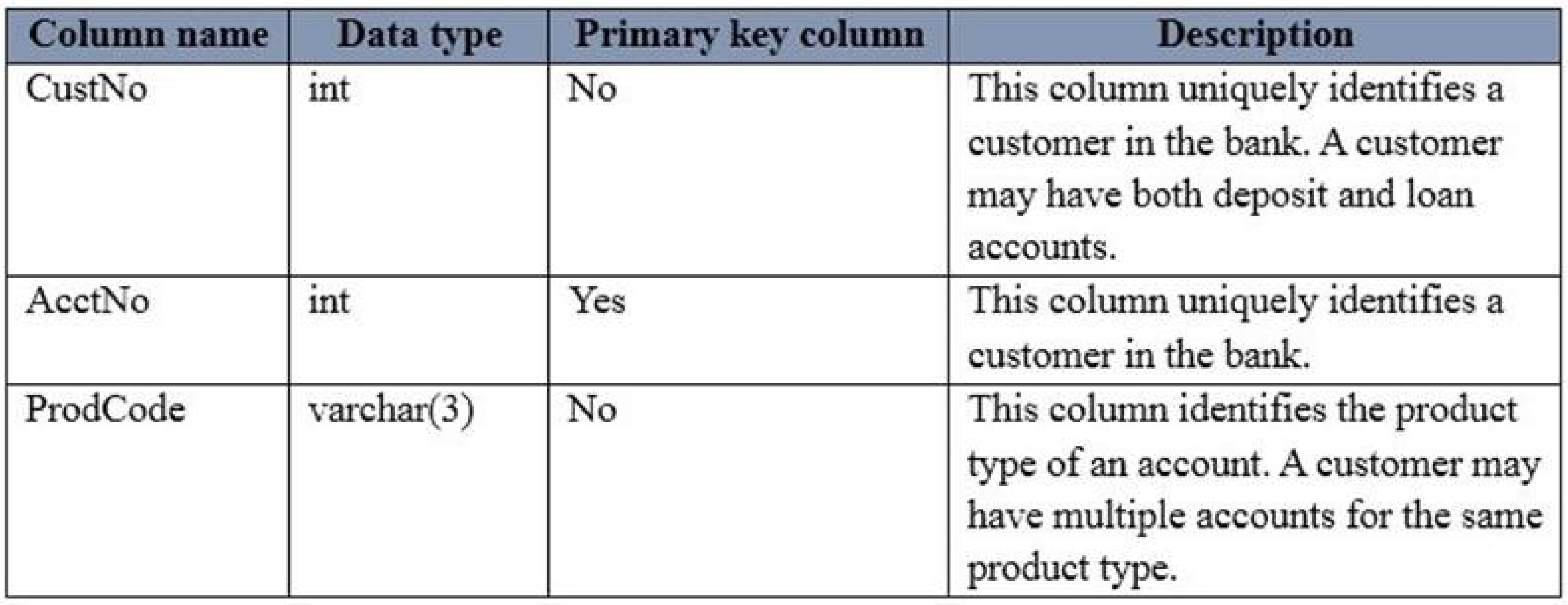
Does the solution meet the goal?

A. YES
B. NO

Answer: B

Explanation:
References: https://docs.microsoft.com/en-us/sql/t-sql/statements/delete-transact-sql?view=sql-server-2017

NEW QUESTION 16
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply to that question.
You have a database for a banking system. The database has two tables named tblDepositAcct and tblLoanAcct that store deposit and loan accounts, respectively. Both tables contain the following columns:



Column name	Data type	Primary key column	Description
CustNo	int	No	This column uniquely identifies a customer in the bank. A customer may have both deposit and loan accounts.
AcctNo	int	Yes	This column uniquely identifies a customer in the bank.
ProdCode	varchar(3)	No	This column identifies the product type of an account. A customer may have multiple accounts for the same product type.

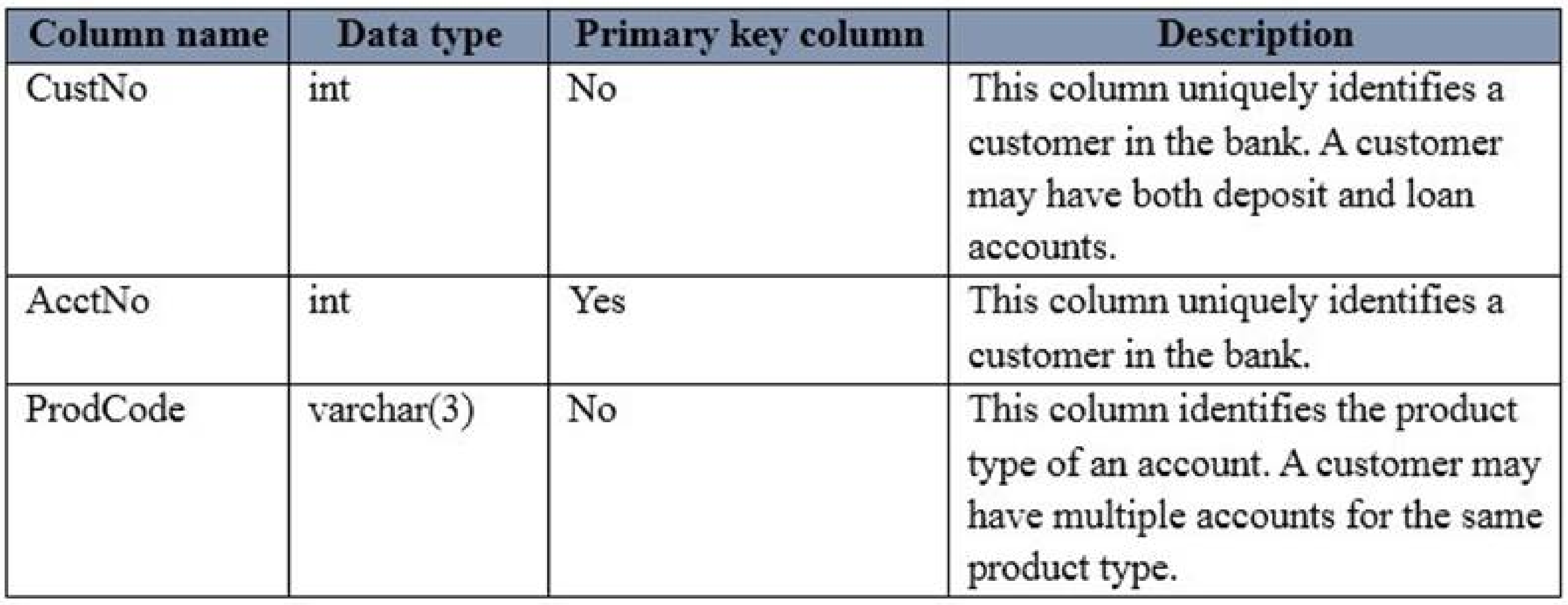
You need to determine the total number of customers who have only loan accounts. Which Transact-SQL statement should you run?

A. SELECT COUNT(*)FROM (SELECT AcctNoFROM tblDepositAcctINTERSECTSELECTAcctNoFROM tblLoanAcct) R
B. SELECT COUNT(*)FROM (SELECT CustNoFROM tblDepositAcctUNIONSELECT CustNoFROMtblLoanAcct) R
C. SELECT COUNT(*)FROM (SELECT CustNoFROM tblDepositAcctUNION ALLSELECTCustNoFROM tblLoanAcct) R
D. SELECT COUNT (DISTINCT D.CustNo)FROM tblDepositAcct D, tblLoanAcct LWHERE D.CustNo= L.CustNo
E. SELECT COUNT(DISTINCT L.CustNo)FROM tblDepositAcct DRIGHT JOIN tblLoanAcct L ON D.CustNo = L.CustNoWHERE D.CustNo IS NULL
F. SELECT COUNT(*)FROM (SELECT CustNoFROM tblDepositAcctEXCEPTSELECT CustNoFROMtblLoanAcct) R
G. SELECT COUNT (DISTINCT COALESCE(D.CustNo, L.CustNo))FROM tblDepositAcct DFULLJOIN tblLoanAcct L ON D.CustNo = L.CustNoWHERE D.CustNo IS NULL OR L.CustNo IS NULL
H. SELECT COUNT(*)FROM tblDepositAcct DFULL JOIN tblLoanAcct L ON D.CustNo = L.CustNo

Answer: E

Explanation:
The RIGHT JOIN keyword returns all records from the right table (table2), and the matched records from the left table (table1). The result is NULL from the left side, when there is no match.
References: https://www.w3schools.com/sql/sql_join_right.asp

NEW QUESTION 18
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply to that question.
You have a database for a banking system. The database has two tables named tblDepositAcct and tblLoanAcct that store deposit and loan accounts, respectively. Both tables contain the following columns:



Column name	Data type	Primary key column	Description
CustNo	int	No	This column uniquely identifies a customer in the bank. A customer may have both deposit and loan accounts.
AcctNo	int	Yes	This column uniquely identifies a customer in the bank.
ProdCode	varchar(3)	No	This column identifies the product type of an account. A customer may have multiple accounts for the same product type.

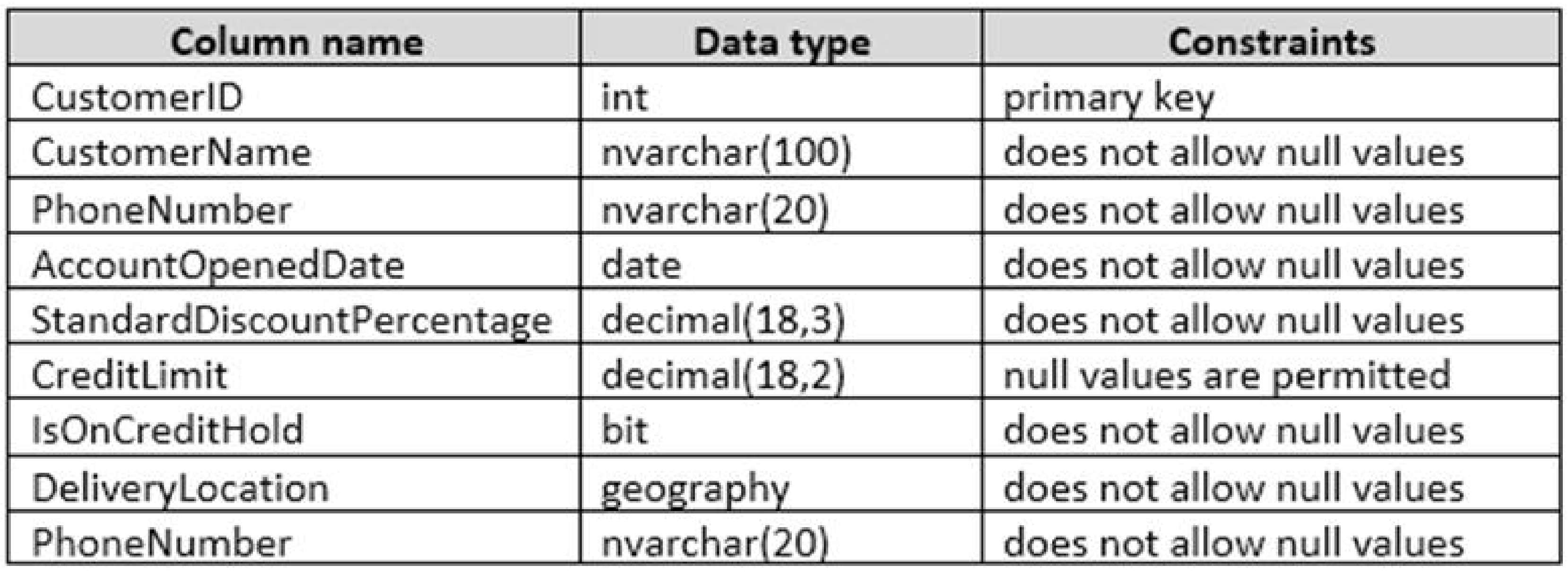
You need to determine the total number of deposit and loan accounts.
Which Transact-SQL statement should you run?

A. SELECT COUNT(*)FROM (SELECT AcctNoFROM tblDepositAcctINTERSECTSELECTAcctNoFROM tblLoanAcct) R
B. SELECT COUNT(*)FROM (SELECT CustNoFROM tblDepositAcctUNIONSELECT CustNoFROMtblLoanAcct) R
C. SELECT COUNT(*)FROM (SELECT CustNoFROMtblDepositAcctUNION ALLSELECTCustNoFROM tblLoanAcct) R
D. SELECT COUNT (DISTINCT D.CustNo)FROM tblDepositAcct D, tblLoanAcct LWHERE D.CustNo= L.CustNo
E. SELECT COUNT(DISTINCT L.CustNo)FROM tblDepositAcct DRIGHT JOIN tblLoanAcct L ON D.CustNo =L.CustNoWHERE D.CustNo IS NULL
F. SELECT COUNT(*)FROM (SELECT CustNoFROM tblDepositAcctEXCEPTSELECT CustNoFROMtblLoanAcct) R
G. SELECT COUNT (DISTINCT COALESCE(D.CustNo, L.CustNo))FROM tblDepositAcct DFULLJOIN tblLoanAcct L ON D.CustNo =L.CustNoWHERE D.CustNo IS NULL OR L.CustNo IS NULL
H. SELECT COUNT(*)FROM tblDepositAcct DFULL JOIN tblLoanAcct L ON D.CustNo = L.CustNo

Answer: C

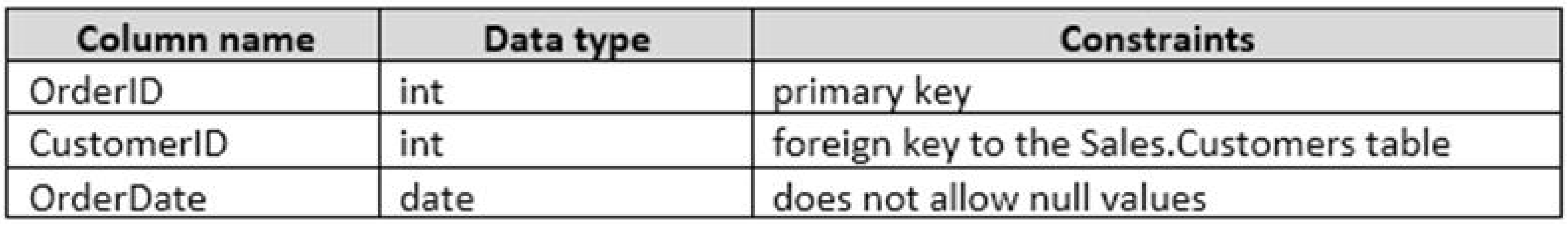
Explanation:
Would list the customers with duplicates, which would equal the number of accounts.

NEW QUESTION 19
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.



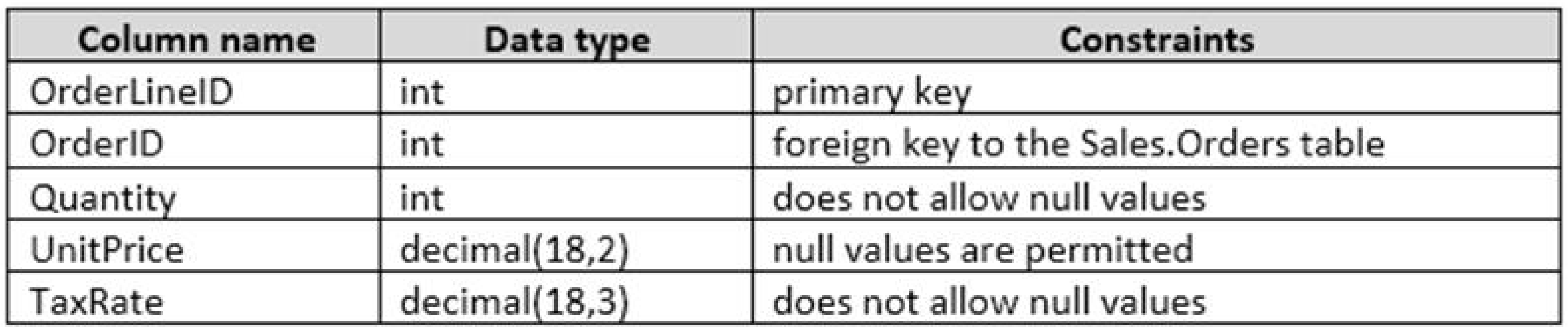
Column name	Data type	Constraints
CustomerID	int	primary key
CustomerName	nvarchar(100)	does not allow null values
PhoneNumber	nvarchar(20)	does not allow null values
AccountOpenedDate	date	does not allow null values
StandardDiscountPercentage	decimal(18,3)	does not allow null values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	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.



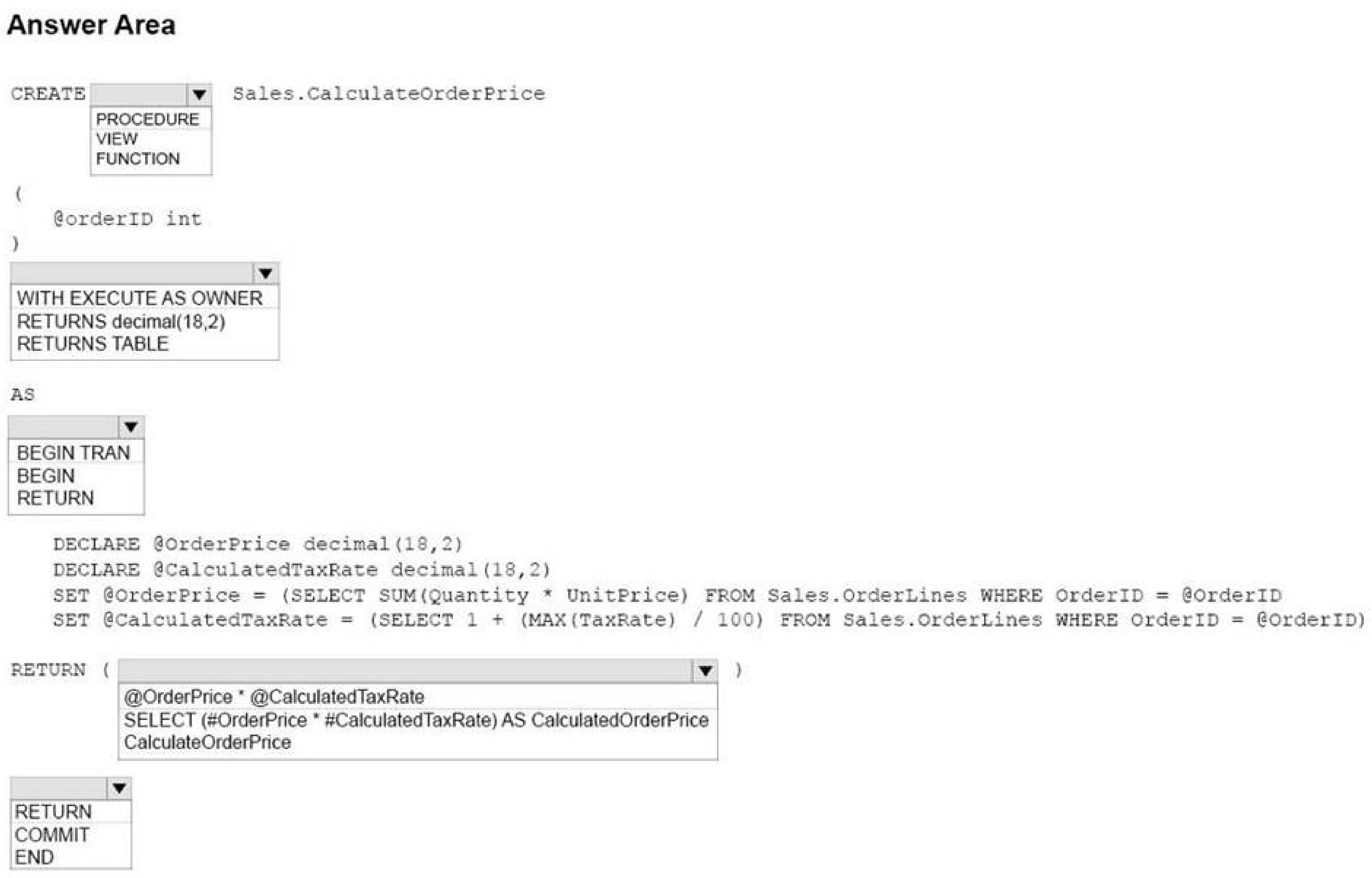
Column name	Data type	Constraints
OrderID	int	primary key
CustomerID	int	foreign key to the Sales.Customers table
OrderDate	date	does not allow null values

The following table describes the columns in Sales.OrderLines.



Column name	Data type	Constraints
OrderLineID	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 database object that calculates the total price of an order including the sales tax. The database object must meet the following requirements:
- Reduce the compilation cost of Transact-SQL code by caching the plans and reusing them for repeated execution.
- Return a value.
- Be callable from a SELECT statement.
How should you complete the Transact-SQL statements? To answer, select the appropriate Transact-SQL segments in the answer area.


Answer Area

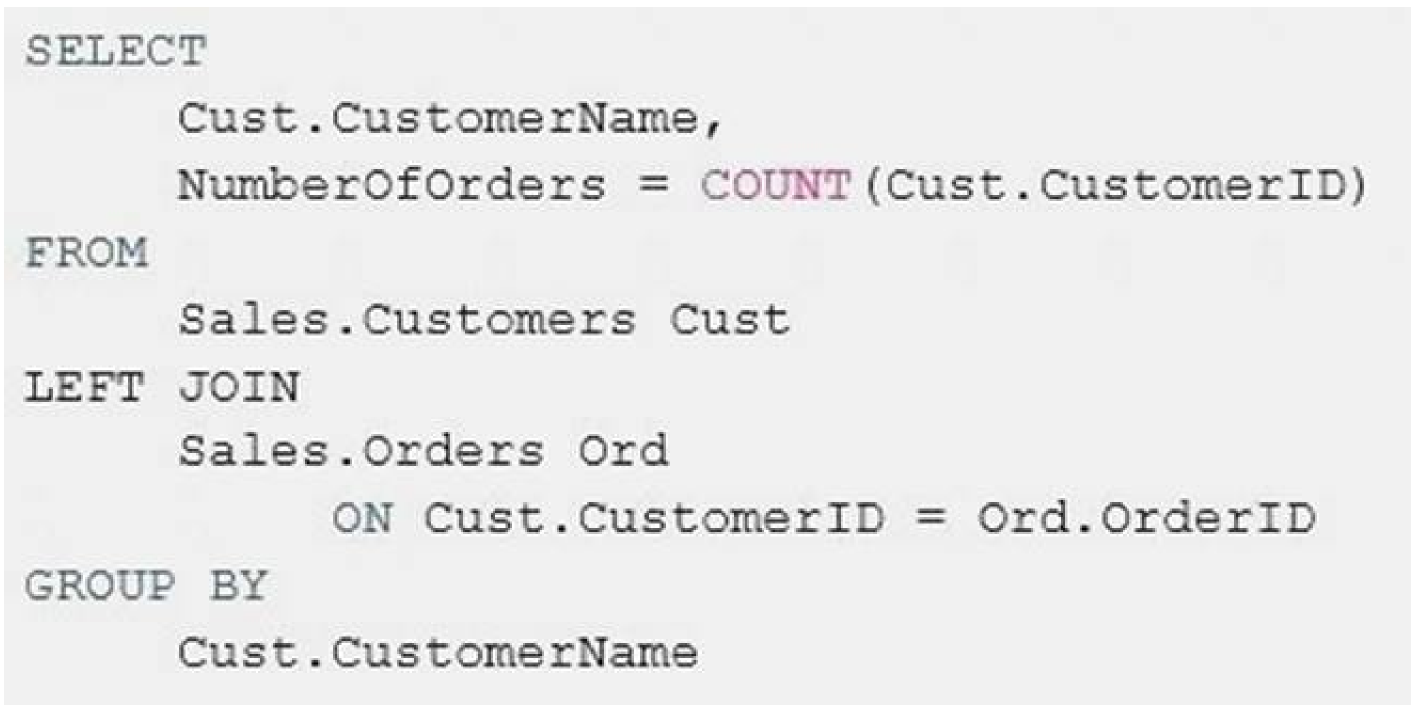
```
CREATE [PROCEDURE | VIEW | FUNCTION] Sales.CalculateOrderPrice
(
    @orderID int
)
[WITH EXECUTE AS OWNER | RETURNS decimal(18,2) | RETURNS TABLE]

AS
[BEGIN TRAN | BEGIN | RETURN]
    DECLARE @OrderPrice decimal(18,2)
    DECLARE @CalculatedTaxRate decimal(18,2)
    SET @OrderPrice = (SELECT SUM(Quantity * UnitPrice) FROM Sales.OrderLines WHERE OrderID = @OrderID
    SET @CalculatedTaxRate = (SELECT 1 + (MAX(TaxRate) / 100) FROM Sales.OrderLines WHERE OrderID = @OrderID)

RETURN ( [@OrderPrice * @CalculatedTaxRate | SELECT (#OrderPrice * #CalculatedTaxRate) AS CalculatedOrderPrice | CalculateOrderPrice] )

[RETURN | COMMIT | END]
```

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Box 1: FUNCTION
To be able to return a value we should use a scalar function.
CREATE FUNCTION creates a user-defined function in SQL Server and Azure SQL Database. The return value can either be a scalar (single) value or a table.
Box 2: RETURNS decimal(18,2)
Use the same data format as used in the UnitPrice column. Box 3: BEGIN
Transact-SQL Scalar Function Syntax include the BEGIN ..END construct.
CREATE [OR ALTER] FUNCTION [schema_name.] function_name
([{ @parameter_name [AS][type_schema_name.] parameter_data_type [= default] [READONLY] }
[,...n]
]
)
RETURNS return_data_type
[WITH <function_option> [,...n]] [AS]
BEGIN
function_body
RETURN scalar_expression END
[;]
Box 4: @OrderPrice * @CalculatedTaxRate Calculate the price including tax.
Box 5: END
Transact-SQL Scalar Function Syntax include the BEGIN ..END construct. References: https://msdn.microsoft.com/en-us/library/ms186755.aspx

NEW QUESTION 21
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a database named DB1 that contains two tables named Sales.Customers and Sales.Orders. Sales.Customers has a foreign key relationship to a column named CustomerID in Sales.Orders.
You need to recommend a query that returns all the customers. The query must also return the number of orders that each customer placed in 2016.
Solution: You recommend the following query:



```
SELECT
     Cust.CustomerName,
     NumberOfOrders = COUNT(Cust.CustomerID)
FROM
     Sales.Customers Cust
LEFT JOIN
     Sales.Orders Ord
          ON Cust.CustomerID = Ord.OrderID
GROUP BY
     Cust.CustomerName
```

Does this meet the goal?

A. Yes
B. No

Answer: A

NEW QUESTION 23
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You have a database that stores sales and order information.
Users must be able to extract information from the tables on an ad hoc basis. They must also be able to reference the extracted information as a single table.
You need to implement a solution that allows users to retrieve the data required, based on variables defined at the time of the query.
What should you implement?

A. the COALESCE function
B. a view
C. a table-valued function
D. the TRY_PARSE function
E. a stored procedure
F. the ISNULL function
G. a scalar function
H. the TRY_CONVERT function

Answer: C

Explanation:
User-defined functions that return a table data type can be powerful alternatives to views. These functions are referred to as table-valued functions. A table-valued user-defined function can be used where table or view expressions are allowed in Transact-SQL queries. While views are limited to a single SELECT statement, user-defined functions can contain additional statements that allow more powerful logic than is possible in views.
A table-valued user-defined function can also replace stored procedures that return a single result set. References: https://technet.microsoft.com/en-us/library/ms191165(v=sql.105).aspx

NEW QUESTION 25
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 on this series.
You have a database that tracks orders and deliveries for customers in North America. System versioning is enabled for all tables. The database contains the Sales.Customers, Application.Cities, and Sales.CustomerCategories tables.
Details for the Sales.Customers table are shown in the following table:

Column	Data type	Notes
CustomerId	int	primary key
CustomerCategoryId	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow values
StandardDiscountPercentage	int	does not allow values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow values
DeliveryLocation	geography	does not allow values
PhoneNumber	nvarchar(20)	does not allow values
ValidFrom	datetime2(7)	does not allow values, GENERATED ALWAYS AS ROW START
ValidTo	datetime2(7)	does not allow values, GENERATED ALWAYS AS ROW END

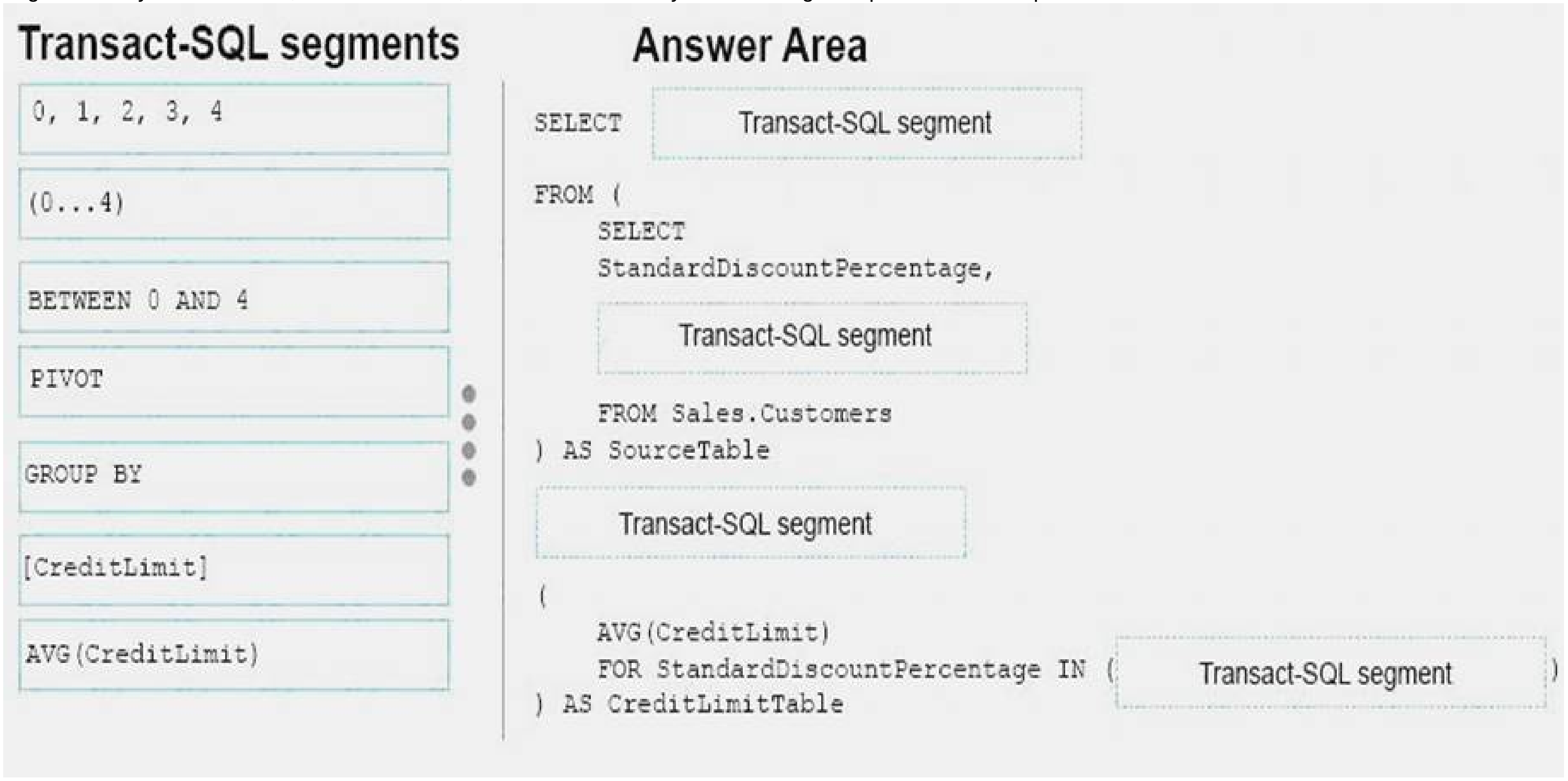
Details for the Application.Cities table are shown in the following table:

Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Details for the Sales.CustomerCategories table are shown in the following table:

Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

The marketing department is performing an analysis of how discount affect credit limits. They need to know the average credit limit per standard discount percentage for customers whose standard discount percentage is between zero and four.
You need to create a query that returns the data for the analysis.
How should you complete the Transact-SQL statement? To answer, drag the appropriate Transact-SQL segments to the correct locations. Each Transact-SQL segments 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.



Transact-SQL segments

- `0, 1, 2, 3, 4`
- `(0...4)`
- `BETWEEN 0 AND 4`
- `PIVOT`
- `GROUP BY`
- `[CreditLimit]`
- `AVG(CreditLimit)`

Answer Area

```
SELECT   Transact-SQL segment
FROM (
     SELECT
     StandardDiscountPercentage,
       Transact-SQL segment
     FROM Sales.Customers
) AS SourceTable
   Transact-SQL segment
(
     AVG(CreditLimit)
     FOR StandardDiscountPercentage IN (  Transact-SQL segment  )
) AS CreditLimitTable
```

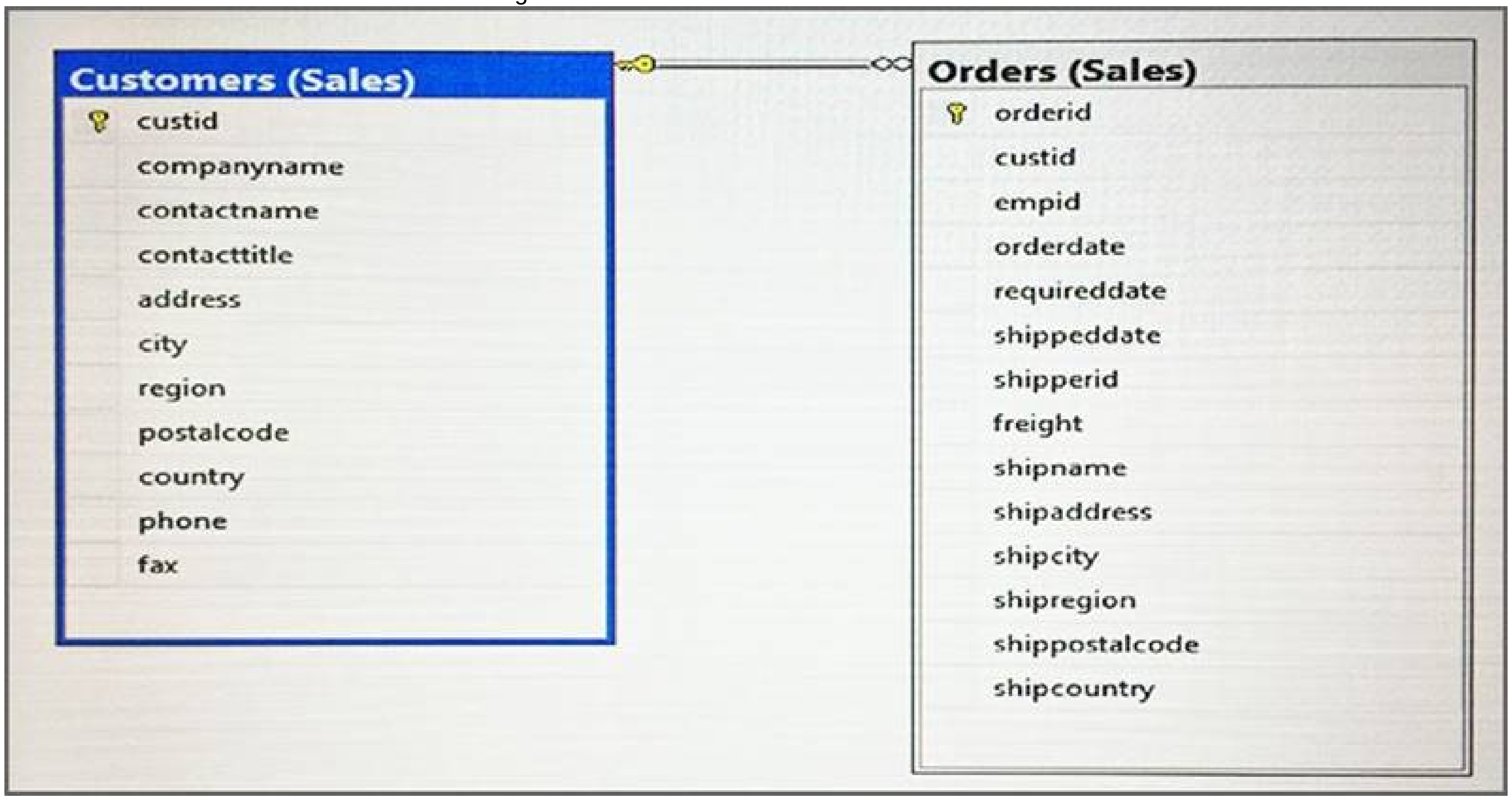
A. Mastered
B. Not Mastered

Answer: A

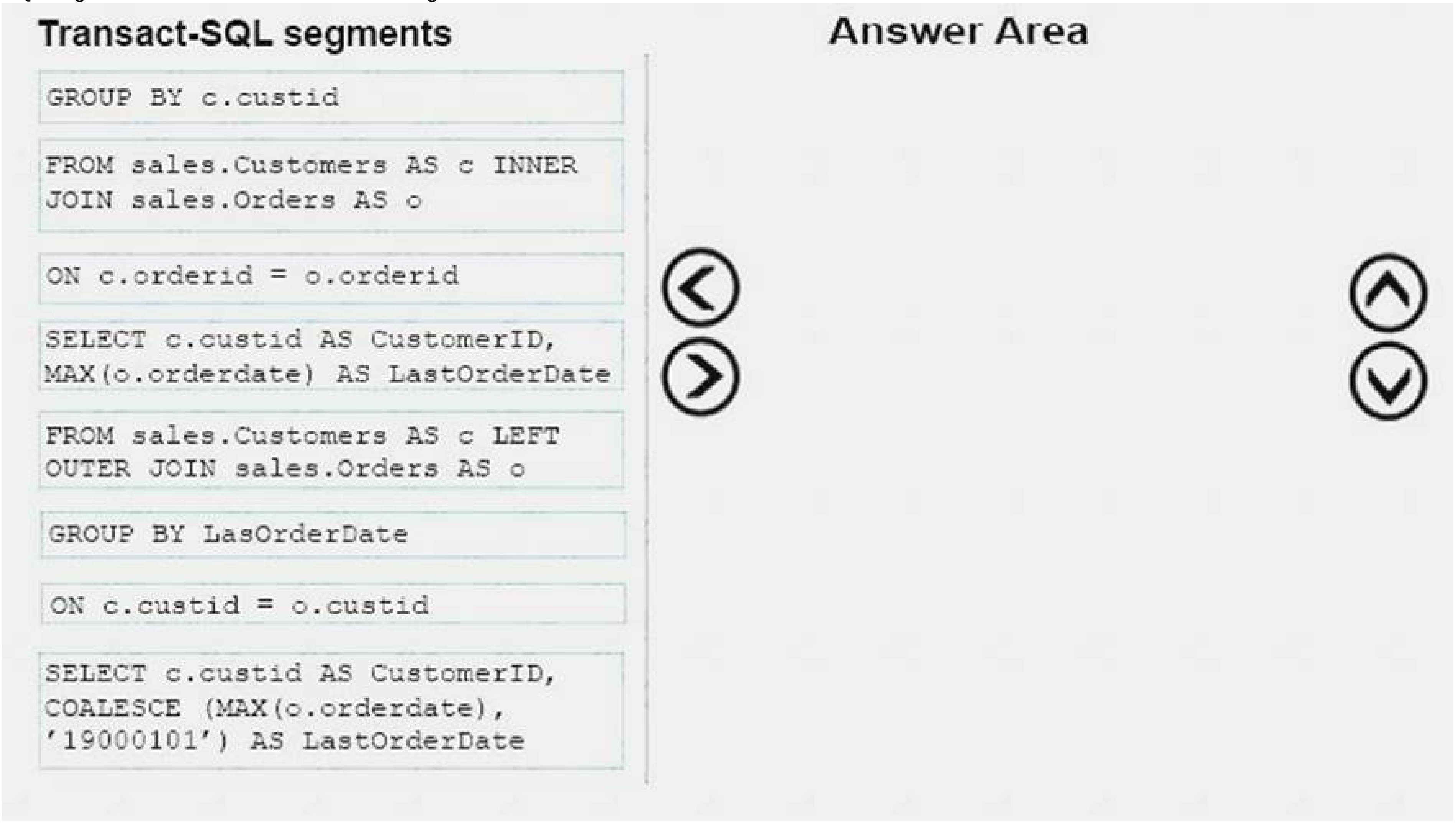
Explanation:
Box 1: 0, 1, 2, 3, 4
Pivot example:

-- Pivot table with one row and five columns
SELECT 'AverageCost' AS Cost_Sorted_By_Production_Days, [0], [1], [2], [3], [4]
FROM
(SELECT DaysToManufacture, StandardCost FROM Production.Product) AS SourceTable PIVOT
(
AVG(StandardCost)
FOR DaysToManufacture IN ([0], [1], [2], [3], [4])
) AS PivotTable; Box 2: [CreditLimit]
Box 3: PIVOT
You can use the PIVOT and UNPIVOT relational operators to change a table-valued expression into another table. PIVOT rotates a table-valued expression by turning the unique values from one column in the expression into multiple columns in the output, and performs aggregations where they are required on any remaining column values that are wanted in the final output.
Box 4: 0, 1, 2, 3, 4
The IN clause determines whether a specified value matches any value in a subquery or a list. Syntax: test_expression [NOT] IN (subquery | expression [,...n])
Where expression[,... n]
is a list of expressions to test for a match. All expressions must be of the same type as test_expression. References: https://technet.microsoft.com/en-us/library/ms177410(v=sql.105).aspx

NEW QUESTION 29
You have a database that includes the following tables:



You need to create a list of all customer IDs and the date of the last order that each customer placed. If the customer has not placed any orders, you must return the date January 1, 1900. The column names must be CustomerID and LastOrderDate.
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.



