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 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
  SET XACT_ABORT ON
  BEGIN TRY
    BEGIN TRANSACTION
      INSERT INTO Products (ProductName, UnitPrice, UnitsInStock, UnitsOnOrder)
      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

NEW QUESTION 2
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	Yossi
3	CUS4	Jack
4	NULL	Jane
5	NULL	Francisco

Customer_HRSystem

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

Records that contain null values for CustomerCode can be uniquely identified by CustomerName. You need to display distinct customers that appear in 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

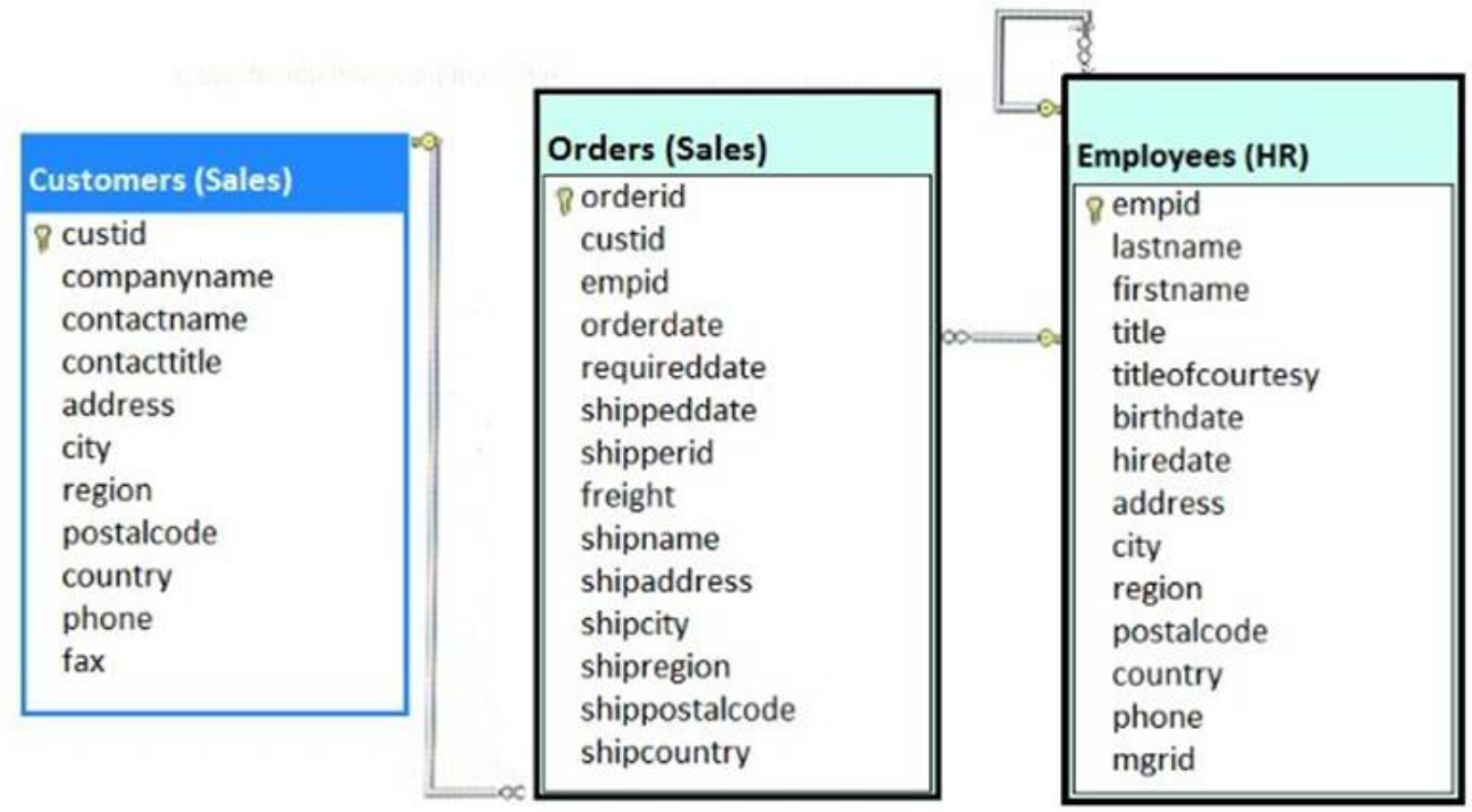
```
SELECT c.CustomerCode, c.CustomerName, h.CustomerCode, h.CustomerName
FROM Customer_CRMSystem c
FULL OUTER JOIN Customer_HRSystem h
ON c.CustomerCode = h.CustomerCode AND c.CustomerName = h.CustomerName
```

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: H

Explanation: To retain the nonmatching information by including nonmatching rows in the results of a join, use a full outer join. SQL Server provides the full outer join operator, FULL OUTER JOIN, which includes all rows from both tables, regardless of whether or not the other table has a matching value.

NEW QUESTION 3
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 4
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 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

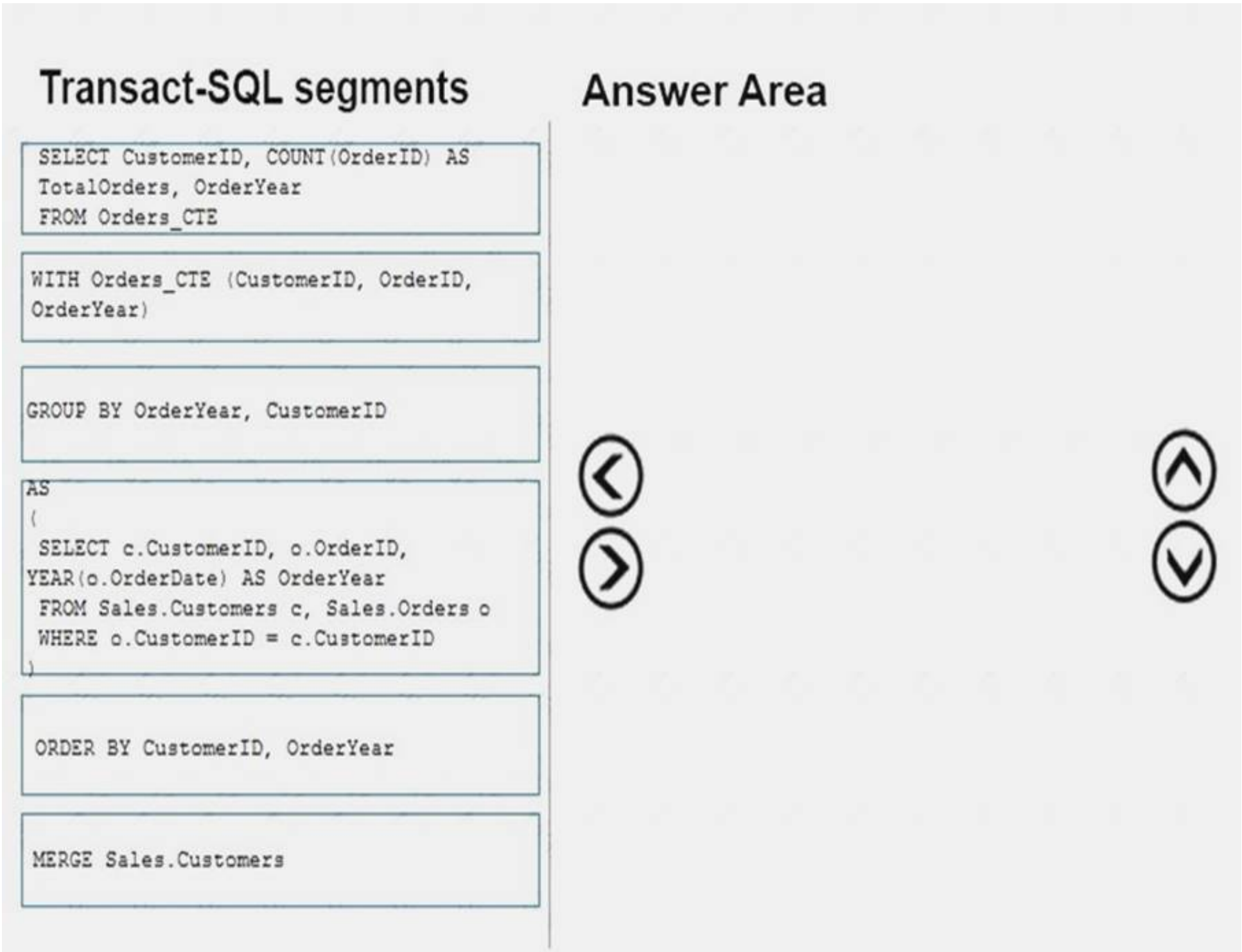
The following table describes the columns in Sales.Orders.

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.

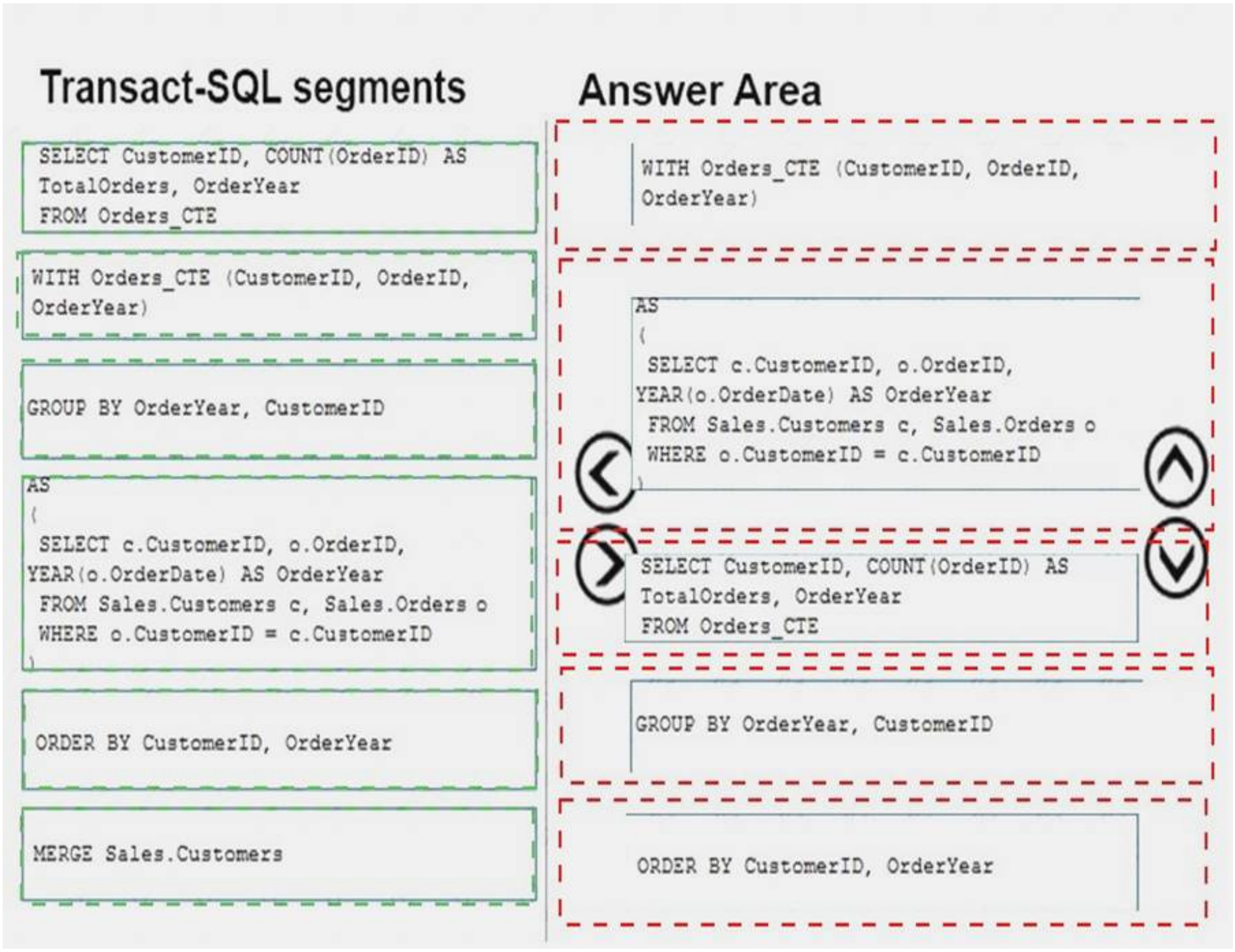
Column name	Data type	Constraints
OrderLineID	int	primary key
OrderID	int	foreign key to the Sales.Orders table
Quantity	int	does not allow null values
UnitPrice	decimal(18,2)	null values are permitted
TaxRate	decimal(18,3)	does not allow null values

End of repeated scenario
You need to create a common table expression (CTE) that returns the total number of orders per year for each customer.
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.



Answer:

Explanation:



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 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
TABLE
WITH SCHEMABINDING
AS
RETURN (
     SELECT TOP 1 @phoneNumber as PhoneNumber, VALUE as AreaCode
     FROM STRING_SPLIT(@phoneNumber, '-')
)
```

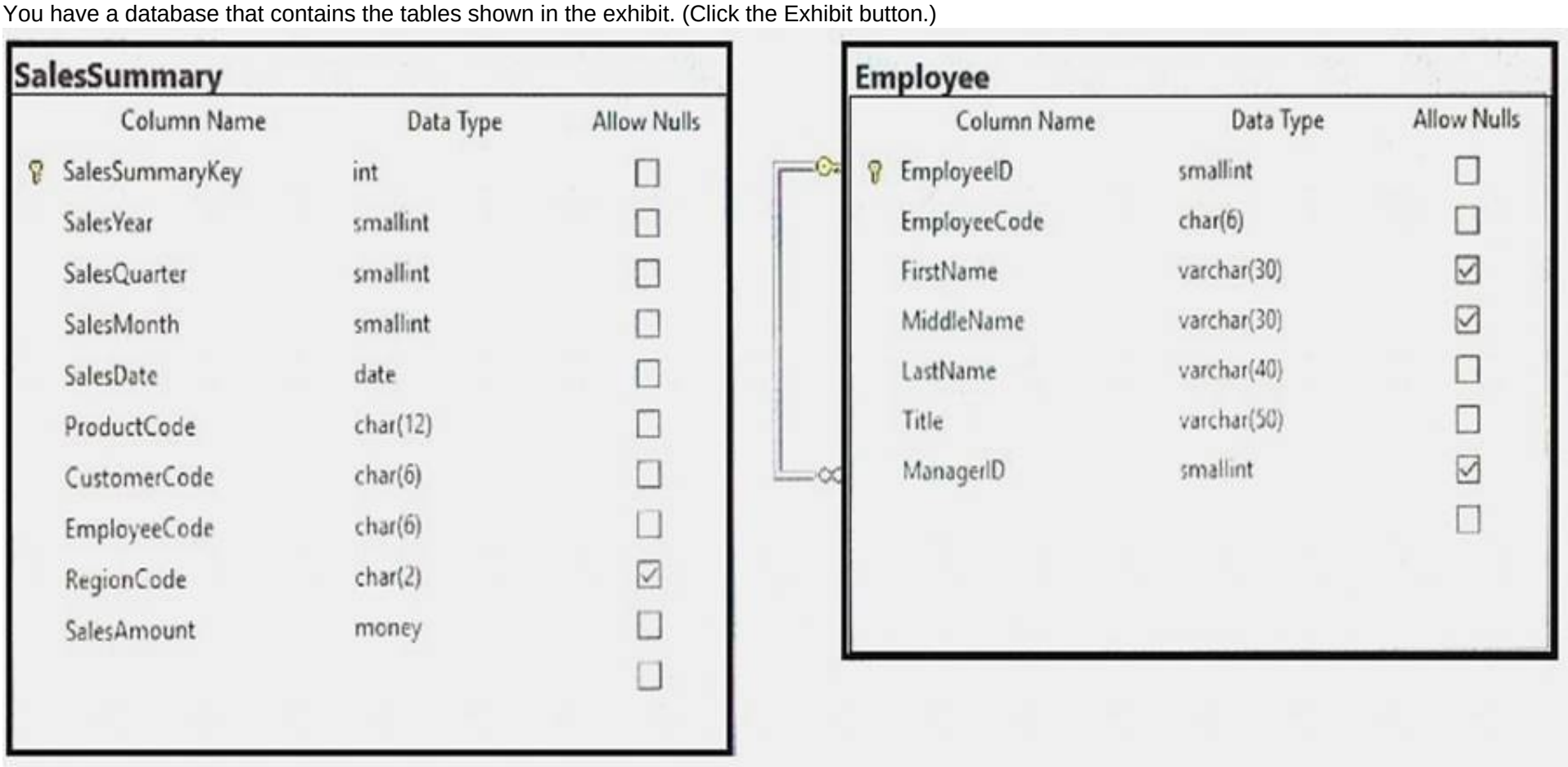
Does the solution meet the goal?

A. Yes
B. No

Answer: A

Explanation: As the result of the function will be used in an indexed view we should use schemabinding. References: https://sqlstudies.com/2014/08/06/schemabinding-what-why/

NEW QUESTION 6
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.)

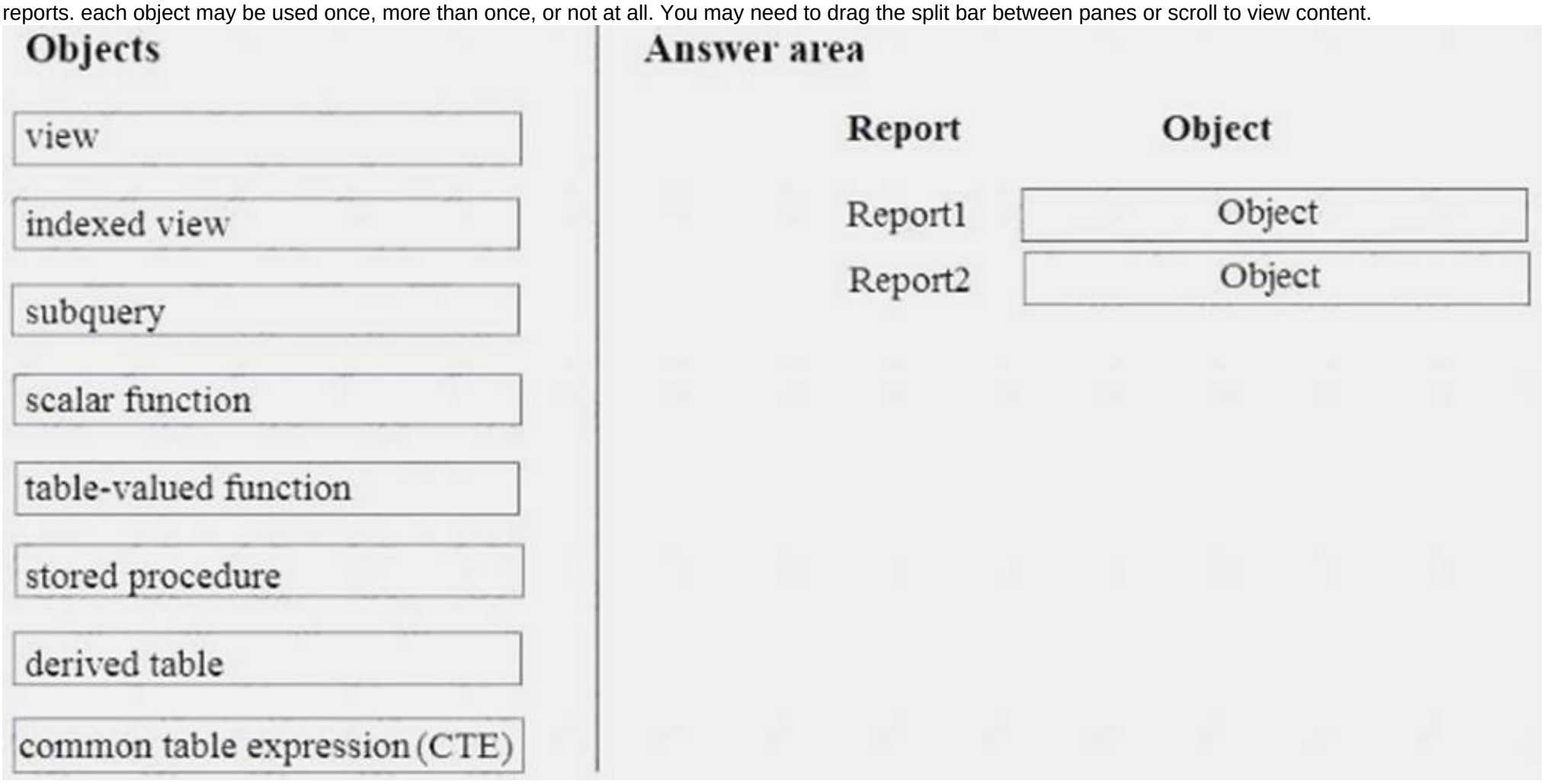


You review the Employee table and make the following observations:
- Every record has a value in the ManagerID except for the Chief Executive Officer (CEO).
- The FirstName and MiddleName columns contain null values for some records.
- The valid values for the Title column are Sales Representative manager, 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.



Answer:

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 7

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 need to identify the owner of each task by using the following rules:
- Return each task's owner if the task has an owner.
- If a task has no owner, but is associated with a project that has an owner, return the project's owner.
- Return the value -1 for all other cases.
How should you complete the Transact-SQL statement? To answer, select the appropriate Transact-SQL segments in the answer area.

Answer Area

