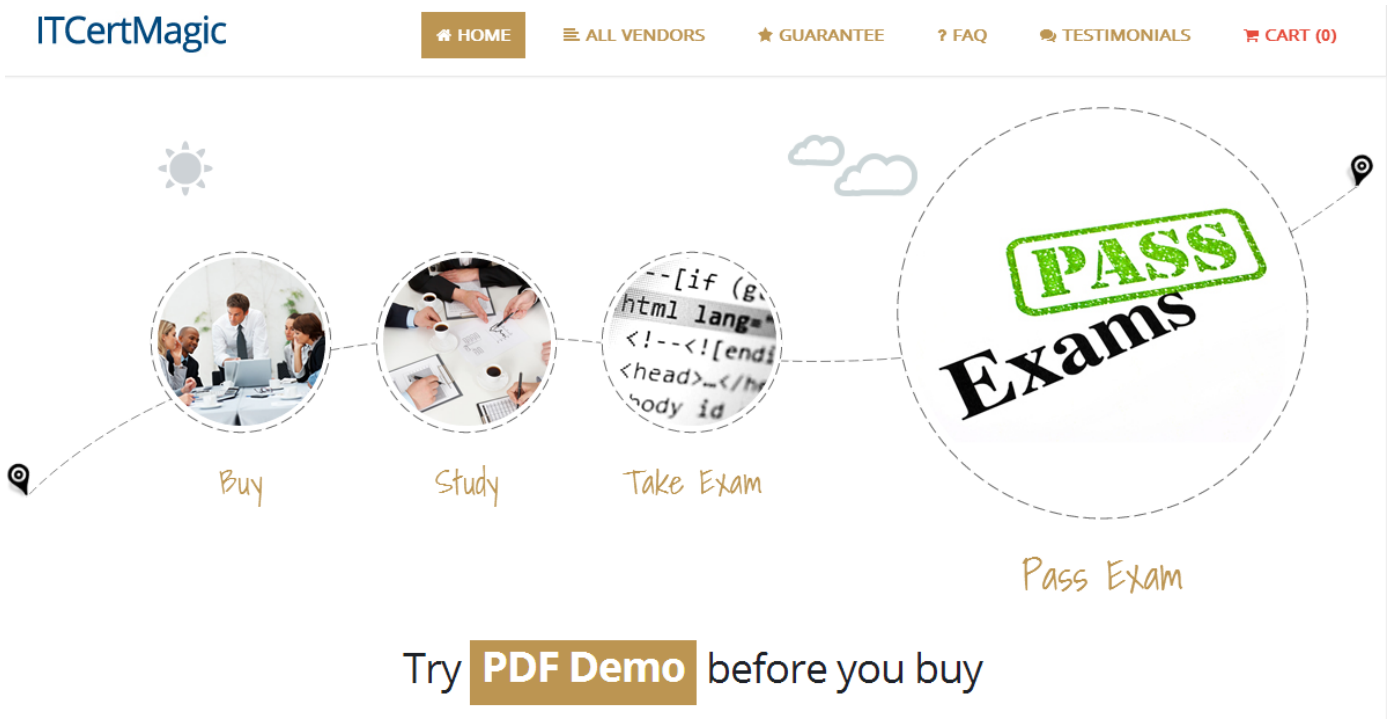
ITCertMagic



http://www.itcertmagic.com/

Pass-Guaranteed Certification Exam Questions | Exam Dumps - ITCertMagic

Exam : **70-761J**

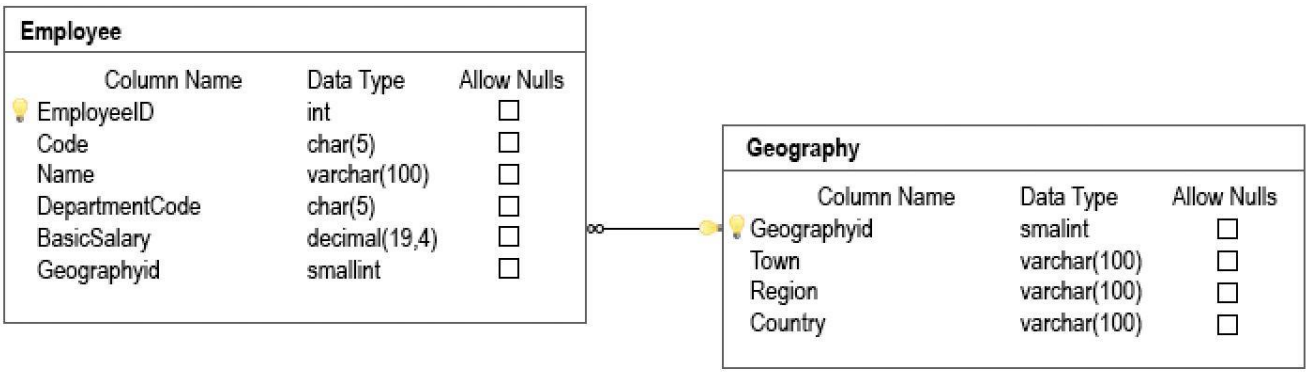
Title : Querying Data with Transact-SQL (70-761日本語版)

