KillTest

质量更高 服务更好

学习资料

http://www.killtest.cn

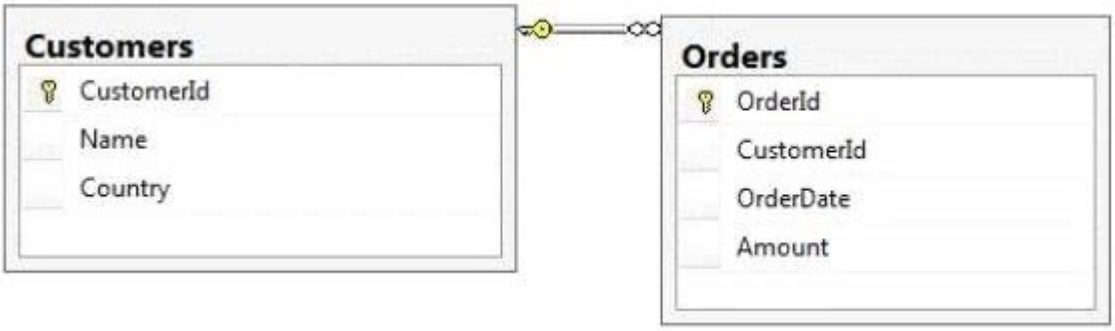
半年免费更新服务

Exam : **70-457**

Title : Transition Your MCTS on SQL Server 2008 to MCSA: SQL Server 2012, Part 1

Version : Demo

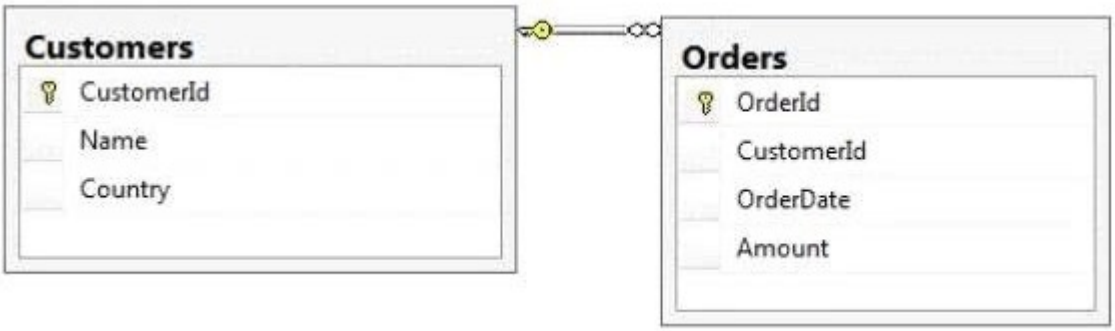
1.You administer a Microsoft SQL Server 2012 database named ContosoDb.
Tables are defined as shown in the exhibit. (Click the Exhibit button.)



You need to display rows from the Orders table for the Customers row having the CustomerId value set to 1 in the following XML format:
<row OrderId="1" OrderDate="2000-01-01T00:00:00" Amount="3400.00" Name="Customer A" Country="Australia" />
<row OrderId="2" OrderDate="2001-01-01T00:00:00" Amount="4300.00" Name="Customer A" Country="Australia" />
Which Transact-SQL query should you use?
A. SELECT OrderId, OrderDate, Amount, Name, Country
FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers.CustomerId
WHERE Customers.CustomerId = 1
FOR XML RAW
B. SELECT OrderId, OrderDate, Amount, Name, Country
FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers.CustomerId
WHERE Customers.CustomerId = 1
FOR XML RAW, ELEMENTS
C. SELECT OrderId, OrderDate, Amount, Name, Country
FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers.CustomerId
WHERE Customers.CustomerId = 1
FOR XML AUTO
D. SELECT OrderId, OrderDate, Amount, Name, Country
FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers.CustomerId
WHERE Customers.CustomerId = 1
FOR XML AUTO, ELEMENTS
E. SELECT Name, Country, OrderId, OrderDate, Amount
FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers.CustomerId
WHERE Customers.CustomerId = 1
FOR XML AUTO
F. SELECT Name, Country, OrderId, OrderDate, Amount
FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers.CustomerId
WHERE Customers.CustomerId = 1
FOR XML AUTO, ELEMENTS
G. SELECT Name AS '@Name', Country AS '@Country', OrderId, OrderDate, Amount
FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers.CustomerId

WHERE Customers.CustomerId = 1
FOR XML PATH ('Customers')
H. SELECT Name AS 'Customers/Name', Country AS 'Customers/Country', OrderId, OrderDate, Amount
FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers.CustomerId
WHERE Customers.CustomerId = 1
FOR XML PATH ('Customers')
Answer: A

2.You administer a Microsoft SQL Server 2012 database named ContosoDb.
Tables are defined as shown in the exhibit. (Click the Exhibit button.)



