Vendor: Microsoft

Exam Code: 70-464

Exam Name: Developing Microsoft SQL Server 2012 Databases

Version: Demo

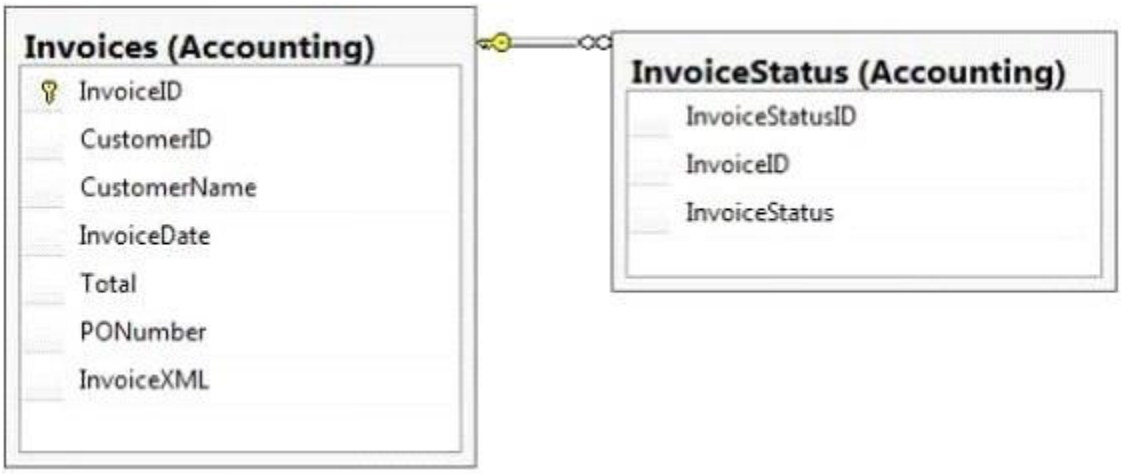
Topic 1, Scenario 1

Application Information
Your company receives invoices in XML format from customers. Currently, the invoices are stored as files and processed by a desktop application. The application has several performance and security issues. The application is being migrated to a SQL Server-based solution. A schema named InvoiceSchema has been created for the invoices xml.

The data in the invoices is sometimes incomplete. The incomplete data must be stored and processed as-is. Users cannot filter the data provided through views.

You are designing a SQL Server database named DB1 that will be used to receive, process, and securely store the invoice data. A third-party Microsoft .NET Framework component will be purchased to perform tax calculations. The third-party tax component will be provided as a DLL file named Treytax.dll and a source code file named Amortize.cs. The component will expose a class named TreyResearch and a method named Amortize().
The files are located in c:\temp\.

The following graphic shows the planned tables:



You have a sequence named Accounting.InvoiceID_Seq.

You plan to create two certificates named CERT1 and CERT2. You will create CERT1 in master. You will create CERT2 in DB1.

You have a legacy application that requires the ability to generate dynamic T-SQL statements against DB1. A sample of the queries generated by the legacy application appears in Legacy.sql.

Application Requirements
The planned database has the following requirements:

- All stored procedures must be signed.
- The original XML invoices must be stored in the database.
- An XML schema must be used to validate the invoice data.
- Dynamic T-SQL statements must be converted to stored procedures.
- Access to the .NET Framework tax components must be available to T-SQL objects.
- Columns must be defined by using data types that minimize the amount of space used by each table.
- Invoices stored in the InvoiceStatus table must refer to an invoice by the same identifier used by

the Invoice table.
▪ To protect against the theft of backup disks, invoice data must be protected by using the highest level of encryption.
▪ The solution must provide a table-valued function that provides users with the ability to filter invoices by customer.
▪ Indexes must be optimized periodically based on their fragmentation by using the minimum amount of administrative effort.

Usp_InsertInvoices.sql

```
01 CREATE PROCEDURE InsertInvoice @XML nvarchar(1000)
02 AS
03 DECLARE @XmlDocumentHandle INT;
04 DECLARE @XmlDocument nvarchar(1000);
05 SET @XmlDocument = @XML;
06
07 EXEC sp_xml_preparedocument @XmlDocumentHandle OUTPUT, @XmlDocument;
08
09 INSERT INTO DB1.Accounting.Invoices (
10   InvoiceID,
11   InvoiceXML,
12   CustomerID,
13   CustomerName,
14   InvoiceDate,
15   Total,
16   PONumber
17 )
18 SELECT (NEXT VALUE FOR Accounting.InvoiceID_Seq),
19   @XML, * FROM OPENXML (@XmlDocumentHandle, '/Invoice',2)
20   WITH (
21     CustomerID nvarchar(11) 'Customer/@ID',
22     CustomerName nvarchar(50) 'Customer/@Name',
23     InvoiceDate date 'InvoiceDate',
24     Total decimal(8, 2) 'Total',
25     PONumber bigint 'PONumber'
26   );
27
28 EXEC sp_xml_removedocument @XmlDocumentHandle;
```

Invoices.xml

All customer IDs are 11 digits. The first three digits of a customer ID represent the customer's country. The remaining eight digits are the customer's account number.

The following is a sample of a customer invoice in XML format:

```
01 <?xml version="1.0"?>
02 <Invoice InvoiceDate="2012-02-20">
03   <Customer ID="00156590099" Name="Litware" />
04   <Total>125</Total>
05   <PONumber>1666</PONumber>
06 </Invoice>
```

InvoicesByCustomer.sql

```
01 (SELECT CustomerID,
02   CustomerName,
03   InvoiceID,
04   InvoiceDate,
05   Total,
06   PONumber
07   FROM Accounting.Invoices
08   WHERE CustomerID=@CustID);
```

Legacy.sql

```
01 DECLARE @sqlstring AS nvarchar(1000);
02 DECLARE @CustomerID AS varchar(11), @Total AS decimal(8,2);
03
04 SET @sqlstring=N'SELECT CustomerID, InvoiceID, Total
05   FROM Accounting.Invoices
06   WHERE CustomerID=@CustomerID AND Total > @Total;';
07
08 EXEC sys.sp_executesql
09   @statement=@sqlstring,
10   @params=N'@CustomerID AS varchar(11), @Total AS decimal(8,2)',
11   @CustomerID=999, @Total=500;
```

CountryFromID.sql

```
01 CREATE FUNCTION CountryFromID (@CustomerID varchar(11)) RETURNS varchar(20)
02 AS
03 BEGIN
04   DECLARE @Country varchar(20);
05   SET @CustomerID = LEFT(@CustomerID,3);
06   SELECT @Country = CASE @CustomerID
07     WHEN '001'
08       THEN 'United States'
09     WHEN '002'
10       THEN 'Spain'
11     WHEN '003'
12       THEN 'Japan'
13     WHEN '004'
14       THEN 'China'
15     WHEN '005'
16       THEN 'Brazil'
17     ELSE 'Other'
18   END;
19   RETURN @CustomerID;
20 END;
```

