Ourexam

Higher Quality Better Service!

We offer free update service for one year

Http://www.ourexam.com

Exam : **70-469**

Title : Recertification for MCSE: Data Platform

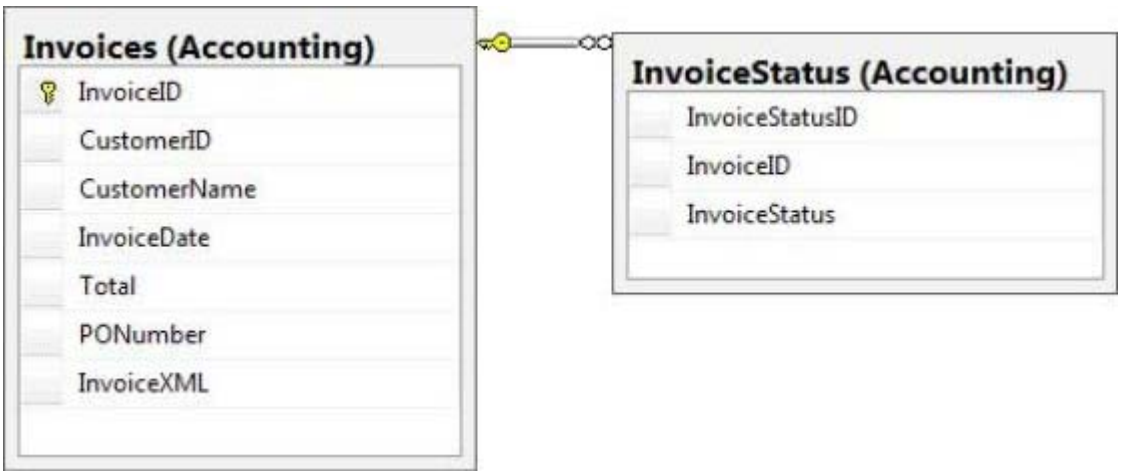
Version : Demo

1. 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 (@CurtomerID varcher(11)) RETURNS varchar(20)
02 AS
03 BEGIN
04   DECLARE @Country varchar(20);
05   SET @CustomerID = LEFT(@CurtomerID,3);
06   SELECT @Country = CASE @CurtomerID
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       @TableName = 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;
```

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 PRIVAIE KEY (FILE = 'CERT2.KEY",
  DECRYPTION BY PASSWORD='p@sswOrd1');
```

B.
```
RREATE CERTIFICATE CERT1
  FROM FILE='CERT2.CER'
  WITH PRIVAIE KEY (FILE = 'CERT1.KEY",
  DECRYPTION BY PASSWORD='p@sswOrd1');
```

C.
```
CREATE CERTIFICATE CERT2
  ENCRYPTION BY PASSWORD='p@sswOrd1'
  WITH SUBJECT = 'EncryptionCertificate';
```

D.
```
CREATE CERTIFICATE CERT1
  ENCRYPTION BY PASSWORD='p@sswOrd1'
  WITH SUBJECT = 'EncryptionCertificate';
```

A. Option A

B. Option B

C. Option C

D. Option D

Answer: B

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

Answer: C

3.Which data type should you use for CustomerID?

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

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
Answer: D

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 10
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',
Answer: C