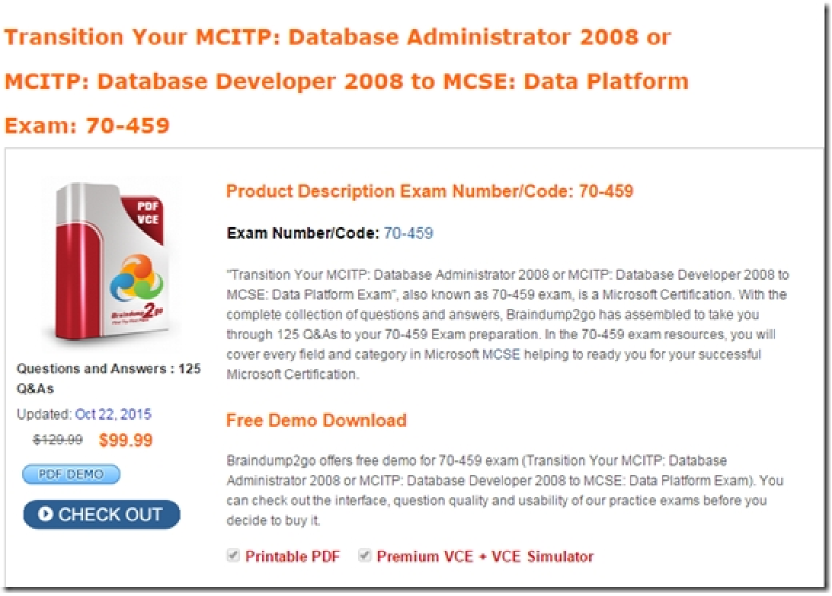
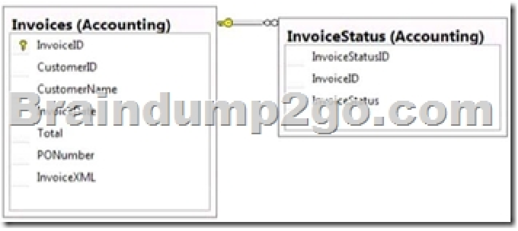
Braindump2go Helps Get Microsoft 70-459 Certification Easily By Latest 70-459 Exam Dumps (71-80)

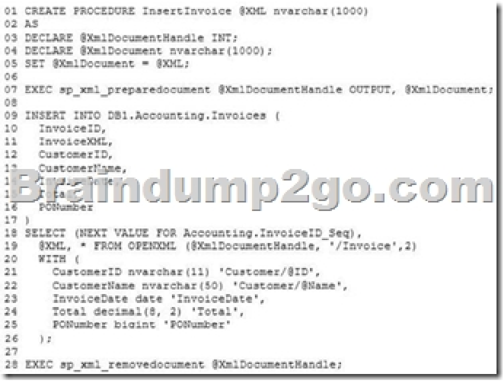
MICROSOFT NEWS: 70-459 Exam Questions has been Updated Today! Get Latest 70-459 VCE and 70-459 Book Instantly! Welcome to Download the Newest Braindump2go 70-459 VCE&70-459 PDF Dumps: http://www.braindump2go.com/70-459.html (125 Q&As) Are You Interested in Successfully Completing the Microsoft 70-459 Certification Then Start to Earning Salary? Braindump2go has Leading Edge Developed Microsoft Exam Questions that will Ensure You Pass this 70-459 Certification! Braindump2go Delivers you the Most Accurate, Current and Latest Updated 70-459 Certification Exam Questions Availabe with a 100% Money Back Guarantee Promise! Exam Code: 70-459Exam Name: Transition Your MCITP: Database Administrator 2008 or MCITP: Database Developer 2008 to MCSE: Data PlatformCertification Provider: MicrosoftCorresponding Certifications: MCP, MCSA, MCSE70-459 Dump,70-459 PDF,70-459 VCE,70-459 Braindump,70-459 Book,70-459 Certification,70-459 Exam Dumps,70-459 Exam Questions,70-459 eBook,70-459 Practice Exam,70-459 Practice Test,70-459 Preparation,70-459 Study Guide,70-459 Study Material,70-459 Training Kit