IndexManagement.sql

```
01 DECLARE @IndexTable TABLE (
02   TableName varchar(100), IndexName varchar(100), Fragmentation int, RowNumber int
03   );
04 DECLARE @TableName sysname, @IndexName sysname, @Fragmentation int,
05   @RowNumber int, @sqlcommand varchar(1000);
06
07 INSERT INTO @IndexTable (TableName, IndexName, Fragmentation, Rownumber)
08   SELECT OBJECT_NAME(i.Object_id),
09     i.name AS IndexName,
10     indexstats.avg_fragmentation_in_percent,
11     ROW_NUMBER() OVER(ORDER BY i.name DESC) AS 'RowNumber'
12   FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, 'DETAILED')
13     AS indexstats INNER JOIN sys.indexes AS i
14     ON i.OBJECT_ID = indexstats.OBJECT_ID AND i.index_id = indexstats.index_id;
15
16 DECLARE @counter int = 0;
17
18 WHILE @counter < (SELECT RowNumber FROM @indextable)
19   BEGIN
20     SET @counter = @counter + 1;
21     WITH t AS (
22       SELECT TableName, IndexName, Fragmentation
23       FROM @IndexTable WHERE RowNumber = @counter
24       )
25     SELECT
26       @TableName= TableName,
27       @IndexName = IndexName,
28       @Fragmentation = Fragmentation
29     FROM t;
30
31     IF @Fragmentation <= 30
32       BEGIN
33         SET @sqlCommand =
34           N'ALTER INDEX '+@indexName+N' ON '+@TableName+N' REORGANIZE';
35         EXEC sp_executesql @sqlCommand;
36       END;
37     ELSE
38       BEGIN
39         SET @sqlCommand=N'ALTER INDEX '+@indexName+N' ON '+@TableName+N' REBUILD';
40         EXEC sp_executesql @sqlCommand;
41       END;
42     END;
```

QUESTION 1
You are testing disaster recovery procedures.

You attempt to restore DB1 to a different server and you receive the following error message: "Msg 33111.

Level 16, State 3, Line 1

Cannot find server certificate with thumbprint

,0xA694FBEA88C9354E5E2567C30A2A69E8FB4C44A9\

Msg 3013, Level 16, State 1, Line 1

RESTORE DATABASE is terminating abnormally."

You need to ensure that you can restore DB1 to a different server.

Which code segment should you execute?

A.
```
RESTORE CERTIFICATE CERT2
  FROM FILE='CERT2.CER'
  WITH PRIVATE KEY (FILE = 'CERT2.KEY',
  DECRYPTION BY PASSWORD='p@ssw0rd1');
```

B.
```
CREATE CERTIFICATE CERT1
  FROM FILE='CERT1.CER'
  WITH PRIVATE KEY (FILE = 'CERT1.KEY',
  DECRYPTION BY PASSWORD='p@ssw0rd1');
```

C.
```
CREATE CERTIFICATE CERT2
  ENCRYPTION BY PASSWORD='p@ssw0rd1'
  WITH SUBJECT = 'EncryptionCertificate';
```

D.
```
CREATE CERTIFICATE CERT1
  ENCRYPTION BY PASSWORD='p@ssw0rd1'
  WITH SUBJECT = 'EncryptionCertificate';
```

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

Correct Answer: B

QUESTION 2
You need to create the InvoiceStatus table in DB1. How should you define the InvoiceID column in the CREATE TABLE statement?

A.
```
InvoiceID bigint
DEFAULT (NEXT VALUE FOR Accounting.InvoiceID_Seq) NOT NULL,
```

B.
```
InvoiceID bigint DEFAULT ((NEXT VALUE
FOR Accounting.InvoiceID_Seq OVER
(ORDER BY InvoiceStatusID))) NOT NULL FOREIGN
KEY REFERENCES Accounting.Invoices(InvoiceID),
```

C.
```
InvoiceID bigint FOREIGN KEY REFERENCES
Accounting.Invoices(InvoiceID) NOT NULL,
```

D.
```
InvoiceID bigint DEFAULT ((NEXT VALUE
FOR Accounting.InvoiceID_Seq
OVER (ORDER BY InvoiceStatusID))) NOT NULL,
```

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

Correct Answer: C

QUESTION 3
Which data type should you use for CustomerID?

A. varchar(11)
B. bigint
C. nvarchar(11)
D. char(11)

Correct Answer: D
Explanation:
Invoices.xml
All customer IDs are 11 digits. The first three digits of a customer ID represent the customer's country. The remaining eight digits are the customer's account number. int: -2^31 (-2,147,483,648) to 2^31-1 (2,147,483,647) (just 10 digits max) bigint: -2^63 (-9,223,372,036,854,775,808) to 2^63-1 (9,223,372,036,854,775,807)
http://msdn.microsoft.com/en-us/library/ms176089.aspx http://msdn.microsoft.com/en-us/library/ms187745.aspx

QUESTION 4
You need to modify InsertInvoice to comply with the application requirements. Which code segment should you execute?

A.
```
OPEN CERT1;
ALTER PROCEDURE Accounting.usp_InsertInvoice
WITH ENCRYPTION;
CLOSE CERT1;
```

B.
```
OPEN CERT2;
ALTER PROCEDURE Accounting.usp_InsertInvoice
WITH ENCRYPTION;
CLOSE CERT2;
```

C.
```
ADD SIGNATURE TO Accounting.usp_InsertInvoice
BY CERTIFICATE CERT1;
```

D.
```
ADD SIGNATURE TO Accounting.usp_InsertInvoice
BY CERTIFICATE CERT2;
```

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

Correct Answer: D

QUESTION 5
You attempt to process an invoice by using usp_InsertInvoice.sql and you receive the following error message:

"Msg 515, Level 16, State 2, Procedure usp_InsertInvoice, Line Cannot insert the value NULL into column 'InvoiceDate', table 'DB1.Accounting.Invoices'; column does not allow nulls. INSERT fails."

You need to modify usp_InsertInvoice.sql to resolve the error. How should you modify the INSERT statement?

A. InvoiceDate varchar(l00) 'InvoiceDate',
B. InvoiceDate varchar(100) 'Customer/InvoiceDate', '
C. InvoiceDate date '@InvoiceDate',
D. InvoiceDate date 'Customer/@InvoiceDate',

Correct Answer: C

QUESTION 6
You need to modify the function in CountryFromID.sql to ensure that the country name is returned instead of the country ID. Which line of code should you modify in CountryFromID.sql?

A. 04
B. 05
C. 06
D. 19

Correct Answer: D
Explanation:
http://msdn.microsoft.com/en-us/library/ms186755.aspx
http://msdn.microsoft.com/en-us/library/ms191320.aspx

QUESTION 7
You execute IndexManagement.sql and you receive the following error message:

"Msg 512, Level 16, State 1, Line 12

Subquery returned more than 1 value. This is not permitted when the subquery follows =,! =, <, <= ,>, > = or when the subquery is used as an expression."

You need to ensure that IndexManagement.sql executes properly.

Which WHILE statement should you use at line 18?