Vendor : Microsoft

Version : DEMO

QUESTION NO: 1
注：この質問は、同じまたは類似の回答の選択肢を使用する一連の質問の一部です。
回答の選択肢は、シリーズの複数の質問に対して正しいかもしれません。
各質問は、このシリーズの他の質問とは独立しています。
質問で提供される情報と詳細は、その質問にのみ適用されます。
複数のプロセスは、Salesという名前のテーブルのデータを使用し、組織内の他のデータベースに配置します。
一部のプロセスでは、Salesテーブルのデータ型を完全に知りません。これにより、データ型変換エラーが発生します。
エラーをスローするのではなく、データの変換に失敗し、NULL値を返す方法が必要です。
あなたは何を実装すべきですか？
A. aストアドプロシージャ
B. TRY_CONVERT関数
C. COALESCE関数
D. スカラー関数
E. テーブル値関数
F. ビュー
G. TRY_PARSE関数
H. ISNULL関数
Answer: B
Explanation
TRY_CONVERT returns a value cast to the specified data type if the cast succeeds; otherwise, returns null.
References: https://docs.microsoft.com/en-us/sql/t-sql/functions/try-convert-transact-sql

QUESTION NO: 2
次の図に示すように、2つのテーブルがあります。



次のクエリを分析する必要があります。（行番号は参照用にのみ含まれています。）

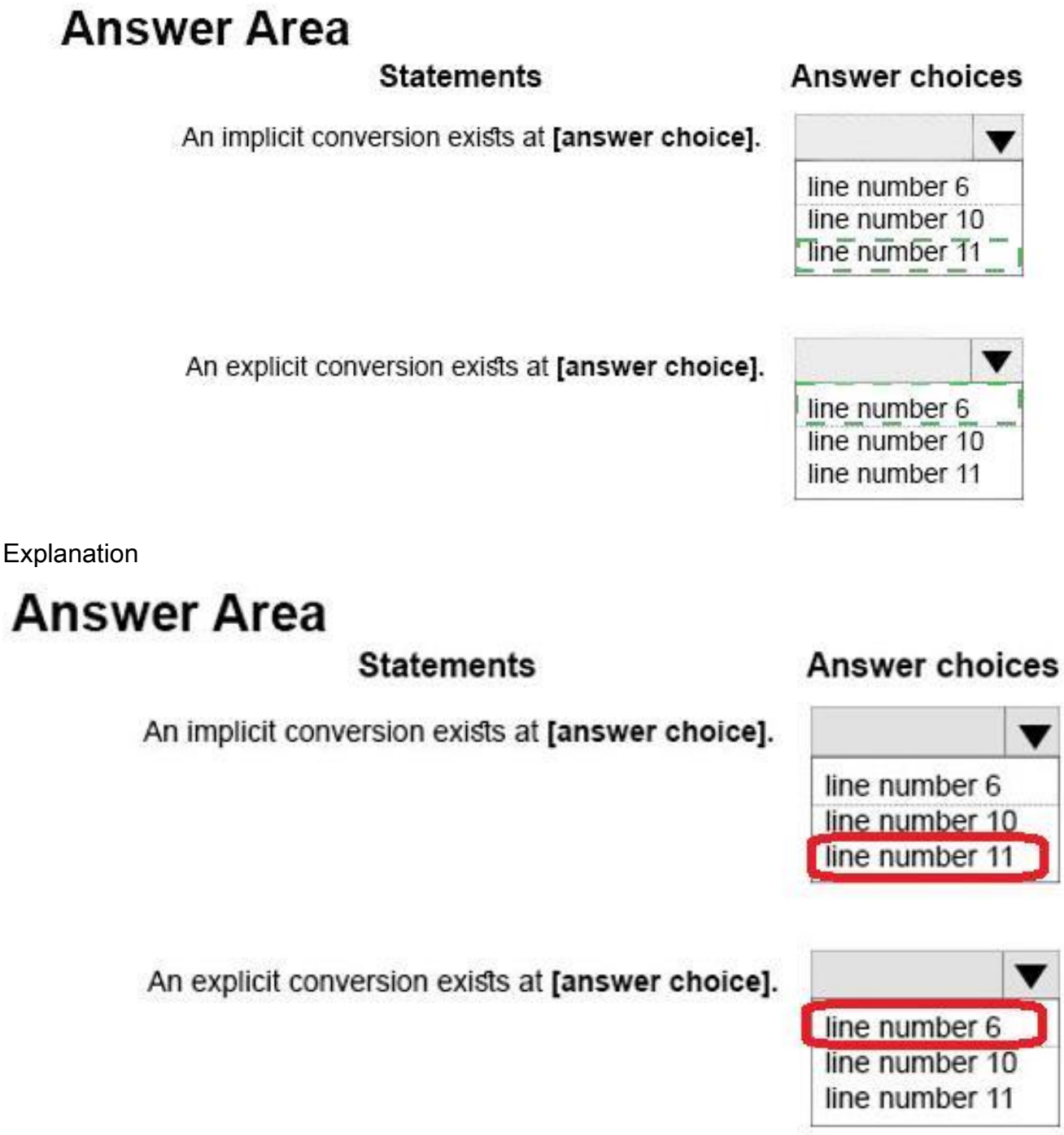
```
01  DECLARE @DepartmentCode nchar(5) = N'DEP01'
02  DECLARE @RoundedUpSalary int
03  DECLARE @EmployeeName nvarchar(100)
04  SELECT
05      Name,
06      CONVERT(int, Code) EmployeeCode,
07      BasicSalary
08  FROM dbo.Employee e
09  INNER JOIN dbo.Geography g
10  ON e.GeographyId = g.GeographyId
11  WHERE DepartmentCode = @DepartmentCode
```

ドロップダウンメニューを使用して、グラフィックに表示されている情報に基づいて各ステートメントを完成させる回答の選択肢を選択します。
注：それぞれ正しい選択は1ポイントの価値があります。