QUESTION 71You 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 variableB. an OFFSET-FETCH clauseC. the ROWNUMBER keywordD. a recursive common table expression Answer: BExplanation: http://www.mssqltips.com/sqlservertip/2696/comparing-performance-for-different-sql-server-paging-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-by-clause/- http://www.sqlservercentral.com/blogs/juggling_with_sql/2011/11/30/using-offset-and-fetch/ Case Study 7 - Invoice Scehma Scenario (QUESTION 72 ~ QUESTION 78)Application InformationYour 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:

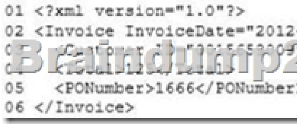


You have a sequence named Accounting.InvoiceID_Seq. You plan to create two certificates named CERT1 and CERT2. You will create CERT1 in master. You will create CERT2 in DB1.You have a legacy application that requires the ability to generate dynamic T-SQL statements against DB1. A sample of the queries generated by the legacy application appears in Legacy.sql.Application RequirementsThe 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



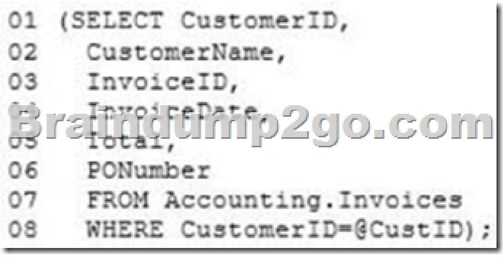
```
CREATE PROCEDURE InsertInvoice @XML nvarchar(1000)
AS
DECLARE @XmlDocumentHandle INT;
DECLARE @XmlDocument nvarchar(1000);
SET @XmlDocument = @XML;

EXEC sp_xml_preparedocument @XmlDocumentHandle OUTPUT, @XmlDocument;

INSERT INTO DB1.Accounting.Invoices (
  InvoiceID,
  InvoiceXML,
  CustomerID,
  CustomerName,
  InvoiceDate,
  Total,
  PONumber
)
SELECT (NEXT VALUE FOR Accounting.InvoiceID_Seq),
  @XML, * FROM OPENXML (@XmlDocumentHandle, '/Invoice',2)
  WITH (
    CustomerID nvarchar(11) 'Customer/@ID',
    CustomerName nvarchar(50) 'Customer/@Name',
    InvoiceDate date 'InvoiceDate',
    Total decimal(8, 2) 'Total',
    PONumber bigint 'PONumber'
  );

EXEC sp_xml_removedocument @XmlDocumentHandle;
```

Invoices.xmlAll 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:



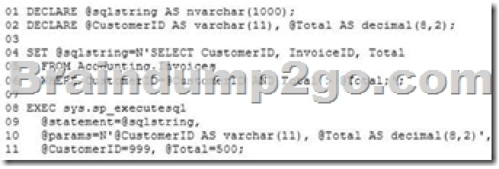
```
<?xml version="1.0"?>
<Invoice InvoiceDate="2012
  <Customer [illegible]
  [illegible]
  <PONumber>1666</PONumber
</Invoice>
```

InvoicesByCustomer.sql



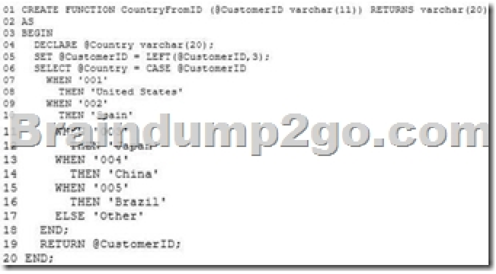
```
(SELECT CustomerID,
  CustomerName,
  InvoiceID,
  InvoiceDate,
  Total,
  PONumber
  FROM Accounting.Invoices
  WHERE CustomerID=@CustID);
```

Legacy.sql



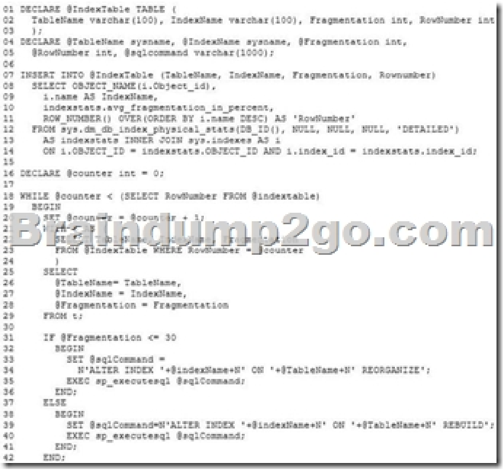
```
DECLARE @sqlstring AS nvarchar(1000);
DECLARE @CustomerID AS varchar(11), @Total AS decimal(8,2);

SET @sqlstring=N'SELECT CustomerID, InvoiceID, Total
  FROM Accounting.Invoices
  WHERE CustomerID=@CustomerID AND Total > @Total;';

EXEC sys.sp_executesql
  @statement=@sqlstring,
  @params=N'@CustomerID AS varchar(11), @Total AS decimal(8,2)',
  @CustomerID=999, @Total=500;
```

