70-761$^{Q\&As}$

Querying Data with Transact-SQL

Pass Microsoft 70-761 Exam with 100% Guarantee

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

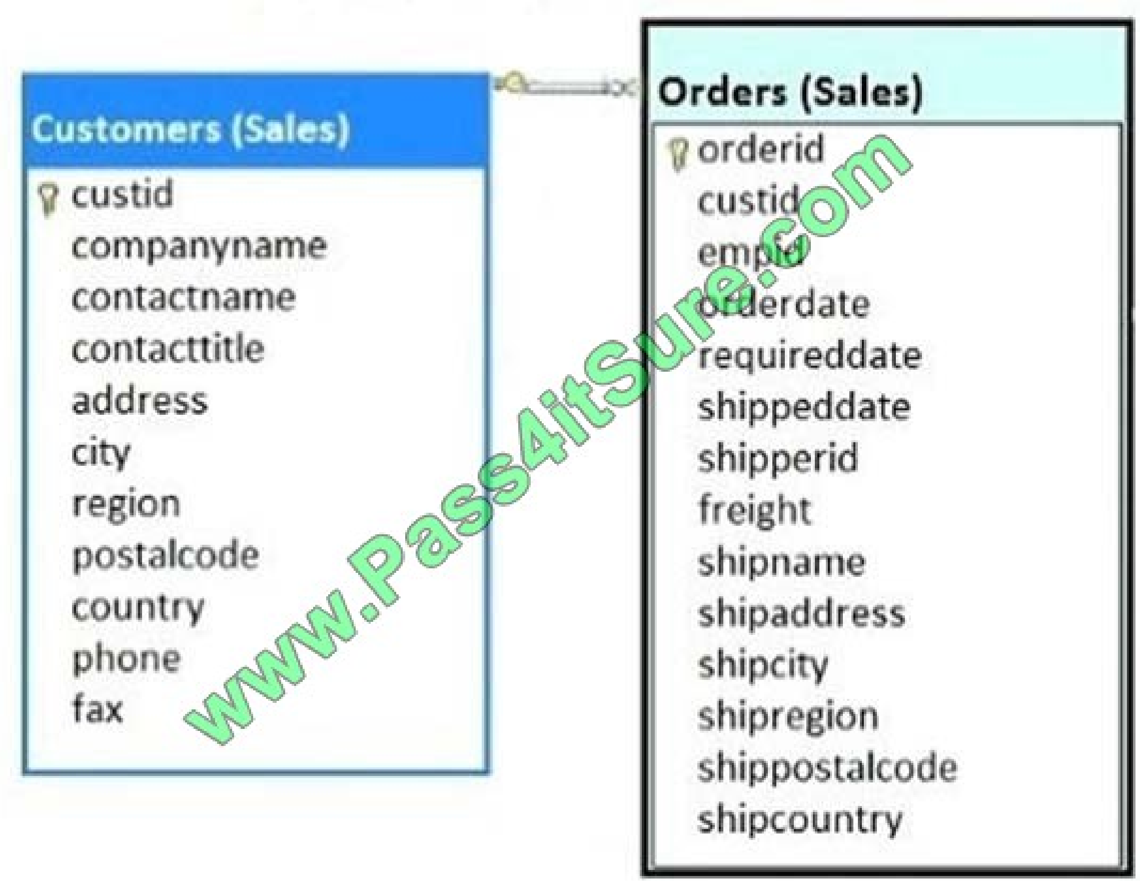
https://www.pass4itsure.com/70-761.html

100% Passing Guarantee
100% Money Back Assurance

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

QUESTION 1

You have a database that contains the following tables:



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

A. SELECT c.custid FROM Sales.Customers c INNER JOIN Sales.Order oON c.custid = o.custid

B. SELECT custid FROM Sales.Customers INTERSECTSELECT custid FROM Sales.Orders

C. SELECTc.custid FROM Sales.Customers c LEFT OUTER Sales.Order oON c.custid = o.custid

D. SELECT c.custid FROM Sales.Customers c LEFT OUTER JOIN Sales.Order oON c.custid = o.custid WHERE orderid IS NULL

E. SELECT custid FROM Sales.Customers UNION ALL SELECT custid FROM Sales.Orders

F. SELECT custid FROM Sales.Customers UNION SELECT custid FROM Sales.Orders

G. SELECT c.custid FROM Sales.Customers c RIGHT OUTER JOIN Sales.Orders o ON c.custid = o.custid

Correct Answer: D

Inner joins return rows only when there is at least one row from both tables that matches the join condition. Inner joins eliminate the rows that do not match with a row from the other table. Outer joins, however, return all rows from at least one of the tables or views mentioned in the FROM clause, as long as those rows meet any WHERE or HAVING search conditions. All rows are retrieved from the left table referenced with a left outer join, and all rows from the right table referenced in a right outer join. All rows from both tables are returned in a full outer join.

References: https://technet.microsoft.com/en-us/library/ms187518(v=sql.105).aspx

QUESTION 2

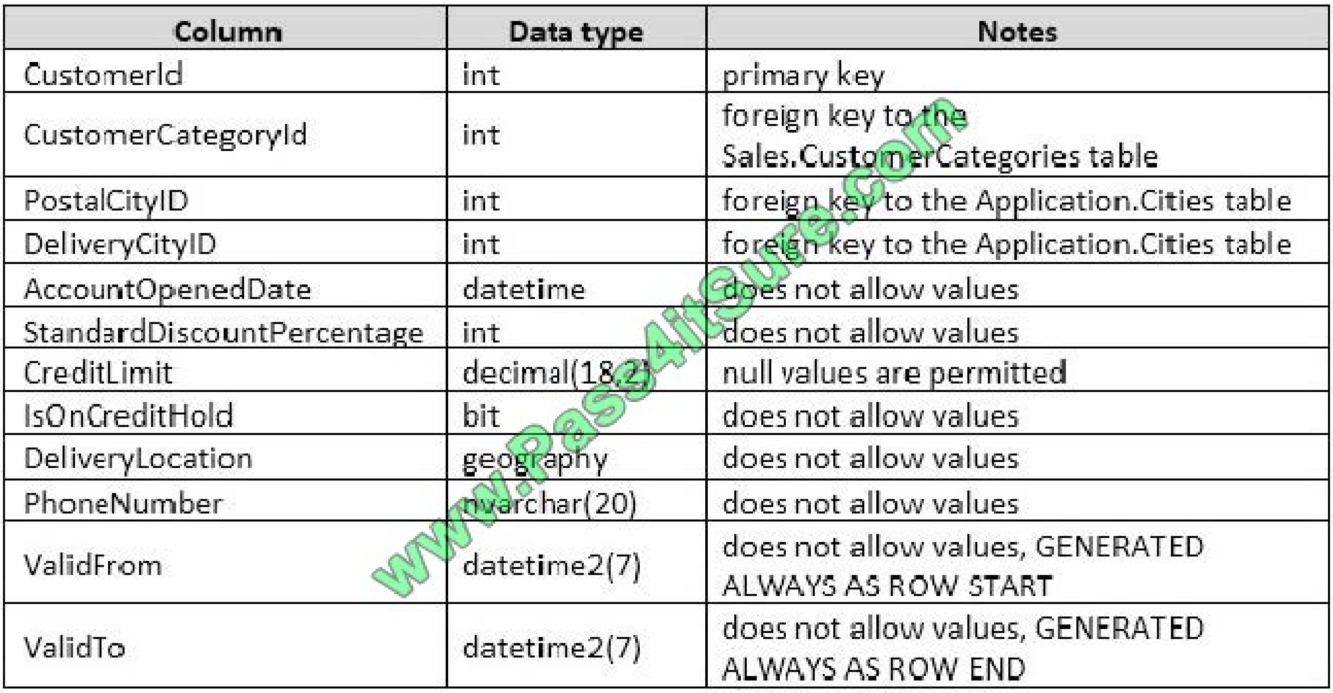
DRAG DROP

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

exactly the same in each question on this series.

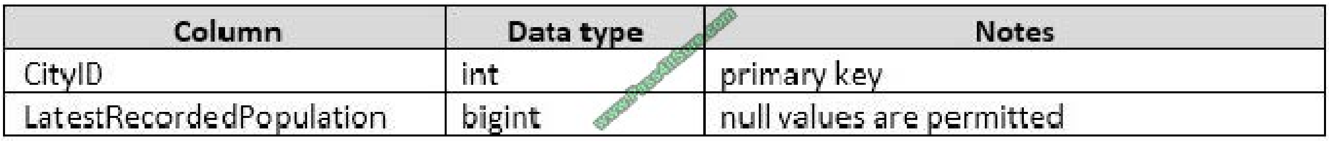
You have a database that tracks orders and deliveries for customers in North America. System versioning is enabled for all tables. The database contains the Sales.Customers, Application.Cities, and Sales.CustomerCategories tables.

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



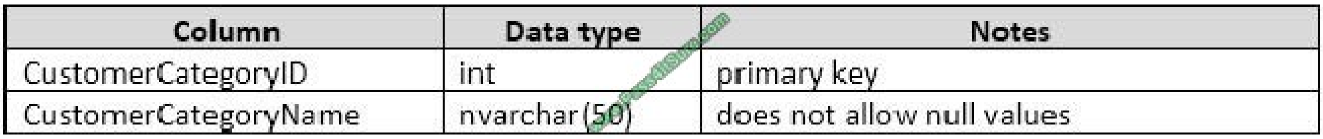
Column	Data type	Notes
CustomerId	int	primary key
CustomerCategoryId	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow values
StandardDiscountPercentage	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:



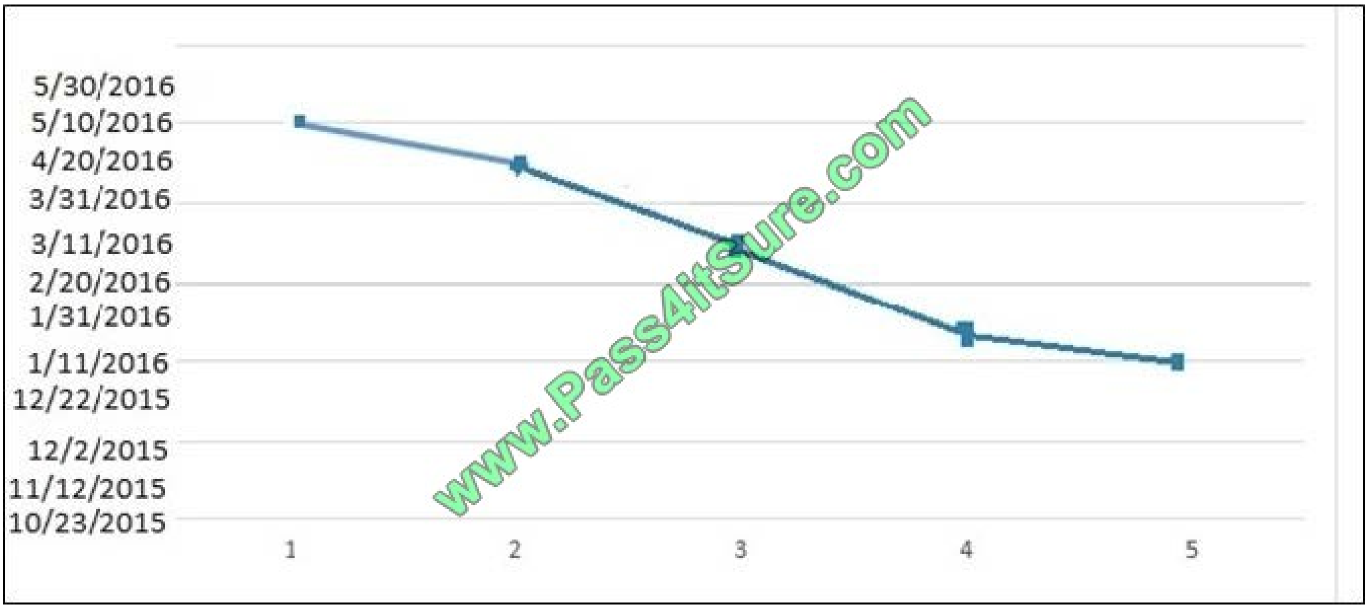
Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

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:

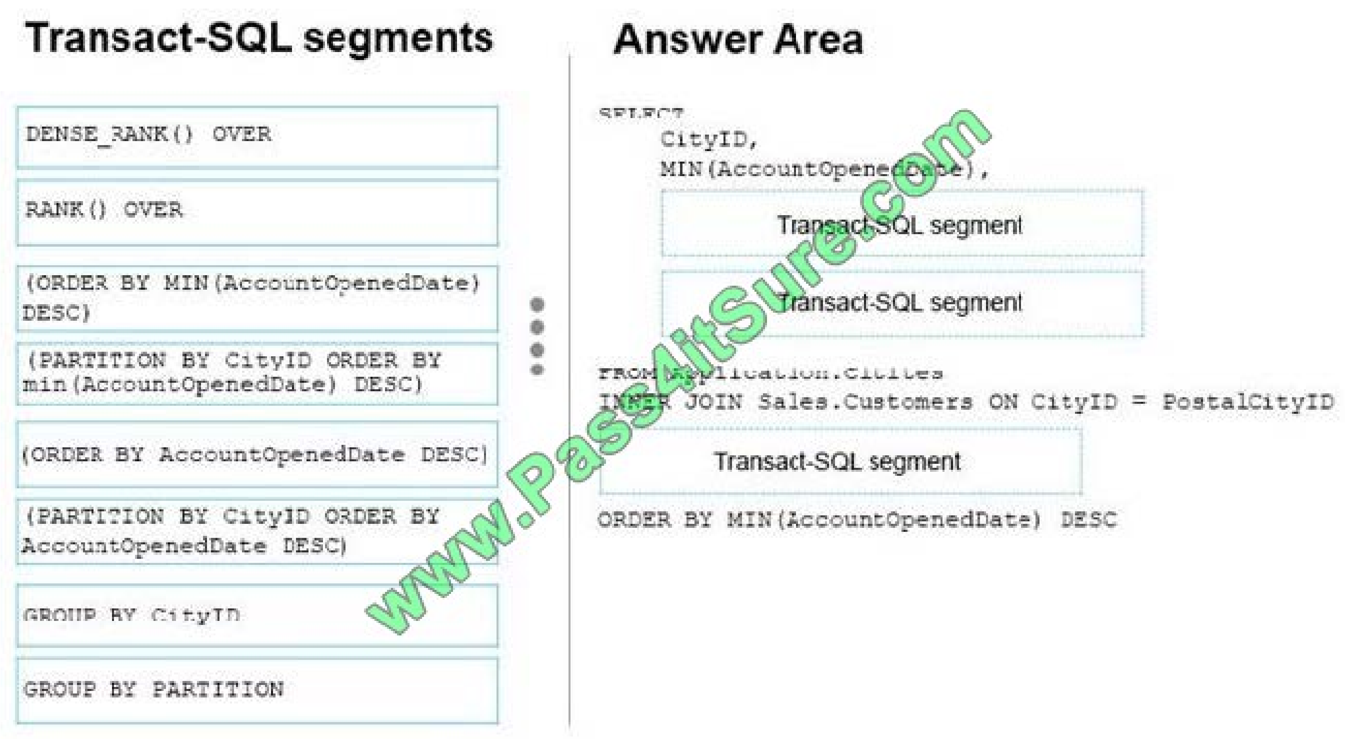


During a sales promotion, customers from various cities open new accounts on the same date. You need to write a query that returns the data for the chart.