```
SELECT T.TaskId, T.TaskName,
[ISNULL / COALESCE / CHOOSE] ( [T.UserId, P.UserId, -1 / P.UserId, T.UserId, -1 / -1, P.UserId, T.UserId / -1, T.UserId, P.UserId] ) AS OwnerUserId
FROM Task T
[INNER JOIN / LEFT JOIN / RIGHT JOIN] Project P ON T.ProjectId = P.ProjectId
```

Answer:

Explanation: Box 1: COALESCE
COALESCE evaluates the arguments in order and returns the current value of the first expression that initially does not evaluate to NULL.
Box 2: T.UserID, p.UserID, -1
- Return each task's owner if the task has an owner.
- If a task has no owner, but is associated with a project that has an owner, return the project's owner.
- Return the value -1 for all other cases. Box 3: LEFT JOIN
The LEFT JOIN keyword returns all rows from the right table (table2), with the matching rows in the left table (table1). The result is NULL in the left side when there is no match. Here the right side could be NULL as the projectID of the task could be NULL.
References:
https://msdn.microsoft.com/en-us/library/ms190349.aspx
http://www.w3schools.com/Sql/sql_join_right.asp

NEW QUESTION 8
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 0 and 100
```

Does the solution meet the goal?

A. Yes
B. No

Answer: B

Explanation: Products with a price of $0.00 would also be increased.

NEW QUESTION 9
You have a table named Table1 that contains 200 million rows. Table1 contains a column named SaleDate that has a data type of DateTime2(3).
Users report that the following query runs slowly.

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

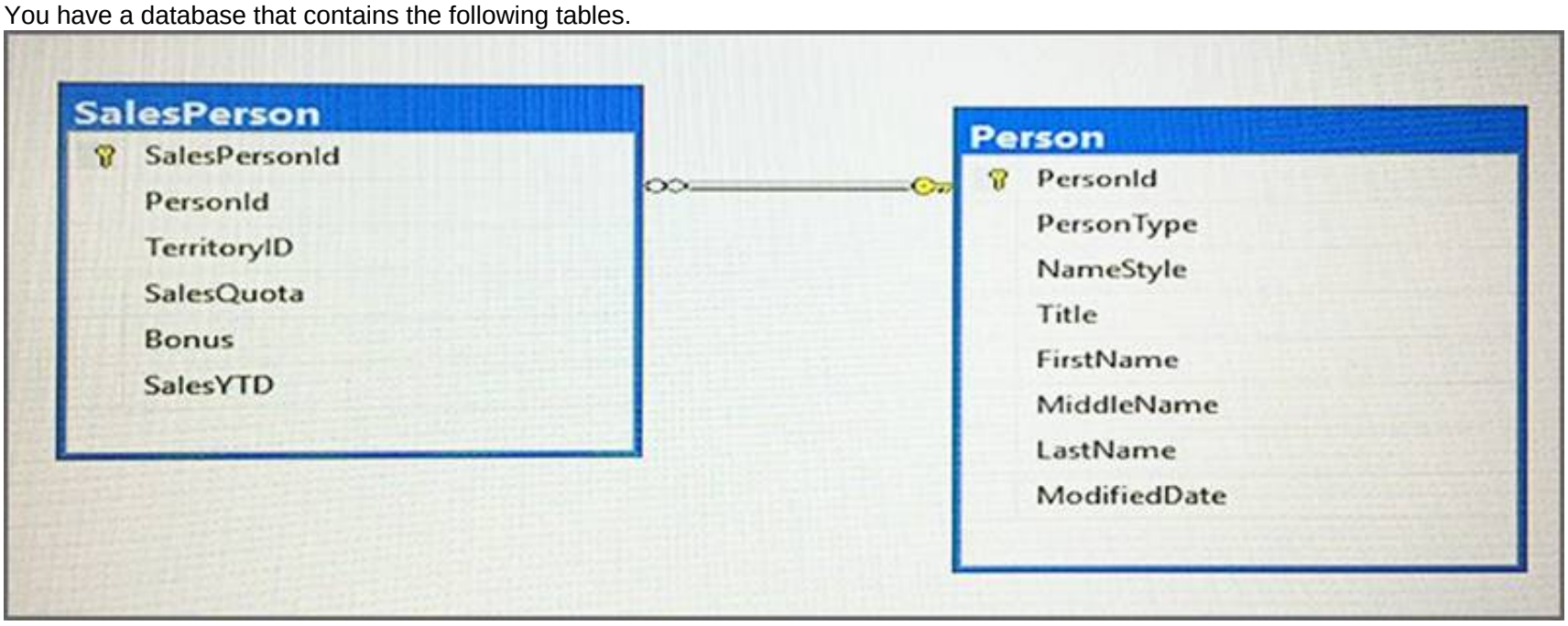
You need to reduce the amount of time it takes to run the query. What should you use to replace the WHERE statement?

A. WHERE SaleDate >= '2017-01-01' AND SaleDate < '2018-01-01'
B. WHERE cast(SaleDate as varchar(10)) BETWEEN '2017-01-01' AND '2017-12-31'
C. WHERE cast(SaleDate as date) BETWEEN '2017-01-01' AND '2017-12-31'
D. WHERE 2017 = year(SaleDate)

Answer: C

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

NEW QUESTION 10
You have a database that contains the following tables.



You need to create a query that lists the highest-performing salespersons based on the current year-to-date sales period. The query must meet the following requirements:
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

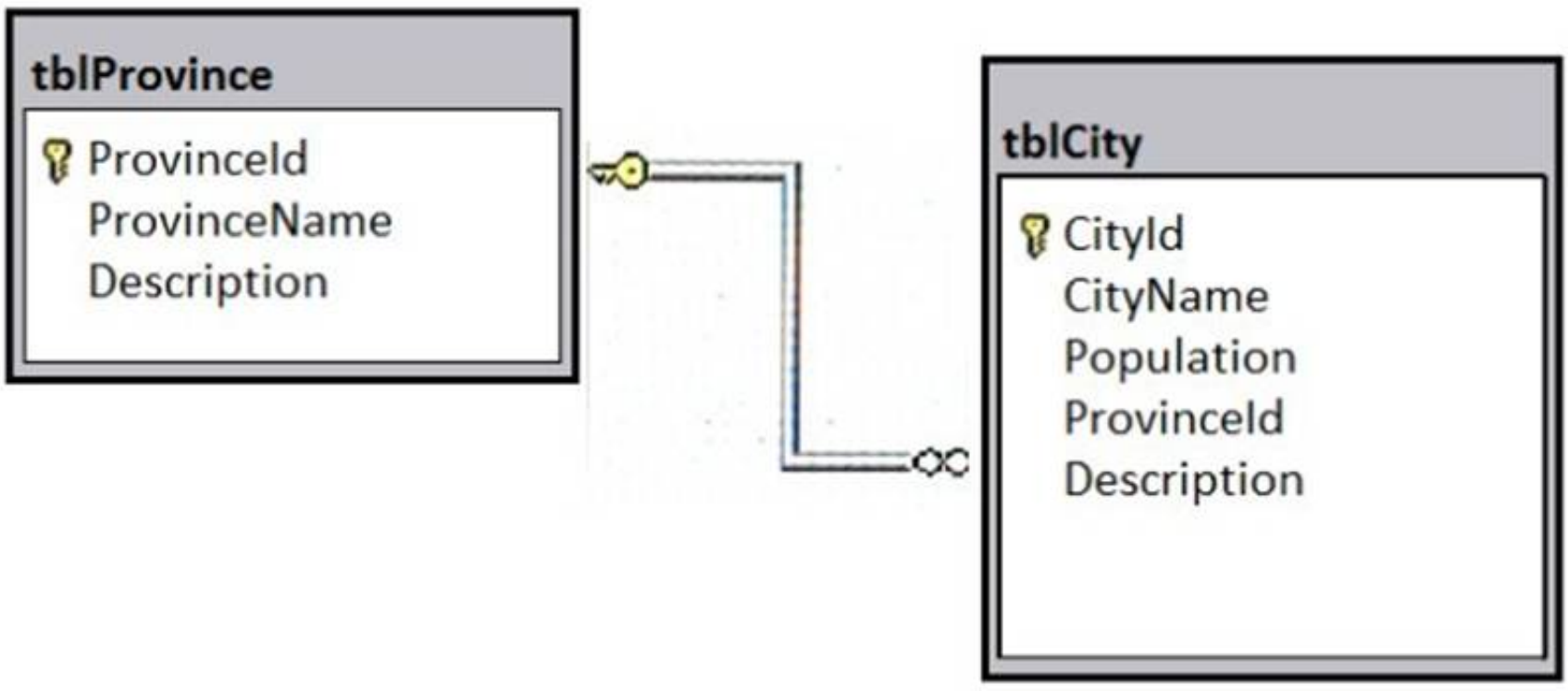
```
DEFAULT
DELETE
DENY
DESC
DISK
DISTINCT
DISTRIBUTED
DOUBLE
DROP
DUMP
ELSE
END
ERRLVL
ESCAPE
ESCEPT
EXEC
EXECUTE
EXISTS
ON
OPEN
OPENDATASOURCE
OPENQUERY
OPENROWSET
OPENXML
OPTION
OR
ORDER
OUTER
OVER
PERCENT
PIVOT
PLAN
PRECISION
PRIMARY
PRINT
TSEQUAL
UNION
UNIQUE
UNPIVOT
UPDATE
UPDATETEXT
USE
USER
VALUES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

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 top 3 lastname,salesYTD
2 FROM Person AS p INNER JOIN SalesPerson AS s 3 ON p.PersonID = s.SalesPersonID
4 WHERE territoryid is null 5 order by salesytd dsec
Use the Check Syntax button to verify your work. Any syntax or spelling errors will be reported by line and character position.

Answer:

Explanation: 1 SELECT top 3 lastname,salesYTD
2 FROM Person AS p INNER JOIN SalesPerson AS s 3 ON p.PersonID = s.SalesPersonID
4 WHERE territoryid is not null 5 order by salesytd desc
Note:
On line 4 add a not before null. On line 5 change dsec to desc.

NEW QUESTION 10
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.
A database has two tables as shown in the following database diagram:



You need to list all provinces that have at least two large cities. A large city is defined as having a population of at least one million residents. The query must return the following columns:

- tblProvince.ProvinceId
- tblProvince.ProvinceName
- a derived column named LargeCityCount that presents the total count of large cities for the province

Solution: You run the following Transact-SQL statement:

```
SELECT P.ProvinceId, P.ProvinceName, CitySummary.LargeCityCount
FROM tblProvince P
CROSS APPLY (
   SELECT COUNT(*) AS LargeCityCount FROM tblCity C
   WHERE C.Population>=1000000 AND C.ProvinceId = P.ProvinceId
) CitySummary
WHERE CitySummary.LargeCityCount >=2
```

Does the solution meet the goal?

A. Yes
B. No

Answer: A

Explanation: The requirement to list all provinces that have at least two large cities is meet by the WHERE CitySummary.LargeCityCount >=2 clause.
CROSS APPLY will work fine here. Note:
The APPLY operator allows you to invoke a table-valued function for each row returned by an outer table expression of a query. The table-valued function acts as the right input and the outer table expression acts as the left input. The right input is evaluated for each row from the left input and the rows produced are combined for the final output. The list of columns produced by the APPLY operator is the set of columns in the left input followed by the list of columns returned by the right input.
There are two forms of APPLY: CROSS APPLY and OUTER APPLY. CROSS APPLY returns only rows from the outer table that produce a result set from the table-valued function. OUTER APPLY returns both rows that produce a result set, and rows that do not, with NULL values in the columns produced by the table-valued function.
References: https://technet.microsoft.com/en-us/library/ms175156(v=sql.105).aspx

NEW QUESTION 13
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 deposit accounts. Which Transact-SQL statement should you run?

A.
```
SELECT COUNT(*)
FROM (SELECT AcctNo
FROM tblDepositAcct
INTERSECT
SELECT AcctNo
FROM tblLoanAcct) R
```

B.
```
SELECT COUNT(*)
FROM (SELECT CustNo
FROM tblDepositAcct
UNION
SELECT CustNo
FROM tblLoanAcct) R
```

C.
```
SELECT COUNT(*)
FROM (SELECTCustNo
FROM tblDepositAcct
UNION ALL
SELECT CustNo
FROM tblLoanAcct) R
```

D.
```
SELECT COUNT (DISTINCT D.CustNo)
FROM tblDepositAcct D, tblLoanAcct L
WHERE D.CustNo = L.CustNo
```

E.
```
SELECT COUNT(DISTINCT L.CustNo)
FROM tblDepositAcct D
RIGHT JOIN tblLoanAcct L OND.CustNo = L.CustNo
WHERE D.CustNo IS NULL
```

F.
```
SELECT COUNT(*)
FROM (SELECT CustNo
FROM tblDepositAcct
EXCEPT
SELECT CustNo
FROM tblLoanAcct) R
```

G.
```
SELECT COUNT (DISTINCT COALESCE(D.CustNo, L.CustNo))
FROM tblDepositAcct D
FULL JOIN tblLoanAcct L ON D.CustNo = L.CustNo
WHERE D.CustNo IS NULL OR L.CustNo IS NULL
```

H.
```
SELECT COUNT(*)
FROM tblDepositAcct D
FULL JOIN tblLoanAcct L ON D.CustNo = L.CustNo
```

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: F

Explanation: References:
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/set-operators-except-and-intersect-transact-sql?vie

NEW QUESTION 14
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 18
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 19
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 contains a single table named tblVehicleRegistration. The table is defined as follows:

Column name	Data type	Description
VehicleId	int	the primary key for the table
RegistrationNumber	varchar(5)	a vehicle registration number that contains only letters and numbers
RegistrationDate	date	the vehicle registration date
UserId	int	an identifier for the vehicle owner

You run the following query:

```
SELECT UserId FROM tblVehicleRegistration
WHERE RegistrationNumber = 20012
AND RegistrationDate > '2016-01-01'
```

The query output window displays the following error message: "Conversion failed when converting the varchar value 'AB012' to data type int."
You need to resolve the error.
Solution: You modify the Transact-SQL statement as follows:

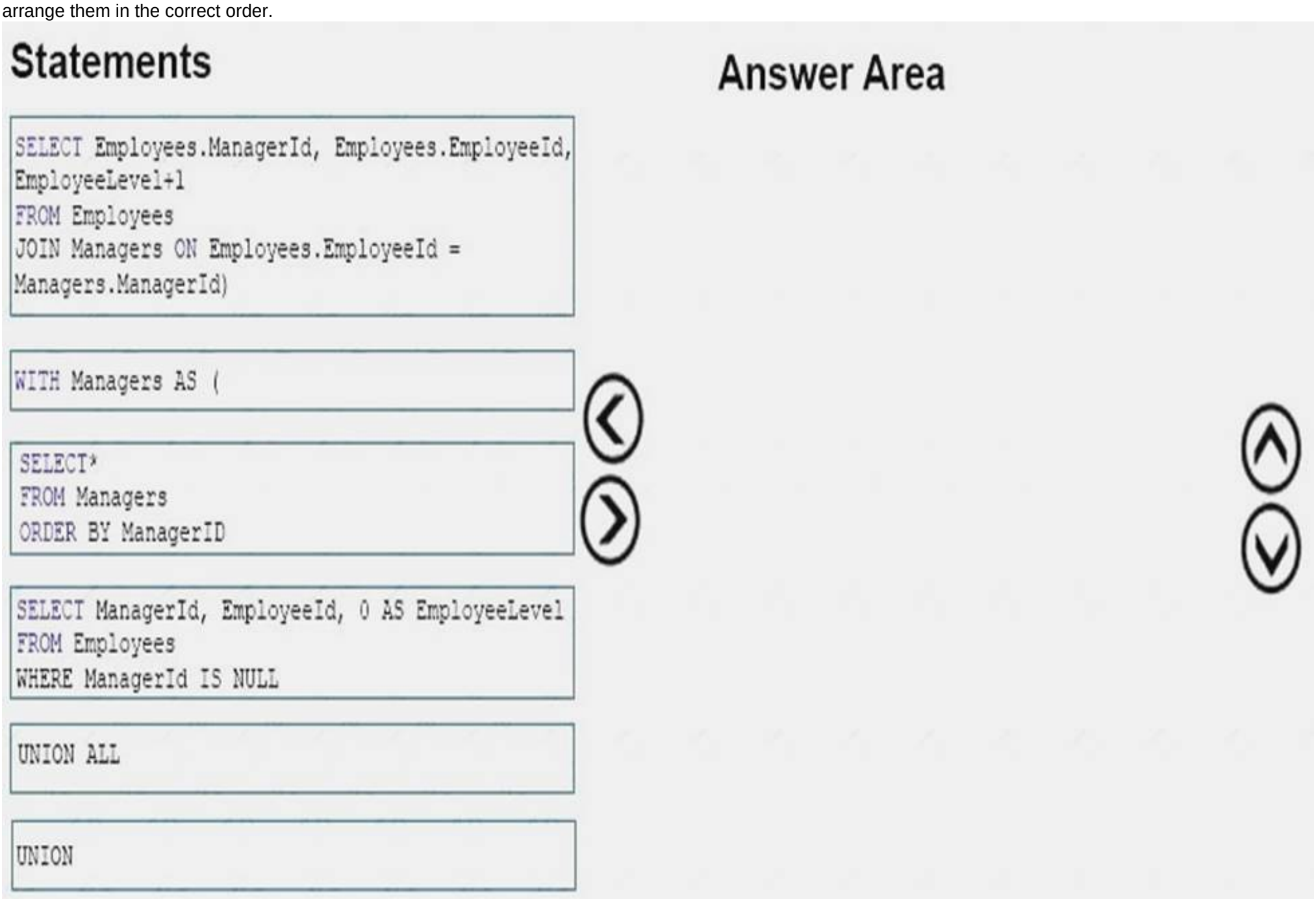
```
SELECT UserId FROM tblVehicleRegistration
WHERE RegistrationNumber =  20012
AND RegistrationDate > CONVERT(DATE, '2016-01-01', 120)
```

Does the solution meet the goal?

A. Yes
B. No

Answer: B

NEW QUESTION 22
You have a database named DB1 that contains a table named HR.Employees. HR.Employees contains two columns named ManagerID and EmployeeID.
ManagerID refers to EmployeeID.
You need to create a query that returns a list of all employees, the manager of each employee, and the numerical level of each employee in your organization's hierarchy.
Which five statements should you add to the query in sequence? To answer, move the appropriate statements from the list of statements to the answer area and arrange them in the correct order.



Answer:

Explanation: References:
https://blog.sqlauthority.com/2012/04/24/sql-server-introduction-to-hierarchical-query-using-a-recursive-cte-a-p

NEW QUESTION 25
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 29
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 datetime(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 = CustomerHistory))
```

You need to view all customer data.
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 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 00:00:00.0000000' AND '2015-01-01 00:00:00.0000000'
```

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: B

Explanation: The FOR SYSTEM_TIME ALL clause returns all the row versions from both the Temporal and History table. References: https://msdn.microsoft.com/en-us/library/dn935015.aspx

NEW QUESTION 31
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 need to return normalized data for all customers that were added in the year 2014. 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
```

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: The following query searches for row versions for Employee row with EmployeeID = 1000 that were active at least for a portion of period between 1st January of 2014 and 1st January 2015 (including the upper boundary):
SELECT * FROM Employee FOR SYSTEM_TIME
BETWEEN '2014-01-01 00:00:00.0000000' AND '2015-01-01 00:00:00.0000000'
WHERE EmployeeID = 1000 ORDER BY ValidFrom;
References: https://msdn.microsoft.com/en-us/library/dn935015.aspx

NEW QUESTION 33
You have the following Transact-SQL query:

```
SELECT
     City.CityID,
     City.CityName,
     TranslateName(Nearby.CityName) AS NearbyCity
FROM Cities AS City
CROSS APPLY NearbyCities(City.CityID) AS Nearby
```

What type of functions are used in the query? To answer, select the appropriate options in the answer area. NOTE: Each correct selection is worth one point.

Answer Area

Function	Type
TranslateName	Scalar / Table-Valued / System / Aggregate
NearbyCities	Scalar / Table-Valued / System / Aggregate

Answer:

Explanation: Box 1: Scalar
The return value of a function can either be a scalar (single) value or a table. Box 2: Table-Valued
The APPLY operator allows you to invoke a table-valued function for each row returned by an outer table expression of a query. The table-valued function acts as the right input and the outer table expression acts as the left input. The right input is evaluated for each row from the left input and the rows produced are combined for the final output. The list of columns produced by the APPLY operator is the set of columns in the left input followed by the list of columns returned by the right input.
References:
https://msdn.microsoft.com/en-us/library/ms186755.aspx https://technet.microsoft.com/en-us/library/ms175156(v=sql.105).aspx

NEW QUESTION 36
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.

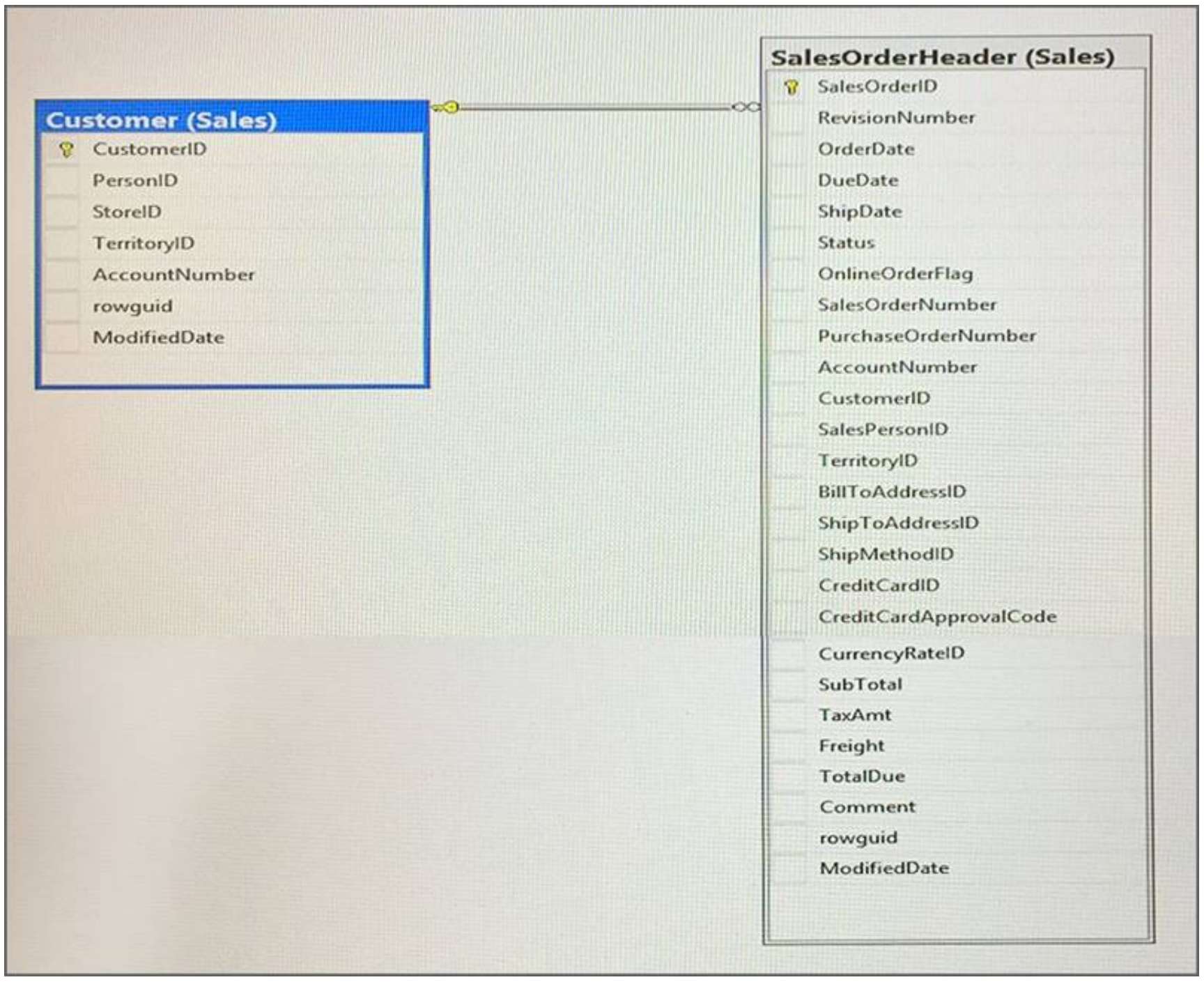
• Do not use the Row_number function.
• Do not surround object names with square brackets.
• Do not use variables.

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.

Answer:

Explanation: TRY_Convert

NEW QUESTION 38
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 a zero for the order ID and 01/01/1990 for the date.
Which Transact-SQL statement should you run?

A

```
SELECT C.CustomerID, COALESCE(MAX(OrderDate), '19000101')
FROM Sales.Customer C LEFT OUTER JOIN Sales.SalesOrderHeader SOH
ON C.CustomerID = SOH.CustomerID
GROUP BY C.CustomerID
ORDER BY C.CustomerID
```

B

```
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
```

C

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

D

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

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

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 42
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
OUTPUT inserted. CustomerId, deleted. CreditLimit, deleted. CreditLimit
INTO CustomerAudit (CustomerID, OldCreditLimit, NewCreditLimit, ChangedBy)
WHERE CustomerId=3
```
B.
```
UPDATE Customer
SET CreditLimit= 1000
OUTPUT inserted. CustomerId, GETDATE (), deleted. CreditLimit, inserted. CreditLimit, SYSTEM_USER
INTO CustomerAudit (CustomerID, DateChanged, OldCreditLimit, NewCreditLimit, ChangedBy)
WHERE CustomerId=3
```
C.
```
UPDATE Customer
SET CreditLimit= 1000
WHERE CustomerId=3
INSERT INTO CustomerAudit (CustomerId, DateChanged, OldCreditLimit, NewCreditLimit,
ChangedBy)
SELECT CustomerId, GETDATE (), CreditLimit, CreditLimit, SYSTEM_USER
FROM Customer
WHERE CustomerID =3
```
D.
```
UPDATE Customer
SET CreditLimit= 1000
OUTPUT inserted. CustomerId, inserted. CreditLimit, inserted. CreditLimit
INTO CustomerAudit (CustomerId, OldCreditLimit, NewCreditLimit)
WHERE CustomerId=3
```

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

Answer: C

NEW QUESTION 46
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 51
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	This is the primary key for the table.
DNS	nvarchar(100)	Null values are not permitted for this column.

Errors

Column	Data type	Notes
ErrorID	int	This is the primary key for the table.
ServerID	int	Null values are not permitted for this column. This column is a foreign key that is related to the ServerID column in the Servers table.
Occurrences	int	Null values are not permitted for this column.
LogMessage	nvarchar(max)	Null values are not permitted for this column.

You need to return all error log messages and the server where the error occurs most often. Which Transact-SQL statement should you run?

A
```
SELECT DISTINCT ServerID, LogMessage FROM Errors AS e1
WHERE Occurrences > ALL (
     SELECT e2.Occurrences FROM Errors AS e2
     WHERE e2.LogMessage = e1.LogMessage AND e2.ServerID <> e1.ServerID
)
```

B
```
SELECT DISTINCT ServerID, LogMessage FROM Errors AS e1
GROUP BY ServerID, LogMessage
HAVING MAX(Occurrences) = 1
```

C
```
SELECT DISTINCT ServerID, LogMessage FROM Errors AS e1
WHERE LogMessage IN (
     SELECT TOP 1 e2.LogMessage FROM Errors AS e2
     WHERE e2.LogMessage = e1.LogMessage AND e2.ServerID <> e1.ServerID
     ORDER BY e2.Occurrences
)
```

D
```
SELECT ServerID, LogMessage FROM Errors AS e1
GROUP BY ServerID, LogMessage, Occurrences
HAVING COUNT(*) = 1
ORDER BY Occurrences
```

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

Answer: C

NEW QUESTION 55
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 60
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 64
You run the following Transact-SQL statement:

```
CREATE TABLE Sales.Customers(
     custid int IDENTITY(1,1) NOT NULL,
     companyname nvarchar(50) NULL,
     contacttitle nvarchar(30) NOT NULL,
     address nvarchar(60) NOT NULL,
     postalcode nvarchar(10) NOT NULL,
     region nvarchar(15) NULL,
     phone nvarchar(24) NOT NULL,
     fax nvarchar(24) NULL,
) ON PPRIMARY
```

You need to ensure that you can insert data into the table.
What are the characteristics of the data? To answer, select the appropriate options in the answer area.

Answer Area

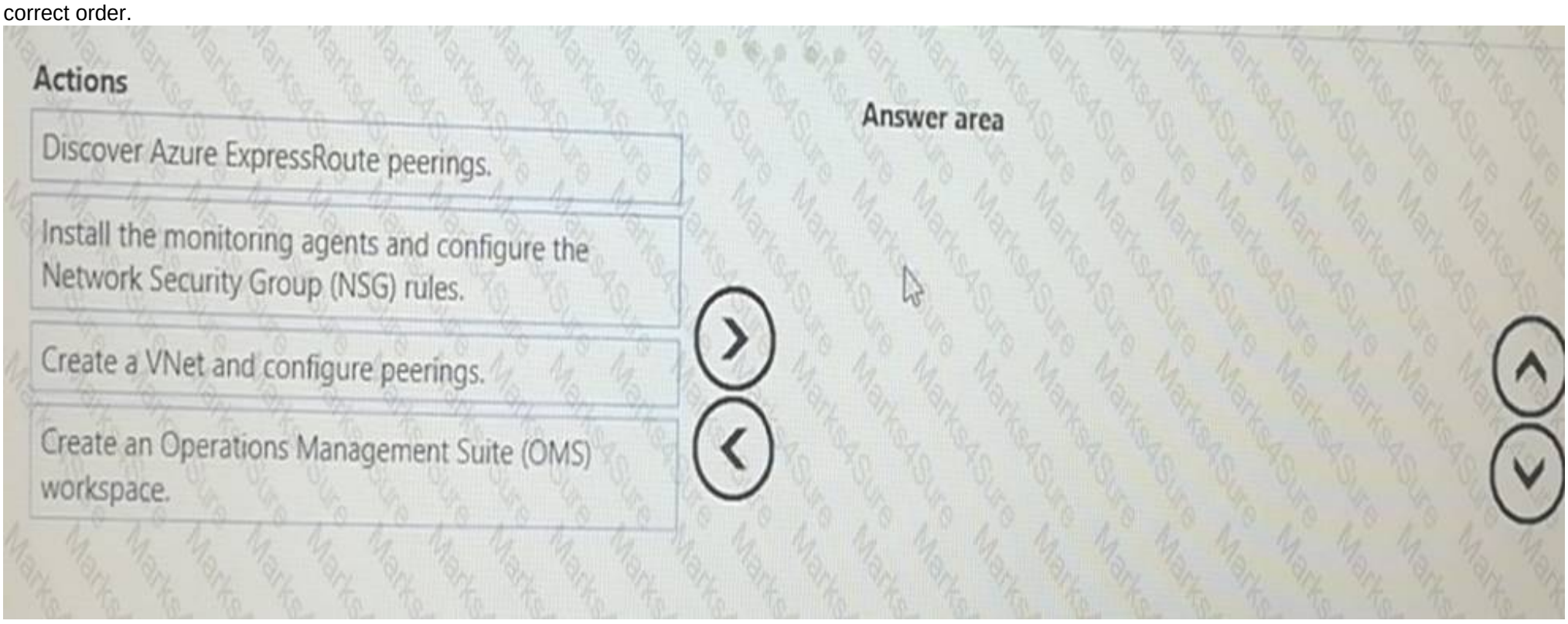
Column input constraint	**Column name**
Values cannot be entered into this column	custid / fax / postalcode / region
A value must be inserted into this column	custid / fax / postalcode / region
Data entry into this column is optional	custid / fax / postalcode / region

Answer:

Explanation: Box 1: custid
IDENTITY indicates that the new column is an identity column. When a new row is added to the table, the Database Engine provides a unique, incremental value for the column. Identity columns are typically used with PRIMARY KEY constraints to serve as the unique row identifier for the table.
Box 2: postalcode

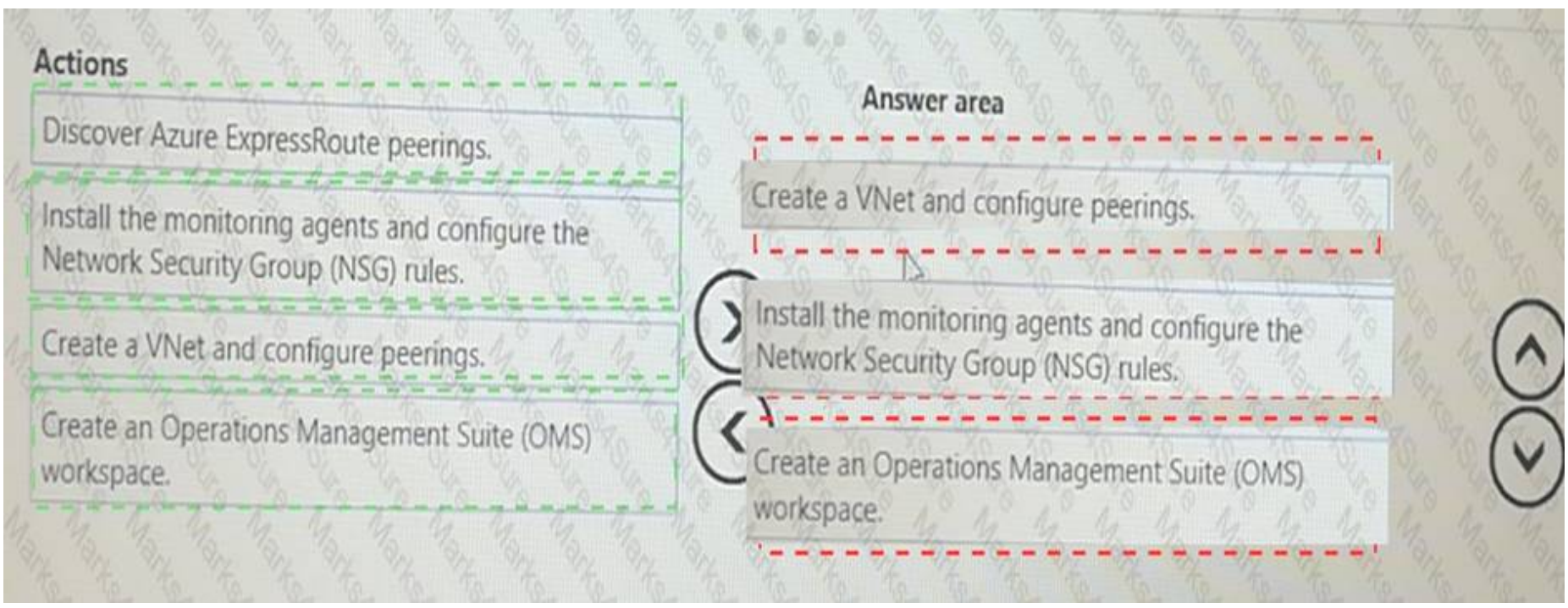
postalcode is declared as NOT NULL, which means that a value must be inserted. Box 3: region
Fax is also a correct answer. Both these two columns are declared as NULL, which means that data entry is optional.
References: https://msdn.microsoft.com/en-us/library/ms174979.aspx

NEW QUESTION 69
Your company plans to use Network Performance Monitor (NPM) on an existing Azure ExpressRoute connection.
You need to configure NPM.
Which three actions should you perform in sequence? To answer, move the appropriate actions from the list of actions to the answer area and arrange them in the correct order.



Answer:

Explanation:

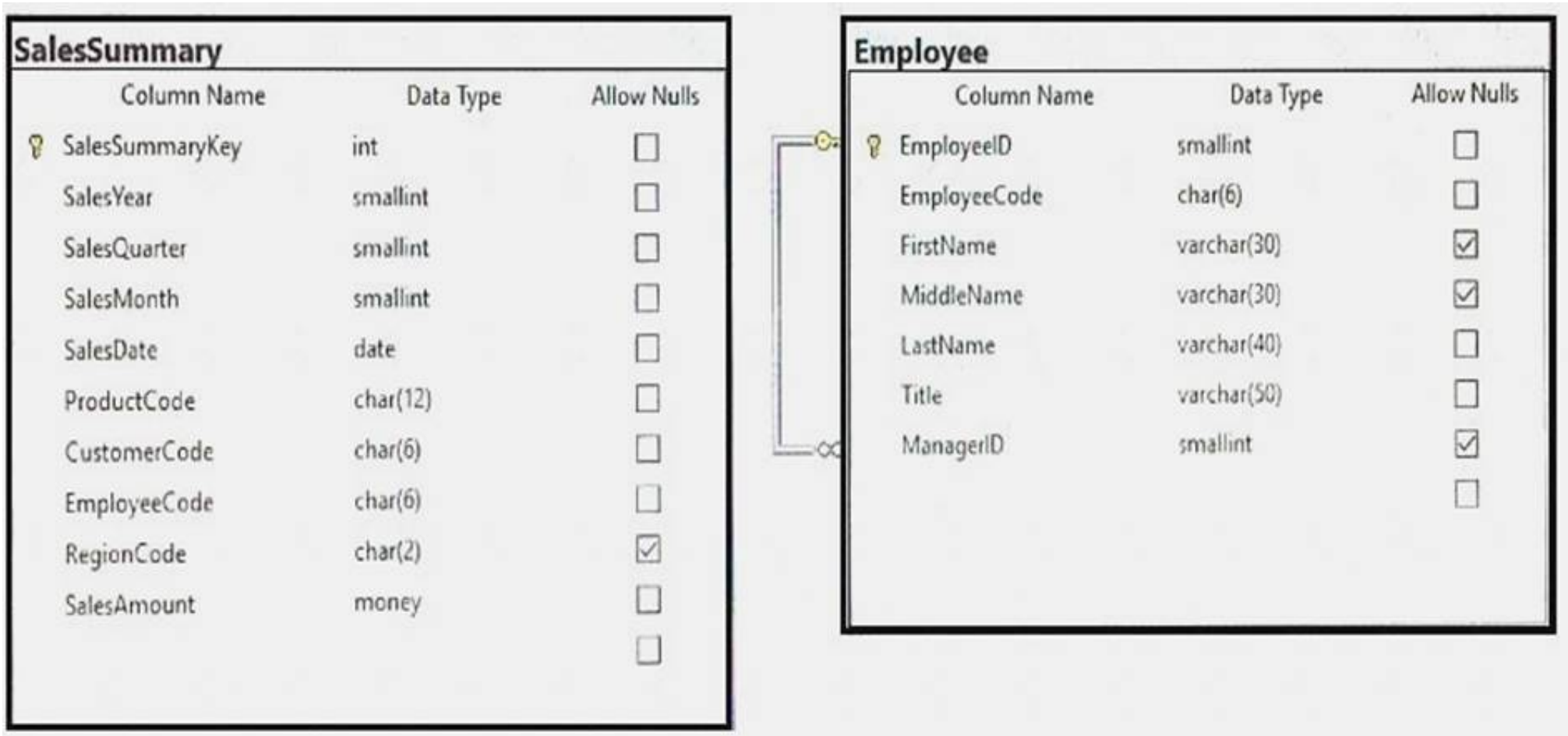


NEW QUESTION 74
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After 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 hash index on the primary key column. Does this meet the goal?

A. Yes
B. No

Answer: B

NEW QUESTION 78
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.)



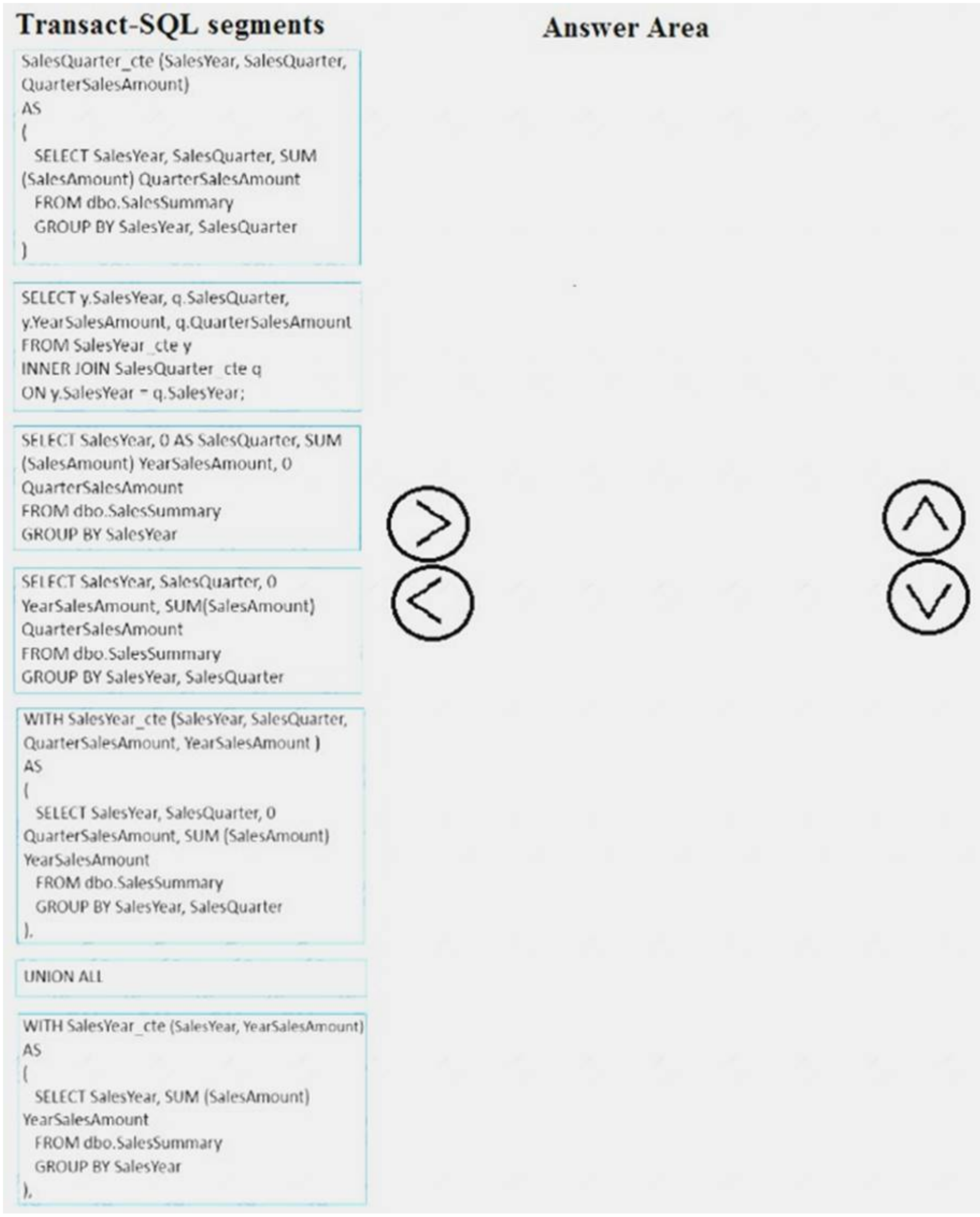
You review the Employee table and make the following observations:
- Every record has a value in the ManagerID except for the Chief Executive Officer (CEO).
- The FirstName and MiddleName columns contain null values for some records.
- The valid values for the Title column are Sales Representative manager, 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 need to create a query to return the data for the Sales Summary report.
Which three 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.



Answer:

Explanation: Use two CTE expressions, one for salesYear and one for SalesQuarter, and combine them with a SELECT statement.
Note: A common table expression (CTE) can be thought of as a temporary result set that is defined within the execution scope of a single SELECT, INSERT, UPDATE, DELETE, or CREATE VIEW statement. A CTE is similar to a derived table in that it is not stored as an object and lasts only for the duration of the query.
References: https://technet.microsoft.com/en-us/library/ms190766(v=sql.105).aspx

NEW QUESTION 81
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 84
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 86

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
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
```

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.

Answer:

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 87
You are developing a mobile app to manage meetups. The app allows for users to view the 25 closest people with similar interests. You have a table that contains records for approximately two million people. You create the table by running the following Transact-SQL statement:

```
CREATE TABLE Person (
     PersonID INT,
     Name NVARCHAR(155) NOT NULL,
     Location GEOGRAPHY,
     Interests NVARCHAR(MAX)
)
```

You create the following table valued function to generate lists of people:

```
CREATE FUNCTION dbo.nearby  (@person AS INT)
     RETURNS @Res TABLE (
     PersonId INT NOT NULL,
     Location GEOGRAPHY
     )
     AS
     BEGIN
        . . .
     END
```

You need to build a report that shows meetings with at least two people only. What should you use?

A. OUTER APPLY
B. CROSS APPLY
C. PIVOT
D. LEFT OUTER JOIN

Answer: B

Explanation: References: https://www.sqlshack.com/the-difference-between-cross-apply-and-outer-apply-in-sql-server/

NEW QUESTION 88
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.

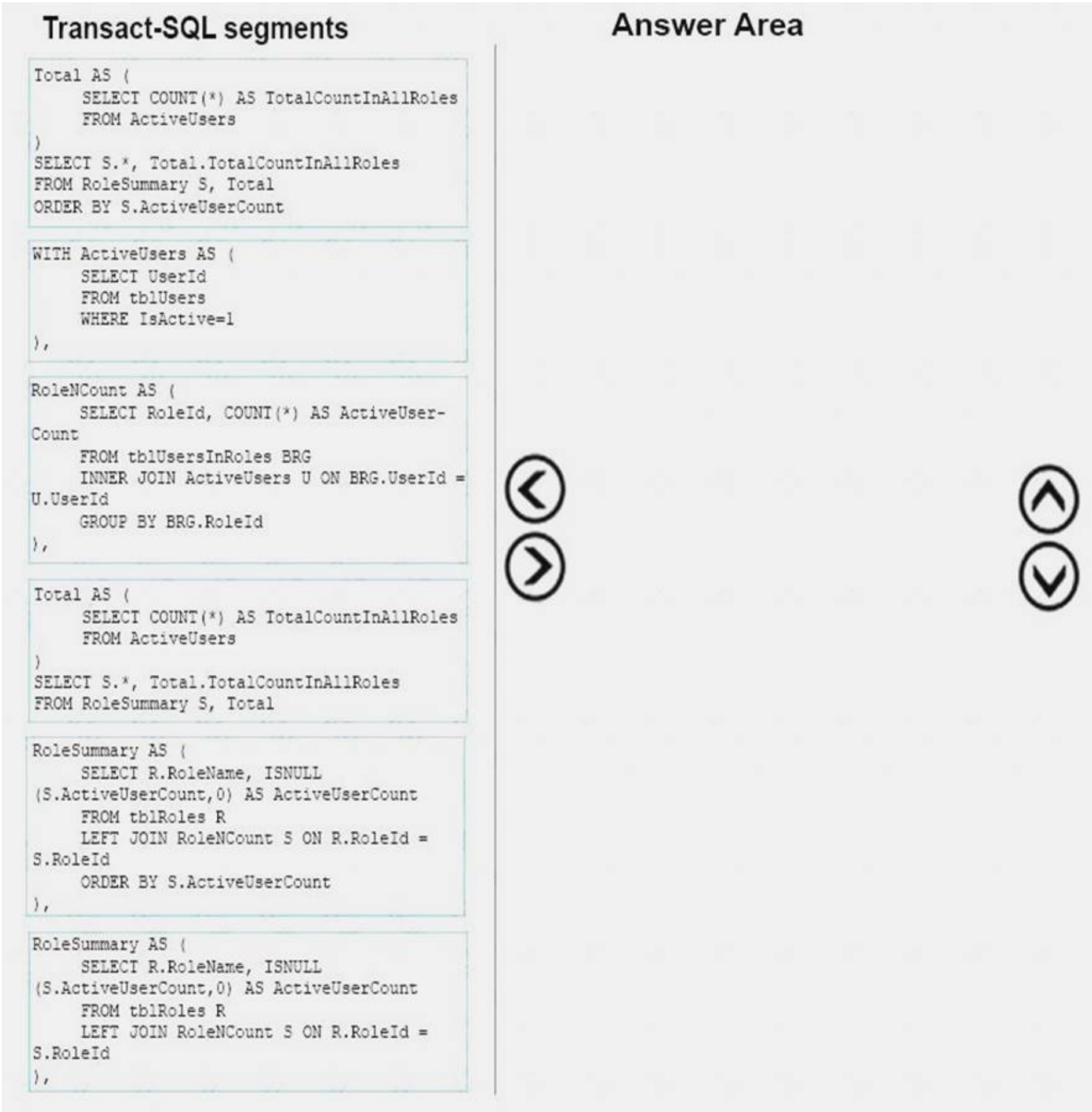
A. a user-defined table-valued function
B. a CRL function
C. a stored procedure
D. a user-defined scalar function

Answer: AC

NEW QUESTION 93
You create three tables by running the following Transact-SQL statements:

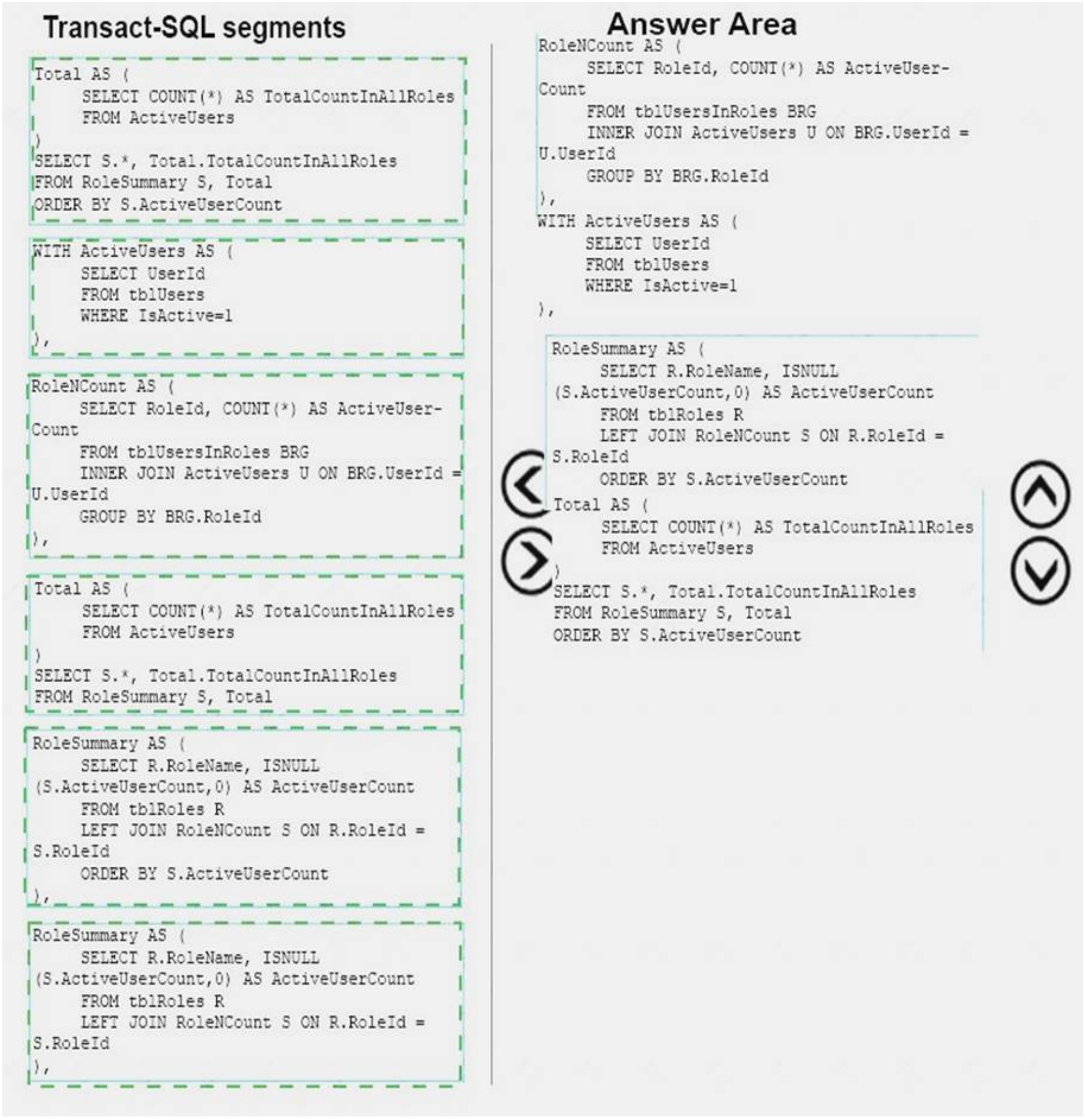
```
CREATE TABLE tblRoles (
     RoleId int NOT NULL IDENTITY(1,1) PRIMARY KEY CLUSTERED,
     RoleName varchar(20) NOT NULL
)
CREATE TABLE tblUsers (
     UserId int NOT NULL IDENTITY(1,1) PRIMARY KEY CLUSTERED,
     UserName varchar(20) UNIQUE NOT NULL,
     IsActive bit NOT NULL DEFAULT(1)
)
CREATE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

For reporting purposes, you need to find the active user count for each role, and the total active user count. The result must be ordered by active user count of each role. You must use common table expressions (CTEs).
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.



Answer:

Explanation:



NEW QUESTION 94
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 is denormalized. Users make frequent changes to data in a primary table.
You need to ensure that users cannot change the tables directly, and that changes made to the primary table also update any related tables.
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: Using an Indexed View would allow you to keep your base data in properly normalized tables and maintain data-integrity while giving you the denormalized "view" of that data.
References:
http://stackoverflow.com/questions/4789091/updating-redundant-denormalized-data-automatically-in-sql-server

NEW QUESTION 95
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
```

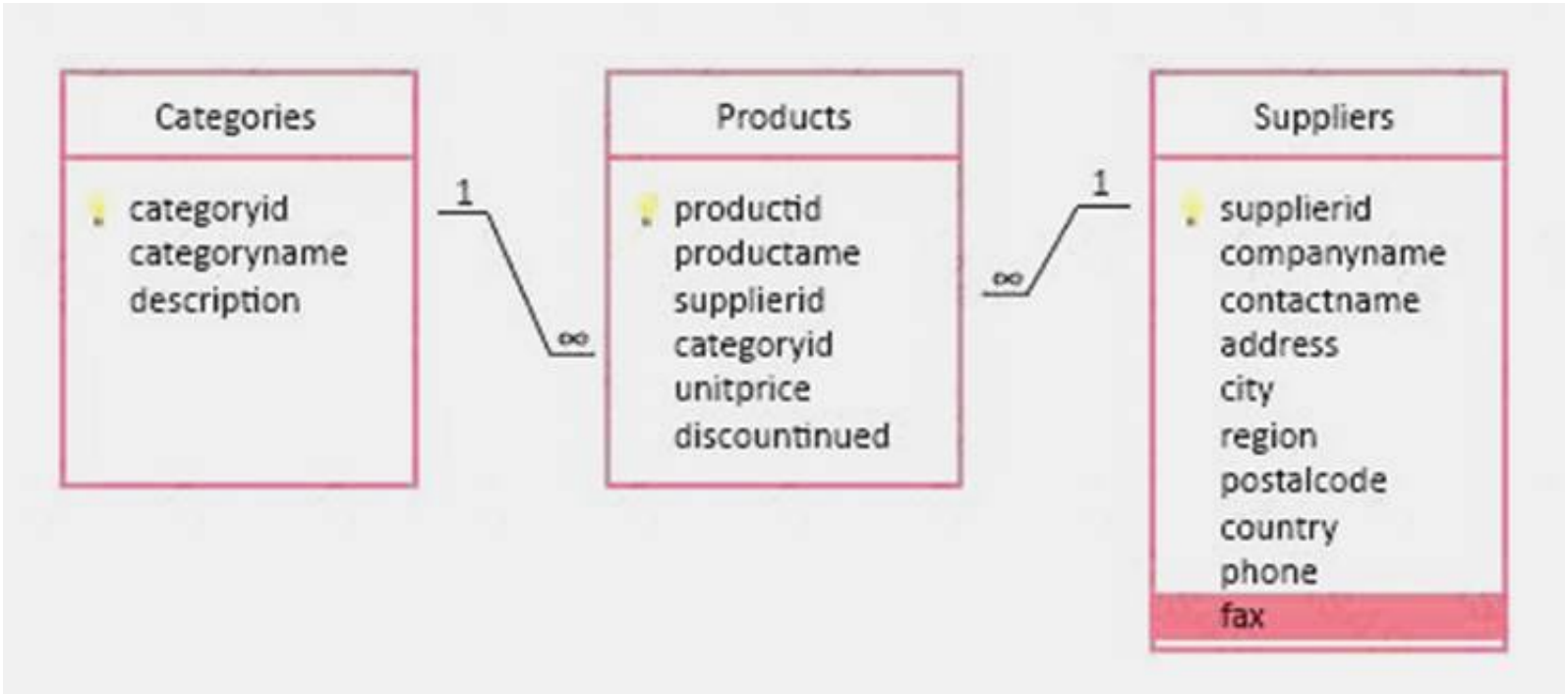
The variable @custID is set to a valid customer. Does the solution meet the goal?

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 100
You have a database that includes the following tables. All of the tables are in the Production schema.



You need to create a query that returns a list of product names for all products in the Beverages category. Construct the query using the following guidelines:
Use the first letter of the table name as the table alias.
Use two-part column names.
Do not surround object names with square brackets.
Do not use implicit joins.
Do not use variables.
Use single quotes to surround literal values.
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 p.productname
2     FROM Production.Categories AS c
3     inner join production.products as p on c.categoryid*p.categoryid
4     WHERE c.categoryname = 'Beverages'
```

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.

Answer:

Explanation: 1 SELECT p.productname
2 FROM Production.categories AS c
3 inner join production.products as p on c.categoryid=p.categoryid 4 WHERE c.categoryname = 'Beverages'
Note: On line 3 change * to =

NEW QUESTION 103
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
```

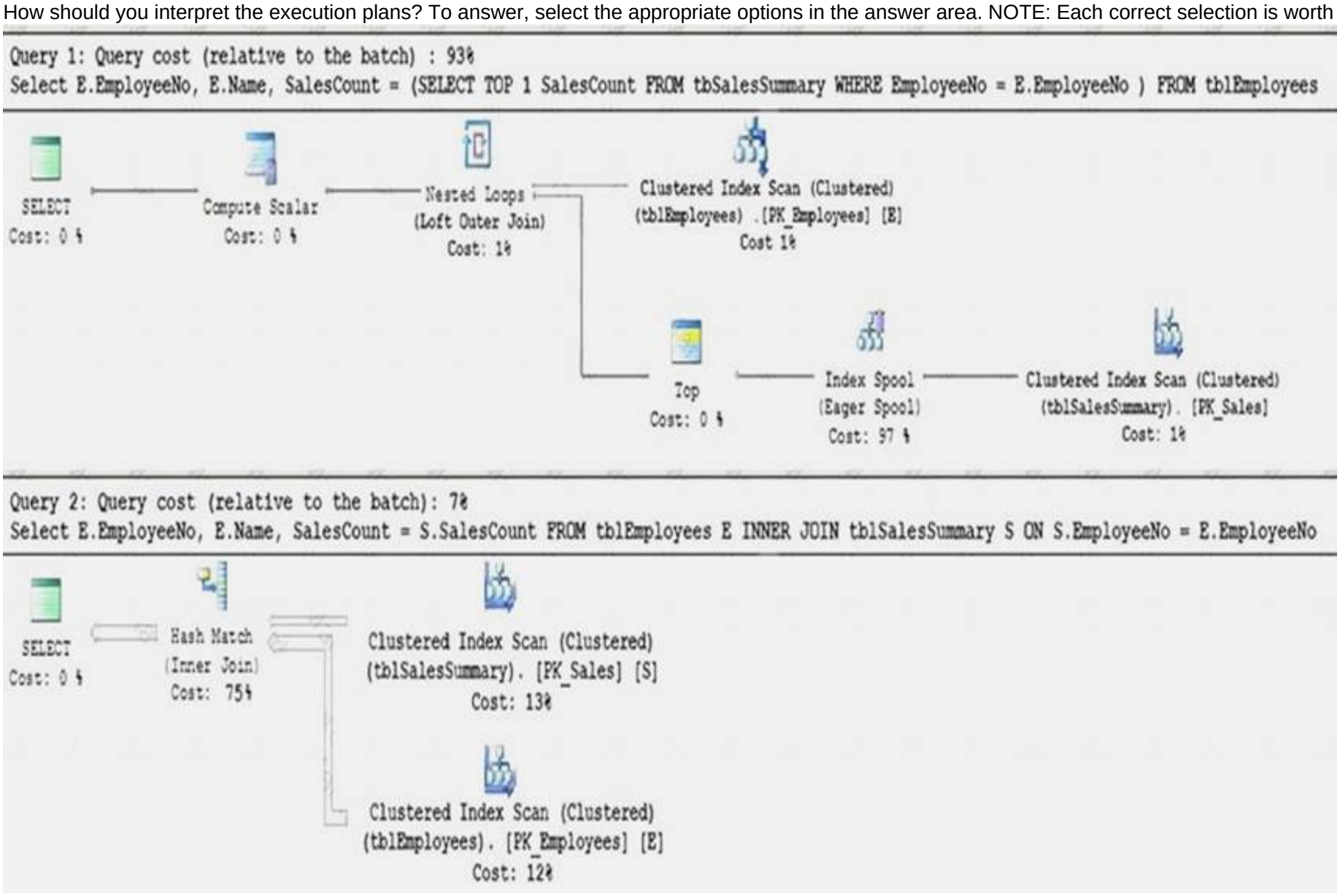
Does the solution meet the goal?

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 104
You have a database that contains the following tables: tblEmployees and tblSalesSummary. Each record contains approximately one million records.
You use Microsoft SQL Server Management Studio (SSMS) to run two queries. The Include Actual Execution Plan option is enabled.
Both queries return the same results. SSMS generates the execution plans shown in the exhibit. (Click the Exhibit button.)
You need to troubleshoot the queries.
How should you interpret the execution plans? To answer, select the appropriate options in the answer area. NOTE: Each correct selection is worth one point.



Questions	Answer choices
Which of the two queries is more efficient?	Query 1 is more efficient. Query 2 is more efficient. Both queries are equally efficient.
Why is the cost of the Index Spool operator highest in the first execution plan?	There is no index in the tblSalesSummary table. The Index Scan on the tblSalesSummary table is executed many times. The Index Spool operator is executed many times.

Answer:

Explanation: References:
https://docs.microsoft.com/en-us/sql/relational-databases/showplan-logical-and-physical-operators-reference?vie

NEW QUESTION 106
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 107
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 named Customers. Data stored in the table must be exchanged between web pages and web servers by using AJAX calls that use REST endpoint.

You need to return all customer information by using a data exchange format that is text-based and lightweight.
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: C

Explanation: JSON can be used to pass AJAX updates between the client and the server.
Export data from SQL Server as JSON, or format query results as JSON, by adding the FOR JSON clause to a SELECT statement.
When you use the FOR JSON clause, you can specify the structure of the output explicitly, or let the structure of the SELECT statement determine the output.
References: https://msdn.microsoft.com/en-us/library/dn921882.aspx

NEW QUESTION 112
You need to create a database object that meets the following requirements:
accepts a product identifies as input

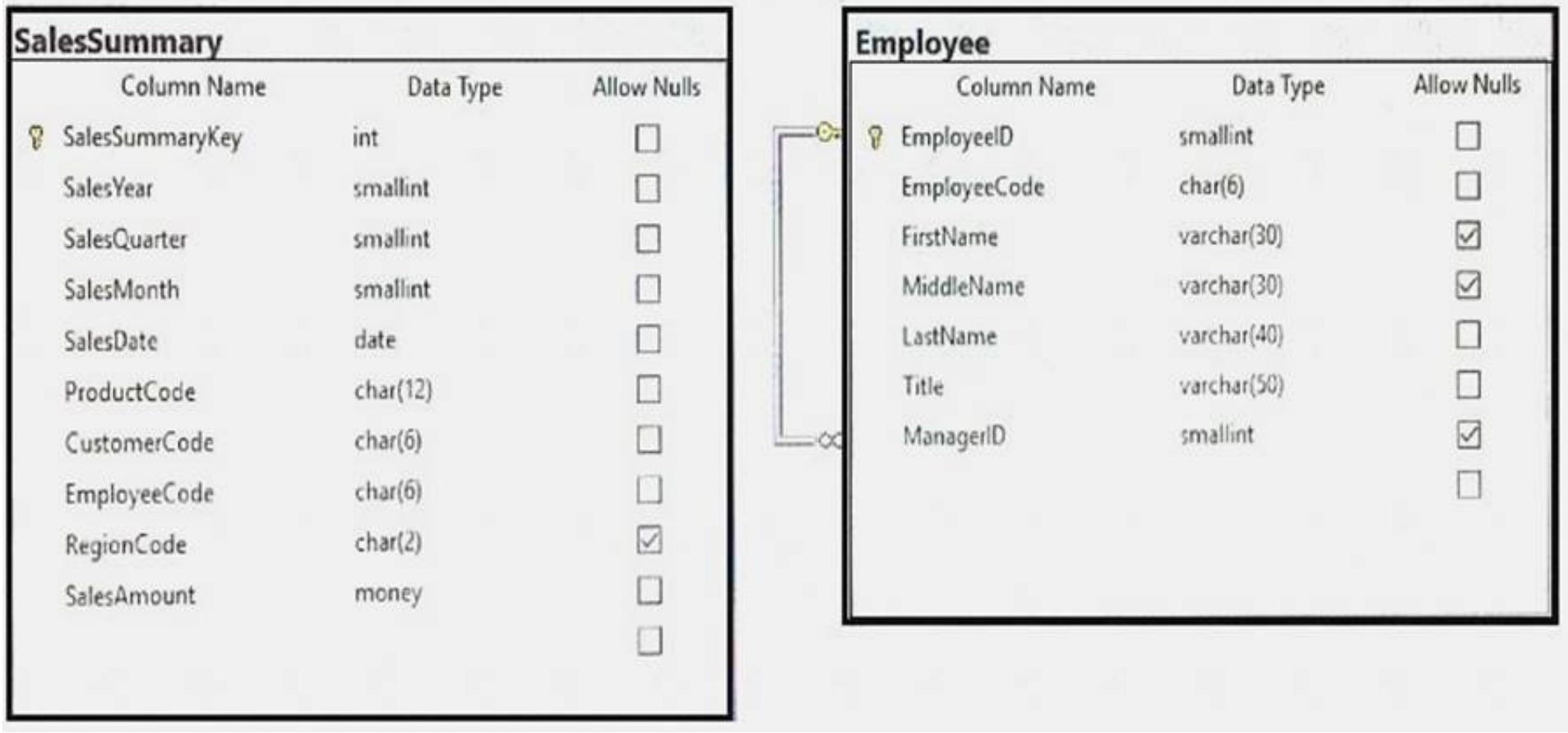
calculates the total quantity of a specific product, including quantity on hand and quantity on order
caches and reuses execution plan
returns a value
can be called from within a SELECT statement
can be used in a JOIN clause
What should you create?