A. WHILE SUM(@RowNumber) < (SELECT @counter FROM @indextable)
B. WHILE @counter < (SELECT COUNT(RowNumber) FROM @indextable)
C. WHILE COUNT(@RowNumber) < (SELECT @counter FROM @indextable)
D. WHILE @counter < (SELECT SUM(RowNumber) FROM @indextabie)

Correct Answer: B

QUESTION 8
You need to convert the functionality of Legacy.sql to use a stored procedure. Which code segment should the stored procedure contain?

A.
```
CREATE PROC usp_InvoicesByCustomerAboveTotal(
  @sqlstring AS nvarchar(1000),
  @CustomerID AS char(11),
  @Total AS decimal(8,2))
AS
...
```

B.
```
CREATE PROC usp_InvoicesByCustomerAboveTotal(
  @sqlstring AS nvarchar(1000))
AS
...
```

C.
```
CREATE PROC usp_InvoicesByCustomerAboveTotal(
  @sqlstring AS nvarchar(1000),
  OUTPUT @CustomerID AS char(11),
  OUTPUT @Total AS decimal(8,2))
AS
...
```

D.
```
CREATE PROC usp_InvoicesByCustomerAboveTotal (
  @CustomerID AS char(11), @Total AS decimal(8,2))
AS
...
```

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

Correct Answer: D
Explanation:
http://msdn.microsoft.com/en-us/library/ms187926.aspx
http://msdn.microsoft.com/en-us/library/ms190782.aspx
http://msdn.microsoft.com/en-us/library/bb669091.aspx
http://msdn.microsoft.com/en-us/library/windows/desktop/ms709342.aspx
http://msdn.microsoft.com/en-us/library/ms188001.aspx

QUESTION 9
You need to create a function that filters invoices by CustomerID. The SELECT statement for the function is contained in InvoicesByCustomer.sql. Which code segment should you use to complete the function?

A.
```
CREATE FUNCTION Accounting.fnInvoicesByCustomertest (@CustID varchar(11))
RETURNS @TblInvoices TABLE (CustomerID bigint, CustomerName NVARCHAR(255),
InvoiceID bigint,InvoiceDate date, Total decimal(8,2), PONumber bigint)
AS
```

B.
```
CREATE FUNCTION Accounting.fnInvoicesByCustomer (@CustID varchar(11))
RETURNS @tblInvoices TABLE (CustomerID bigint, CustomerName NVARCHAR(255),
InvoiceID bigint,InvoiceDate date, Total decimal(8,2), PONumber bigint)
AS
INSERT INTO @tblInvoices
```

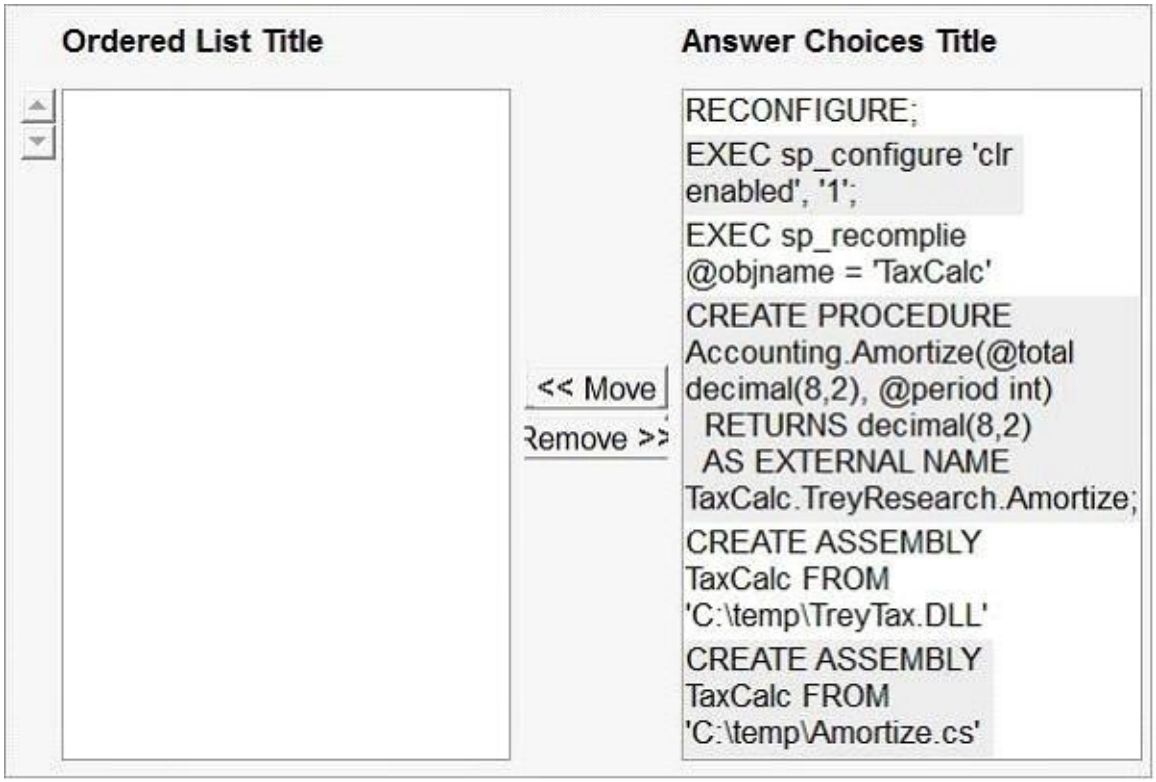
C.
```
CREATE FUNCTION Accounting.fnInvoicesByCustomer (@CustID varchar(11))
RETURNS xml
AS
RETURN
```

D.
```
CREATE FUNCTION Accounting.fnInvoicesByCustomertest (@CustID varchar(11))
RETURNS @TblInvoices TABLE (CustomerID bigint, CustomerName NVARCHAR(255),
InvoiceID bigint,InvoiceDate date, Total decimal(8,2), PONumber bigint)
AS
```