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.

Select and Place:



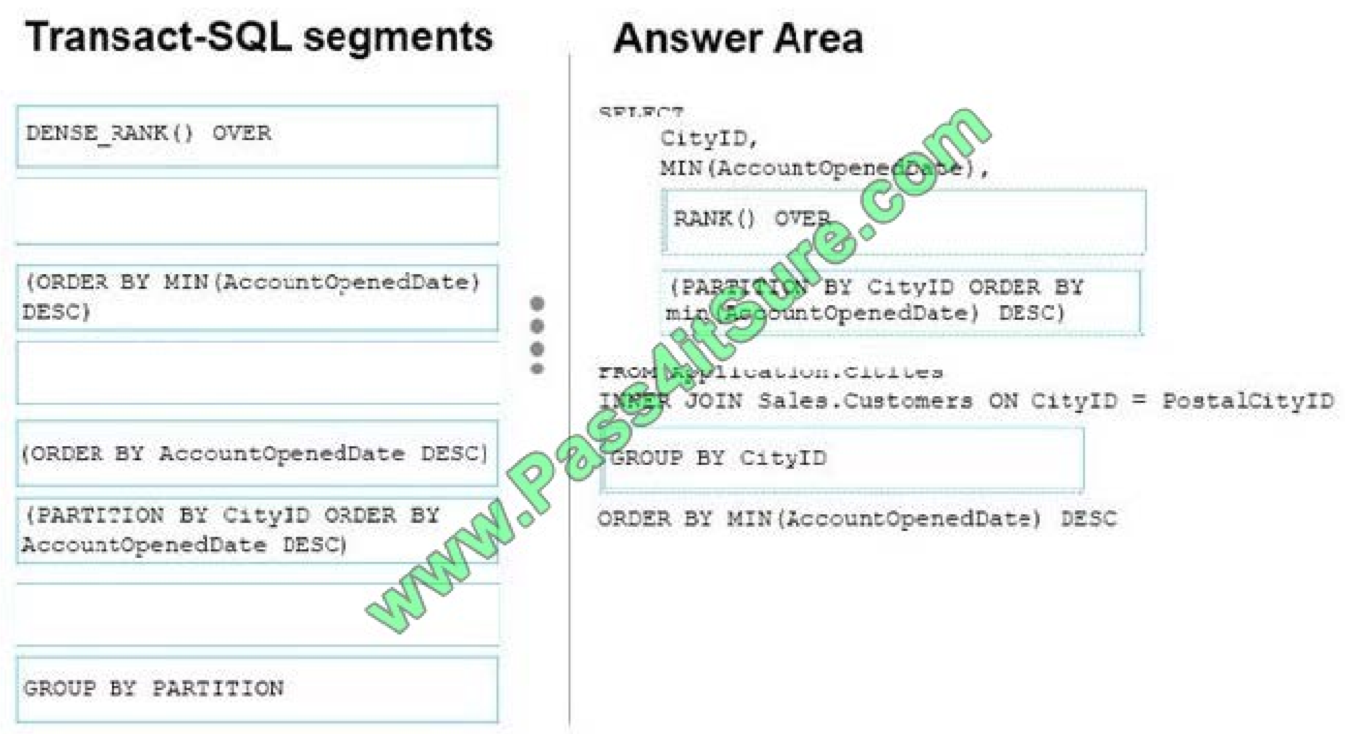
Transact-SQL segments

```
DENSE_RANK() OVER
RANK() OVER
(ORDER BY MIN(AccountOpenedDate) DESC)
(PARTITION BY CityID ORDER BY min(AccountOpenedDate) DESC)
(ORDER BY AccountOpenedDate DESC)
(PARTITION BY CityID ORDER BY AccountOpenedDate DESC)
GROUP BY CityID
GROUP BY PARTITION
```

Answer Area

```
SELECT
    CityID,
    MIN(AccountOpenedDate),
     Transact-SQL segment
     Transact-SQL segment
FROM Application.Cities
INNER JOIN Sales.Customers ON CityID = PostalCityID
 Transact-SQL segment
ORDER BY MIN(AccountOpenedDate) DESC
```

Correct Answer:



Transact-SQL segments

```
DENSE_RANK() OVER

(ORDER BY MIN(AccountOpenedDate) DESC)

(ORDER BY AccountOpenedDate DESC)
(PARTITION BY CityID ORDER BY AccountOpenedDate DESC)

GROUP BY PARTITION
```

Answer Area

```
SELECT
    CityID,
    MIN(AccountOpenedDate),
     RANK() OVER
     (PARTITION BY CityID ORDER BY min(AccountOpenedDate) DESC)
FROM Application.Cities
INNER JOIN Sales.Customers ON CityID = PostalCityID
 GROUP BY CityID
ORDER BY MIN(AccountOpenedDate) DESC
```

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 thenumber 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).

Incorrect Answers:

DENSE_RANK returns the rank of rows within the partition of a result set, without any gaps in the ranking. The rank of a row is one plus the number of distinct ranks that come before the row in question.

Box 2: (PARTITION BY CityID ORDER BY MIN(AccountOpenedDate) DESC)

Syntax for RANK: RANK ( ) OVER ([partition_by_clause] order_by_clause)

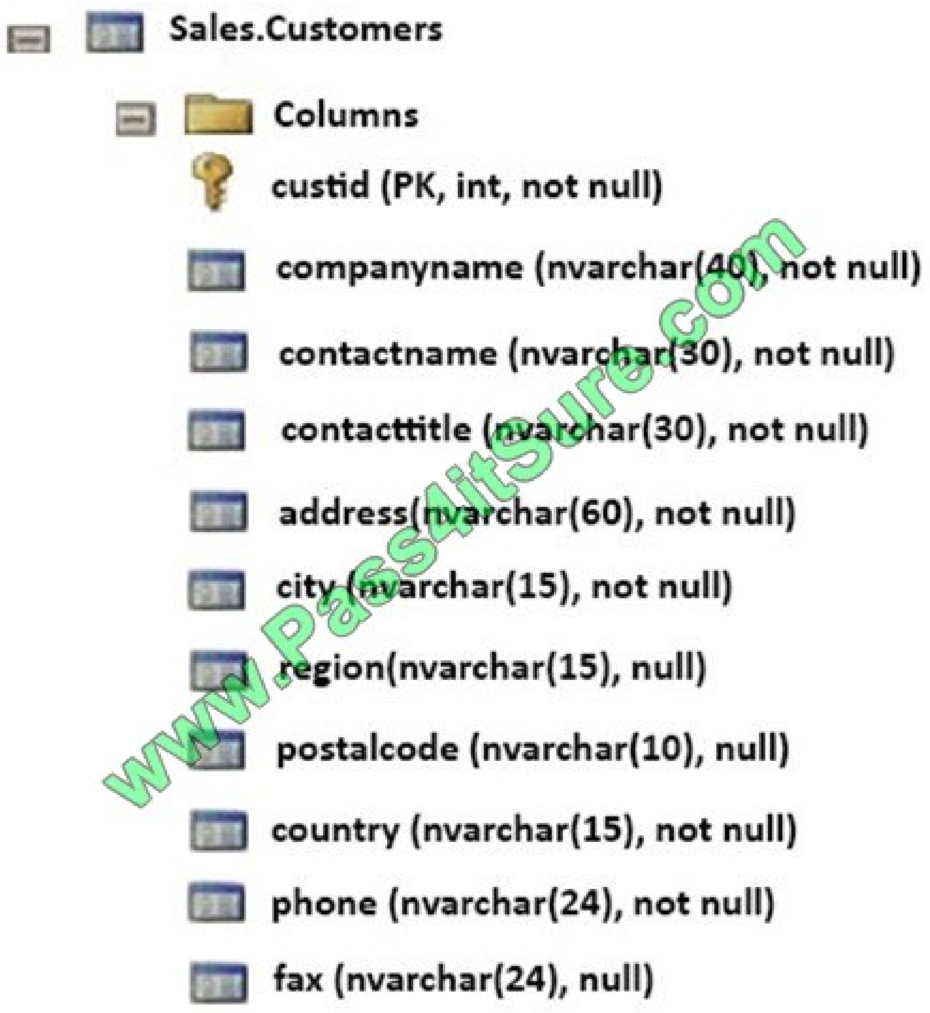
Box 3: GROUP BY CityID

References: https://msdn.microsoft.com/en-us/library/ms176102.aspx

QUESTION 3

DRAG DROP

You need to create a stored procedure to update a table named Sales.Customers. The structure of the table is shown in the exhibit. (Click the exhibit button.)



Sales.Customers
Columns
custid (PK, int, not null)
companyname (nvarchar(40), not null)
contactname (nvarchar(30), not null)
contacttitle (nvarchar(30), not null)
address(nvarchar(60), not null)
city (nvarchar(15), not null)
region(nvarchar(15), null)
postalcode (nvarchar(10), null)
country (nvarchar(15), not null)
phone (nvarchar(24), not null)
fax (nvarchar(24), null)

The stored procedure must meet the following requirements:

Accept two input parameters.

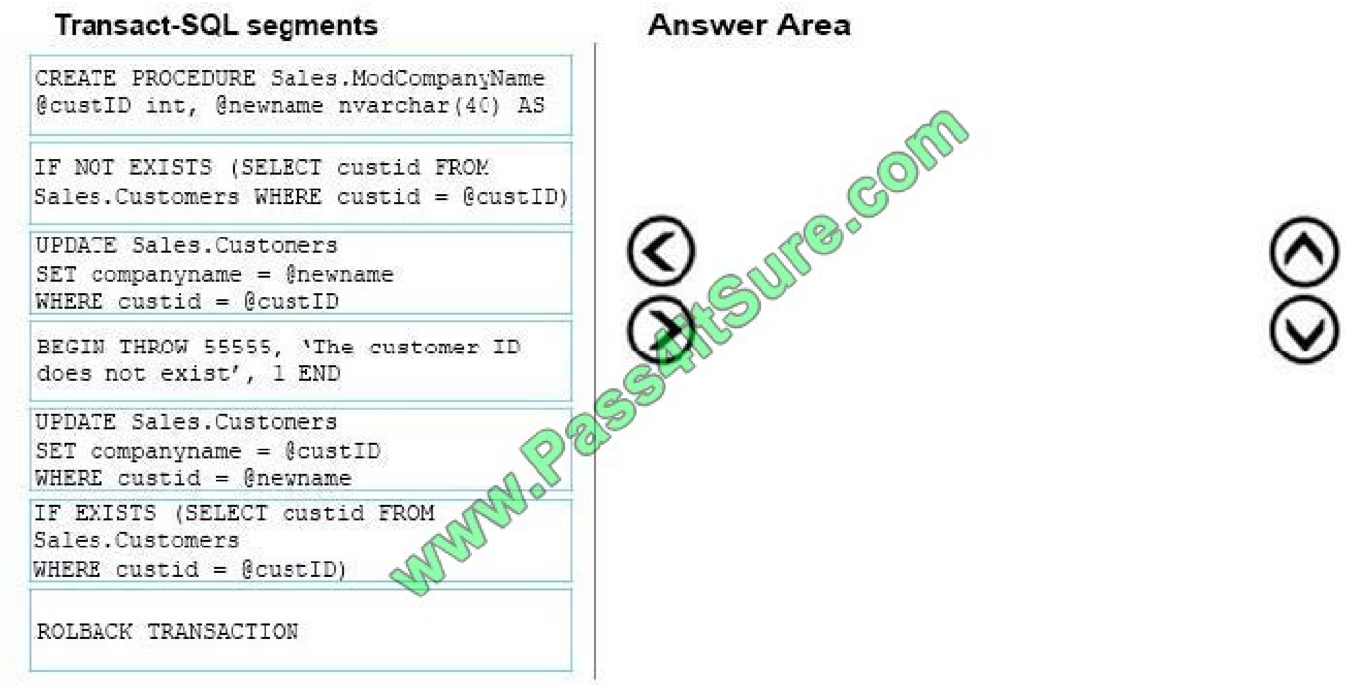
Update the company name if the customer exists.

Return a custom error message if the customer does not exist.

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.

NOTE: More than one order of answer choices is correct. You will receive credit for any of the correct orders you select.