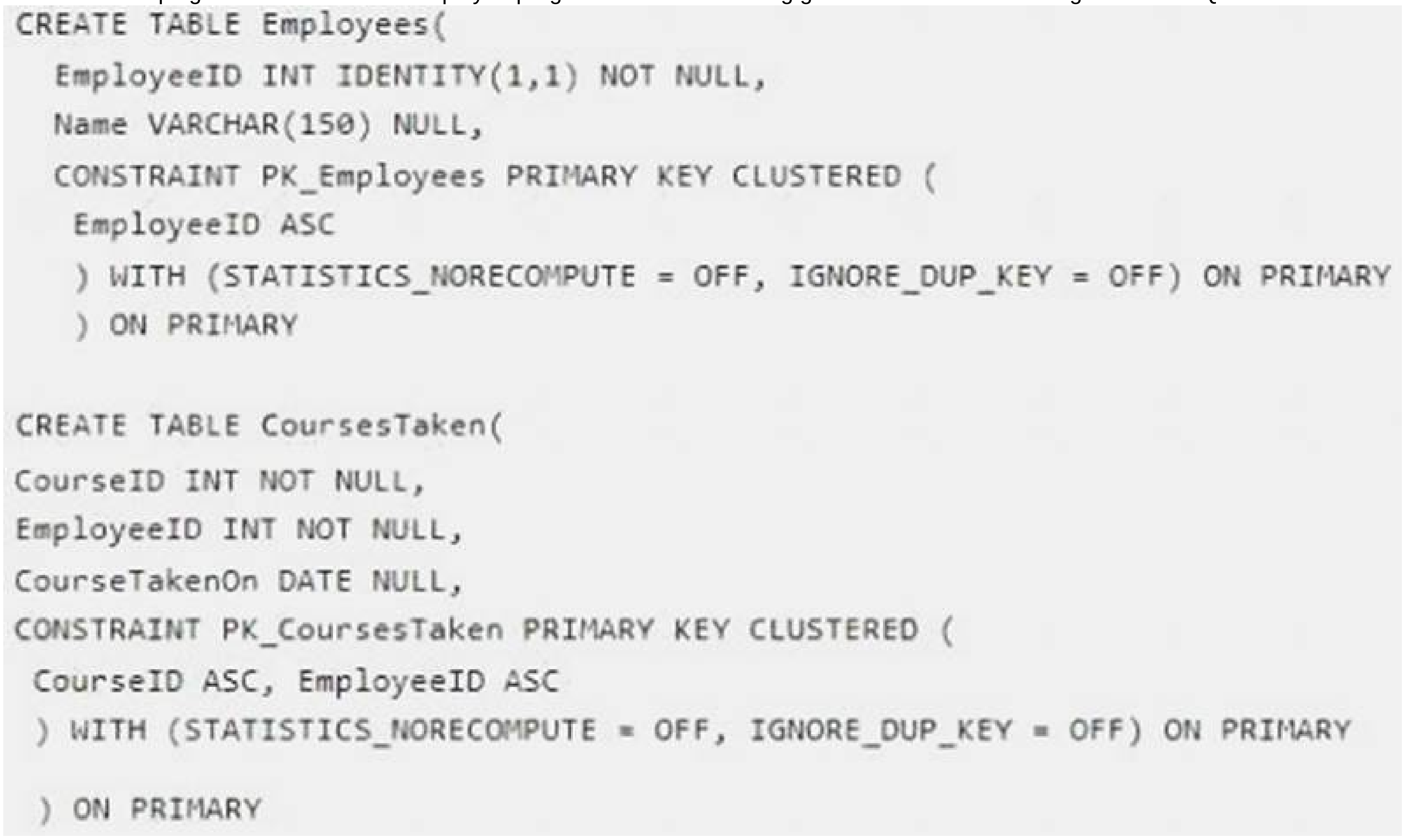
```
GROUP BY c.custid
```
```
FROM sales.Customers AS c INNER
JOIN sales.Orders AS o
```
```
ON c.orderid = o.orderid
```
```
SELECT c.custid AS CustomerID,
MAX(o.orderdate) AS LastOrderDate
```
```
FROM sales.Customers AS c LEFT
OUTER JOIN sales.Orders AS o
```
```
GROUP BY LasOrderDate
```
```
ON c.custid = o.custid
```
```
SELECT c.custid AS CustomerID,
COALESCE (MAX(o.orderdate),
'19000101') AS LastOrderDate
```

A. Mastered
B. Not Mastered

Answer: A

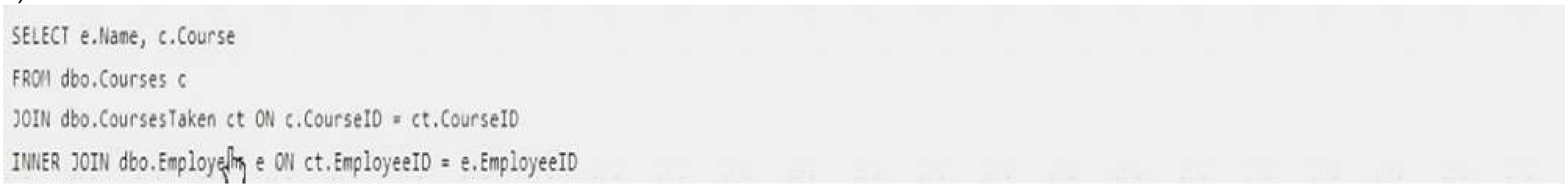
Explanation:
Box 1: SELECT..COALESCE…
The COALESCE function evaluates the arguments in order and returns the current value of the first expression that initially does not evaluate to NULL.
Box 2: ..LEFT OUTER JOIN..
The LEFT JOIN (LEFT OUTER JOIN) keyword returns all rows from the left table (table1), with the matching rows in the right table (table2). The result is NULL in the right side when there is no match. A customer might have no orders so the right table must be allowed have a NULL value.
Box 3: ON c.custid = o.custid
We JOIN on the custID column, which is available in both tables. Box 4: GROUP BY c.custid
References:
https://technet.microsoft.com/en-us/library/ms189499(v=sql.110).aspx
http://www.w3schools.com/sql/sql_join_left.asp

NEW QUESTION 30
You are developing a database to track employee progress relative to training goals. You run the following Transact-SQL statements:



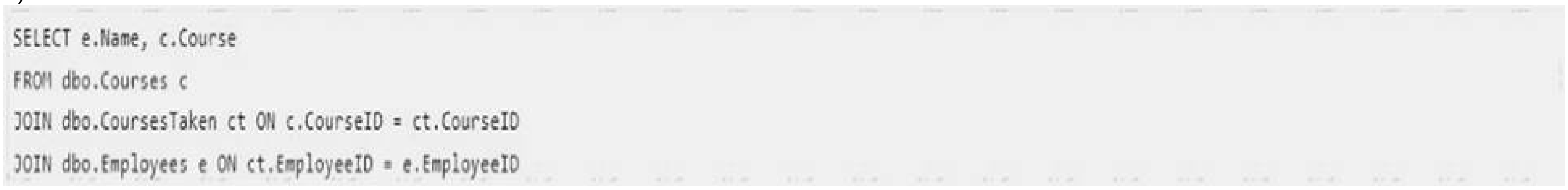
```
CREATE TABLE Employees(
  EmployeeID INT IDENTITY(1,1) NOT NULL,
  Name VARCHAR(150) NULL,
  CONSTRAINT PK_Employees PRIMARY KEY CLUSTERED (
   EmployeeID ASC
   ) WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF) ON PRIMARY
   ) ON PRIMARY

CREATE TABLE CoursesTaken(
CourseID INT NOT NULL,
EmployeeID INT NOT NULL,
CourseTakenOn DATE NULL,
CONSTRAINT PK_CoursesTaken PRIMARY KEY CLUSTERED (
 CourseID ASC, EmployeeID ASC
 ) WITH (STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF) ON PRIMARY

 ) ON PRIMARY
```

You must build a report that shows all Employees and the courses that they have taken. Employees that have not taken training courses must still appear in the report. The report must display NULL in the course column for these employees.
You need to create a query for the report.
A)



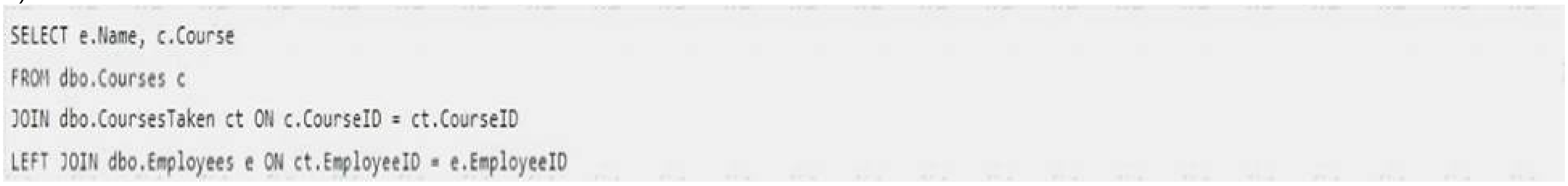
```
SELECT e.Name, c.Course
FROM dbo.Courses c
JOIN dbo.CoursesTaken ct ON c.CourseID = ct.CourseID
INNER JOIN dbo.Employees e ON ct.EmployeeID = e.EmployeeID
```

B)



```
SELECT e.Name, c.Course
FROM dbo.Courses c
JOIN dbo.CoursesTaken ct ON c.CourseID = ct.CourseID
JOIN dbo.Employees e ON ct.EmployeeID = e.EmployeeID
```

C)



```
SELECT e.Name, c.Course
FROM dbo.Courses c
JOIN dbo.CoursesTaken ct ON c.CourseID = ct.CourseID
LEFT JOIN dbo.Employees e ON ct.EmployeeID = e.EmployeeID
```

D)

```
SELECT e.Name, c.Course
FROM dbo.Courses c
JOIN dbo.CoursesTaken ct ON c.CourseID = ct.CourseID
RIGHT JOIN dbo.Employees e ON ct.EmployeeID = e.EmployeeID
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: A

NEW QUESTION 33
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You create a table named Products by running the following Transact-SQL statement:

```
CREATE TABLE Products (
     ProductID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
     ProductName nvarchar(100) NULL,
     UnitPrice decimal(18, 2) NOT NULL,
     UnitsInStock int NOT NULL,
     UnitsOnOrder int NULL
)
```

You have the following stored procedure:

```
CREATE PROCEDURE InsertProduct
     @ProductName nvarchar(100),
     @UnitPrice decimal(18,2),
     @UnitsInStock int,
     @UnitsOnOrder int
AS
BEGIN
     INSERT INTO Products(ProductName,ProductPrice,ProductsInStock,ProductsOnOrder)
     VALUES (@ProductName,@UnitPrice,@UnitsInStock,@UnitsOnOrder)
END
```

You need to modify the stored procedure to meet the following new requirements:
- Insert product records as a single unit of work.
- Return error number 51000 when a product fails to insert into the database.
- If a product record insert operation fails, the product information must not be permanently written to the database.
Solution: You run the following Transact-SQL statement:

```
ALTER PROCEDURE InsertProduct
@ProductName nvarchar(100),
@UnitPrice decimal(18,2),
@UnitsInStock int,
@UnitsOnOrder int
AS
BEGIN
     SET XACT_ABORT ON
     BEGIN TRY
          BEGIN TRANSACTION
               INSERT INTO Products(ProductName,ProductPrice,ProductsInStock,ProductsOnOrder)
               VALUES (@ProductName,@UnitPrice,@UnitsInStock,@UnitsOnOrder)
          COMMIT TRANSACTION
     END TRY
     BEGIN CATCH
          IF XACT_STATE() <> 0 ROLLBACK TRANSACTION
          THROW 51000, 'The product could not be created.', 1
     END CATCH
END
```

Does the solution meet the goal?

A. Yes
B. No

Answer: B

Explanation:
With X_ABORT ON the INSERT INTO statement and the transaction will be rolled back when an error is raised, it would then not be possible to ROLLBACK it again in the IF XACT_STATE() <> 0 ROLLACK TRANSACTION statement.
Note: A transaction is correctly defined for the INSERT INTO ..VALUES statement, and if there is an error in the transaction it will be caught ant he transaction will be rolled back, finally an error 51000 will be raised.
Note: When SET XACT_ABORT is ON, if a Transact-SQL statement raises a run-time error, the entire transaction is terminated and rolled back.
XACT_STATE is a scalar function that reports the user transaction state of a current running request. XACT_STATE indicates whether the request has an active user transaction, and whether the transaction is capable of being committed.
The states of XACT_STATE are:
0 There is no active user transaction for the current request.
1 The current request has an active user transaction. The request can perform any actions, including writing data and committing the transaction.
2 The current request has an active user transaction, but an error has occurred that has caused the transaction to be classified as an committable transaction.
References:
https://msdn.microsoft.com/en-us/library/ms188792.aspx https://msdn.microsoft.com/en-us/library/ms189797.aspx

NEW QUESTION 38
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a table named Products that stores information about products your company sells. The table has a column named ListPrice that stores retail pricing information for products.
Some products are used only internally by the company. Records for these products are maintained in the Products table for inventory purposes. The price for each of these products is $0.00. Customers are not permitted to order these products.
You need to increase the list price for products that cost less than $100 by 10 percent. You must only increase pricing for products that customers are permitted to order.
Solution: You run the following Transact-SQL statement:

```
UPDATE Production.Product
SET ListPrice = ListPrice + 1.1
WHERE ListPrice
BETWEEN .01 and 99.99
```

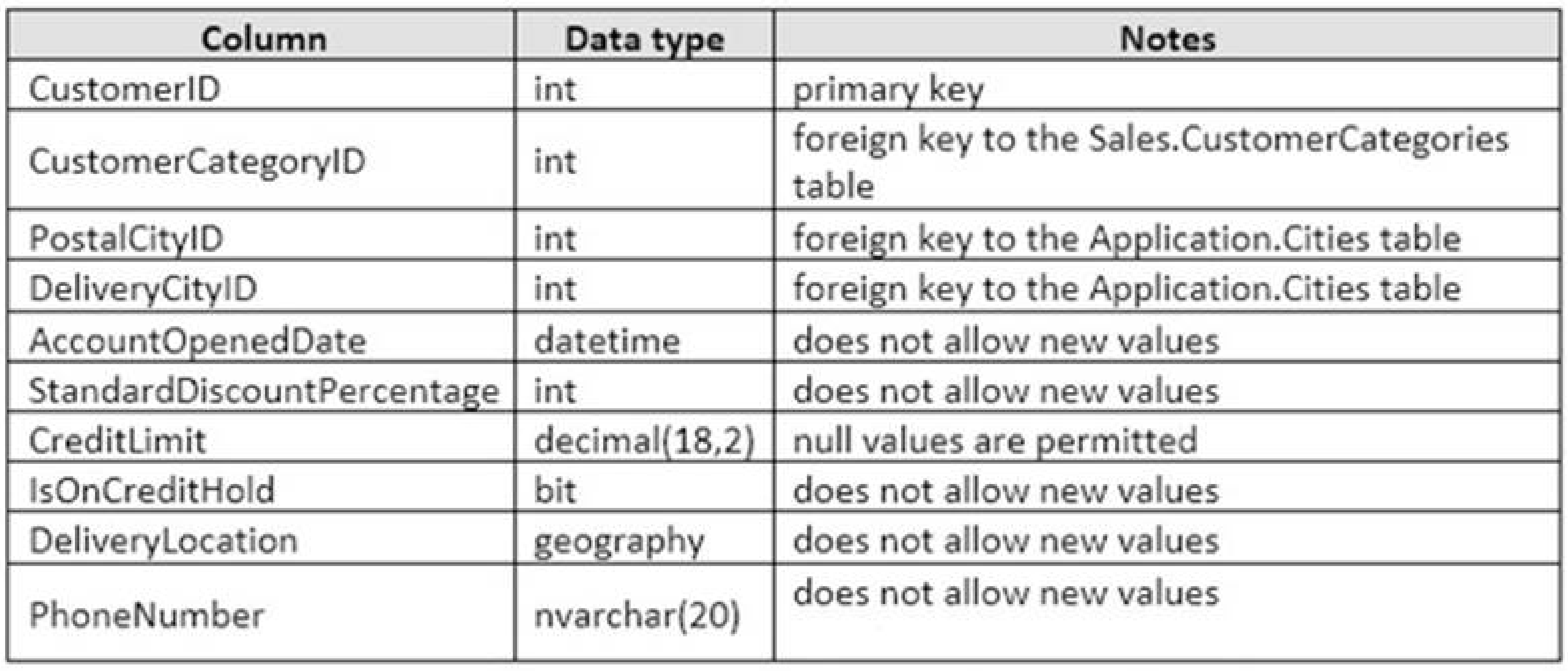
Does the solution meet the goal?

A. Yes
B. No

Answer: B

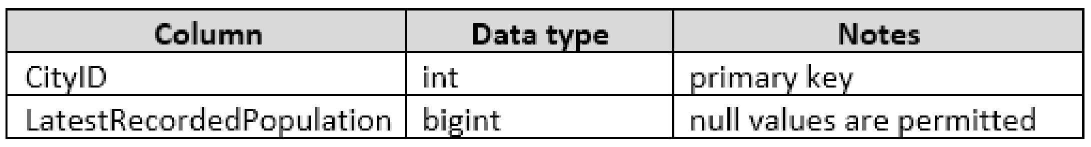
Explanation:
Products with a price between $0.00 and $100 will be increased, while products with a price of $0.00 would not be increased.

NEW QUESTION 43
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a database that tracks orders and deliveries for customers in North America. The database contains the following tables:
Sales.Customers



Column	Data type	Notes
CustomerID	int	primary key
CustomerCategoryID	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow new values
StandardDiscountPercentage	int	does not allow new values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow new values
DeliveryLocation	geography	does not allow new values
PhoneNumber	nvarchar(20)	does not allow new values

Application.Cities



Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Sales.CustomerCategories

Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

Your company is developing a new social application that connects customers to each other based on the distance between their delivery locations.
You need to write a query that returns the nearest customer. Solution: You run the following Transact-SQL statement:

```
SELECT TOP 1 B.CustomerID, B.DeliveryLocation ^ A.DeliveryLocation AS Dist
FROM Sales.Customers AS A
JOIN Sales.Customers AS B
ON A.DeliveryCityID = B.DeliveryCityID
WHERE A.CustomerID = @custID AND A.CustomerID <> B.CustomerID
```

The variable @custID is set to a valid customer. Does the solution meet the goal?

A. Yes
B. No

Answer: B

NEW QUESTION 48
You have a database named MyDb. You run the following Transact-SQL statements:

```
CREATE TABLE tblRoles (
     RoleId int NOT NULL IDENTITY(1,1) PRIMARY KEY CLUSTERED,
     RoleName varchar(20) NOT NULL
)
CREATE TABLE tblUsers (
     UserId int NOT NULL IDENTITY(10000,1) PRIMARY KEY CLUSTERED,
     UserName varchar(20) UNIQUE NOT NULL,
     RoleId int NULL FOREIGN KEY REFERENCES tbRoles(RoleId),
     IsActive bit NOT NULL DEFAULT(1)
)
```

A value of 1 in the IsActive column indicates that a user is active.
You need to create a count for active users in each role. If a role has no active users. You must display a zero as the active users count.
Which Transact-SQL statement should you run?

A. SELECT R.RoleName, COUNT(*) AS ActiveUserCount FROM tblRoles RCROSS JOIN (SELECT UserId, RoleId FROM tblUsers WHERE IsActive = 1) UWHERE U.RoleId = R.RoleIdGROUP BY R.RoleId, R.RoleName
B. SELECT R.RoleName, COUNT(*) AS ActiveUserCount FROM tblRoles RLEFT JOIN (SELECTUserId, RoleId FROM tblUsers WHERE IsActive = 1) UON U.RoleId = R.RoleIdGROUP BY R.RoleId, R.RoleName
C. SELECT R.RoleName, U.ActiveUserCount FROM tblRoles R CROSS JOIN(SELECT RoleId, COUNT(*) AS ActiveUserCountFROM tblUsers WHERE IsActive = 1 GROUP BY R.RoleId) U
D. SELECT R.RoleName, ISNULL (U.ActiveUserCount,0) AS ActiveUserCountFROM tblRoles R LEFT JOIN (SELECT RoleId, COUNT(*) AS ActiveUserCountFROM tblUsers WHERE IsActive = 1 GROUP BY R.RoleId) U

Answer: B

NEW QUESTION 51
You create a table named Sales.Categories by running the following Transact-SQL statement:

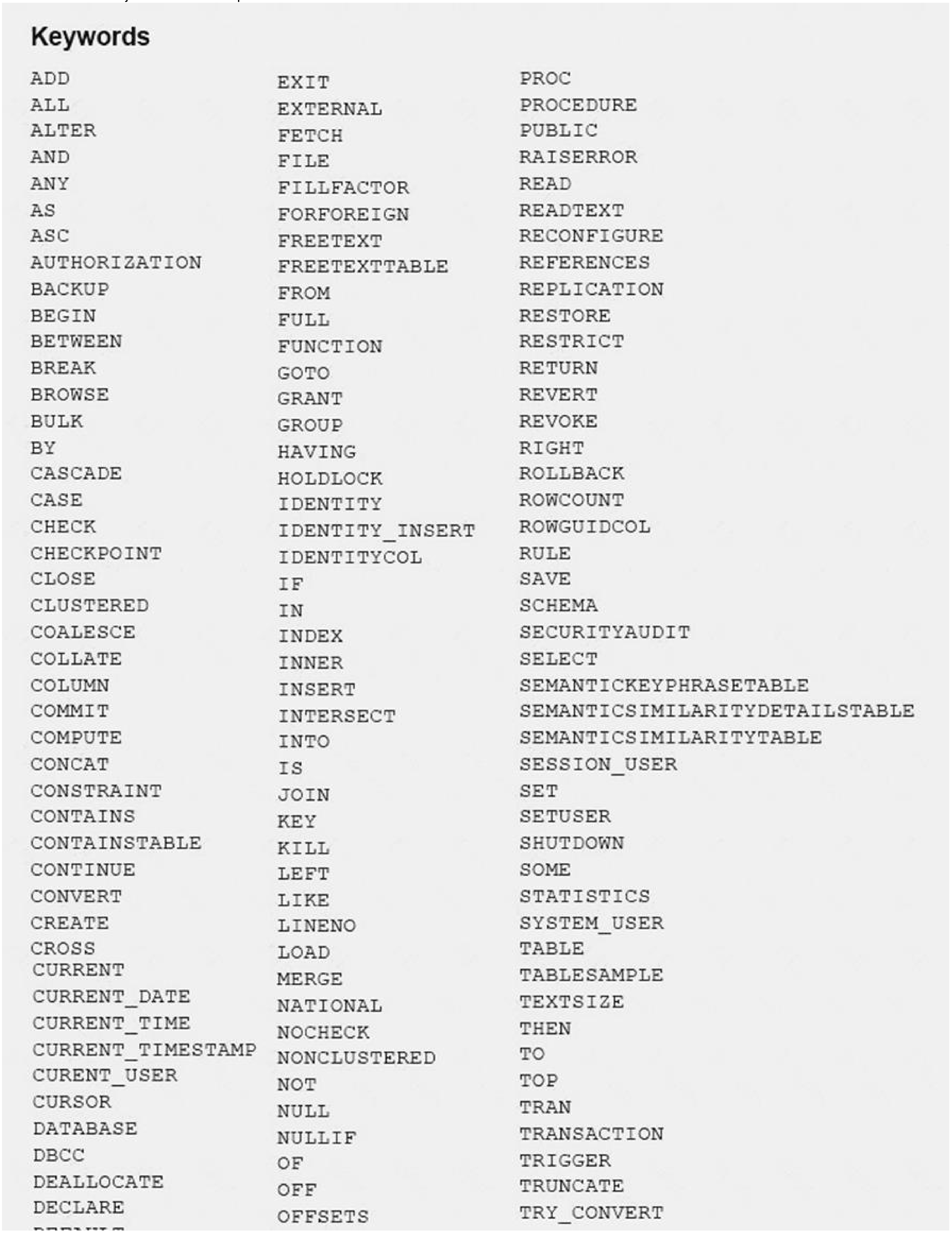
```
CREATE TABLE Sales.Categories (
     CategoryID smallint NOT NULL PRIMARY KEY,
     Name nvarchar(50) NOT NULL,
     ParentCategoryID int NULL
)
```

You add the following data to the table.

CategoryID	Name	ParentCategoryID
1	Electronics	NULL
2	Cameras and photography	1
3	Computers and tablets	1
4	Cell phones and accessories	1
5	TV and audio	1
6	Digital cameras	2
9	laptops	3
13	Household goods	NULL
14	Bathroom items	13
15	Shower curtains	14

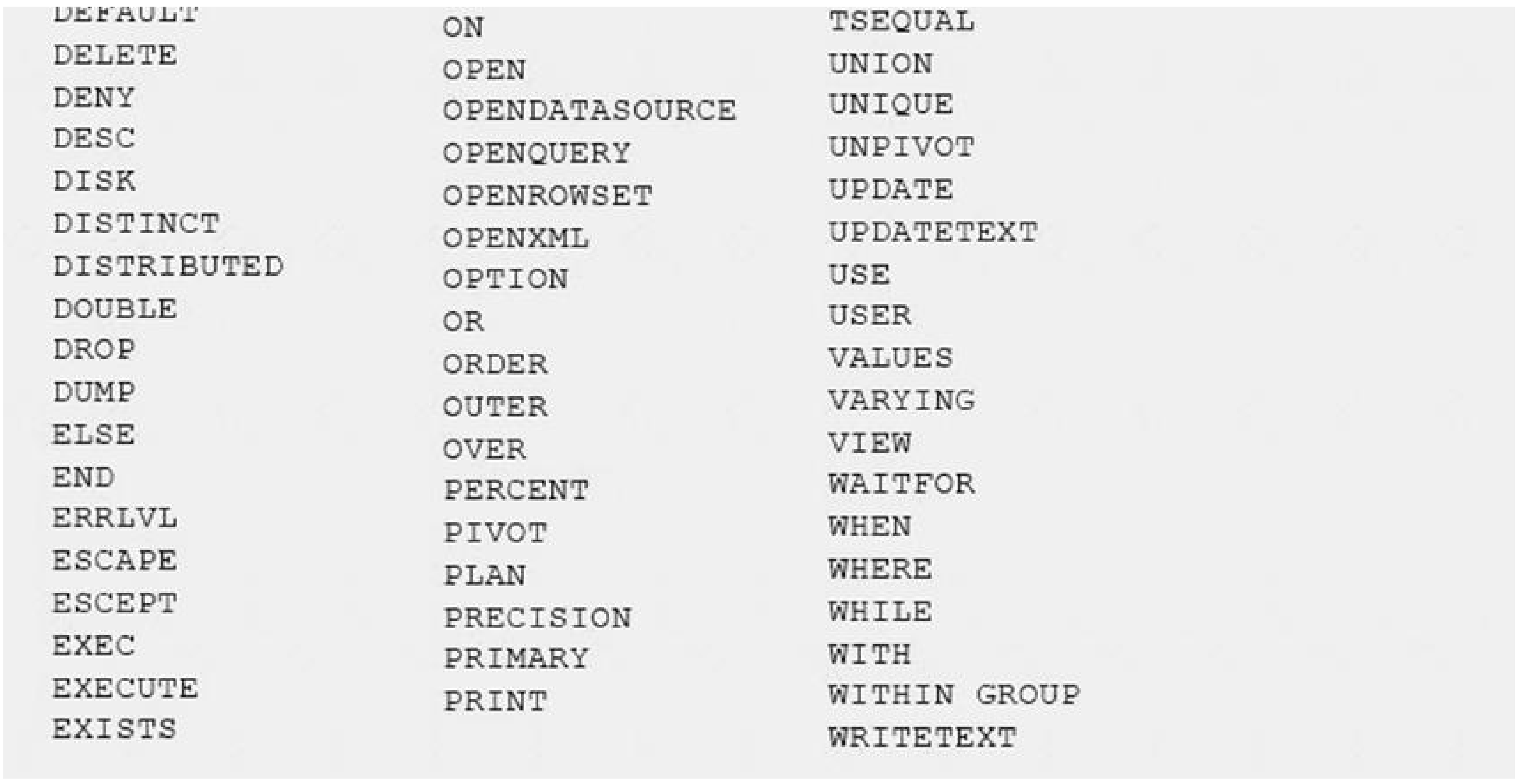
You need to create a query that uses a common table expression (CTE) to show the parent category of each category. The query must meet the following requirements:

Return all columns from the Categories table in the order shown.
Exclude all categories that do not have a parent category.
Construct the query using the following guidelines:
Name the expression ParentCategories.
Use PC as the alias for the expression.
Use C as the alias for the Categories table.
Use the AS keyword for all table aliases.
Use individual column names for each column that the query returns.
Do not use a prefix for any column name.
Do not surround object names with square brackets.



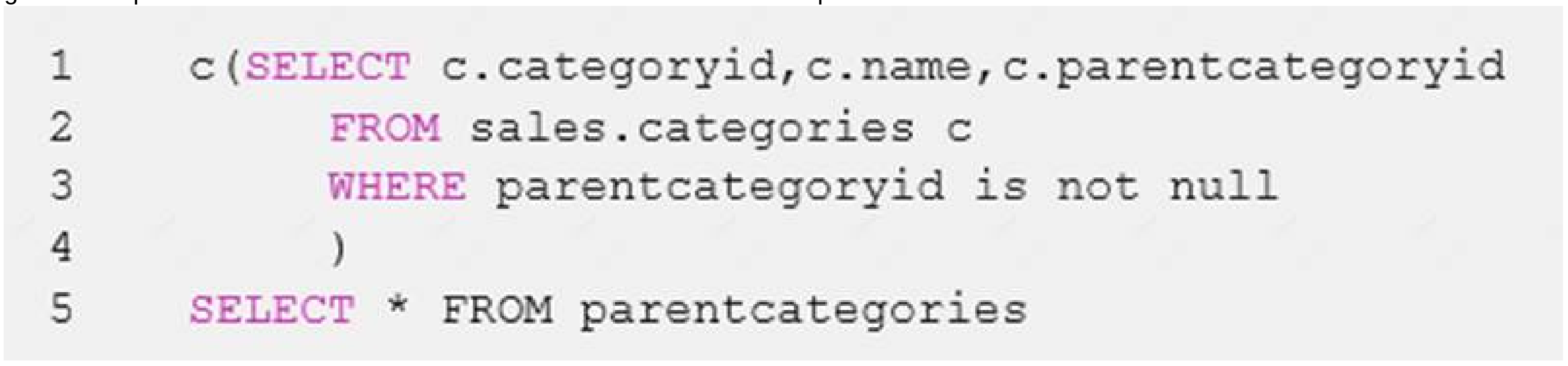
Keywords

```
ADD                    EXIT                  PROC
ALL                    EXTERNAL              PROCEDURE
ALTER                  FETCH                 PUBLIC
AND                    FILE                  RAISERROR
ANY                    FILLFACTOR            READ
AS                     FORFOREIGN            READTEXT
ASC                    FREETEXT              RECONFIGURE
AUTHORIZATION          FREETEXTTABLE         REFERENCES
BACKUP                 FROM                  REPLICATION
BEGIN                  FULL                  RESTORE
BETWEEN                FUNCTION              RESTRICT
BREAK                  GOTO                  RETURN
BROWSE                 GRANT                 REVERT
BULK                   GROUP                 REVOKE
BY                     HAVING                RIGHT
CASCADE                HOLDLOCK              ROLLBACK
CASE                   IDENTITY              ROWCOUNT
CHECK                  IDENTITY_INSERT       ROWGUIDCOL
CHECKPOINT             IDENTITYCOL           RULE
CLOSE                  IF                    SAVE
CLUSTERED              IN                    SCHEMA
COALESCE               INDEX                 SECURITYAUDIT
COLLATE                INNER                 SELECT
COLUMN                 INSERT                SEMANTICKEYPHRASETABLE
COMMIT                 INTERSECT             SEMANTICSIMILARITYDETAILSTABLE
COMPUTE                INTO                  SEMANTICSIMILARITYTABLE
CONCAT                 IS                    SESSION_USER
CONSTRAINT             JOIN                  SET
CONTAINS               KEY                   SETUSER
CONTAINSTABLE          KILL                  SHUTDOWN
CONTINUE               LEFT                  SOME
CONVERT                LIKE                  STATISTICS
CREATE                 LINENO                SYSTEM_USER
CROSS                  LOAD                  TABLE
CURRENT                MERGE                 TABLESAMPLE
CURRENT_DATE           NATIONAL              TEXTSIZE
CURRENT_TIME           NOCHECK               THEN
CURRENT_TIMESTAMP      NONCLUSTERED          TO
CURENT_USER            NOT                   TOP
CURSOR                 NULL                  TRAN
DATABASE               NULLIF                TRANSACTION
DBCC                   OF                    TRIGGER
DEALLOCATE             OFF                   TRUNCATE
DECLARE                OFFSETS               TRY_CONVERT
```



```
DEFAULT         ON               TSEQUAL
DELETE          OPEN             UNION
DENY            OPENDATASOURCE   UNIQUE
DESC            OPENQUERY        UNPIVOT
DISK            OPENROWSET       UPDATE
DISTINCT        OPENXML          UPDATETEXT
DISTRIBUTED     OPTION           USE
DOUBLE          OR               USER
DROP            ORDER            VALUES
DUMP            OUTER            VARYING
ELSE            OVER             VIEW
END             PERCENT          WAITFOR
ERRLVL          PIVOT            WHEN
ESCAPE          PLAN             WHERE
ESCEPT          PRECISION        WHILE
EXEC            PRIMARY          WITH
EXECUTE         PRINT            WITHIN GROUP
EXISTS                           WRITETEXT
```

Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the code that has been provided as well as below it.



```
1    c(SELECT c.categoryid,c.name,c.parentcategoryid
2         FROM sales.categories c
3         WHERE parentcategoryid is not null
4         )
5    SELECT * FROM parentcategories
```

Use the Check Syntax button to verify your work. Any syntax or spelling errors will be reported by line and character position. You may check syntax as many times as needed.

A. Mastered
B. Not Mastered

Answer: A

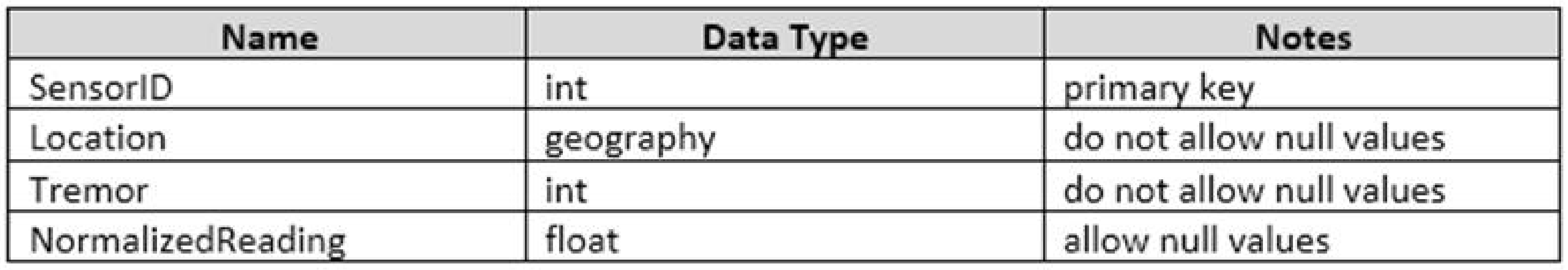
Explanation:
1 WITH ParentCategories pc (CategoryID, Name, PatentCategoryID) AS (SELECT c.categoryID,c.name,c.parentcategoryid
2 FROM sales.categories c
3 WHERE parentcategoryid is not null
4)
5 SELECT * FROM parentcategories
Note: On Line 1 replace c with WITH ParentCategories pc (CategoryID, Name, PatentCategoryID) AS Note: The basic syntax structure for a CTE is:
WITH expression_name [(column_name [,...n])] AS
(CTE_query_definition)
References: https://technet.microsoft.com/en-us/library/ms190766(v=sql.105).aspx

NEW QUESTION 55
You need to create an indexed view that requires logic statements to manipulate the data that the view displays.
Which two database objects should you use? Each correct answer presents a complete solution.

A. a user-defined table-valued function
B. a CRL function
C. a stored procedure
D. a user-defined scalar function

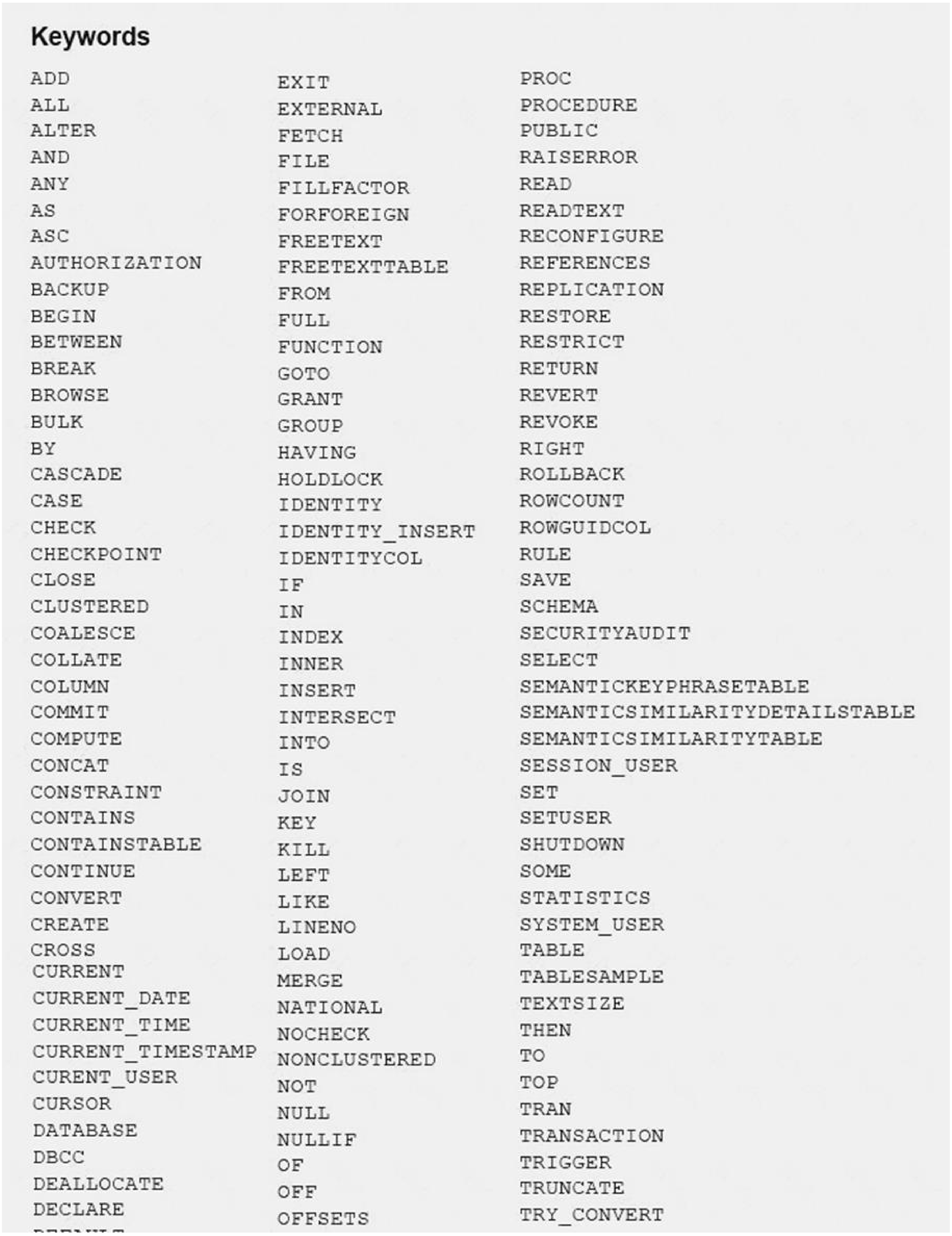
Answer: AC

NEW QUESTION 57
You work for an organization that monitors seismic activity around volcanos. You have a table named GroundSensors. The table stored data collected from seismic sensors. It includes the columns describes in the following table:



Name	Data Type	Notes
SensorID	int	primary key
Location	geography	do not allow null values
Tremor	int	do not allow null values
NormalizedReading	float	allow null values

The database also contains a scalar value function named NearestMountain that returns the name of the mountain that is nearest to the sensor.
You need to create a query that shows the average of the normalized readings from the sensors for each mountain. The query must meet the following requirements:
* Include the average normalized readings and nearest mountain name.
* Exclude sensors for which no normalized reading exists.
* Exclude those sensors with value of zero for tremor. Construct the query using the following guidelines:
* Use one part names to reference tables, columns and functions.
* Do not use parentheses unless required.
* Do not use aliases for column names and table names.
* Do not surround object names with square brackets.

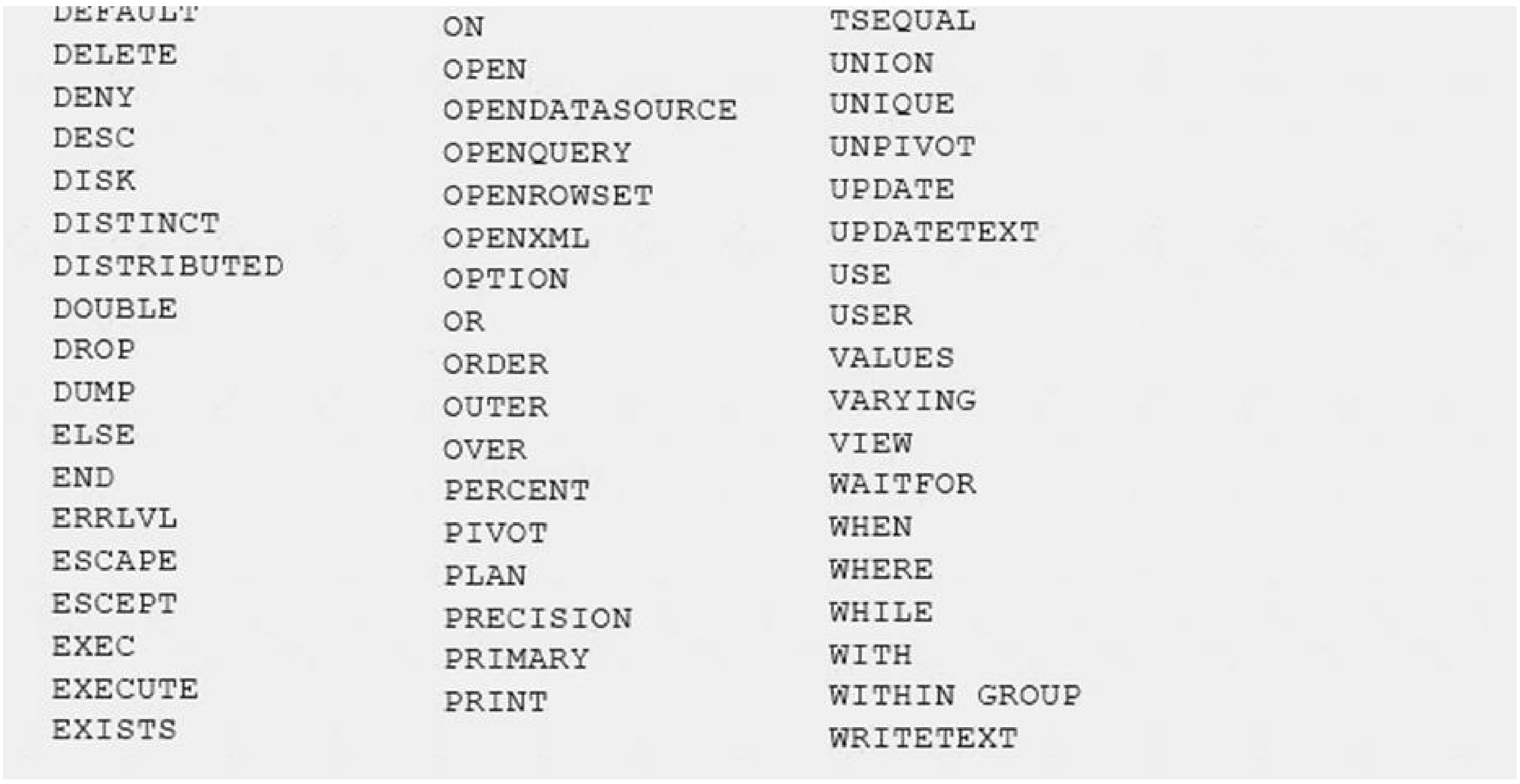


Keywords

ADD
ALL
ALTER
AND
ANY
AS
ASC
AUTHORIZATION
BACKUP
BEGIN
BETWEEN
BREAK
BROWSE
BULK
BY
CASCADE
CASE
CHECK
CHECKPOINT
CLOSE
CLUSTERED
COALESCE
COLLATE
COLUMN
COMMIT
COMPUTE
CONCAT
CONSTRAINT
CONTAINS
CONTAINSTABLE
CONTINUE
CONVERT
CREATE
CROSS
CURRENT
CURRENT_DATE
CURRENT_TIME
CURRENT_TIMESTAMP
CURENT_USER
CURSOR
DATABASE
DBCC
DEALLOCATE
DECLARE

EXIT
EXTERNAL
FETCH
FILE
FILLFACTOR
FORFOREIGN
FREETEXT
FREETEXTTABLE
FROM
FULL
FUNCTION
GOTO
GRANT
GROUP
HAVING
HOLDLOCK
IDENTITY
IDENTITY_INSERT
IDENTITYCOL
IF
IN
INDEX
INNER
INSERT
INTERSECT
INTO
IS
JOIN
KEY
KILL
LEFT
LIKE
LINENO
LOAD
MERGE
NATIONAL
NOCHECK
NONCLUSTERED
NOT
NULL
NULLIF
OF
OFF
OFFSETS

PROC
PROCEDURE
PUBLIC
RAISERROR
READ
READTEXT
RECONFIGURE
REFERENCES
REPLICATION
RESTORE
RESTRICT
RETURN
REVERT
REVOKE
RIGHT
ROLLBACK
ROWCOUNT
ROWGUIDCOL
RULE
SAVE
SCHEMA
SECURITYAUDIT
SELECT
SEMANTICKEYPHRASETABLE
SEMANTICSIMILARITYDETAILSTABLE
SEMANTICSIMILARITYTABLE
SESSION_USER
SET
SETUSER
SHUTDOWN
SOME
STATISTICS
SYSTEM_USER
TABLE
TABLESAMPLE
TEXTSIZE
THEN
TO
TOP
TRAN
TRANSACTION
TRIGGER
TRUNCATE
TRY_CONVERT



DEFAULT	ON	TSEQUAL
DELETE	OPEN	UNION
DENY	OPENDATASOURCE	UNIQUE
DESC	OPENQUERY	UNPIVOT
DISK	OPENROWSET	UPDATE
DISTINCT	OPENXML	UPDATETEXT
DISTRIBUTED	OPTION	USE
DOUBLE	OR	USER
DROP	ORDER	VALUES
DUMP	OUTER	VARYING
ELSE	OVER	VIEW
END	PERCENT	WAITFOR
ERRLVL	PIVOT	WHEN
ESCAPE	PLAN	WHERE
ESCEPT	PRECISION	WHILE
EXEC	PRIMARY	WITH
EXECUTE	PRINT	WITHIN GROUP
EXISTS		WRITETEXT

Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the code that has been provided as well as below it.

```
select
```

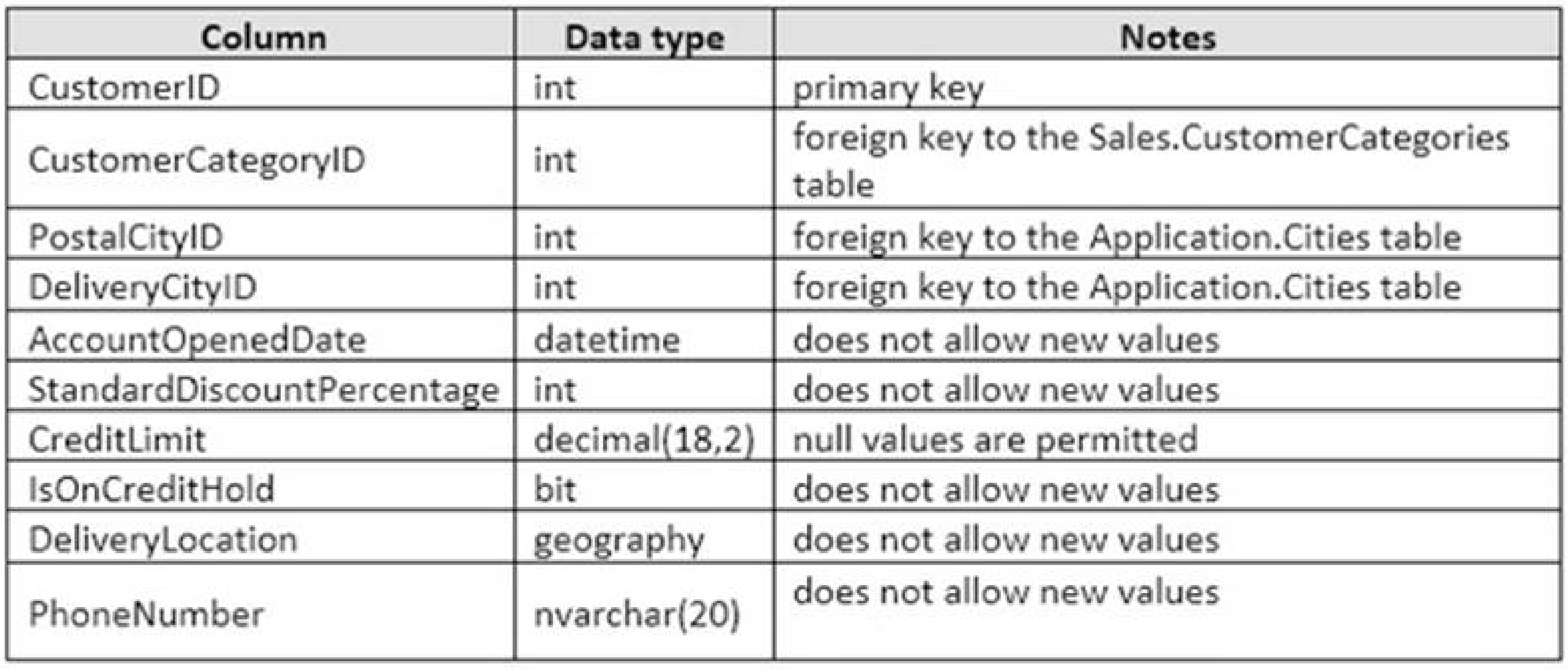
Use the Check Syntax button to verify your work. Any syntax or spelling errors will be reported by line and character position.

A. Mastered
B. Not Mastered

Answer: A

Explanation:
GROUP BY is a SELECT statement clause that divides the query result into groups of rows, usually for the purpose of performing one or more aggregations on each group. The SELECT statement returns one row per group.
SELECT SensorID, NearestMountain(Location) FROM GroundSensors
WHERE TREMOR &lt;&gt; 0 AND NormalizedReading IS NOT NULL
GROUP BY SensorID, NearestMountain(Location)
References: https://msdn.microsoft.com/en-us/library/ms177673.aspx

NEW QUESTION 62
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a database that tracks orders and deliveries for customers in North America. The database contains the following tables:
Sales.Customers



Column	Data type	Notes
CustomerID	int	primary key
CustomerCategoryID	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow new values
StandardDiscountPercentage	int	does not allow new values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow new values
DeliveryLocation	geography	does not allow new values
PhoneNumber	nvarchar(20)	does not allow new values

Application.Cities

Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Sales.CustomerCategories

Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

Your company is developing a new social application that connects customers to each other based on the distance between their delivery locations.
You need to write a query that returns the nearest customer. Solution: You run the following Transact-SQL statement:

```
WITH DIST_CTE (CustA, CustB, Dist)
AS (
      SELECT A.CustomerID AS CustA, B.CustomerID AS CustB,
B.DeliveryLocation.ShortestLineTo(A.DeliveryLocation).STLength() AS Dist
      FROM Sales.Customers AS A
      CROSS JOIN Sales.Customers AS B
      WHERE A.CustomerID <> B.CustomerID
)
SELECT TOP 1 CustB, Dist
FROM DIST_CTE
WHERE CustA = @custID
ORDER BY Dist
```

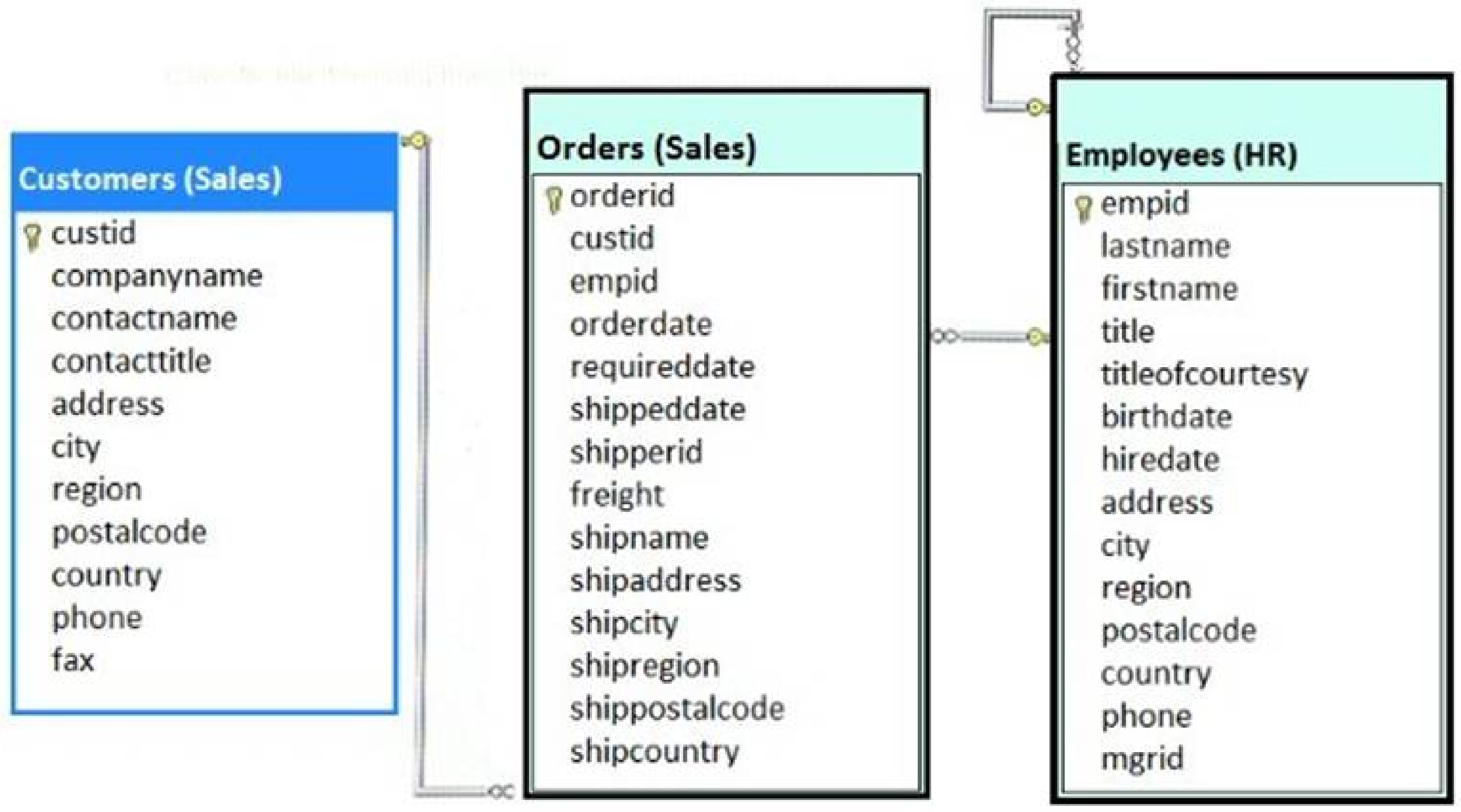
The variable @custID is set to a valid customer. Does the solution meet the goal?

A. Yes
B. No

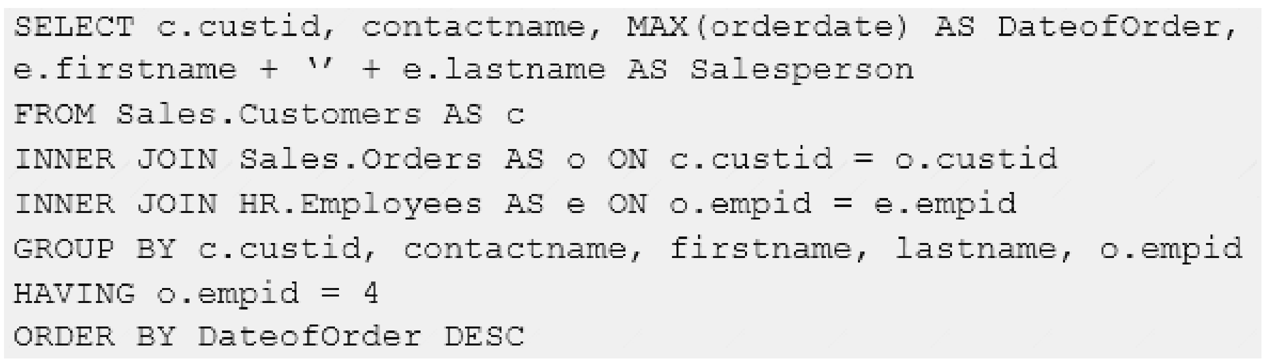
Answer: A

Explanation:
ShortestLineTo (geometry Data Type) Returns a LineString instance with two points that represent the shortest distance between the two geometry instances. The length of the LineString instance returned is the distance between the two geometry instances.
STLength (geometry Data Type) returns the total length of the elements in a geometry instance. References: https://docs.microsoft.com/en-us/sql/t-sql/spatial-geometry/shortestlineto-geometry-data-type

NEW QUESTION 67
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a database that includes the tables shown in the exhibit (Click the Exhibit button.)



You need to create a Transact-SQL query that returns the following information:
*the customer number
* the customer contact name
*the date the order was placed, with a name of DateofOrder
*a column named Salesperson, formatted with the employee first name, a space, and the employee last name
*orders for customers where the employee identifier equals 4
The output must be sorted by order date, with the newest orders first. The solution must return only the most recent order for each customer. Solution: You run the following Transact-SQL statement:



```
SELECT c.custid, contactname, MAX(orderdate) AS DateofOrder,
e.firstname + ' ' + e.lastname AS Salesperson
FROM Sales.Customers AS c
INNER JOIN Sales.Orders AS o ON c.custid = o.custid
INNER JOIN HR.Employees AS e ON o.empid = e.empid
GROUP BY c.custid, contactname, firstname, lastname, o.empid
HAVING o.empid = 4
ORDER BY DateofOrder DESC
```

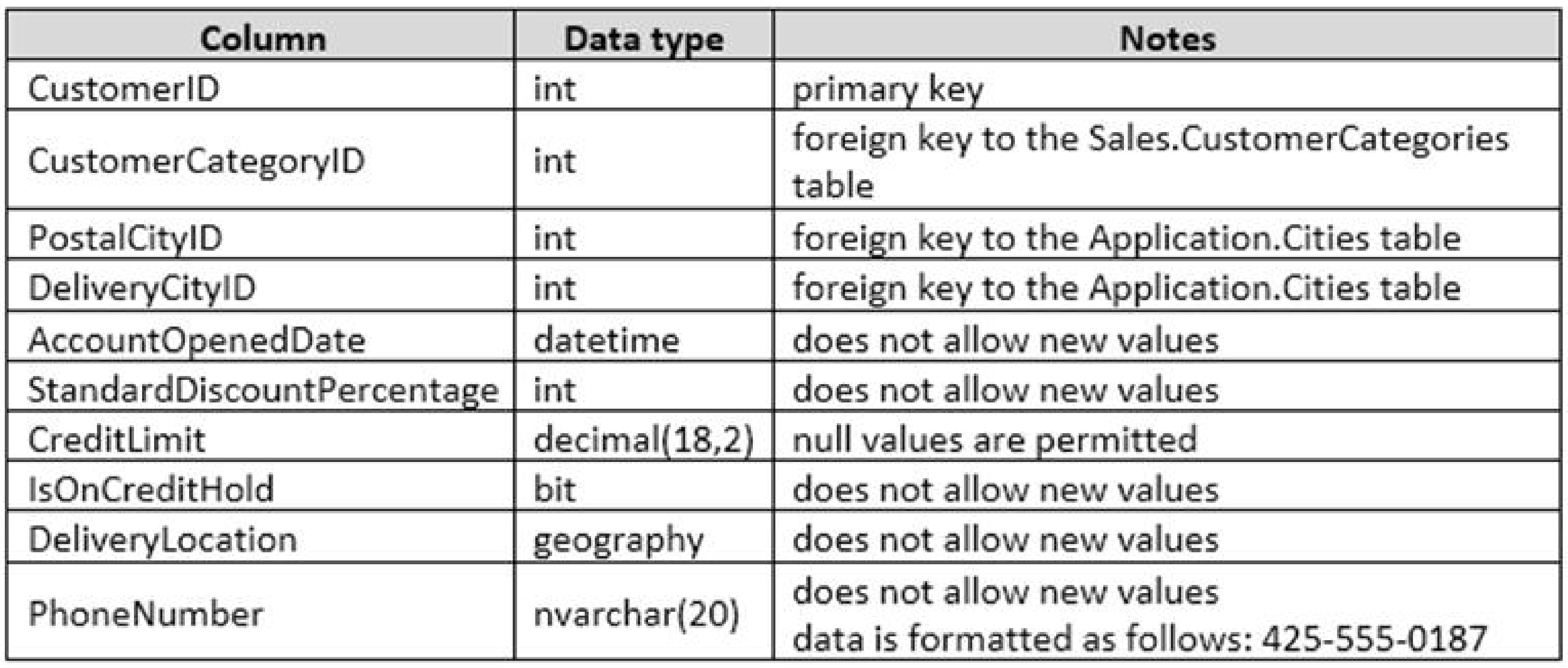
Does the solution meet the goal?

A. Yes
B. No

Answer: B

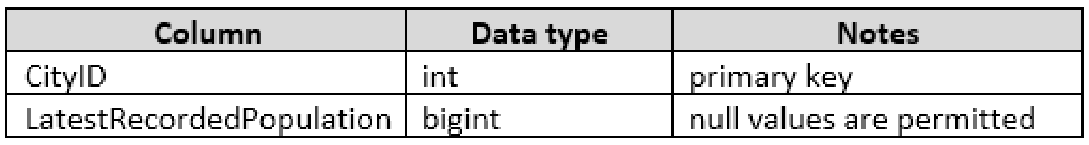
Explanation:
We should use a WHERE clause, not a HAVING clause. The HAVING clause would refer to aggregate data.

NEW QUESTION 68
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a database that tracks orders and deliveries for customers in North America. The database contains the following tables:
Sales.Customers



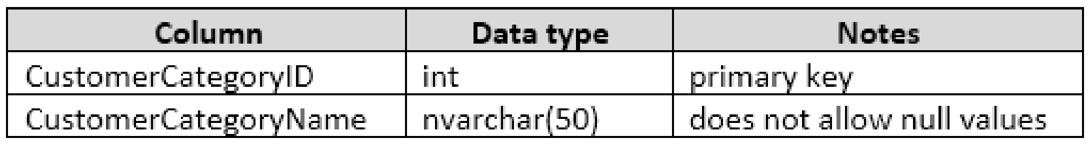
Column	Data type	Notes
CustomerID	int	primary key
CustomerCategoryID	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow new values
StandardDiscountPercentage	int	does not allow new values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow new values
DeliveryLocation	geography	does not allow new values
PhoneNumber	nvarchar(20)	does not allow new values data is formatted as follows: 425-555-0187

Application.Cities



Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Sales.CustomerCategories



Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

Your company is developing a new social application that connects customers to each other based on the distance between their delivery locations.
You need to write a query that returns the nearest customer. Solution: You run the following Transact-SQL statement:
SELECT TOP 1 B.CustomerID, A.DeliveryLocation.STDistance(B.DeliveryLocation) AS Dist FROM Sales.Customers AS A
CROSS JOIN Sales.Customers AS B
WHERE A.CustomerID = @custID AND A.CustomerID <> B.CustomerID ORDER BY Dist
The variable @custID is set to a valid customer. Does the solution meet the goal?

A. Yes
B. No

Answer: B

NEW QUESTION 69
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a table that was created by running the following Transact-SQL statement:

```
CREATE TABLE Products (
     ProductID int NOT NULL PRIMARY KEY,
     ProductName nvarchar(100) NULL,
     UnitPrice decimal(18, 2) NOT NULL,
     UnitsInStock int NOT NULL,
     UnitsOnOrder int NULL
)
```

The Products table includes the data shown in the following table:

ProductID	ProductName	UnitPrice	UnitsInStock	UnitsOnOrder
1	ProductA	10.00	10	15
2	ProductB	30.00	20	Null
3	ProductC	15.00	5	20

TotalUnitPrice is calculated by using the following formula: TotalUnitPrice = UnitPrice * (UnitsInStock + UnitsOnOrder)
You need to ensure that the value returned for TotalUnitPrice for ProductB is equal to 600.00. Solution: You run the following Transact-SQL statement:

```
SELECT ProductName, UnitPrice*(UnitsInStock+COALESCE(UnitsOnOnrder,0)) AS
TotalUnitPrice FROM Products
```

Does the solution meet the goal?

A. Yes
B. No

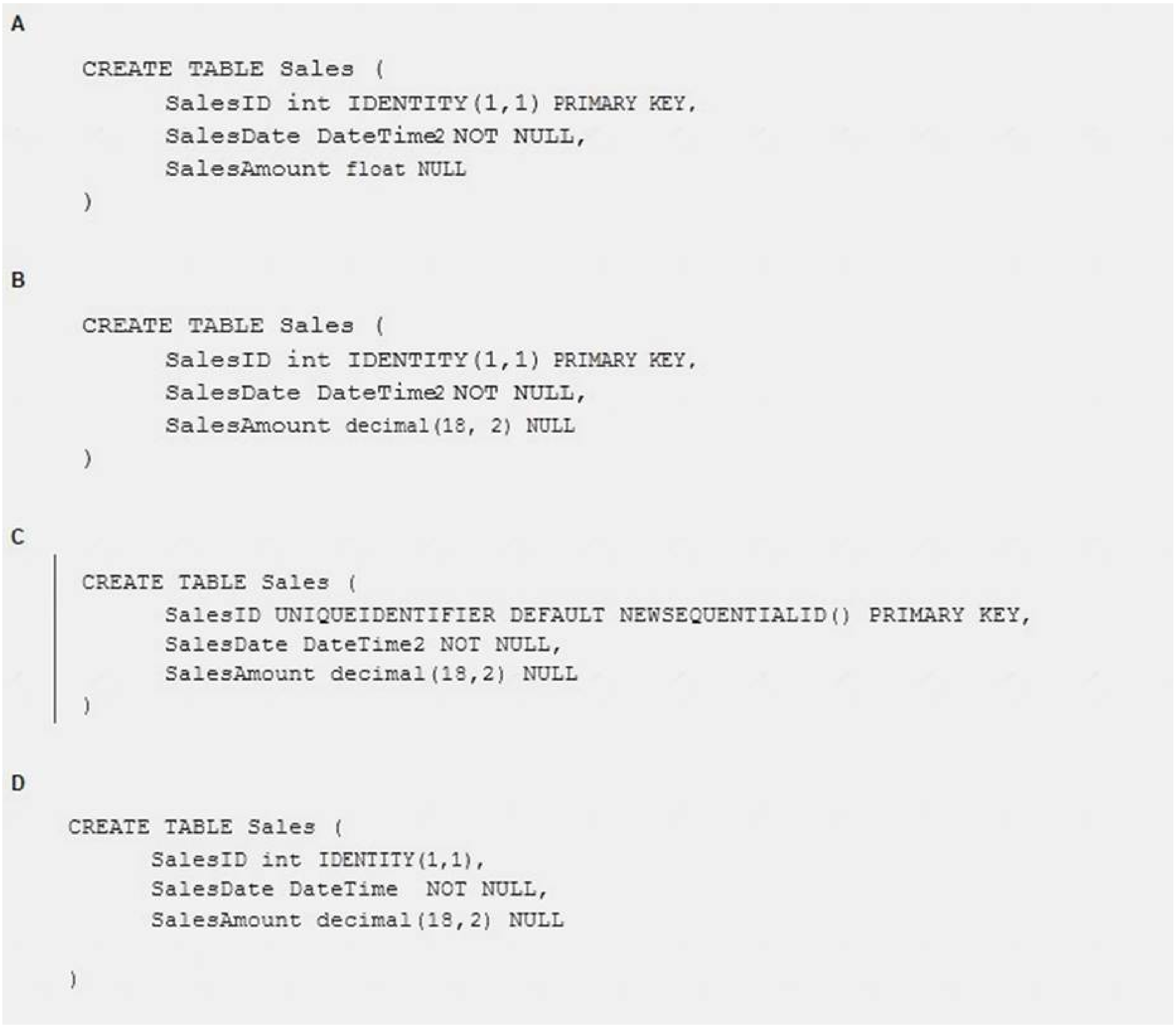
Answer: A

Explanation:
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

NEW QUESTION 72
You need to create a table named Sales that meets the following requirements:

Column name	Requirements
SalesID	- uniquely identify the row of data - automatically generate when data is inserted - use the least amount of storage space
SalesDate	- store the date and time of the sale based on 24-hour clock - use an ANSI SQL compliant data type
SalesAmount	- store the amount of the sale - avoid rounding errors when used in arithmetic calculations

Which Transact-SQL statement should you run?



A

```
CREATE TABLE Sales (
        SalesID int IDENTITY(1,1) PRIMARY KEY,
        SalesDate DateTime2 NOT NULL,
        SalesAmount float NULL
)
```

B

```
CREATE TABLE Sales (
        SalesID int IDENTITY(1,1) PRIMARY KEY,
        SalesDate DateTime2 NOT NULL,
        SalesAmount decimal(18, 2) NULL
)
```

C

```
CREATE TABLE Sales (
        SalesID UNIQUEIDENTIFIER DEFAULT NEWSEQUENTIALID() PRIMARY KEY,
        SalesDate DateTime2 NOT NULL,
        SalesAmount decimal(18,2) NULL
)
```

D

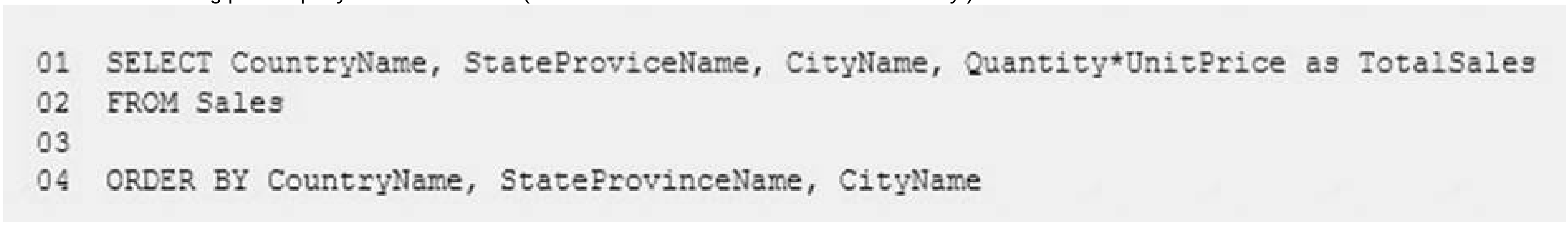
```
CREATE TABLE Sales (
        SalesID int IDENTITY(1,1),
        SalesDate DateTime  NOT NULL,
        SalesAmount decimal(18,2) NULL

)
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: B

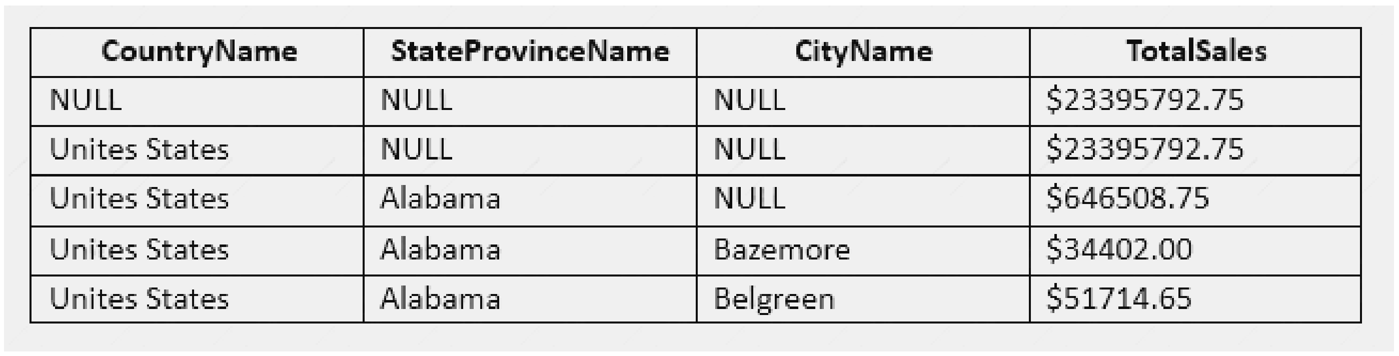
Explanation:
References:
https://docs.microsoft.com/en-us/sql/t-sql/data-types/decimal-and-numeric-transact-sql?view=sql-server-2017 https://docs.microsoft.com/en-us/sql/t-sql/data-types/float-and-real-transact-sql?view=sql-server-2017

NEW QUESTION 77
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You have a database that contains several connected tables. The tables contain sales data for customers in the United States only.
You have the following partial query for the database. (Line numbers are included for reference only.)



```
01  SELECT CountryName, StateProvinceName, CityName, Quantity*UnitPrice as TotalSales
02  FROM Sales
03
04  ORDER BY CountryName, StateProvinceName, CityName
```

You need to complete the query to generate the output shown in the following table.



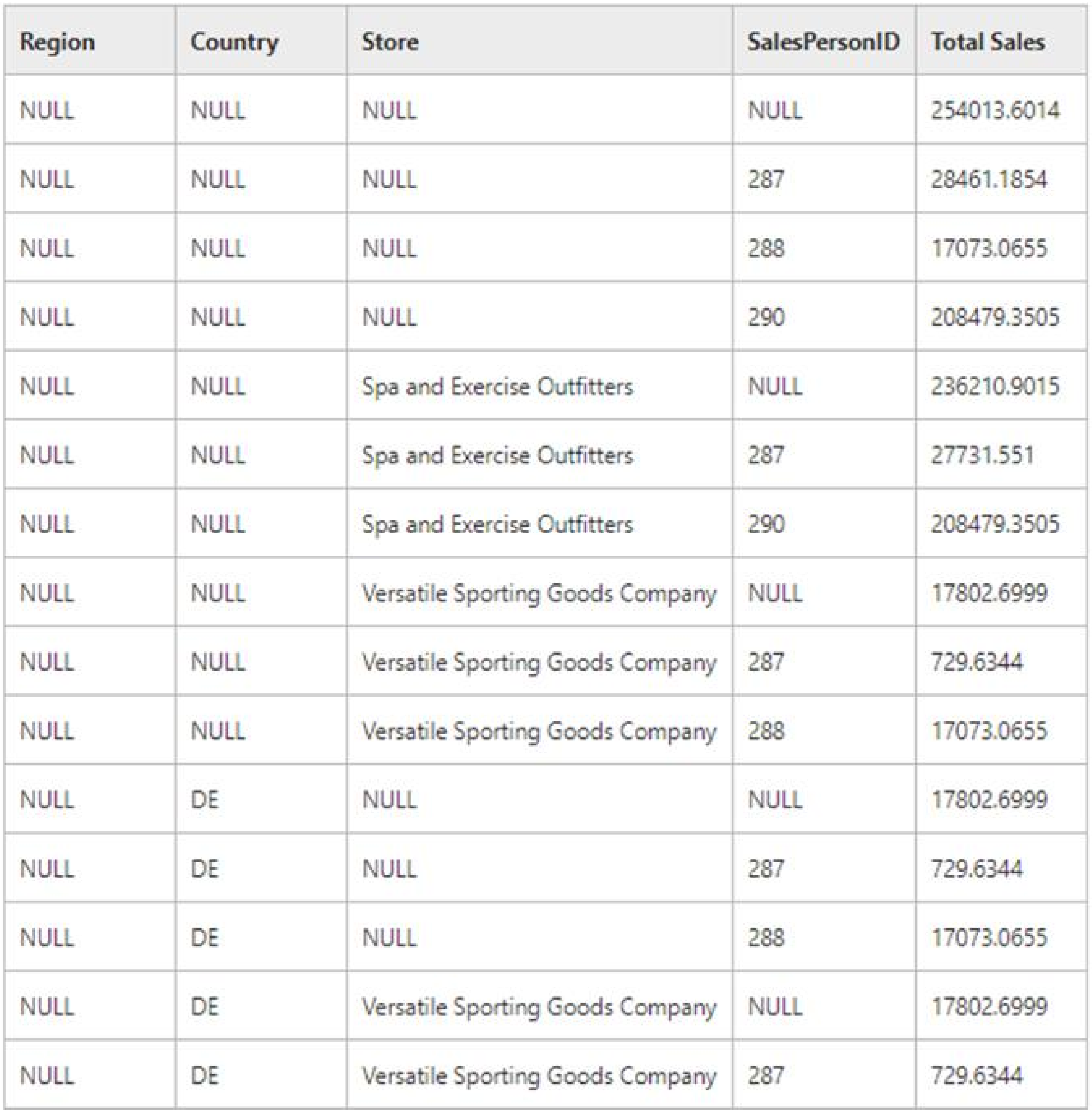
CountryName	StateProvinceName	CityName	TotalSales
NULL	NULL	NULL	$23395792.75
Unites States	NULL	NULL	$23395792.75
Unites States	Alabama	NULL	$646508.75
Unites States	Alabama	Bazemore	$34402.00
Unites States	Alabama	Belgreen	$51714.65

Which statement clause should you add at line 3?

A. GROUP BY
B. MERGE
C. GROUP BY ROLLUP
D. LEFT JOIN
E. GROUP BY CUBE
F. CROSS JOIN
G. PIVOT
H. UNPIVOT

Answer: E

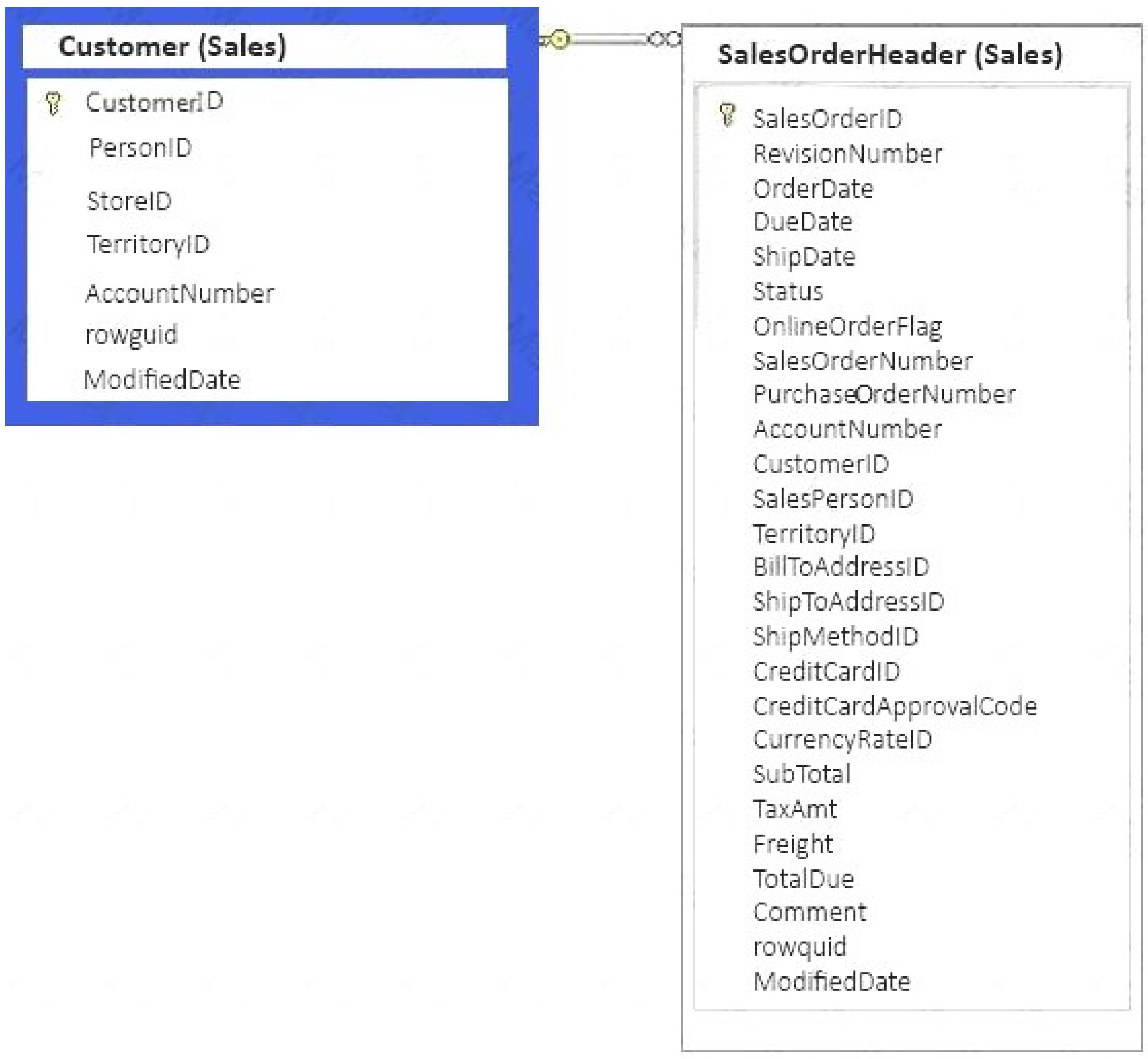
Explanation:
Example of GROUP BY CUBE result set:
In the following example, the CUBE operator returns a result set that has one grouping for all possible combinations of columns in the CUBE list and a grand total grouping.



Region	Country	Store	SalesPersonID	Total Sales
NULL	NULL	NULL	NULL	254013.6014
NULL	NULL	NULL	287	28461.1854
NULL	NULL	NULL	288	17073.0655
NULL	NULL	NULL	290	208479.3505
NULL	NULL	Spa and Exercise Outfitters	NULL	236210.9015
NULL	NULL	Spa and Exercise Outfitters	287	27731.551
NULL	NULL	Spa and Exercise Outfitters	290	208479.3505
NULL	NULL	Versatile Sporting Goods Company	NULL	17802.6999
NULL	NULL	Versatile Sporting Goods Company	287	729.6344
NULL	NULL	Versatile Sporting Goods Company	288	17073.0655
NULL	DE	NULL	NULL	17802.6999
NULL	DE	NULL	287	729.6344
NULL	DE	NULL	288	17073.0655
NULL	DE	Versatile Sporting Goods Company	NULL	17802.6999
NULL	DE	Versatile Sporting Goods Company	287	729.6344

References: https://technet.microsoft.com/en-us/library/bb522495(v=sql.105).aspx

NEW QUESTION 81
You have a database that includes the tables shown in the exhibit. (Click the exhibit button.)



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 01/01/1990 for the date.
Which Transact-SQL statement should you run?

A

```
SELECT C.CustomerID,  ISNULL (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
```

B

```
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
```

C

```
SELECT C.CustomerID, MAX(OrderDate)
FROM Sales.Customer C RIGHT OUTER JOIN Sales.SalesOrderHeader SOH
ON C.CustomerID = SOH.CustomerID
GROUP BY C.CustomerID
ORDER BY C.CustomerID
```

D

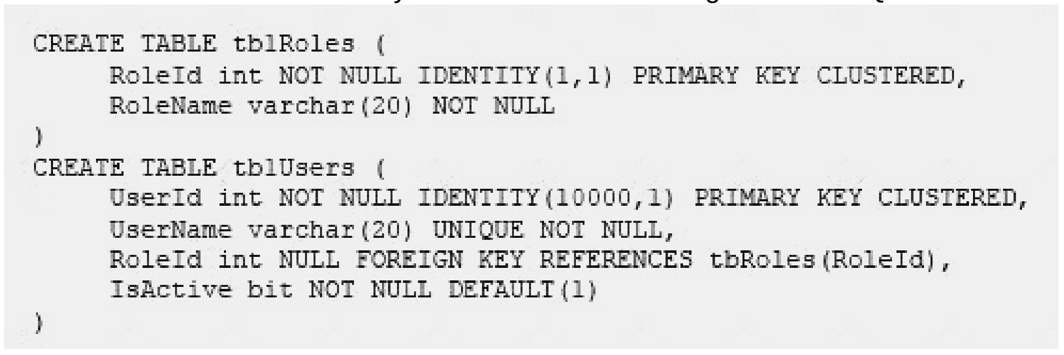
```
SELECT C.CustomerID, MAX(OrderDate)
FROM Sales.Customer C LEFT  OUTER JOIN Sales.SalesOrderHeader SOH
ON C.CustomerID = SOH.CustomerID
GROUP BY C.CustomerID
ORDER BY C.CustomerID
```

A. Option A
B. Option B
C. Option C

D. Option D

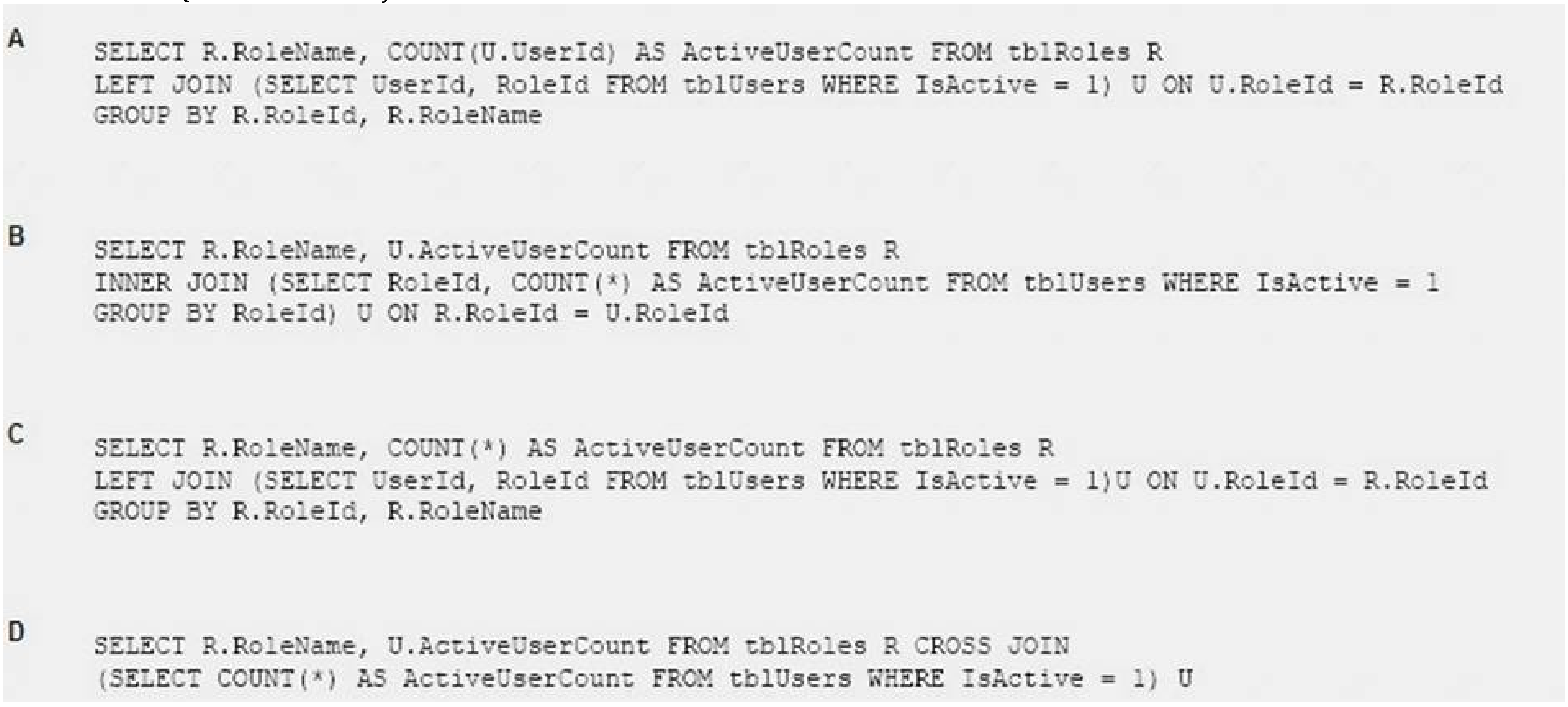
Answer: A

NEW QUESTION 86
You have a database named MyDb. You run the following Transact-SQL statements:



```
CREATE TABLE tblRoles (
     RoleId int NOT NULL IDENTITY(1,1) PRIMARY KEY CLUSTERED,
     RoleName varchar(20) NOT NULL
)
CREATE TABLE tblUsers (
     UserId int NOT NULL IDENTITY(10000,1) PRIMARY KEY CLUSTERED,
     UserName varchar(20) UNIQUE NOT NULL,
     RoleId int NULL FOREIGN KEY REFERENCES tbRoles(RoleId),
     IsActive bit NOT NULL DEFAULT(1)
)
```

A value of 1 in the IsActive column indicates that a user is active.
You need to create a count for active users in each role. If a role has no active users. you must display a zero as the active users count.
Which Transact-SQL statement should you run?



A
```
SELECT R.RoleName, COUNT(U.UserId) AS ActiveUserCount FROM tblRoles R
LEFT JOIN (SELECT UserId, RoleId FROM tblUsers WHERE IsActive = 1) U ON U.RoleId = R.RoleId
GROUP BY R.RoleId, R.RoleName
```

B
```
SELECT R.RoleName, U.ActiveUserCount FROM tblRoles R
INNER JOIN (SELECT RoleId, COUNT(*) AS ActiveUserCount FROM tblUsers WHERE IsActive = 1
GROUP BY RoleId) U ON R.RoleId = U.RoleId
```

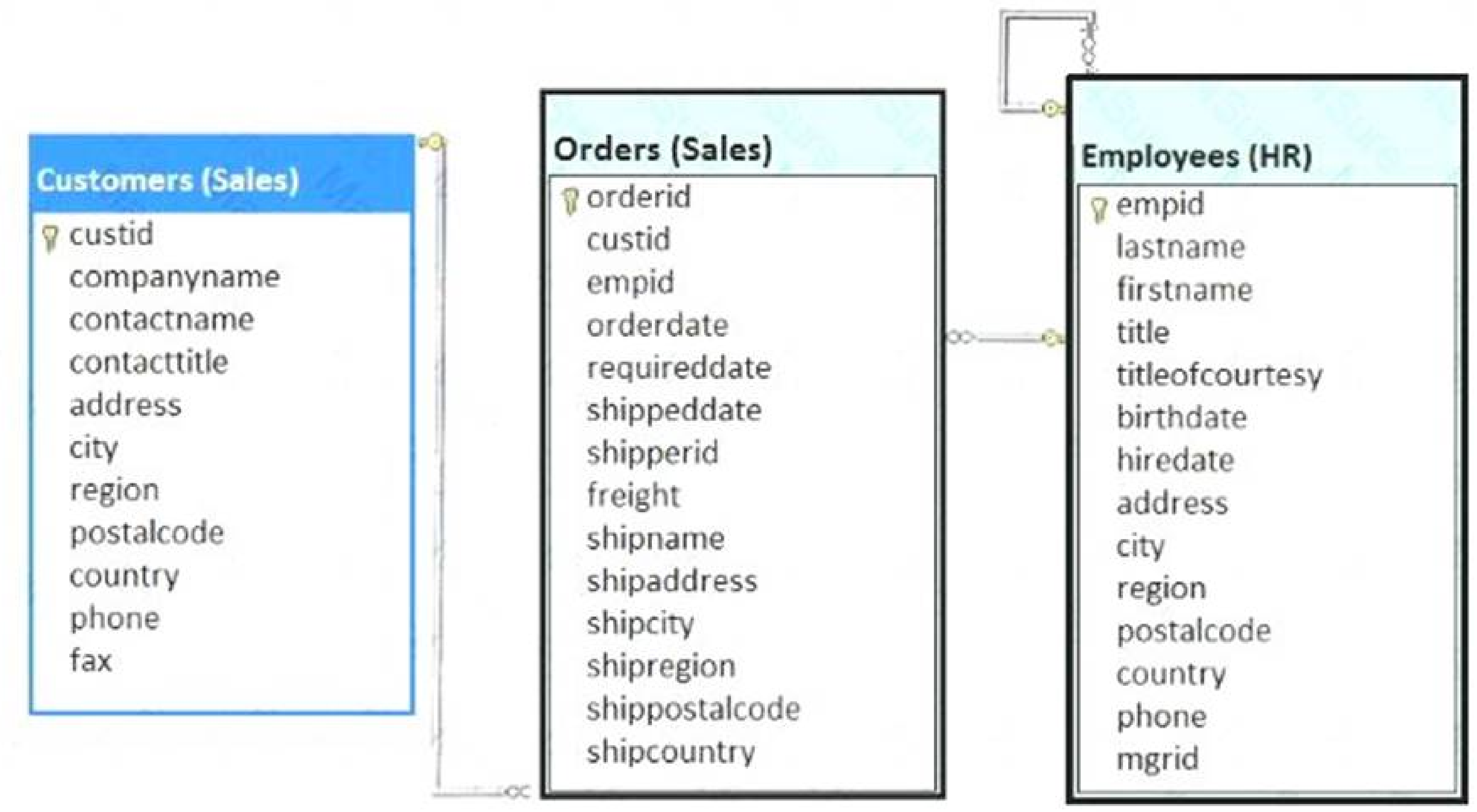
C
```
SELECT R.RoleName, COUNT(*) AS ActiveUserCount FROM tblRoles R
LEFT JOIN (SELECT UserId, RoleId FROM tblUsers WHERE IsActive = 1)U ON U.RoleId = R.RoleId
GROUP BY R.RoleId, R.RoleName
```

D
```
SELECT R.RoleName, U.ActiveUserCount FROM tblRoles R CROSS JOIN
(SELECT COUNT(*) AS ActiveUserCount FROM tblUsers WHERE IsActive = 1) U
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: C

NEW QUESTION 90
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a database that includes the tables shown in the exhibit (Click the Exhibit button.)



You need to create a Transact-SQL query that returns the following information:
- the customer number
- the customer contact name
- the date the order was placed, with a name of DateofOrder
- a column named Salesperson, formatted with the employee first name, a space, and the employee last name
- orders for customers where the employee identifier equals 4

The output must be sorted by order date, with the newest orders first. The solution must return only the most recent order for each customer. Solution: You run the following Transact-SQL statement:

```
SELECT c.custid, contactname, MAX(orderdate) AS DateofOrder,
e.firstname + ' ' + e.lastname AS Salesperson
FROM Sales.Customers AS c
INNER JOIN Sales.Orders AS o ON c.custid = o.custid
INNER JOIN HR.Employees AS e ON o.empid = e.empid
WHERE o.empid = 4
GROUP BY c.custid, contactname, firstname, lastname
ORDER BY DateofOrder DESC
```

Does the solution meet the goal?

A. Yes
B. No

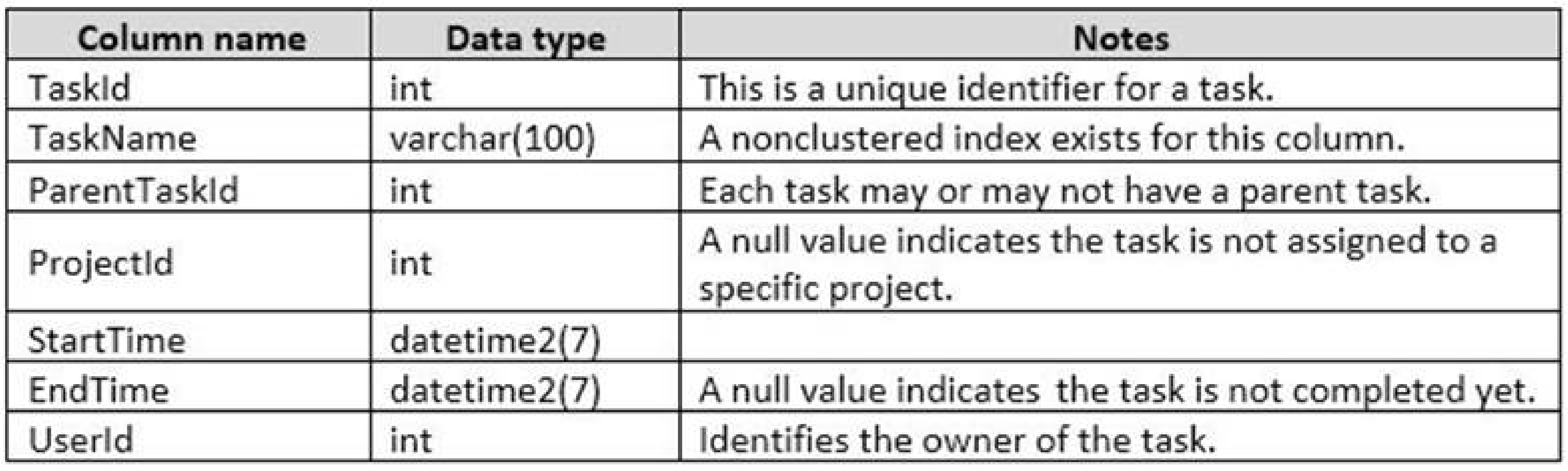
Answer: A

Explanation:
The MAX(orderdate) in the SELECT statement makes sure we return only the most recent order. A WHERE o.empiD =4 clause is correctly used.
GROUP BY is also required.

NEW QUESTION 92
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 query a database that includes two tables: Project and Task. The Project table includes the following columns:

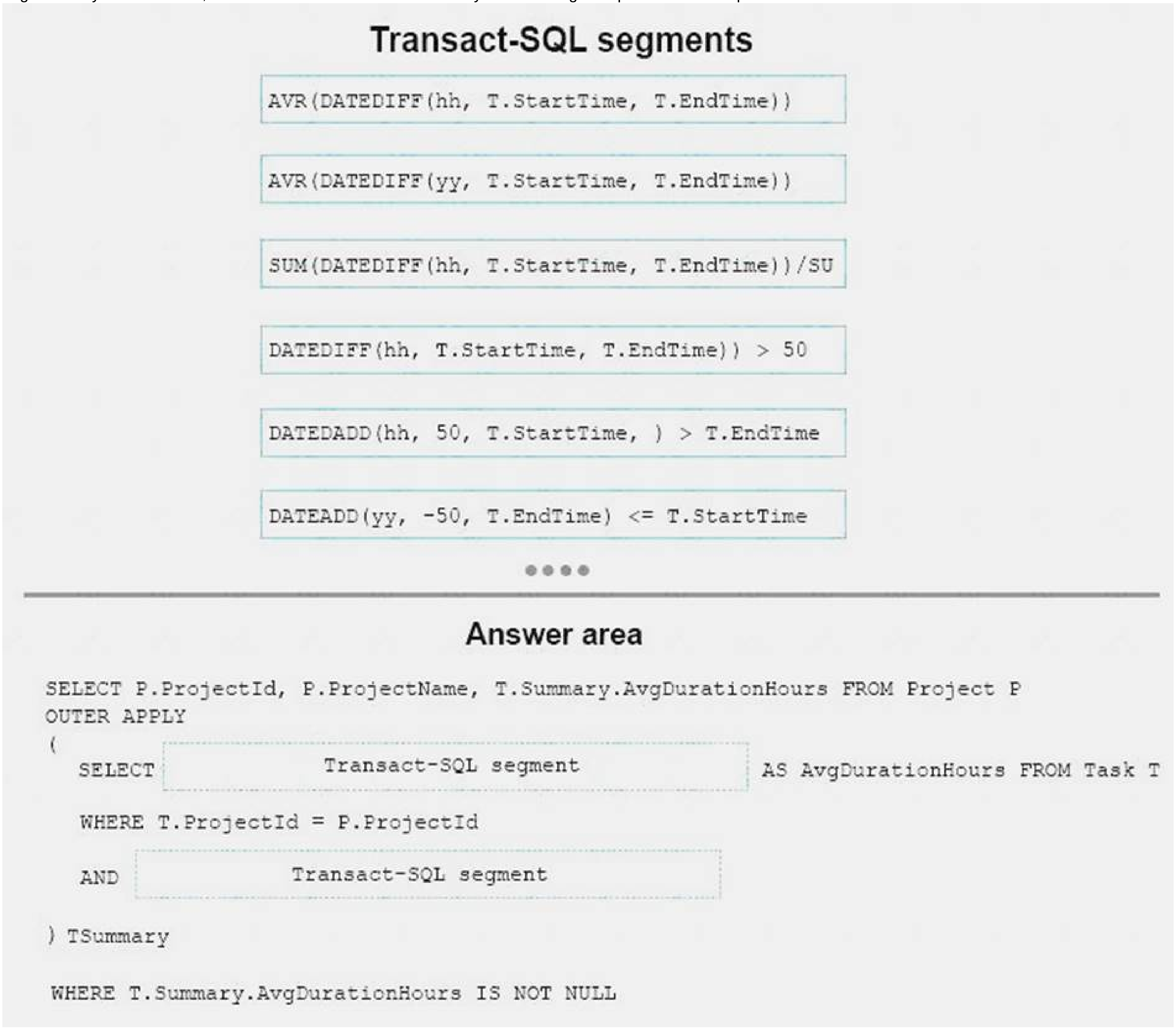
Column name	Data type	Notes
ProjectId	int	This is a unique identifier for a project.
ProjectName	varchar(100)	
StartTime	datetime2(7)	
EndTime	datetime2(7)	A null value indicates the project is not finished yet.
UserId	int	Identifies the owner of the project.

The Task table includes the following columns:



Column name	Data type	Notes
TaskId	int	This is a unique identifier for a task.
TaskName	varchar(100)	A nonclustered index exists for this column.
ParentTaskId	int	Each task may or may not have a parent task.
ProjectId	int	A null value indicates the task is not assigned to a specific project.
StartTime	datetime2(7)	
EndTime	datetime2(7)	A null value indicates the task is not completed yet.
UserId	int	Identifies the owner of the task.

You need to find all projects that have at least one task that took more than 50 hours to complete. You must also determine the average duration of the tasks that took more that took more than 50 hours to complete for each project.
How should you complete the Transact-SQL statement? 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.



Transact-SQL segments

```
AVR(DATEDIFF(hh, T.StartTime, T.EndTime))

AVR(DATEDIFF(yy, T.StartTime, T.EndTime))

SUM(DATEDIFF(hh, T.StartTime, T.EndTime))/SU

DATEDIFF(hh, T.StartTime, T.EndTime)) > 50

DATEDADD(hh, 50, T.StartTime, ) > T.EndTime

DATEADD(yy, -50, T.EndTime) <= T.StartTime
```

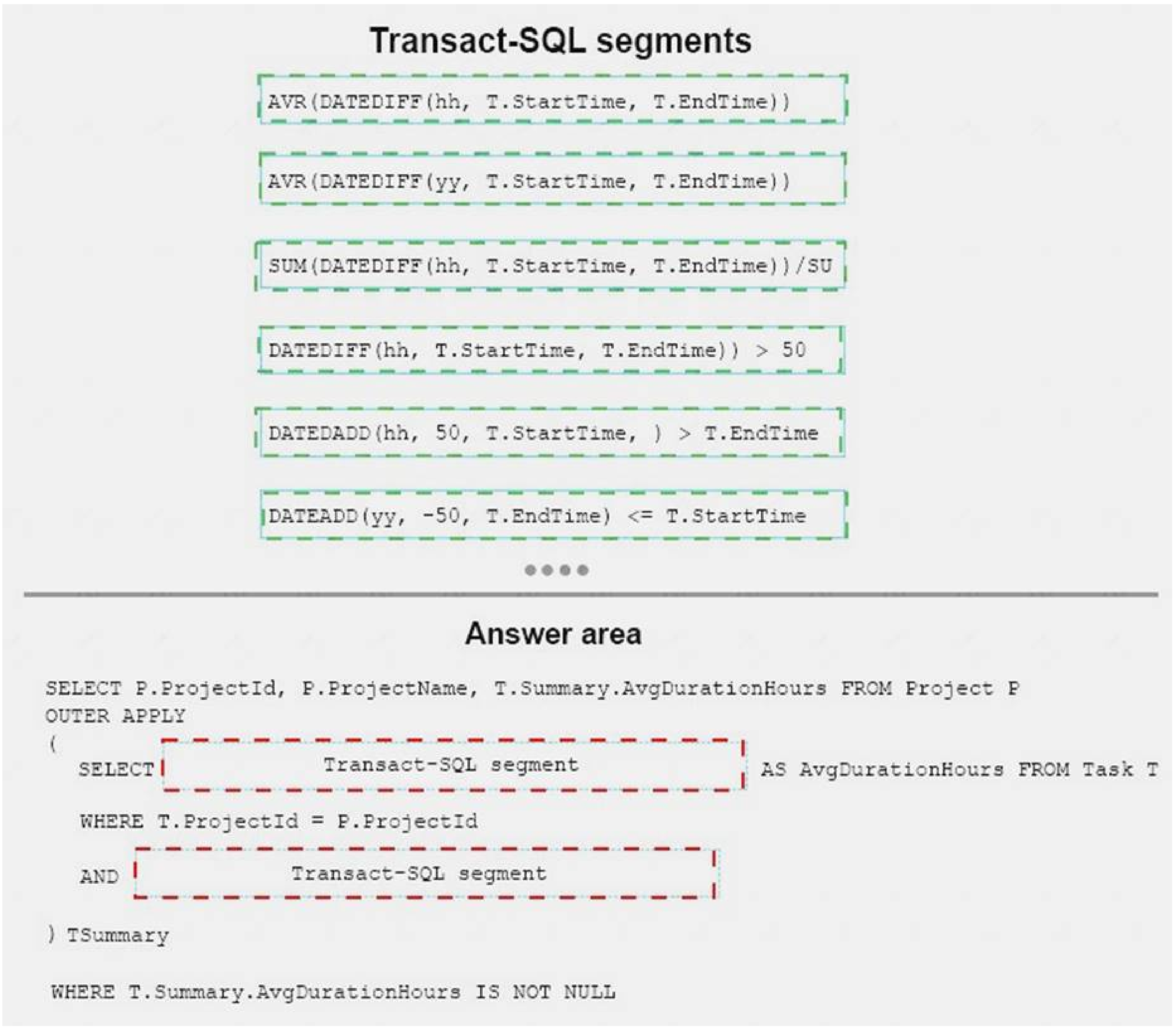
Answer area