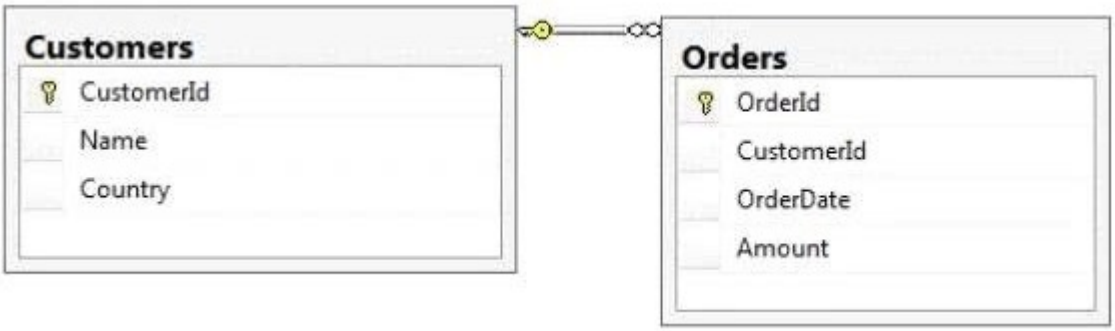
You need to display rows from the Orders table for the Customers row having the CustomerIdvalue set to 1 in the following XML format.

```
<Orders OrderId="1" OrderDate="2000-01-01T00:00:00" Amount="3400.00">
  <Customers Name="Customer A" Country="Australia" />
</Orders>
<Orders OrderId="2" OrderDate="2001-01-01T00:00:00" Amount="4300.00">
  <Customers Name="Customer A" Country="Australia" />
</Orders>
```

Which Transact-SQL query should you use?
A. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML RAW
B. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers=CustomerId = 1 FOR XML RAW, ELEMENTS
C. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1
FOR XML AUTO
D. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers.CustomerId WHERE Customers.CustomerId = 1 FOR XML AUTO, ELEMENTS
E. SELECT Name, Country, OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML AUTO
F. SELECT Name, Country, OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML AUTO, ELEMENTS

G. SELECT Name AS '@Name', Country AS '@Country', OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML PATH ('Customers')
H. SELECT Name AS 'Customers/Name', Country AS 'Customers/Country', OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML PATH ('Customers')
Answer: C

3.You administer a Microsoft SQL Server 2012 database named ContosoDb.
Tables are defined as shown in the exhibit. (Click the Exhibit button.)



You need to display rows from the Orders table for the Customers row having the CustomerId value set to 1 in the following XML format.

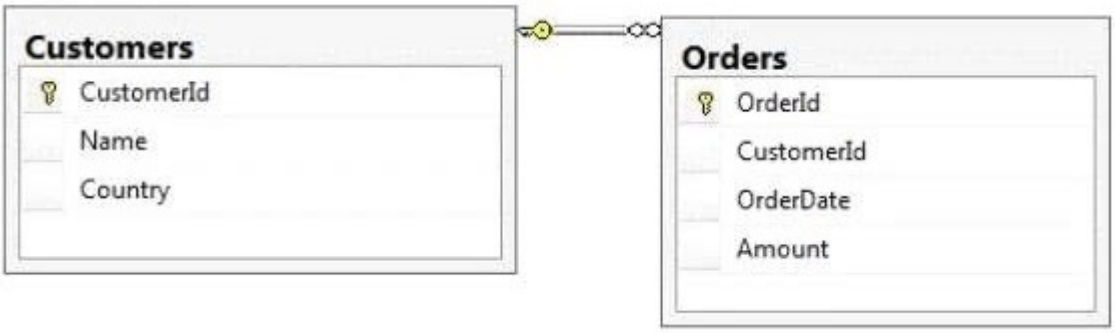
```
<CUSTOMERS Name="Customer A" Country="Australia">
  <ORDERS OrderID="1" OrderDate="2001-01-01" Amount="3400.00" />
  <ORDERS OrderID="2" OrderDate="2002-01-01" Amount="4300.00" />
</CUSTOMERS>
```

Which Transact-SQL query should you use?
A. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML RAW
B. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML RAW, ELEMENTS
C. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML AUTO
D. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML AUTO, ELEMENTS
E. SELECT Name, Country, OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML AUTO
F. SELECT Name, Country, OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML AUTO, ELEMENTS
G. SELECT Name AS '@Name', Country AS '@Country', OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML PATH ('Customers')

H. SELECT Name AS 'Customers/Name', Country AS 'Customers/Country', OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML PATH ('Customers')

Answer: E

4.You administer a Microsoft SQL Server 2012 database named ContosoDb.
Tables are defined as shown in the exhibit. (Click the Exhibit button.)