A. a temporary table that has a columnstore index
B. a user-defined table-valued function
C. a memory-optimized table that has updated statistics
D. a natively-complied stored procedure that has an OUTPUT parameter

Answer: B

Explanation: A table-valued user-defined function can also replace stored procedures that return a single result set. The table returned by a user-defined function can be referenced in the FROM clause of a Transact-SQL statement, but stored procedures that return result sets cannot.
References: https://technet.microsoft.com/en-us/library/ms191165(v=sql.105).aspx

NEW QUESTION 115
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.)



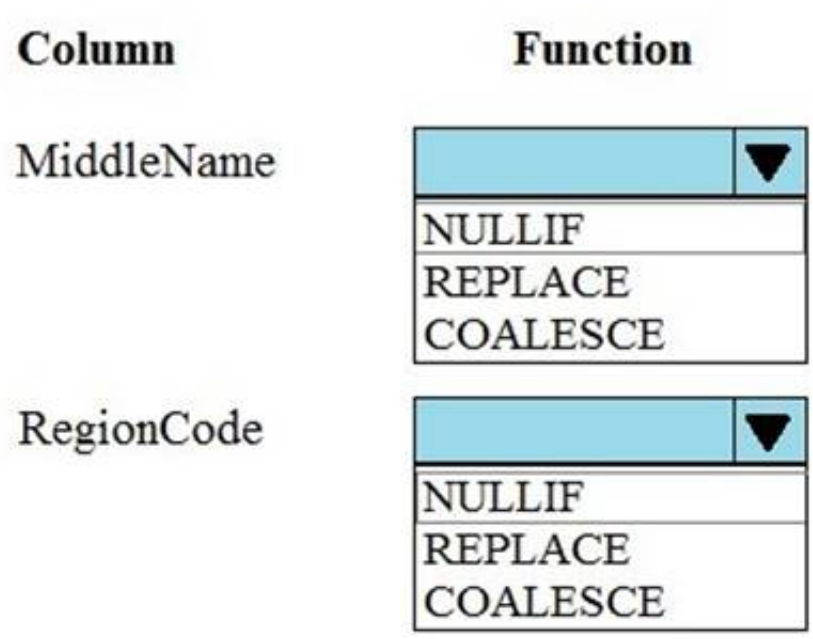
You review the Employee table and make the following observations:
- Every record has a value in the ManagerID except for the Chief Executive Officer (CEO).
- The FirstName and MiddleName columns contain null values for some records.
- The valid values for the Title column are Sales Representative manager, 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 need to create the query for the Sales by Region report.
Which function should you apply to each column? To answer, select the appropriate options in the answer area.

Answer area



Answer:

Explanation: Box 1: COALESCE
COALESCE evaluates the arguments in order and returns the current value of the first expression that initially does not evaluate to NULL.
If MiddleName is NULL, FirstName must be displayed. If both FirstName and MiddleName have null values, the world Unknown must be displayed.
The following example shows how COALESCE selects the data from the first column that has a nonnull value.
SELECT Name, Class, Color, ProductNumber, COALESCE(Class, Color, ProductNumber) AS FirstNotNull FROM Production.Product;
Not NULLIF: NULLIF returns the first expression if the two expressions are not equal. If the expressions are equal, NULLIF returns a null value of the type of the first expression.
Box 2: COALESCE
If RegionCode is NULL, the word Unknown must be displayed.
References: https://docs.microsoft.com/en-us/sql/t-sql/language-elements/coalesce-transact-sql

NEW QUESTION 116
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 global temporary table in the stored procedure. Does this meet the goal?

A. Yes
B. No

Answer: A

NEW QUESTION 118
You are building a stored procedure that will update data in a table named Table1 by using a complex query as the data source.
You need to ensure that the SELECT statement in the stored procedure meets the following requirements:
Data being processed must be usable in several statements in the stored procedure.
Data being processed must contain statistics. What should you do?

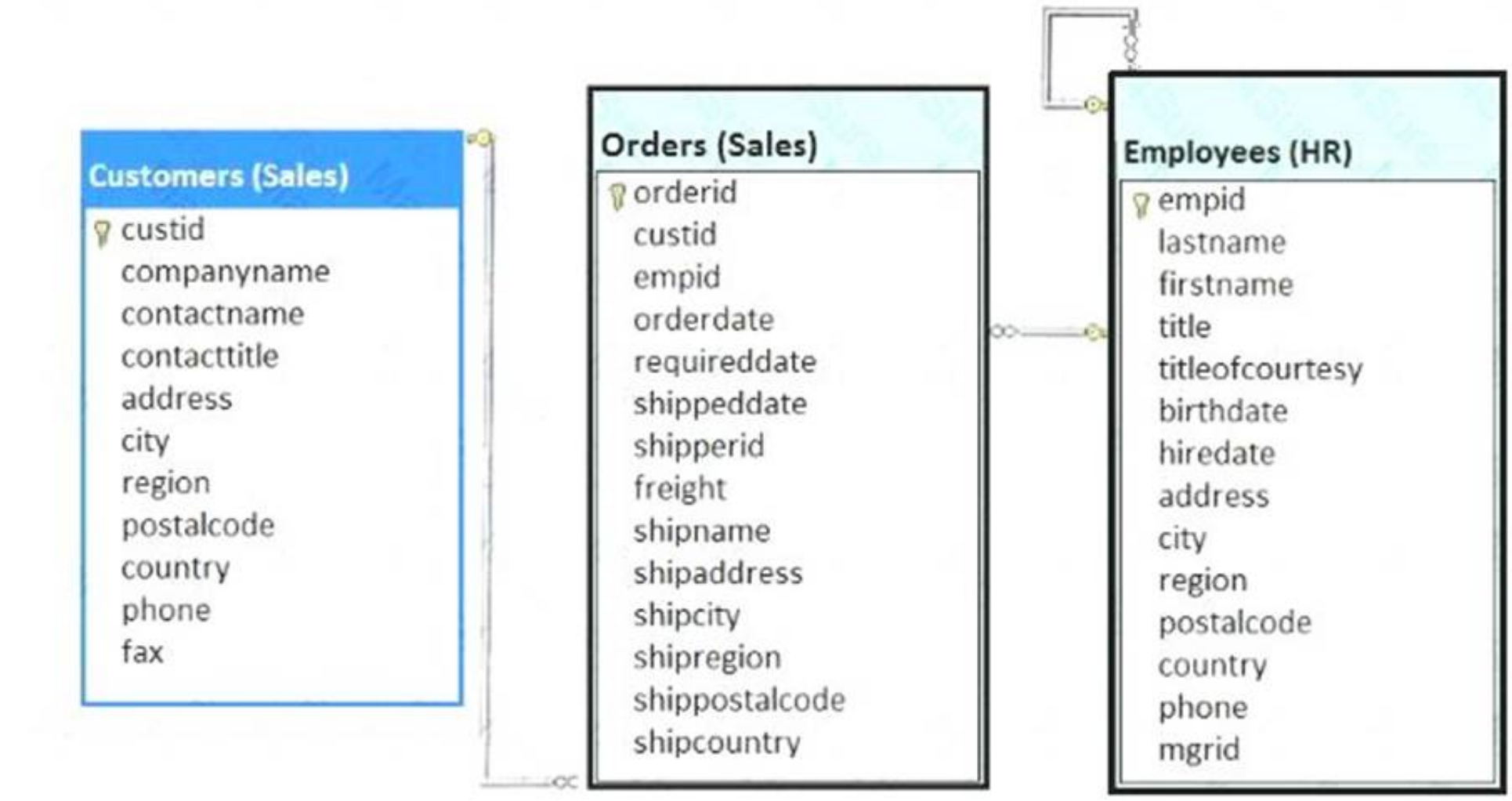
A. Update Table1 by using a common table expression (CTE).
B. Insert the data into a temporary table, and then update Table1 from the temporary table.
C. Place the SELECT statement in a derived table, and then update Table1 by using a JOIN to the derived table.
D. Insert the data into a table variable, and then update Table1 from the table variable.

Answer: B

Explanation: Temp Tables...
Are real materialized tables that exist in tempdb Have dedicated stats generated by the engine Can be indexed

Can have constraints
Persist for the life of the current CONNECTION Can be referenced by other queries or subproce

NEW QUESTION 123
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 125
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
```

Answer:

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 130
You run the following Transact SQL statement:

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

You use the table to store data about training courses: when they finished, the location, and the number of participants in the courses. You need to display a result set that shows aggregates for all possible combination of the number of participants. Which Transact-SQL statement should you run?

A)

```
A.  SELECT CourseID, CourseDate, SUM(NumParticipants)
    FROM CourseParticipants
    GROUP BY CourseID, CourseDate WITH ROLLUP
```

B)

```
B.  SELECT CourseID, CourseDate, SUM(DISTINCT NumParticipants)
    FROM CourseParticipants
    GROUP BY CourseID, CourseDate
```

C)

```
C.  SELECT CourseID, CourseDate, SUM(NumParticipants)
    FROM CourseParticipants
    GROUP BY CourseID, CourseDate
```

D)

```
D.  SELECT CourseID, CourseDate, SUM(NumParticipants)
    FROM CourseParticipants
    GROUP BY CourseID, CourseDate WITH CUBE
```

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

Answer: A

NEW QUESTION 132
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 contains a single table named tblVehicleRegistration. The table is defined as follows:

Column name	Data type	Description
VehicleId	int	the primary key for the table
RegistrationNumber	varchar(5)	a vehicle registration number that contains only letters and numbers
RegistrationDate	date	the vehicle registration date
UserId	int	an identifier for the vehicle owner

You run the following query:

```
SELECT UserId FROM tblVehicleRegistration
WHERE RegistrationNumber = 20012
AND RegistrationDate > '2016-01-01'
```

The query output window displays the following error message: "Conversion failed when converting the varchar value 'AB012' to data type int."
You need to resolve the error.
Solution: You modify the Transact-SQL statement as follows:

```
SELECT UserId FROM tblVehicleRegistration
WHERE CAST(RegistrationNumber AS int) = 20012
AND RegistrationDate > '2016-01-01'
```

Does the solution meet the goal?

A. Yes
B. No

Answer: A

NEW QUESTION 135
A company's sales team is divided in two different regions, North and South. You create tables named SalesNorth and SalesSouth. The SalesNorth table stores sales data from the North region. The SalesSouth table stores sales data from the South region. Both tables use the following structure:

Column name	Data type	Allow nulls
region	CHAR(1)	Yes
salesID	INT	Yes
customer	VARCHAR(150)	Yes
amount	MONEY	Yes

You need to create a consolidated result set that includes all records from both tables. Which Transact-SQL statement should you run?

A. SELECT SalesNorth.salesID, SalesNorth.customer, SalesNorth.amount, SalesSouth.SalesID, SalesSouth.customer, SalesSouth.amountFROM SalesNorthJOIN SalesSouth ON SalesNorth.salesID = SalesSouth.salesID
B. SELECT SalesNorth.salesID, SalesNorth.customer, SalesNorth.amount, SalesSouth.salesID, SalesSouth.customer, SalesSouth.amountFROM SalesNorth LEFT JOIN SalesSouthON SalesNorth.salesID=SalesSouth.salesID
C. SELECT salesID, customer, amount FROM SalesNorthUNION ALLSELECT salesID, customer, amount FROM SalesSouth
D. SELECT salesID, customer, amount FROM SalesNorthUNIONSELECT salesID, customer, amountFROM SalesSouth

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

NEW QUESTION 138
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
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
```

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.

Answer:

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 142
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+ISNULL(UnitsOnOnrder,0)) AS
TotalUnitPrice FROM Products
```

Does the solution meet the goal?

A. Yes
B. No

Answer: A

Explanation: ISNULL (check_expression , replacement_value) Arguments:

check_expression
Is the expression to be checked for NULL. check_expression can be of any type. replacement_value
Is the expression to be returned if check_expression is NULL. replacement_value must be of a type that is implicitly convertible to the type of check_expresssion.
References: https://docs.microsoft.com/en-us/sql/t-sql/functions/isnull-transact-sql

NEW QUESTION 144
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 audit all customer data.
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 G

Answer: B

Explanation: The FOR SYSTEM_TIME ALL clause returns all the row versions from both the Temporal and History table. Note: A system-versioned temporal table defined through is a new type of user table in SQL Server 2016, here defined on the last line WITH (SYSTEM_VERSIONING = ON…, is designed to keep a full history of data changes and allow easy point in time analysis.
To query temporal data, the SELECT statement FROM<table> clause has a new clause FOR SYSTEM_TIME with five temporal-specific sub-clauses to query data across the current and history tables.
References: https://msdn.microsoft.com/en-us/library/dn935015.aspx

NEW QUESTION 147
You have a database named DB1 that contains two tables named Sales.Customers and Sales.CustomerTransaction. Sales.CustomerTransactions has a foreign key relationship to column named CustomerID in Sales.Customers.
You need to recommend a query that returns the number of customers who never completed a transaction. Which query should you recommend?

A

```
SELECT
      COUNT(Cust.CustomerID)
FROM
      Sales.Customers Cust
            LEFT JOIN
      Sales.CustomerTransactions Trans
            ON Cust.CustomerID = Trans.CustomerID
WHERE
      Trans.CustomerTransactionID IS NULL;
```

B

```
SELECT
      COUNT(CustomerID)
FROM
      Sales.Customers Cust
            LEFT JOIN
      Sales.CustomerTransactions Trans
            ON Cust.CustomerID = Trans.CustomerID
WHERE
      Trans.CustomerTransactionID IS NULL;
```

C

```
SELECT
      COUNT(Cust.CustomerID)
FROM
      Sales.Customers Cust
            LEFT JOIN
      Sales.CustomerTransactions Trans
            ON Cust.CustomerID = Trans.CustomerID
```

D

```
SELECT
      COUNT(Cust.CustomerID)
FROM
      Sales.Customers Cust
            INNER JOIN
      Sales.CustomerTransactions Trans
            ON Cust.CustomerID = Trans.CustomerID
WHERE
      Trans.CustomerTransactionID IS NULL;
```

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

Answer: A

NEW QUESTION 149
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 ALWAIS 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 154
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 157
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
```
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
```
B.

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
```
C.

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
```
D.

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

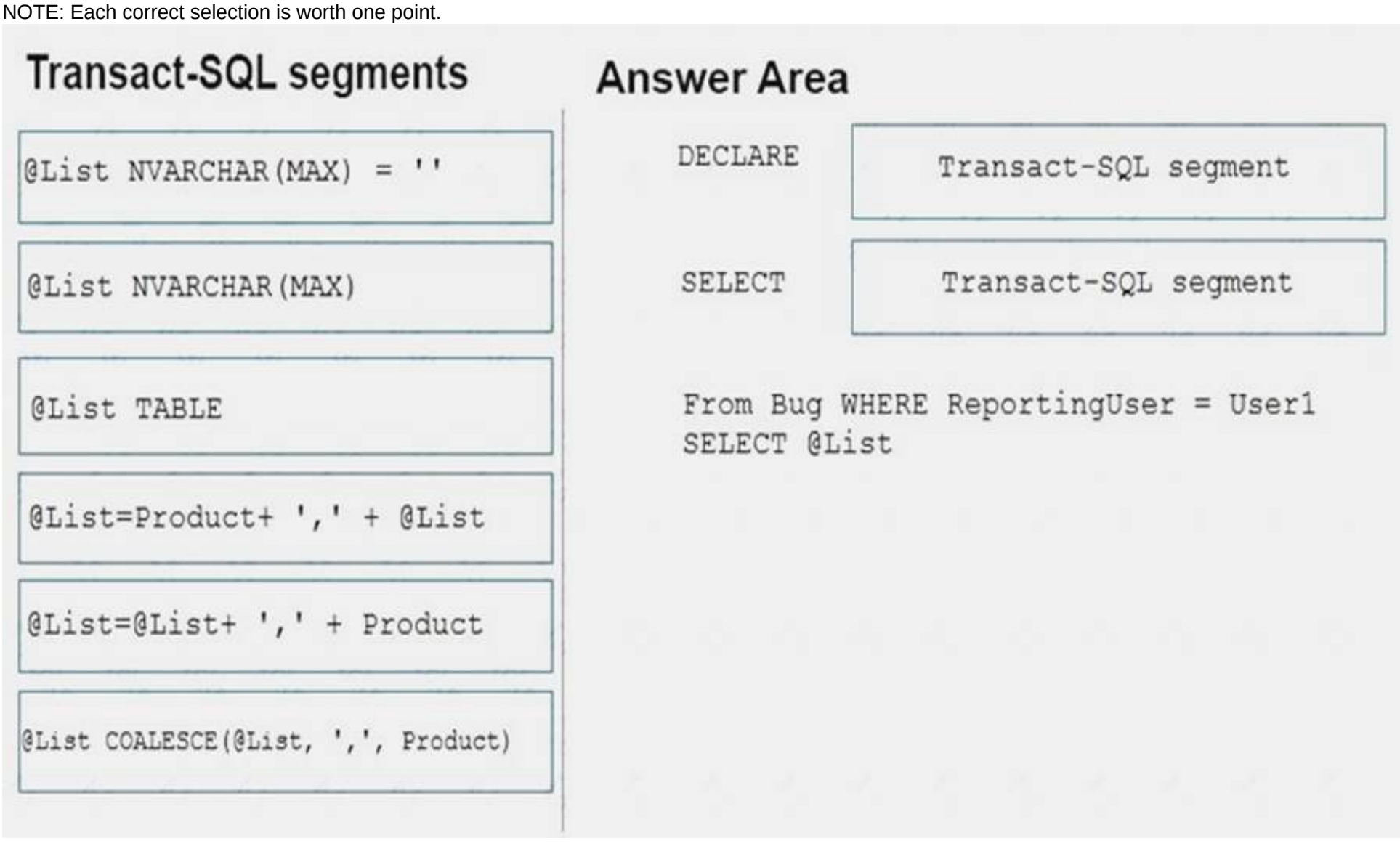
Answer: B

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 159
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.



Answer:

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

NEW QUESTION 164
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 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 only to that question.
You have a table named Products that contains information about the products that your company sells. The table contains many columns that do not always

contain values.
You need to implement an ANSI standard method to convert the NULL values in the query output to the phrase “Not Applicable”.
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: F

Explanation: The ISNULL function replaces NULL with the specified replacement value. References: https://msdn.microsoft.com/en-us/library/ms184325.aspx

NEW QUESTION 171
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 SalesOrders.
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(*)
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 172
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
```