Answer Area

Statements	Answer choices
An implicit conversion exists at **[answer choice]**.	line number 6 / line number 10 / line number 11
An explicit conversion exists at **[answer choice]**.	line number 6 / line number 10 / line number 11

Answer:



Explanation

To compare char(5) and nchar(5) an implicit conversion has to take place.
Explicit conversions use the CAST or CONVERT functions, as in line number 6.
References:
https://docs.microsoft.com/en-us/sql/t-sql/data-types/data-type-conversion-database-engine#implicit-and-explicit

QUESTION NO: 3
次のTransact-
SQLステートメントを実行して、オンラインセールスアプリケーションのセールス情報を格納するためのテーブルを作成します。

```
CREATE TABLE Sales(
  SalesOrderID INT NOT NULL,
  OrderDate DATETIME NOT NULL,
  Total MONEY NULL
)
```

各四半期および年の売上を要約した履歴レポートがあります。
レポートのデータを生成したクエリは利用できなくなりました。
代表レポートには、以下のデータが含まれています。

Total	Quarter	Year	G1	G2
$8,771,886.36	1	2013	0	0
$14,373,277.48	1	2014	0	0
$23,145,163.83	1	NULL	0	1
$12,225,061.38	2	2013	0	0
$8,046,220.84	2	2014	0	0
$20,271,282.22	2	NULL	0	1
$14,339,319.19	3	2013	0	0
$14,339,319.19	3	NULL	0	1
$13,629,621.04	4	2013	0	0
$13,629,621.04	4	NULL	0	1
$71,385,386.28	NULL	NULL	1	1

レポートのクエリを再作成する必要があります。
Transact-SQLステートメントをどのように完成させるべきですか？
回答するには、回答領域で適切なTransact-SQLセグメントを選択します。
注：それぞれ正しい選択は1ポイントの価値があります。

```
SELECT
  SUM(Total) AS Total,
  DATEPART(quarter, OrderDate)AS Quarter,
  YEAR(OrderDate) AS Year,
```

[▼] AS G1,

- DATEPART(quarter, OrderDate)% 1
- MAX(DATEPART(quarter, OrderDate))
- GROUPING(DATEPART(quarter, OrderDate))
- CASE WHEN DATEPART(quarter, OrderDate)=NULL THEN 1 ELSE 0 END

[▼] AS G2

- GROUPING(YEAR(OrderDate))
- DATEPART(YEAR, OrderDate)%1
- MAX(DATEPART(YEAR, OrderDate))
- CASE WHEN DATEPART(YEAR, OrderDate)=NULL THEN 1 ELSE 0 END

```
FROM Sales
GROUP BY
```

[▼]

- DATEPART(quarter, OrderDate)
- DATEPART(quarter, OrderDate) WITH ROLLUP
- DATEPART(quarter, OrderDate), YEAR(OrderDate)
- DATEPART(quarter, OrderDate), YEAR(OrderDate) WITH ROLLUP

Answer:

```
SELECT
  SUM(Total) AS Total,
  DATEPART(quarter, OrderDate)AS Quarter,
  YEAR(OrderDate) AS Year,
```

[▼] AS G1,

- DATEPART(quarter, OrderDate)% 1
- MAX(DATEPART(quarter, OrderDate))
- GROUPING(DATEPART(quarter, OrderDate))
- CASE WHEN DATEPART(quarter, OrderDate)=NULL THEN 1 ELSE 0 END