You need to display rows from the Orders table for the Customers row having the CustomerId value set to 1 in the following XML format.

```
<Orders>
  <OrderId>1</OrderId>
  <OrderDate>2000-01-01T00:00:00</OrderDate>
  <Amount>3400.00</Amount>
  <Customers>
    <Name>Customer A</Name>
    <Country>Australia</Country>
  </Customers>
</Orders>
<Orders>
  <OrderId>2</OrderId>
  <OrderDate>2001-01-01T00:00:00</OrderDate>
  <Amount>4300.00</Amount>
  <Customers>
    <Name>Customer A</Name>
    <Country>Australia</Country>
  </Customers>
</Orders>
```

Which Transact-SQL query should you use?

A. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML RAW

B. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML RAW, ELEMENTS

C. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML AUTO

D. SELECT OrderId, OrderDate, Amount, Name, Country FROM Orders INNER JOIN Customers ON

Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML AUTO, ELEMENTS
E. SELECT Name, Country, OrderId, OrderDate, Amount
FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers.
CustomerId
WHERE Customers.CustomerId = 1
FOR XML AUTO
F. SELECT Name, Country, OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML AUTO, ELEMENTS
G. SELECT Name AS '@Name', Country AS '@Country', OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML PATH ('Customers')
H. SELECT Name AS 'Customers/Name', Country AS 'Customers/Country', OrderId, OrderDate, Amount FROM Orders INNER JOIN Customers ON Orders.CustomerId = Customers. CustomerId WHERE Customers.CustomerId = 1 FOR XML PATH ('Customers')
Answer: D

5.You develop a Microsoft SQL Server 2012 server database that supports an application.
The application contains a table that has the following definition:
CREATE TABLE Inventory (
ItemID int NOT NULL PRIMARY KEY,
ItemsInStore int NOT NULL,
ItemsInWarehouse int NOT NULL)
You need to create a computed column that returns the sum total of the ItemsInStore and ItemsInWarehouse values for each row. The new column is expected to be queried heavily, and you need to be able to index the column.
Which Transact-SQL statement should you use?
A. ALTER TABLE Inventory ADD TotalItems AS ItemsInStore + ItemsInWarehouse
B. ALTER TABLE Inventory ADD TotalItems AS ItemsInStore + ItemsInWarehouse PERSISTED
C. ALTER TABLE Inventory ADD TotalItems AS SUM(ItemsInStore, ItemsInWarehouse) PERSISTED
D. ALTER TABLE Inventory ADD TotalItems AS SUM(ItemsInStore, ItemsInWarehouse)
Answer: B

6.You develop a Microsoft SQL Server 2012 database that contains a table named Customers.
The Customers table has the following definition:

```
CREATE TABLE [dbo].[Customers](
   [CustomerId] [bigint] NOT NULL,
   [MobileNumber] [nvarchar](25) NOT NULL,
   [HomeNumber] [nvarchar](25) NULL,
   [Name] [nvarchar](50) NOT NULL,
   [Country] [nvarchar](25) NOT NULL,
    CONSTRAINT [PK_Customers] PRIMARY KEY CLUSTERED
   (
     [CustomerId] ASC
   ) ON [PRIMARY]
) ON [PRIMARY]
```

You need to create an audit record only when either the MobileNumber or HomeNumber column is updated.
Which Transact-SQL query should you use?
A. CREATE TRIGGER TrgPhoneNumberChange ON Customers FOR UPDATE AS IF COLUMNS_UPDATED (HomeNumber, MobileNumber)
- - Create Audit Records
B. CREATE TRIGGER TrgPhoneNumberChange ON Customers FOR UPDATE AS IF EXISTS(SELECT HomeNumber FROM inserted) OR EXISTS (SELECT MobileNumber FROM inserted)
- - Create Audit Records
C. CREATE TRIGGER TrgPhoneNumberChange ON Customers FOR UPDATE AS IF COLUMNS_CHANGED (HomeNumber, MobileNumber)
- - Create Audit Records
D. CREATE TRIGGER TrgPhoneNumberChange ON Customers FOR UPDATE AS IF UPDATE (HomeNumber) OR UPDATE (MobileNumber)
- - Create Audit Records
Answer: D

7.You develop a Microsoft SQL Server 2012 database that has two tables named SavingAccounts and LoanAccounts. Both tables have a column named AccountNumber of the nvarchar data type. You use a third table named Transactions that has columns named TransactionId, AccountNumber, Amount, and TransactionDate. You need to ensure that when multiple records are inserted in the Transactions table, only the records that have a valid AccountNumber in the SavingAccounts or LoanAccounts are inserted.
Which Transact-SQL statement should you use?
A. CREATE TRIGGER TrgValidateAccountNumber ON Transactions INSTEAD OF INSERT AS BEGIN
 INSERT INTO Transactions
 SELECT TransactionID,AccountNumber,Amount,TransactionDate FROM inserted
 WHERE AccountNumber IN
 (SELECT AccountNumber FROM LoanAccounts
 UNION SELECT AccountNumber FROM SavingAccounts))
END
B. CREATE TRIGGER TrgValidateAccountNumber ON Transactions FOR INSERT AS BEGIN
 INSERT INTO Transactions
 SELECT TransactionID,AccountNumber,Amount,TransactionDate FROM inserted

```
  WHERE AccountNumber IN
  (SELECT AccountNumber FROM LoanAccounts
  UNION SELECT AccountNumber FROM SavingAccounts))
END
```