CountryFromID.sql



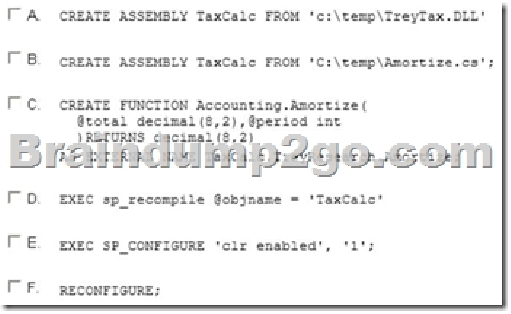
```
CREATE FUNCTION CountryFromID (@CustomerID varchar(11)) RETURNS varchar(20)
AS
BEGIN
  DECLARE @Country varchar(20);
  SET @CustomerID = LEFT(@CustomerID,3);
  SELECT @Country = CASE @CustomerID
    WHEN '001'
      THEN 'United States'
    WHEN '002'
      THEN 'Spain'
     WHEN '003'
       THEN 'Japan'
     WHEN '004'
       THEN 'China'
     WHEN '005'
       THEN 'Brazil'
     ELSE 'Other'
   END;
   RETURN @CustomerID;
END;
```

IndexManagement.sql



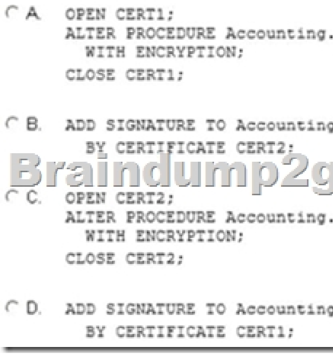
```
DECLARE @IndexTable TABLE (
  TableName varchar(100), IndexName varchar(100), Fragmentation int, RowNumber int
  );
DECLARE @TableName sysname, @IndexName sysname, @Fragmentation int,
  @RowNumber int, @sqlcommand varchar(1000);

INSERT INTO @IndexTable (TableName, IndexName, Fragmentation, Rownumber)
  SELECT OBJECT_NAME(i.Object_id),
    i.name AS IndexName,
    indexstats.avg_fragmentation_in_percent,
    ROW_NUMBER() OVER(ORDER BY i.name DESC) AS 'RowNumber'
  FROM sys.dm_db_index_physical_stats(DB_ID(), NULL, NULL, NULL, 'DETAILED')
    AS indexstats INNER JOIN sys.indexes AS i
    ON i.OBJECT_ID = indexstats.OBJECT_ID AND i.index_id = indexstats.index_id;

DECLARE @counter int = 0;

WHILE @counter < (SELECT RowNumber FROM @indextable)
  BEGIN
    SET @counter = @counter + 1;
    WITH t AS (
      SELECT TableName, IndexName, Fragmentation
      FROM @IndexTable WHERE RowNumber = @counter
      )
    SELECT
      @TableName= TableName,
      @IndexName = IndexName,
      @Fragmentation = Fragmentation
    FROM t;

    IF @Fragmentation <= 30
      BEGIN
        SET @sqlCommand =
          N'ALTER INDEX '+@indexName+N' ON '+@TableName+N' REORGANIZE';
        EXEC sp_executesql @sqlCommand;
      END;
    ELSE
      BEGIN
        SET @sqlCommand=N'ALTER INDEX '+@indexName+N' ON '+@TableName+N' REBUILD';
        EXEC sp_executesql @sqlCommand;
      END;
    END;
```