Select and Place:



Transact-SQL segments

```
CREATE PROCEDURE Sales.ModCompanyName
@custID int, @newname nvarchar(40) AS
```

```
IF NOT EXISTS (SELECT custid FROM
Sales.Customers WHERE custid = @custID)
```

```
UPDATE Sales.Customers
SET companyname = @newname
WHERE custid = @custID
```

```
BEGIN THROW 55555, 'The customer ID
does not exist', 1 END
```

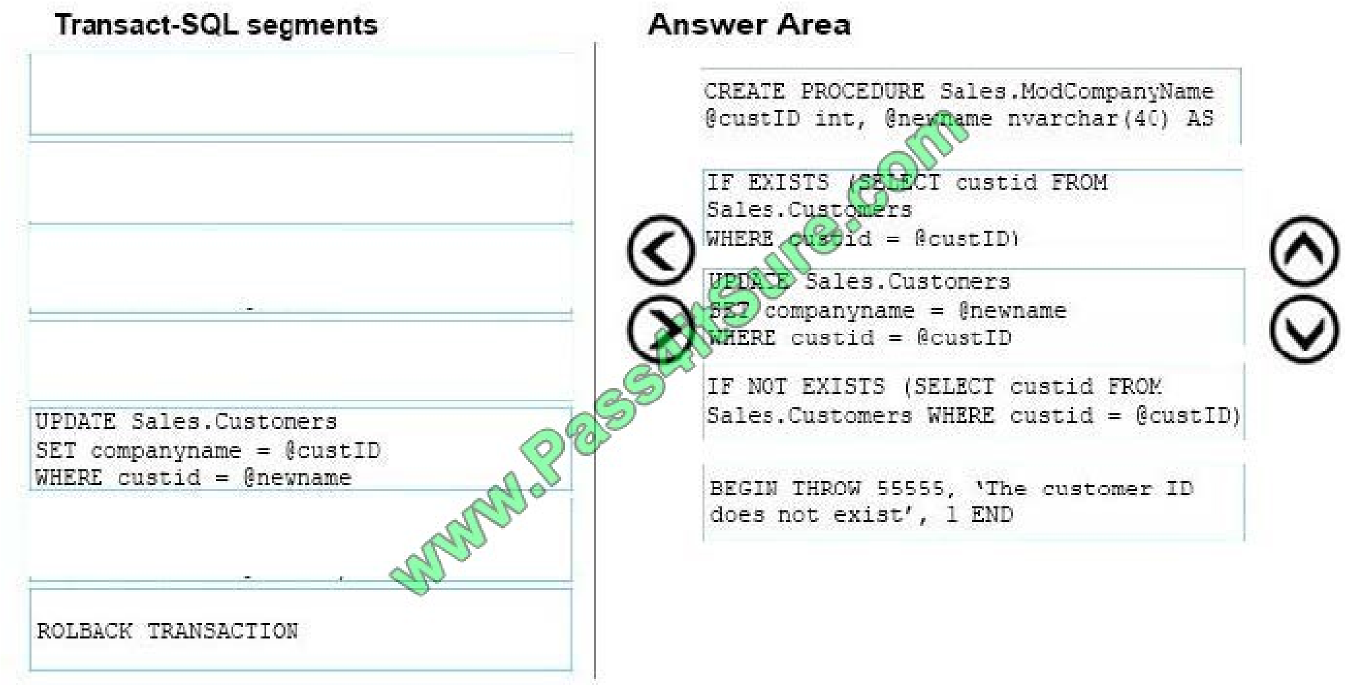
```
UPDATE Sales.Customers
SET companyname = @custID
WHERE custid = @newname
```

```
IF EXISTS (SELECT custid FROM
Sales.Customers
WHERE custid = @custID)
```

```
ROLBACK TRANSACTION
```

Answer Area

Correct Answer:



```
UPDATE Sales.Customers
SET companyname = @custID
WHERE custid = @newname
```

```
ROLBACK TRANSACTION
```

```
CREATE PROCEDURE Sales.ModCompanyName
@custID int, @newname nvarchar(40) AS
```

```
IF EXISTS (SELECT custid FROM
Sales.Customers
WHERE custid = @custID)
```

```
UPDATE Sales.Customers
SET companyname = @newname
WHERE custid = @custID
```

```
IF NOT EXISTS (SELECT custid FROM
Sales.Customers WHERE custid = @custID)
```

```
BEGIN THROW 55555, 'The customer ID
does not exist', 1 END
```

QUESTION 4

DRAG DROP

You have two tables named UserLogin and Employee respectively.

You need to create a Transact-SQL script that meets the following requirements:

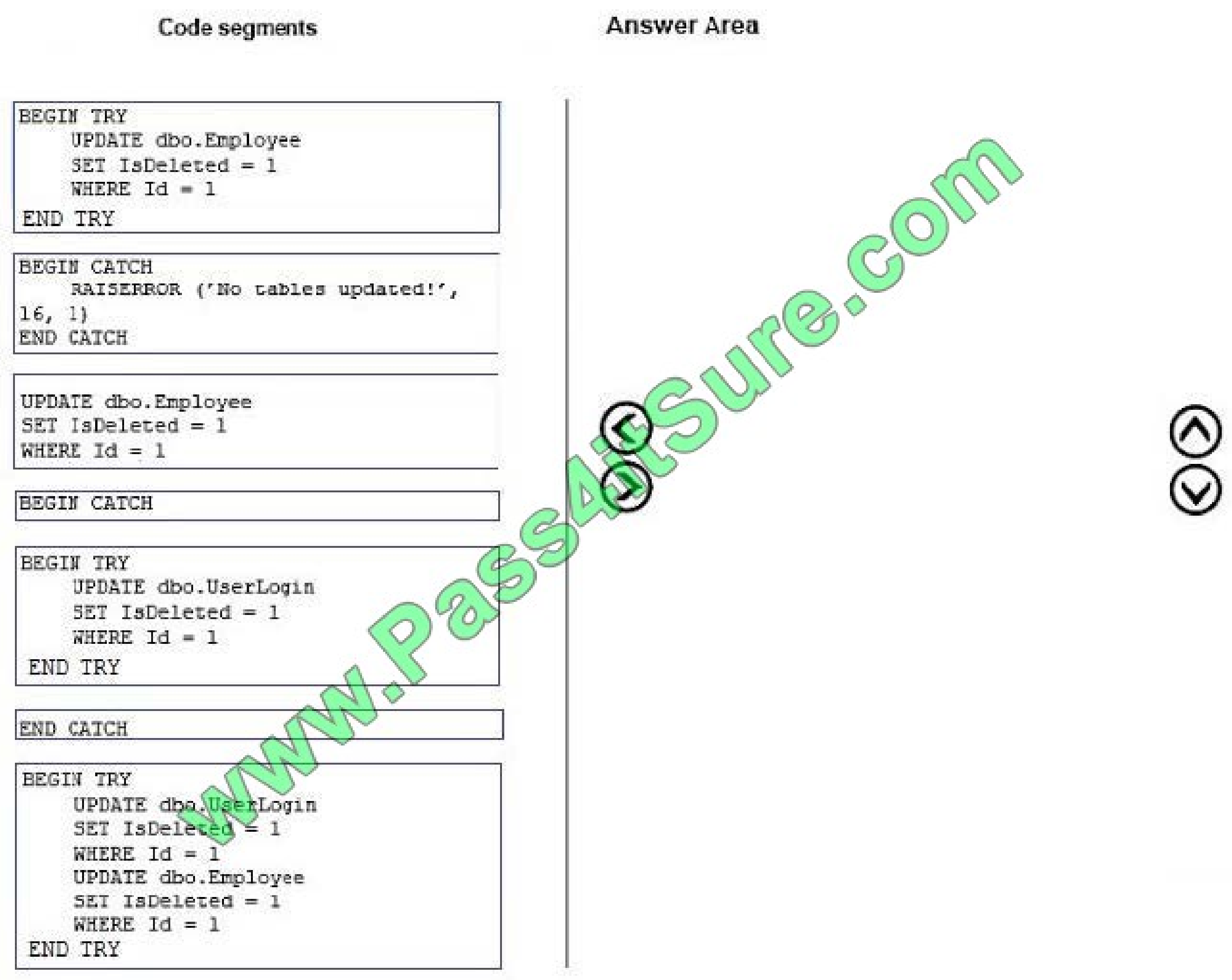
The script must update the value of the IsDeleted column for the UserLogin table to 1 if the value of the Id column for the UserLogin table is equal to 1.

The script must update the value of the IsDeleted column of the Employee table to 1 if the value of the Id column is equal to 1 for the Employee table when an update to the UserLogin table throws an error.

The error message "No tables updated!" must be produced when an update to the Employee table throws an error.

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

Select and Place:



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

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

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

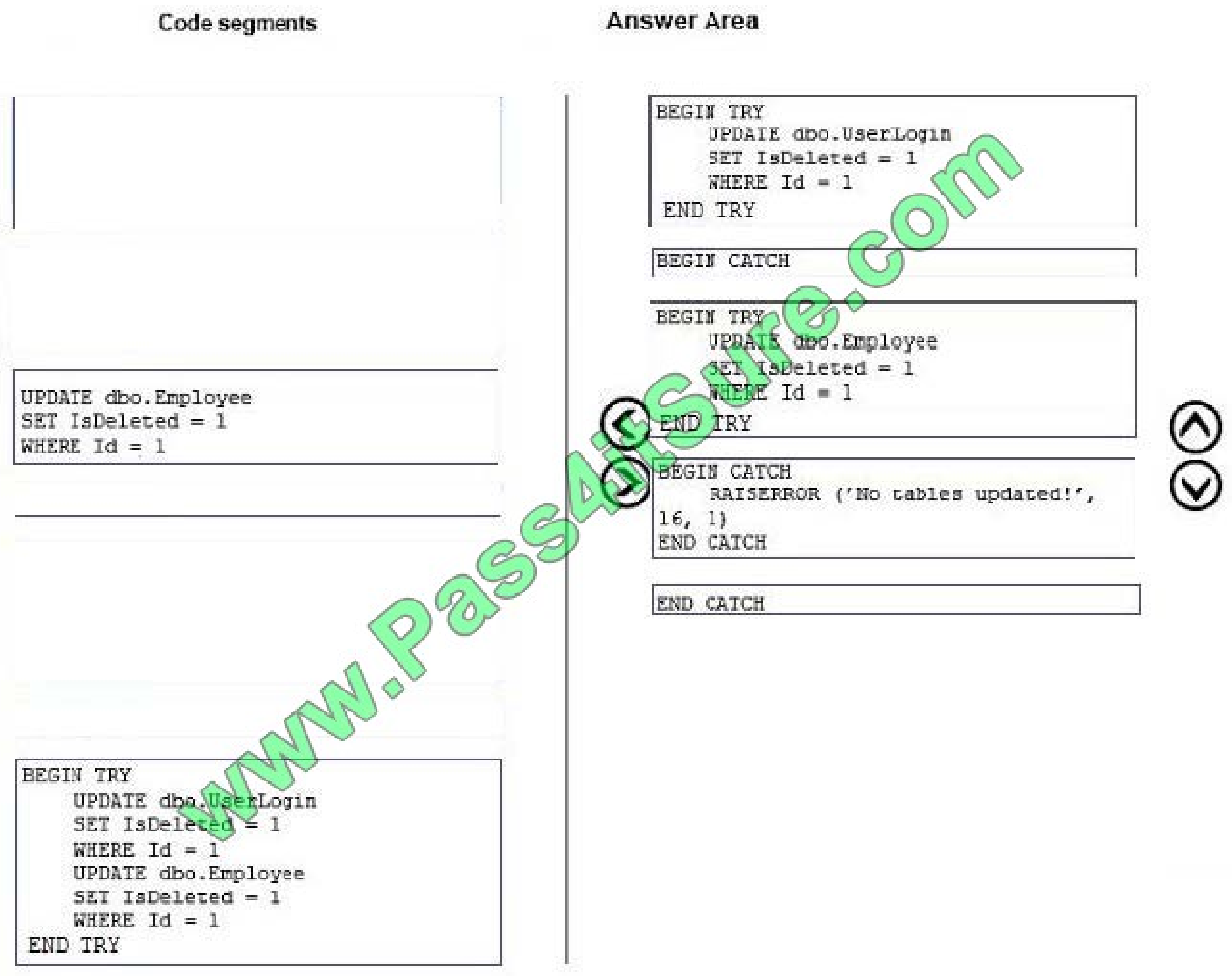
```
BEGIN CATCH
```

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

```
END CATCH
```

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

Correct Answer:



Code segments

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

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

Answer Area

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

```
BEGIN CATCH
```

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

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

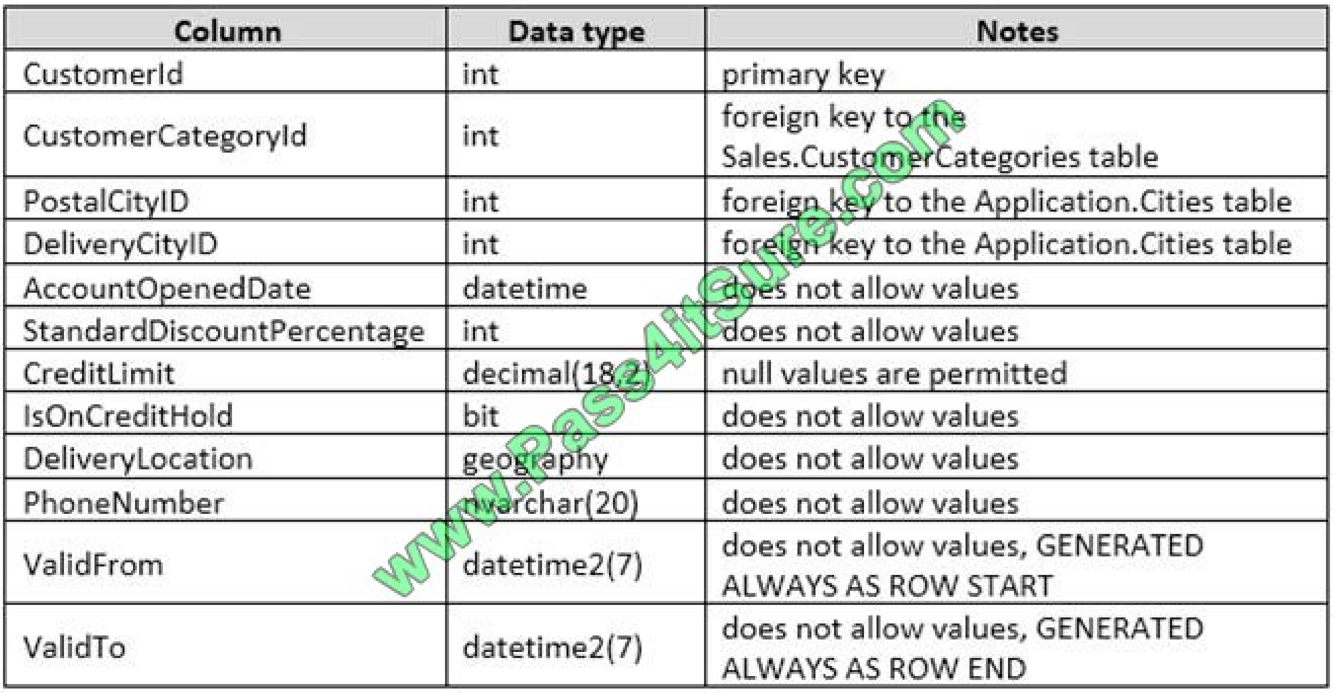
```
END CATCH
```

QUESTION 5

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.