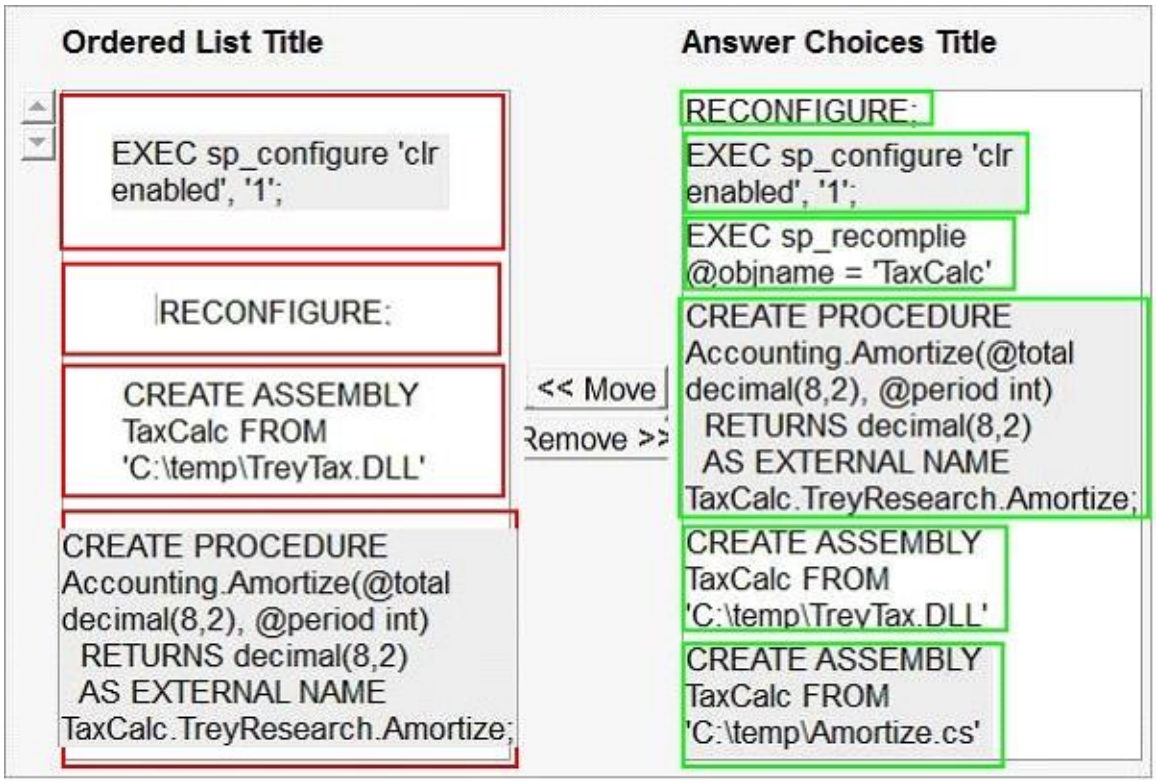
A. Option A
B. Option B
C. Option C
D. Option D

Correct Answer: A

QUESTION 10
DRAG DROP
You need to build a stored procedure that amortizes the invoice amount. Which code segment should you use to create the stored procedure? To answer, move the appropriate code segments from the list of code segments to the answer area and arrange them in the correct order.



Correct Answer:



Topic 2, Scenario 2

Application Information
You have two servers named SQL1 and SQL2 that have SQL Server 2012 installed.

You have an application that is used to schedule and manage conferences.

Users report that the application has many errors and is very slow.

You are updating the application to resolve the issues.

You plan to create a new database on SQL1 to support the application. A junior database administrator has created all the scripts that will be used to create the database. The script that you plan to use to create the tables for the new database is shown in Tables.sql. The script that you plan to use to create the stored procedures for the new database is shown in StoredProcedures.sql. The script that you plan to use to create the indexes for the new database is shown in Indexes.sql. (Line numbers are included for reference only.)

A database named DB2 resides on SQL2. DB2 has a table named SpeakerAudit that will audit changes to a table named Speakers.

A stored procedure named usp_UpdateSpeakersName will be executed only by other stored procedures. The stored procedures executing usp_UpdateSpeakersName will always handle transactions.

A stored procedure named usp_SelectSpeakersByName will be used to retrieve the names of speakers. Usp_SelectSpeakersByName can read uncommitted data.

A stored procedure named usp_GetFutureSessions will be used to retrieve sessions that will occur in the future.

Procedures.sql

```
01 CREATE PROCEDURE usp_UpdateSpeakerName
02   @SpeakerID int,
03   @LastName nvarchar(100)
04 AS
05
06 BEGIN TRY
07
08 UPDATE Speakers
09 SET LastName = @LastName
10 WHERE SpeakerID = @SpeakerID;
11
12 INSERT INTO SQL2.DB2.dbo.SpeakerAudit(SpeakerID, LastName)
13 VALUES (@SpeakerID, @LastName);
14
15 END TRY
16 BEGIN CATCH
17
18 END CATCH;
19
20 GO
21
22 CREATE PROCEDURE usp_SelectSpeakersByName
23   @LastName nvarchar(100)
24 AS
25 SELECT SpeakerID,
26   FirstName,
27   LastName
28 FROM Speakers
29 WHERE LastName LIKE @LastName + '%'
30
31 GO
32
33 CREATE PROCEDURE usp_InsertSessions
34   @SessionData SessionDataTable READONLY
35 AS
36 INSERT INTO Sessions
37   (SpeakerID, Title, Absract, DeliveryTime, TitleAndSpeaker)
38 SELECT SpeakerID, Title, Absract, DeliveryTime, TitleAndSpeaker
39 FROM @SessionData;
40 GO
41
42 CREATE PROCEDURE usp_UpdateSessionRoom
43   @RoomID int,
44   @SpeakerID int
45 AS
```

```
46 SET TRANSACTION ISOLATION LEVEL SNAPSHOT
47 BEGIN TRANSACTION;
48
49 SELECT SessionID,
50   Title
51 FROM Sessions
52 WHERE SpeakerID = @SpeakerID;
53
54 UPDATE Sessions
55 SET RoomID = @RoomID
56 WHERE SpeakerID = @SpeakerID;
57
58 COMMIT TRANSACTION;
59
60 CREATE PROCEDURE usp_AttendeesReport
61   @LastName varchar(100)
62 AS
63 SELECT FirstName + ' ' + LastName AS FullName
64 FROM Attendees
65 WHERE LastName = @LastName;
66 GO
67
68 CREATE PROCEDURE usp_GetFutureSessions
69 AS
70 SELECT SpeakerID,
71   RoomID,
72   DeliveryTime
73 FROM Sessions
74
75 GO
76
77 CREATE PROCEDURE usp_TestSpeakers
78 AS
79 EXECUTE usp_SelectSpeakersByName 'a';
80 EXECUTE usp_SelectSpeakersByName 'an';
81 EXECUTE usp_SelectSpeakersByName 'and';
82 EXECUTE usp_SelectSpeakersByName 'ander';
83 EXECUTE usp_SelectSpeakersByName 'anderson';
84 EXECUTE usp_SelectSpeakersByName 'b';
85 EXECUTE usp_SelectSpeakersByName 'bi';
86 ...
87 EXECUTE usp_SelectSpeakersByName 'zzz';
88 GO
```

Indexes.sql

```
01 CREATE INDEX IX_Sessions ON Sessions
02 (SessionID, DeliveryTime)
03 INCLUDE (RoomID)
04
05 GO
06
07 CREATE INDEX IX_Speakers ON Speakers
08 (LastName);
09 GO
10
11 CREATE INDEX IX_Attendees_Name ON Attendees
12 (FirstName, LastName);
13
14 GO
15
16 CREATE INDEX IX_Attendees_Confirmed ON Attendees
17 (Confirmed);
18 GO
```