QUESTION 72You 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.   404B.   06C.   19D.   05 Answer: CExplanation: http://msdn.microsoft.com/en-us/library/ms186755.aspx http://msdn.microsoft.com/en-us/library/ms191320.aspx QUESTION 73 Which data type should you use for CustomerlD? A.   varchar(11)B.   bigintC.   nvarchar(11)D.   char(11) Answer: DExplanation: Invoices.xmlAll 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: -231 (-2,147,483,648) to 231-1 (2,147,483,647) (just 10 digits max) bigint: -263 (-9,223,372,036,854,775,808) to 263-1 (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 74You have a SQL Server 2012 database named database1. Database1 has a data file named databasel_data.mdf and a transaction log file named databaseljog.ldf. Databasel_data.mdf is 1.5 GB. Databaseljog.ldf is 1.5 terabytes.A full backup of Database1 is performed every day.You need to reduce the size of the log file. The solution must ensure that you can perform transaction log backups in the future. Which code segment should you execute? To answer, move the appropriate code segments from the list of code segments to the answer area and arrange them in the correct order.



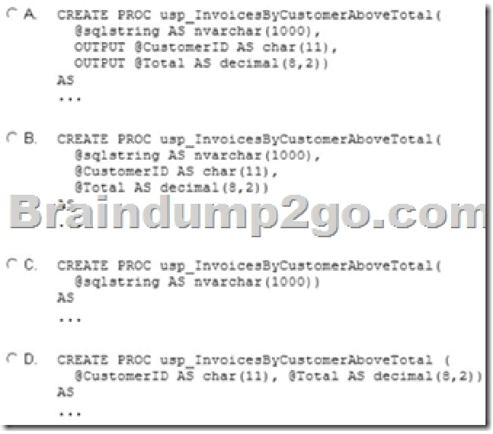
```
□ A.  CREATE ASSEMBLY TaxCalc FROM 'c:\temp\TreyTax.DLL'

□ B.  CREATE ASSEMBLY TaxCalc FROM 'C:\temp\Amortize.cs';

□ C.  CREATE FUNCTION Accounting.Amortize(
        @total decimal(8,2),@period int
        )RETURNS decimal(8,2)
        AS EXTERNAL NAME TaxCalc.TreyResearch.Amortize;

□ D.  EXEC sp_recompile @objname = 'TaxCalc'

□ E.  EXEC SP_CONFIGURE 'clr enabled', '1';

□ F.  RECONFIGURE;
```

A.   Option AB.   Option BC.   Option CD.   Option DE.   Option EF.   Option F Answer: ACDE QUESTION 75You need to modify InsertInvoice to comply with the application requirements. Which code segment should you execute?



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

○ B.  ADD SIGNATURE TO Accounting
        BY CERTIFICATE CERT2;

○ C.  OPEN CERT2;
      ALTER PROCEDURE Accounting.
        WITH ENCRYPTION;
      CLOSE CERT2;

○ D.  ADD SIGNATURE TO Accounting
        BY CERTIFICATE CERT1;
```

A.   Option AB.   Option BC.   Option CD.   Option D Answer: BExplanation: http://msdn.microsoft.com/en-us/library/bb669102.aspx QUESTION 76You 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),
  OUTPUT @CustomerID AS char(11),
  OUTPUT @Total AS decimal(8,2))
AS
...
```

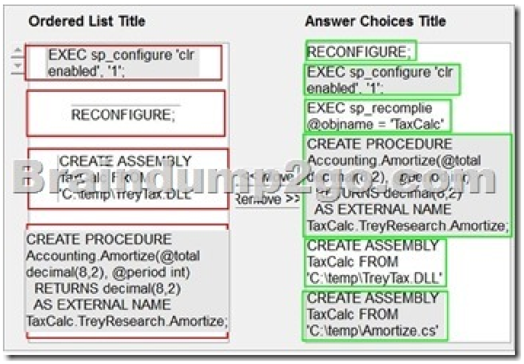
○ B.
```
CREATE PROC usp_InvoicesByCustomerAboveTotal(
  @sqlstring AS nvarchar(1000),
  @CustomerID AS char(11),
  @Total AS decimal(8,2))
AS
...
```

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

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

A. Option AB. Option BC. Option CD. Option D Answer: DExplanation: 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 77Drag and Drop QuestionYou need to build a stored procedure thatamortizes 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.