C.
```
CREATE TRIGGER TrgValidateAccountNumber ON Transactions INSTEAD OF INSERT AS BEGIN
  IF EXISTS (
  SELECT AccountNumber FROM inserted EXCEPT
  (SELECT AccountNumber FROM LoanAccounts
  UNION SELECT AccountNumber FROM SavingAccounts))
  BEGIN
  ROLLBACK TRAN
  END
END
```

D.
```
CREATE TRIGGER TrgValidateAccountNumber ON Transactions FOR INSERT AS BEGIN
  IF EXISTS (
  SELECT AccountNumber FROM inserted EXCEPT
  (SELECT AccountNumber FROM LoanAccounts
  UNION SELECT AccountNumber FROM SavingAccounts))
  BEGIN
  ROLLBACK TRAN
  END
END
```

Answer: A

8.You develop a Microsoft SQL Server 2012 database.
You create a view that performs the following tasks:
- Joins 8 tables that contain up to 500,000 records each.
- Performs aggregations on 5 fields.

The view is frequently used in several reports. You need to improve the performance of the reports.
What should you do?
A. Convert the view into a table-valued function.
B. Convert the view into a Common Table Expression (CTE).
C. Convert the view into an indexed view.
D. Convert the view into a stored procedure and retrieve the result from the stored procedure into a temporary table.
Answer: C

9.You are a database developer of a Microsoft SQL Server 2012 database.
The database contains a table named Customers that has the following definition:

```
CREATE TABLE Customer
(CustomerID INT NOT NULL PRIMARY KEY,
 CustomerName VARCHAR(255) NOT NULL,
 CustomerAddress VARCHAR(1000) NOT NULL)
```

You are designing a new table named Orders that has the following definition:

```
CREATE TABLE Orders
(OrderID INT NOT NULL PRIMARY KEY,
 CustomerID INT NOT NULL,
 OrderDescription VARCHAR(2000))
```

You need to ensure that the CustomerId column in the Orders table contains only values that exist in the CustomerId column of the Customer table.
Which Transact-SQL statement should you use?
A. ALTER TABLE Orders ADD CONSTRAINT FK_Orders_CustomerID FOREIGN KEY (CustomerID) REFERENCES
Customer (CustomerID)
B. ALTER TABLE Customer ADD CONSTRAINT FK_Customer_CustomerID FOREIGN KEY (CustomerID) REFERENCES Orders (CustomerID)
C. ALTER TABLE Orders ADD CONSTRAINT CK_Orders_CustomerID CHECK (CustomerID IN (SELECT CustomerId FROM Customer))
D. ALTER TABLE Customer ADD OrderID INT NOT NULL;
ALTER TABLE Customer
ADD CONSTRAINT FK_Customer_OrderID FOREIGN KEY (OrderID) REFERENCES Orders (OrderID);
E. ALTER TABLE Orders ADD CONSTRAINT PK_Orders_CustomerID PRIMARY KEY (CustomerID)
Answer: A

10.You have three tables that contain data for vendors, customers, and agents. You create a view that is used to look up telephone numbers for these companies. The view has the following definition: You need to ensure that users can update only the phone numbers by using this view.
What should you do?

```
Create view apt.vwCompanyPhoneList
(Source, CompanyID, CompanyNumber,
 LastName, FirstName, BusinessName, Phone)
as

SELECT 'Customer' as Source
  , CustomerID
  , CustomerNumber
  , CustomerLastName
  , CustomerFirstName
  , CustomerBusinessName
  , Phone
FROM apt.Customer
UNION ALL
SELECT 'Agent' as Source
  , AgentID
  , AgentNumber
  , AgentLastName
  , AgentFirstName
  , AgentBusinessName
  , Phone
FROM apt.Agent
UNION ALL
SELECT 'Vendor' as Source
  , VendorID
  , VendorNumber
  , VendorLastName
  , VendorFirstName
  , VendorBusinessName
  , Phone
FROM apt.Vendor
GO
```

A. Alter the view. Use the EXPAND VIEWS query hint along with each SELECT statement.
B. Create an INSTEAD OF UPDATE trigger on the view.
C. Drop the view. Re-create the view by using the SCHEMABINDING clause, and then create an index on the view.
D. Create an AFTER UPDATE trigger on the view.
Answer: B

11.You develop a Microsoft SQL Server 2012 database. You create a view from the Orders and OrderDetails tables by using the following definition.

```
CREATE VIEW vOrders
WITH SCHEMABINDING
AS
SELECT o.ProductID,
  o.OrderDate,
  SUM(od.UnitPrice * od.OrderQty) AS Amount
FROM OrderDetails AS od INNER JOIN
  Orders AS o ON od.OrderID = o.OrderID
WHERE od.SalesOrderID = o.SalesOrderID
GROUP BY o.OrderDate, o.ProductID
GO
```