Tables.sql

```
01 CREATE DATABASE Conference;
02 GO
03
04 ALTER DATABASE Conference
05 SET READ_COMMITTED_SNAPSHOT ON;
06 GO
07
08 CREATE TABLE Attendees
09 (
10   AttendeeID int IDENTITY (1,1) NOT NULL,
11    FirstName nvarchar(100) NOT NULL,
12    LastName nvarchar(100) NOT NULL,
13    EmailAddress nvarchar(100) NOT NULL,
14
15    CONSTRAINT PK_Attendees_AttendeeID PRIMARY KEY (AttendeeID)
16 );
17 GO
18
19 CREATE TABLE Speakers
20 (
21   SpeakerID int IDENTITY(1,1) NOT NULL,
22    FirstName nvarchar(100) NOT NULL,
23    LastName nvarchar(100) NOT NULL,
24    Photo varbinary(max),
25    CONSTRAINT PK_Speakers_SpeakerID PRIMARY KEY (SpeakerID)
26 );
27 GO
28
29 CREATE TABLE Sessions
30 (
31   SessionID uniqueidentifier NOT NULL
32     CONSTRAINT DF_SessionID DEFAULT (NEWID()),
33    SpeakerID int NOT NULL,
34    Title nvarchar(100) NOT NULL,
35    Abstract nvarchar(max) NOT NULL,
36    DeliveryTime datetime NOT NULL,
37    TitleAndSpeaker nvarchar(200)
38
39 );
40 GO
41
42 CREATE TABLE Rooms
43 (
44    RoomID uniqueidentifier NOT NULL CONSTRAINT DF_RoomID DEFAULT (NEWID()),
45     Location varchar(100) NOT NULL
46 );
```

QUESTION 11
You need to provide referential integrity between the Sessions table and Speakers table. Which code segment should you add at line 47 of Tables.sql?

A.
```
ALTER TABLE dbo.Sessions ADD CONSTRAINT
FK_Sessions_Speakers FOREIGN KEY (SessionID)
REFERENCES dbo.Speakers (SpeakerID);
```

B.
```
ALTER TABLE dbo.Sessions ADD CONSTRAINT
FK_Sessions_Speakers FOREIGN KEY (SpeakerID)
REFERENCES dbo.Speakers (SpeakerID);
```

C.
```
ALTER TABLE dbo.Speakers ADD CONSTRAINT
FK_Speakers_Sessions FOREIGN KEY (SpeakerID)
REFERENCES dbo.Sessions (SessionID);
```

D.
```
ALTER TABLE dbo.Speakers ADD CONSTRAINT
FK_Speakers_Sessions FOREIGN KEY (SessionID)
REFERENCES dbo.Sessions (SessionID);
```

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

Correct Answer: B
Explanation:
http://msdn.microsoft.com/en-us/library/ms189049.aspx
http://msdn.microsoft.com/en-us/library/ms179610.aspx
http://msdn.microsoft.com/en-us/library/ff878370.aspx

QUESTION 12
You execute usp_TestSpeakers. You discover that usp_SelectSpeakersByName uses inefficient execution plans. You need to update usp_SelectSpeakersByName to ensure that the most efficient execution plan is used. What should you add at line 30 of Procedures.sql?

A. OPTION (FORCESCAN)
B. OPTION (FORCESEEK)
C. OPTION (OPTIMIZE FOR UNKNOWN)
D. OPTION (OPTIMIZE FOR (@LastName= 'Anderson'))

Correct Answer: C
Explanation:
http://msdn.microsoft.com/en-us/library/ms181714.aspx

QUESTION 13
You need to recommend a solution to ensure that SQL1 supports the auditing requirements of usp_UpdateSpeakerName. What should you include in the recommendation?

A. The Distributed Transaction Coordinator (DTC)
B. Transactional replication
C. Change data capture
D. Change tracking

Correct Answer: A

QUESTION 14
You are evaluating the table design. You need to recommend a change to Tables.sql that reduces the amount of time it takes for usp_AttendeesReport to execute. What should you add at line 14 of Tables.sql?

A. FullName nvarchar(100) NOT NULL CONSTRAINT DF_FullName DEFAULT (dbo.CreateFullName (FirstName, LastName)),
B. FullName AS (FirstName +` '+ LastName),
C. FullName nvarchar(100) NOT NULL DEFAULT (dbo.CreateFullName (FirstName, LastName)).
D. FullName AS (FirstName +` '+ LastName) PERSISTED,

Correct Answer: D
Explanation:
http://msdn.microsoft.com/en-us/library/ms188300.aspx
http://msdn.microsoft.com/en-us/library/ms191250.aspx

QUESTION 15
You need to modify usp_SelectSpeakersByName to support server-side paging. The solution must minimize the amount of development effort required. What should you add to usp_SelectSpeakersByName?

A. A table variable
B. An OFFSET-FETCH clause
C. The ROWNUMBER keyword
D. A recursive common table expression

Correct Answer: B
Explanation:
http://www.mssqltips.com/sqlservertip/2696/comparing-performance-for-different-sql-serverpaging-methods/
http://msdn.microsoft.com/en-us/library/ms188385.aspx
http://msdn.microsoft.com/en-us/library/ms180152.aspx
http://msdn.microsoft.com/en-us/library/ms186243.aspx
http://msdn.microsoft.com/en-us/library/ms186734.aspx
http://www.sqlserver-training.com/how-to-use-offset-fetch-option-in-sql-server-order-byclause/-
http://www.sqlservercentral.com/blogs/juggling_with_sql/2011/11/30/using-offset-and-fetch/

QUESTION 16
You need to add a new column named Confirmed to the Attendees table. The solution must meet the following requirements:

- Have a default value of false.
- Minimize the amount of disk space used.

Which code block should you use?

A. ALTER TABLE Attendees
ADD Confirmed bit DEFAULT 0;
B. ALTER TABLE Attendees
ADD Confirmed char(l) DEFAULT '1';
C. ALTER TABLE Attendees
ADD Confirmed bit DEFAULT 1;
D. ALTER TABLE Attendees
ADD Confirmed char(l) DEFAULT `1';

Correct Answer: A
Explanation:
http://msdn.microsoft.com/en-us/library/ms177603.aspx

QUESTION 17
You need to create the object used by the parameter of usp_InsertSessions. Which statement should you use?

A. CREATE XML SCHEMA COLLECTION SessionDataTable
B. CREATE TYPE SessionDataTable AS Table
C. CREATE SCHEMA SessionDataTable
D. CREATE TABLE SessionDataTable

Correct Answer: B

QUESTION 18
Developers report that usp_UpdateSessionRoom periodically returns error 3960. You need to prevent the error from occurring. The solution must ensure that the stored procedure returns the original values to all of the updated rows. What should you configure in Procedures.sql?

A. Replace line 46 with the following code:
SET TRANSACTION ISOLATION LEVEL SERIALIZABLE
B. Replace line 46 with the following code:
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ
C. Move the SELECT statement at line 49 to line 57.
D. Move the SET statement at line 46 to line 53.

Correct Answer: A

QUESTION 19
You discover that usp.SelectSpeakersByName executes slowly if usp_UpdateSpeakerName executes simultaneously. You need to minimize the execution time of usp.SelectSpeakersByName. The solution must not affect the performance of the other stored procedures. What should you update?