```
SELECT P.ProjectId, P.ProjectName, T.Summary.AvgDurationHours FROM Project P
OUTER APPLY
(
   SELECT      Transact-SQL segment       AS AvgDurationHours FROM Task T

   WHERE T.ProjectId = P.ProjectId

   AND      Transact-SQL segment

) TSummary

 WHERE T.Summary.AvgDurationHours IS NOT NULL
```

A. Mastered
B. Not Mastered

Answer: A

Explanation:



Transact-SQL segments

```
AVR(DATEDIFF(hh, T.StartTime, T.EndTime))
AVR(DATEDIFF(yy, T.StartTime, T.EndTime))
SUM(DATEDIFF(hh, T.StartTime, T.EndTime))/SU
DATEDIFF(hh, T.StartTime, T.EndTime)) > 50
DATEDADD(hh, 50, T.StartTime, ) > T.EndTime
DATEADD(yy, -50, T.EndTime) <= T.StartTime
```

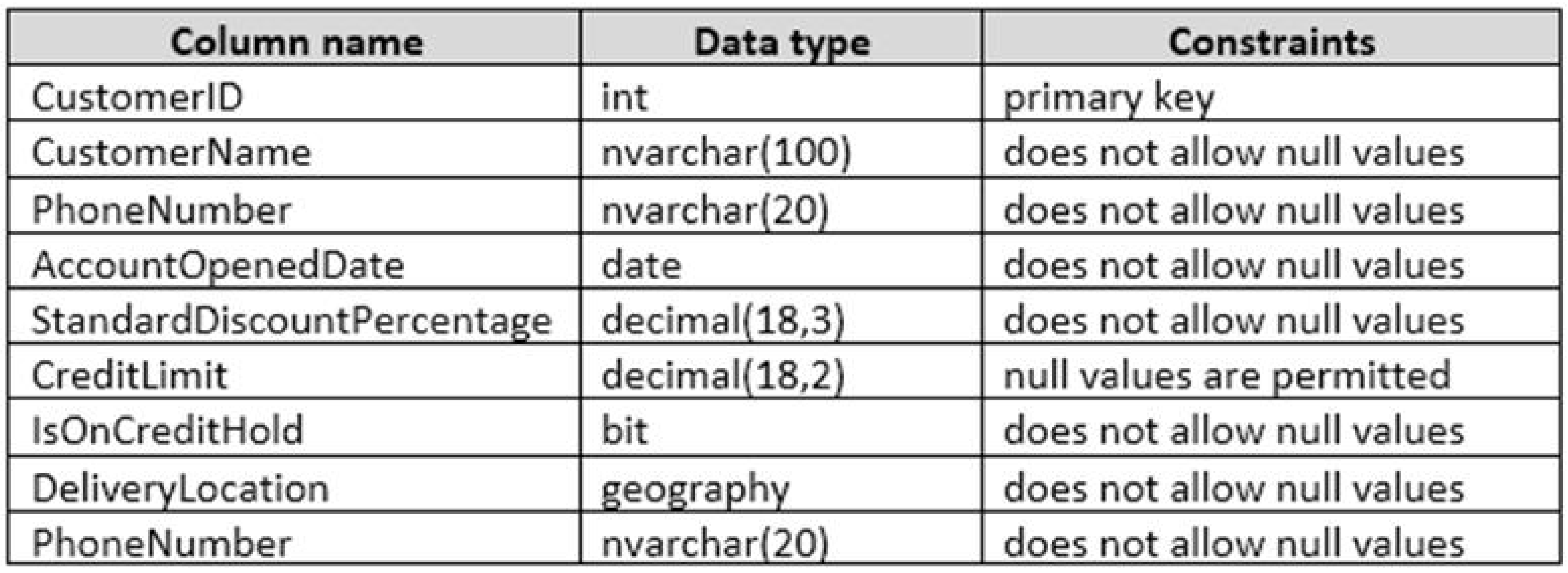
Answer area

```
SELECT P.ProjectId, P.ProjectName, T.Summary.AvgDurationHours FROM Project P
OUTER APPLY
(
  SELECT [Transact-SQL segment] AS AvgDurationHours FROM Task T

  WHERE T.ProjectId = P.ProjectId

  AND [Transact-SQL segment]

) TSummary

WHERE T.Summary.AvgDurationHours IS NOT NULL
```

NEW QUESTION 95

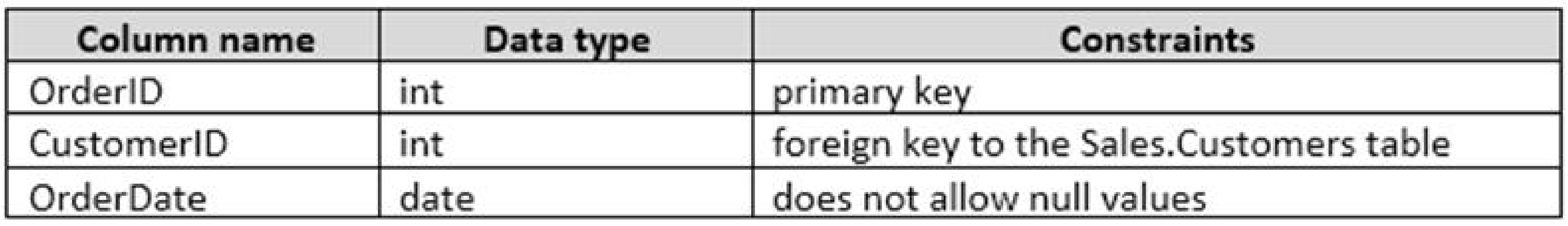
Note: This question is part of a series of questions that use the same scenario. For your convenience, the scenario is repeated in each question. Each question presents a different goal and answer choices, but the text of the scenario is exactly the same in each question in this series.

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.



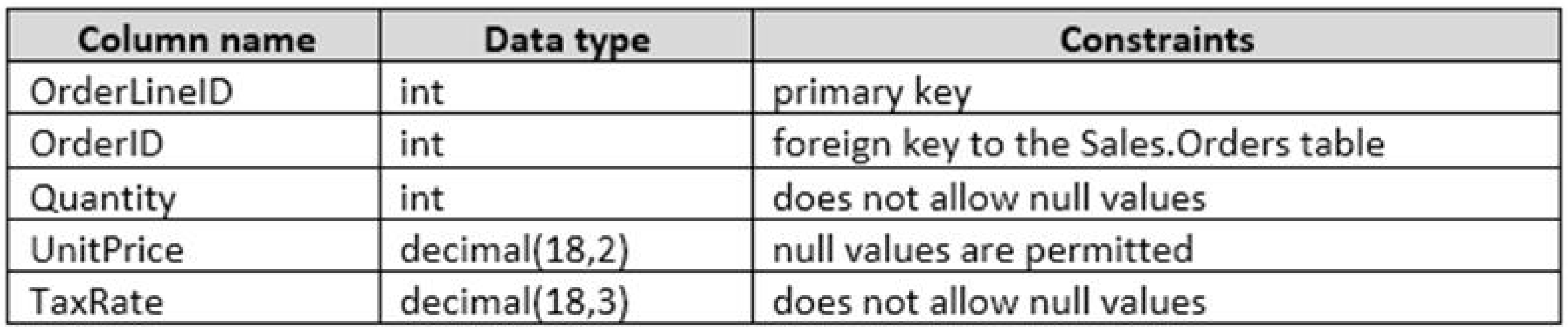
Column name	Data type	Constraints
CustomerID	int	primary key
CustomerName	nvarchar(100)	does not allow null values
PhoneNumber	nvarchar(20)	does not allow null values
AccountOpenedDate	date	does not allow null values
StandardDiscountPercentage	decimal(18,3)	does not allow null values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	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.



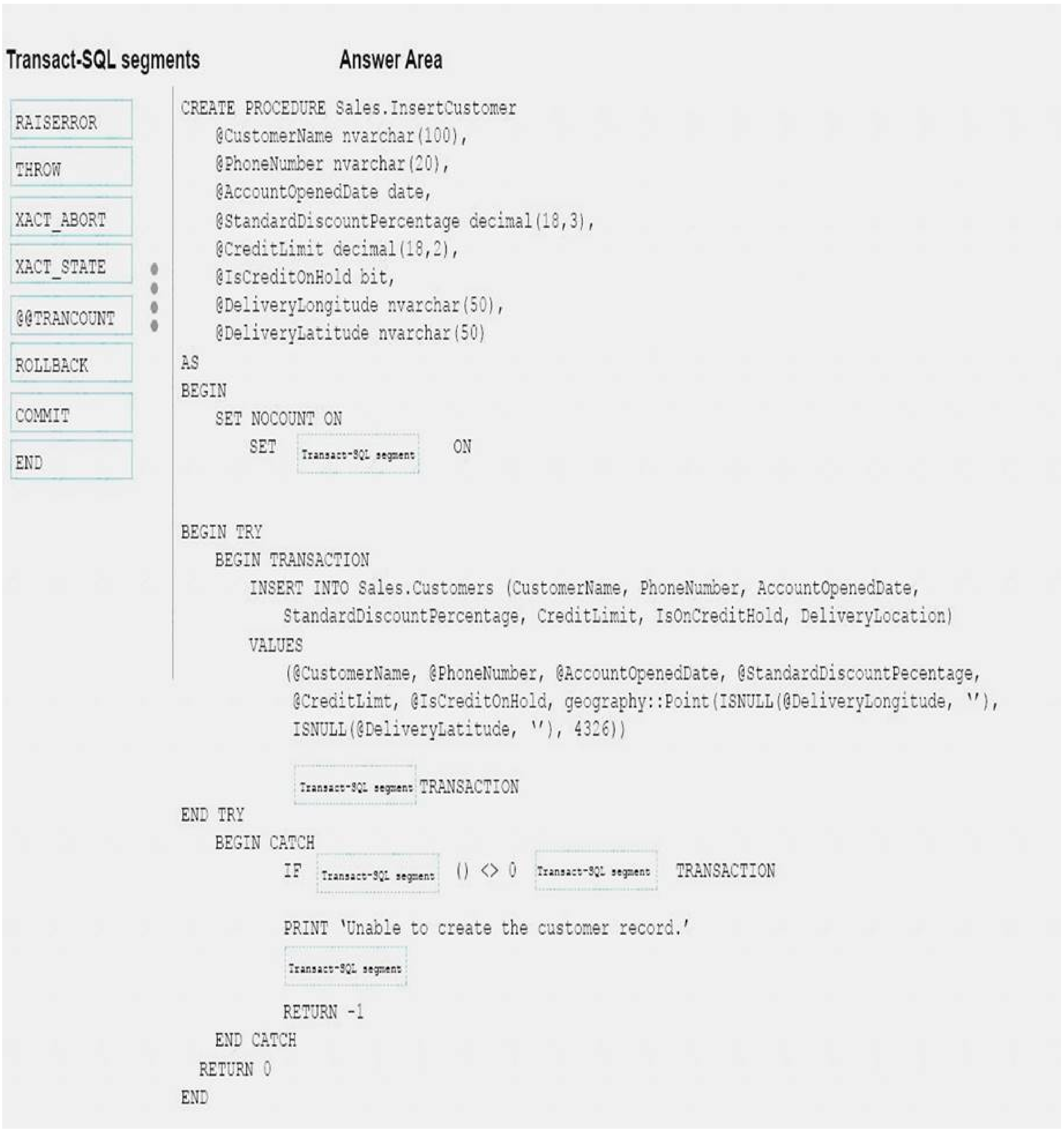
Column name	Data type	Constraints
OrderID	int	primary key
CustomerID	int	foreign key to the Sales.Customers table
OrderDate	date	does not allow null values

The following table describes the columns in Sales.OrderLines.



Column name	Data type	Constraints
OrderLineID	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 stored procedure that inserts data into the Customers table. The stored procedure must meet the following requirements:
- Data changes occur as a single unit of work.
- Data modifications that are successful are committed and a value of 0 is returned.
- Data modifications that are unsuccessful are rolled back. The exception severity level is set to 16 and a value of -1 is returned.
- The stored procedure uses a built-it scalar function to evaluate the current condition of data modifications.
- The entire unit of work is terminated and rolled back if a run-time error occurs during execution of the stored procedure.
How should complete the stored procedure definition? To answer, drag the appropriate Transact-SQL segments to the correct targets. 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.



Transact-SQL segments

RAISERROR
THROW
XACT_ABORT
XACT_STATE
@@TRANCOUNT
ROLLBACK
COMMIT
END

Answer Area

```
CREATE PROCEDURE Sales.InsertCustomer
    @CustomerName nvarchar(100),
    @PhoneNumber nvarchar(20),
    @AccountOpenedDate date,
    @StandardDiscountPercentage decimal(18,3),
    @CreditLimit decimal(18,2),
    @IsCreditOnHold bit,
    @DeliveryLongitude nvarchar(50),
    @DeliveryLatitude nvarchar(50)
AS
BEGIN
    SET NOCOUNT ON
        SET  Transact-SQL segment  ON


BEGIN TRY
    BEGIN TRANSACTION
        INSERT INTO Sales.Customers (CustomerName, PhoneNumber, AccountOpenedDate,
            StandardDiscountPercentage, CreditLimit, IsOnCreditHold, DeliveryLocation)
        VALUES
            (@CustomerName, @PhoneNumber, @AccountOpenedDate, @StandardDiscountPecentage,
             @CreditLimt, @IsCreditOnHold, geography::Point(ISNULL(@DeliveryLongitude, ''),
             ISNULL(@DeliveryLatitude, ''), 4326))

             Transact-SQL segment TRANSACTION
END TRY
    BEGIN CATCH
            IF  Transact-SQL segment  () <> 0  Transact-SQL segment  TRANSACTION

            PRINT 'Unable to create the customer record.'
            Transact-SQL segment

            RETURN -1
    END CATCH
  RETURN 0
END
```

A. Mastered
B. Not Mastered

Answer: A

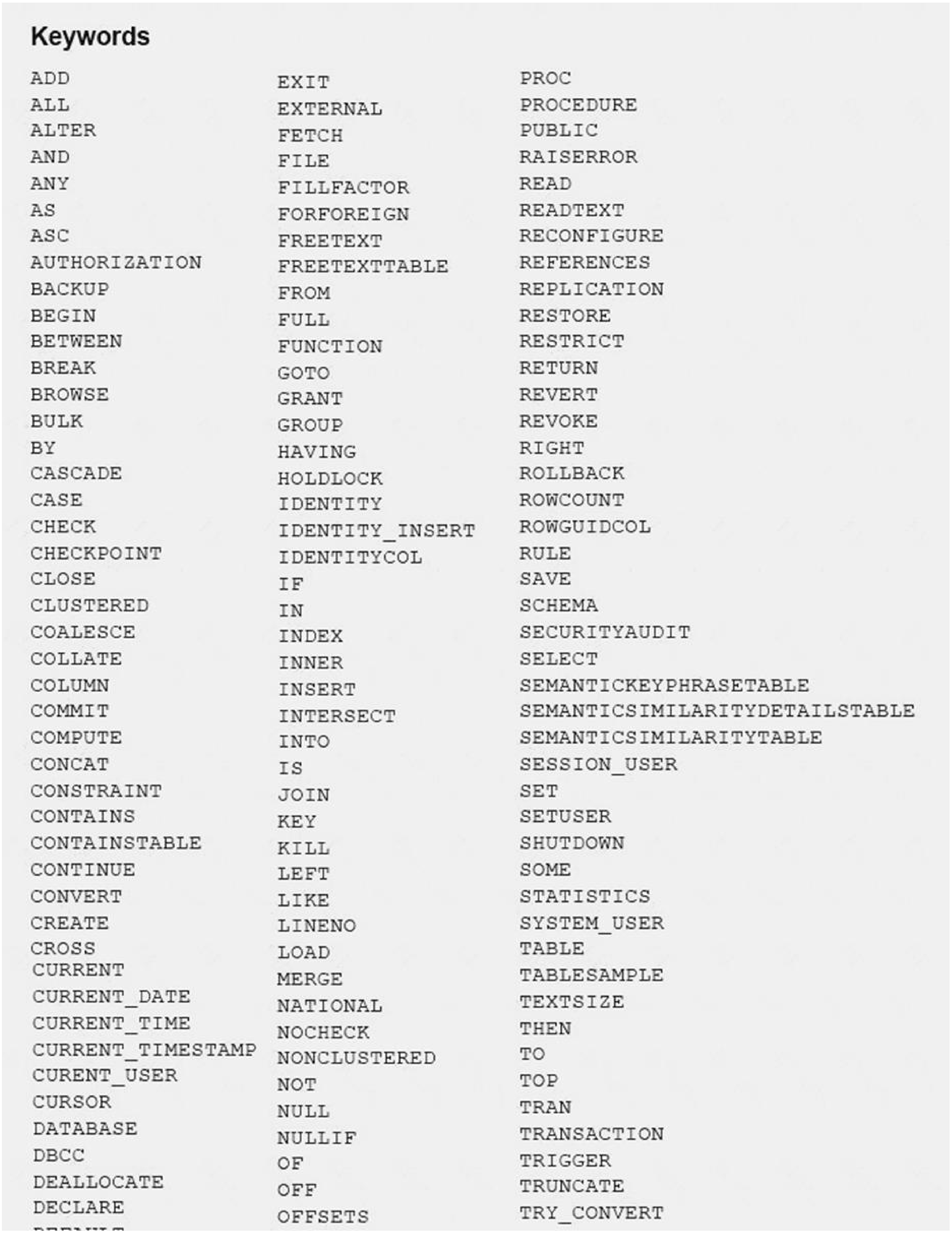
Explanation:
Box 1: XACT_ABORT
XACT_ABORT specifies whether SQL Server automatically rolls back the current transaction when a Transact-SQL statement raises a run-time error.
When SET XACT_ABORT is ON, if a Transact-SQL statement raises a run-time error, the entire transaction is terminated and rolled back.
Box 2: COMMIT
Commit the transaction. Box 3: XACT_STATE
Box 4: ROLLBACK
Rollback the transaction Box 5: THROW
THROW raises an exception and the severity is set to 16.
Requirement: Data modifications that are unsuccessful are rolled back. The exception severity level is set to 16 and a value of -1 is returned.
References:
https://msdn.microsoft.com/en-us/library/ms188792.aspx https://msdn.microsoft.com/en-us/library/ee677615.aspx

NEW QUESTION 98
You work for an organization that monitors seismic activity around volcanos. You have a table named GroundSensors. The table stored data collected from

seismic sensors. It includes the columns describes in the following table:

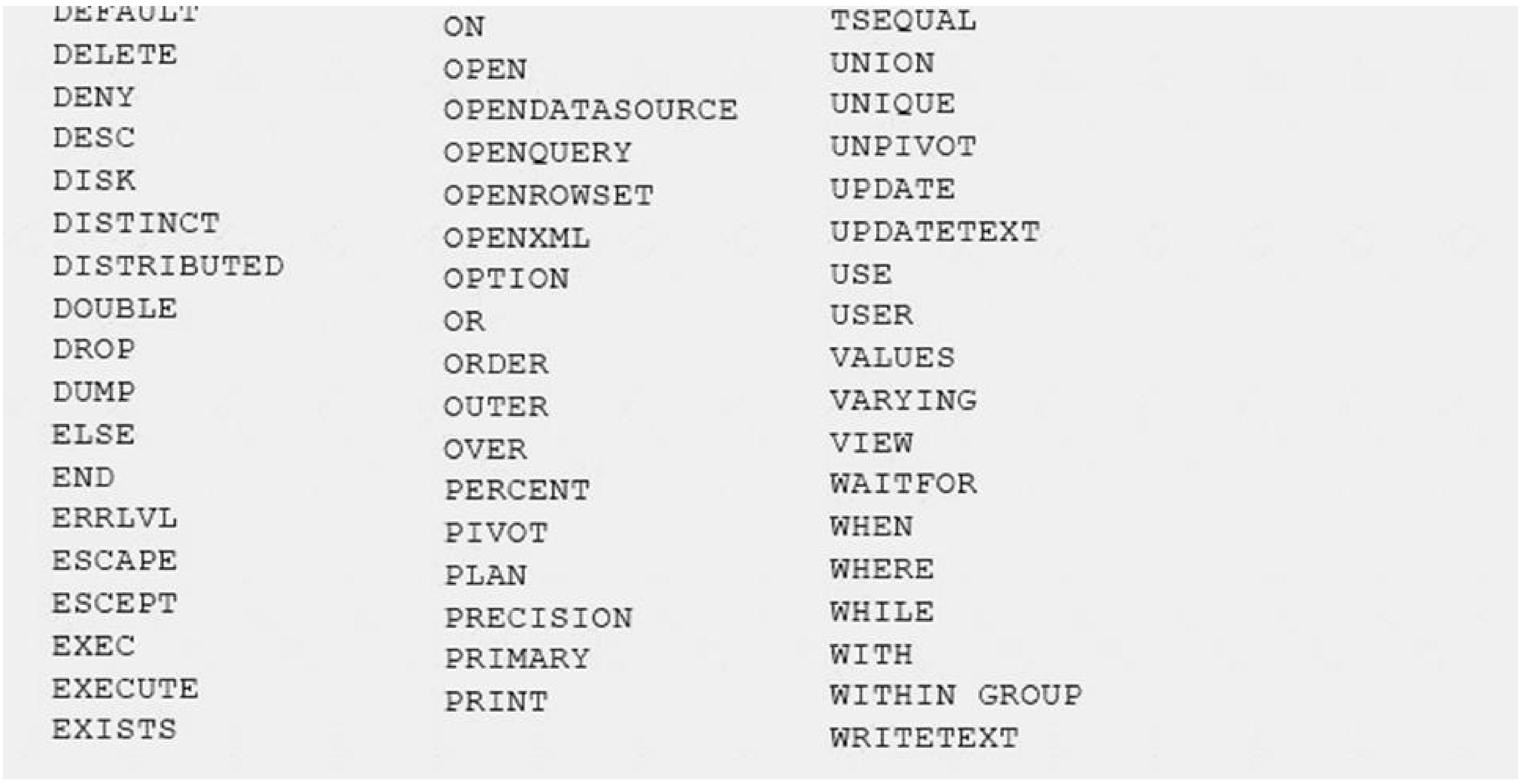
Name	Data Type	Notes
SensorID	int	primary key
Location	geography	do not allow null values
Tremor	int	do not allow null values
NormalizedReading	float	allow null values

The database also contains a scalar value function named NearestMountain that returns the name of the mountain that is nearest to the sensor.
You need to create a query that shows the average of the normalized readings from the sensors for each mountain. The query must meet the following requirements:
- Include the average normalized readings and nearest mountain name.
- Exclude sensors for which no normalized reading exists.
- Exclude those sensors with value of zero for tremor. Construct the query using the following guidelines:
- Use one part names to reference tables, columns and functions.
- Do not use parentheses unless required.
- Do not use aliases for column names and table names.
- Do not surround object names with square brackets.



Keywords

ADD	EXIT	PROC
ALL	EXTERNAL	PROCEDURE
ALTER	FETCH	PUBLIC
AND	FILE	RAISERROR
ANY	FILLFACTOR	READ
AS	FORFOREIGN	READTEXT
ASC	FREETEXT	RECONFIGURE
AUTHORIZATION	FREETEXTTABLE	REFERENCES
BACKUP	FROM	REPLICATION
BEGIN	FULL	RESTORE
BETWEEN	FUNCTION	RESTRICT
BREAK	GOTO	RETURN
BROWSE	GRANT	REVERT
BULK	GROUP	REVOKE
BY	HAVING	RIGHT
CASCADE	HOLDLOCK	ROLLBACK
CASE	IDENTITY	ROWCOUNT
CHECK	IDENTITY_INSERT	ROWGUIDCOL
CHECKPOINT	IDENTITYCOL	RULE
CLOSE	IF	SAVE
CLUSTERED	IN	SCHEMA
COALESCE	INDEX	SECURITYAUDIT
COLLATE	INNER	SELECT
COLUMN	INSERT	SEMANTICKEYPHRASETABLE
COMMIT	INTERSECT	SEMANTICSIMILARITYDETAILSTABLE
COMPUTE	INTO	SEMANTICSIMILARITYTABLE
CONCAT	IS	SESSION_USER
CONSTRAINT	JOIN	SET
CONTAINS	KEY	SETUSER
CONTAINSTABLE	KILL	SHUTDOWN
CONTINUE	LEFT	SOME
CONVERT	LIKE	STATISTICS
CREATE	LINENO	SYSTEM_USER
CROSS	LOAD	TABLE
CURRENT	MERGE	TABLESAMPLE
CURRENT_DATE	NATIONAL	TEXTSIZE
CURRENT_TIME	NOCHECK	THEN
CURRENT_TIMESTAMP	NONCLUSTERED	TO
CURENT_USER	NOT	TOP
CURSOR	NULL	TRAN
DATABASE	NULLIF	TRANSACTION
DBCC	OF	TRIGGER
DEALLOCATE	OFF	TRUNCATE
DECLARE	OFFSETS	TRY_CONVERT



DEFAULT	ON	TSEQUAL
DELETE	OPEN	UNION
DENY	OPENDATASOURCE	UNIQUE
DESC	OPENQUERY	UNPIVOT
DISK	OPENROWSET	UPDATE
DISTINCT	OPENXML	UPDATETEXT
DISTRIBUTED	OPTION	USE
DOUBLE	OR	USER
DROP	ORDER	VALUES
DUMP	OUTER	VARYING
ELSE	OVER	VIEW
END	PERCENT	WAITFOR
ERRLVL	PIVOT	WHEN
ESCAPE	PLAN	WHERE
ESCEPT	PRECISION	WHILE
EXEC	PRIMARY	WITH
EXECUTE	PRINT	WITHIN GROUP
EXISTS		WRITETEXT

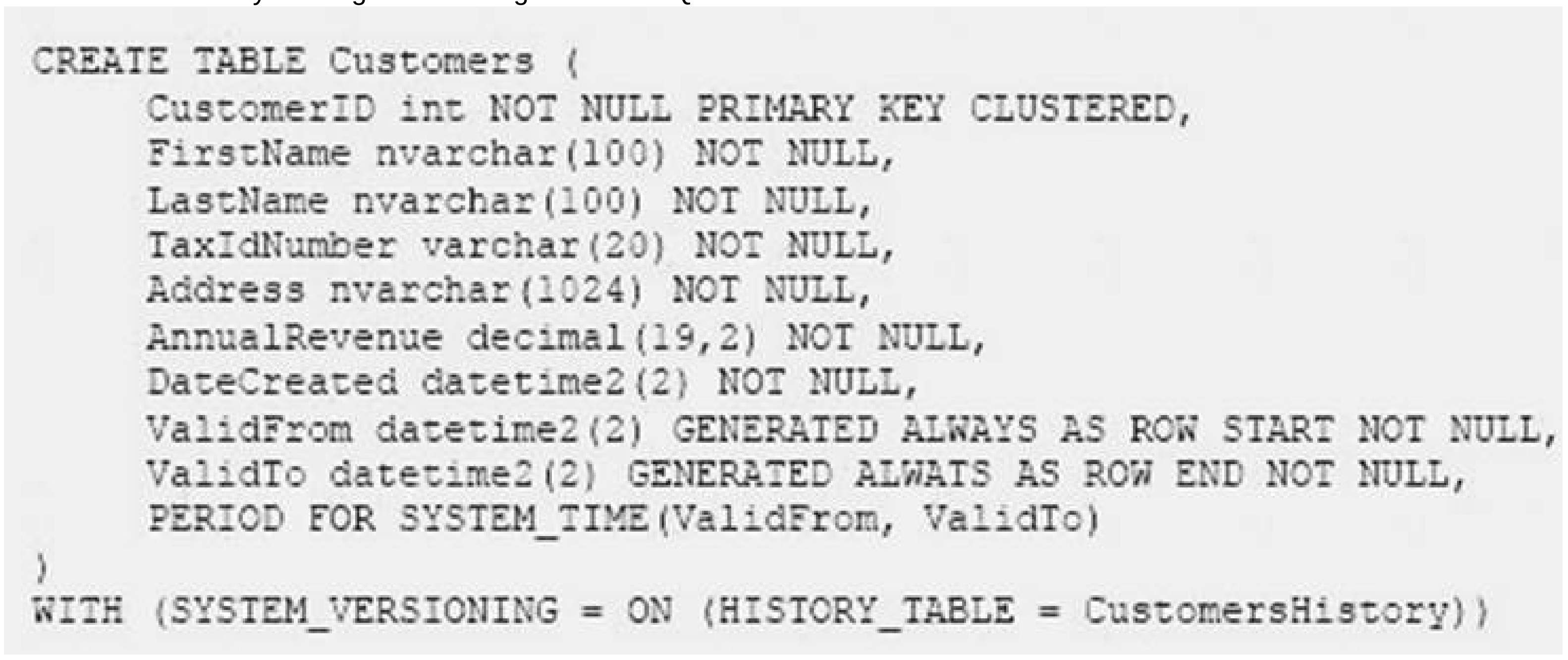
Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the code that has been provided as well as below it.
1. SELECT
2. FROM Sales.Products AS P
Use the Check Syntax button to verify your work. Any syntax or spelling errors will be reported by line and character position. You may check syntax as many times as needed.

A. Mastered
B. Not Mastered

Answer: A

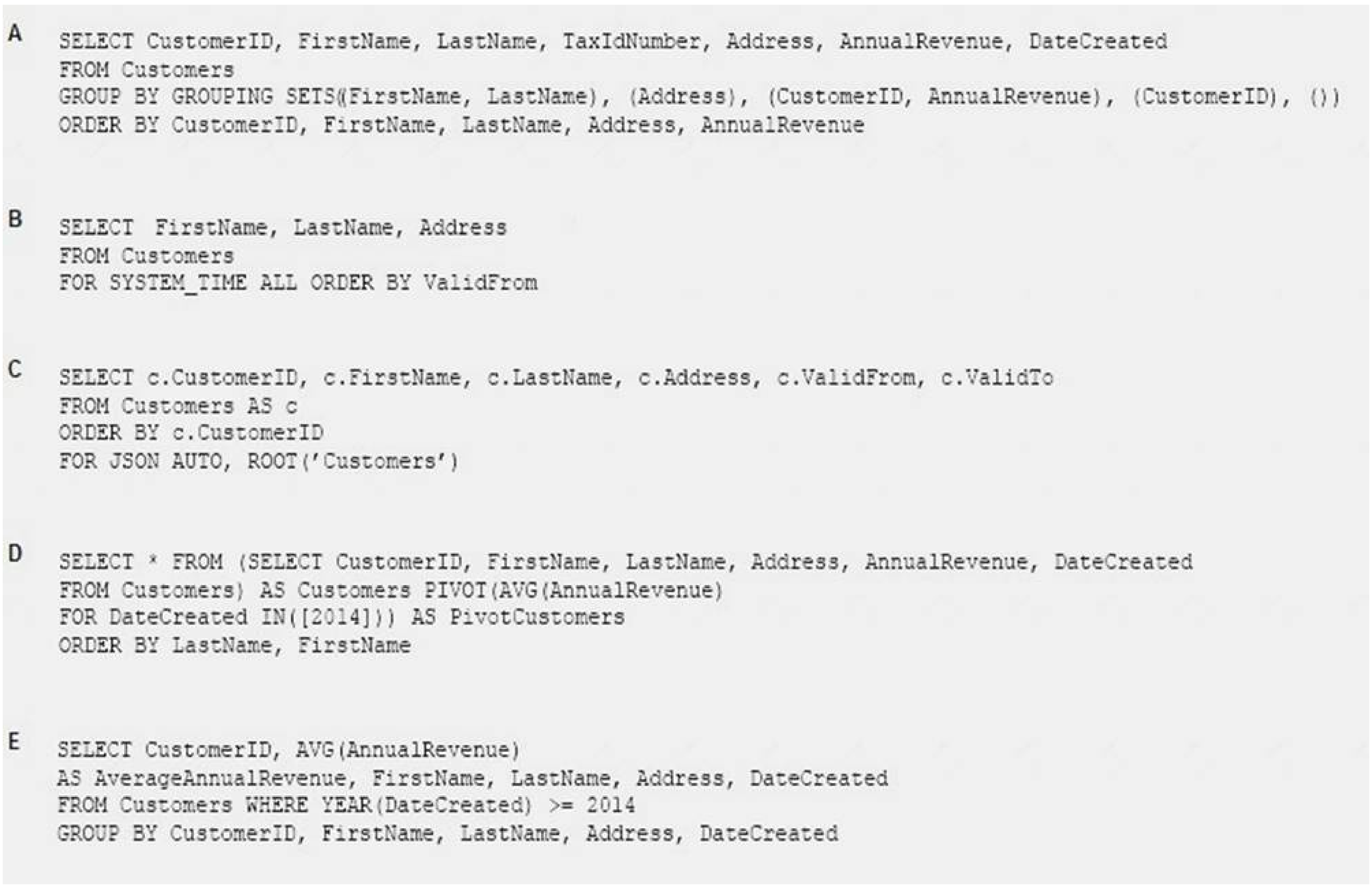
Explanation:
1. SELECT avg(P.ProductPrice) AS Average, min(P.ProductsInStock) AS LowestNumber, max(P.ProductPrice) AS HighestPrice
2. FROM Sales.Products AS P Make the additions to line 1.
References: https://www.mssqltips.com/sqlservertip/4424/max-min-and-avg-sql-server-functions/

NEW QUESTION 103
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You create a table by running the following Transact-SQL statement:



```
CREATE TABLE Customers (
     CustomerID int NOT NULL PRIMARY KEY CLUSTERED,
     FirstName nvarchar(100) NOT NULL,
     LastName nvarchar(100) NOT NULL,
     TaxIdNumber varchar(20) NOT NULL,
     Address nvarchar(1024) NOT NULL,
     AnnualRevenue decimal(19,2) NOT NULL,
     DateCreated datetime2(2) NOT NULL,
     ValidFrom datetime2(2) GENERATED ALWAYS AS ROW START NOT NULL,
     ValidTo datetime2(2) GENERATED ALWATS AS ROW END NOT NULL,
     PERIOD FOR SYSTEM_TIME(ValidFrom, ValidTo)
)
WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE = CustomersHistory))
```

You are developing a report that displays customer information. The report must contain a grand total column. You need to write a query that returns the data for the report.
Which Transact-SQL statement should you run?



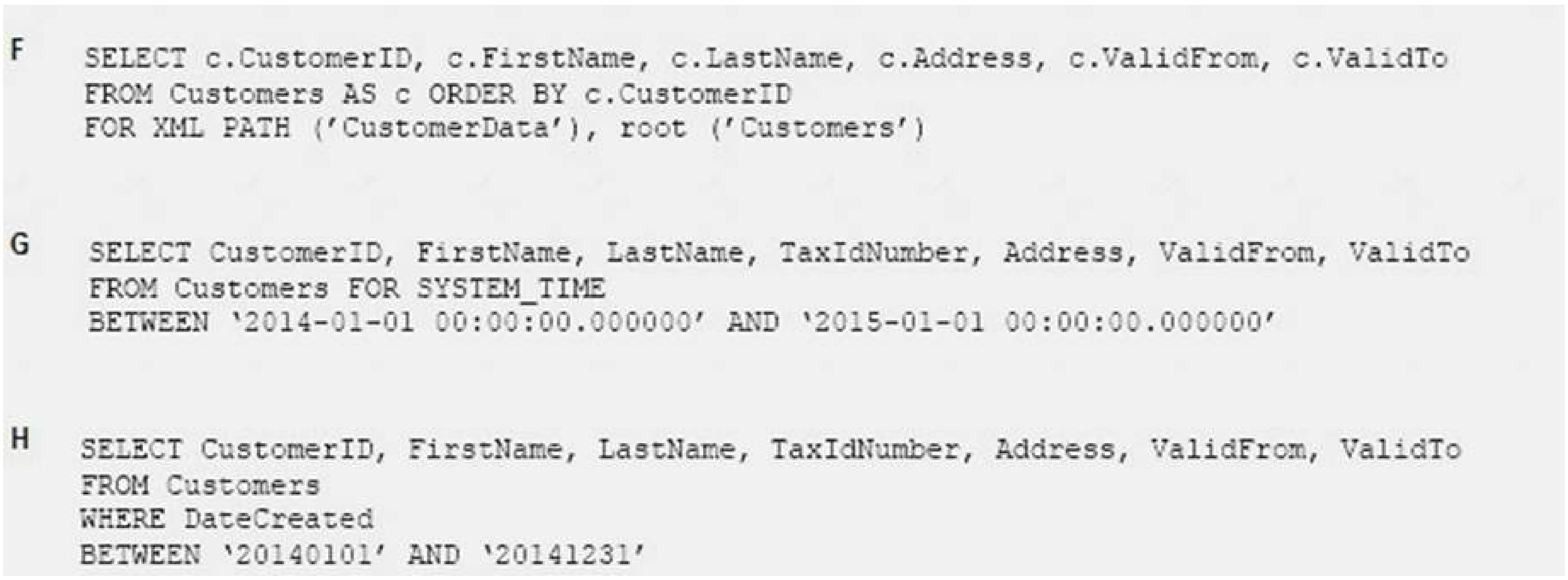
A
```
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, AnnualRevenue, DateCreated
FROM Customers
GROUP BY GROUPING SETS((FirstName, LastName), (Address), (CustomerID, AnnualRevenue), (CustomerID), ())
ORDER BY CustomerID, FirstName, LastName, Address, AnnualRevenue
```

B
```
SELECT  FirstName, LastName, Address
FROM Customers
FOR SYSTEM_TIME ALL ORDER BY ValidFrom
```

C
```
SELECT c.CustomerID, c.FirstName, c.LastName, c.Address, c.ValidFrom, c.ValidTo
FROM Customers AS c
ORDER BY c.CustomerID
FOR JSON AUTO, ROOT('Customers')
```

D
```
SELECT * FROM (SELECT CustomerID, FirstName, LastName, Address, AnnualRevenue, DateCreated
FROM Customers) AS Customers PIVOT(AVG(AnnualRevenue)
FOR DateCreated IN([2014])) AS PivotCustomers
ORDER BY LastName, FirstName
```

E
```
SELECT CustomerID, AVG(AnnualRevenue)
AS AverageAnnualRevenue, FirstName, LastName, Address, DateCreated
FROM Customers WHERE YEAR(DateCreated) >= 2014
GROUP BY CustomerID, FirstName, LastName, Address, DateCreated
```



F
```
SELECT c.CustomerID, c.FirstName, c.LastName, c.Address, c.ValidFrom, c.ValidTo
FROM Customers AS c ORDER BY c.CustomerID
FOR XML PATH ('CustomerData'), root ('Customers')
```

G
```
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, ValidFrom, ValidTo
FROM Customers FOR SYSTEM_TIME
BETWEEN '2014-01-01 00:00:00.000000' AND '2015-01-01 00:00:00.000000'
```

H
```
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, ValidFrom, ValidTo
FROM Customers
WHERE DateCreated
BETWEEN '20140101' AND '20141231'
```

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E
F. Option F
G. Option G
H. Option H

Answer: E

Explanation:
Calculate aggregate column through AVG function and GROUP BY clause.

NEW QUESTION 106
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a database that tracks orders and deliveries for customers in North America. The database contains the following tables:
Sales.Customers

Column	Data type	Notes
CustomerID	int	primary key
CustomerCategoryID	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow new values
StandardDiscountPercentage	int	does not allow new values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow new values
DeliveryLocation	geography	does not allow new values
PhoneNumber	nvarchar(20)	does not allow new values data is formatted as follows: 425-555-0187

Application.Cities

Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Sales.CustomerCategories

Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

The company's development team is designing a customer directory application. The application must list customers by the area code of their phone number. The area code is defined as the first three characters of the phone number.
The main page of the application will be based on an indexed view that contains the area and phone number for all customers.
You need to return the area code from the PhoneNumber field. Solution: You run the following Transact-SQL statement:

```
CREATE FUNCTION AreaCode (
     @phoneNumber nvarchar(20)
)
RETURNS nvarchar(10)
WITH SCHEMABINDING
AS
BEGIN
     DECLARE @areaCode nvarchar(max)
     SELECT @areaCode = value FROM STRING_SPLIT(@phoneNumber, '-')
     RETURN @areaCode
END
```

Does the solution meet the goal?

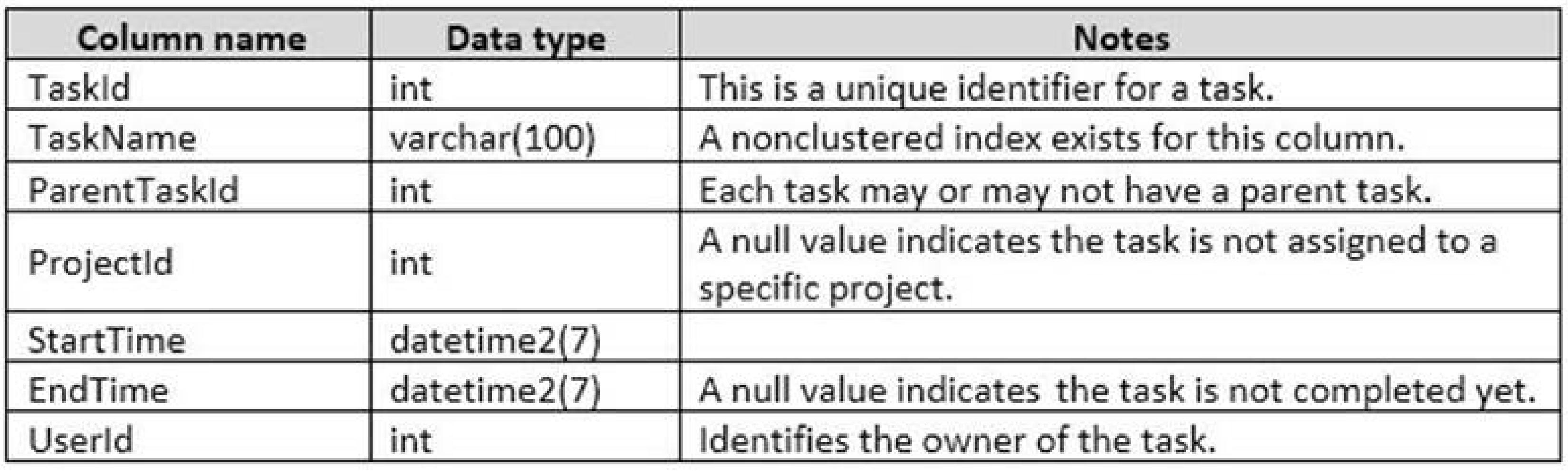
A. Yes
B. No

Answer: B

Explanation:
The variable max, in the line DECLARE @areaCode nvarchar(max), is not defined.

NEW QUESTION 109
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 query a database that includes two tables: Project and Task. The Project table includes the following columns:

Column name	Data type	Notes
ProjectId	int	This is a unique identifier for a project.
ProjectName	varchar(100)	
StartTime	datetime2(7)	
EndTime	datetime2(7)	A null value indicates the project is not finished yet.
UserId	int	Identifies the owner of the project.

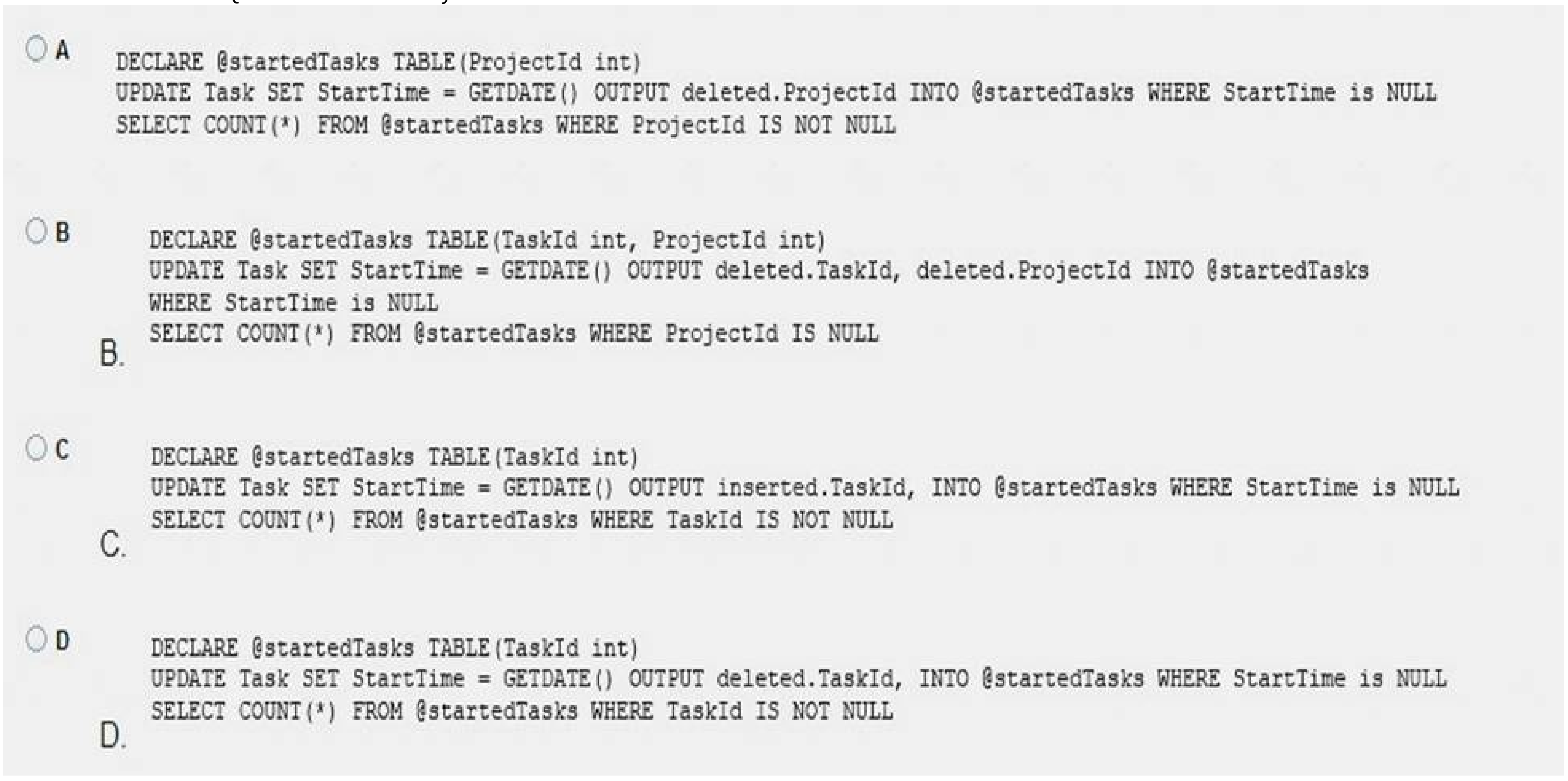


Column name	Data type	Notes
TaskId	int	This is a unique identifier for a task.
TaskName	varchar(100)	A nonclustered index exists for this column.
ParentTaskId	int	Each task may or may not have a parent task.
ProjectId	int	A null value indicates the task is not assigned to a specific project.
StartTime	datetime2(7)	
EndTime	datetime2(7)	A null value indicates the task is not completed yet.
UserId	int	Identifies the owner of the task.

You plan to run the following query to update tasks that are not yet started:

```
UPDATE Task SET StartTime = GETDATE() WHERE StartTime IS NULL
```

You need to return the total count of tasks that are impacted by this UPDATE operation, but are not associated with a project.
What set of Transact-SQL statements should you run?



A

```
DECLARE @startedTasks TABLE(ProjectId int)
UPDATE Task SET StartTime = GETDATE() OUTPUT deleted.ProjectId INTO @startedTasks WHERE StartTime is NULL
SELECT COUNT(*) FROM @startedTasks WHERE ProjectId IS NOT NULL
```

B

B.