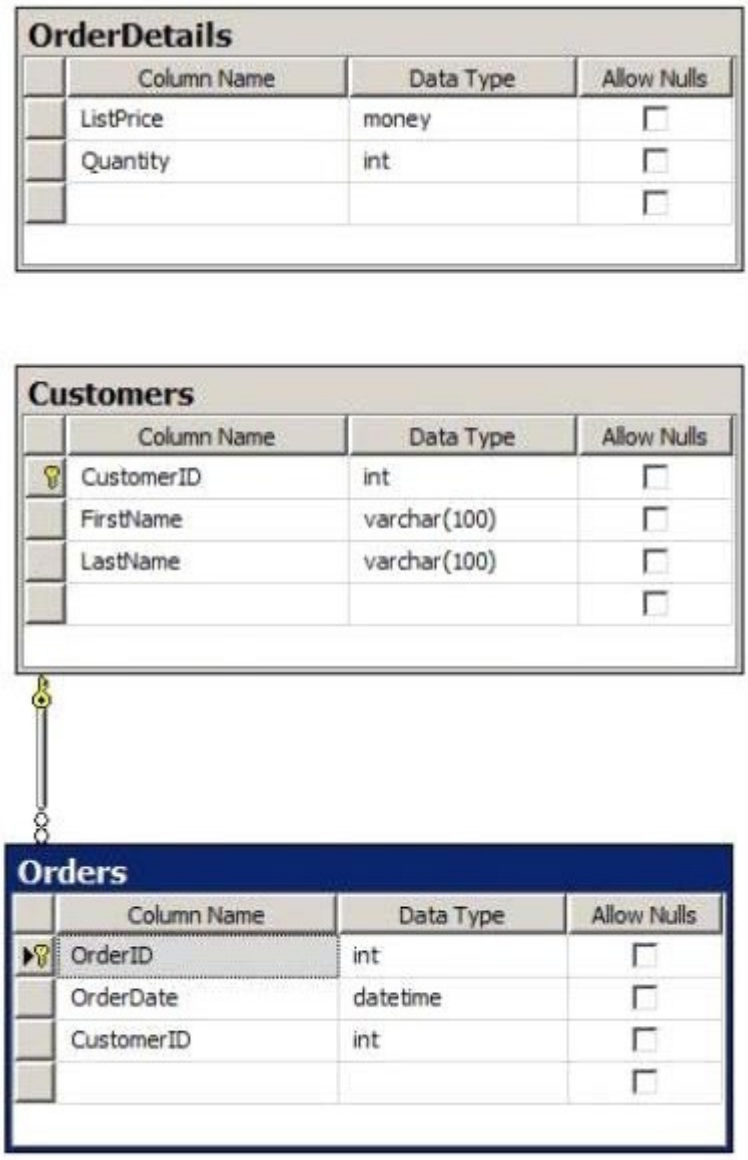
You need to ensure that users are able to modify data by using the view.
What should you do?
A. Create an AFTER trigger on the view.
B. Modify the view to use the WITH VIEW_METADATA clause.
C. Create an INSTEAD OF trigger on the view.
D. Modify the view to an indexed view.
Answer: C

12.You have a view that was created by using the following code:

```
CREATE VIEW Sales.OrdersByTerritory
AS
SELECT OrderID
      ,OrderDate
      ,SalesTerritoryID
      ,TotalDue
FROM Sales.Orders;
```

You need to create an inline table-valued function named Sales.fn_OrdersByTerritory, Which must meet the following requirements:
- Accept the @T integer parameter.
- Use one-part names to reference columns.
- Filter the query results by SalesTerritoryID.
- Return the columns in the same order as the order used in OrdersByTerritoryView.
Which code segment should you use? To answer, type the correct code in the answer area.
A. CREATE FUNCTION Sales.fn_OrdersByTerritory (@T int) RETURNS TABLE AS RETURN
(
 SELECT OrderID,OrderDate,SalesTerrirotyID,TotalDue
 FROM Sales.OrdersByTerritory
 WHERE SalesTerritoryID = @T
)
Answer: A

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



You deploy a new server that has SQL Server 2012 installed. You need to create a table named Sales. OrderDetails on the new server. Sales.OrderDetails must meet the following requirements:

- Write the results to a disk.
- Contain a new column named LineItemTotal that stores the product of ListPrice and Quantity for each row.
- The code must NOT use any object delimiters.

The solution must ensure that LineItemTotal is stored as the last column in the table.