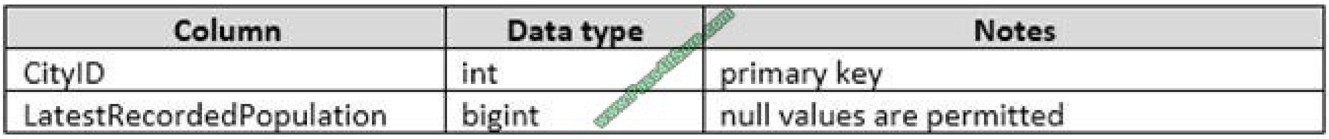
You have a database that tracks orders and deliveries for customers in North America. System versioning is enabled for all tables. The database contains the Sales.Customers, Application.Cities, and Sales.CustomerCategories tables.

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



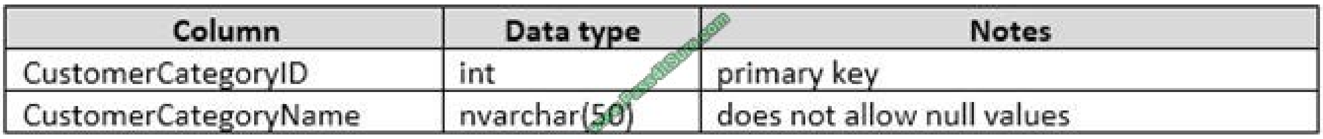
Column	Data type	Notes
CustomerId	int	primary key
CustomerCategoryId	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow values
StandardDiscountPercentage	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

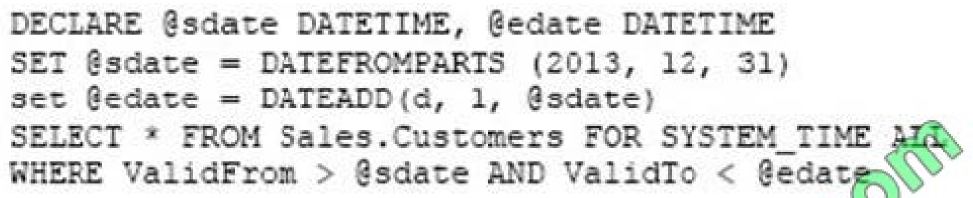
You discover an application bug that impacts customer data for records created on or after January 1, 2014. In order to fix the data impacted by the bug, application programmers require a report that contains customer data as it existed on

December 31, 2013.

You need to provide the query for the report.

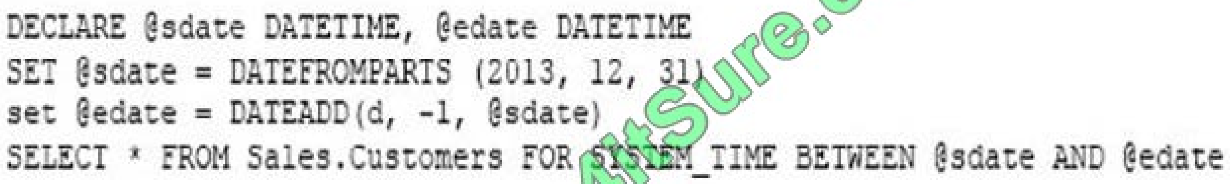
Which Transact-SQL statement should you use?

A.



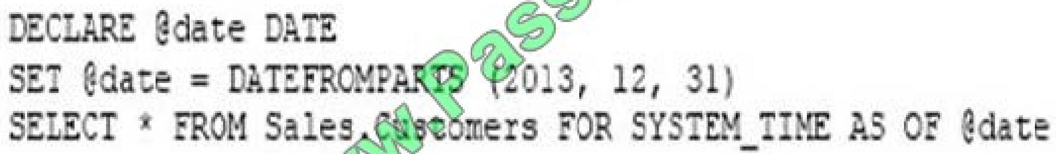
```
DECLARE @sdate DATETIME, @edate DATETIME
SET @sdate = DATEFROMPARTS (2013, 12, 31)
set @edate = DATEADD(d, 1, @sdate)
SELECT * FROM Sales.Customers FOR SYSTEM_TIME ALL
WHERE ValidFrom > @sdate AND ValidTo < @edate
```

B.



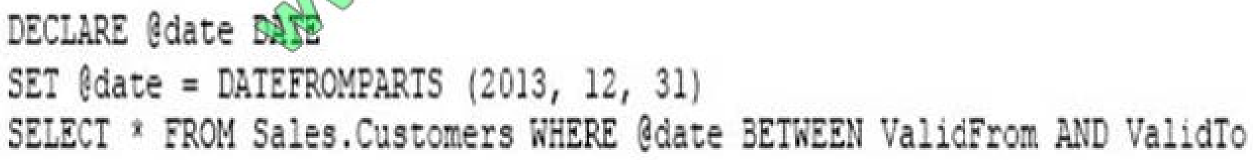
```
DECLARE @sdate DATETIME, @edate DATETIME
SET @sdate = DATEFROMPARTS (2013, 12, 31)
set @edate = DATEADD(d, -1, @sdate)
SELECT * FROM Sales.Customers FOR SYSTEM_TIME BETWEEN @sdate AND @edate
```

C.



```
DECLARE @date DATE
SET @date = DATEFROMPARTS (2013, 12, 31)
SELECT * FROM Sales.Customers FOR SYSTEM_TIME AS OF @date
```

D.



```
DECLARE @date DATE
SET @date = DATEFROMPARTS (2013, 12, 31)
SELECT * FROM Sales.Customers WHERE @date BETWEEN ValidFrom AND ValidTo
```

A. B. C. D.

Correct Answer: D

The datetime datetype defines a date that is combined with a time of day with fractional seconds that is based on a 24-hour clock. The DATEFROMPARTS function returns a date value for the specified year, month, and day.

Incorrect Answers:

A: ValidFrom should be less () than @edate.

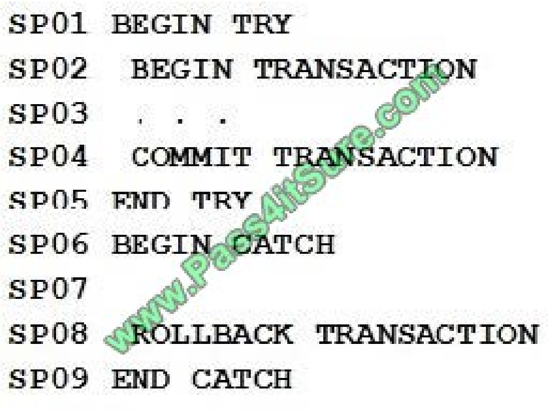
B: We should add a day with DATEADD, not subtract one day.

C: We cannot compare a date to an exact datetime.

References: https://msdn.microsoft.com/en-us/library/ms187819.aspx

QUESTION 6

You are performing a code review of stored procedures. Code at line SP03 fails to run (Line numbers are included for reference only.)



```
SP01 BEGIN TRY
SP02  BEGIN TRANSACTION
SP03  . . .
SP04  COMMIT TRANSACTION
SP05 END TRY
SP06 BEGIN CATCH
SP07
SP08  ROLLBACK TRANSACTION
SP09 END CATCH
```

You need to ensure that transactions are rolled back when an error occurs. Which Transact-SQL segment should you insert at line SP07?

A. If @@Error 0

B. If @@ TRANCOUNT = 0

C. If @@ TRANCOUNT > 0

D. If @@ Error = 0

Correct Answer: C

Using TRY...CATCH in a transaction

The following example shows how a TRY...CATCH block works inside a transaction. The statement inside the TRY block generates a constraint violation error.

BEGIN TRANSACTION;

BEGIN TRY

-- Generate a constraint violation error.

DELETE FROM Production.Product

WHERE ProductID = 980;

END TRY

BEGIN CATCH

SELECT

ERROR_NUMBER() AS ErrorNumber

,ERROR_SEVERITY() AS ErrorSeverity

,ERROR_STATE() AS ErrorState

,ERROR_PROCEDURE() AS ErrorProcedure

,ERROR_LINE() AS ErrorLine

,ERROR_MESSAGE() AS ErrorMessage;

IF @@TRANCOUNT > 0

ROLLBACK TRANSACTION;

END CATCH;

IF @@TRANCOUNT > 0

COMMIT TRANSACTION;

GO

References: https://docs.microsoft.com/en-us/sql/t-sql/language-elements/try-catch-transact-sql

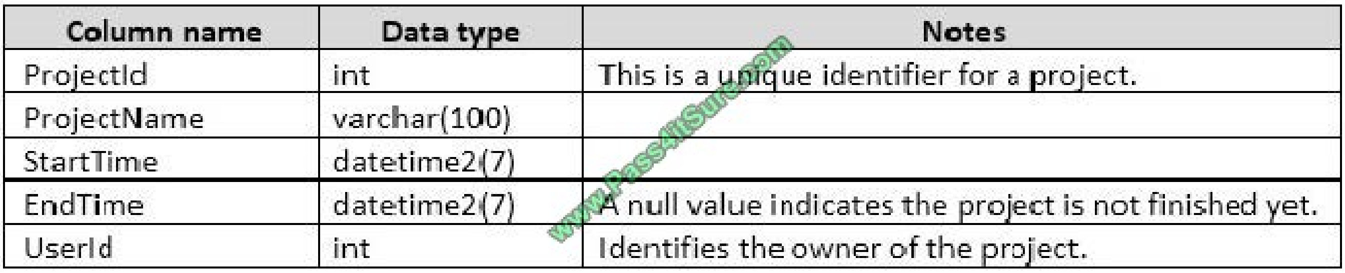
QUESTION 7

DRAG DROP

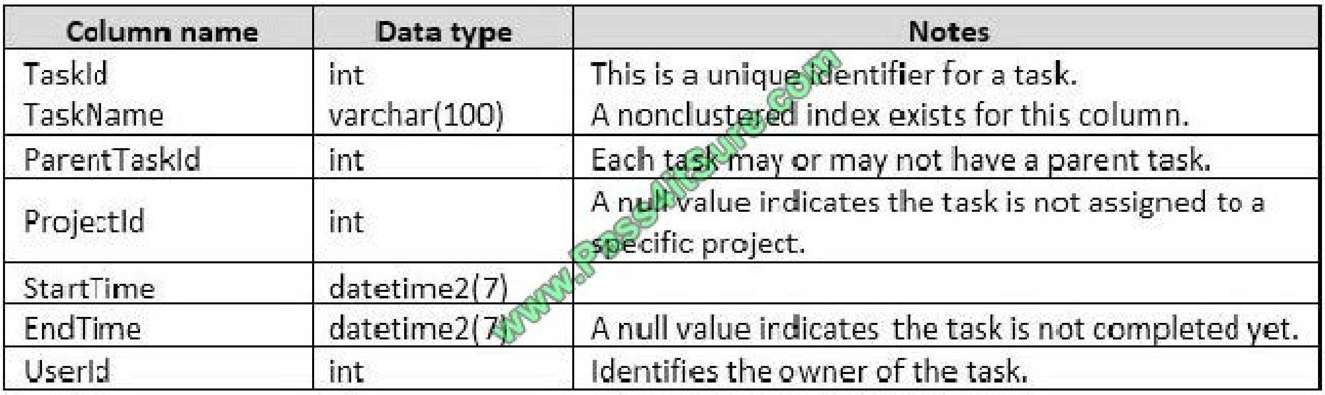
Note: This question is part of a series of questions that use the same scenario. For your convenience, the scenario is repeated in each question. Each question presents a different goal and answer choices, but the text of the scenario is

exactly the same in each question in this series.

You 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 TaskName	int varchar(100)	This is a unique identifier for a task. 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.

When running an operation, you updated a column named EndTime for several records in the Project table, but updates to the corresponding task records in the Task table failed.

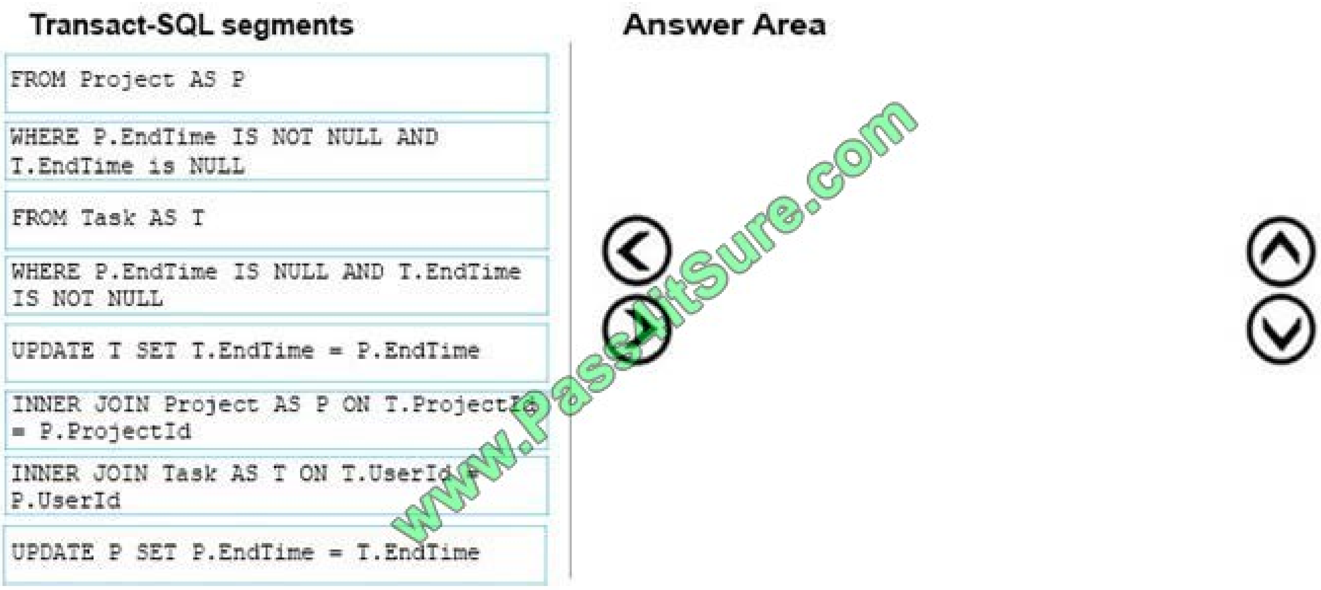
You need to synchronize the value of the EndTime column in the Task table with the value of the EndTime column in the project table. The solution must meet the following requirements:

If the EndTime column has a value, make no changes to the record.