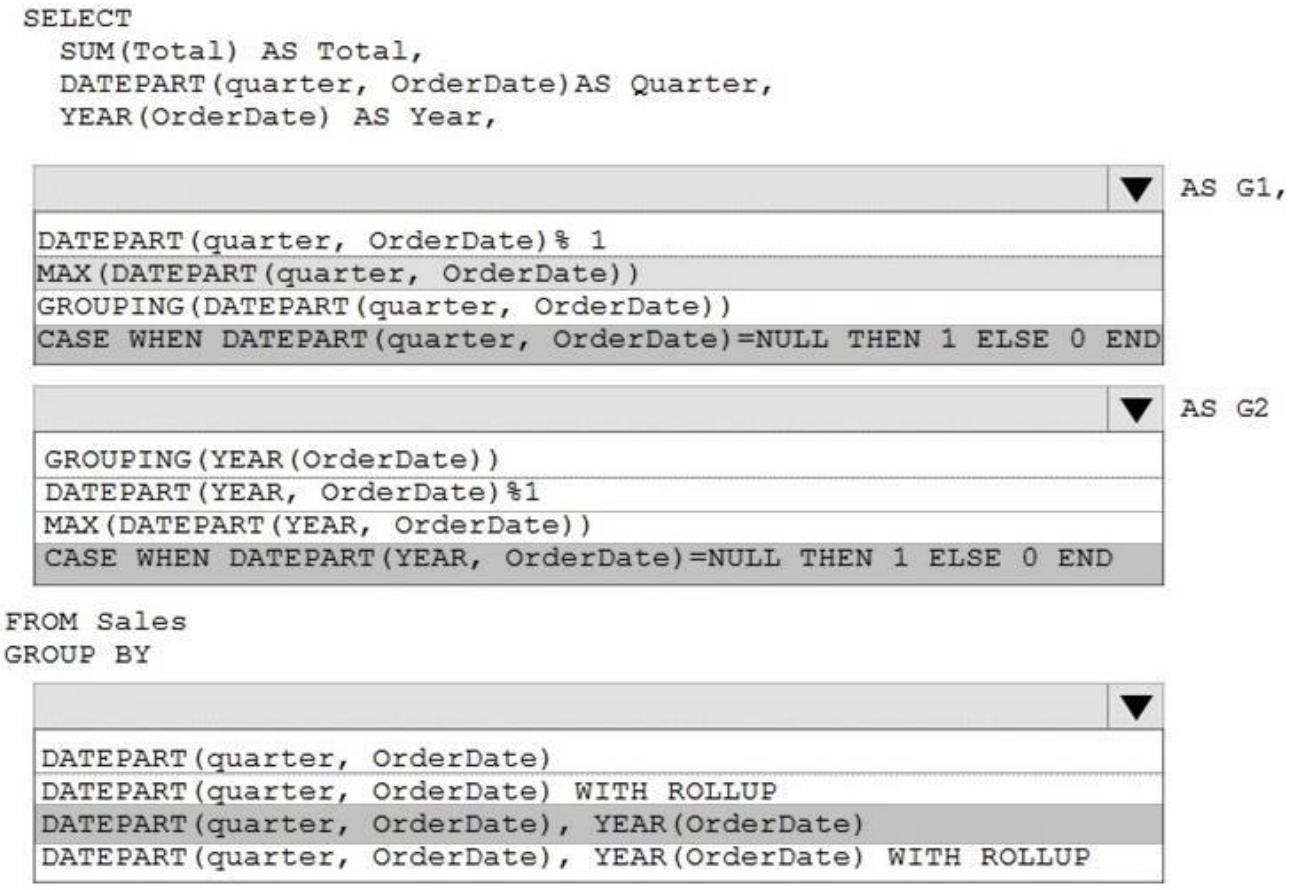
[▼] AS G2

- GROUPING(YEAR(OrderDate))
- DATEPART(YEAR, OrderDate)%1
- MAX(DATEPART(YEAR, OrderDate))
- CASE WHEN DATEPART(YEAR, OrderDate)=NULL THEN 1 ELSE 0 END

```
FROM Sales
GROUP BY
```

[▼]

- DATEPART(quarter, OrderDate)
- DATEPART(quarter, OrderDate) WITH ROLLUP
- DATEPART(quarter, OrderDate), YEAR(OrderDate)
- DATEPART(quarter, OrderDate), YEAR(OrderDate) WITH ROLLUP



QUESTION NO: 4
それぞれUserLoginとEmployeeという名前の2つのテーブルがあります。
次の要件を満たすTransact-SQLスクリプトを作成する必要があります。
-
UserLoginテーブルのId列の値が1の場合、スクリプトはUserLoginテーブルのIsDeleted列の値を1に更新する必要があります。
-
UserLoginテーブルの更新でエラーが発生したときに、EmployeeテーブルのId列の値が1の場合、スクリプトはEmployeeテーブルのIsDeleted列の値を1に更新する必要があります。
- エラーメッセージ "No tables updated！"
Employeeテーブルへの更新でエラーが発生したときに生成する必要があります。
ソリューションを開発するためにどの5つのTransact-SQLセグメントを使用する必要がありますか？ 回答するには、適切なTransact-SQLセグメントをTransact-SQLセグメントのリストから回答領域に移動して正しい順序で配置します。

Code segments

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

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

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

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

```
BEGIN CATCH
```

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

```
END CATCH
```

Answer Area

Answer:

Code segments

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

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

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

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

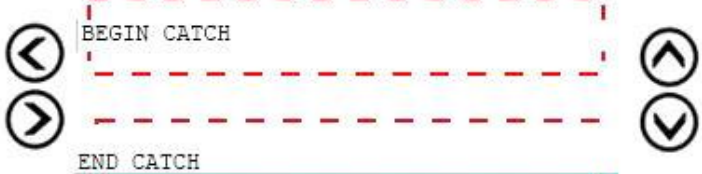
```
BEGIN CATCH
```

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

```
END CATCH
```

Answer Area

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



```
BEGIN CATCH
```

```
END CATCH
```

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

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

Explanation

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

QUESTION NO: 5
注：この質問は、同じまたは類似の回答の選択肢を使用する一連の質問の一部です。
回答の選択肢は、シリーズの複数の質問に対して正しいかもしれません。
各質問は、このシリーズの他の質問とは独立しています。
質問で提供される情報と詳細は、その質問にのみ適用されます。
Customer_CRMSystemおよびCustomer_HRSystemという名前のテーブルを含むデータベースがあります。 どちらのテーブルも次の構造を使用します：

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

テーブルには、次のレコードが含まれています。
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

CustomerCodeのnull値を含むレコードは、CustomerNameによって一意に識別できます。
Customer_HRSystem表に表示されない顧客のリストを表示する必要があります。
どのTransact-SQL文を実行する必要がありますか？

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. オプションE
B. オプションG

C. オプションA
D. オプションD
E. オプションH
F. オプションF
G. オプションC
H. オプションB
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

QUESTION NO: 6
注：この質問は、同じまたは類似の回答の選択肢を使用する一連の質問の一部です。
回答の選択肢は、シリーズの複数の質問に対して正しいかもしれません。
各質問は、このシリーズの他の質問とは独立しています。
質問で提供される情報と詳細は、その質問にのみ適用されます。
次のTransact-SQLステートメントを実行します。

```
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))
```