```
DECLARE @startedTasks TABLE(TaskId int, ProjectId int)
UPDATE Task SET StartTime = GETDATE() OUTPUT deleted.TaskId, deleted.ProjectId INTO @startedTasks
WHERE StartTime is NULL
SELECT COUNT(*) FROM @startedTasks WHERE ProjectId IS NULL
```

C

C.

```
DECLARE @startedTasks TABLE(TaskId int)
UPDATE Task SET StartTime = GETDATE() OUTPUT inserted.TaskId, INTO @startedTasks WHERE StartTime is NULL
SELECT COUNT(*) FROM @startedTasks WHERE TaskId IS NOT NULL
```

D

D.

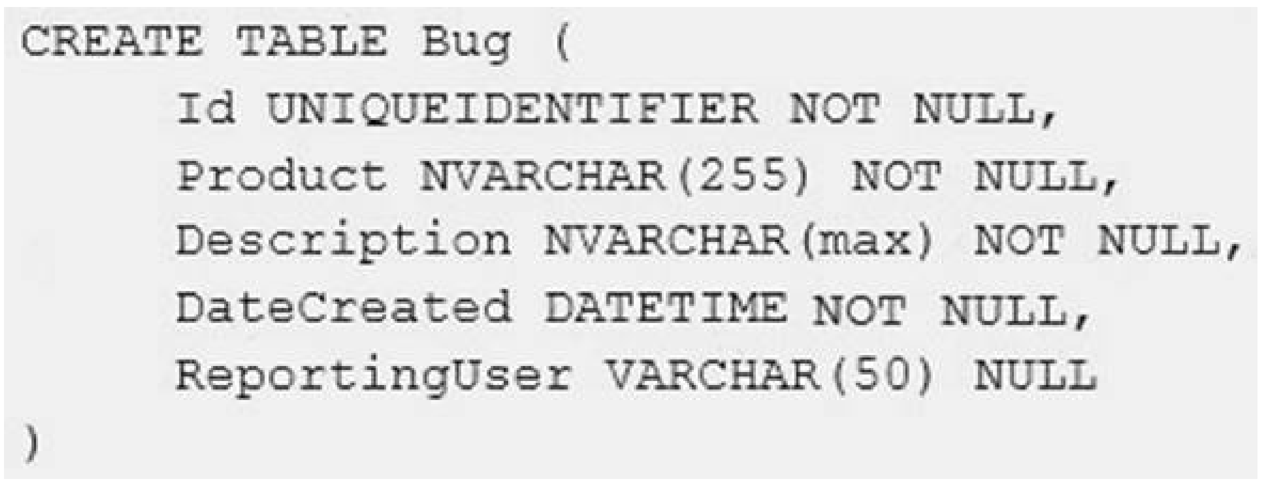
```
DECLARE @startedTasks TABLE(TaskId int)
UPDATE Task SET StartTime = GETDATE() OUTPUT deleted.TaskId, INTO @startedTasks WHERE StartTime is NULL
SELECT COUNT(*) FROM @startedTasks WHERE TaskId IS NOT NULL
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: B

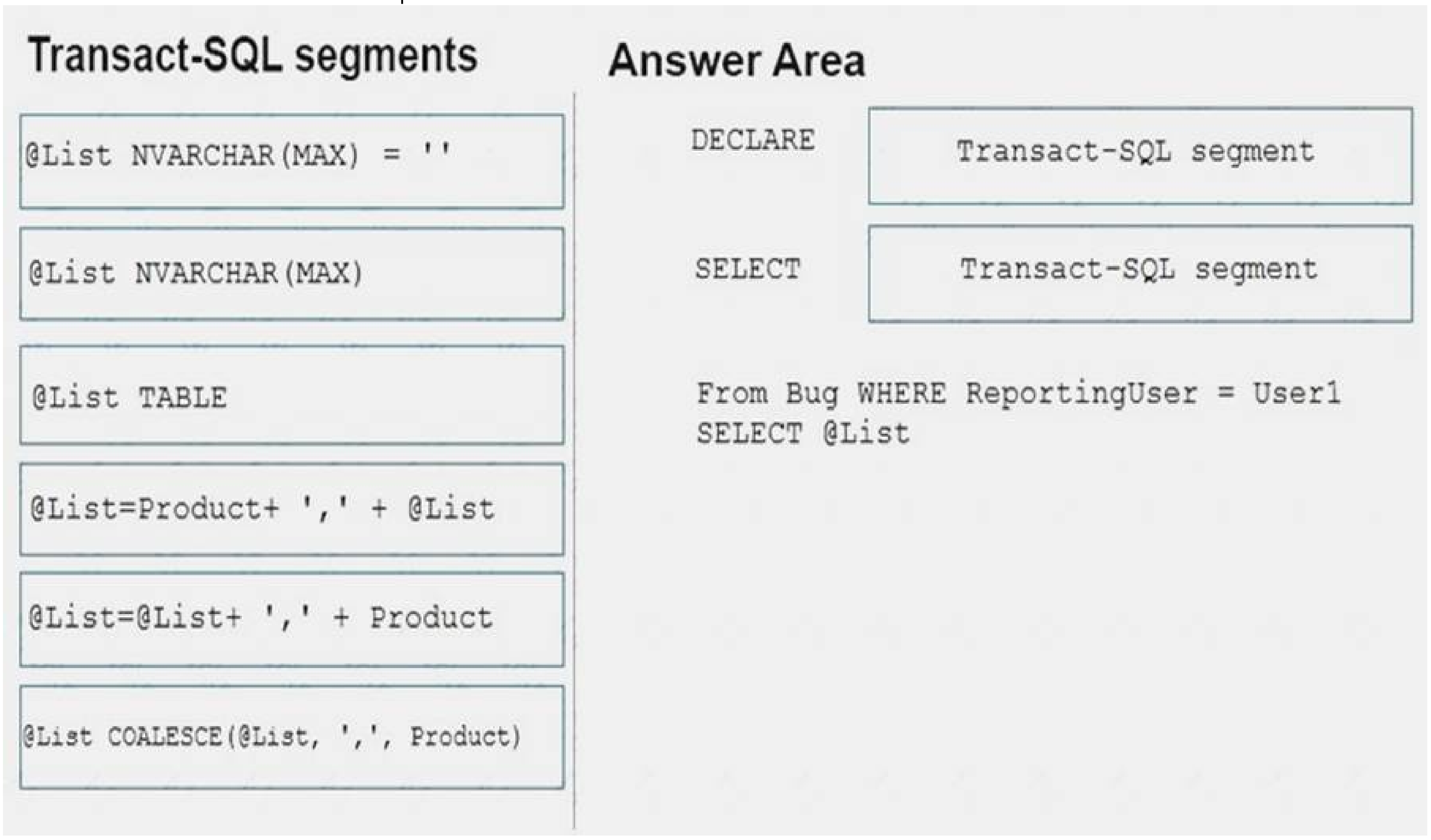
Explanation:
The WHERE clause of the third line should be WHERE ProjectID IS NULL, as we want to count the tasks that are not associated with a project.

NEW QUESTION 112
You have a project management application. The application uses a Microsoft SQL Server database to store data. You are developing a software bug tracking add-on for the application.
The add-on must meet the following requirements:
 Allow case sensitive searches for product.
 Filter search results based on exact text in the description.
 Support multibyte Unicode characters.
You run the following Transact-SQL statement:



```
CREATE TABLE Bug (
      Id UNIQUEIDENTIFIER NOT NULL,
      Product NVARCHAR(255) NOT NULL,
      Description NVARCHAR(max) NOT NULL,
      DateCreated DATETIME NOT NULL,
      ReportingUser VARCHAR(50) NULL
)
```

You need to display a comma separated list of all product bugs filed by a user named User1.
How should you complete the Transact-SQL statement? 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.



Transact-SQL segments

- `@List NVARCHAR(MAX) = ''`
- `@List NVARCHAR(MAX)`
- `@List TABLE`
- `@List=Product+ ',' + @List`
- `@List=@List+ ',' + Product`
- `@List COALESCE(@List, ',', Product)`

Answer Area

```
DECLARE    Transact-SQL segment

SELECT     Transact-SQL segment

From Bug WHERE ReportingUser = User1
SELECT @List
```

A. Mastered
B. Not Mastered

Answer: A

Explanation:
References: https://docs.microsoft.com/en-us/sql/t-sql/functions/string-split-transact-sql?view=sql-server-2017

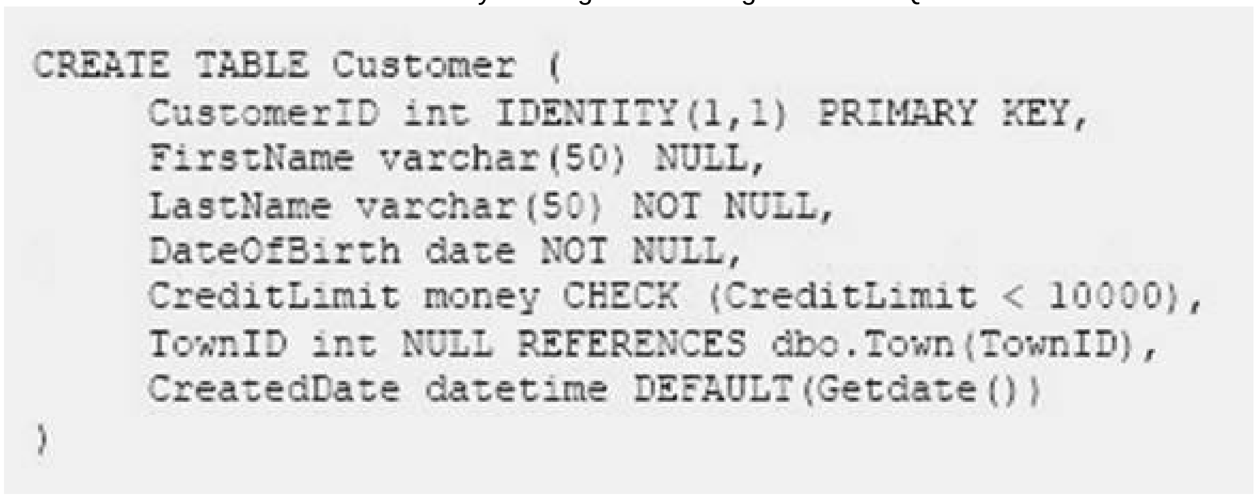
NEW QUESTION 114
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You have a database that contains several connected tables. The tables contain sales data for customers in the United States only.
All the sales data is stored in a table named table1. You have a table named table2 that contains city names. You need to create a query that lists only the cities that have no sales.
Which statement clause should you add to the query?

A. GROUP BY
B. MERGE
C. GROUP BY ROLLUP
D. LEFT JOIN
E. GROUP BY CUBE
F. CROSS JOIN
G. PIVOT
H. UNPIVOT

Answer: D

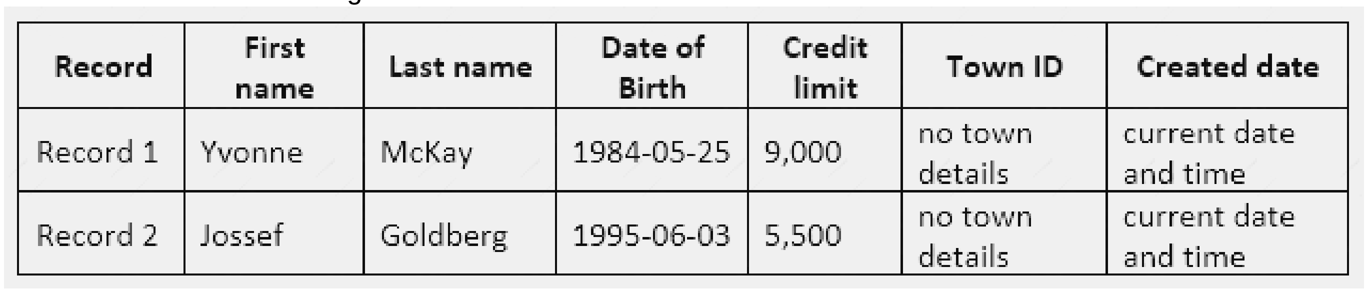
NEW QUESTION 119
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the

stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You create a table named Customer by running the following Transact-SQL statement:



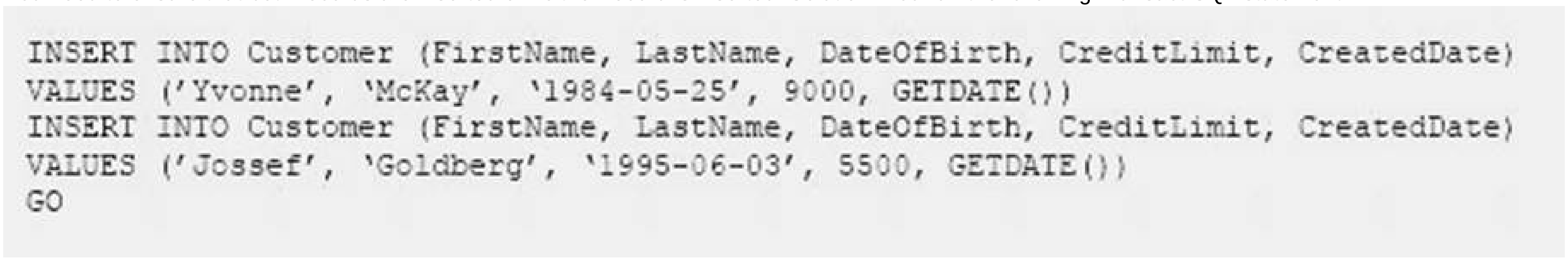
```
CREATE TABLE Customer (
     CustomerID int IDENTITY(1,1) PRIMARY KEY,
     FirstName varchar(50) NULL,
     LastName varchar(50) NOT NULL,
     DateOfBirth date NOT NULL,
     CreditLimit money CHECK (CreditLimit < 10000),
     TownID int NULL REFERENCES dbo.Town(TownID),
     CreatedDate datetime DEFAULT(Getdate())
)
```

You must insert the following data into the Customer table:



Record	First name	Last name	Date of Birth	Credit limit	Town ID	Created date
Record 1	Yvonne	McKay	1984-05-25	9,000	no town details	current date and time
Record 2	Jossef	Goldberg	1995-06-03	5,500	no town details	current date and time

You need to ensure that both records are inserted or neither record is inserted. Solution: You run the following Transact-SQL statement:



```
INSERT INTO Customer (FirstName, LastName, DateOfBirth, CreditLimit, CreatedDate)
VALUES ('Yvonne', 'McKay', '1984-05-25', 9000, GETDATE())
INSERT INTO Customer (FirstName, LastName, DateOfBirth, CreditLimit, CreatedDate)
VALUES ('Jossef', 'Goldberg', '1995-06-03', 5500, GETDATE())
GO
```

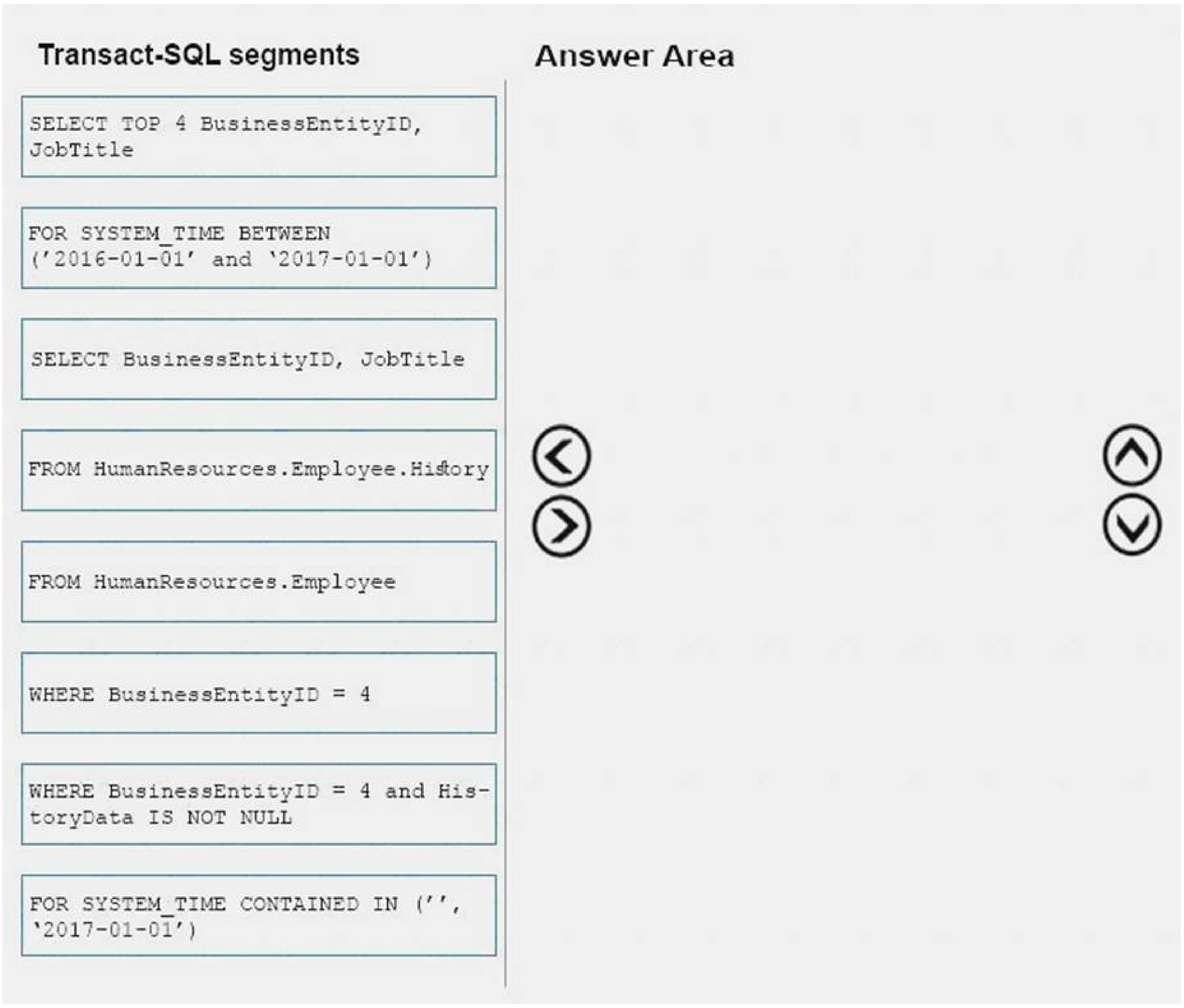
Does the solution meet the goal?

A. Yes
B. No

Answer: B

Explanation:
As there are two separate INSERT INTO statements we cannot ensure that both or neither records is inserted.

NEW QUESTION 122
You have a table named HumanResources.Employee. You configure the table to use a default history table that contains 10 years of data.
You need to write a query that retrieves the values of the BusinessEntityID and JobTitle fields. You must retrieve all historical data up to January 1, 2017 where the value of the BusinessEntityID column equals 4.
Which four Transact-SQL segments should you use to develop the solution? To answer, move the appropriate Transact-SQL segments to the answer area and arrange them in the correct order.


Transact-SQL segments

```
SELECT TOP 4 BusinessEntityID,
JobTitle
```

```
FOR SYSTEM_TIME BETWEEN
('2016-01-01' and '2017-01-01')
```

```
SELECT BusinessEntityID, JobTitle
```

```
FROM HumanResources.Employee.History
```

```
FROM HumanResources.Employee
```

```
WHERE BusinessEntityID = 4
```

```
WHERE BusinessEntityID = 4 and His-
toryData IS NOT NULL
```

```
FOR SYSTEM_TIME CONTAINED IN ('',
'2017-01-01')
```

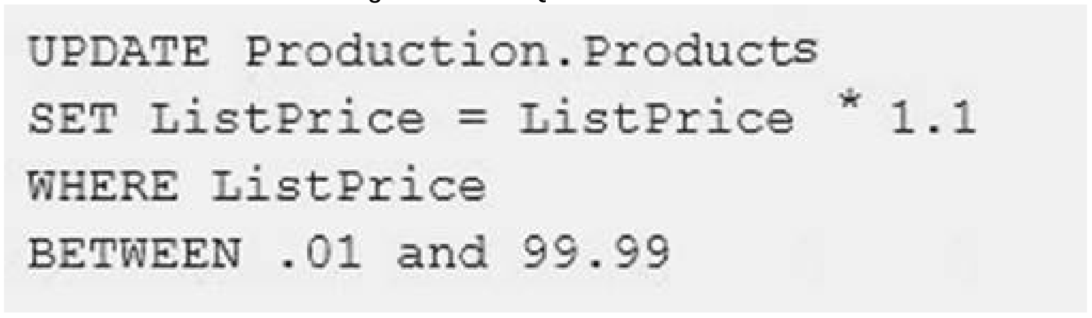
Answer Area

A. Mastered
B. Not Mastered

Answer: A

Explanation:
References:
https://docs.microsoft.com/en-us/sql/relational-databases/tables/querying-data-in-a-system-versioned-temporal-t

NEW QUESTION 127
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a table named Products that stores information about products your company sells. The table has a column named ListPrice that stores retail pricing information for products.
Some products are used only internally by the company. Records for these products are maintained in the Products table for inventory purposes. The price for each of these products is $0.00. Customers are not permitted to order these products.
You need to increase the list price for products that cost less than $100 by 10 percent. You must only increase pricing for products that customers are permitted to order.
Solution: You run the following Transact-SQL statement:


```
UPDATE Production.Products
SET ListPrice = ListPrice * 1.1
WHERE ListPrice
BETWEEN .01 and 99.99
```

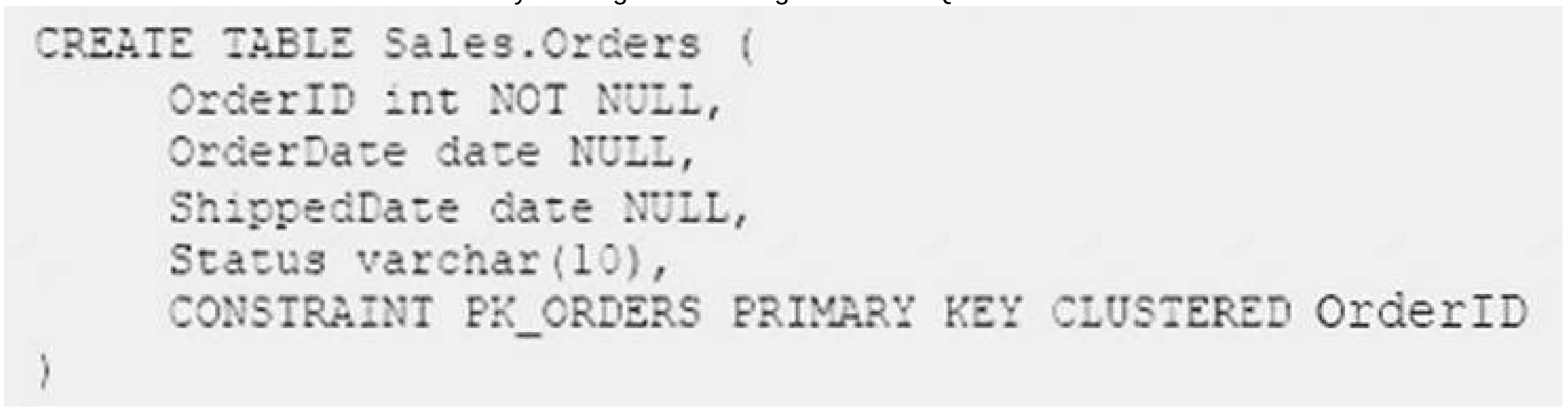
Does the solution meet the goal?

A. Yes

B. No

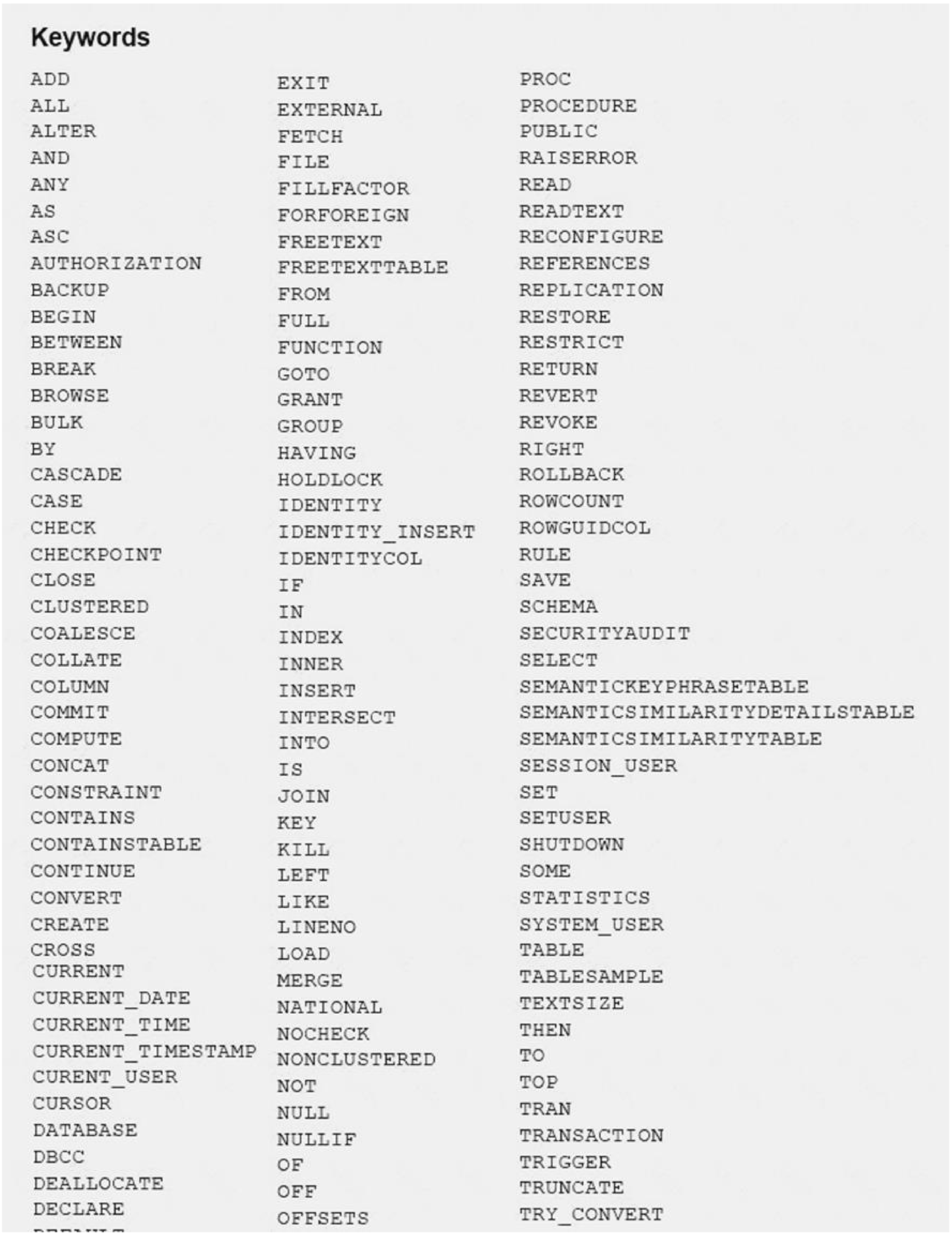
Answer: A

NEW QUESTION 130
You create a table named Sales.Orders by running the following Transact-SQL statement:



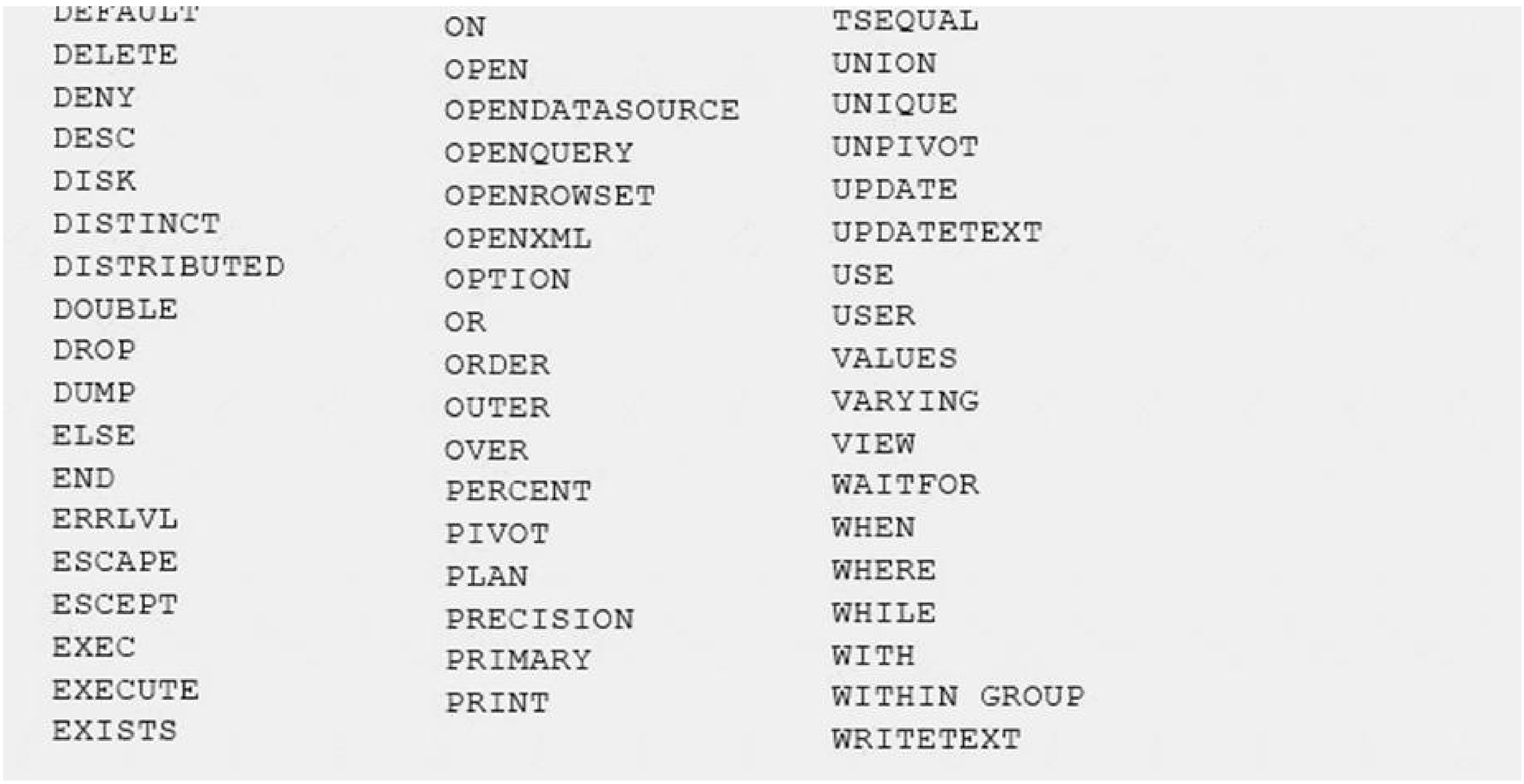
```
CREATE TABLE Sales.Orders (
     OrderID int NOT NULL,
     OrderDate date NULL,
     ShippedDate date NULL,
     Status varchar(10),
     CONSTRAINT PK_ORDERS PRIMARY KEY CLUSTERED OrderID
)
```

You need to write a query that removes orders from the table that have a Status of Canceled. Construct the query using the following guidelines:



Keywords

ADD	EXIT	PROC
ALL	EXTERNAL	PROCEDURE
ALTER	FETCH	PUBLIC
AND	FILE	RAISERROR
ANY	FILLFACTOR	READ
AS	FORFOREIGN	READTEXT
ASC	FREETEXT	RECONFIGURE
AUTHORIZATION	FREETEXTTABLE	REFERENCES
BACKUP	FROM	REPLICATION
BEGIN	FULL	RESTORE
BETWEEN	FUNCTION	RESTRICT
BREAK	GOTO	RETURN
BROWSE	GRANT	REVERT
BULK	GROUP	REVOKE
BY	HAVING	RIGHT
CASCADE	HOLDLOCK	ROLLBACK
CASE	IDENTITY	ROWCOUNT
CHECK	IDENTITY_INSERT	ROWGUIDCOL
CHECKPOINT	IDENTITYCOL	RULE
CLOSE	IF	SAVE
CLUSTERED	IN	SCHEMA
COALESCE	INDEX	SECURITYAUDIT
COLLATE	INNER	SELECT
COLUMN	INSERT	SEMANTICKEYPHRASETABLE
COMMIT	INTERSECT	SEMANTICSIMILARITYDETAILSTABLE
COMPUTE	INTO	SEMANTICSIMILARITYTABLE
CONCAT	IS	SESSION_USER
CONSTRAINT	JOIN	SET
CONTAINS	KEY	SETUSER
CONTAINSTABLE	KILL	SHUTDOWN
CONTINUE	LEFT	SOME
CONVERT	LIKE	STATISTICS
CREATE	LINENO	SYSTEM_USER
CROSS	LOAD	TABLE
CURRENT	MERGE	TABLESAMPLE
CURRENT_DATE	NATIONAL	TEXTSIZE
CURRENT_TIME	NOCHECK	THEN
CURRENT_TIMESTAMP	NONCLUSTERED	TO
CURENT_USER	NOT	TOP
CURSOR	NULL	TRAN
DATABASE	NULLIF	TRANSACTION
DBCC	OF	TRIGGER
DEALLOCATE	OFF	TRUNCATE
DECLARE	OFFSETS	TRY_CONVERT



```
DEFAULT        ON               TSEQUAL
DELETE         OPEN             UNION
DENY           OPENDATASOURCE   UNIQUE
DESC           OPENQUERY        UNPIVOT
DISK           OPENROWSET       UPDATE
DISTINCT       OPENXML          UPDATETEXT
DISTRIBUTED    OPTION           USE
DOUBLE         OR               USER
DROP           ORDER            VALUES
DUMP           OUTER            VARYING
ELSE           OVER             VIEW
END            PERCENT          WAITFOR
ERRLVL         PIVOT            WHEN
ESCAPE         PLAN             WHERE
ESCEPT         PRECISION        WHILE
EXEC           PRIMARY          WITH
EXECUTE        PRINT            WITHIN GROUP
EXISTS                          WRITETEXT
```

Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the code that has been provided as well as below it.

```
DELETE from sales.orders where status='calceled'
```

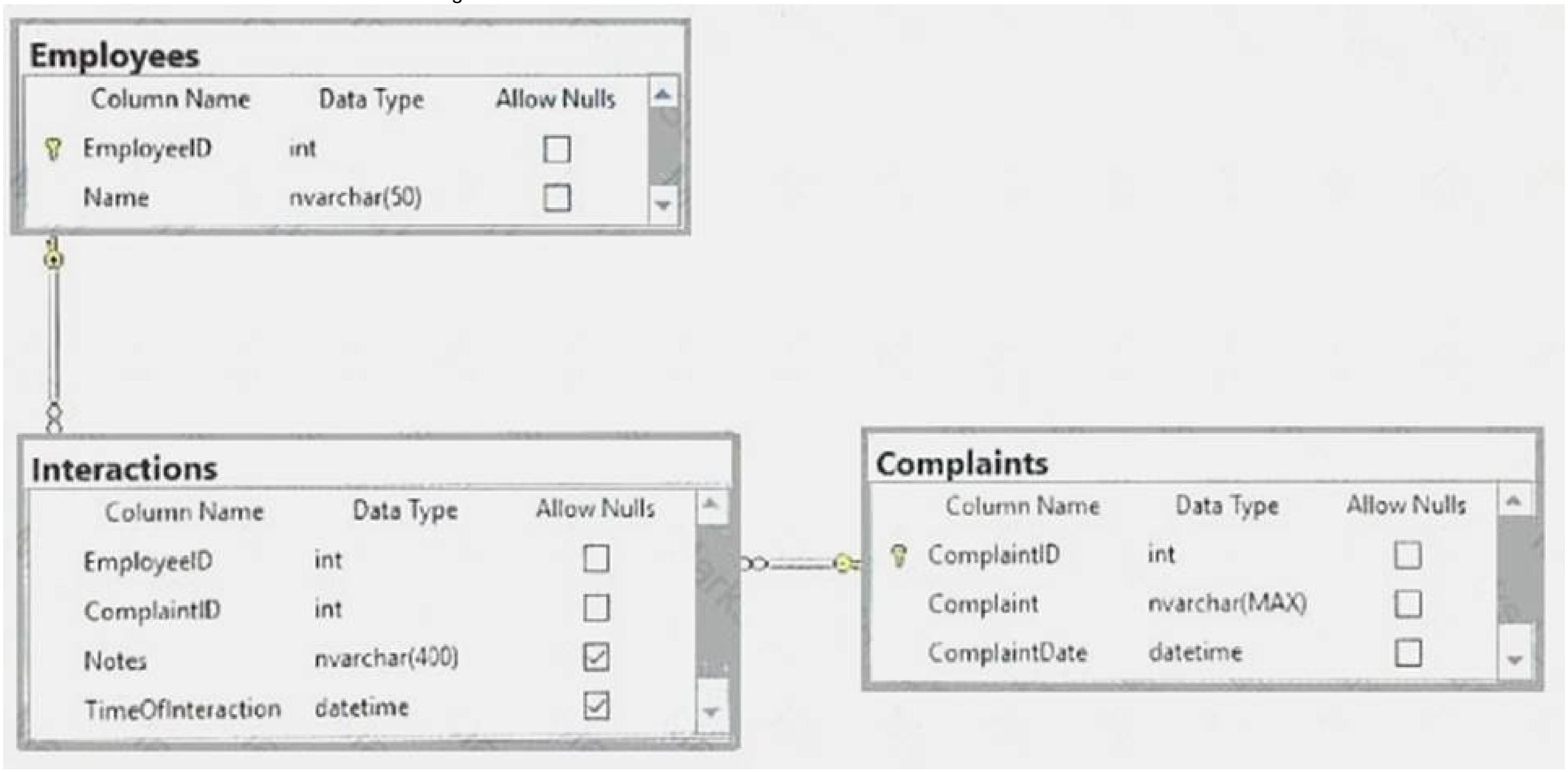
Use the Check Syntax button to verify your work. Any syntax or spelling errors will be reported by line and character position.

A. Mastered
B. Not Mastered

Answer: A

Explanation:
1. DELETE from sales.orders where status='Canceled' Note: On line 1 change calceled to Canceled
Example: Using the WHERE clause to delete a set of rows
The following example deletes all rows from the ProductCostHistory table in the AdventureWorks2012 database in which the value in the StandardCost column is more than 1000.00.
DELETE FROM Production.ProductCostHistory WHERE StandardCost > 1000.00;
References: https://docs.microsoft.com/en-us/sql/t-sql/statements/delete-transact-sql

NEW QUESTION 131
You have a database that contains the following tables.



Employees

Column Name	Data Type	Allow Nulls
EmployeeID	int	☐
Name	nvarchar(50)	☐

Interactions

Column Name	Data Type	Allow Nulls
EmployeeID	int	☐
ComplaintID	int	☐
Notes	nvarchar(400)	☑
TimeOfInteraction	datetime	☑

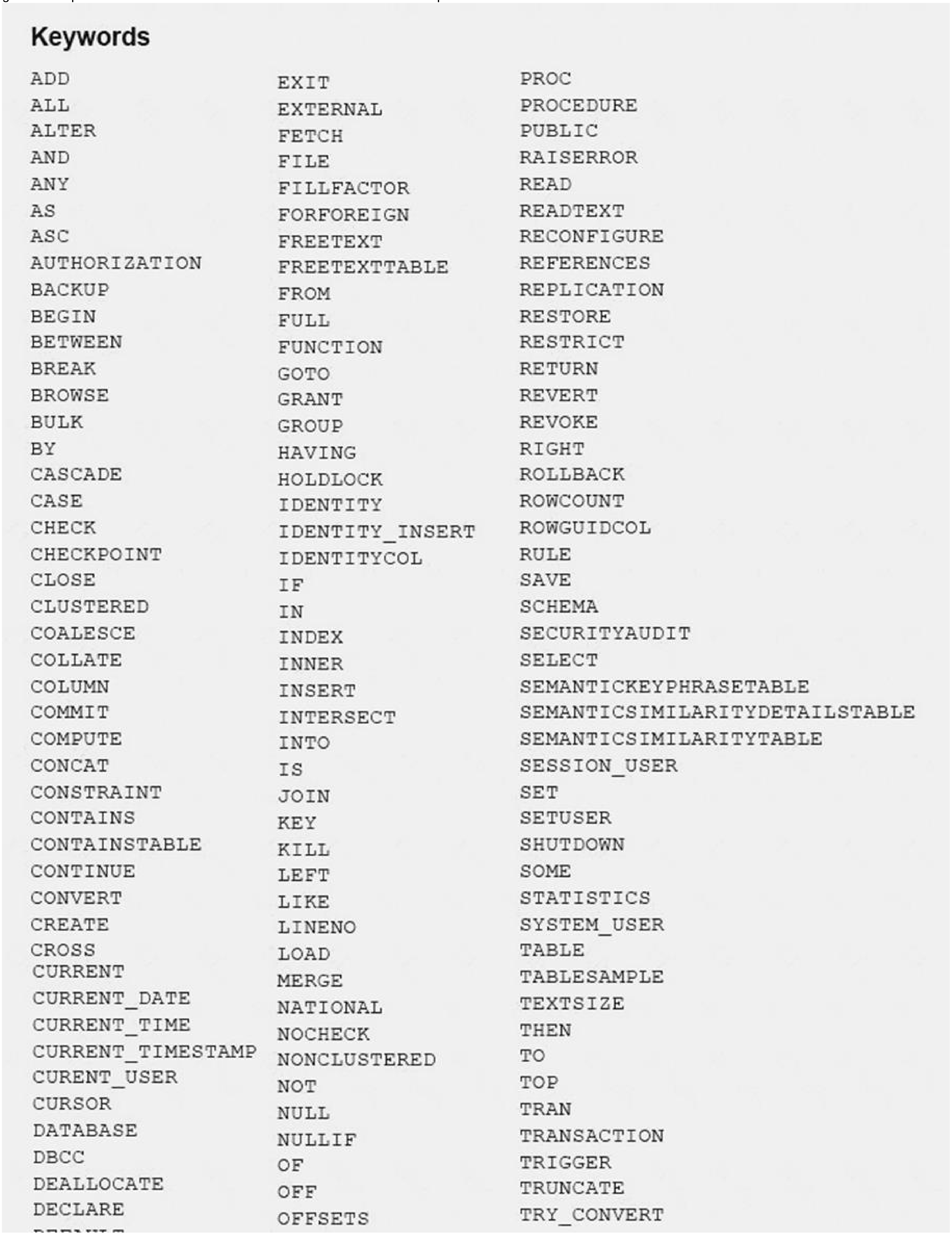
Complaints

Column Name	Data Type	Allow Nulls
ComplaintID	int	☐
Complaint	nvarchar(MAX)	☐
ComplaintDate	datetime	☐

You need to create a query that returns each complaint, the names of the employees handling the complaint, and the notes on each interaction. The Complaint field must be displayed first, followed by the employee's name and the notes. Complaints must be returned even if no interaction has occurred.
Construct the query using the following guidelines:

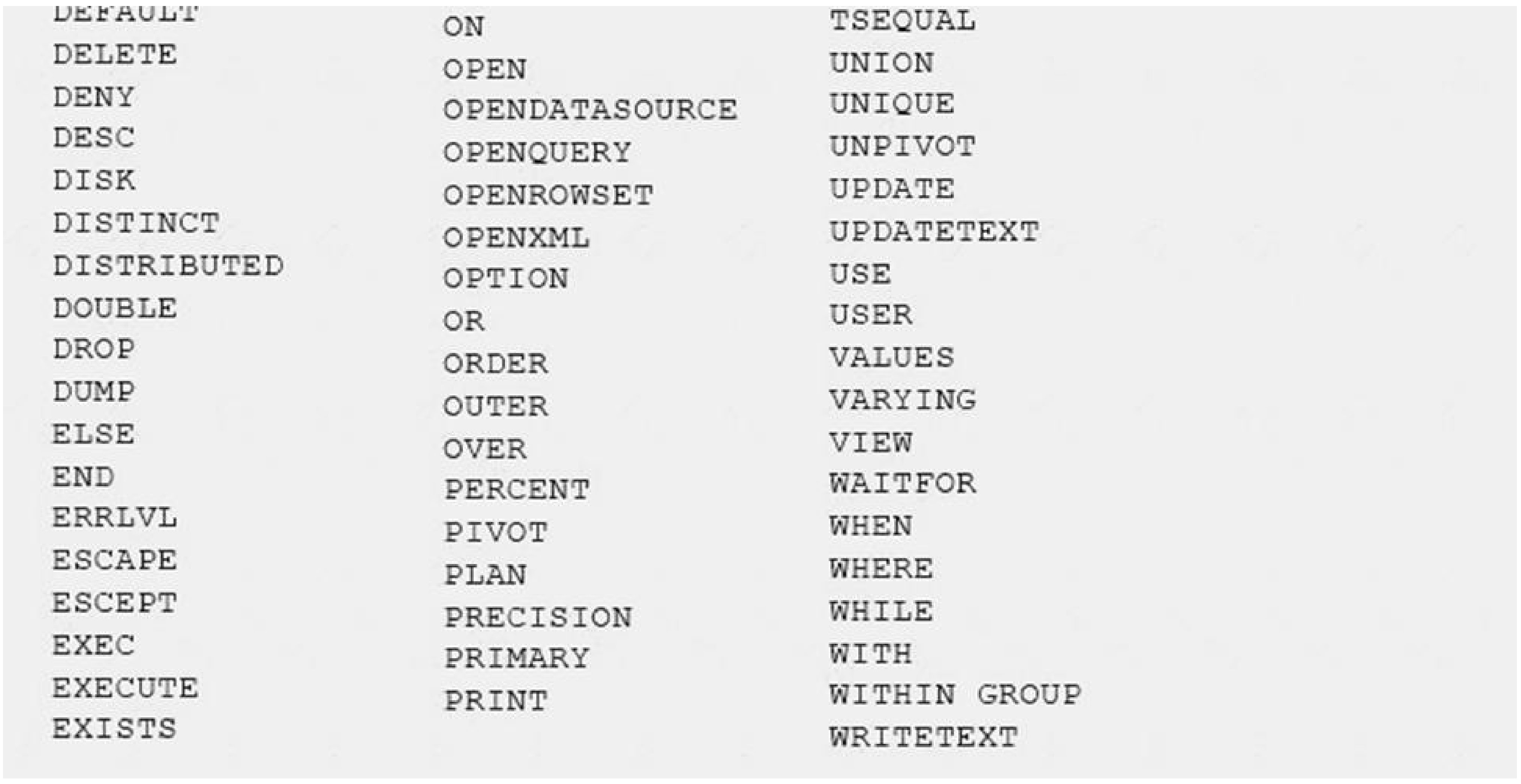
- Use two-part column names.
- Use one-part table names.
- Use the first letter of the table name as its alias.
- Do not Transact-SQL functions.
- Do not use implicit joins.
- Do not surround object names with square brackets.

Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the code that has been provided as well as below it.