If the value of the EndTime column is null and the corresponding project record is marked as completed, update the record with the project finish time.

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

Select and Place:



Transact-SQL segments

```
FROM Project AS P
```

```
WHERE P.EndTime IS NOT NULL AND
T.EndTime is NULL
```

```
FROM Task AS T
```

```
WHERE P.EndTime IS NULL AND T.EndTime
IS NOT NULL
```

```
UPDATE T SET T.EndTime = P.EndTime
```

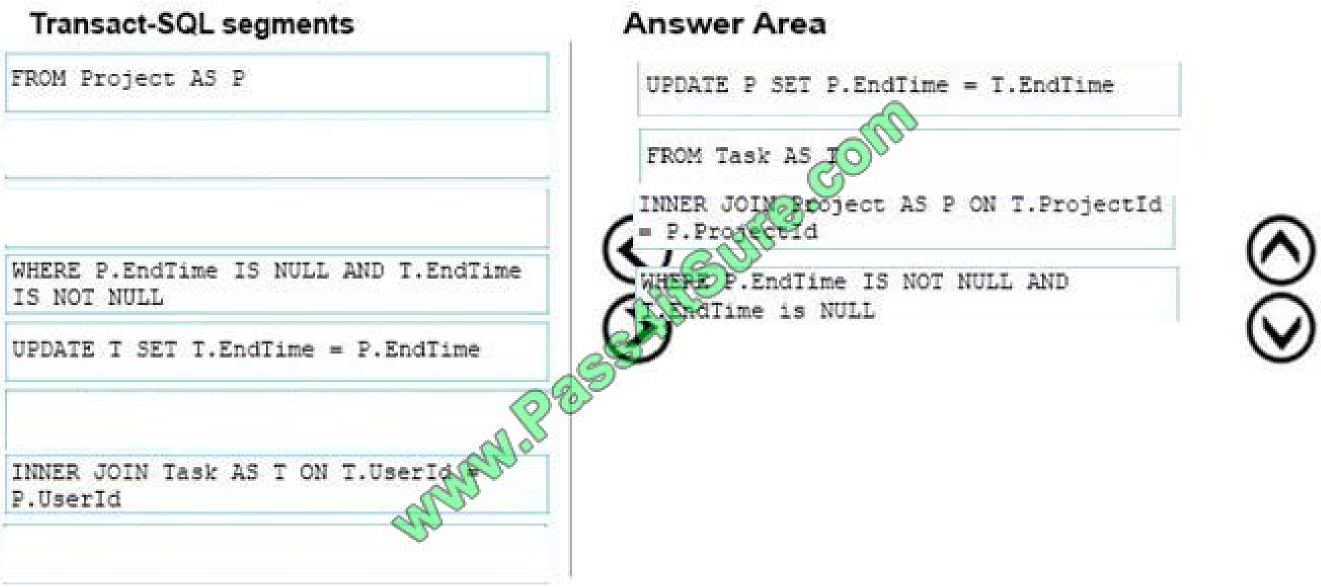
```
INNER JOIN Project AS P ON T.ProjectId
= P.ProjectId
```

```
INNER JOIN Task AS T ON T.UserId =
P.UserId
```

```
UPDATE P SET P.EndTime = T.EndTime
```

Answer Area

Correct Answer:



Transact-SQL segments

```
FROM Project AS P
```

```
WHERE P.EndTime IS NULL AND T.EndTime
IS NOT NULL
```

```
UPDATE T SET T.EndTime = P.EndTime
```

```
INNER JOIN Task AS T ON T.UserId =
P.UserId
```

Answer Area

```
UPDATE P SET P.EndTime = T.EndTime
```

```
FROM Task AS T
```

```
INNER JOIN Project AS P ON T.ProjectId
= P.ProjectId
```

```
WHERE P.EndTime IS NOT NULL AND
T.EndTime is NULL
```

Box 1: UPDATE T SET T.EndTime = P.EndTime

We are updating the EndTime column in the Task table.

Box 2: FROM Task AS T

Where are updating the task table.

Box 3:INNER JOIN Project AS P on T.ProjectID = P.ProjectID

We join with the Project table (on the ProjectID columnID column).

Box 4: WHERE P.EndTime is NOT NULL AND T.EndTime is NULL

We select the columns in the Task Table where the EndTime column in the Project table has a value (NOT NULL),but where it is NULL in the Task Table.

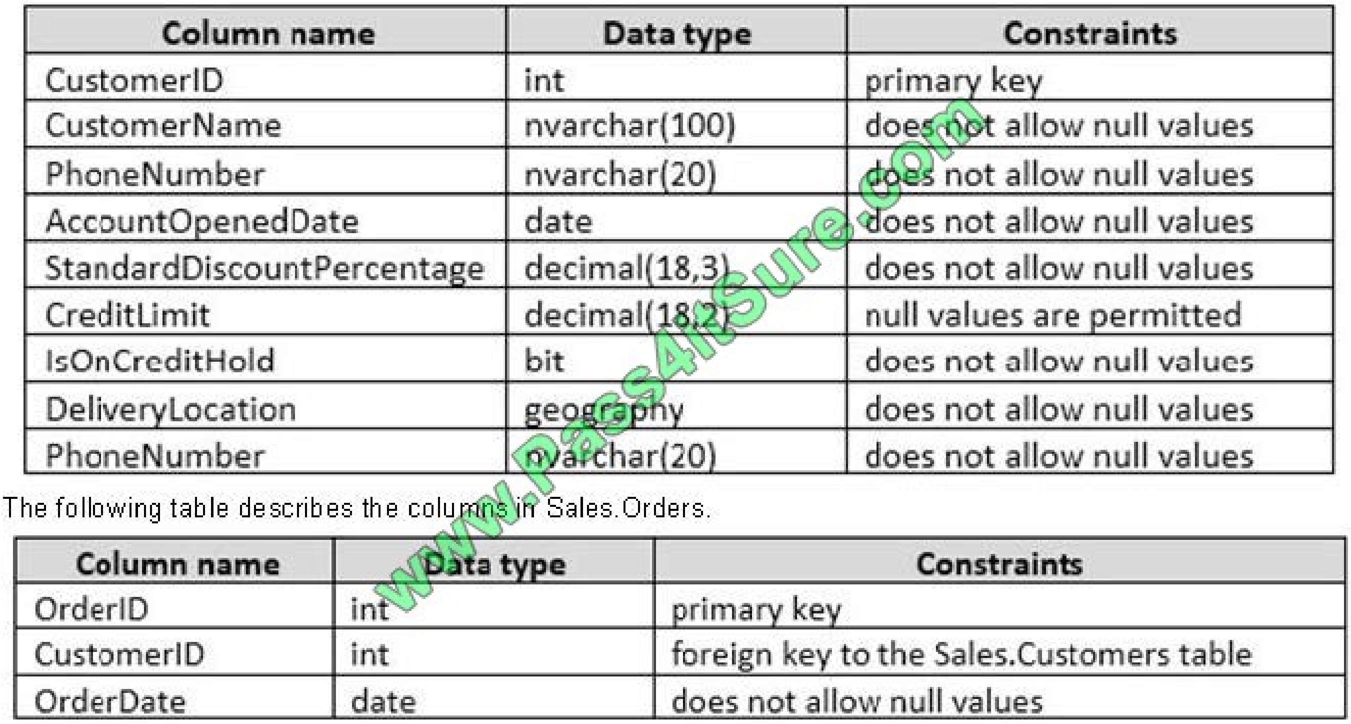
References: https://msdn.microsoft.com/en-us/library/ms177523.aspx

QUESTION 8

DRAG DROP

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



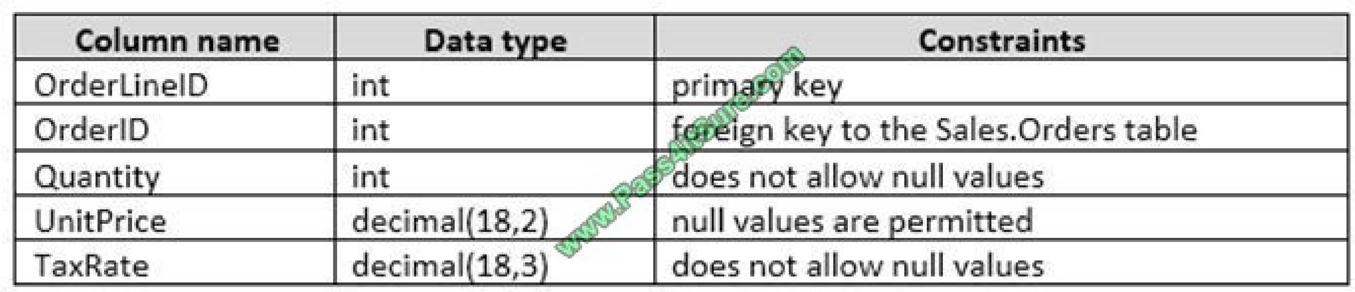
Column name	Data type	Constraints
CustomerID	int	primary key
CustomerName	nvarchar(100)	does not allow null values
PhoneNumber	nvarchar(20)	does not allow null values
AccountOpenedDate	date	does not allow null values
StandardDiscountPercentage	decimal(18,3)	does not allow null values
CreditLimit	decimal(18,2)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

The following table describes the columns in Sales.OrderLines.



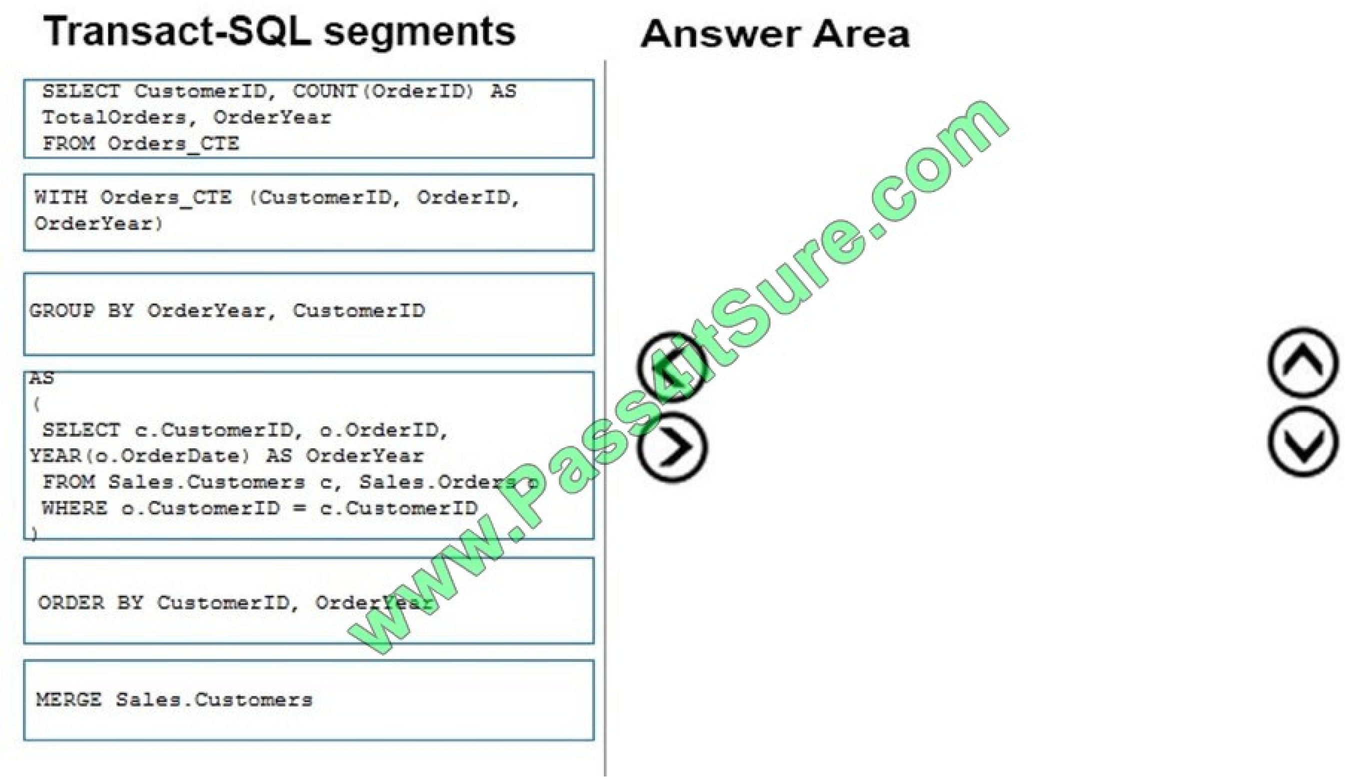
Column name	Data type	Constraints
OrderLineID	int	primary key
OrderID	int	foreign key to the Sales.Orders table
Quantity	int	does not allow null values
UnitPrice	decimal(18,2)	null values are permitted
TaxRate	decimal(18,3)	does not allow null values

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.