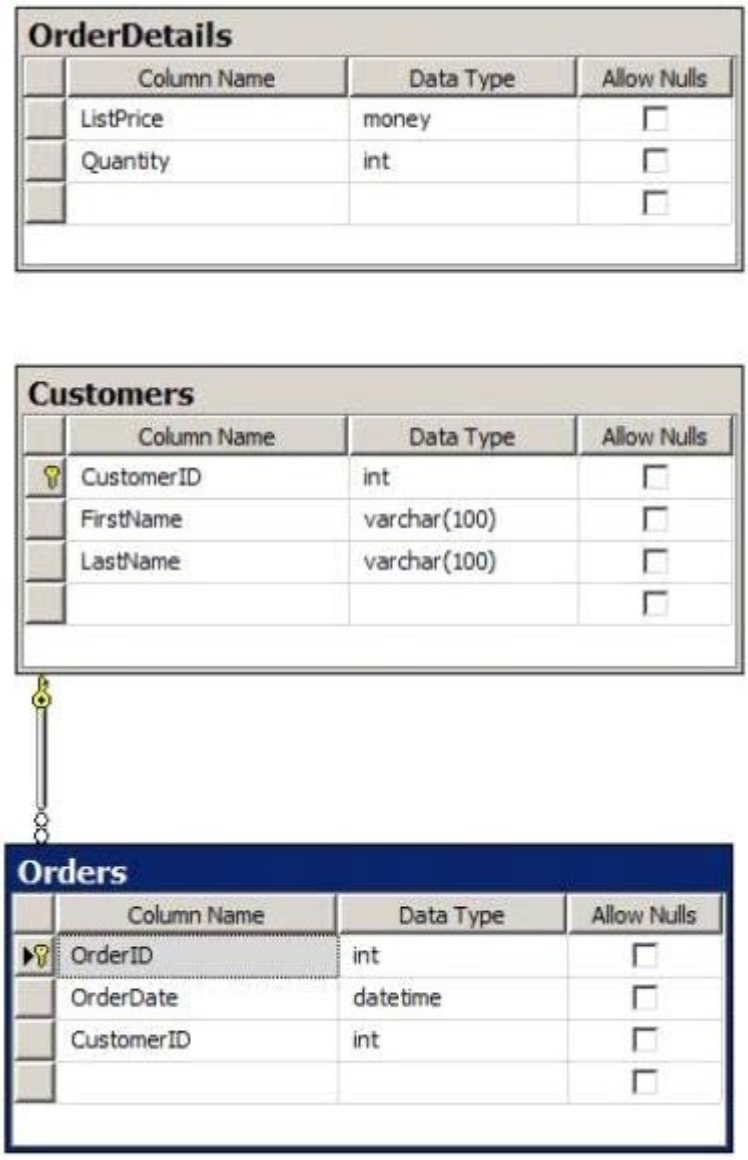
Which code segment should you use? To answer, type the correct code in the answer area.

A. CREATE TABLE Sales.OrderDetails (ListPrice money not null, Quantity int not null, LineItemTotal as (ListPrice * Quantity) PERSISTED)

B. CREATE TABLE Sales.OrderDetails (ListPrice money not null, Quantity int not null, LineItemTotal as (ListPrice * Quantity))

Answer: A

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



You need to create a view named uv_CustomerFullName to meet the following requirements:
- The code must NOT include object delimiters.
- The view must be created in the Sales schema.
- Columns must only be referenced by using one-part names.
- The view must return the first name and the last name of all customers.
- The view must prevent the underlying structure of the customer table from being changed.
- The view must be able to resolve all referenced objects, regardless of the user's default schema.

Which code segment should you use? To answer, type the correct code in the answer area.

A. CREATE VIEW Sales.uv_CustomerFullName with Schemabinding AS SELECT FirstName, LastName FROM Customers

B. CREATE VIEW Sales.uv_CustomerFullName AS SELECT FirstName, LastName FROM Customers

Answer: A

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

You need to create a query that calculates the total sales of each OrderId from the Sales.Details table. The solution must meet the following requirements:
- Use one-part names to reference columns.
- Sort the order of the results from OrderId.
- NOT depend on the default schema of a user.
- Use an alias of TotalSales for the calculated ExtendedAmount.
- Display only the OrderId column and the calculated TotalSales column.

Which code segment should you use? To answer, type the correct code in the answer area.
A. SELECT OrderID, SUM(ExtendedAmount) AS TotalSales FROM Sales.Details GROUP BY OrderID ORDER BY OrderID
B. SELECT OrderID, SUM(ExtendedAmount) AS TotalSales FROM Sales.Details ORDER BY OrderID
Answer: A

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

You have the following query:

```
SELECT SalesTerritoryID,
  ProductID,
  AVG(UnitPrice),
  MAX(OrderQty),
  MAX(DiscountAmount)
FROM Sales.Details
```

You need to recreate the query to meet the following requirements:
- Reference columns by using one-part names only.
- Sort aggregates by SalesTerritoryID, and then by ProductID.
- Order the results in descending order from SalesTerritoryID to ProductID.
- The solution must use the existing SELECT clause and FROM clause.