Keywords

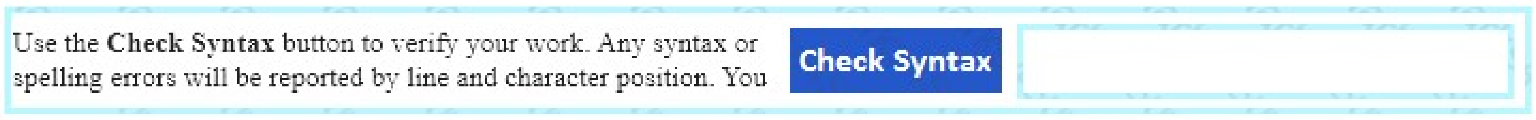
```
ADD
ALL
ALTER
AND
ANY
AS
ASC
AUTHORIZATION
BACKUP
BEGIN
BETWEEN
BREAK
BROWSE
BULK
BY
CASCADE
CASE
CHECK
CHECKPOINT
CLOSE
CLUSTERED
COALESCE
COLLATE
COLUMN
COMMIT
COMPUTE
CONCAT
CONSTRAINT
CONTAINS
CONTAINSTABLE
CONTINUE
CONVERT
CREATE
CROSS
CURRENT
CURRENT_DATE
CURRENT_TIME
CURRENT_TIMESTAMP
CURENT_USER
CURSOR
DATABASE
DBCC
DEALLOCATE
DECLARE
EXIT
EXTERNAL
FETCH
FILE
FILLFACTOR
FORFOREIGN
FREETEXT
FREETEXTTABLE
FROM
FULL
FUNCTION
GOTO
GRANT
GROUP
HAVING
HOLDLOCK
IDENTITY
IDENTITY_INSERT
IDENTITYCOL
IF
IN
INDEX
INNER
INSERT
INTERSECT
INTO
IS
JOIN
KEY
KILL
LEFT
LIKE
LINENO
LOAD
MERGE
NATIONAL
NOCHECK
NONCLUSTERED
NOT
NULL
NULLIF
OF
OFF
OFFSETS
PROC
PROCEDURE
PUBLIC
RAISERROR
READ
READTEXT
RECONFIGURE
REFERENCES
REPLICATION
RESTORE
RESTRICT
RETURN
REVERT
REVOKE
RIGHT
ROLLBACK
ROWCOUNT
ROWGUIDCOL
RULE
SAVE
SCHEMA
SECURITYAUDIT
SELECT
SEMANTICKEYPHRASETABLE
SEMANTICSIMILARITYDETAILSTABLE
SEMANTICSIMILARITYTABLE
SESSION_USER
SET
SETUSER
SHUTDOWN
SOME
STATISTICS
SYSTEM_USER
TABLE
TABLESAMPLE
TEXTSIZE
THEN
TO
TOP
TRAN
TRANSACTION
TRIGGER
TRUNCATE
TRY_CONVERT
```


DEFAULT
DELETE
DENY
DESC
DISK
DISTINCT
DISTRIBUTED
DOUBLE
DROP
DUMP
ELSE
END
ERRLVL
ESCAPE
ESCEPT
EXEC
EXECUTE
EXISTS

ON
OPEN
OPENDATASOURCE
OPENQUERY
OPENROWSET
OPENXML
OPTION
OR
ORDER
OUTER
OVER
PERCENT
PIVOT
PLAN
PRECISION
PRIMARY
PRINT

TSEQUAL
UNION
UNIQUE
UNPIVOT
UPDATE
UPDATETEXT
USE
USER
VALUES
VARYING
VIEW
WAITFOR
WHEN
WHERE
WHILE
WITH
WITHIN GROUP
WRITETEXT

1 SELECT c.Complaint, e.Name, i.Notes 2 FROM Complaints c
3 JOIN
4 JOIN

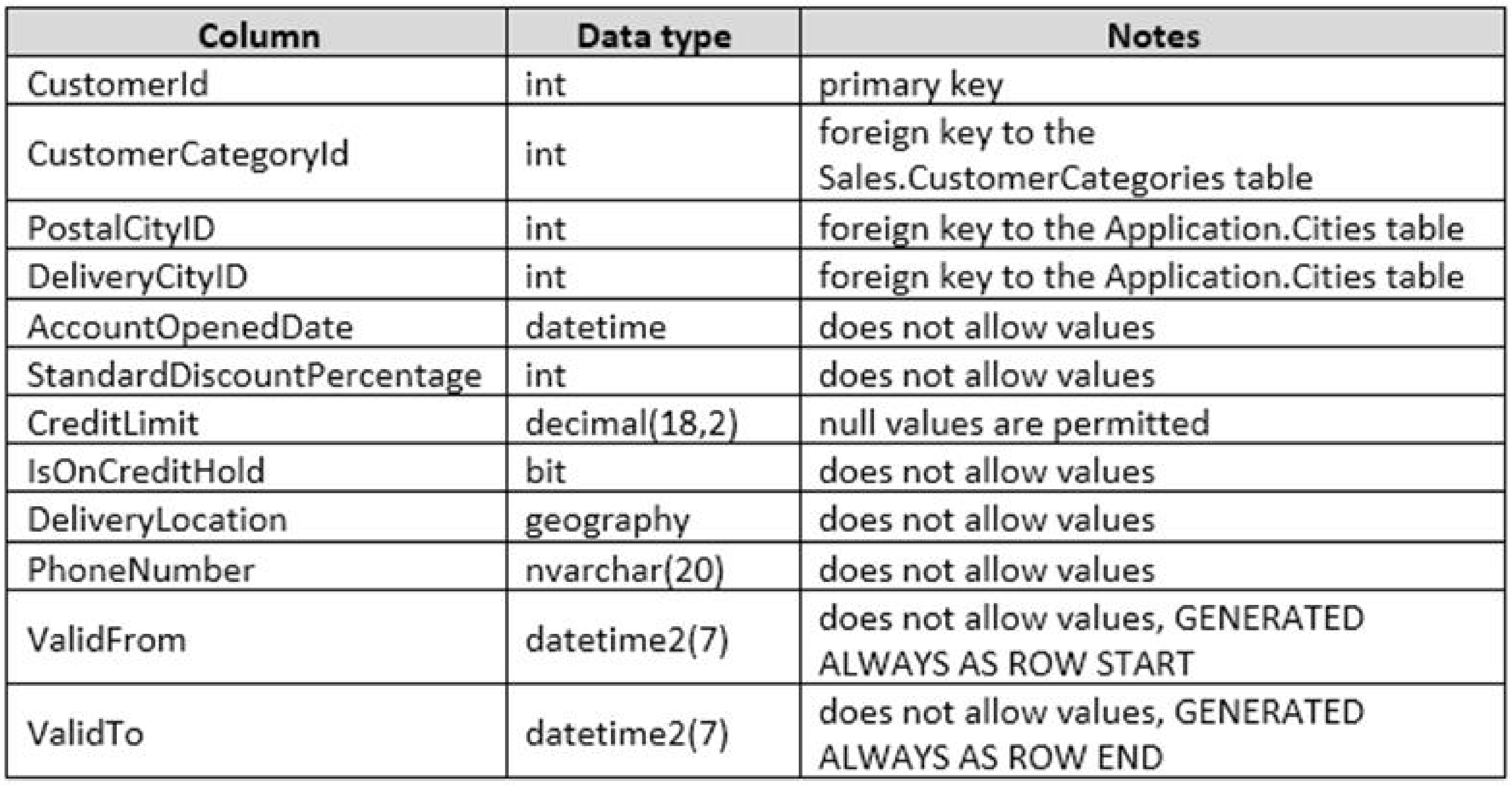

Use the **Check Syntax** button to verify your work. Any syntax or spelling errors will be reported by line and character position. You

A. Mastered
B. Not Mastered

Answer: A

Explanation:
1 SELECT c.Complaint, e.Name, i.Notes
2 FROM Complaints c
3 JOIN Interactions i ON c.ComplaintID = i.ComplaintID
4 JOIN Employees e ON i.EmployeeID = E.EmployeeID

NEW QUESTION 134
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 on this series.
You have a database that tracks orders and deliveries for customers in North America. System versioning is enabled for all tables. The database contains the Sales.Customers, Application.Cities, and Sales.CustomerCategories tables.
Details for the Sales.Customers table are shown in the following table:



Column	Data type	Notes
CustomerId	int	primary key
CustomerCategoryId	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow values
StandardDiscountPercentage	int	does not allow values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow values
DeliveryLocation	geography	does not allow values
PhoneNumber	nvarchar(20)	does not allow values
ValidFrom	datetime2(7)	does not allow values, GENERATED ALWAYS AS ROW START
ValidTo	datetime2(7)	does not allow values, GENERATED ALWAYS AS ROW END

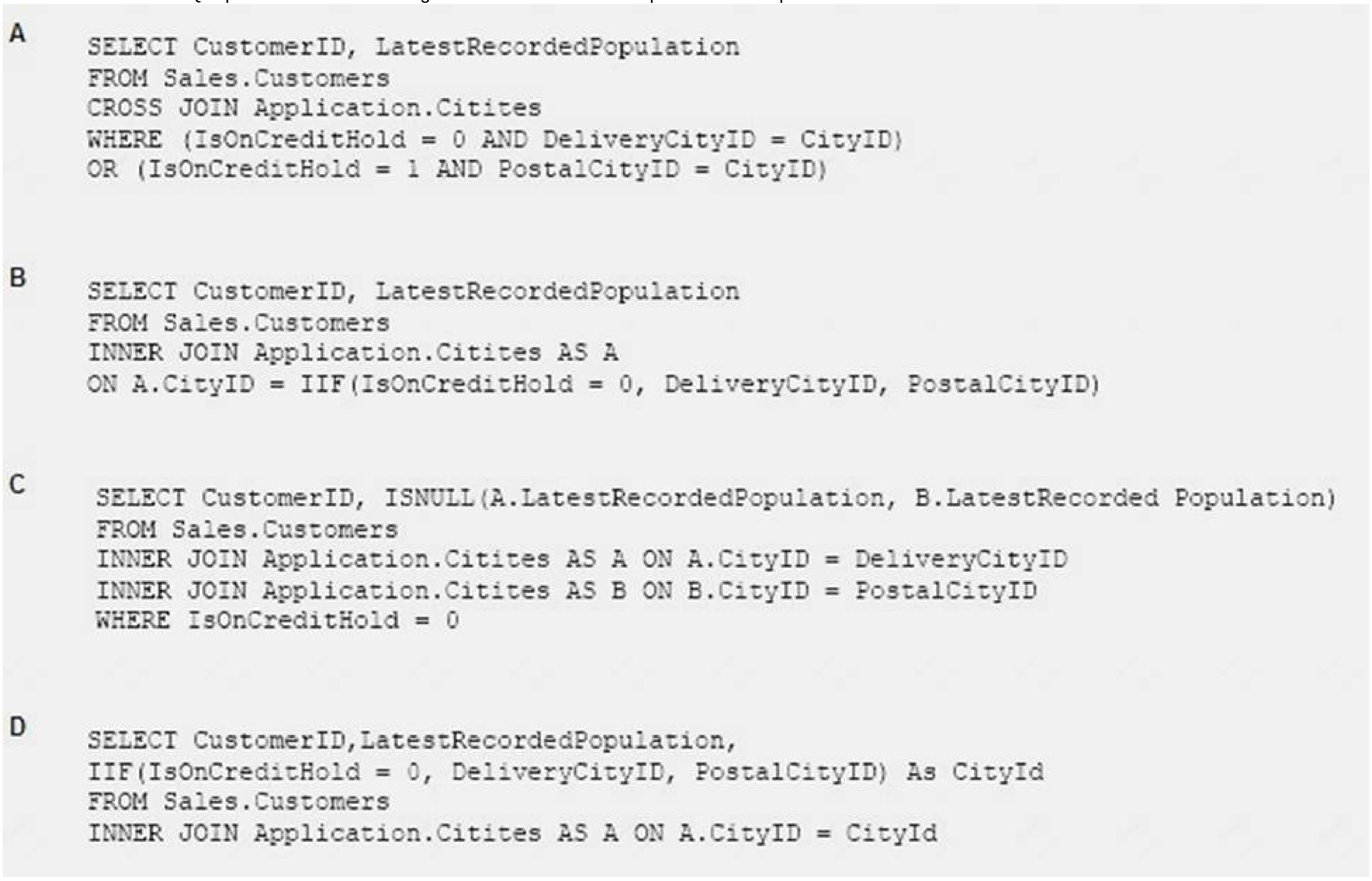
Details for the Application.Cities table are shown in the following table:

Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Details for the Sales.CustomerCategories table are shown in the following table:

Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

You need to create a query that meets the following requirements:
- For customers that are not on a credit hold, return the CustomerID and the latest recorded population for the delivery city that is associated with the customer.
- For customers that are on a credit hold, return the CustomerID and the latest recorded population for the postal city that is associated with the customer.
Which two Transact-SQL queries will achieve the goal? Each correct answer presents a complete solution.



A

```
SELECT CustomerID, LatestRecordedPopulation
FROM Sales.Customers
CROSS JOIN Application.Citites
WHERE (IsOnCreditHold = 0 AND DeliveryCityID = CityID)
OR (IsOnCreditHold = 1 AND PostalCityID = CityID)
```

B

```
SELECT CustomerID, LatestRecordedPopulation
FROM Sales.Customers
INNER JOIN Application.Citites AS A
ON A.CityID = IIF(IsOnCreditHold = 0, DeliveryCityID, PostalCityID)
```

C

```
SELECT CustomerID, ISNULL(A.LatestRecordedPopulation, B.LatestRecorded Population)
FROM Sales.Customers
INNER JOIN Application.Citites AS A ON A.CityID = DeliveryCityID
INNER JOIN Application.Citites AS B ON B.CityID = PostalCityID
WHERE IsOnCreditHold = 0
```

D

```
SELECT CustomerID,LatestRecordedPopulation,
IIF(IsOnCreditHold = 0, DeliveryCityID, PostalCityID) As CityId
FROM Sales.Customers
INNER JOIN Application.Citites AS A ON A.CityID = CityId
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: AB

Explanation:
Using Cross Joins
A cross join that does not have a WHERE clause produces the Cartesian product of the tables involved in the join. The size of a Cartesian product result set is the number of rows in the first table multiplied by the number of rows in the second table.
However, if a WHERE clause is added, the cross join behaves as an inner join. B: You can use the IIF in the ON-statement.
IIF returns one of two values, depending on whether the Boolean expression evaluates to true or false in SQL Server.
References:
https://technet.microsoft.com/en-us/library/ms190690(v=sql.105).aspx https://msdn.microsoft.com/en-us/library/hh213574.aspx

NEW QUESTION 138
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You have a database that contains tables named Customer_CRMSystem and Customer_HRSystem. Both tables use the following structure:

Column name	Data type	Allow null
CustomerID	int	No
CustomerCode	char(4)	Yes
CustomerName	varchar(50)	No

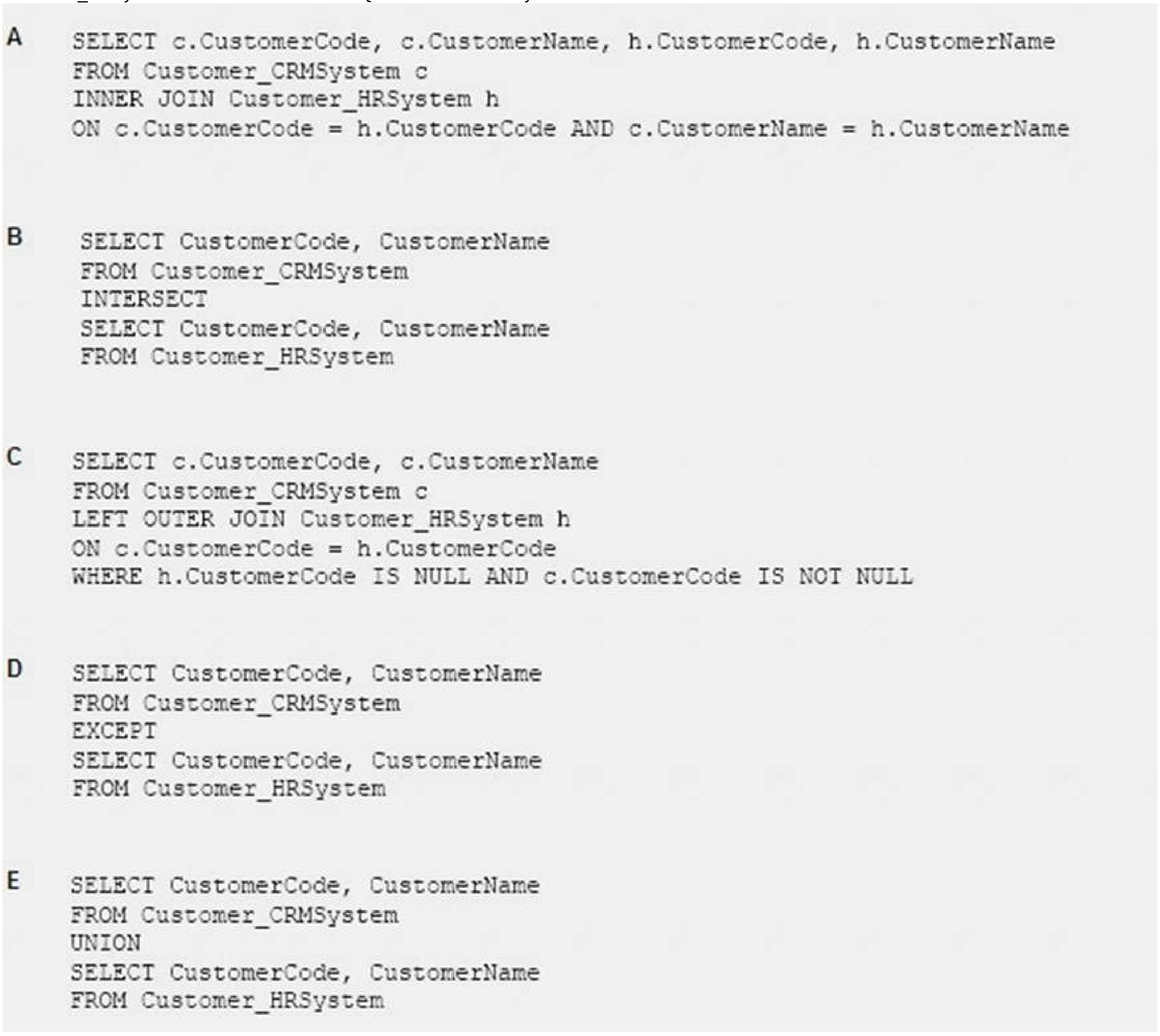
The tables include the following records: Customer_CRMSystem

CustomerID	CustomerCode	CustomerName
1	CUS1	Roya
2	CUS9	Almudena
3	CUS4	Jack
4	NULL	Jane
5	NULL	Francisco

Customer_HRSystem

CustomerID	CustomerCode	CustomerName
1	CUS1	Roya
2	CUS2	Jose
3	CUS9	Almudena
4	NULL	Jane

Records that contain null values for CustomerCode can be uniquely identified by CustomerName. You need to display a list of customers that do not appear in the Customer_HRSystem table. Which Transact-SQL statement should you run?



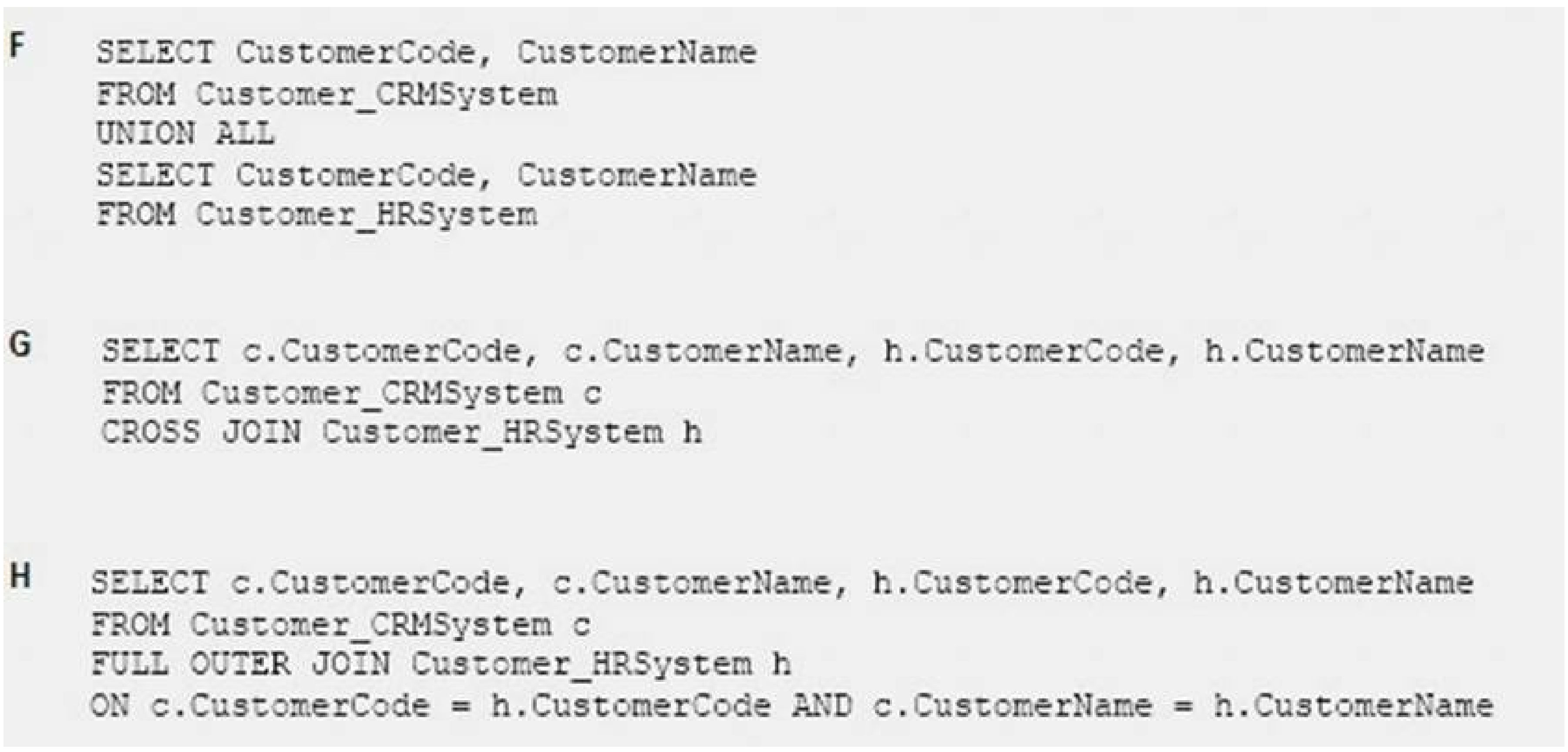
A
```
SELECT c.CustomerCode, c.CustomerName, h.CustomerCode, h.CustomerName
FROM Customer_CRMSystem c
INNER JOIN Customer_HRSystem h
ON c.CustomerCode = h.CustomerCode AND c.CustomerName = h.CustomerName
```

B
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
INTERSECT
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```

C
```
SELECT c.CustomerCode, c.CustomerName
FROM Customer_CRMSystem c
LEFT OUTER JOIN Customer_HRSystem h
ON c.CustomerCode = h.CustomerCode
WHERE h.CustomerCode IS NULL AND c.CustomerCode IS NOT NULL
```

D
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
EXCEPT
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```

E
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
UNION
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```

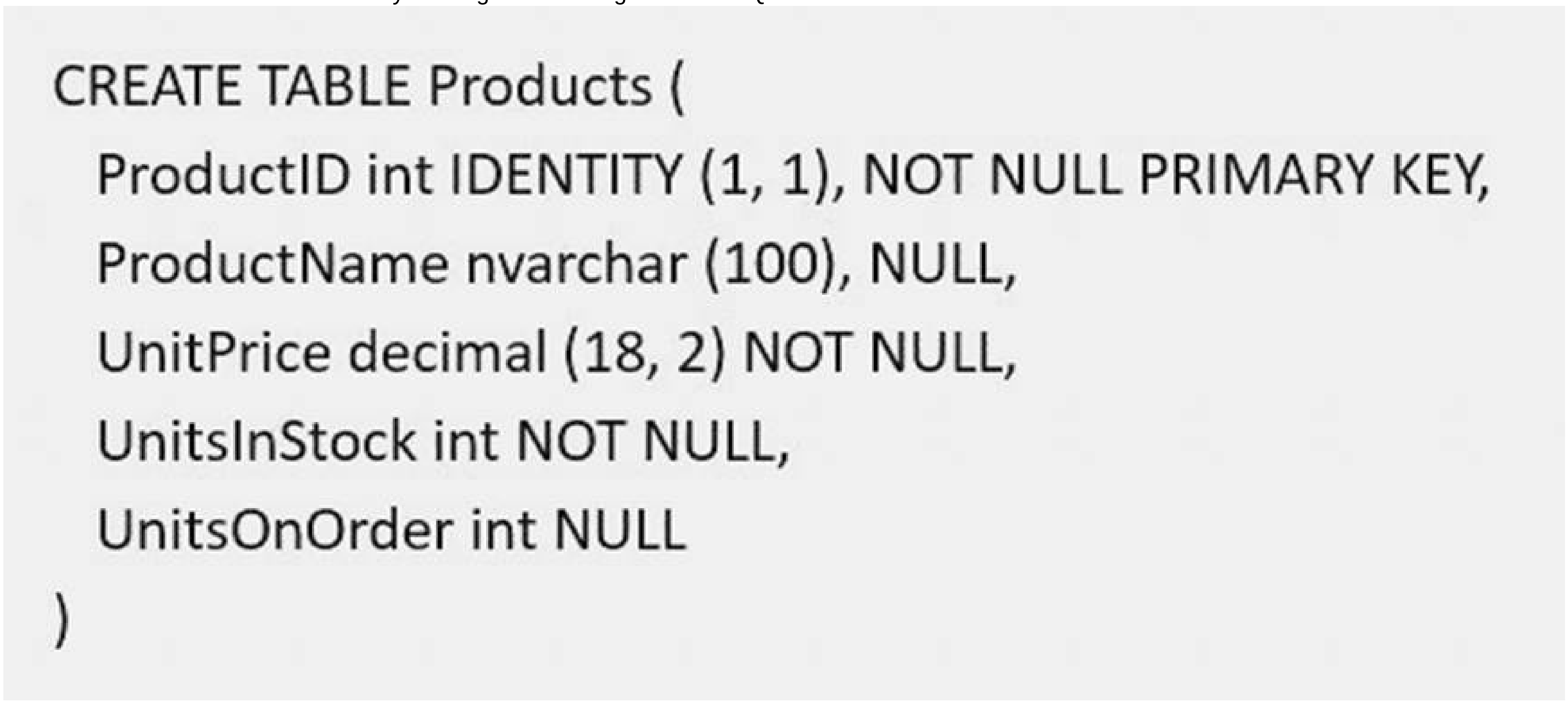

F
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
UNION ALL
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```
G
```
SELECT c.CustomerCode, c.CustomerName, h.CustomerCode, h.CustomerName
FROM Customer_CRMSystem c
CROSS JOIN Customer_HRSystem h
```
H
```
SELECT c.CustomerCode, c.CustomerName, h.CustomerCode, h.CustomerName
FROM Customer_CRMSystem c
FULL OUTER JOIN Customer_HRSystem h
ON c.CustomerCode = h.CustomerCode AND c.CustomerName = h.CustomerName
```

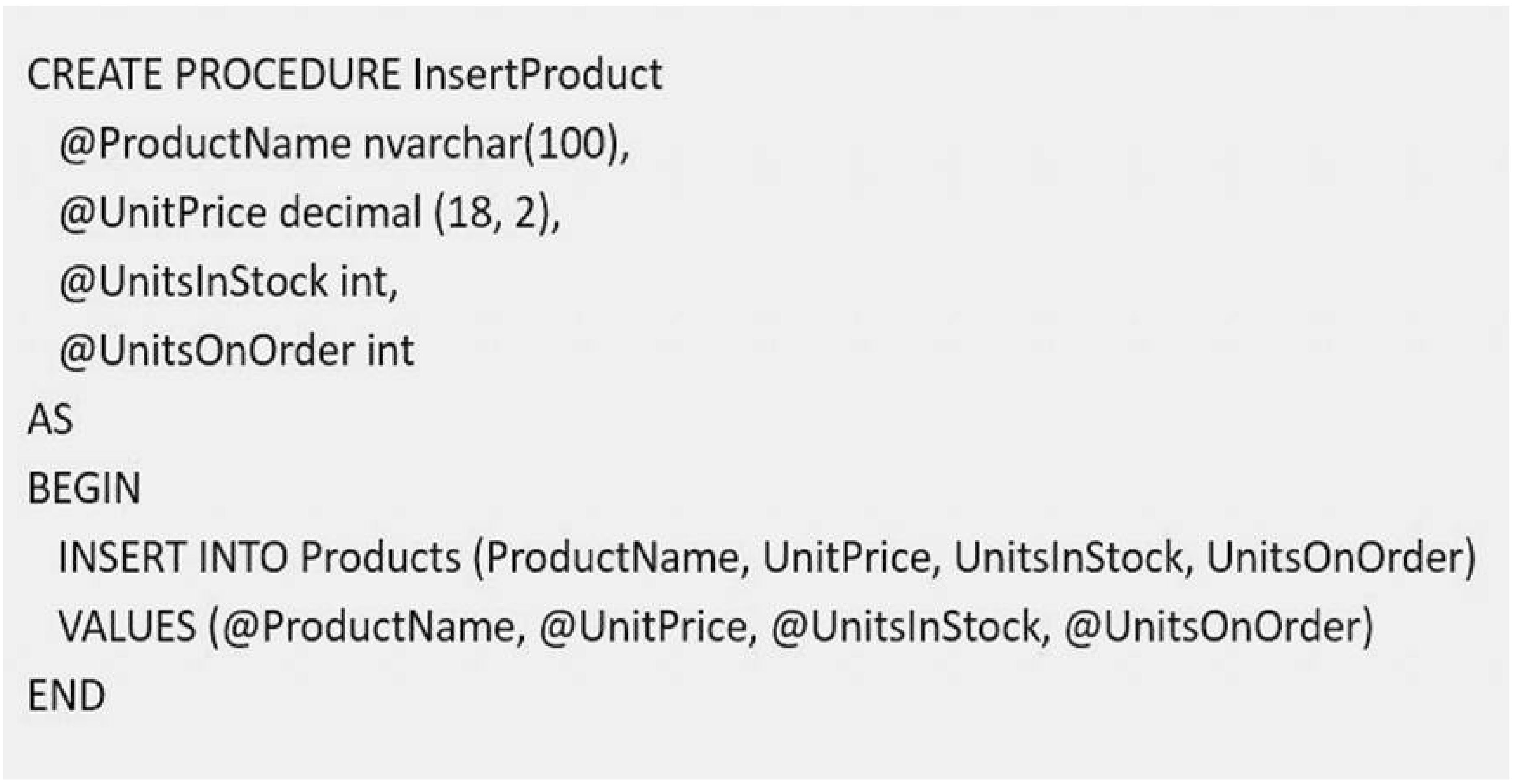
A. Option A
B. Option B
C. Option C
D. Option D
E. Option E
F. Option F
G. Option G
H. Option H

Answer: D

Explanation:
EXCEPT returns distinct rows from the left input query that aren't output by the right input query. References: https://msdn.microsoft.com/en-us/library/ms188055.aspx

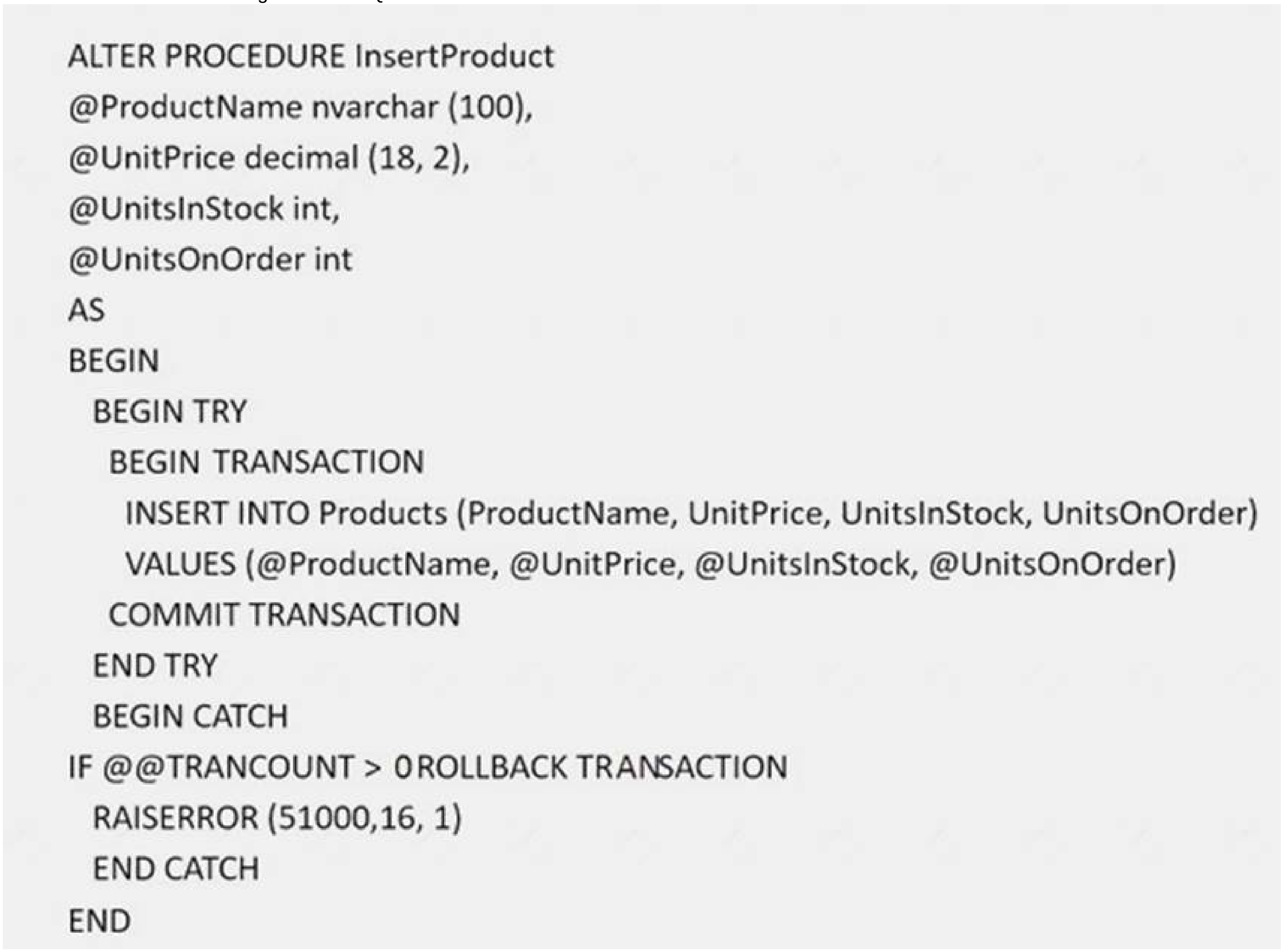
NEW QUESTION 143
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You create a table named Products by running the following Transact-SQL statement:


```
CREATE TABLE Products (
   ProductID int IDENTITY (1, 1), NOT NULL PRIMARY KEY,
   ProductName nvarchar (100), NULL,
   UnitPrice decimal (18, 2) NOT NULL,
   UnitsInStock int NOT NULL,
   UnitsOnOrder int NULL
)
```
You have the following stored procedure:



```
CREATE PROCEDURE InsertProduct
  @ProductName nvarchar(100),
  @UnitPrice decimal (18, 2),
  @UnitsInStock int,
  @UnitsOnOrder int
AS
BEGIN
  INSERT INTO Products (ProductName, UnitPrice, UnitsInStock, UnitsOnOrder)
  VALUES (@ProductName, @UnitPrice, @UnitsInStock, @UnitsOnOrder)
END
```

You need to modify the stored procedure to meet the following new requirements:
 Insert product records as a single unit of work.
 Return error number 51000 when a product fails to insert into the database.
 If a product record insert operation fails, the product information must not be permanently written to the database.
Solution: You run the following Transact-SQL statement:



```
ALTER PROCEDURE InsertProduct
@ProductName nvarchar (100),
@UnitPrice decimal (18, 2),
@UnitsInStock int,
@UnitsOnOrder int
AS
BEGIN
  BEGIN TRY
    BEGIN TRANSACTION
      INSERT INTO Products (ProductName, UnitPrice, UnitsInStock, UnitsOnOrder)
      VALUES (@ProductName, @UnitPrice, @UnitsInStock, @UnitsOnOrder)
    COMMIT TRANSACTION
  END TRY
  BEGIN CATCH
IF @@TRANCOUNT > 0 ROLLBACK TRANSACTION
  RAISERROR (51000,16, 1)
  END CATCH
END
```

Does the solution meet the goal?

A. Yes
B. No

Answer: B

NEW QUESTION 144
You have a database that contains the following tables:

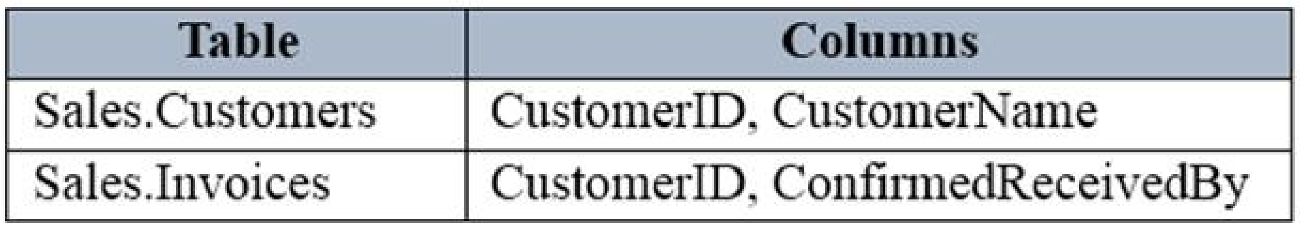
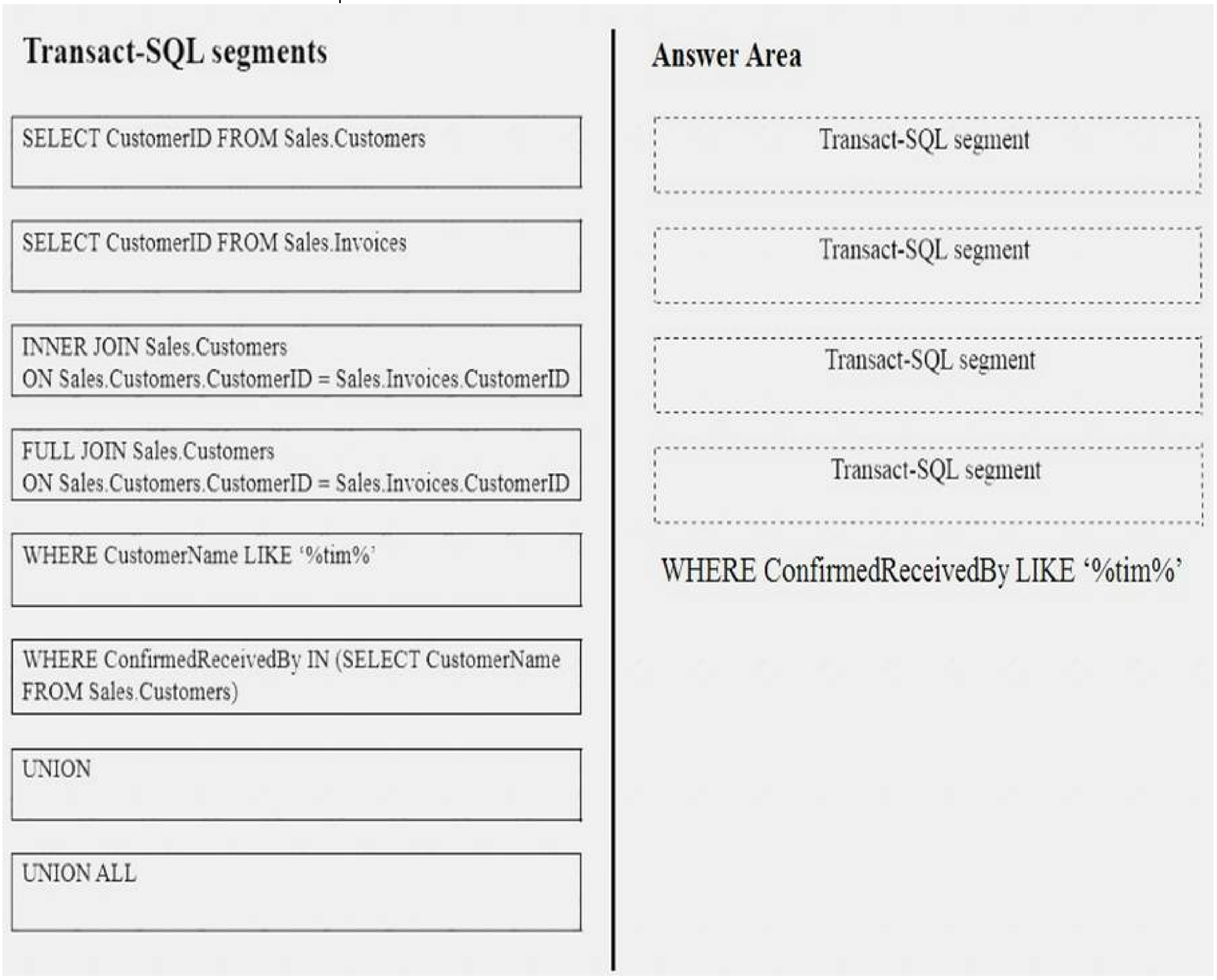


Table	Columns
Sales.Customers	CustomerID, CustomerName
Sales.Invoices	CustomerID, ConfirmedReceivedBy

A delivery person enters an incorrect value for the CustomerID column in the Invoices table and enters the following text in the ConfirmedReceivedBy column: “Package signed for by the owner Tim.”
You need to find the records in the Invoices table that contain the word Tim in the CustomerName field. How should you complete the Transact-SQL statement? 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.



Transact-SQL segments

- SELECT CustomerID FROM Sales.Customers
- SELECT CustomerID FROM Sales.Invoices
- INNER JOIN Sales.Customers
  ON Sales.Customers.CustomerID = Sales.Invoices.CustomerID
- FULL JOIN Sales.Customers
  ON Sales.Customers.CustomerID = Sales.Invoices.CustomerID
- WHERE CustomerName LIKE ‘%tim%’
- WHERE ConfirmedReceivedBy IN (SELECT CustomerName
  FROM Sales.Customers)
- UNION
- UNION ALL

Answer Area

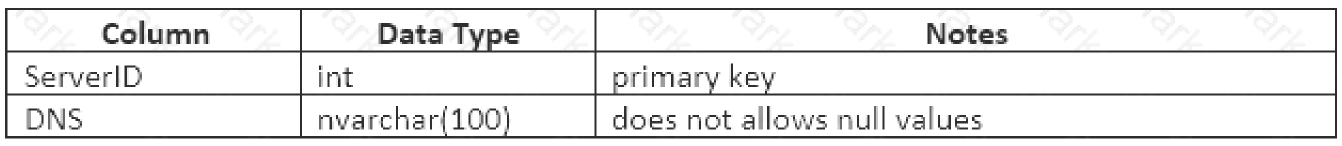
Transact-SQL segment
Transact-SQL segment
Transact-SQL segment
Transact-SQL segment
WHERE ConfirmedReceivedBy LIKE ‘%tim%’

A. Mastered
B. Not Mastered

Answer: A

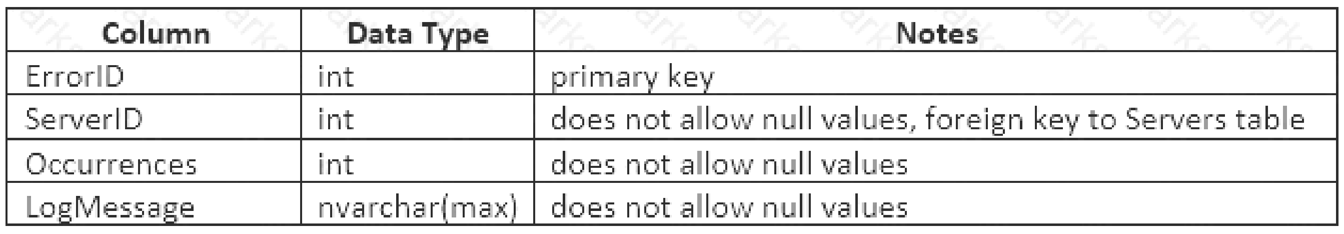
Explanation:
Box 1: SELECT CustomerID FROM Sales.Invoices
Box 2: INNER JOIN Sales.Customers.CustomerID = Sales.Invoices.CustomerID Box 3: WHERE CustomerName LIKE '%tim%'
Box 4: WHERE ConfirmedReceiveBy IN (SELECT CustomerName FROM Sales.Customers)

NEW QUESTION 148
You have a database that stored information about servers and application errors. The database contains the following tables.
Servers



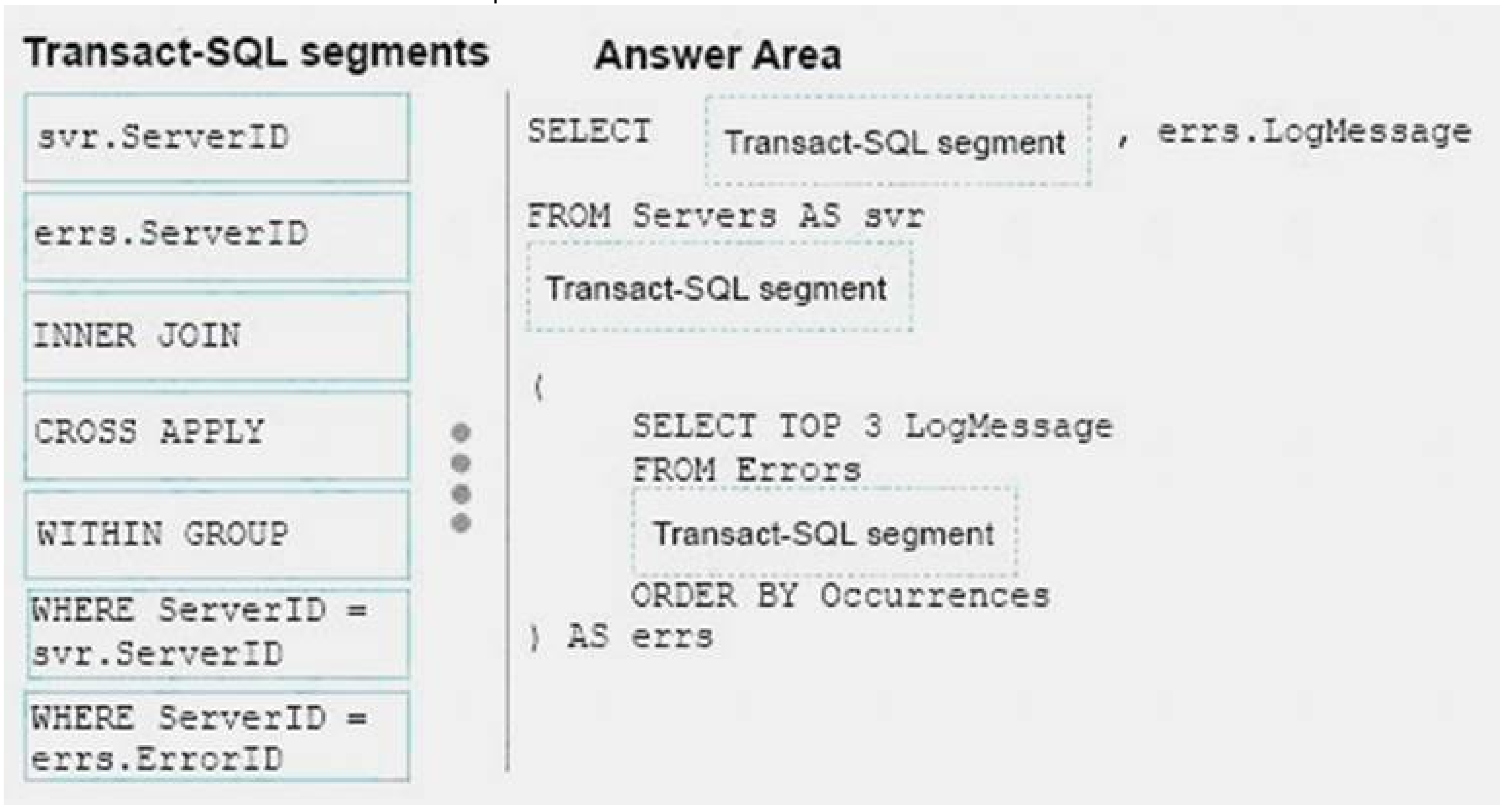
Column	Data Type	Notes
ServerID	int	primary key
DNS	nvarchar(100)	does not allows null values

Errors



Column	Data Type	Notes
ErrorID	int	primary key
ServerID	int	does not allow null values, foreign key to Servers table
Occurrences	int	does not allow null values
LogMessage	nvarchar(max)	does not allow null values

You are building a webpage that shows the three most common errors for each server. You need to return the data for the webpage.
How should you complete the Transact-SQL statement? To answer, drag the appropriate Transact-SQL segments to the correct location. 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.



Transact-SQL segments

- svr.ServerID
- errs.ServerID
- INNER JOIN
- CROSS APPLY
- WITHIN GROUP
- WHERE ServerID = svr.ServerID
- WHERE ServerID = errs.ErrorID

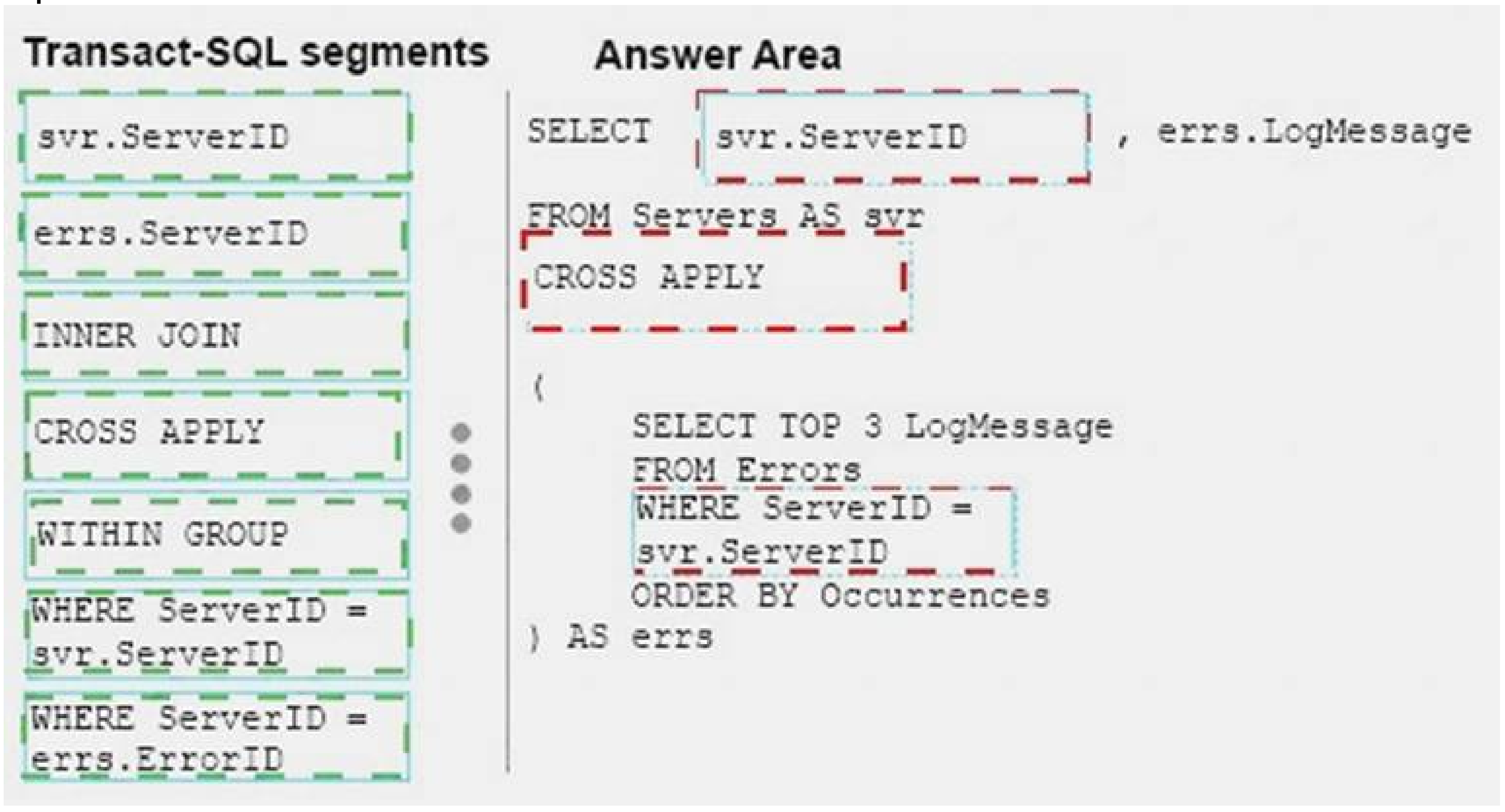
Answer Area