すべての顧客の合計年間収益を返し、顧客名と年間収益を示す各顧客の行を戻す必要があります。
どのTransact-SQL文を実行する必要がありますか？

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'
```

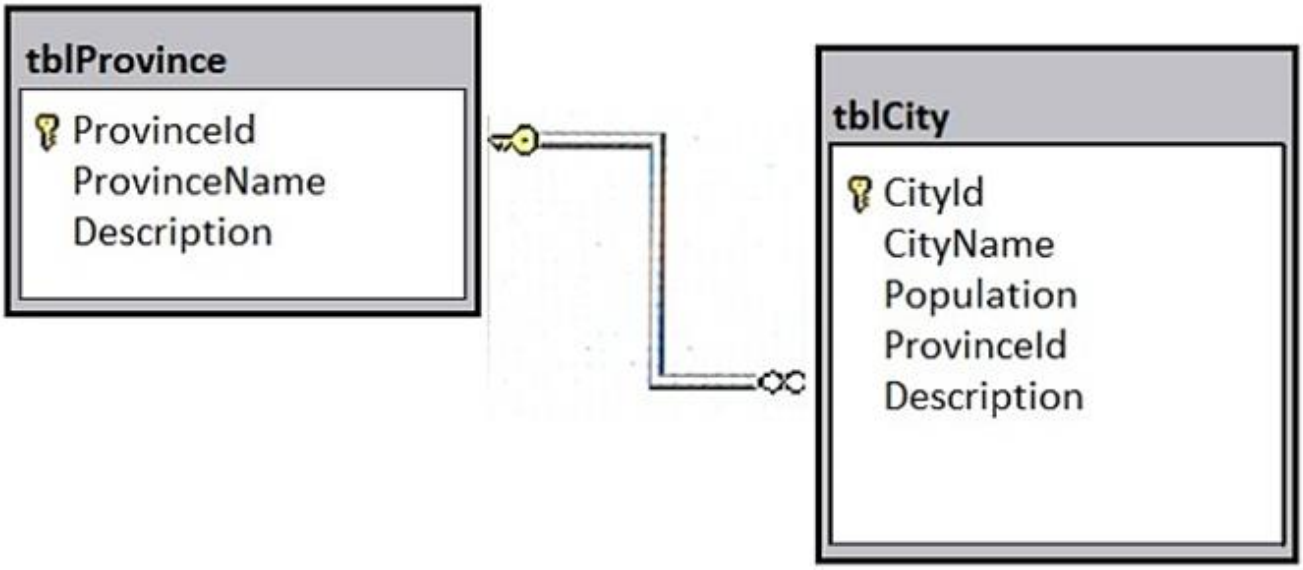
H

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

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

E. Option E
F. Option F
G. Option C
H. Option D
Answer: D

QUESTION NO: 7
注：この質問は、同じシナリオを提示する一連の質問の一部です。シリーズの各質問には、指定された目標を達成する可能性のある独自のソリューションが含まれています。一部の質問セットには複数の正しい解決策がある場合がありますが、他の質問セットには正しい解決策がない場合があります。
このセクションの質問に回答した後は、その質問に戻ることはできません。その結果、これらの質問はレビュー画面に表示されません。
次のデータベースダイアグラムに示すように、データベースには2つのテーブルがあります。



少なくとも2つの大都市があるすべての州をリストする必要があります。大都市とは、少なくとも100万人の人口を持つと定義されています。クエリは次の列を返す必要があります。
* tblProvince.ProvinceId
* tblProvince.ProvinceName
*州の大都市の総数を示すLargeCityCountという名前の派生列ソリューション：次のTransact-SQLステートメントを実行します。

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

ソリューションは目標を達成していますか？
A. いいえ

B. はい

Answer: A

Explanation

We should use CROSS APPLY rather than OUTER APPLY.

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

QUESTION NO: 8

あなたは、インデックス付きのビューの恩恵を受ける日付関連のクエリを持っています。

インデックス付きビューを作成する必要があります。

どの2つのTransact-SQL関数を使用できますか？ それぞれの正解は完全な解を提示します。

注：それぞれの正しい選択は1ポイントに値する

A. AT TIME ZONE

B. GETUTCDATE

C. DATEADD

D. DATEDIFF

Answer: B,D

Explanation

References:

https://docs.microsoft.com/en-us/sql/t-sql/functions/date-and-time-data-types-and-functions-transact-sql?view=s

QUESTION NO: 9

注：この質問は、同じシナリオを提示する一連の質問の一部です。

シリーズの各質問には、上記の目標を達成できる独自の解決策が含まれています。

いくつかの質問セットには1つ以上の正しい解決策があるかもしれないが、他の質問セットには正しい解決策がないかもしれない。

このセクションの質問に答えた後。 あなたはそれに戻ることはできません。

その結果、これらの質問はレビュー画面に表示されません。

北米の顧客の注文と配送を追跡するデータベースがあります。データベースには、次の表が含まれています：

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.
最寄りの顧客を返すクエリを作成する必要があります。
解決策：次のTransact-SQLステートメントを実行します。