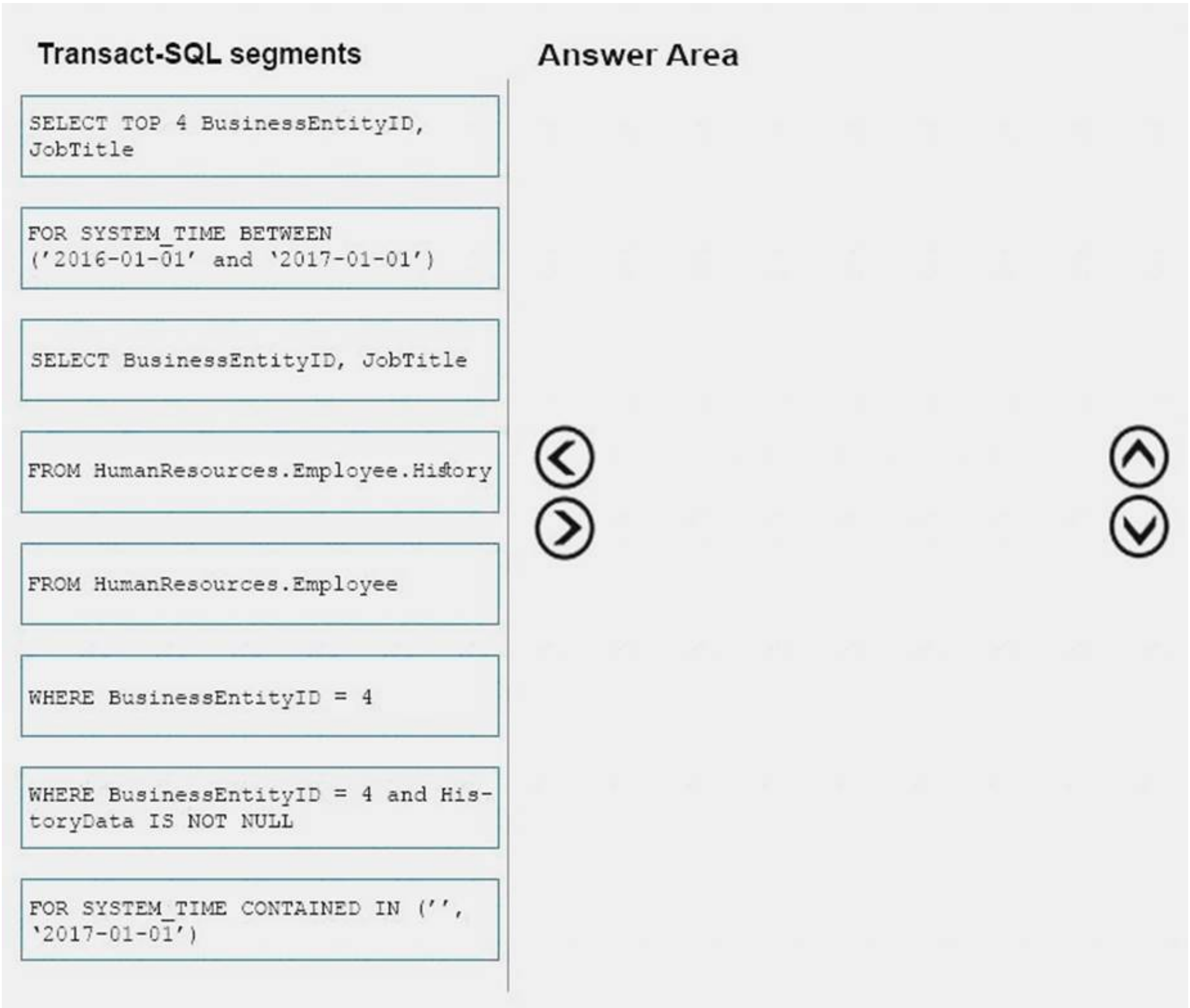
Does the solution meet the goal?

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 177
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.



Answer:

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

NEW QUESTION 180
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, StateProviceName, CityName, Quantity*UnitPrice as TotalSales
02  FROM Sales
03
```

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

CountryName	StateProvinceName	CityName	TotalSales
United States	Wyoming	Yoder	$7638.11
United States	Wyoming	NULL	$1983745.99
United States	NULL	NULL	$2387435981.22
NULL	NULL	NULL	$2387435981.22

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: F

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 185
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 187
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 does NOT include columns. Does this meet the goal?

A. YES
B. NO

Answer: A

Explanation: References:
https://docs.microsoft.com/en-us/sql/relational-databases/indexes/clustered-and-nonclustered-indexes-described?

NEW QUESTION 190
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
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.

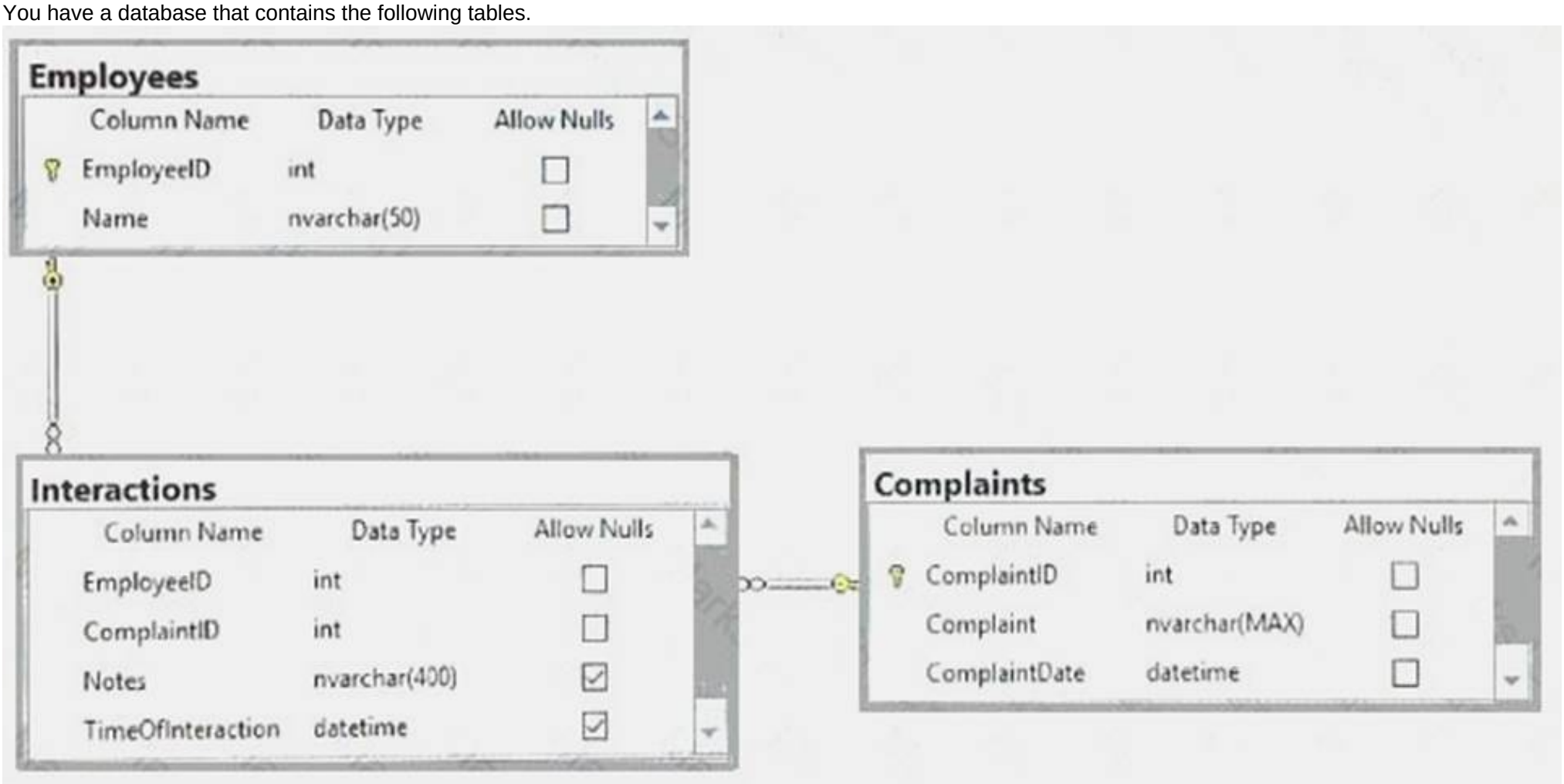
```
1 DELETE from sales.orders where status='calceled'
```

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

Answer:

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 194
You have a database that contains the following tables.



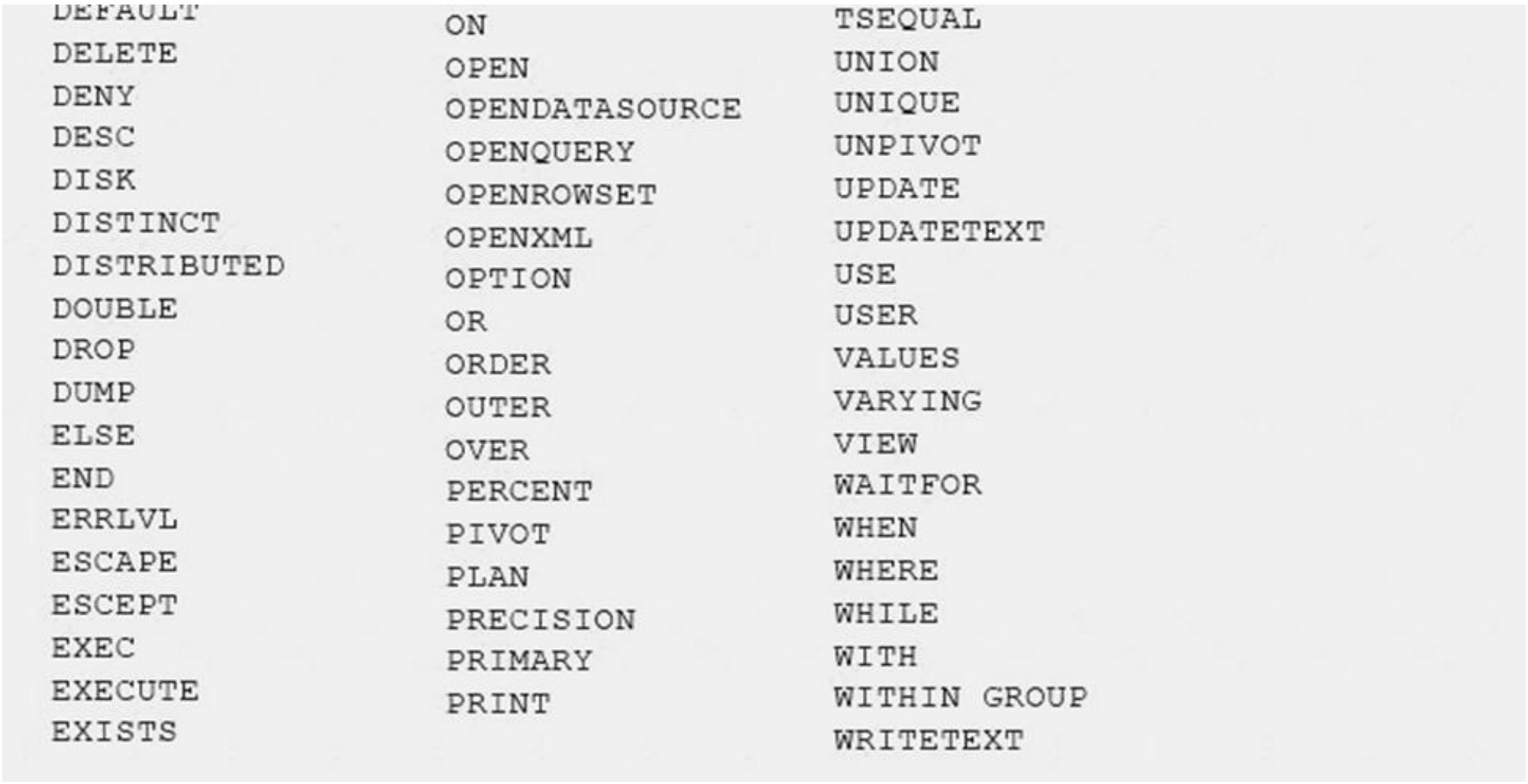
You need to create a query that returns each complaint, the names of the employees handling the complaint, and the notes on each interaction. The Complaint field must be displayed first, followed by the employee's name and the notes. Complaints must be returned even if no interaction has occurred.
Construct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

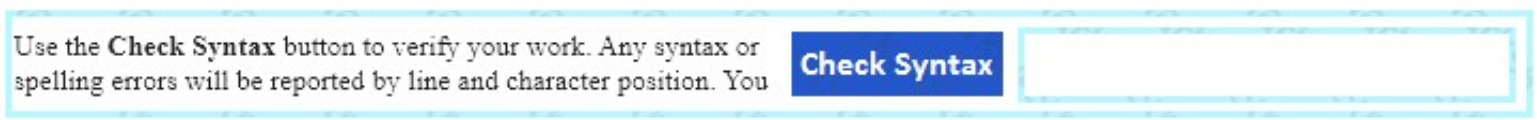
Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the code that has been provided as well as below it.

Keywords

ADD	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



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



Answer:

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 198
You need to develop a function that returns a list of courses grouped by the total number of students in a course.
The function must list only courses that have more than a specific number of students. The specific number of students is defined as an input variable for the function.
How should you complete the function? To answer, select the appropriate Transact-SQL segments in the answer area.
NOTE: Each correct selection is worth one point.

```
CREATE FUNCTION CoursesWithMoreThan (@totalStudents INT)
RETURNS [ ▼ ]
AS
RETRUN
 SELECT c.Course, [ ▼ ]
FROM dbo.Courses c
INNER JOIN dbo.CourseStudents cp ON c.CourseID = cp.CourseID
[ ▼ ] SUM(cp.NumStudents) > @totalStudents
```

RETURNS / SELECT c.Course, / [▼]:
- TABLE
- HAVING
- WHERE
- INT
- SUM(cp.NumStudents) AS NumStudents
- SUM(cp.NumStudents)

Answer:

Explanation:

```
CREATE FUNCTION CoursesWithMoreThan (@totalStudents INT)
RETURNS [TABLE ▼]
AS
RETRUN
 SELECT c.Course, [SUM(cp.NumStudents) AS NumStudents ▼]
FROM dbo.Courses c
INNER JOIN dbo.CourseStudents cp ON c.CourseID = cp.CourseID
[WHERE ▼] SUM(cp.NumStudents) > @totalStudents
```

RETURNS: TABLE, HAVING, WHERE, INT, SUM(cp.NumStudents) AS NumStudents, SUM(cp.NumStudents)

SELECT c.Course, [▼]: TABLE, HAVING, WHERE, INT, SUM(cp.NumStudents) AS NumStudents, SUM(cp.NumStudents)

[▼] SUM(cp.NumStudents) > @totalStudents: TABLE, HAVING, WHERE, INT, SUM(cp.NumStudents) AS NumStudents, SUM(cp.NumStudents)

NEW QUESTION 203
You have a table named Cities that has the following two columns: CityID and CityName. The CityID column uses the int data type, and CityName uses nvarchar(max).
You have a table named RawSurvey. Each row includes an identifier for a question and the number of persons that responded to that question from each of four cities. The table contains the following representative data:

QuestionID	Tokyo	Boston	London	New York
Q1	1	42	48	51
Q2	22	39	58	42
Q3	29	41	61	33
Q4	62	70	60	50
Q5	63	31	41	21
Q6	32	1	16	34

A reporting table named SurveyReport has the following columns: CityID, QuestionID, and RawCount, where RawCount is the value from the RawSurvey table.
You need to write a Transact-SQL query to meet the following requirements:
Retrieve data from the RawSurvey table in the format of the SurveyReport table.
The CityID must contain the CityID of the city that was surveyed.
The order of cities in all SELECT queries must match the order in the RawSurvey table.
The order of cities in all IN statements must match the order in the RawSurvey table.
Construct the query using the following guidelines:
Use one-part names to reference tables and columns, except where not possible.
ALL SELECT statements must specify columns.
Do not use column or table aliases, except those provided.
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
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
```