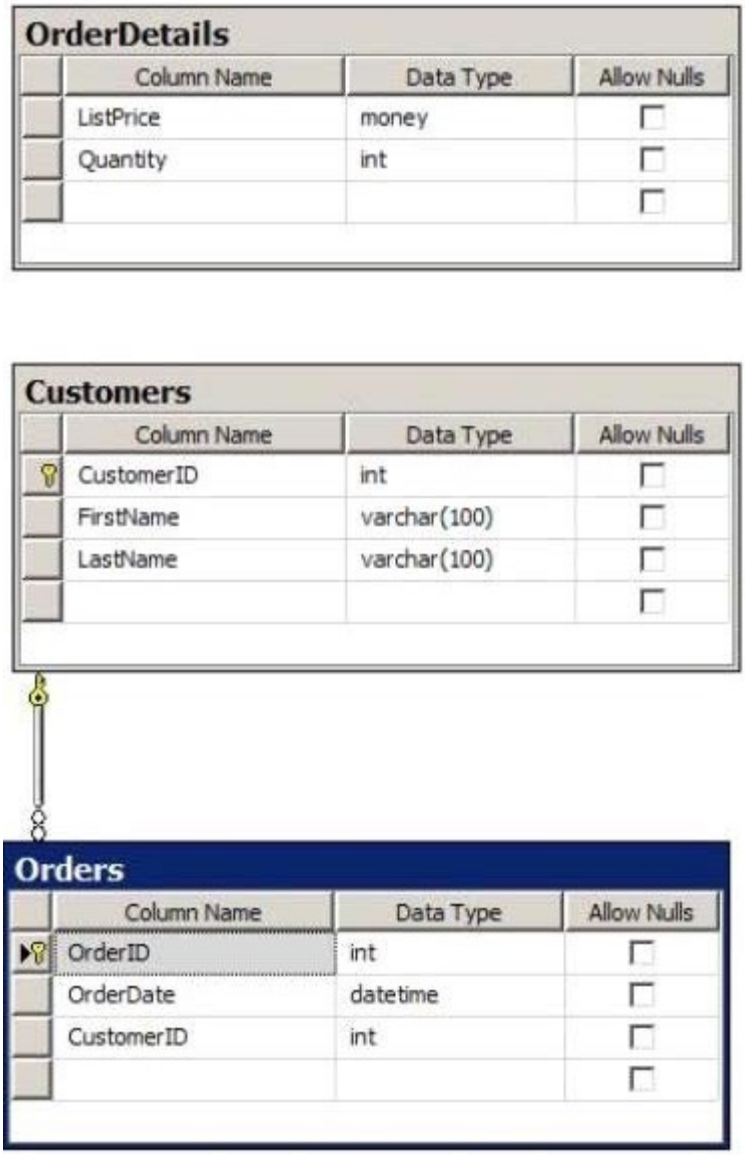
Which code segment should you use? To answer, type the correct code in the answer area.

A. SELECT SalesTerritoryID, ProductID, AVG(UnitPrice), MAX(OrderQty), MAX(DiscountAmount) FROM Sales.Details GROUP BY SalesTerritoryID,ProductID ORDER BY SalesTerritoryID DESC, ProductID DESC

B. SELECT SalesTerritoryID, ProductID, AVG(UnitPrice), MAX(OrderQty), MAX(DiscountAmount) FROM Sales.Details ORDER BY SalesTerritoryID DESC, ProductID DESC

Answer: A

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



You need to create a query for a report. The query must meet the following requirements:
- NOT use object delimiters.
- Return the most recent orders first.
- Use the first initial of the table as an alias.
- Return the most recent order date for each customer.
- Retrieve the last name of the person who placed the order.
- Return the order date in a column named MostRecentOrderDate that appears as the last column in the report.

The solution must support the ANSI SQL-99 standard.
Which code segment should you use? To answer, type the correct code in the answer area.
A. SELECT c.CustomerID -- optional c.LastName, max(o.OrderDate) 'MostRecentOrderDate' FROM Customer c LEFT OUTER JOIN Orders o ON o.CustomerID = c.CustomerID GROUP BY c.CustomerID, c.LastName ORDER BY 3 DESC
B. select C.Lastname, P.MostRecentOrderDate from customers AS C inner join (select customID,

MostRecentOrderDate=max(orderDate) from orders group by customID
) P
on C.customerID=P.CustomerID
order by P.MostRecentOrderDate desc
C. SELECT C.LastName, O.OrderDate AS MostRecentOrderDate FROM Customers AS C INNER JOIN Orders AS O ON C.CustomerID = O.CustomerID ORDER BY O.OrderDate DESC
Answer: A

18.You have an XML schema collection named Sales.InvoiceSchema. You need to declare a variable of the XML type named XML1. The solution must ensure that XML1 is validated by using Sales.InvoiceSchema.
Which code segment should you use? To answer, type the correct code in the answer area.
A. Declare @XML1=XML(Sales.InvoiceSchema)
B. DECLARE @XML1 XML @XML1 = Sales.InvoiceSchema CREATE XML SCHEMA COLLECTION XML1 AS @XML1
Answer: A

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

You need to create a query that returns a list of products from Sales.ProductCatalog.
The solution must meet the following requirements:

- UnitPrice must be returned in descending order.
- The query must use two-part names to reference the table.
- The query must use the RANK function to calculate the results.
- The query must return the ranking of rows in a column named PriceRank.
- The list must display the columns in the order that they are defined in the table.
- PriceRank must appear last.