```
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
```

変数@custIDは有効な顧客に設定されています。
解決策は目標を達成していますか？

A. はい

B. いいえ

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

QUESTION NO: 10
注：この質問は、同じまたは類似の回答の選択肢を使用する一連の質問の一部です。
回答の選択肢は、シリーズの複数の質問に対して正しいかもしれません。
各質問は、このシリーズの他の質問とは独立しています。
質問で提供される情報と詳細は、その質問にのみ適用されます。
いくつかの接続されたテーブルを含むデータベースがあります。
この表には、米国内の顧客の販売データのみが含まれています。
データベース内の販売テーブルのサンプルデータを生成するクエリを作成する必要があります。 クエリには、各顧客の在庫内のすべての商品を含める必要があります。
どのステートメント節を使用しますか？
A. GROUP BY CUBE
B. PIVOT
C. GROUP BY ROLLUP
D. CROSS JOIN
E. LEFT JOIN
F. UNPIVOT
G. MERGE
H. GROUP BY
Answer: C

QUESTION NO: 11
注：この質問は、同じまたは類似の回答の選択肢を使用する一連の質問の一部です。
回答の選択肢は、シリーズの複数の質問に対して正しいかもしれません。
各質問は、このシリーズの他の質問とは独立しています。
質問で提供される情報と詳細は、その質問にのみ適用されます。
Customer_CRMSystemおよびCustomer_HRSystemという名前のテーブルを含むデータベースがあります。 どちらのテーブルも次の構造を使用します：

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

テーブルには、次のレコードが含まれています。
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

CustomerCodeのnull値を含むレコードは、CustomerNameによって一意に識別できます。
両方のテーブルに表示され、適切な顧客コードを持つ顧客を表示する必要があります。
どのTransact-SQL文を実行する必要がありますか？

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 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. オプションC
B. オプションE
C. オプションA
D. オプションB
E. オプションD
Answer: C
Explanation
When there are null values in the columns of the tables being joined, the null values do not match each other.
The presence of null values in a column from one of the tables being joined can be returned only by using an outer join (unless the WHERE clause excludes null values).
References: https://technet.microsoft.com/en-us/library/ms190409(v=sql.105).aspx

QUESTION NO: 12
次の要件を満たすSalesという名前のテーブルを作成する必要があります。

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

どのTransact-SQLステートメントを実行しますか？

A

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

B