Answer:



Ordered List Title:

```
EXEC sp_configure 'clr enabled', '1';
```

```
RECONFIGURE;
```

```
CREATE ASSEMBLY TaxCalc FROM 'C:\temp\TreyTax.DLL'
```

```
CREATE PROCEDURE Accounting.Amortize(@total decimal(8,2), @period int)
  RETURNS decimal(8,2)
  AS EXTERNAL NAME TaxCalc.TreyResearch.Amortize;
```

Answer Choices Title:

```
RECONFIGURE;
```

```
EXEC sp_configure 'clr enabled', '1';
```

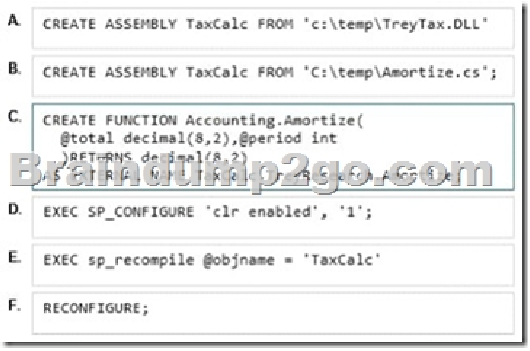
```
EXEC sp_recomplie @objname = 'TaxCalc'
```

```
CREATE PROCEDURE Accounting.Amortize(@total decimal(8,2), @period int)
  RETURNS decimal(8,2)
  AS EXTERNAL NAME TaxCalc.TreyResearch.Amortize;
```

```
CREATE ASSEMBLY TaxCalc FROM 'C:\temp\TreyTax.DLL'
```

```
CREATE ASSEMBLY TaxCalc FROM 'C:\temp\Amortize.cs'
```

Explanation:http://msdn.microsoft.com/en-us/library/ms131089.aspx http://msdn.microsoft.com/en-us/library/ms131048.aspx http://msdn.microsoft.com/en-us/library/ms187926.aspx QUESTION 78You need to prepare the database to use the .NET Framework tax component.Which code segments should you execute? (Each correct answer presents part of the solution. Chooseall that apply.)



A.
```
CREATE ASSEMBLY TaxCalc FROM 'c:\temp\TreyTax.DLL'
```

B.
```
CREATE ASSEMBLY TaxCalc FROM 'C:\temp\Amortize.cs';
```

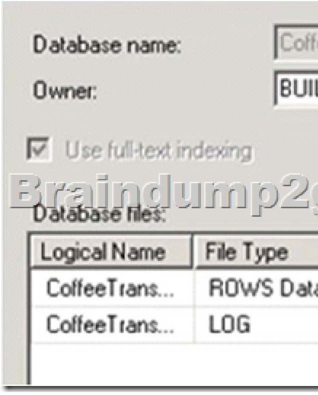
C.
```
CREATE FUNCTION Accounting.Amortize(
  @total decimal(8,2),@period int
  )RETURNS decimal(8,2)
AS EXTERNAL NAME TaxCalc.TreyResearch.Amortize;
```

D.
```
EXEC SP_CONFIGURE 'clr enabled', '1';
```

E.
```
EXEC sp_recompile @objname = 'TaxCalc'
```

F.
```
RECONFIGURE;
```

A. Option AB. Option BC. Option CD. Option DE. Option EF. Option F Answer: ACDE Case Study 8 - Fourth Coffee (QUESTION 79 ~ QUESTION 93)BackgroundCorporate InformationFourth Coffee is global restaurant chain. There are more than 5,000 locations worldwide.Physical LocationsCurrently a server at each location hosts a SQL Server 2012 instance. Each instance contains a database called StoreTransactions that stores all transactions from point of sale and uploads summary batches nightly. Each server belongs to the COFFECORP domain. Local computer accounts access the StoreTransactions database at each store using sysadmin and data reader writer roles.Planned ChangesFourth Coffee has three major initiatives:- The IT department must consolidate the point of sales database infrastructure. - The marketing department plans to launch a mobile application for micropayments. - The finance department wants to deploy an internal tool that will help detect fraud.Initially, the mobile application will allow customers to make micropayments to buy coffee and other items on the company web site. These micropayments may be sent as gifts to other users and redeemed within an hour of ownership transfer. Later versions will generate profiles based on customer activity that will push texts and ads generated by an analytics application.When the consolidation is finished and the mobile application is in production, the micropayments and point of sale transactions will use the same database.Existing EnvironmentExisting Application EnvironmentSome stores have been using several pilot versions of the micropayment application. Each version currently is in a database that is independent from the point of sales systems. Some versions have been used in field tests at local stores, and others are hosted at corporate servers. All pilot versions were developed by using SQL Server 2012.Existing Supporting InfrastructureThe proposed database for consolidating micropayments and transactions is called Coffee Transactions. The database is hosted on a SQL Server 2014 Enterprise Edition instance and has the following file structures:



Database name: Coff

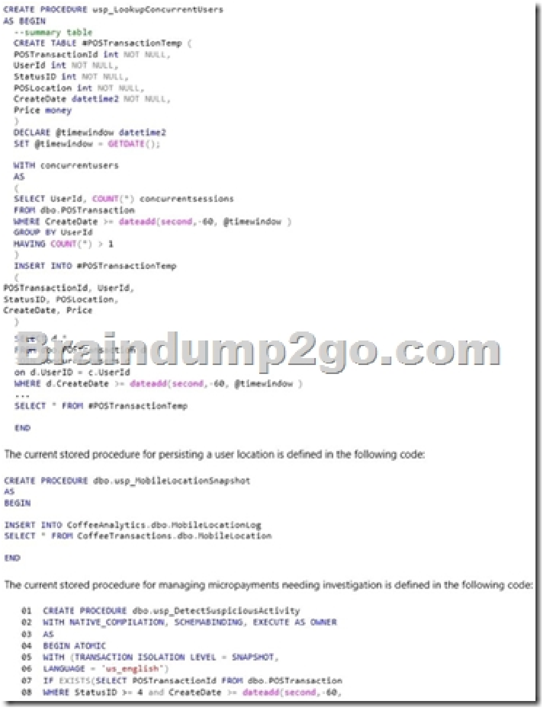
Owner: BUI

☑ Use full-text indexing

Database files:

Logical Name	File Type
CoffeeTrans...	ROWS Dat
CoffeeTrans...	LOG

Business RequirementsGeneral Application Solution RequirementsThe database infrastructure must support a phased global rollout of the micropaymentapplication and consolidation.The consolidated micropayment and point of sales database will be into a Coffee Transactions database.The infrastructure also will include a new Coffee Analytics database for reporting on content from Coffee Transactions.Mobile applications will interact most frequently with the micropayment database for the following activities:- retrieving the current status of a micropayment- modifying the status of the current micropayment; - and canceling the micropayment.The mobile application will need to meet the following requirements:- Communicate with web services that assign a new user to a micropayment by using a stored procedure named usp_AssignUser.- Update the location of the user by using a stored procedure namedusp_AddMobileLocation.The fraud detection service will need to meet the following requirements:- Query the current open micropayments for users who own multiple micropayments by using a stored procedure named usp_LookupConcurrentUsers. - Persist the currentuser locations by using a stored procedure named usp.MobileLocationSnapshot.- Look at the status of micropayments and mark micropayments for internal investigations.- Move micropayments to dbo.POSException table by using a stored procedure named ups_DetectSuspiciousActivity.The Coffee Analytics database will combine imports of the POSTransaction and Mobile Location tables to create a UserActivity table for reports on the trends in activity. Queries against the UserActivity table will include aggregated calculations on all columns that are not used in filters or groupings.Micropayments need to be updated and queried for only a week after their creation by the mobile application or fraud detection services.PerformanceThe most critical performance requirement is keeping the response time for any queries of the POSTransaction table predictable and fast.Web service queries will take a higher priority in performance tuning decisions over the fraud detection agent queries.ScalabilityQueries of the user ofa micropayment cannot return while the micropayment is being updated, but can show different users during different stages of the transaction.The fraud detection service frequently will run queries over the micropayments that occur over different time periods that range between 30 seconds and ten minutes.The POSTransaction table must have its structure optimized for hundreds of thousands of active micropayments that are updated frequently.All changes to the POSTransaction table will require testing inorder to confirm the expected throughput that will support the first year's performance requirements.Updates of a user's location can tolerate some data loss.Initial testing has determined that the POSTransaction and POSException tables will be migrated to an in-memory optimized table.AvailabilityIn order to minimize disruption at local stores during consolidation, nightly processes will restore the databases to a staging server at corporate headquarters.Technical RequirementsSecurityThe sensitive nature of financial transactions in the store databases requires certification of the COFFECORPAuditors group at corporate that will perform audits of the data. Members of the COFFECORPAuditors group cannot have sysadmin or datawriter access to the database.Compliance requires that the data stewards have access to any restored StoreTransactions database without changing any security settings at a database level.Nightly batch processes are run by the services account in the COFFECORPStoreAgent group and need to be able to restore and verify the schema of the store databases match.No Windows group should have more access to store databases than is necessary.MaintainabilityYou need to anticipate when POSTransaction table will need index maintenance.When the daily maintenance finishes, micropayments that are one week old must be available for queries in UserActivity table but will be queried most frequently within their first week and will require support for in-memory queries for data withinfirst week.The maintenance of the UserActivity table must allow frequent maintenance on the day's most recent activities with minimal impact on the use of disk space and the resources available to queries. The processes that add data to the UserActivitytable must be able to update data from any time period, even while maintenance is running.The index maintenance strategy for the UserActivity table must provide the optimal structure for both maintainability and query performance.All micropaymentsqueries must include the most permissive isolation level available for the maximum throughput.In the event of unexpected results, all stored procedures must provide error messages in text message to the calling web service.Any modifications to stored procedures will require the minimal amount of schema changes necessary to increase the performance.PerformanceStress testing of the mobile application on the proposed Coffee Transactions database uncovered performance bottlenecks. The sys.dm_os_wait_stats Dynamic Management View (DMV) shows high wait_time values for WRTTELOG and PAGEIOLATCHJJP wait types when updating the MobileLocation table.Updates to the MobileLocation table must have minimal impact on physical resources.Supporting InfrastructureThe stored procedure usp_LookupConcurrentUsers has the current implementation:



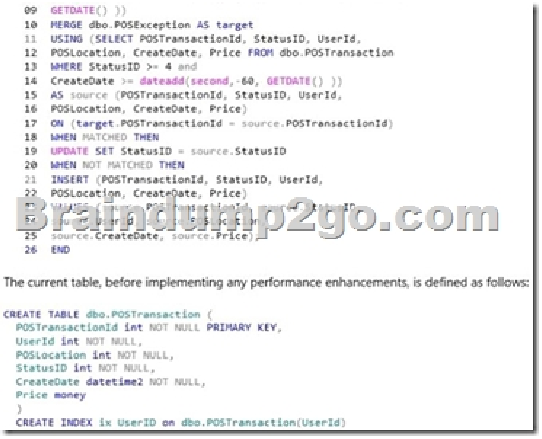
```
CREATE PROCEDURE usp_LookupConcurrentUsers
AS BEGIN
  --summary table
  CREATE TABLE #POSTransactionTemp (
  POSTransactionId int NOT NULL,
  UserId int NOT NULL,
  StatusID int NOT NULL,
  POSLocation int NOT NULL,
  CreateDate datetime2 NOT NULL,
  Price money
  )
  DECLARE @timewindow datetime2
  SET @timewindow = GETDATE();

  WITH concurrentusers
  AS
  (
  SELECT UserId, COUNT(*) concurrentsessions
  FROM dbo.POSTransaction
  WHERE CreateDate >= dateadd(second,-60, @timewindow )
  GROUP BY UserId
  HAVING COUNT(*) > 1
  )
  INSERT INTO #POSTransactionTemp
  (
POSTransactionId, UserId,
StatusID, POSLocation,
CreateDate, Price
  )
  SELECT d.*
  FROM dbo.POSTransaction d
  JOIN concurrentusers c
  on d.UserID = c.UserId
  WHERE d.CreateDate >= dateadd(second,-60, @timewindow )
  ...
  SELECT * FROM #POSTransactionTemp

  END
```

The current stored procedure for persisting a user location is defined in the following code:

```
CREATE PROCEDURE dbo.usp_MobileLocationSnapshot
AS
BEGIN

INSERT INTO CoffeeAnalytics.dbo.MobileLocationLog
SELECT * FROM CoffeeTransactions.dbo.MobileLocation

END
```

The current stored procedure for managing micropayments needing investigation is defined in the following code:

```
CREATE PROCEDURE dbo.usp_DetectSuspiciousActivity
WITH NATIVE_COMPILATION, SCHEMABINDING, EXECUTE AS OWNER
AS
BEGIN ATOMIC
WITH (TRANSACTION ISOLATION LEVEL = SNAPSHOT,
LANGUAGE = 'us_english')
IF EXISTS(SELECT POSTransactionId FROM dbo.POSTransaction
WHERE StatusID >= 4 and CreateDate >= dateadd(second,-60,
```