Select and Place:



```
SELECT CustomerID, COUNT(OrderID) AS
TotalOrders, OrderYear
FROM Orders_CTE
```

```
WITH Orders_CTE (CustomerID, OrderID,
OrderYear)
```

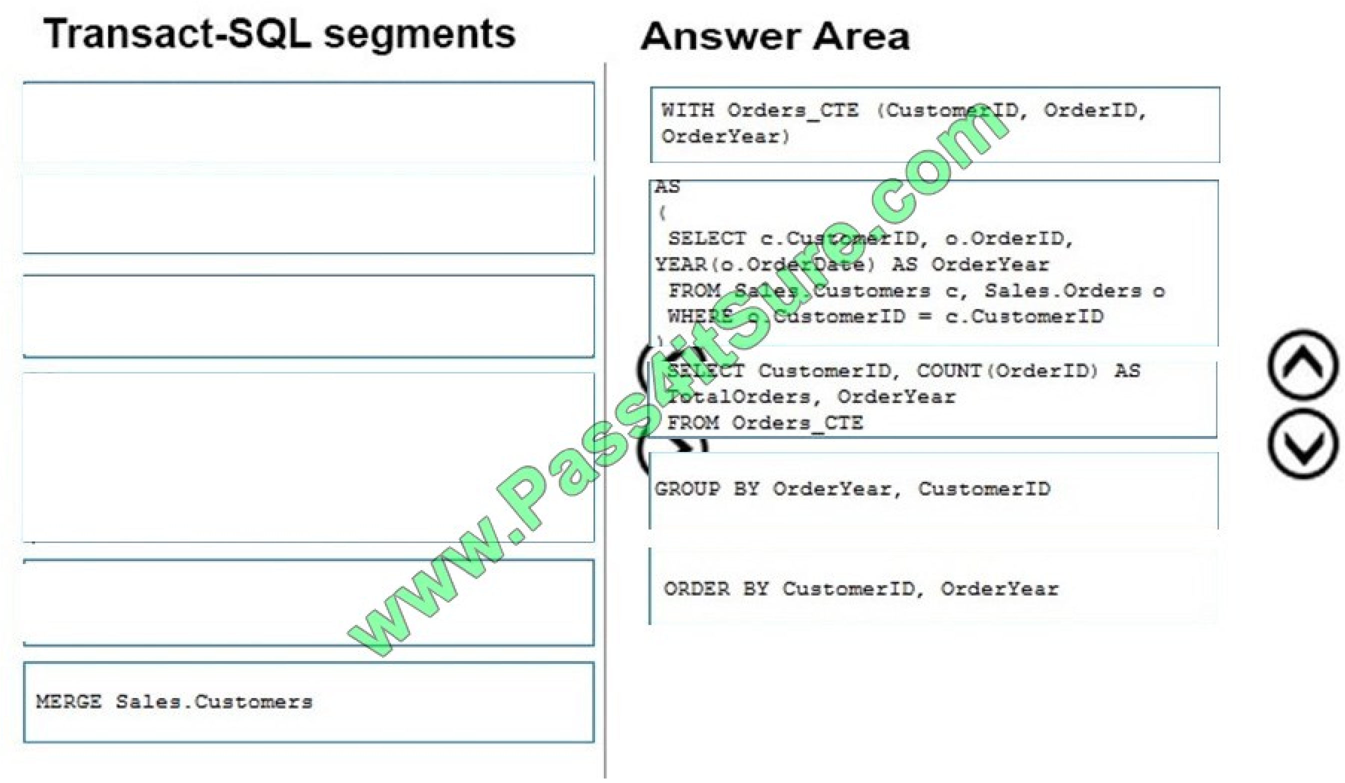
```
GROUP BY OrderYear, CustomerID
```

```
AS
(
 SELECT c.CustomerID, o.OrderID,
YEAR(o.OrderDate) AS OrderYear
 FROM Sales.Customers c, Sales.Orders o
 WHERE o.CustomerID = c.CustomerID
)
```

```
ORDER BY CustomerID, OrderYear
```

```
MERGE Sales.Customers
```

Correct Answer:



```
MERGE Sales.Customers
```

```
WITH Orders_CTE (CustomerID, OrderID,
OrderYear)
```

```
AS
(
 SELECT c.CustomerID, o.OrderID,
YEAR(o.OrderDate) AS OrderYear
 FROM Sales.Customers c, Sales.Orders o
 WHERE o.CustomerID = c.CustomerID
)
```

```
SELECT CustomerID, COUNT(OrderID) AS
TotalOrders, OrderYear
FROM Orders_CTE
```

```
GROUP BY OrderYear, CustomerID
```

```
ORDER BY CustomerID, OrderYear
```

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

QUESTION 9

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while

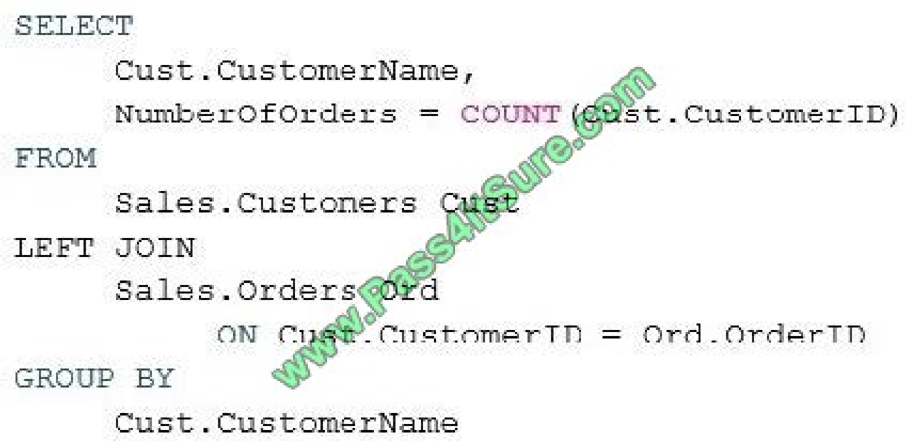
others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You 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

Correct Answer: A

QUESTION 10

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

Correct Answer: B

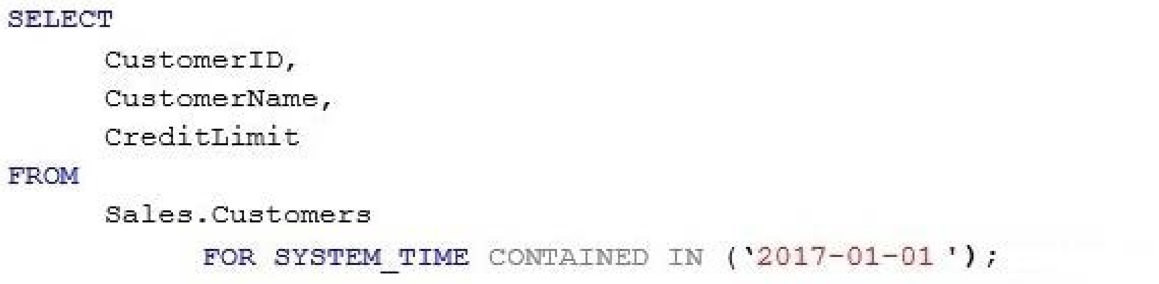
Using an Indexed View would allow you to keep your base data in properly normalized tables and maintain data-integrity while giving you the denormalized "view" of that data. References: http://stackoverflow.com/questions/4789091/updating-redundant-denormalized-data-automatically-in-sql-server

QUESTION 11

You have a database named DB1 that contains a temporal table named Sales.Customers.

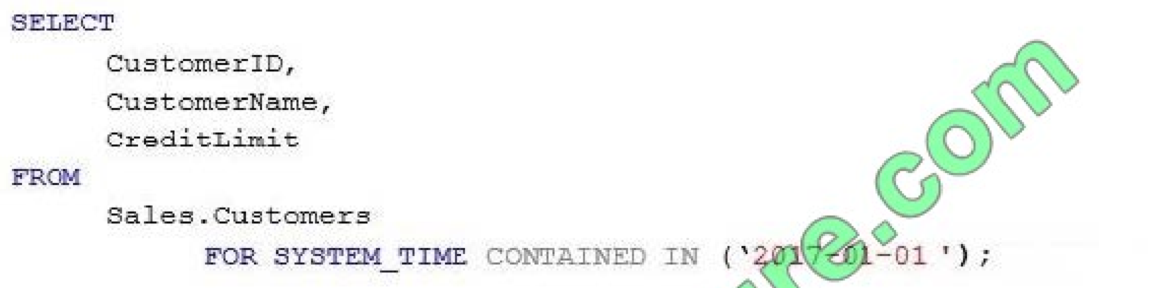
You need to create a query that returns the credit limit that was available to each customer in DB1 at the beginning of 2017.

Which query should you execute?



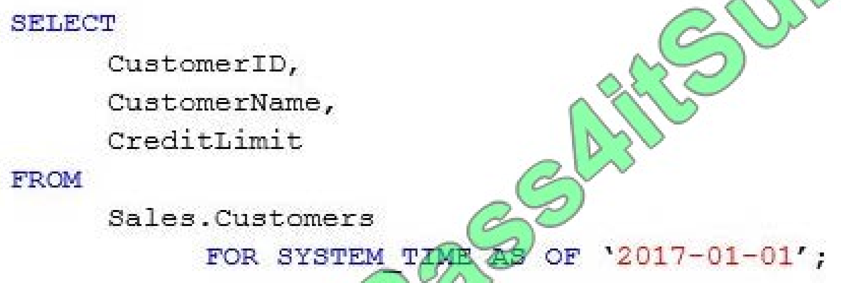
```
SELECT
        CustomerID,
        CustomerName,
        CreditLimit
FROM
        Sales.Customers
                FOR SYSTEM_TIME CONTAINED IN (‘2017-01-01 ');
```

A.



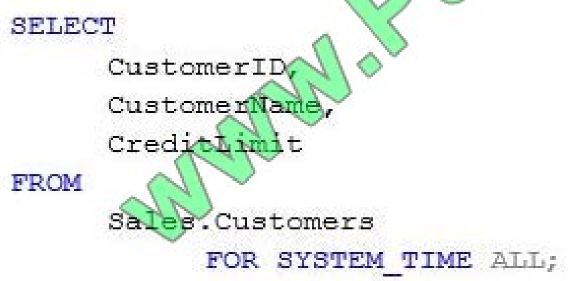
```
SELECT
        CustomerID,
        CustomerName,
        CreditLimit
FROM
        Sales.Customers
                FOR SYSTEM_TIME CONTAINED IN (‘2017-01-01 ');
```

B.



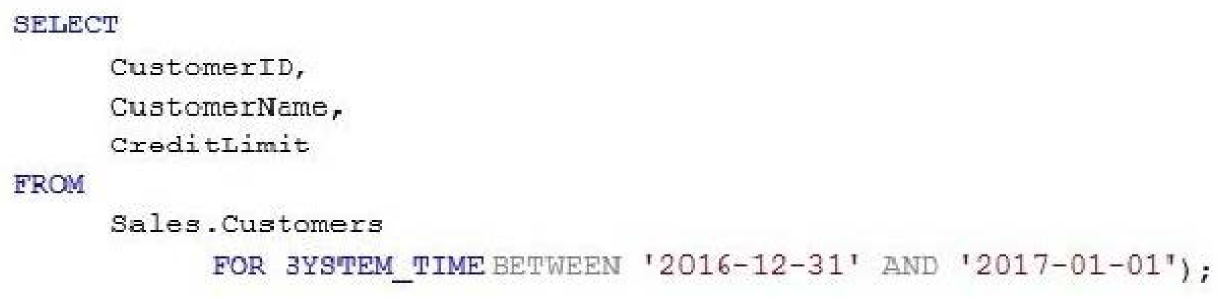
```
SELECT
        CustomerID,
        CustomerName,
        CreditLimit
FROM
        Sales.Customers
                FOR SYSTEM_TIME AS OF ‘2017-01-01’;
```

C.



```
SELECT
        CustomerID,
        CustomerName,
        CreditLimit
FROM
        Sales.Customers
                FOR SYSTEM_TIME ALL;
```

D.



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

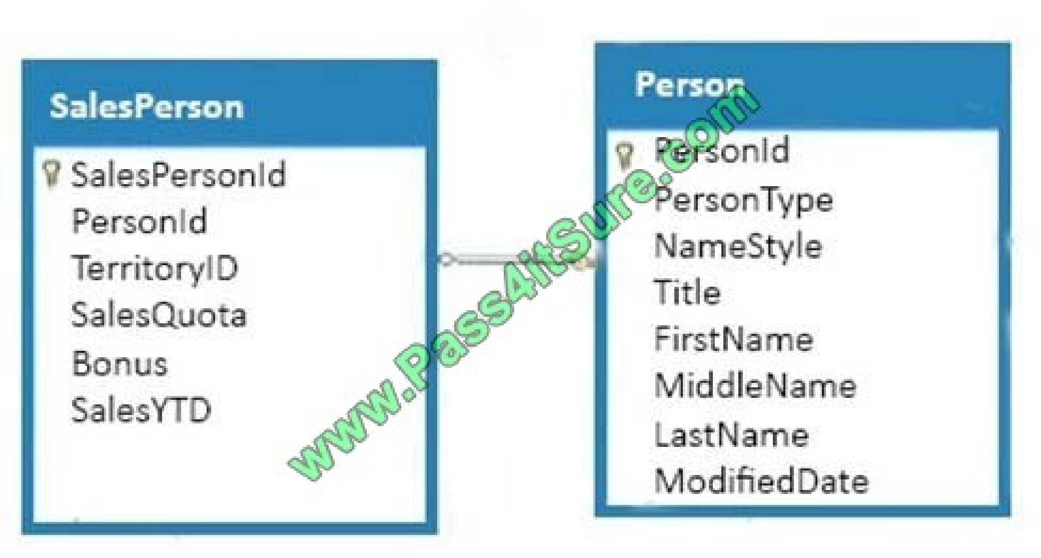
A. B. C. D.

Correct Answer: B

QUESTION 12

SIMULATION

You have a database that contains the following tables.



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

Return a column named Fullname that includes the salesperson FirstName, a space, and then LastName.

Include the current year-to-date sales for each salesperson.

Display only data for the three salespersons with the lowest year-to-year sales values.

Exclude salespersons that have no value for TerritoryID.

Construct the query using the following guidelines:

Use the first letter of a table name as the table alias.