```
SELECT  Transact-SQL segment  , errs.LogMessage
FROM Servers AS svr
Transact-SQL segment
(
      SELECT TOP 3 LogMessage
      FROM Errors
      Transact-SQL segment
      ORDER BY Occurrences
) AS errs
```

A. Mastered
B. Not Mastered

Answer: A

Explanation:



Transact-SQL segments

- svr.ServerID
- errs.ServerID
- INNER JOIN
- CROSS APPLY
- WITHIN GROUP
- WHERE ServerID = svr.ServerID
- WHERE ServerID = errs.ErrorID

Answer Area

```
SELECT  svr.ServerID  , errs.LogMessage
FROM Servers AS svr
CROSS APPLY
(
      SELECT TOP 3 LogMessage
      FROM Errors
      WHERE ServerID =
      svr.ServerID
      ORDER BY Occurrences
) AS errs
```

NEW QUESTION 153
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You have a table named Person that contains information about employees. Users are requesting a way to access specific columns from the Person table without specifying the Person table in the query statement. The columns that users can access will be determined when the query is running against the data. There are some records that are restricted, and a trigger will evaluate whether the request is attempting to access a restricted record.
You need to ensure that users can access the needed columns while minimizing storage on the database server. What should you implement?

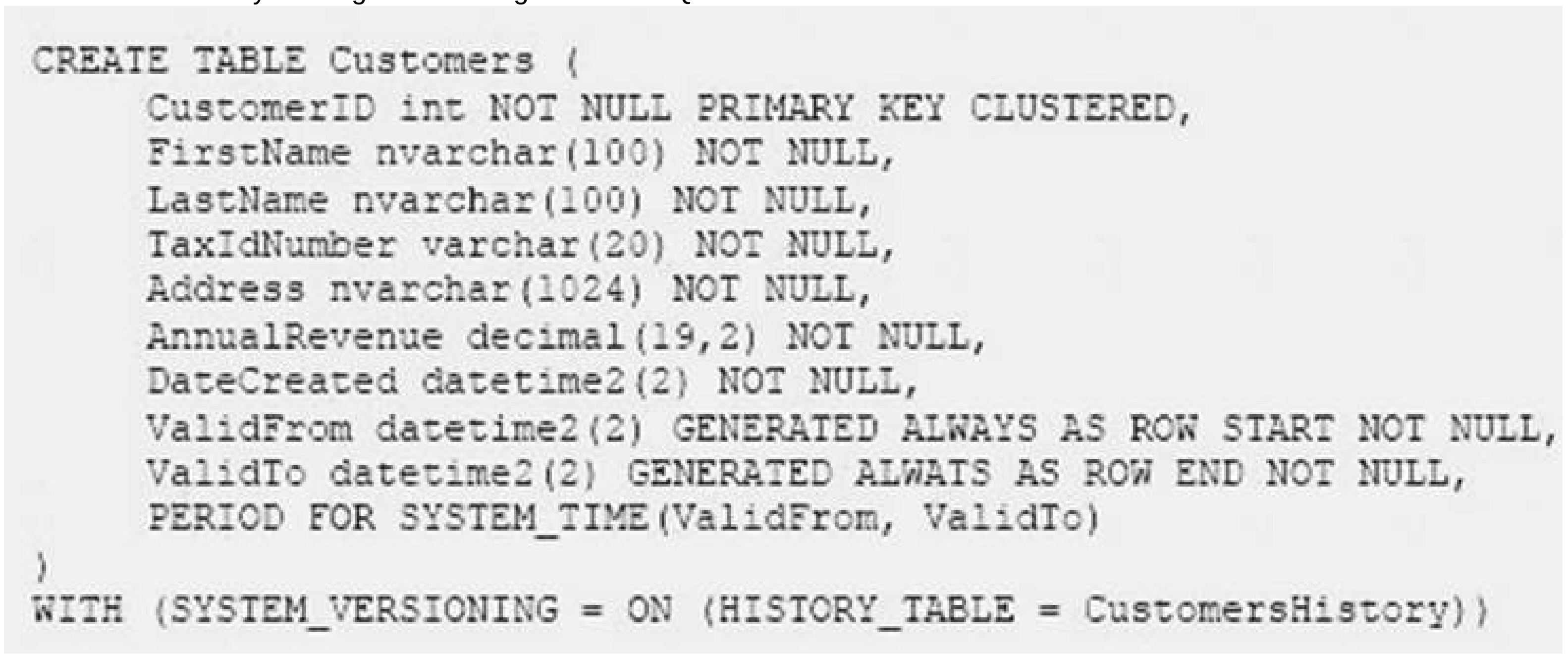
A. the COALESCE function
B. a view

C. a table-valued function
D. the TRY_PARSE function
E. a stored procedure
F. the ISNULL function
G. a scalar function
H. the TRY_CONVERT function

Answer: B

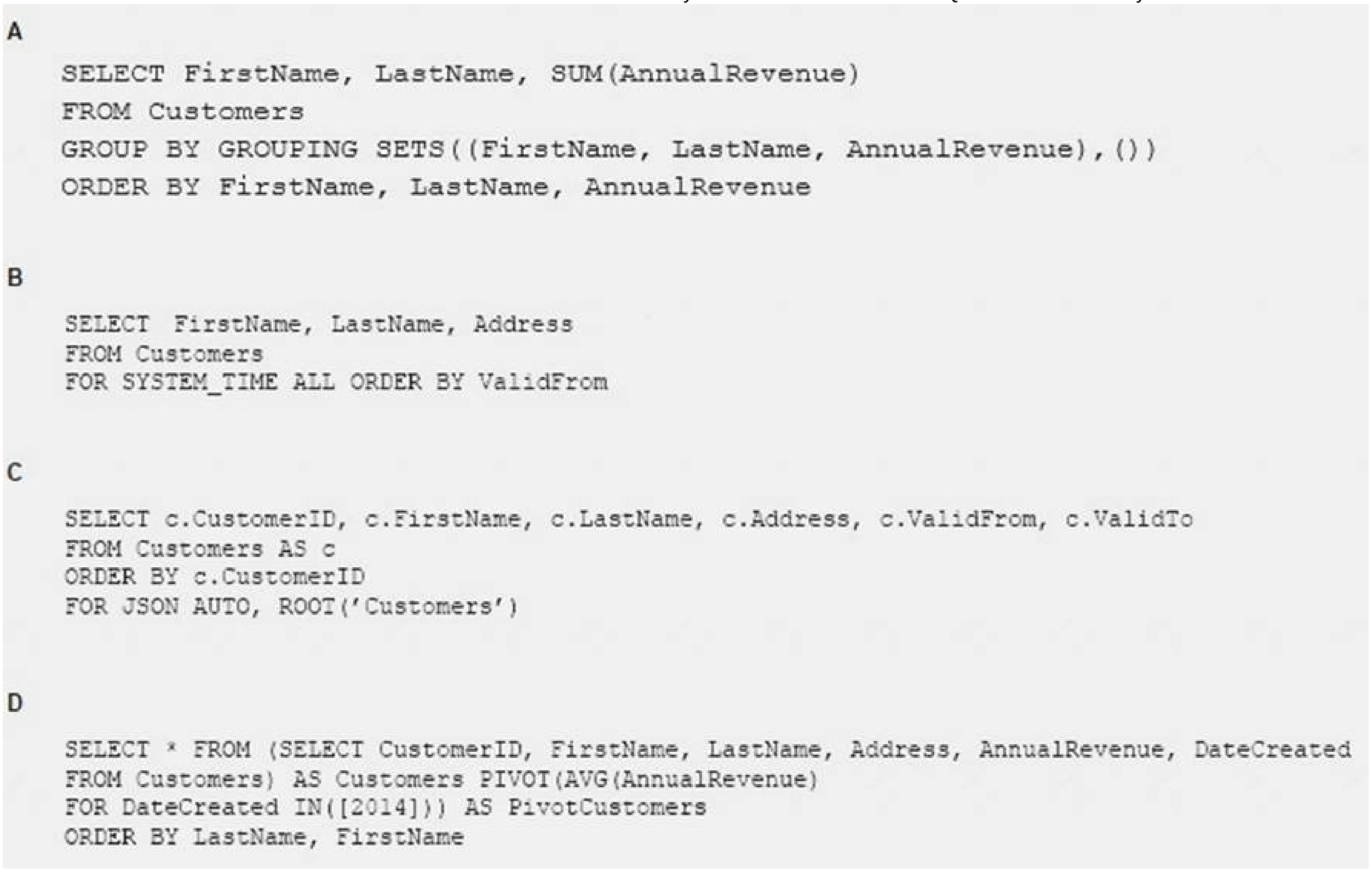
Explanation:
References:
https://docs.microsoft.com/en-us/sql/t-sql/statements/create-view-transact-sql?view=sql-server-2017

NEW QUESTION 156
Note: This question is part of a series of questions that use the same or similar answer choices. An answer
choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You create a table by running the following Transact-SQL statement:



```
CREATE TABLE Customers (
     CustomerID int NOT NULL PRIMARY KEY CLUSTERED,
     FirstName nvarchar(100) NOT NULL,
     LastName nvarchar(100) NOT NULL,
     TaxIdNumber varchar(20) NOT NULL,
     Address nvarchar(1024) NOT NULL,
     AnnualRevenue decimal(19,2) NOT NULL,
     DateCreated datetime2(2) NOT NULL,
     ValidFrom datetime2(2) GENERATED ALWAYS AS ROW START NOT NULL,
     ValidTo datetime2(2) GENERATED ALWATS AS ROW END NOT NULL,
     PERIOD FOR SYSTEM_TIME(ValidFrom, ValidTo)
)
WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE = CustomersHistory))
```

You need to return normalized data for all customers that were added in the year 2014. Which Transact-SQL statement should you run?



A

```
SELECT FirstName, LastName, SUM(AnnualRevenue)
FROM Customers
GROUP BY GROUPING SETS((FirstName, LastName, AnnualRevenue),())
ORDER BY FirstName, LastName, AnnualRevenue
```

B

```
SELECT  FirstName, LastName, Address
FROM Customers
FOR SYSTEM_TIME ALL ORDER BY ValidFrom
```

C

```
SELECT c.CustomerID, c.FirstName, c.LastName, c.Address, c.ValidFrom, c.ValidTo
FROM Customers AS c
ORDER BY c.CustomerID
FOR JSON AUTO, ROOT('Customers')
```

D

```
SELECT * FROM (SELECT CustomerID, FirstName, LastName, Address, AnnualRevenue, DateCreated
FROM Customers) AS Customers PIVOT(AVG(AnnualRevenue)
FOR DateCreated IN([2014])) AS PivotCustomers
ORDER BY LastName, FirstName
```

E

```
SELECT CustomerID, AVG(AnnualRevenue)
AS AverageAnnualRevenue, FirstName, LastName, Address, DateCreated
FROM Customers WHERE YEAR(DateCreated) >= 2014
GROUP BY CustomerID, FirstName, LastName, Address, DateCreated
```

F

```
SELECT c.CustomerID, c.FirstName, c.LastName, c.Address, c.ValidFrom, c.ValidTo
FROM Customers AS c ORDER BY c.CustomerID
FOR XML PATH ('CustomerData'), root ('Customers')
```

G

```
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, ValidFrom, ValidTo
FROM Customers FOR SYSTEM_TIME
BETWEEN '2014-01-01 00:00:00.000000' AND '2015-01-01 00:00:00.000000'
```

H

```
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, ValidFrom, ValidTo
FROM Customers
WHERE DateCreated
BETWEEN '20140101' AND '20141231'
```

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E
F. Option F
G. Option G
H. Option H

Answer: G

NEW QUESTION 158
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a database named DB1 that contains two tables named Sales.Customers and Sales.Orders. Sales.Customers has a foreign key relationship to a column named CustomerID in Sales.Orders.
You need to recommend a query that returns all the customers. The query must also return the number of orders that each customer placed in 2016.
Solution: You recommend the following query:

```
SELECT
     Cust.CustomerName,
     NumberOfOrders = COUNT(Ord.OrderID)
FROM
     Sales.Customers Cust
LEFT JOIN
     Sales.Orders Ord
          ON Cust.CustomerID = Ord.OrderID
GROUP BY
     Cust.CustomerName;
```

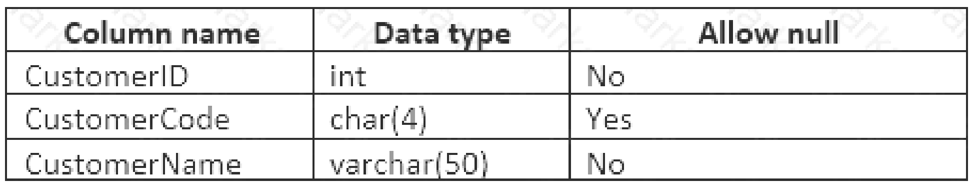
Does this meet the goal?

A. Yes

B. No

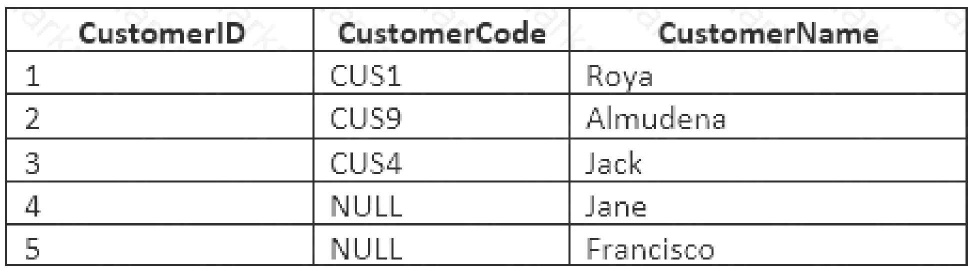
Answer: B

NEW QUESTION 160
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You have a database that contains tables named Customer_CRMSystem and Customer_HRSystem. Both tables use the following structure:



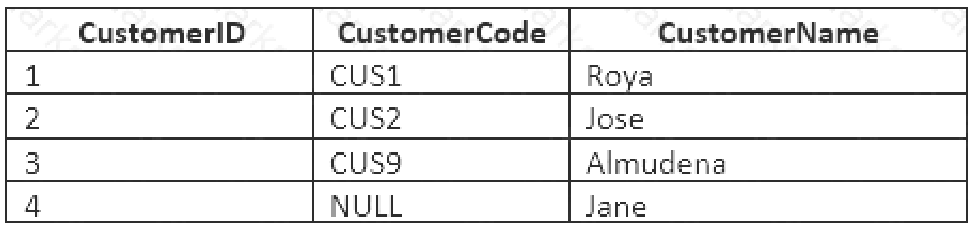
Column name	Data type	Allow null
CustomerID	int	No
CustomerCode	char(4)	Yes
CustomerName	varchar(50)	No

The tables include the following records: Customer_CRMSystem



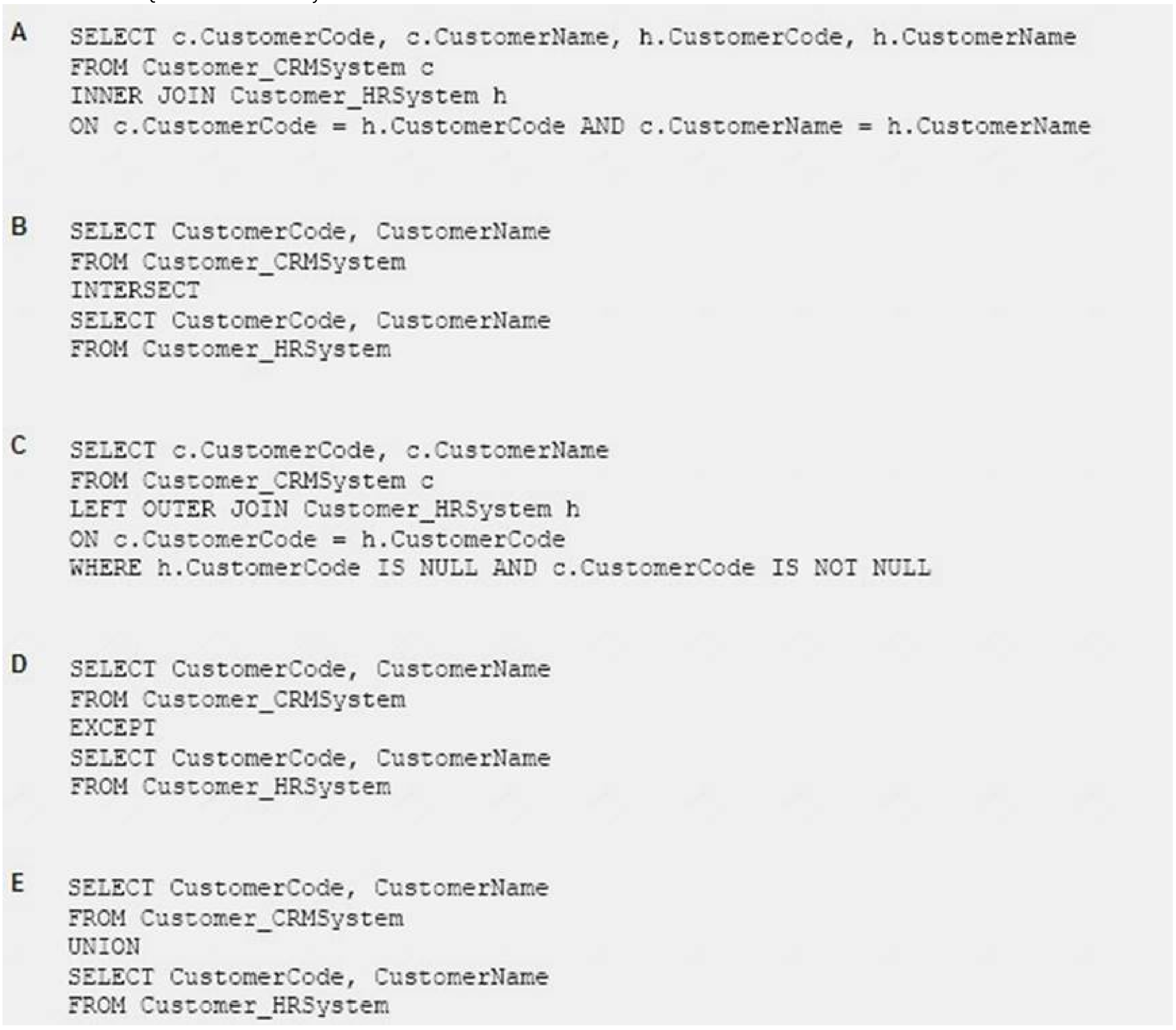
CustomerID	CustomerCode	CustomerName
1	CUS1	Roya
2	CUS9	Almudena
3	CUS4	Jack
4	NULL	Jane
5	NULL	Francisco

Customer_HRSystem



CustomerID	CustomerCode	CustomerName
1	CUS1	Roya
2	CUS2	Jose
3	CUS9	Almudena
4	NULL	Jane

Records that contain null values for CustomerCode can be uniquely identified by CustomerName. You need to display a Cartesian product, combining both tables.
Which Transact-SQL statement should you run?



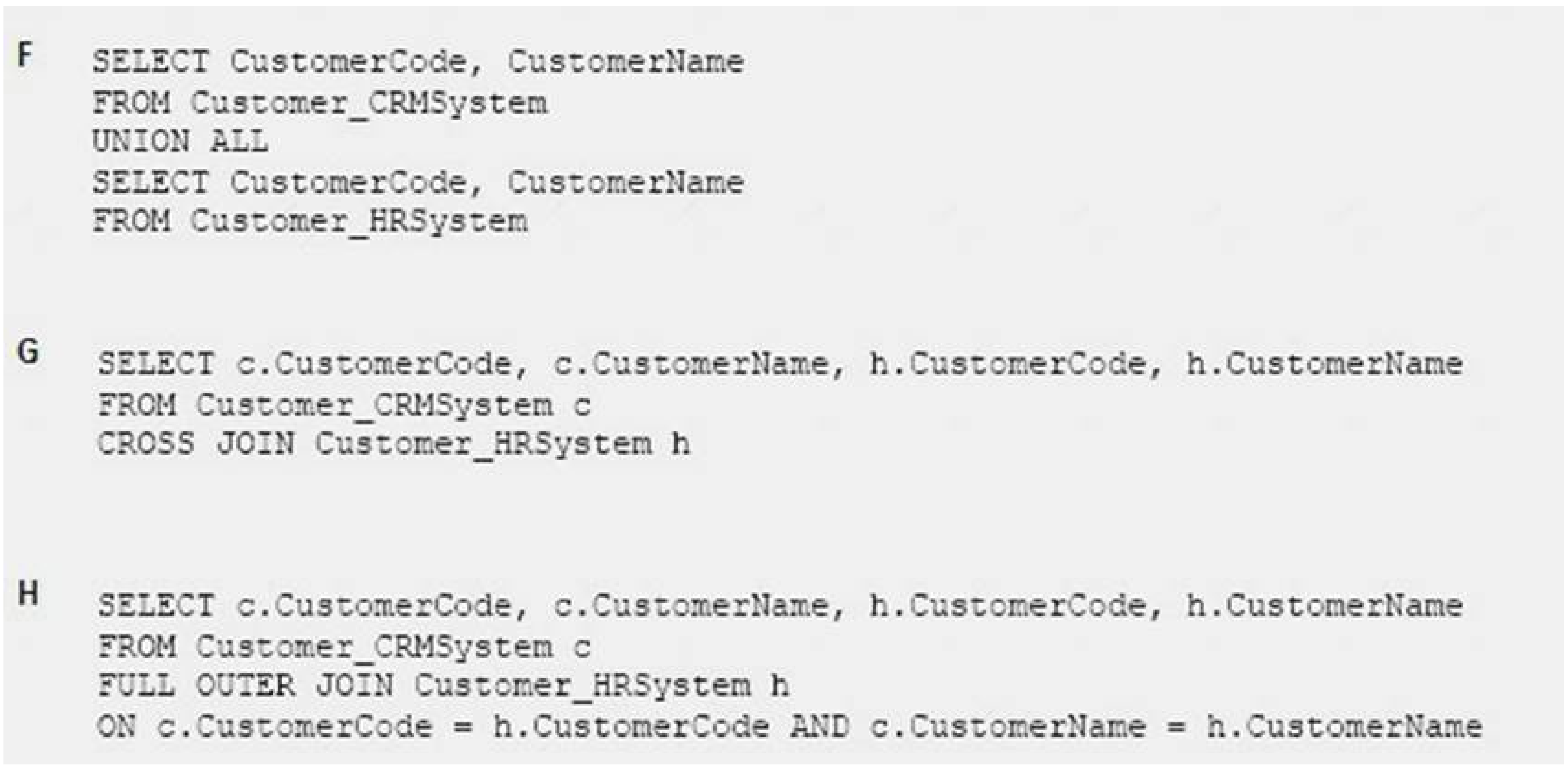
A
```
SELECT c.CustomerCode, c.CustomerName, h.CustomerCode, h.CustomerName
FROM Customer_CRMSystem c
INNER JOIN Customer_HRSystem h
ON c.CustomerCode = h.CustomerCode AND c.CustomerName = h.CustomerName
```

B
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
INTERSECT
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```

C
```
SELECT c.CustomerCode, c.CustomerName
FROM Customer_CRMSystem c
LEFT OUTER JOIN Customer_HRSystem h
ON c.CustomerCode = h.CustomerCode
WHERE h.CustomerCode IS NULL AND c.CustomerCode IS NOT NULL
```

D
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
EXCEPT
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```

E
```
SELECT CustomerCode, CustomerName
FROM Customer_CRMSystem
UNION
SELECT CustomerCode, CustomerName
FROM Customer_HRSystem
```



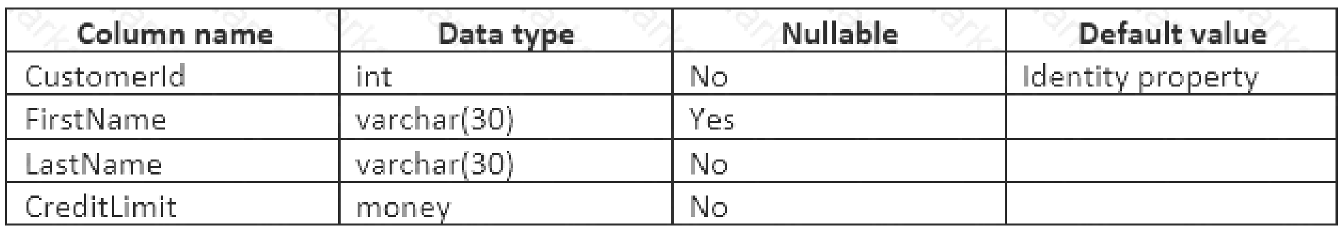
```
F   SELECT CustomerCode, CustomerName
    FROM Customer_CRMSystem
    UNION ALL
    SELECT CustomerCode, CustomerName
    FROM Customer_HRSystem

G   SELECT c.CustomerCode, c.CustomerName, h.CustomerCode, h.CustomerName
    FROM Customer_CRMSystem c
    CROSS JOIN Customer_HRSystem h

H   SELECT c.CustomerCode, c.CustomerName, h.CustomerCode, h.CustomerName
    FROM Customer_CRMSystem c
    FULL OUTER JOIN Customer_HRSystem h
    ON c.CustomerCode = h.CustomerCode AND c.CustomerName = h.CustomerName
```

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E
F. Option F
G. Option G
H. Option H

Answer: G

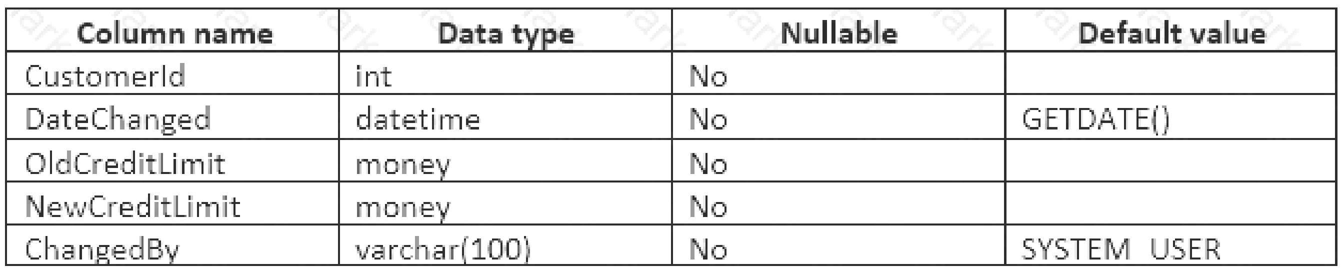
Explanation:
A cross join that does not have a WHERE clause produces the Cartesian product of the tables involved in the join. The size of a Cartesian product result set is the number of rows in the first table multiplied by the number of rows in the second table.
References: https://technet.microsoft.com/en-us/library/ms190690(v=sql.105).aspx

NEW QUESTION 162
You have a database that contains the following tables: Customer



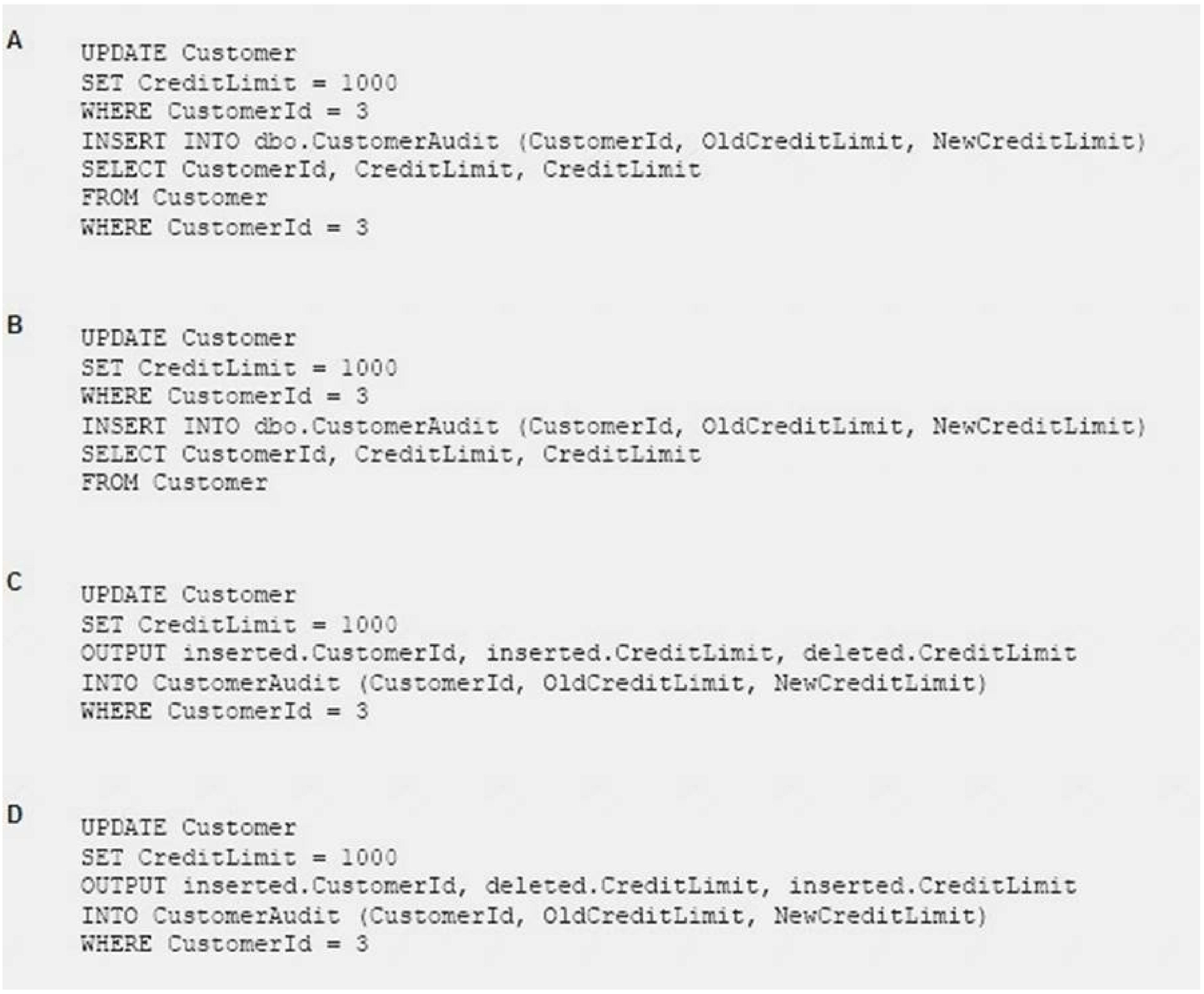
Column name	Data type	Nullable	Default value
CustomerId	int	No	Identity property
FirstName	varchar(30)	Yes	
LastName	varchar(30)	No	
CreditLimit	money	No	

CustomerAudit



Column name	Data type	Nullable	Default value
CustomerId	int	No	
DateChanged	datetime	No	GETDATE()
OldCreditLimit	money	No	
NewCreditLimit	money	No	
ChangedBy	varchar(100)	No	SYSTEM_USER

Where the value of the CustomerID column equals 3, you need to update the value of the CreditLimit column to 1000 for the customer. You must ensure that the change to the record in the Customer table is recorded on the CustomerAudit table.
Which Transact-SQL statement should you run?


A

```
UPDATE Customer
SET CreditLimit = 1000
WHERE CustomerId = 3
INSERT INTO dbo.CustomerAudit (CustomerId, OldCreditLimit, NewCreditLimit)
SELECT CustomerId, CreditLimit, CreditLimit
FROM Customer
WHERE CustomerId = 3
```

B

```
UPDATE Customer
SET CreditLimit = 1000
WHERE CustomerId = 3
INSERT INTO dbo.CustomerAudit (CustomerId, OldCreditLimit, NewCreditLimit)
SELECT CustomerId, CreditLimit, CreditLimit
FROM Customer
```

C

```
UPDATE Customer
SET CreditLimit = 1000
OUTPUT inserted.CustomerId, inserted.CreditLimit, deleted.CreditLimit
INTO CustomerAudit (CustomerId, OldCreditLimit, NewCreditLimit)
WHERE CustomerId = 3
```

D

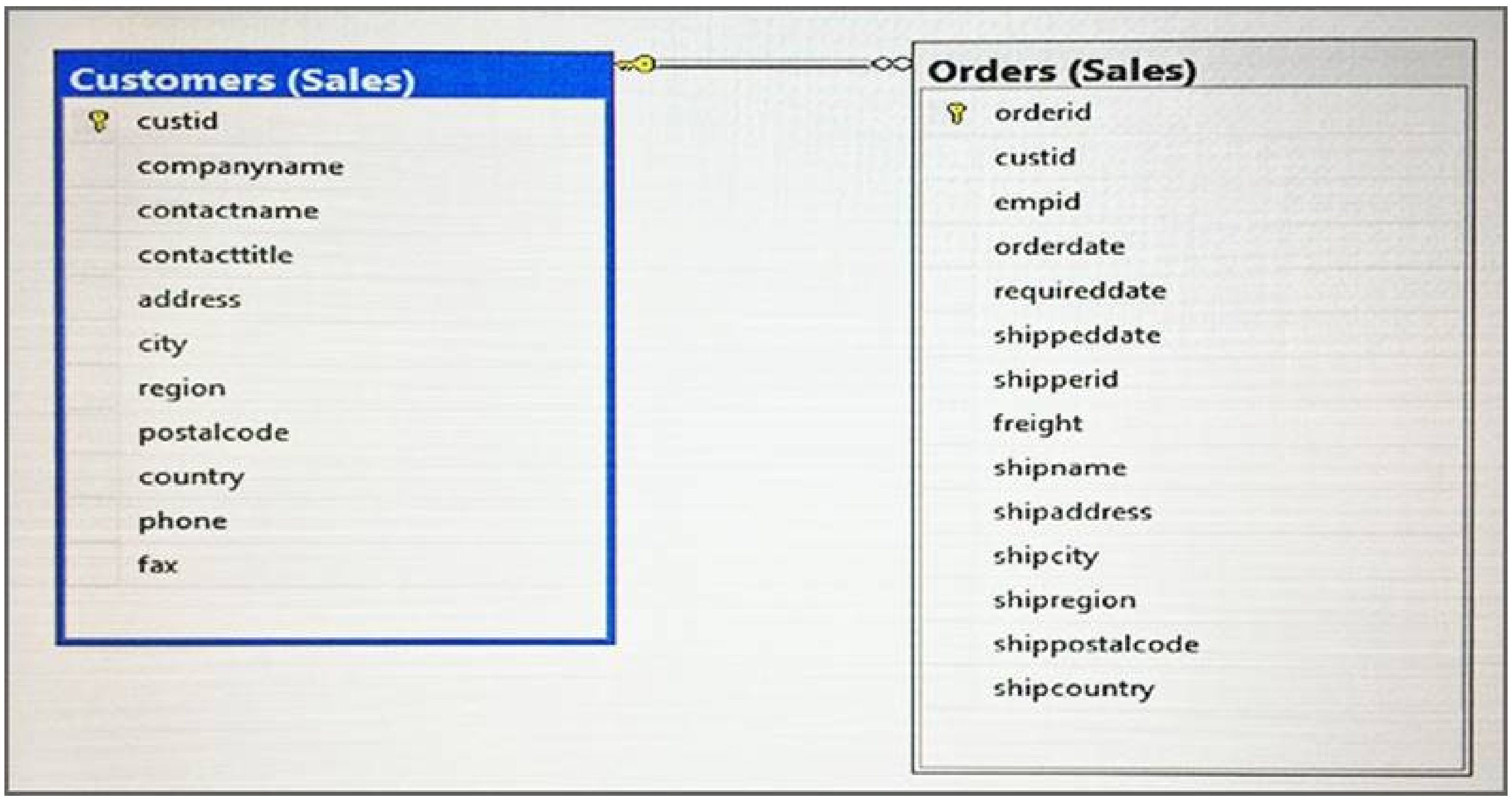
```
UPDATE Customer
SET CreditLimit = 1000
OUTPUT inserted.CustomerId, deleted.CreditLimit, inserted.CreditLimit
INTO CustomerAudit (CustomerId, OldCreditLimit, NewCreditLimit)
WHERE CustomerId = 3
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: D

Explanation:
The OUTPUT Clause returns information from, or expressions based on, each row affected by an INSERT, UPDATE, DELETE, or MERGE statement. These results can be returned to the processing application for use in such things as confirmation messages, archiving, and other such application requirements. The results can also be inserted into a table or table variable. Additionally, you can capture the results of an OUTPUT clause in a nested INSERT, UPDATE, DELETE, or MERGE statement, and insert those results into a target table or view.
Note: If the column modified by the .RITE clause is referenced in an OUTPUT clause, the complete value of the column, either the before image in deleted.column_name or the after image in inserted.column_name, is returned to the specified column in the table variable.

NEW QUESTION 164
You have a database that contains the following tables:



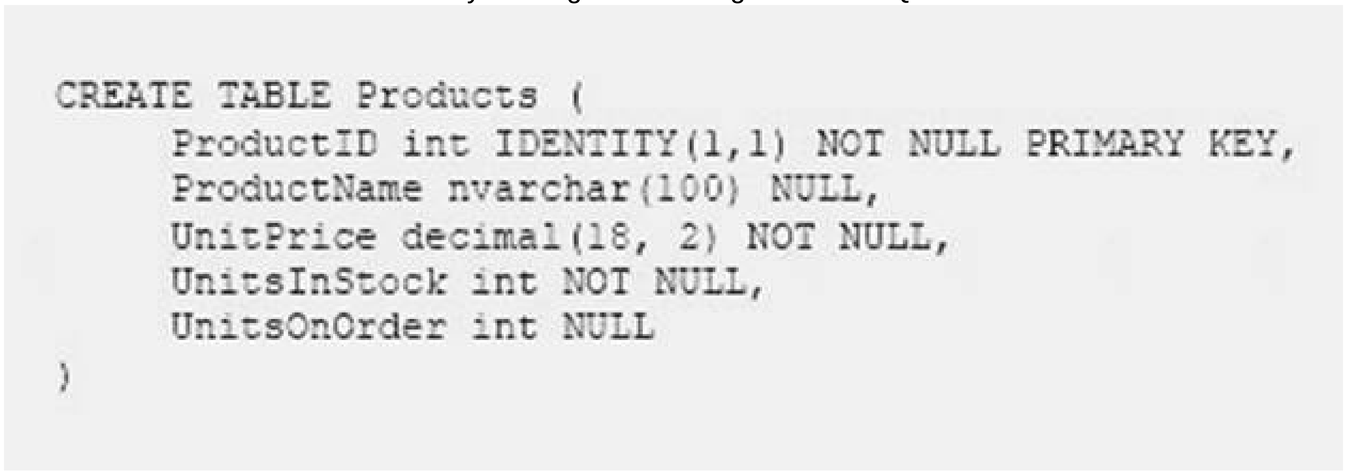
You need to write a query that returns a list of all customers who have not placed orders. Which Transact-SQL statement should you run?

A. SELECT c.custidFROM Sales.Customers c INNER JOIN Sales.Order oON c.custid = o.custid
B. SELECT custid FROM Sales.CustomersINTERSECTSELECT custid FROM Sales.Orders
C. SELECT c.custidFROM Sales.Customers c LEFT OUTER Sales.Order oON c.custid = o.custid
D. SELECT c.custidFROM Sales.Customers c LEFT OUTER JOIN Sales.Order o ON c.custid = o.custidWHERE orderid IS NULL

Answer: D

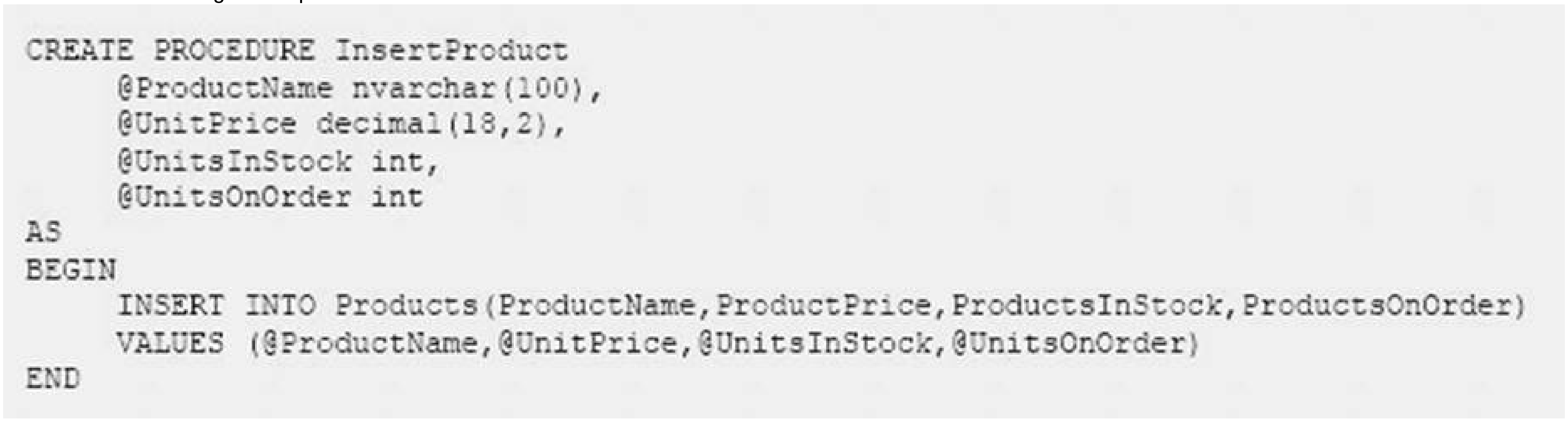
Explanation:
Inner joins return rows only when there is at least one row from both tables that matches the join condition. Inner joins eliminate the rows that do not match with a row from the other table. Outer joins, however, return all rows from at least one of the tables or views mentioned in the FROM clause, as long as those rows meet any WHERE or HAVING search conditions. All rows are retrieved from the left table referenced with a left outer join, and all rows from the right table referenced in a right outer join. All rows from both tables are returned in a full outer join.
References: https://technet.microsoft.com/en-us/library/ms187518(v=sql.105).aspx

NEW QUESTION 166
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You create a table named Products by running the following Transact-SQL statement:



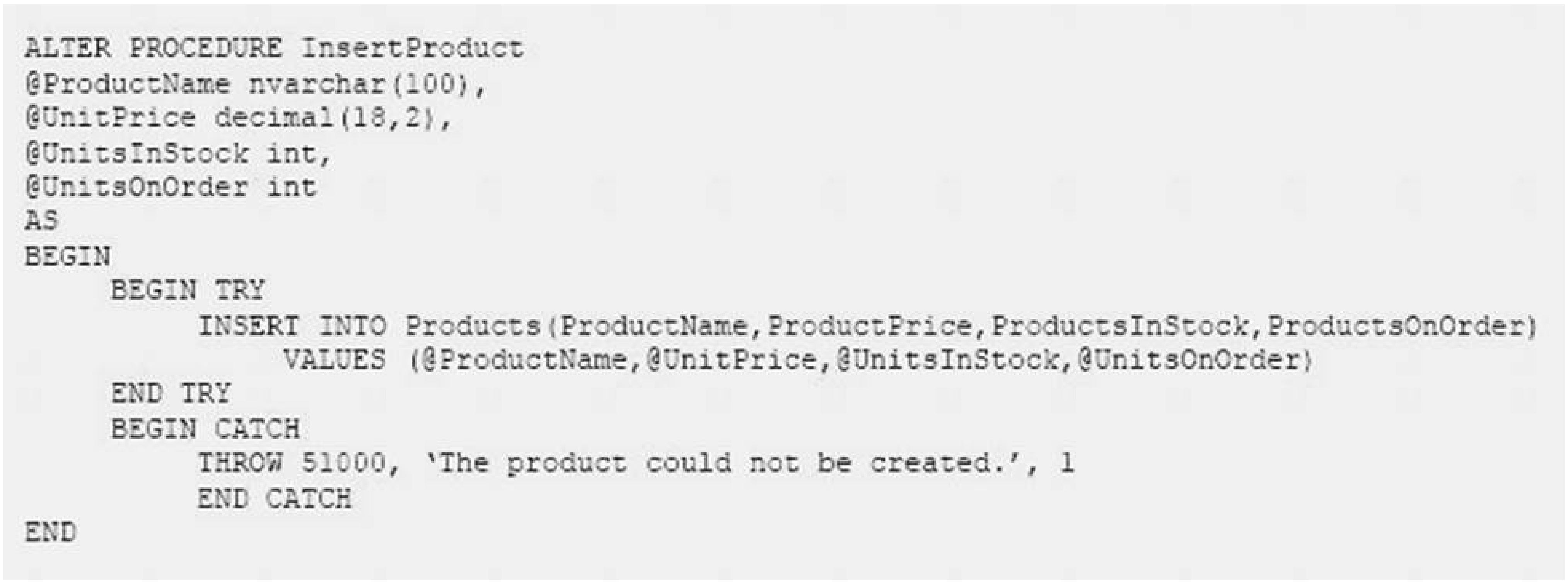
```
CREATE TABLE Products (
     ProductID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
     ProductName nvarchar(100) NULL,
     UnitPrice decimal(18, 2) NOT NULL,
     UnitsInStock int NOT NULL,
     UnitsOnOrder int NULL
)
```

You have the following stored procedure:



```
CREATE PROCEDURE InsertProduct
     @ProductName nvarchar(100),
     @UnitPrice decimal(18,2),
     @UnitsInStock int,
     @UnitsOnOrder int
AS
BEGIN
     INSERT INTO Products(ProductName,ProductPrice,ProductsInStock,ProductsOnOrder)
     VALUES (@ProductName,@UnitPrice,@UnitsInStock,@UnitsOnOrder)
END
```

You need to modify the stored procedure to meet the following new requirements:
- Insert product records as a single unit of work.
- Return error number 51000 when a product fails to insert into the database.
- If a product record insert operation fails, the product information must not be permanently written to the database.
Solution: You run the following Transact-SQL statement:



```
ALTER PROCEDURE InsertProduct
@ProductName nvarchar(100),
@UnitPrice decimal(18,2),
@UnitsInStock int,
@UnitsOnOrder int
AS
BEGIN
     BEGIN TRY
          INSERT INTO Products(ProductName,ProductPrice,ProductsInStock,ProductsOnOrder)
               VALUES (@ProductName,@UnitPrice,@UnitsInStock,@UnitsOnOrder)
     END TRY
     BEGIN CATCH
          THROW 51000, 'The product could not be created.', 1
          END CATCH
END
```

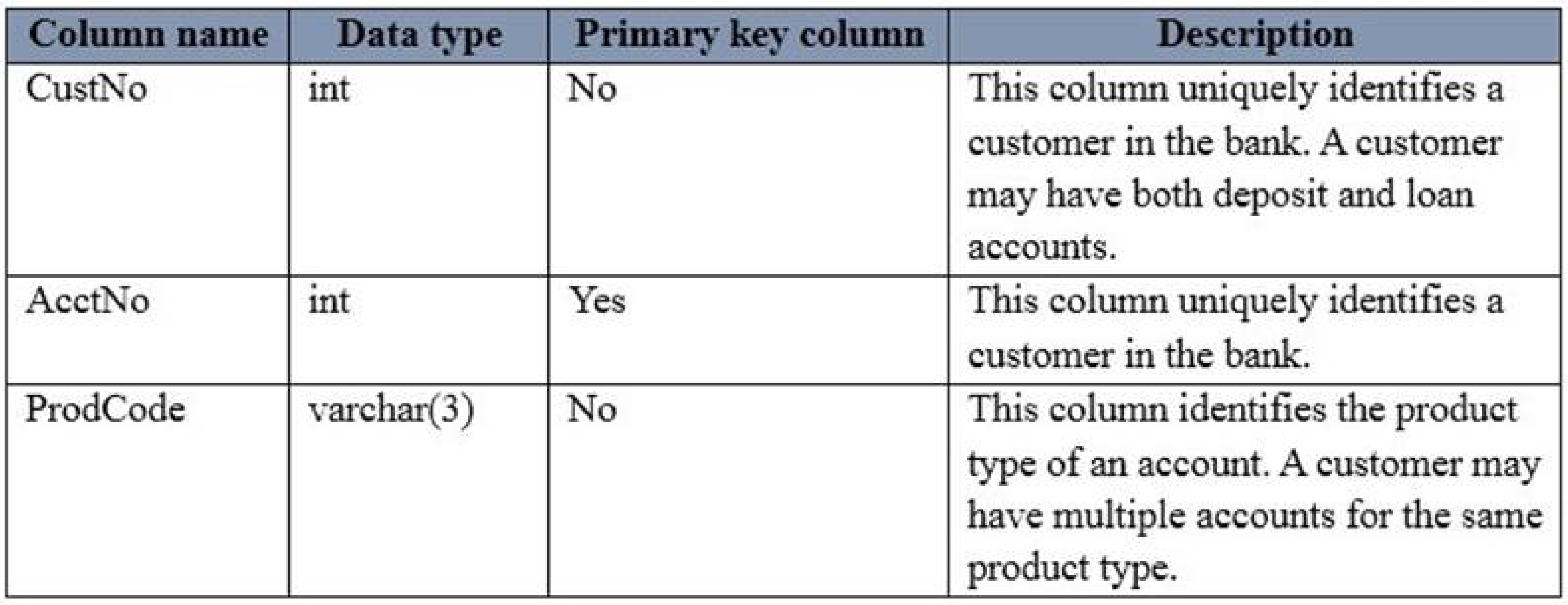
Does the solution meet the goal?