A. Usp_UpdateSpeakerName to use the NOLOCK query hint
B. Usp_UpdateSpeakerName to use snapshot isolation
C. Usp_SelectSpeakersByName to use the NOLOCK query hint
D. Usp_SelectSpeakersByName to use snapshot isolation

Correct Answer: C
Explanation:
NOLOCK
Is equivalent to READUNCOMMITTED.
READUNCOMMITTED
Specifies that dirty reads are allowed.

QUESTION 20
While testing usp.GetFutureSessions, you discover that IX_Sessions is accessed by a scan rather than a seek. You need to minimize the amount of time it takes to execute usp_GetFutureSessions. What should you do? (Each correct answer presents part of the solution. Choose all that apply.)

☐ A. Change line 02 of Indexes.sql to:

```
(DeliveryTime, SessionID)
```

☐ B. At line 04 of Indexes.sql, add:

```
WHERE GETDATE() < DeliveryTime;
```

☐ C. Change line 02 of Indexes.sql to:

```
(SpeakerID, RoomID, DeliveryTime)
```

☐ D. Change line 74 of Procedures.sql to:

```
WHERE GETDATE() > DeliveryTime;
```

☐ E. Change line 74 of Procedures.sql to:

```
WHERE GETDATE() < DeliveryTime;
```

☐ F. At line 04 of Indexes.sql, add:

```
WHERE GETDATE() > DeliveryTime;
```

A. Option A
B. Option B

C. Option C
D. Option D
E. Option E
F. Option F

Correct Answer: BE
Explanation:
Future delivery dates.

QUESTION 21
You need to ensure that if any of the statements in usp_UpdateSpeakerName return an error message, all of the changes executed by usp_UpdateSpeakerName are not committed to the database. What should you do in Procedures.sql? (Each correct answer presents part of the solution. Choose all that apply.)

☐ A. Add the following at line 17:

```
ROLLBACK TRANSACTION
```

☐ B. Add the following at line 05:

```
BEGIN TRANSACTION SpeakerUpdate
```

☐ C. Add the following at line 05:

```
SAVE TRANSACTION SpeakerUpdate
```

☐ D. Add the following at line 17:

```
ROLLBACK TRANSACTION SpeakerUpdate
```

☐ E. Add the following at line 07:

```
BEGIN TRANSACTION
```

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

Correct Answer: BD

QUESTION 22
You are evaluating the index design. You need to recommend a change to Indexes.sql that will minimize the amount of time it takes for usp_AttendeesReport to execute. The solution must minimize the amount of database fragmentation. Which line of code should you use to replace line 12 of Indexes.sql?

A. (LastName);
B. (FirstName) INCLUDE (LastName);
C. (LastName, FirstName);
D. (LastName) INCLUDE (FirstName);

Correct Answer: C

QUESTION 23
You need to create the object used by the parameter of usp_InsertSessions. Which statement should you use?

A. CREATE SCHEMA SessionDataTable
B. CREATE TYPE SessionDataTable AS Table
C. CREATE TABLE SessionDataTable
D. CREATE XML SCHEMA COLLECTION SessionDataTable

Correct Answer: A

Topic 3, Scenario 3

Application Information
You have two servers named SQL1 and SQL2. SQL1 has SQL Server 2012 Enterprise installed. SQL2 has SQL Server 2008 Standard installed.

You have an application that is used to manage employees and office space. Users report that the application has many errors and is very slow.

You are updating the application to resolve the issues. You plan to create a new database on SQL1 to support the application. The script that you plan to use to create the tables for the new database is shown in Tables.sql. The script that you plan to use to create the stored procedures for the new database is shown in StoredProcedures.sql. The script that you plan to use to create the indexes for the new database is shown in Indexes.sql.
A database named DB2 resides on SQL2. DB2 has a table named EmployeeAudit that will audit changes to a table named Employees.

A stored procedure named usp_UpdateEmployeeName will be executed only by other stored procedures. The stored procedures executing usp_UpdateEmp!oyeeName will always handle transactions.

A stored procedure named usp_SelectEmployeesByName will be used to retrieve the names of employees. Usp_SelectEmployeesByName can read uncommitted data.

A stored procedure named usp_GetFutureOfficeAssignments will be used to retrieve office assignments that will occur in the future.

StoredProcedures.sql

```
01 CREATE PROCEDURE usp_UpdateEmployeeName
02   @EmployeesInfo EmployeesInfo READONLY
03 AS
04
05 BEGIN TRY
06
07 UPDATE Employees
08 SET LastName = ei.LastName
09 FROM Employees e
10   INNER JOIN @ EmployeesInfo ei ON e.EmployeeID = ei.EmployeeID;
11
12 INSERT INTO SQL2.DB2.dbo.EmployeeAudit(EmployeeID, LastName)
13 SELECT EmployeeID, LastName
14 FROM @EmployeesInfo;
15
16 END TRY
17 BEGIN CATCH
18
19 END CATCH;
20
21 GO
22
23 CREATE PROCEDURE usp_SelectEmployeesByName
24   @LastName nvarchar(100)
25 AS
26 SELECT EmployeeID,
27   FirstName,
28   LastName
29 FROM Employees
30 WHERE LastName LIKE @LastName + '%'
31
32 GO
33
34 CREATE PROCEDURE usp_UpdateOffice
35   @OfficeID int,
36   @EmployeeID int
37 AS
38 SET TRANSACTION ISOLATION LEVEL SNAPSHOT
39 BEGIN TRANSACTION;
40
41 SELECT OfficeID,
42   OfficeName
43 FROM Offices
44 WHERE EmployeeID = @EmployeeID;
45
46 UPDATE Offices
47 SET EmployeeID = @EmployeeID,
48   StartDate = GETDATE()
49 WHERE OfficeID = @OfficeID;
50
51 COMMIT TRANSACTION;
52
53 CREATE PROCEDURE usp_GetFutureOfficeAssignments
54 AS
55 SELECT EmployeeID,
56   OfficeID,
57   StartDate
58 FROM Offices
59 WHERE StartDate > GETDATE();
60 GO
61
```

Indexes.sql

```
01 CREATE INDEX IX_Offices ON Offices
02 (EmployeeID, StartDate)
03 INCLUDE (OfficeID)
04
05 GO
06
07 CREATE INDEX IX_Employees ON Employees
08 (LastName);
09 GO
10
```

Tables.sql

```
01 CREATE DATABASE HumanResources;
02 GO
03
04 ALTER DATABASE HumanResources
05 SET ALLOW_SNAPSHOT_ISOLATION ON;
06 GO
07
08 USE HumanResources
09 GO
10
11 CREATE TABLE Employees
12 (
13   EmployeeID int IDENTITY(1,1) NOT NULL,
14    FirstName nvarchar(100) NOT NULL,
15    LastName nvarchar(100) NOT NULL,
16
17 );
18 GO
19
20 CREATE TABLE Offices
21 (
22   OfficeID int IDENTITY(1,1) NOT NULL,
23    EmployeeID int NOT NULL,
24    OfficeName nvarchar(100) NOT NULL,
25    StartDate datetime NOT NULL
26 );
27 GO
```

QUESTION 24
You execute usp_SelectEmployeesByName multiple times, passing strings of varying lengths to @LastName. You discover that usp_SelectEmployeesByName uses inefficient execution plans. You need to update usp_SelectEmployeesByName to ensure that the most efficient execution plan is used. What should you add at line 31 of StoredProcedures.sql?