```
CREATE TABLE Sales (
      SalesID UNIQUEIDENTIFIER DEFAULT NEWSEQUENTIALID() PRIMARY KEY,
      SalesDate DateTime2 NOT NULL,
      SalesAmount money 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 NOT NULL IDENTITY(1,1),
      SalesDate DateTime2 NOT NULL,
      SalesAmount decimal(18,4) NULL,
      CONSTRAINT PK_SalesID PRIMARY KEY CLUSTERED (SalesID)
)
```

A. Option A
B. Option D
C. Option C
D. Option B
Answer: B

Explanation
datetime2 Defines a date that is combined with a time of day that is based on 24-hour clock. datetime2 can be considered as an extension of the existing datetime type that has a larger date range, a larger default fractional precision, and optional user-specified precision.

QUESTION NO: 13
注：この質問は、同じまたは類似の回答の選択肢を使用する一連の質問の一部です。
回答の選択肢は、シリーズの複数の質問に対して正しいかもしれません。各質問は、このシリーズの他の質問とは独立しています。質問で提供される情報と詳細は、その質問にのみ適用されます。
他のテーブルのデータに対する変更を追跡するAuditTrailという名前のテーブルがあります。
AuditTrailテーブルは、多くのプロセスによって更新されます。AuditTrailに入力されたデータに、不適切な形式の日付時刻値が含まれている可能性があります。あなたは、AuditTrailでいろいろなコラムからデータを検索するプロセスを実行します、しかし、時々、それはデータを有効な日付回の価格に変えることができないとき、プロセスはエラーを投げます。
あなたは、エヌ-
USフォーマット文化コードを使用している有効な日付回の価値にデータを変える必要があります。変換に失敗した場合は、列出力にNULL値を戻す必要があります。変換処理でエラーが発生してはなりません。
あなたは、何をインプリメントしなければなりませんか？
A. スカラー関数
B. テーブル値関数
C. ISNULL関数
D. COALESCE関数
E. TRY_CONVERT関数
F. TRY_PARSE関数
G. ストアドプロシージャ
H. ビュー
Answer: E
Explanation
A TRY_CONVERT function returns a value cast to the specified data type if the cast succeeds; otherwise, returns null.
References: https://msdn.microsoft.com/en-us/library/hh230993.aspx

QUESTION NO: 14
銀行システム用のデータベースがあります。
データベースには、tblDepositAcctとtblLoanAcctという2つのテーブルがあり、それぞれ預金口座と口ーン口座を格納しています。 どちらの表にも、次の列があります。

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.

あなたは預金口座だけを持っている顧客の総数を決定する必要があります。
どのTransact-SQLステートメントを実行しますか？

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 B
B. Option A
C. Option E
D. Option F
E. Option H
F. Option D
G. Option G
H. Option C
Answer: D
Explanation
References:
https://docs.microsoft.com/en-us/sql/t-sql/language-elements/set-operators-except-and-intersect-transact-sql?view

QUESTION NO: 15
Table1という名前の200,000,000行のテーブルがあります。
Table1には、DateTime2（3）のデータ型を持つSaleDateという名前の列が含まれています。
ユーザーは、次のクエリが遅く実行されることを報告します。

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

クエリを実行するのにかかる時間を短縮する必要があります。
WHEREステートメントを置き換えるのに何を使用すべきですか？
A. WHERE SaleDate >= '2017-01-01' AND SaleDate < '2018-01-01'
B. WHERE 2017 = year(SaleDate)
C. WHERE cast(SaleDate as varchar(10)) BETWEEN '2017-01-01' AND '2017-12-31'
D. WHERE cast(SaleDate as date) BETWEEN '2017-01-01' AND '2017-12-31'
Answer: D
Explanation
References: https://docs.microsoft.com/en-us/sql/t-sql/queries/select-transact-sql?view=sql-server-2017

QUESTION NO: 16
注：この質問は、同じシナリオを提示する一連の質問の一部です。
シリーズの各質問には、記載された目標を達成できる独自の解決策が含まれています。
いくつかの質問セットには1つ以上の正しい解決策があるかもしれないが、他の質問セットには正しい解決策がないかもしれない。
このセクションの質問に答えると、それに戻ることはできません。
その結果、これらの質問はレビュー画面に表示されません。

データウェアハウスで索引を作成しています。
10,000行のTable1というディメンション表があります。
行は、いくつかのレポートを生成するために使用されます。
レポートは主キーである列を結合します。
実行計画には、Table1のブックマークルックアップが含まれています。
レポートが予想よりも遅く実行されることがわかります。
レポートの実行に要する時間を短縮する必要があります。
解決策：主キー列にクラスタード・インデックスを作成します。
これは目標を満たしていますか？
A. Yes
B. No
Answer: A

QUESTION NO: 17
会社の営業チームは、北と南の2つの異なる地域に分かれています。
SalesNorthおよびSalesSouthという名前のテーブルを作成します。
SalesNorthテーブルには、北地域からの販売データが保存されます。
SalesSouthテーブルには、南地域の販売データが保存されます。
どちらのテーブルも次の構造を使用します。

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

両方のテーブルのすべてのレコードを含む統合結果セットを作成する必要があります。
どのTransact-SQL文を実行する必要がありますか？
A. SELECT salesID, customer, amount
FROM SalesNorth
UNION ALL
SELECT salesID, customer, amount
FROM SalesSouth
B. SELECT SalesNorth.salesID, SalesNorth.customer,
SalesNorth.amount, SalesSouth.salesID, SalesSouth.customer,
SalesSouth.amount
FROM SalesNorth
LEFT JOIN SalesSouth
ON SalesNorth.salesID=SalesSouth.salesID
C. SELECT salesID, customer, amount
FROM SalesNorth
UNION
SELECT salesID, customer, amount
FROM SalesSouth
D. SELECT SalesNorth.salesID, SalesNorth.customer,
SalesNorth.amount, SalesSouth.SalesID, SalesSouth.customer,

SalesSouth.amount
FROM SalesNorth
JOIN SalesSouth ON SalesNorth.salesID = SalesSouth.salesID

Answer: A

Explanation

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

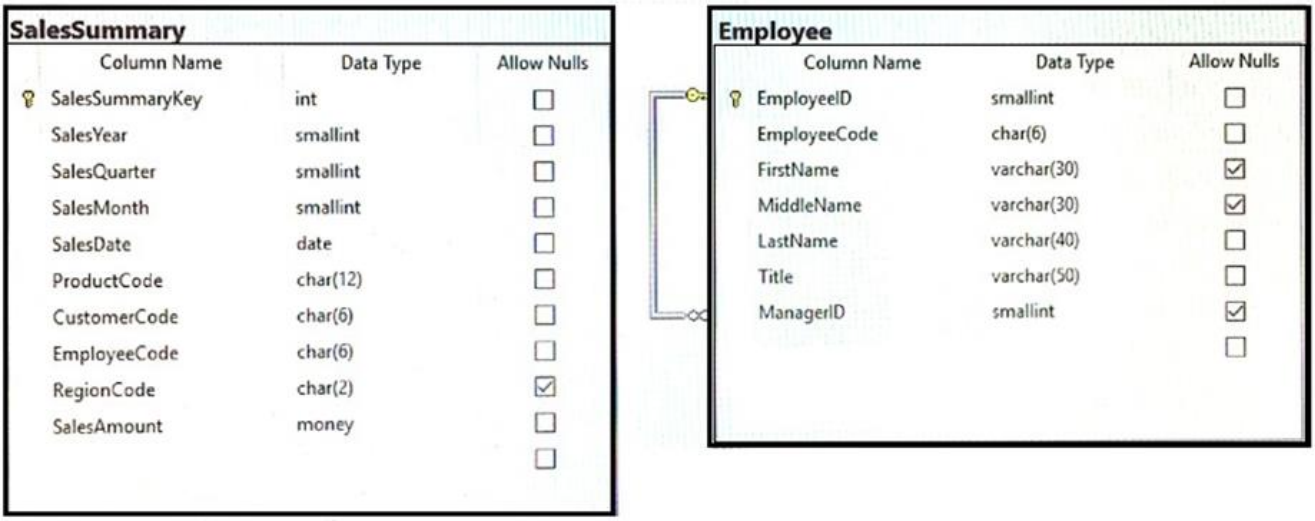
QUESTION NO: 18

注：この質問は、同じシナリオを使用する一連の質問の一部です。便宜上、シナリオは各質問で繰り返されます。各質問は異なる目標と回答の選択肢を提示しますが、シナリオのテキストは、このシリーズの各質問でまったく同じです。

繰り返しシナリオの開始

展示に示されているテーブルを含むデータベースがあります。

（[公開]ボタンをクリックします。）



Employeeテーブルを確認し、次のことを観察します。

*最高経営責任者（CEO）を除き、すべてのレコードのManagerIDに値があります。

* FirstNameおよびMiddleName列には、一部のレコードのnull値が含まれています。

* Title列の有効な値は、営業担当者、マネージャー、およびCEOです。

SalesSummaryテーブルを確認し、次の観察を行います。

*

ProductCode列には2つの部分があります。最初の5桁は製品コードを表し、最後の7桁は単価を表します。単価は、####。##というパターンを使用します。

*多くのレコードで、ProductCode列の単価部分に値が含まれていることがわかります。

*一部のレコードでは、RegionCode列にNULLが含まれています。

*販売データは、営業担当者についてのみ記録されます。

ビジネスをサポートするための一連のレポートと手順を作成しています。各レポートまたは手順の詳細は次のとおりです。

販売概要レポート：このレポートは、年と四半期ごとにデータを集計します。レポートは次の表のようになります。

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

セールスマネージャーレポート：このレポートには、各セールスマネージャーと、セールスマネージャーに報告する全従業員の総売上額がリストされます。
地域別売上高レポート：このレポートには、従業員別および地域別の総売上高がリストされます。レポートには、EmployeeCode、MiddleName、LastName、RegionCode、SalesAmountの列を含める必要があります。
MiddleNameがNULLの場合、FirstNameを表示する必要があります。
FirstNameとMiddleNameの両方にnull値がある場合、不明な世界を表示する必要があります。 RegionCodeがNULLの場合、Unknownという単語を表示する必要があります。
Report1：このレポートは、SalesSummaryのデータをEmployeeテーブルおよびその他のテーブルと結合します。
Report1をサポートするオブジェクトを作成する予定です。オブジェクトには次の要件があります。
*レポートのデータを提供するSELECTステートメントと結合可能であること
*レポートのSELECTステートメントで複数回使用できます
*レポートのSELECTステートメントでのみ使用可能
*永久オブジェクトとして保存されません
Report2：このレポートは、SalesSummaryのデータをEmployeeテーブルおよびその他のテーブルと結合します。
Report1をサポートするオブジェクトを作成する予定です。オブジェクトには次の要件があります。
*レポートのデータを提供するSELECTステートメントと結合可能であること
*このレポートと他のレポートで複数回使用できます
*パラメータを受け入れる
*永久オブジェクトとして保存されます
販売階層レポート。このレポートは、単一の結果セットの行を集計し、小計行を作成し、SalesAmount列の行を超集計します。レポートでは、SaleYear、SaleQuarter、SaleMonth列が階層として使用されます。結果セットには、総計またはクロス集計の集計行を含めることはできません。
現在の価格のストアドプロシージャ：このストアドプロシージャは、製品コードが提供されたときに製品の単価を返す必要があります。単価には、最初にドル記号を含める必要があります。また、単価には、小数点の左側に3桁ごとにコンマが含まれ、小数点の左側に2桁表示される必要があります。ストアード・プロシージャーは、製品コードに無効なデータが含まれている場合でも、エラーをスローしてはなりません。
繰り返しシナリオの終わり
現在の単価を返すストアドプロシージャを作成する必要があります。
ストアドプロシージャの定義をどのように完了する必要がありますか？回答するには、回答領域で適切なTransact-SQLセグメントを選択します。