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 Rawcount
2    from (select cityid,questionid,rawcount) AS t1
3    unpivot
4    (rawcount for questionid in (QuestionID)) AS t2
```

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

Answer:

Explanation: 1 SELECT Rawcount
2 from (select cityid,questioned,rawcount) AS t1
3 unpivot
4 (rawcount for questioned in (QuestionID)) AS t2
5 JOIN t2
6. ON t1.CityName = t2.cityName
UNPIVOT must be used to rotate columns of the Rawsurvey table into column values. References: https://technet.microsoft.com/en-us/library/ms177410(v=sql.105).aspx

NEW QUESTION 204
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 208
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 211
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:

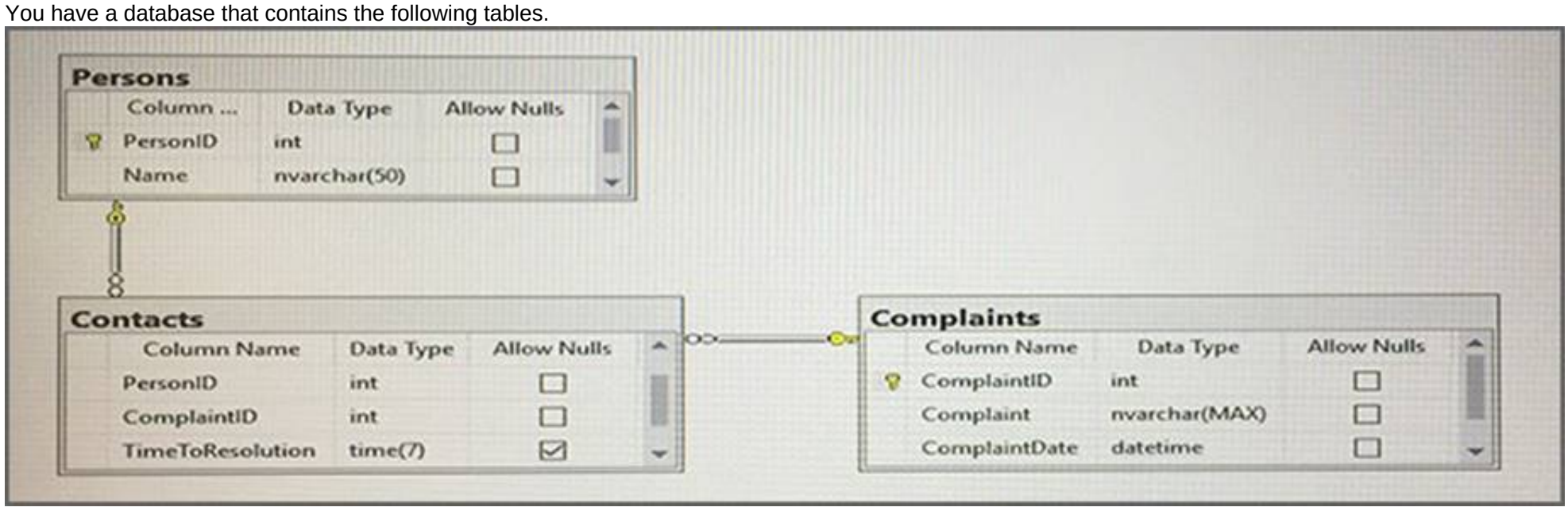
```
ALTER PROCEDURE InsertProduct
@ProductName nvarchar (100),
@UnitPrice decimal (18, 2),
@UnitsInStock int,
@UnitsOnOrder int
AS
BEGIN
  BEGIN TRY
    BEGIN TRANSACTION
      INSERT INTO Products (ProductName, UnitPrice, UnitsInStock, UnitsOnOrder)
      VALUES (@ProductName, @UnitPrice, @UnitsInStock, @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 216
You have a database that contains the following tables.



You need to create a query that lists all complaints from the Complaints table, and the name of the person handling the complaints if a person is assigned. The ComplaintID must be displayed first, followed by the person name.
Construct the query using the following guidelines:
- Use two-part column names.
- Use one-part table names.
- Do not use aliases for column names or table names.
- Do not use 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 Complaints.ComplaintId,
2 FROM
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.

Answer:

Explanation: SELECT
Complaints.ComplaintID, Persons.Name FROM
Complaints LEFT OUTER JOIN Contacts ON Complaints.ComplaintID = Contacts.ComplaintID
LEFT OUTER JOIN Persons ON Contacts.PersonID = Persons.PersonID

NEW QUESTION 221
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 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 inserted.ProjectId INTO @startedTasks WHERE StartTime is NULL
SELECT COUNT(*) FROM @startedTasks WHERE ProjectId IS NOT NULL
```

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

```
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
```

D

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

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

Answer: B

NEW QUESTION 223
You have a database named DB1 that contains a temporal table named Sales.Customers.
You need to create a query that returns the credit limit that was available to each customer in DB1 at the beginning of 2017.
Which query should you execute?

A

```
SELECT
      CustomerID,
      CustomerName,
      CreditLimit
 FROM
       Sales.Customers
            FOR SYSTEM_TIME CONTAINED IN ('2017-01-01 00:00:00'));
```

B

```
SELECT
      CustomerID,
      CustomerName,
      CreditLimit
FROM
      Sales.Customers
           FOR SYSTEM_TIME AS OF '2017-01-01 00:00:00';
```

C

```
SELECT
      CustomerID,
      CustomerName,
      CreditLimit
FROM
      Sales.Customers
            FOR SYSTEM_TIME CONTAINED IN ('2016-12-31', '2017-01-01');
```

D

```
SELECT
      CustomerID,
      CustomerName,
      CreditLimit
FROM
      Sales.Customers
            FOR SYSTEM_TIME BETWEEN '2016-12-31' AND '2017-01-01');
```

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

Answer: B

NEW QUESTION 226
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 231
You have the following Transact-SQL statement: DELETE FROM Person
WHERE PersonID = 5
You need to implement error handling.
How should you complete Transact-SQL statement? To answer, select the appropriate options in the answer area.
NOTE: Each correct selection is worth one point.

```
BEGIN TRANSACTION
```

[▼]
- END TRY
- COMMIT TRANSACTION
- END CATCH
- BEGIN TRY
- ROLLBACK TRANSACTION
- BEGIN CATCH

```
    DELETE FROM Person
    WHERE PersonID = 5
```

[▼]
- END TRY
- COMMIT TRANSACTION
- END CATCH
- BEGIN TRY
- ROLLBACK TRANSACTION
- BEGIN CATCH

[▼]
- END TRY
- COMMIT TRANSACTION
- END CATCH
- BEGIN TRY
- ROLLBACK TRANSACTION
- BEGIN CATCH

```
IF @@TRANCOUNT > 0
```

[▼]
- END TRY
- COMMIT TRANSACTION
- END CATCH
- BEGIN TRY
- ROLLBACK TRANSACTION
- BEGIN CATCH

[▼]
- END TRY
- COMMIT TRANSACTION
- END CATCH
- BEGIN TRY
- ROLLBACK TRANSACTION
- BEGIN CATCH

```
IF @@TRANCOUNT > 0
 COMMIT TRANSACTION
```

Answer:

Explanation:

```
BEGIN TRANSACTION
```

- END TRY
- COMMIT TRANSACTION
- END CATCH
- **BEGIN TRY**
- ROLLBACK TRANSACTION
- BEGIN CATCH

```
    DELETE FROM Person
    WHERE PersonID = 5
```

- **END TRY**
- COMMIT TRANSACTION
- END CATCH
- BEGIN TRY
- ROLLBACK TRANSACTION
- BEGIN CATCH

- END TRY
- COMMIT TRANSACTION
- END CATCH
- BEGIN TRY
- ROLLBACK TRANSACTION
- **BEGIN CATCH**

```
IF @@TRANCOUNT > 0
```

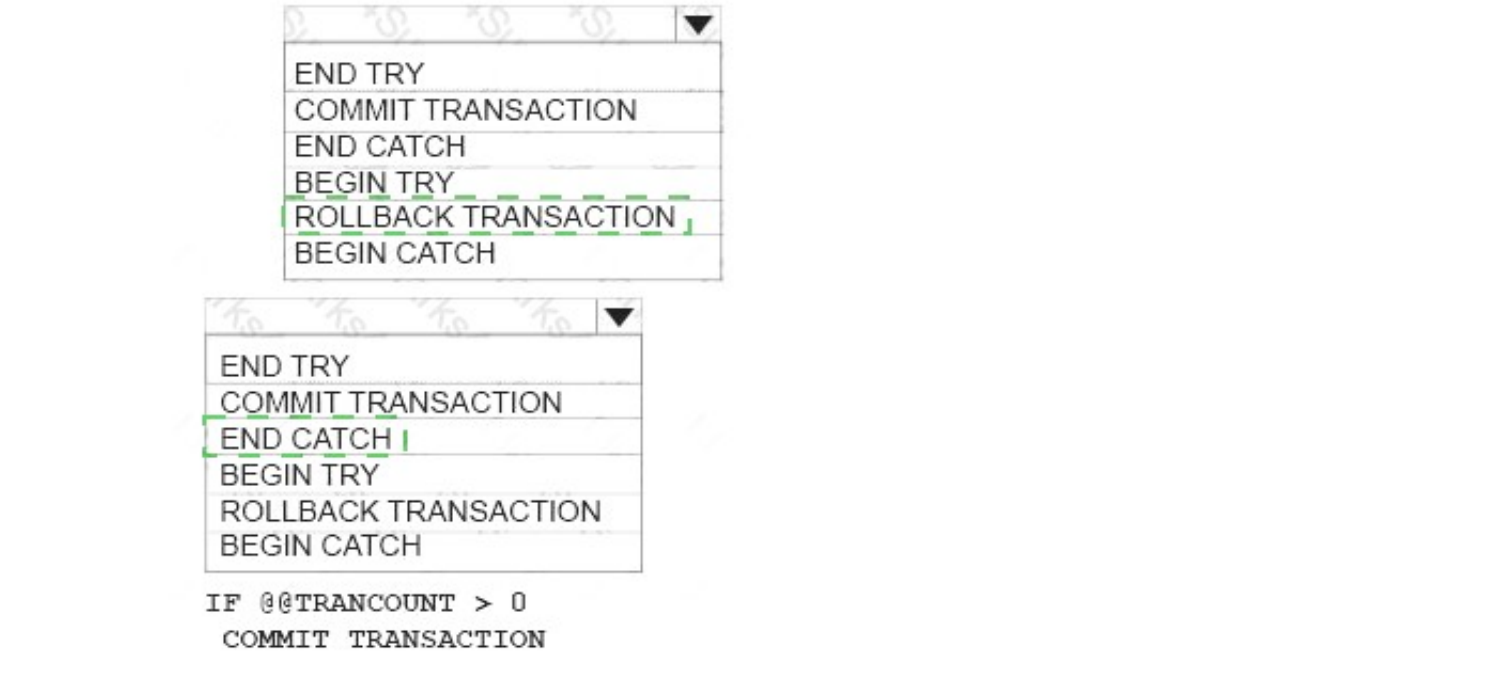
- END TRY
- COMMIT TRANSACTION
- END CATCH
- **BEGIN TRY**
- **ROLLBACK TRANSACTION**
- BEGIN CATCH

- END TRY
- **COMMIT TRANSACTION**
- **END CATCH**
- BEGIN TRY
- ROLLBACK TRANSACTION
- BEGIN CATCH

```
IF @@TRANCOUNT > 0
 COMMIT TRANSACTION
```

NEW QUESTION 235

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.
A database has two tables as shown in the following database diagram:



You need to list all provinces that have at least two large cities. A large city is defined as having a population of at least one million residents. The query must return the following columns:

- tblProvince.ProvinceId
- tblProvince.ProvinceName
- a derived column named LargeCityCount that presents the total count of large cities for the province

Solution: You run the following Transact-SQL statement:

```
SELECT P.ProvinceId, P.ProvinceName, CitySummary.LargeCityCount
FROM tblProvince P
CROSS JOIN (
   SELECT COUNT(*) AS LargeCityCount FROM tblCity C
   WHERE C.Population>=1000000
) CitySummary
WHERE CitySummary.LargeCityCount >=2
```

Does the solution meet the goal?

A. Yes
B. No

Answer: B

Explanation: The SQL CROSS JOIN produces a result set which is the number of rows in the first table multiplied by the number of rows in the second table if no WHERE clause is used along with CROSS JOIN.This kind of result is called as Cartesian Product.
This is not what is required in this scenario.
References: https://technet.microsoft.com/en-us/library/ms190690(v=sql.105).aspx

NEW QUESTION 238
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 241
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 242
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:

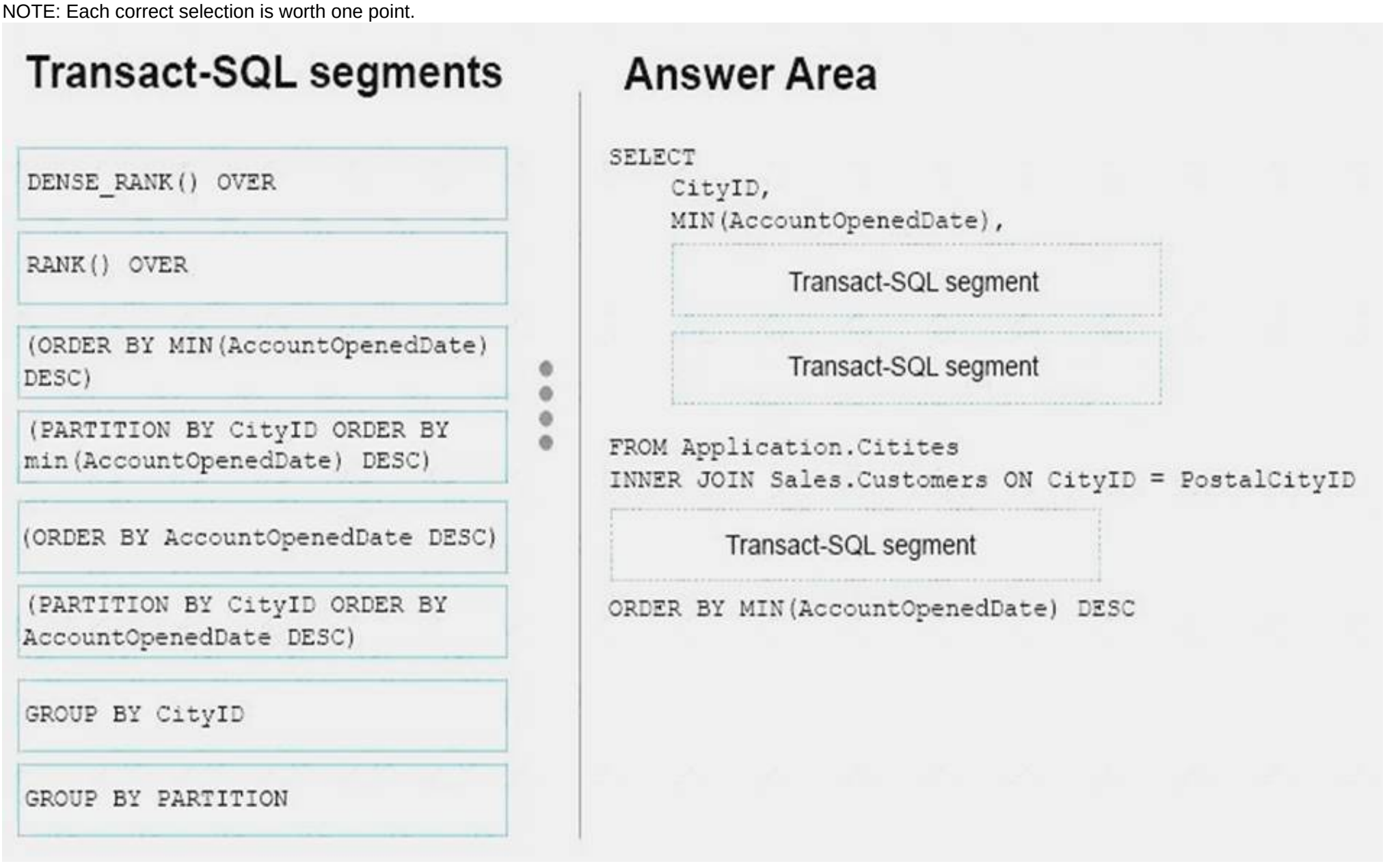
Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

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 are creating a report to show when the first customer account was opened in each city. The report contains a line chart with the following characteristics:
- The chart contains a data point for each city, with lines connecting the points.
- The X axis contains the position that the city occupies relative to other cities.
- The Y axis contains the date that the first account in any city was opened. An example chart is shown below for five cities:

During a sales promotion, customers from various cities open new accounts on the same date. You need to write a query that returns the data for the chart.
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.



Answer:

Explanation: Box 1: RANK() OVER
RANK returns the rank of each row within the partition of a result set. The rank of a row is one plus the number of ranks that come before the row in question.
ROW_NUMBER and RANK are similar. ROW_NUMBER numbers all rows sequentially (for example 1, 2, 3, 4, 5).

NEW QUESTION 243

.....

THANKS FOR TRYING THE DEMO OF OUR PRODUCT

Visit Our Site to Purchase the Full Set of Actual 70-761 Exam Questions With Answers.

We Also Provide Practice Exam Software That Simulates Real Exam Environment And Has Many Self-Assessment Features. Order the 70-761 Product From:

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

Money Back Guarantee

70-761 Practice Exam Features:

* 70-761 Questions and Answers Updated Frequently

* 70-761 Practice Questions Verified by Expert Senior Certified Staff

* 70-761 Most Realistic Questions that Guarantee you a Pass on Your FirstTry

* 70-761 Practice Test Questions in Multiple Choice Formats and Updatesfor 1 Year