A. OPTION (ROBUST PLAN)
B. OPTION (OPTIMIZE FOR UNKNOWN)
C. OPTION (KEEP PLAN)
D. OPTION (KEEPFIXED PLAN)

Correct Answer: B
Explanation:
http://msdn.microsoft.com/en-us/library/ms181714.aspx

QUESTION 25
You need to recommend a solution to ensure that SQL1 supports the auditing requirements of usp_UpdateEmployeeName. What should you include in the recommendation?

A. Change data capture
B. Change tracking
C. Transactional replication
D. The Distributed Transaction Coordinator (DTC)

Correct Answer: D

QUESTION 26
You need to add a new column named Confirmed to the Employees table. The solution must meet the following requirements:

- Have a default value of TRUE.
- Minimize the amount of disk space used.

Which code segment should you use?

A.
```
ALTER TABLE Employees
ADD Confirmed char(1) DEFAULT '1';
```

B.
```
ALTER TABLE Employees
ADD Confirmed char(1) DEFAULT '0';
```

C.
```
ALTER TABLE Employees
ADD Confirmed bit DEFAULT 0;
```

D.
```
ALTER TABLE Employees
ADD Confirmed bit DEFAULT 1;
```

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

Correct Answer: D

QUESTION 27
You need to create the object used by the parameter of usp_UpdateEmployeeName. Which code segment should you use?

A. CREATE XML SCHEMA COLLECTION EmployeesInfo
B. CREATE TYPE EmployeesInfo AS Table
C. CREATE SCHEMA EmployeesInfo
D. CREATE TABLE EmployeesInfo

Correct Answer: B
Explanation:
Example Usage of Table-Valued Parameters (Database Engine) http://msdn.microsoft.com/en-us/library/bb510489.aspx (Benefits of using Table-Valued Parameters)

```
/* Create a table type. */
CREATE TYPE LocationTableType AS TABLE
( LocationName VARCHAR(50)
, CostRate INT );
GO
/* Create a procedure to receive data for the table-valued parameter. */ CREATE PROCEDURE dbo. usp_InsertProductionLocation @TVP LocationTableType READONLY
AS
SET NOCOUNT ON
INSERT INTO AdventureWorks2012.Production.Location (Name
,CostRate
,Availability
,ModifiedDate)
SELECT *, 0, GETDATE()
FROM @TVP;
GO
```

Also:
http://msdn.microsoft.com/en-us/library/ms175007.aspx(CREATE TYPE *tabletypename* AS TABLE)
http://msdn.microsoft.com/en-us/library/ms175010.aspx(table data types)

Wrong Answers:
http://msdn.microsoft.com/en-us/library/ms174979.aspx(CREATE TABLE)
http://msdn.microsoft.com/en-us/library/ms189462.aspx(CREATE SCHEMA)
http://msdn.microsoft.com/en-us/library/ms176009.aspx(CREATE XML SCHEMA COLLECTION)

QUESTION 28
You need to provide referential integrity between the Offices table and Employees table. Which code segment or segments should you add at line 27 of Tables.sql? (Each correct answer presents part of the solution. Choose all that apply.)

A.
```
ALTER TABLE dbo.Offices ADD CONSTRAINT
PK_Offices_EmployeeID PRIMARY KEY (EmployeeID);
```

B.
```
ALTER TABLE dbo.Employees ADD CONSTRAINT
FK_Employees_Offices FOREIGN KEY (OfficeID)
REFERENCES dbo.Offices (OfficeID);
```

C.
```
ALTER TABLE dbo.Employees ADD CONSTRAINT
PK_Employees_EmployeeID PRIMARY KEY (EmployeeID);
```

D.
```
ALTER TABLE dbo.Offices ADD CONSTRAINT
FK_Offices_Employees FOREIGN KEY (EmployeeID)
REFERENCES dbo.Employees (EmployeeID);
```

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

Correct Answer: CD
Explanation:http://msdn.microsoft.com/en-us/library/ms189049.aspx

QUESTION 29
You need to modify usp_SelectEmployeesByName to support server-side paging. The solution must minimize the amount of development effort required. What should you add to usp_SelectEmployeesByName?

A. A table variable
B. The ROWNUMBER keyword
C. An OFFSET-FETCH clause
D. A recursive common table expression

Correct Answer: C
Explanation:
http://www.mssqltips.com/sqlservertip/2696/comparing-performance-for-different-sql-serverpaging-methods/
http://msdn.microsoft.com/en-us/library/ms188385.aspx
http://msdn.microsoft.com/en-us/library/ms180152.aspx
http://msdn.microsoft.com/en-us/library/ms186243.aspx
http://msdn.microsoft.com/en-us/library/ms186734.aspx
http://www.sqlserver-training.com/how-to-use-offset-fetch-option-in-sql-server-order-byclause/-
http://www.sqlservercentral.com/blogs/juggling_with_sql/2011/11/30/using-offset-and-fetch/

Topic 4, Scenario 4

Application Information
You are a database administrator for a manufacturing company.

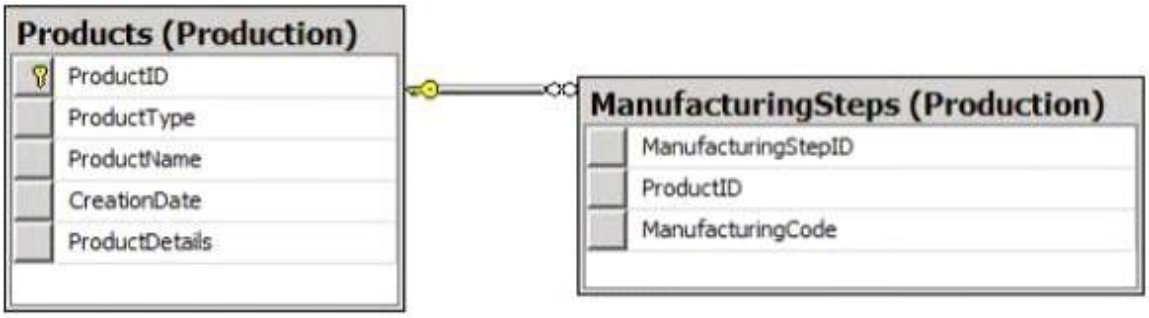
You have an application that stores product data. The data will be converted to technical diagrams for the manufacturing process.

The product details are stored in XML format. Each XML must contain only one product that has a root element named Product. A schema named Production.ProductSchema has been created for the products xml.

You develop a Microsoft .NET Framework assembly named ProcessProducts.dll that will be used to convert the XML files to diagrams. The diagrams will be stored in the database as images. ProcessProducts.dll contains one class named ProcessProduct that has a method name of Convert(). ProcessProducts.dll was created by using a source code file named ProcessProduct.cs.