```
CREATE PROC dbo.GetUnitPrice @Key int
AS
DECLARE @ColumnValue varchar(100),
  @UnitPrice money,
  @FormattedUnitPrice varchar(100)
SELECT @ColumnValue = ProductCode
FROM dbo.SalesSummary
WHERE SalesSummaryKey = @Key;
SET @ColumnValue = RIGHT(@ColumnValue, 7)
```

```
SET @UnitPrice = CAST(@ColumnValue AS money)
SET @UnitPrice = TRY_PARSE(@ColumnValue AS money)
SET @UnitPrice = PARSE(@ColumnValue AS money)
SET @UnitPrice = CONVERT(money, @ColumnValue)
```

```
SET @FormattedUnitPrice = FORMAT(@UnitPrice, 'c', 'en-US')
SET @FormattedUnitPrice = FORMAT(@UnitPrice, '$ #,###.##')
SET @FormattedUnitPrice = CONVERT(varchar(20), @UnitPrice, 1)
SET @FormattedUnitPrice = CONVERT(varchar(20), @UnitPrice, 2)
```

```
SELECT @FormattedUnitPrice AS UnitPrice
```

Answer:

```
CREATE PROC dbo.GetUnitPrice @Key int
AS
DECLARE @ColumnValue varchar(100),
  @UnitPrice money,
  @FormattedUnitPrice varchar(100)
SELECT @ColumnValue = ProductCode
FROM dbo.SalesSummary
WHERE SalesSummaryKey = @Key;
SET @ColumnValue = RIGHT(@ColumnValue, 7)
```

```
SET @UnitPrice = CAST(@ColumnValue AS money)
SET @UnitPrice = TRY_PARSE(@ColumnValue AS money)
SET @UnitPrice = PARSE(@ColumnValue AS money)
SET @UnitPrice = CONVERT(money, @ColumnValue)
```

```
SET @FormattedUnitPrice = FORMAT(@UnitPrice, 'c', 'en-US')
SET @FormattedUnitPrice = FORMAT(@UnitPrice, '$ #,###.##')
SET @FormattedUnitPrice = CONVERT(varchar(20), @UnitPrice, 1)
SET @FormattedUnitPrice = CONVERT(varchar(20), @UnitPrice, 2)
```

```
SELECT @FormattedUnitPrice AS UnitPrice
```

```
CREATE PROC dbo.GetUnitPrice @Key int
AS
DECLARE @ColumnValue varchar(100),
  @UnitPrice money,
  @FormattedUnitPrice varchar(100)
SELECT @ColumnValue = ProductCode
FROM dbo.SalesSummary
WHERE SalesSummaryKey = @Key;
SET @ColumnValue = RIGHT(@ColumnValue, 7)
```

SET @UnitPrice = CAST(@ColumnValue AS money)
SET @UnitPrice = TRY_PARSE(@ColumnValue AS money)
SET @UnitPrice = PARSE(@ColumnValue AS money)
SET @UnitPrice = CONVERT(money, @ColumnValue)

SET @FormattedUnitPrice = FORMAT(@UnitPrice, 'c', 'en-US')
SET @FormattedUnitPrice = FORMAT(@UnitPrice, '$ #,###.##')
SET @FormattedUnitPrice = CONVERT(varchar(20), @UnitPrice, 1)
SET @FormattedUnitPrice = CONVERT(varchar(20), @UnitPrice, 2)

```
SELECT @FormattedUnitPrice AS UnitPrice
```