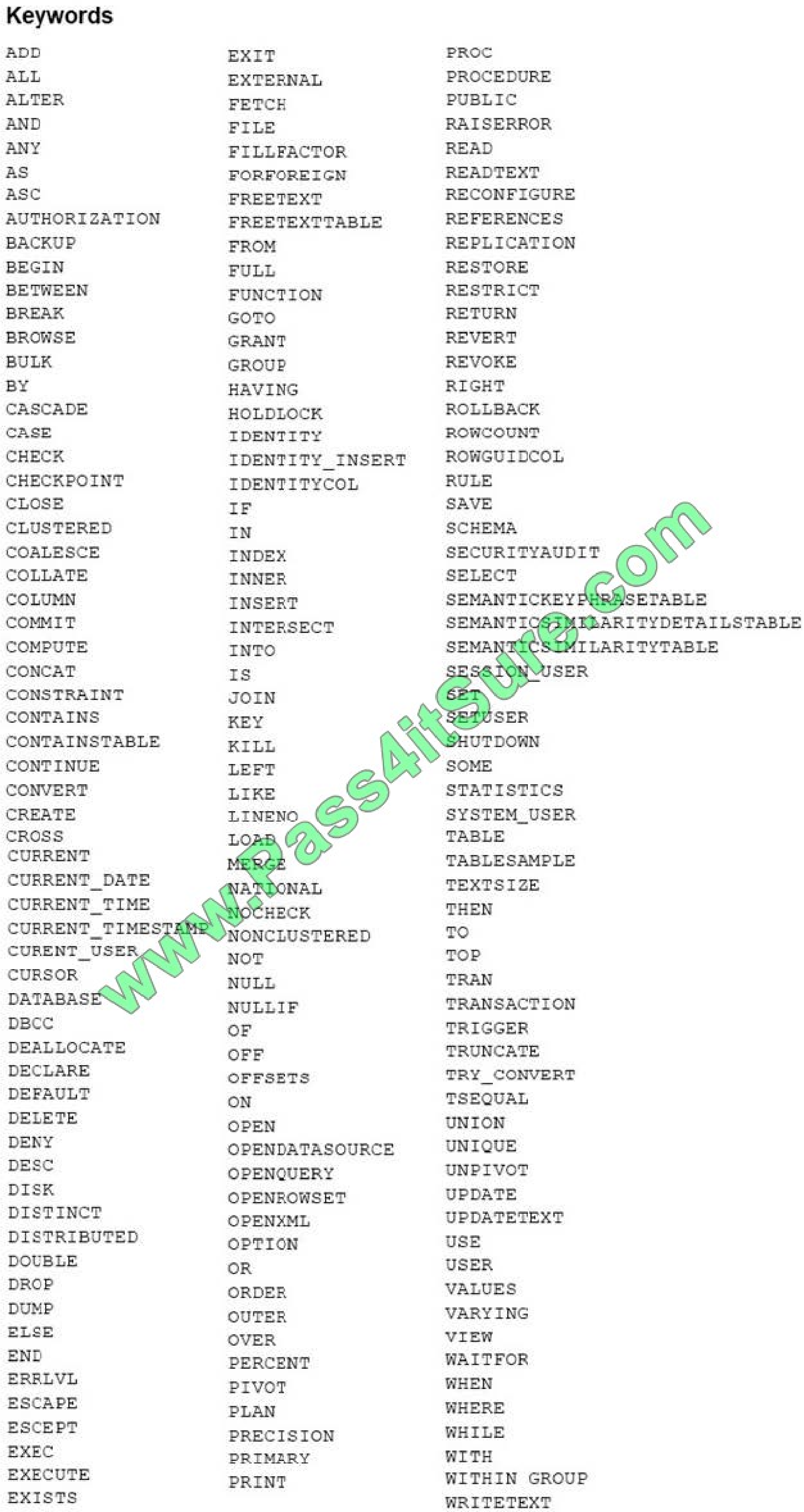
Use two-part column names.

Do not surround object names with square brackets.

Do not use implicit joins.

Use only single quotes for literal text.

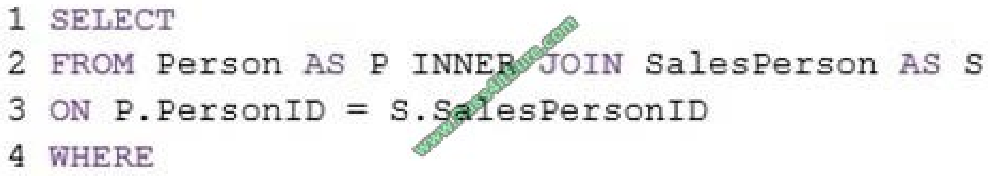
Use aliases only if required.



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
DEFAULT
DELETE
DENY
DESC
DISK
DISTINCT
DISTRIBUTED
DOUBLE
DROP
DUMP
ELSE
END
ERRLVL
ESCAPE
ESCEPT
EXEC
EXECUTE
EXISTS
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
ON
OPEN
OPENDATASOURCE
OPENQUERY
OPENROWSET
OPENXML
OPTION
OR
ORDER
OUTER
OVER
PERCENT
PIVOT
PLAN
PRECISION
PRIMARY
PRINT
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
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.



```
SELECT
FROM Person AS P INNER JOIN SalesPerson AS S
ON P.PersonID = S.SalesPersonID
WHERE
```

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

A. Check the answer in explanation.

Correct Answer: A

QUESTION 13

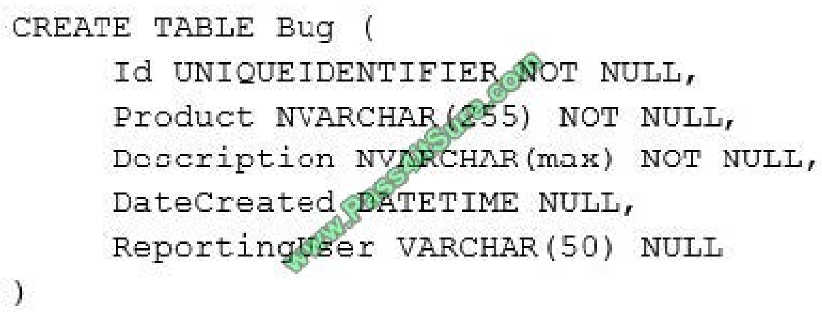
You develop and deploy a project management application. The application uses a Microsoft SQL Server database to store data. You are developing a software bug tracking add-on for the application. The add-on must meet the following requirements:

Allow case sensitive searches for product.

Filter search results based on exact text in the description.

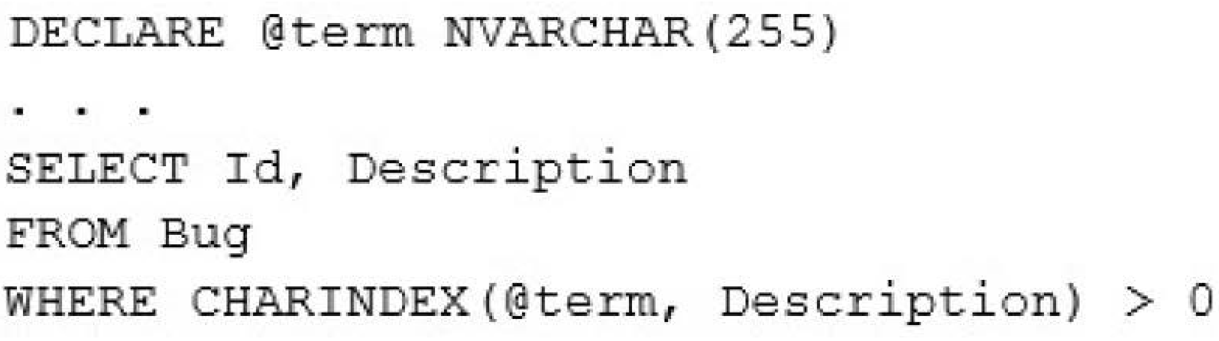
Support multibyte Unicode characters.

You run the following Transact-SQL statement:



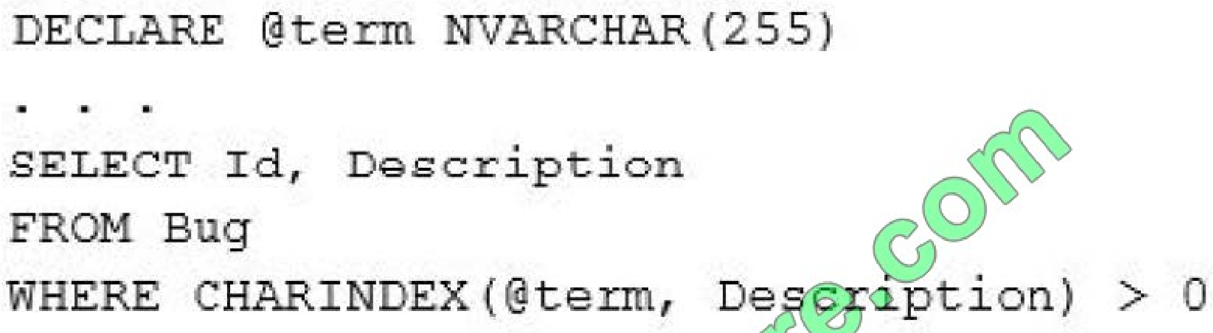
```
CREATE TABLE Bug (
      Id UNIQUEIDENTIFIER NOT NULL,
      Product NVARCHAR(255) NOT NULL,
      Description NVARCHAR(max) NOT NULL,
      DateCreated DATETIME NULL,
      ReportingUser VARCHAR(50) NULL
)
```

You need to ensure that users can perform searches of descriptions. Which Transact-SQL statement should you run?



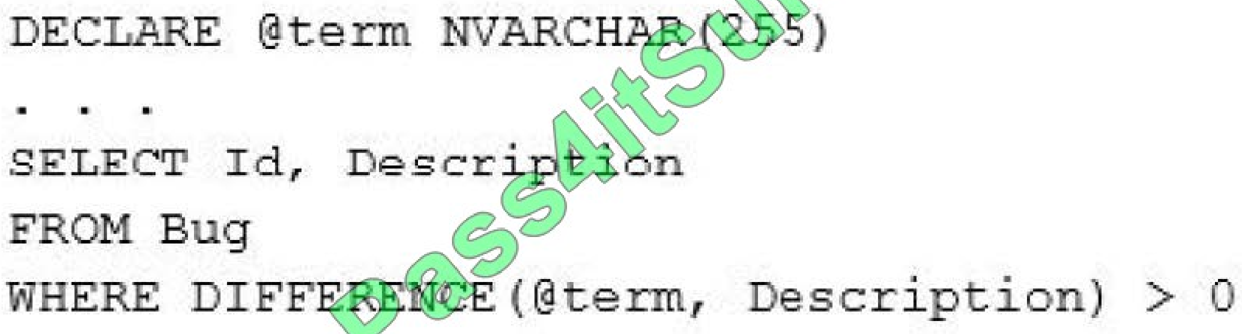
```
DECLARE @term NVARCHAR(255)
. . .
SELECT Id, Description
FROM Bug
WHERE CHARINDEX(@term, Description) > 0
```

A.



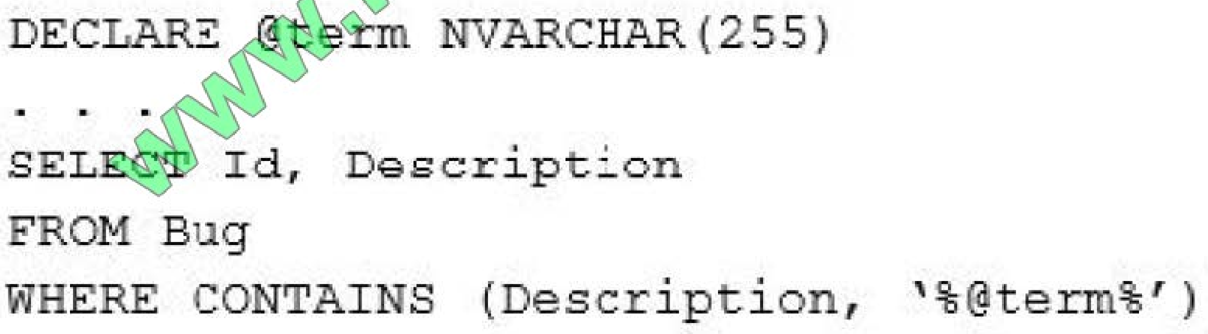
```
DECLARE @term NVARCHAR(255)
. . .
SELECT Id, Description
FROM Bug
WHERE CHARINDEX(@term, Description) > 0
```

B.



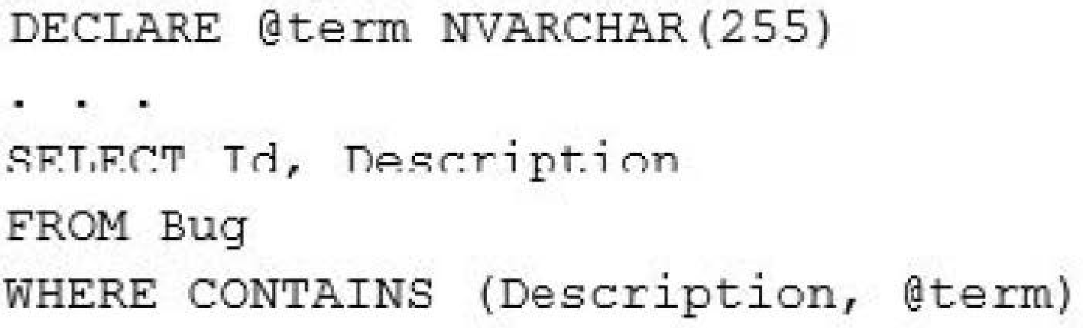
```
DECLARE @term NVARCHAR(255)
. . .
SELECT Id, Description
FROM Bug
WHERE DIFFERENCE(@term, Description) > 0
```

C.



```
DECLARE @term NVARCHAR(255)
. . .
SELECT Id, Description
FROM Bug
WHERE CONTAINS (Description, '%@term%')
```

D.



```
DECLARE @term NVARCHAR(255)
. . .
SELECT Id, Description
FROM Bug
WHERE CONTAINS (Description, @term)
```

A. B. C. D.

Correct Answer: D

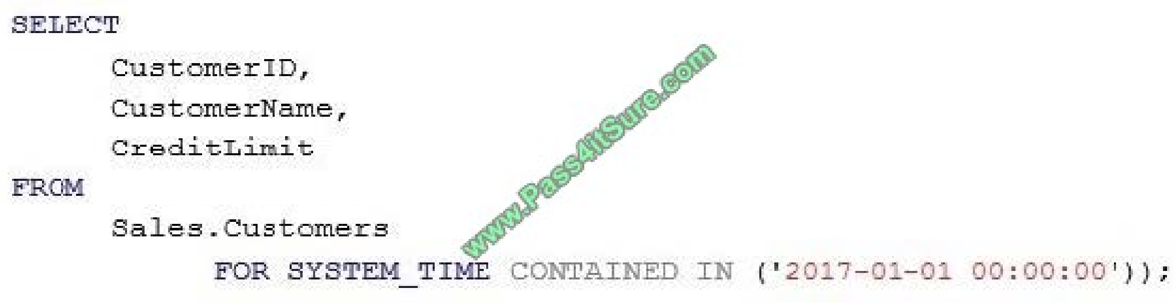
References: https://docs.microsoft.com/en-us/sql/t-sql/queries/contains-transact-sql?view=sql-server-2017

QUESTION 14

You have a database named DB1 that contains a temporal table named Sales.Customers.