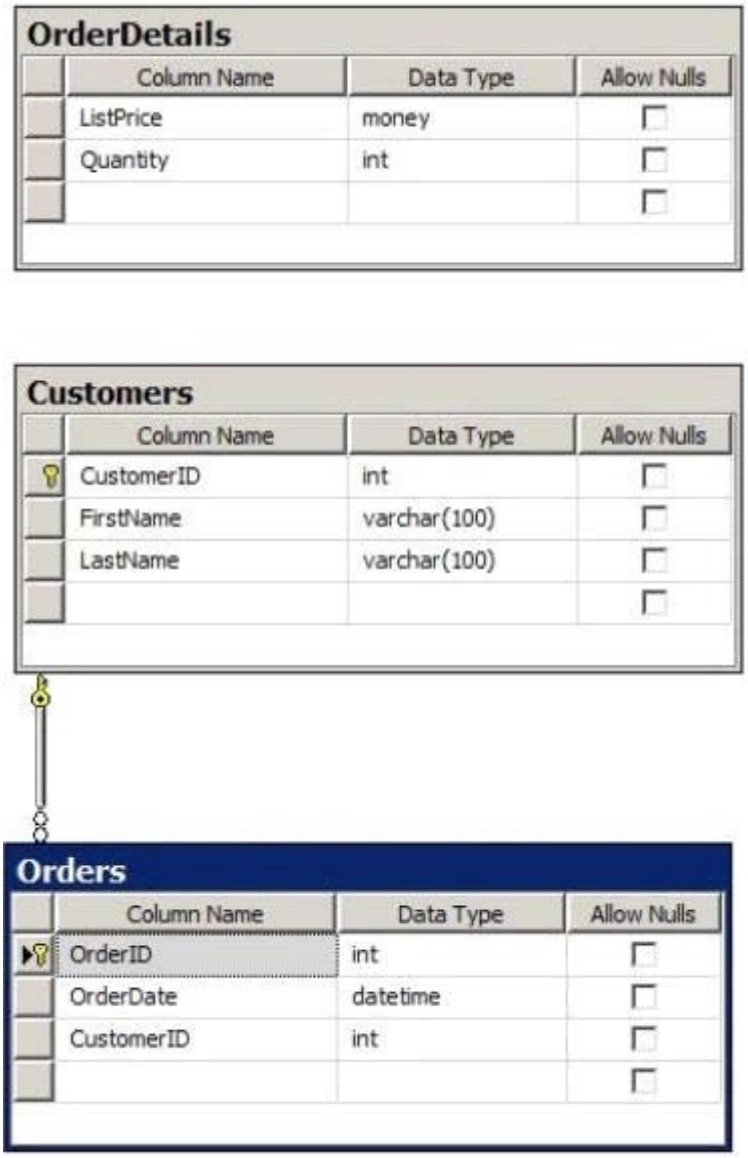
Which code segment should you use? To answer, type the correct code in the answer area.

A. SELECT ProductCatalog.CatID, ProductCatalog.CatName, ProductCatalog. ProductID, ProductCatalog.ProdName, ProductCatalog.UnitPrice, RANK() OVER (ORDER BY ProductCatalog.UnitPrice DESC) AS PriceRank
FROM Sales.ProductCatalog
ORDER BY ProductCatalog.UnitPrice DESC

B. SELECT ProductCatalog.CatID, ProductCatalog.CatName, ProductCatalog. ProductID, ProductCatalog.ProdName, ProductCatalog.UnitPrice, RANK() OVER (PARTITION BY ProductCatalog.UnitPrice ORDER BY ProductCatalog. UnitPrice DESC) AS PriceRank FROM Sales.ProductCatalog ORDER BY ProductCatalog.UnitPrice DESC

Answer: A

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



You have an application named Appl. You have a parameter named @Count that uses the int data type. App1 is configured to pass @Count to a stored procedure.

You need to create a stored procedure named usp_Customers for Appl. Usp_Customers must meet the following requirements:

- NOT use object delimiters.
- Minimize sorting and counting.
- Return only the last name of each customer in alphabetical order.
- Return only the number of rows specified by the @Count parameter.
- The solution must NOT use BEGIN and END statements.

Which code segment should you use? To answer, type the correct code in the answer area.

A. CREATE PROCEDURE usp_Customers @Count int AS SELECT TOP(@Count) LastName FROM Customers ORDER BY LastName

Answer: A