All of the files are located in C:\Products\.
The application has several performance and security issues. You will create a new database named ProductsDB on a new server that has SQL Server 2012 installed. ProductsDB will support the application.

The following graphic shows the planned tables for ProductsDB:



You will also add a sequence named Production.ProductID_Seq.

You plan to create two certificates named DBCert and ProductsCert. You will create ProductsCert in master. You will create DBCert in ProductsDB.

You have an application that executes dynamic T-SQL statements against ProductsDB. A sample of the queries generated by the application appears in Dynamic.sql.

Application Requirements
The planned database has the following requirements:

- All stored procedures must be signed.
- The amount of disk space must be minimized.
- Administrative effort must be minimized at all times.
- The original product details must be stored in the database.
- An XML schema must be used to validate the product details.
- The assembly must be accessible by using T-SQL commands.
- A table-valued function will be created to search products by type.
- Backups must be protected by using the highest level of encryption.

- Dynamic T-SQL statements must be converted to stored procedures.
- Indexes must be optimized periodically based on their fragmentation.
- Manufacturing steps stored in the ManufacturingSteps table must refer to a product by the same identifier used by the Products table.

ProductDetails_Insert.sql

```
01 CREATE PROCEDURE Production.ProductDetails_Insert @XML nvarchar(1000)
02 AS
03 DECLARE @handle INT;
04 DECLARE @document nvarchar(1000);
05 SET @document = @XML;
06
07 EXEC sp_xml_preparedocument @handle OUTPUT, @document;
08
09 INSERT INTO PRODUCTSDB.Production.Invoices (
10   ProductID,
11   ProductDetails,
12   ProductType,
13   ProductName,
14   CreationDate
15 )
16 SELECT (NEXT VALUE FOR Production.ProductID_Seq),
17   @XML, * FROM OPENXML (@handle, '/Invoice',2)
18   WITH (
19     ProductType nvarchar(11) 'ProductType/ID',
20     ProductName nvarchar(50) '@ProductName',
21     CreationDate date 'CreationDate'
22   );
23
24 EXEC sp_xml_removedocument @handle;
```

Product.xml

All product types are 11 digits. The first five digits of the product id reference the category of the product and the remaining six digits are the subcategory of the product.

The following is a sample customer invoice in XML format:

```
01 <?xml version="1.0"?>
02 <Product ProductName="Widget">
03   <ProductType ID="00156590099" />
04   <CreationDate>2011-08-05</CreationDate>
05   </Invoice>
```

ProductsByProductType.sql

```
01 (SELECT ProductID,
02   ProductType,
03   CreationDate
04   FROM Production.Products
05     WHERE ProductType=@ProductType);
```

Dynamic.sql

```
01 DECLARE @tsql AS nvarchar(500);
02 DECLARE @ProductType AS varchar(11), @CreationDate AS date;
03
04 SET @sqlstring=N'SELECT ProductID, ProductType, CreationDate
05   FROM Production.Product
06   WHERE ProductID=@ProductID AND CreationDate > @CreationDate;';
07
08 EXEC sys.sp_executesql
09   @statement=@sqlstring,
10   @params=N'@ ProductType AS varchar(11), @CreationDate AS date',
11 @ProductType=00125061246, @Total='2012-05-10';
```

Category FromType.sql

```
01 CREATE FUNCTION CategoryFromType (@Type varchar(11)) RETURNS nvarchar(20)
02 AS
03 BEGIN
04   DECLARE @Category AS varchar(20);
05   SET @Category = LEFT(@Category,5);
06   SELECT @Category = CASE @Type
07     WHEN '00001'
08       THEN 'Bikes'
09     WHEN '00002'
10       THEN 'Wheels'
11     ...
12     ELSE 'Other'
13   END;
14 RETURN @Category;
15 END;
```

IndexManagement.sql

```
01 DECLARE @IndexTable TABLE (
02   TableName varchar(100), IndexName varchar(100), Fragmentation int, RowNumber int
03   );
04 DECLARE @TableName sysname, @IndexName sysname, @Fragmentation int,
05   @RowNumber int, @sqlcommand varchar(1000);
06
07 INSERT INTO @IndexTable (TableName, IndexName, Fragmentation, Rownumber)
08   SELECT OBJECT_NAME(i.Object_id),
09     i.name AS IndexName,
10     indexstats.avg_fragmentation_in_percent,
11     ROW_NUMBER() OVER(ORDER BY i.name DESC) AS 'RowNumber'
12   FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, 'DETAILED')
13     AS indexstats INNER JOIN sys.indexes AS i
14     ON i.OBJECT_ID = indexstats.OBJECT_ID AND i.index_id = indexstats.index_id;
15
16 DECLARE @counter int = 0;
17
18 WHILE @counter < (SELECT RowNumber FROM @indextable)
19   BEGIN
20     SET @counter = @counter + 1;
21     WITH t AS (
22       SELECT TableName, IndexName, Fragmentation
23       FROM @IndexTable WHERE RowNumber = @counter
24       )
25     SELECT
26       @TableName= TableName,
27       @IndexName = IndexName,
28       @Fragmentation = Fragmentation
29     FROM t;
30
31     IF @Fragmentation <= 30
32       BEGIN
33         SET @sqlCommand =
34           N'ALTER INDEX '+@indexName+N' ON '+@TableName+N' REORGANIZE';
35         EXEC sp_executesql @sqlCommand;
36       END;
37     ELSE
38       BEGIN
39         SET @sqlCommand=N'ALTER INDEX '+@indexName+N' ON '+@TableName+N' REBUILD';
40         EXEC sp_executesql @sqlCommand;
41       END;
42     END;
```

QUESTION 30
Which code segment should you use to define the ProductDetails column?

A. ProductDetails xml (DOCUMENT Production.ProductDetailsSchema) NULL
B. ProductDetails xml NULL
C. ProductDetails xml (CONTENT Production.ProductDetailsSchema) NULL
D. ProductDetails varchar(MAX) NULL

Correct Answer: D

EnsurePass.com Members Features:

1. Verified Answers researched by industry experts.
2. Q&As are downloadable in PDF and VCE format.
3. 98% success Guarantee and **Money Back** Guarantee.
4. Free updates for **180** Days.
5. **Instant Access to download the Items**

View list of All Exam provided:
http://www.ensurepass.com/certfications?index=A

To purchase Lifetime Full Access Membership click here:
http://www.ensurepass.com/user/register

Valid Discount Code for 2015: JREH-G1A8-XHC6

To purchase the HOT Microsoft Exams:

Microsoft			
70-243	70-347	70-466	70-515
70-246	70-410	70-467	70-516
70-247	70-411	70-480	70-519
70-321	70-412	70-483	70-583
70-331	70-413	70-484	70-640
70-332	70-414	70-485	70-649
70-336	70-417	70-486	70-668
70-337	70-461	70-487	70-680
70-341	70-462	70-488	70-687
70-342	70-463	70-489	70-688
70-346	70-464	70-513	70-689