A. Yes
B. No

Answer: A

Explanation:
If the INSERT INTO statement raises an error, the statement will be caught and an error 51000 will be thrown. In this case no records will have been inserted.
Note:
You can implement error handling for the INSERT statement by specifying the statement in a TRY…CATCH construct.
If an INSERT statement violates a constraint or rule, or if it has a value incompatible with the data type of the column, the statement fails and an error message is returned.
References: https://msdn.microsoft.com/en-us/library/ms174335.aspx

NEW QUESTION 168
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply to that question.
You have a database for a banking system. The database has two tables named tblDepositAcct and tblLoanAcct that store deposit and loan accounts, respectively/ Both tables contain the following columns:



Column name	Data type	Primary key column	Description
CustNo	int	No	This column uniquely identifies a customer in the bank. A customer may have both deposit and loan accounts.
AcctNo	int	Yes	This column uniquely identifies a customer in the bank.
ProdCode	varchar(3)	No	This column identifies the product type of an account. A customer may have multiple accounts for the same product type.

You need to run a query to find the total number of customers who have both deposit and loan accounts. Which Transact-SQL statement should you run?

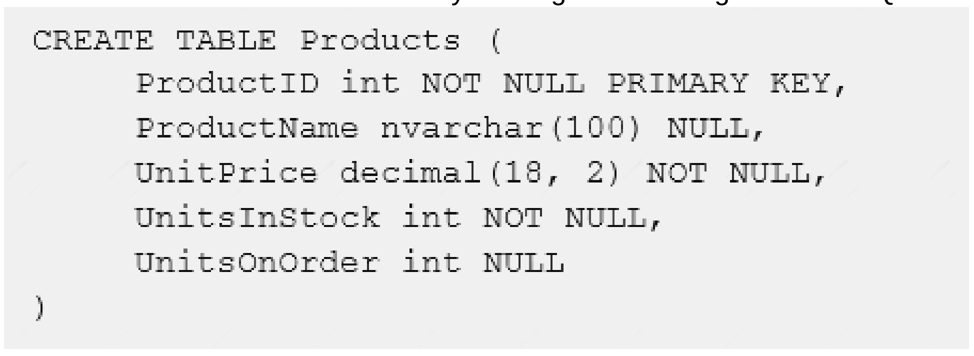
A. SELECT COUNT(*)FROM (SELECT AcctNoFROM tblDepositAcctINTERSECTSELECTAcctNoFROM tblLoanAcct) R
B. SELECT COUNT(*)FROM (SELECT CustNoFROM tblDepositAcctUNIONSELECT CustNoFROMtblLoanAcct) R
C. SELECT COUNT(*)FROM (SELECT CustNoFROM tblDepositAcctUNION ALLSELECTCustNoFROM tblLoanAcct) R
D. SELECT COUNT (DISTINCT D.CustNo)FROM tblDepositAcct D, tblLoanAcct LWHERE D.CustNo= L.CustNo
E. SELECT COUNT(DISTINCT L.CustNo)FROM tblDepositAcct DRIGHT JOIN tblLoanAcct L ON D.CustNo = L.CustNoWHERE D.CustNo IS NULL
F. SELECT COUNT(*)FROM (SELECT CustNoFROM tblDepositAcctEXCEPTSELECT CustNoFROMtblLoanAcct) R
G. SELECT COUNT (DISTINCT COALESCE(D.CustNo, L.CustNo))FROM tblDepositAcct DFULLJOIN tblLoanAcct L ON D.CustNo = L.CustNoWHERE D.CustNo IS NULL OR L.CustNo IS NULL
H. SELECT COUNT(*)FROM tblDepositAcct DFULL JOIN tblLoanAcct L ON D.CustNo = L.CustNo

Answer: A

Explanation:
The SQL INTERSECT operator is used to return the results of 2 or more SELECT statements. However, it only returns the rows selected by all queries or data sets. If a record exists in one query and not in the other, it will be omitted from the INTERSECT results.
References: https://www.techonthenet.com/sql/intersect.php

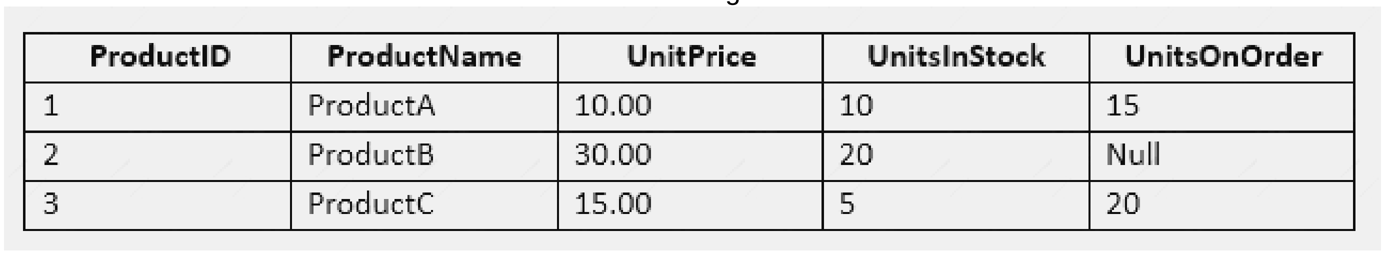
NEW QUESTION 173

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a table that was created by running the following Transact-SQL statement:



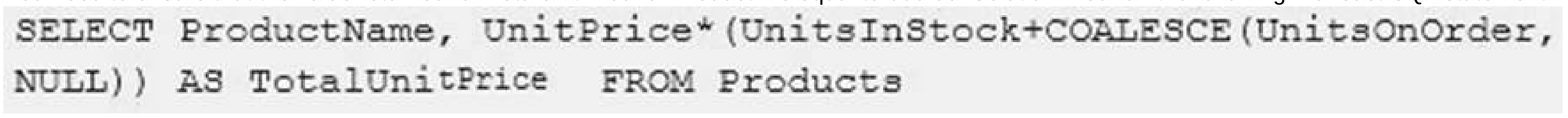
```
CREATE TABLE Products (
      ProductID int NOT NULL PRIMARY KEY,
      ProductName nvarchar(100) NULL,
      UnitPrice decimal(18, 2) NOT NULL,
      UnitsInStock int NOT NULL,
      UnitsOnOrder int NULL
)
```

The Products table includes the data shown in the following table:



ProductID	ProductName	UnitPrice	UnitsInStock	UnitsOnOrder
1	ProductA	10.00	10	15
2	ProductB	30.00	20	Null
3	ProductC	15.00	5	20

TotalUnitPrice is calculated by using the following formula: TotalUnitPrice = UnitPrice * (UnitsInStock + UnitsOnOrder)
You need to ensure that the value returned for TotalUnitPrice for ProductB is equal to 600.00. Solution: You run the following Transact-SQL statement:



```
SELECT ProductName, UnitPrice*(UnitsInStock+COALESCE(UnitsOnOrder,
NULL)) AS TotalUnitPrice  FROM Products
```

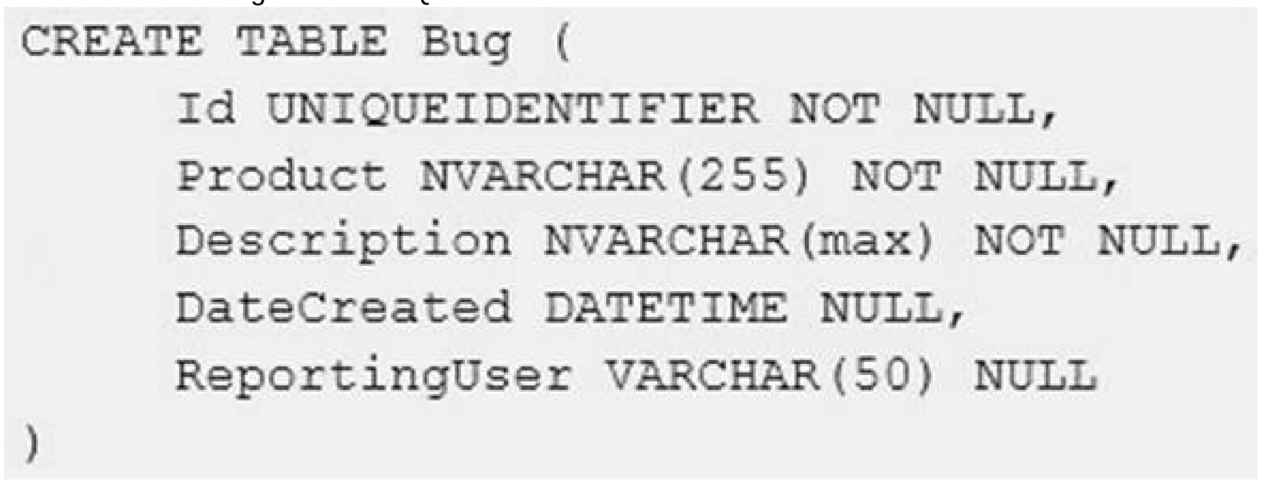
Does the solution meet the goal?

A. Yes
B. No

Answer: B

NEW QUESTION 177
You have a project management application. The application uses a Microsoft SQL Server database to store data. You are developing a software bug tracking add-on for the application.
The add-on must meet the following requirements:
- Allow case sensitive searches for product.
- Filter search results based on exact text in the description.
- Support multibyte Unicode characters.

You run the following Transact-SQL statement:



```
CREATE TABLE Bug (
      Id UNIQUEIDENTIFIER NOT NULL,
      Product NVARCHAR(255) NOT NULL,
      Description NVARCHAR(max) NOT NULL,
      DateCreated DATETIME NULL,
      ReportingUser VARCHAR(50) NULL
)
```

Users connect to an instance of the bug tracking application that is hosted in New York City. Users in Seattle must be able to display the local date and time for any bugs that they create.
You need to ensure that the DateCreated column displays correctly. Which Transact-SQL statement should you run?

A. SELECT Id,Product,DateCreated AT TIME ZONE 'Pacific Standard Time' FROM Bug
B. SELECT Id,Product, DATEADD(hh, -8, DateCreated) FROM Bug
C. SELECT Id,Product, TODATETIMEOFFSET(DateCreated, -8) FROM Bug
D. SELECT Id,Product,CAST(DateCreated AS DATETIMEOFFSET) FROM Bug

Answer: C

Explanation:
References:
https://docs.microsoft.com/en-us/sql/t-sql/functions/todatetimeoffset-transact-sql?view=sql-server-2017

NEW QUESTION 180
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)	does not allow null values
PhoneNumber	nvarchar(20)	does not allow null values
AccountOpenedDate	date	does not allow null values
StandardDiscountPercentage	decimal(18,3)	does not allow null values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	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.

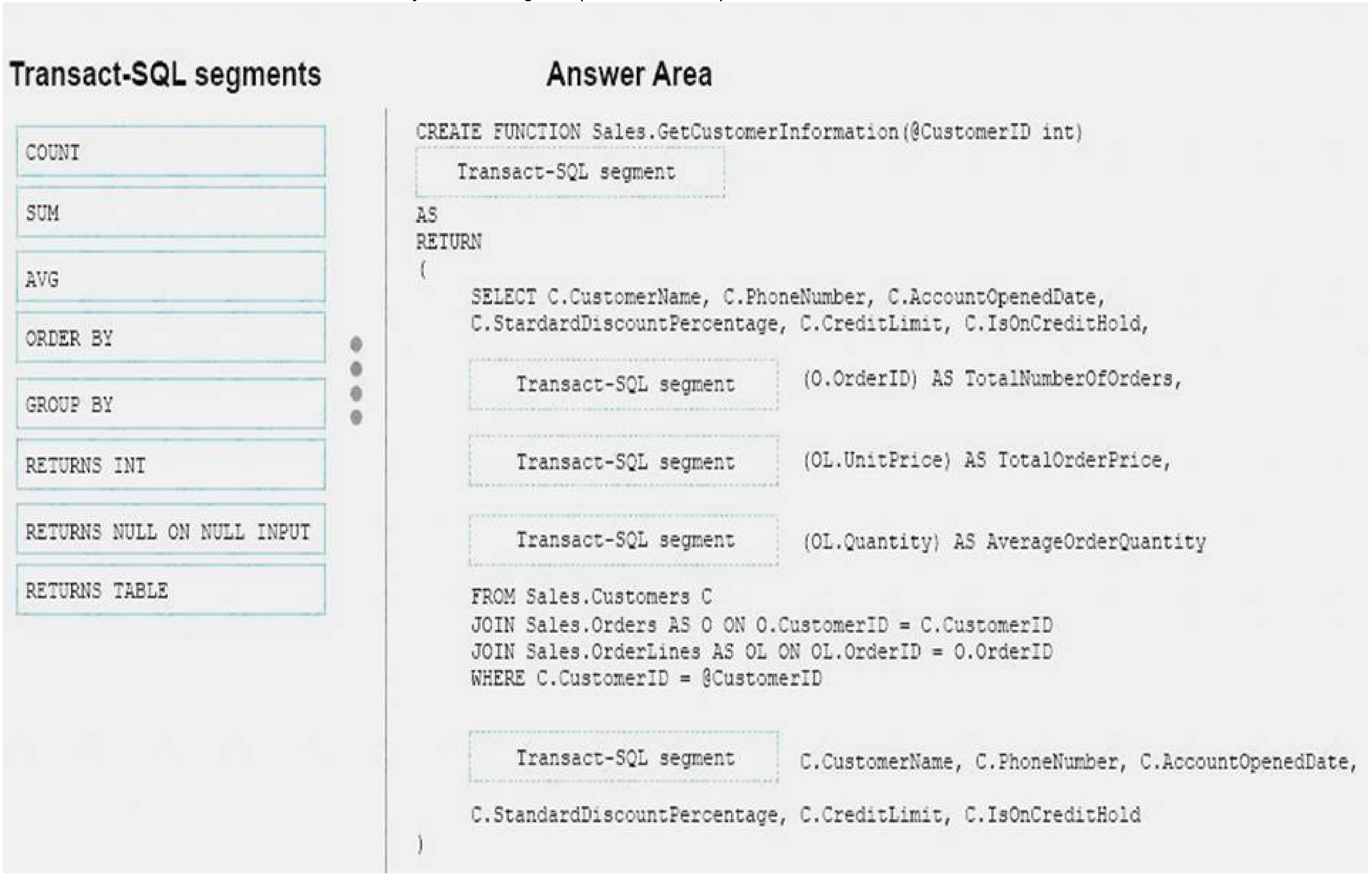
Column name	Data type	Constraints
OrderID	int	primary key
CustomerID	int	foreign key to the Sales.Customers table
OrderDate	date	does not allow null values

The following table describes the columns in Sales.OrderLines.

Column name	Data type	Constraints
OrderLineID	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.



Transact-SQL segments

- COUNT
- SUM
- AVG
- ORDER BY
- GROUP BY
- RETURNS INT
- RETURNS NULL ON NULL INPUT
- RETURNS TABLE

Answer Area

```
CREATE FUNCTION Sales.GetCustomerInformation(@CustomerID int)
  Transact-SQL segment
AS
RETURN
(
    SELECT C.CustomerName, C.PhoneNumber, C.AccountOpenedDate,
    C.StardardDiscountPercentage, C.CreditLimit, C.IsOnCreditHold,
      Transact-SQL segment  (O.OrderID) AS TotalNumberOfOrders,
      Transact-SQL segment  (OL.UnitPrice) AS TotalOrderPrice,
      Transact-SQL segment  (OL.Quantity) AS AverageOrderQuantity
    FROM Sales.Customers C
    JOIN Sales.Orders AS O ON O.CustomerID = C.CustomerID
    JOIN Sales.OrderLines AS OL ON OL.OrderID = O.OrderID
    WHERE C.CustomerID = @CustomerID

      Transact-SQL segment  C.CustomerName, C.PhoneNumber, C.AccountOpenedDate,

    C.StandardDiscountPercentage, C.CreditLimit, C.IsOnCreditHold
)
```

A. Mastered
B. Not Mastered

Answer: A

Explanation:
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

NEW QUESTION 184
You have a date related query that would benefit from an indexed view. You need to create the indexed view.
Which two Transact-SQL functions can you use? Each correct answer presents a complete solution. NOTE: Each correct selection is worth one point

A. DATEADD
B. AT TIME ZONE
C. GETUTCDATE
D. DATEDIFF

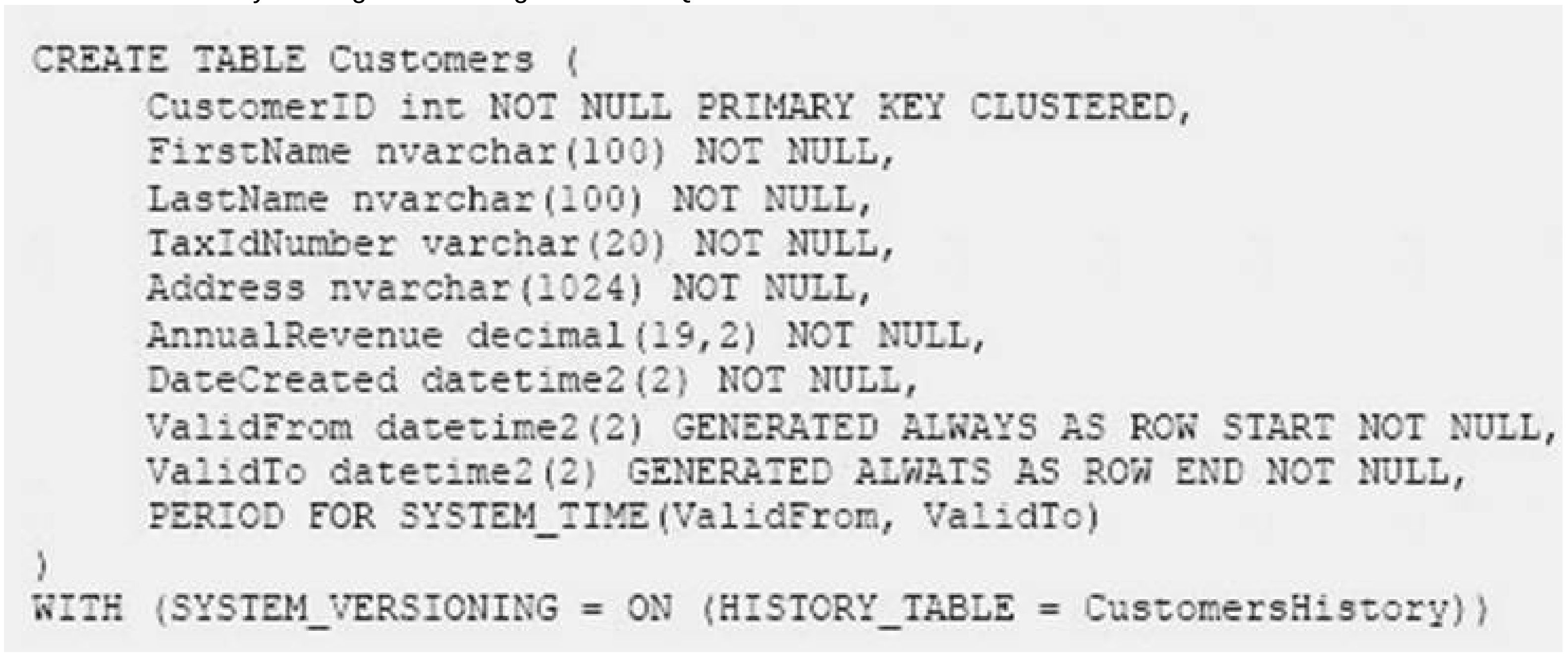
Answer: A

NEW QUESTION 186
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You are creating indexes in a data warehouse.
You have a dimension table named Table1 that has 10,000 rows. The rows are used to generate several reports. The reports join a column that is the primary key.
The execution plan contains bookmark lookups for Table1. You discover that the reports run slower than expected.
You need to reduce the amount of time it takes to run the reports. Solution: You create a clustered index on the primary key column. Does this meet the goal?

A. Yes
B. No

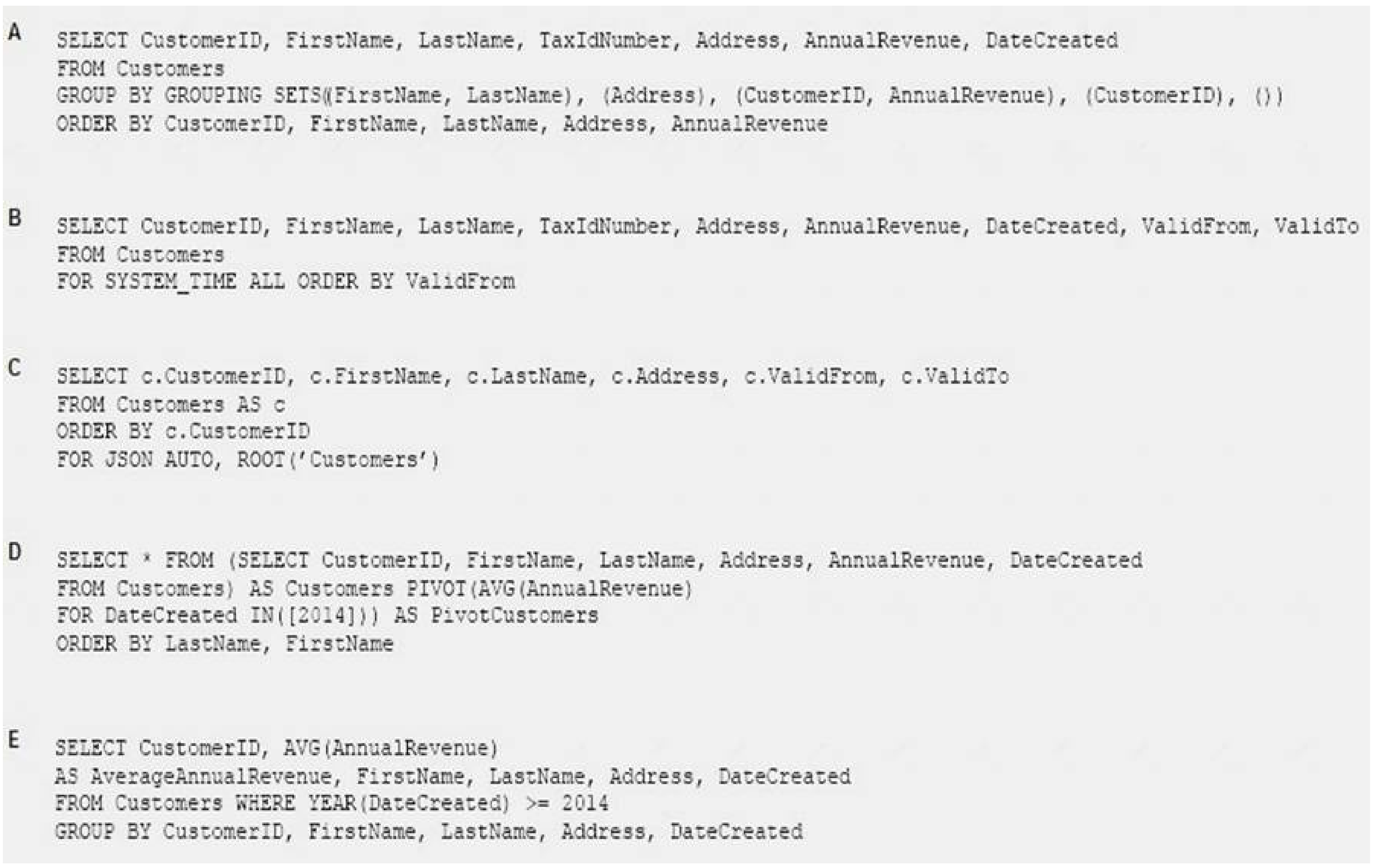
Answer: A

NEW QUESTION 187
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You create a table by running the following Transact-SQL statement:



```
CREATE TABLE Customers (
     CustomerID int NOT NULL PRIMARY KEY CLUSTERED,
     FirstName nvarchar(100) NOT NULL,
     LastName nvarchar(100) NOT NULL,
     TaxIdNumber varchar(20) NOT NULL,
     Address nvarchar(1024) NOT NULL,
     AnnualRevenue decimal(19,2) NOT NULL,
     DateCreated datetime2(2) NOT NULL,
     ValidFrom datetime2(2) GENERATED ALWAYS AS ROW START NOT NULL,
     ValidTo datetime2(2) GENERATED ALWAYS AS ROW END NOT NULL,
     PERIOD FOR SYSTEM_TIME(ValidFrom, ValidTo)
)
WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE = CustomersHistory))
```

You are developing a report that aggregates customer data only for the year 2014. The report requires that the data be denormalized.
You need to return the data for the report. Which Transact-SQL statement should you run?



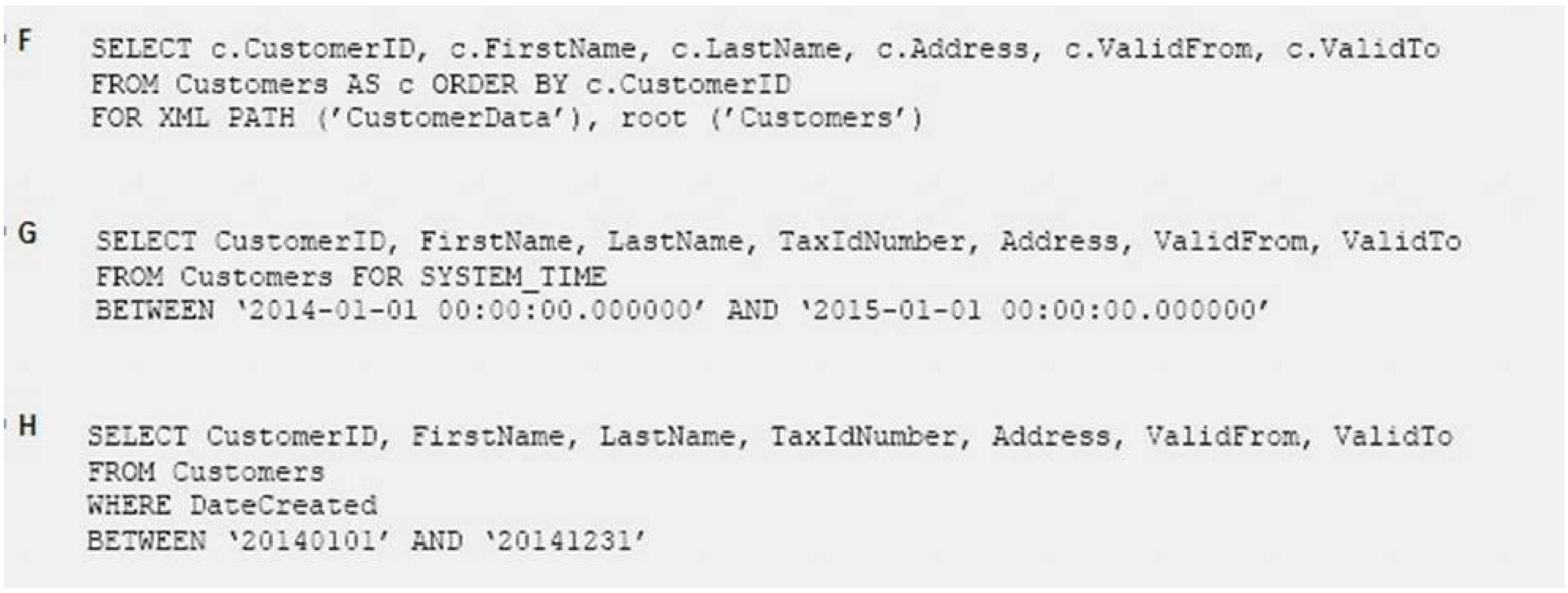
A
```
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, AnnualRevenue, DateCreated
FROM Customers
GROUP BY GROUPING SETS((FirstName, LastName), (Address), (CustomerID, AnnualRevenue), (CustomerID), ())
ORDER BY CustomerID, FirstName, LastName, Address, AnnualRevenue
```

B
```
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, AnnualRevenue, DateCreated, ValidFrom, ValidTo
FROM Customers
FOR SYSTEM_TIME ALL ORDER BY ValidFrom
```

C
```
SELECT c.CustomerID, c.FirstName, c.LastName, c.Address, c.ValidFrom, c.ValidTo
FROM Customers AS c
ORDER BY c.CustomerID
FOR JSON AUTO, ROOT('Customers')
```

D
```
SELECT * FROM (SELECT CustomerID, FirstName, LastName, Address, AnnualRevenue, DateCreated
FROM Customers) AS Customers PIVOT(AVG(AnnualRevenue)
FOR DateCreated IN([2014])) AS PivotCustomers
ORDER BY LastName, FirstName
```

E
```
SELECT CustomerID, AVG(AnnualRevenue)
AS AverageAnnualRevenue, FirstName, LastName, Address, DateCreated
FROM Customers WHERE YEAR(DateCreated) >= 2014
GROUP BY CustomerID, FirstName, LastName, Address, DateCreated
```



F
```
SELECT c.CustomerID, c.FirstName, c.LastName, c.Address, c.ValidFrom, c.ValidTo
FROM Customers AS c ORDER BY c.CustomerID
FOR XML PATH ('CustomerData'), root ('Customers')
```

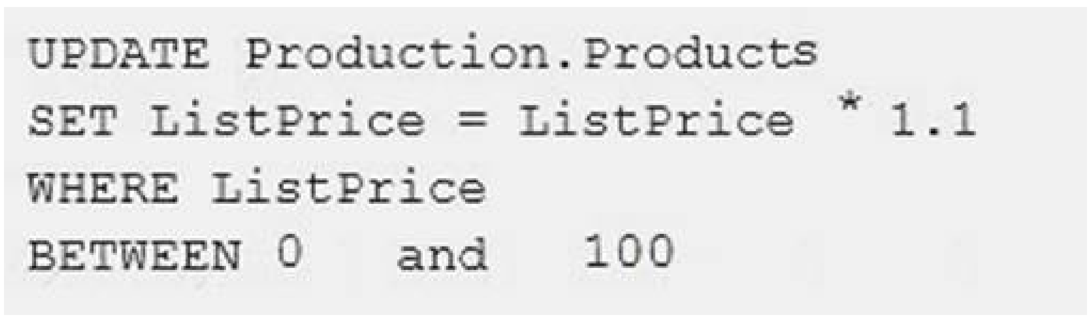
G
```
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, ValidFrom, ValidTo
FROM Customers FOR SYSTEM_TIME
BETWEEN '2014-01-01 00:00:00.000000' AND '2015-01-01 00:00:00.000000'
```

H
```
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, ValidFrom, ValidTo
FROM Customers
WHERE DateCreated
BETWEEN '20140101' AND '20141231'
```

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E
F. Option F
G. Option G
H. Option H

Answer: G

NEW QUESTION 191
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section. You will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You have a table named Products that stores information about products your company sells. The table has a column named ListPrice that stores retail pricing information for products.
Some products are used only internally by the company. Records for these products are maintained in the Products table for inventory purposes. The price for each of these products is $0.00. Customers are not permitted to order these products.
You need to increase the list price for products that cost less than $100 by 10 percent. You must only increase pricing for products that customers are permitted to order.
Solution: You run the following Transact-SQL statement:



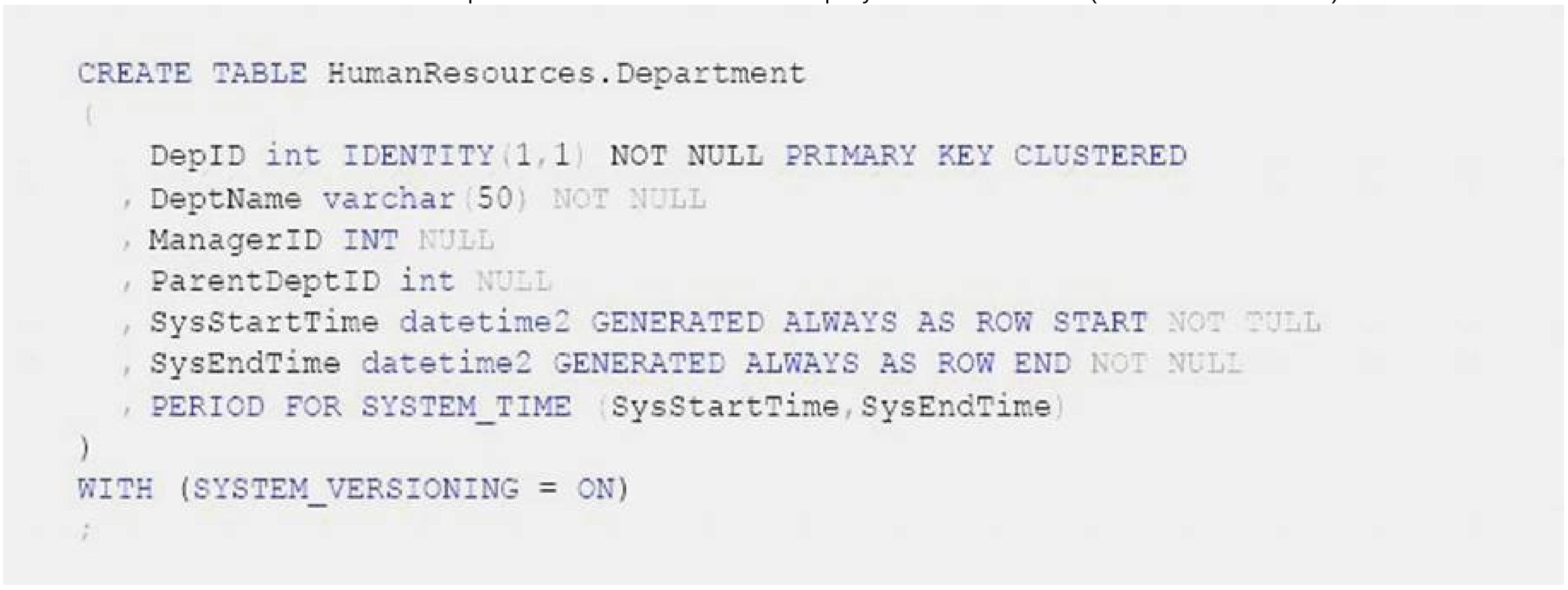
```
UPDATE Production.Products
SET ListPrice = ListPrice * 1.1
WHERE ListPrice
BETWEEN 0 and 100
```

Does the solution meet the goal?

A. Yes
B. No

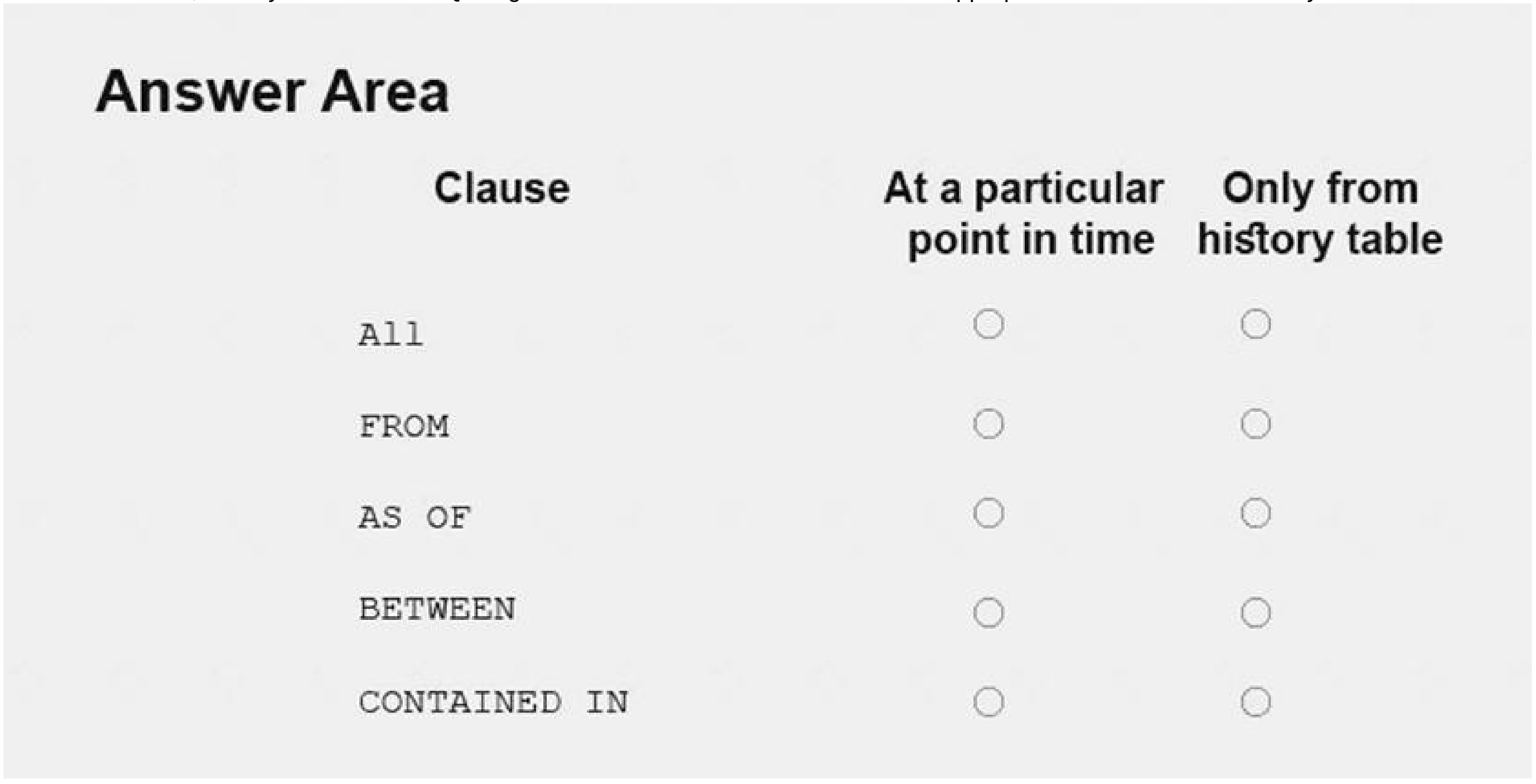
Answer: B

NEW QUESTION 192
You have a table named HumanResources.Department that was created with the query shown in the exhibit. (Click the Exhibit button.)



```
CREATE TABLE HumanResources.Department
(
    DepID int IDENTITY(1,1) NOT NULL PRIMARY KEY CLUSTERED
  , DeptName varchar(50) NOT NULL
  , ManagerID INT NULL
  , ParentDeptID int NULL
  , SysStartTime datetime2 GENERATED ALWAYS AS ROW START NOT TULL
  , SysEndTime datetime2 GENERATED ALWAYS AS ROW END NOT NULL
  , PERIOD FOR SYSTEM_TIME (SysStartTime,SysEndTime)
)
WITH (SYSTEM_VERSIONING = ON)
;
```

You need to query temporal data in the table.
In the table below, identify the Transact-SQL segments that must be used to retrieve the appropriate data. NOTE: Make only one selection in each column.



Answer Area

Clause	At a particular point in time	Only from history table
All	○	○
FROM	○	○
AS OF	○	○
BETWEEN	○	○
CONTAINED IN	○	○

A. Mastered
B. Not Mastered

Answer: A

Explanation:
AS OF: Returns a table with a rows containing the values that were actual (current) at the specified point in time in the past.
CONTAINED IN: If you search for non-current row versions only, we recommend you to use CONTAINED IN as it works only with the history table and will yield the best query performance.

NEW QUESTION 195
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 query a database that includes two tables: Project and Task. The Project table includes the following columns:

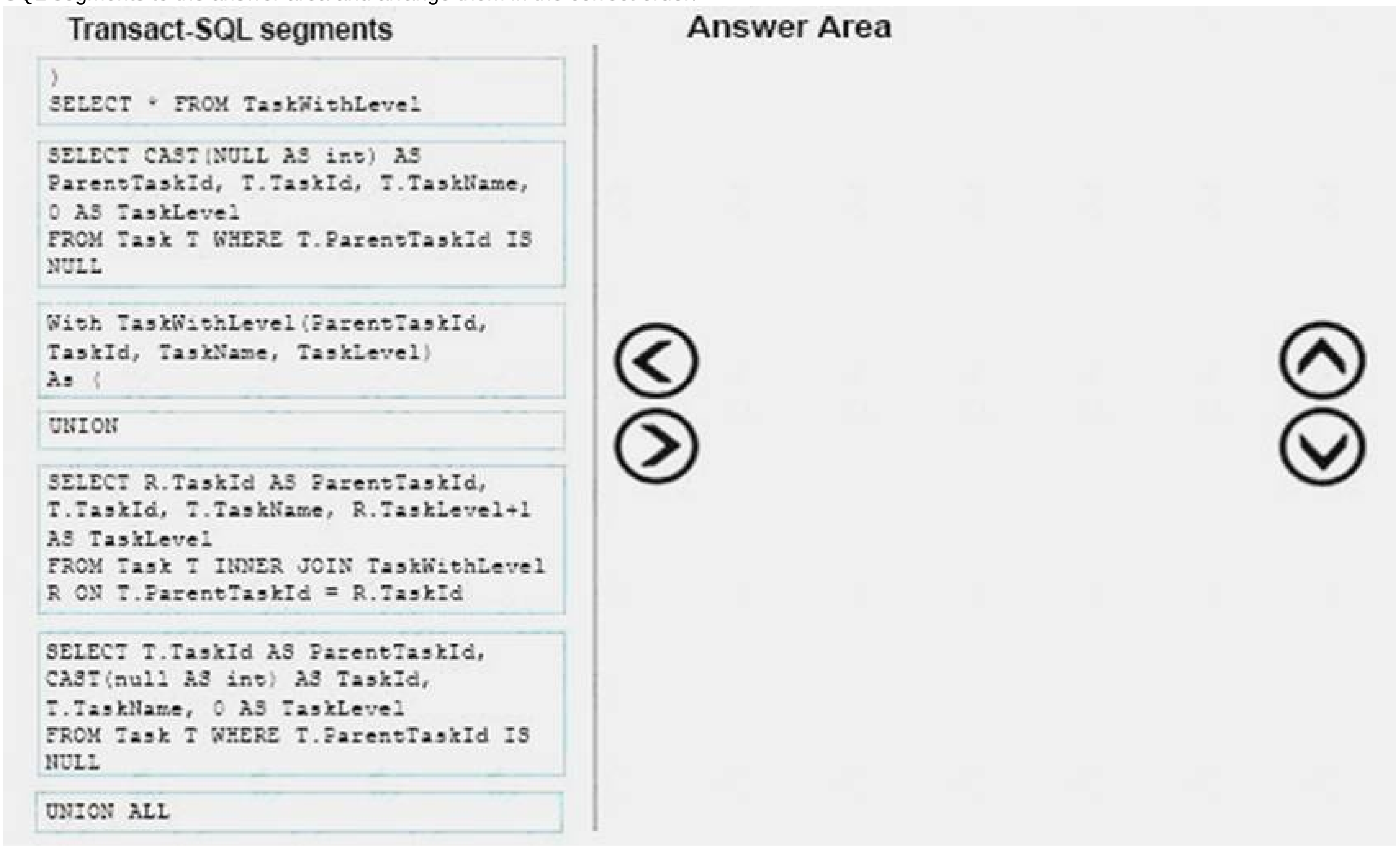
Column name	Data type	Notes
ProjectId	int	This is a unique identifier for a project.
ProjectName	varchar(100)	
StartTime	datetime2(7)	
EndTime	datetime2(7)	A null value indicates the project is not finished yet.
UserId	int	Identifies the owner of the project.

Column name	Data type	Notes
TaskId	int	This is a unique identifier for a task.
TaskName	varchar(100)	A nonclustered index exists for this column.
ParentTaskId	int	Each task may or may not have a parent task.
ProjectId	int	A null value indicates the task is not assigned to a specific project.
StartTime	datetime2(7)	
EndTime	datetime2(7)	A null value indicates the task is not completed yet.
UserId	int	Identifies the owner of the task.

Task level is defined using the following rules:

Task category	Task level definition
Tasks that have no parent task	[Task Level] = 0
Tasks that have a parent task	[Task Level] = [Parent Task's Level] + 1

You need to determine the task level for each task in the hierarchy.
Which five 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 * FROM TaskWithLevel
```

```
SELECT CAST(NULL AS int) AS
ParentTaskId, T.TaskId, T.TaskName,
0 AS TaskLevel
FROM Task T WHERE T.ParentTaskId IS
NULL
```

```
With TaskWithLevel(ParentTaskId,
TaskId, TaskName, TaskLevel)
As (
```

```
UNION
```

```
SELECT R.TaskId AS ParentTaskId,
T.TaskId, T.TaskName, R.TaskLevel+1
AS TaskLevel
FROM Task T INNER JOIN TaskWithLevel
R ON T.ParentTaskId = R.TaskId
```

```
SELECT T.TaskId AS ParentTaskId,
CAST(null AS int) AS TaskId,
T.TaskName, 0 AS TaskLevel
FROM Task T WHERE T.ParentTaskId IS
NULL
```

```
UNION ALL
```

A. Mastered
B. Not Mastered

Answer: A

Explanation:
Box 1: SELECT CAST (NULL AS INT) AS ParentTaskID, etc.
This statement selects all tasks with task level 0.
The ParentTaskID could be null so we should use CAST (NULL AS INT) AS ParentTaskID. Box 2: UNION
We should use UNION and not UNION ALL as we do not went duplicate rows.
UNION specifies that multiple result sets are to be combined and returned as a single result set.
Incorrect: Not UNION ALL: ALL incorporates all rows into the results. This includes duplicates. If not specified, duplicate rows are removed.
Box 3, Box 4, Box 5:

These statements select all tasks with task level >0. References:
https://msdn.microsoft.com/en-us/library/ms180026.aspx

NEW QUESTION 199
Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series. Information and details provided in a question apply only to that question.
You have a table named AuditTrail that tracks modifications to data in other tables. The AuditTrail table is updated by many processes. Data input into AuditTrail may contain improperly formatted date time values. You implement a process that retrieves data from the various columns in AuditTrail, but sometimes the process throws an error when it is unable to convert the data into valid date time values.
You need to convert the data into a valid date time value using the en-US format culture code. If the conversion fails, a null value must be returned in the column output. The conversion process must not throw an error.
What should you implement?

A. the COALESCE function
B. a view
C. a table-valued function
D. the TRY_PARSE function
E. a stored procedure
F. the ISNULL function
G. a scalar function
H. the TRY_CONVERT function

Answer: H

Explanation:
A TRY_CONVERT function returns a value cast to the specified data type if the cast succeeds; otherwise, returns null.
References: https://msdn.microsoft.com/en-us/library/hh230993.aspx

NEW QUESTION 203
......