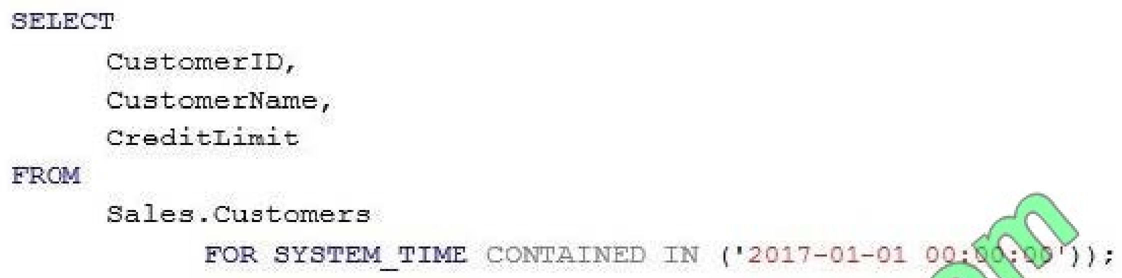
You need to create a query that returns the credit limit that was available to each customer in DB1 at the beginning of 2017.

Which query should you execute?



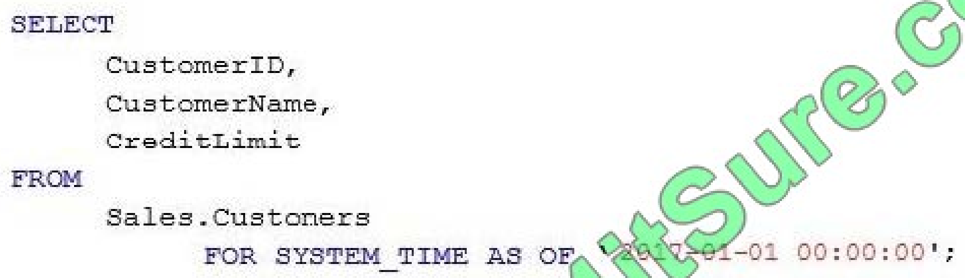
```
SELECT
      CustomerID,
      CustomerName,
      CreditLimit
FROM
      Sales.Customers
            FOR SYSTEM_TIME CONTAINED IN ('2017-01-01 00:00:00'));
```

A.



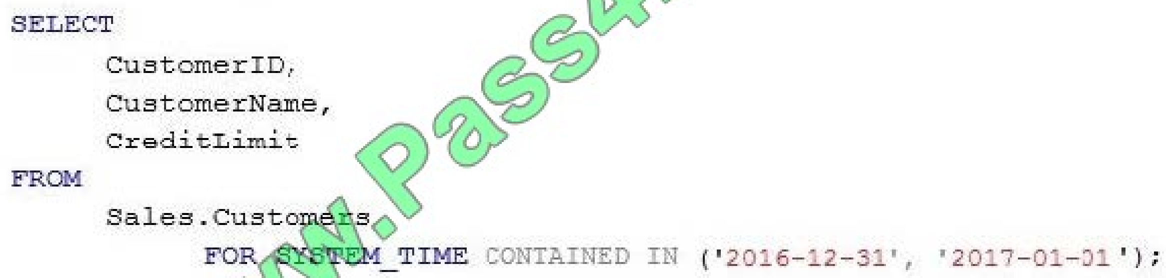
```
SELECT
        CustomerID,
        CustomerName,
        CreditLimit
FROM
        Sales.Customers
                FOR SYSTEM_TIME CONTAINED IN ('2017-01-01 00:00:00'));
```

B.



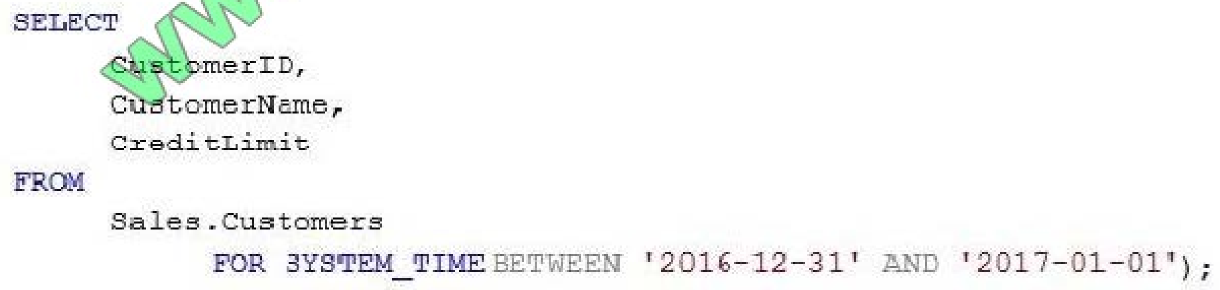
```
SELECT
        CustomerID,
        CustomerName,
        CreditLimit
FROM
        Sales.Custoners
                FOR SYSTEM_TIME AS OF '2017-01-01 00:00:00';
```

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 3YSTEM_TIME BETWEEN '2016-12-31' AND '2017-01-01');
```

A. B. C. D.

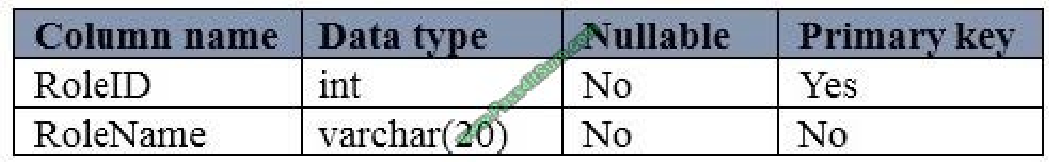
Correct Answer: B

QUESTION 15

HOTSPOT

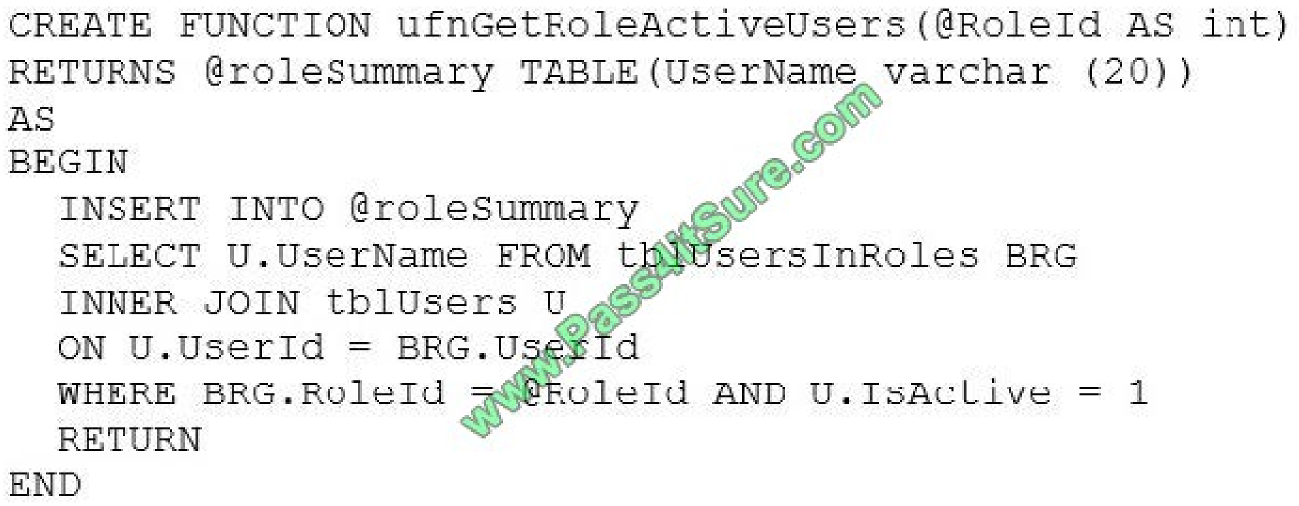
You have a database that contains the following tables: tblRoles, tblUsers, and tblUsersInRoles.

The table tblRoles is defined as follows.



Column name	Data type	Nullable	Primary key
RoleID	int	No	Yes
RoleName	varchar(20)	No	No

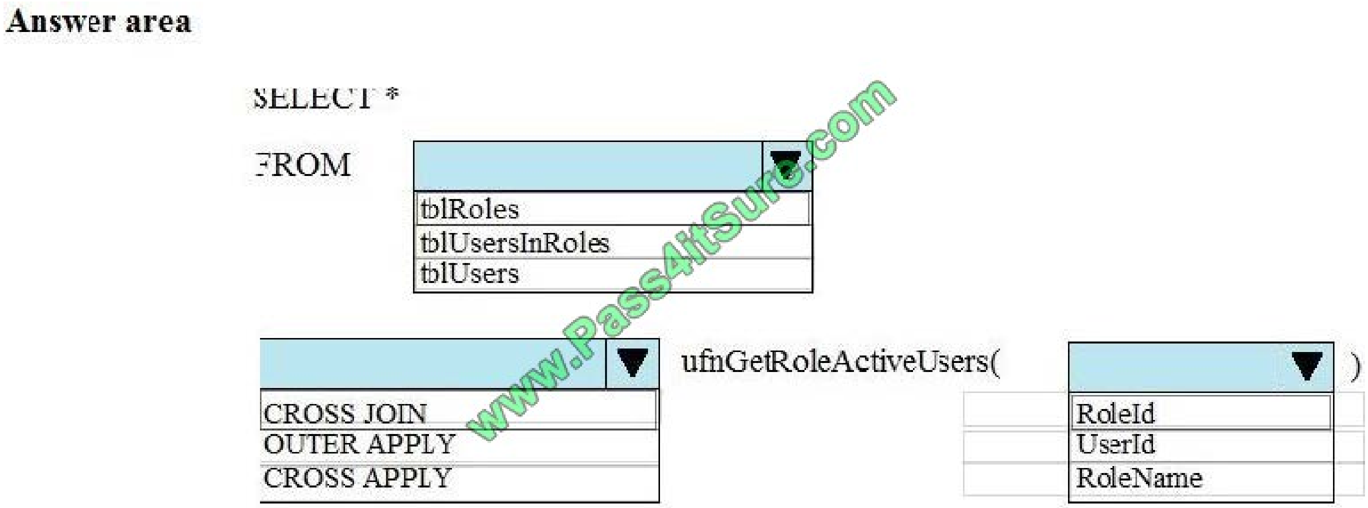
You have a function named ufnGetRoleActiveUsers that was created by running the following Transact-SQL statement:



```
CREATE FUNCTION ufnGetRoleActiveUsers(@RoleId AS int)
RETURNS @roleSummary TABLE(UserName varchar (20))
AS
BEGIN
   INSERT INTO @roleSummary
   SELECT U.UserName FROM tblUsersInRoles BRG
   INNER JOIN tblUsers U
   ON U.UserId = BRG.UserId
   WHERE BRG.RoleId = @RoleId AND U.IsActive = 1
   RETURN
END
```

You need to list all roles and their corresponding active users. The query must return the RoleId, RoleName, and UserName columns. If a role has no active users, a NULL value should be returned as the UserName for that role. How should you complete the Transact-SQL statement? To answer, select the appropriate Transact-SQL segments in the answer area.

Hot Area:

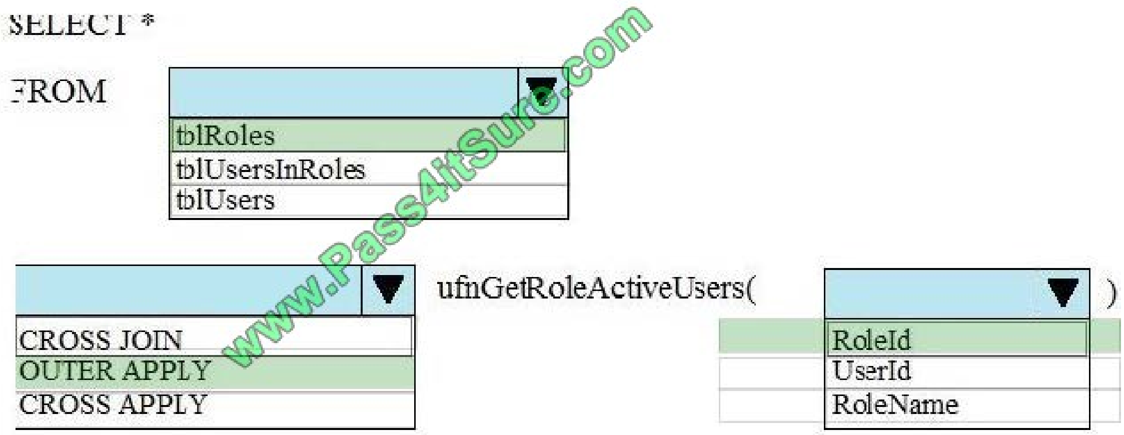


Answer area

```
SELECT *
FROM [ ▼ ]
      tblRoles
      tblUsersInRoles
      tblUsers
[ ▼ ] ufnGetRoleActiveUsers( [ ▼ ] )
CROSS JOIN                      RoleId
OUTER APPLY                     UserId
CROSS APPLY                     RoleName
```

Correct Answer:

Answer area

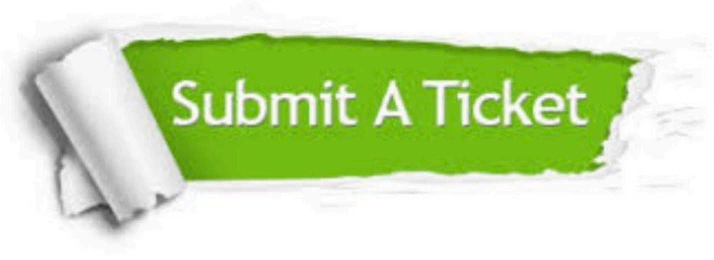


SELECT *

FROM
[✓ tblRoles]
tblUsersInRoles
tblUsers

CROSS JOIN
[✓ OUTER APPLY]
CROSS APPLY

ufnGetRoleActiveUsers(
[✓ RoleId]
UserId
RoleName
)


Submit A Ticket

100%