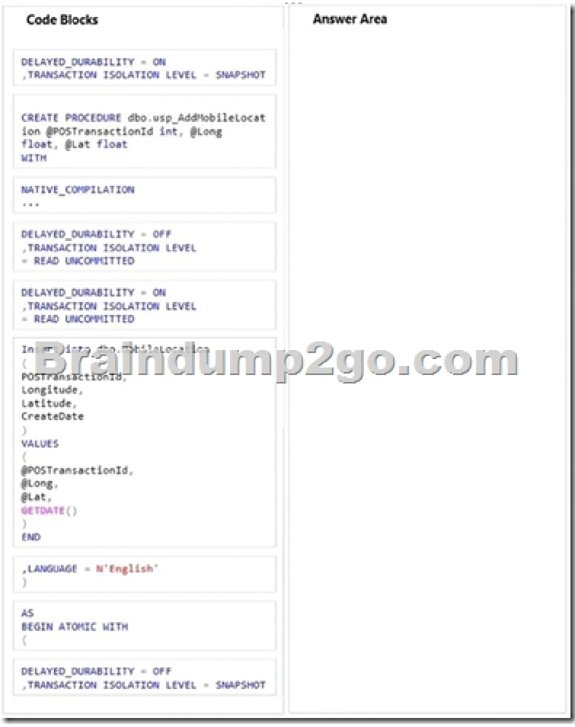


```
GETDATE() ))
MERGE dbo.POSException AS target
USING (SELECT POSTransactionId, StatusID, UserId,
POSLocation, CreateDate, Price FROM dbo.POSTransaction
WHERE StatusID >= 4 and
CreateDate >= dateadd(second,-60, GETDATE() ))
AS source (POSTransactionId, StatusID, UserId,
POSLocation, CreateDate, Price)
ON (target.POSTransactionId = source.POSTransactionId)
WHEN MATCHED THEN
UPDATE SET StatusID = source.StatusID
WHEN NOT MATCHED THEN
INSERT (POSTransactionId, StatusID, UserId,
POSLocation, CreateDate, Price)
VALUES (source.POSTransactionId, source.StatusID,
source.UserId, source.POSLocation,
source.CreateDate, source.Price);
END
```

The current table, before implementing any performance enhancements, is defined as follows:

```
CREATE TABLE dbo.POSTransaction (
  POSTransactionId int NOT NULL PRIMARY KEY,
  UserId int NOT NULL,
  POSLocation int NOT NULL,
  StatusID int NOT NULL,
  CreateDate datetime2 NOT NULL,
  Price money
  )
  CREATE INDEX ix UserID on dbo.POSTransaction(UserId)
```

QUESTION 79You need to monitor the health of your tables and indexes in order to implement the required index maintenance strategy.What should you do? A. Query system DMVs to monitor avg_chain_length and max_chain_length. Create alerts to notify you when these values converge.B. Create a SQL Agent alert when the File Table: Avg time per file I/O request value is increasing.C. Query system DMVs to monitor total_bucket_count. Create alerts to notify you when this value increases.D. Query system DMVs to monitor total_bucket_count. Create alerts to notify you when this value decreases. Answer: AExplanation:From scenario:* You need to anticipate when POSTransaction table willneed index maintenance.* The index maintenance strategy for the UserActivity table must provide the optimal structure for both maintainability and query performance. QUESTION 80Drag and Drop QuestionYou need to implement a new version of usp_AddMobileLocation.Develop the solution by selecting and arranging the required code blocks in the correct order. You may not need all of the code blocks.



Code Blocks

```
DELAYED_DURABILITY = ON
,TRANSACTION ISOLATION LEVEL = SNAPSHOT
```

```
CREATE PROCEDURE dbo.usp_AddMobileLocat
ion @POSTransactionId int, @Long
float, @Lat float
WITH
```

```
NATIVE_COMPILATION
...
```

```
DELAYED_DURABILITY = OFF
,TRANSACTION ISOLATION LEVEL
= READ UNCOMMITTED
```

```
DELAYED_DURABILITY = ON
,TRANSACTION ISOLATION LEVEL
= READ UNCOMMITTED
```

```
Insert into dbo.MobileLocation
(
POSTransactionId,
Longitude,
Latitude,
CreateDate
)
VALUES
(
@POSTransactionId,
@Long,
@Lat,
GETDATE()
)
END
```

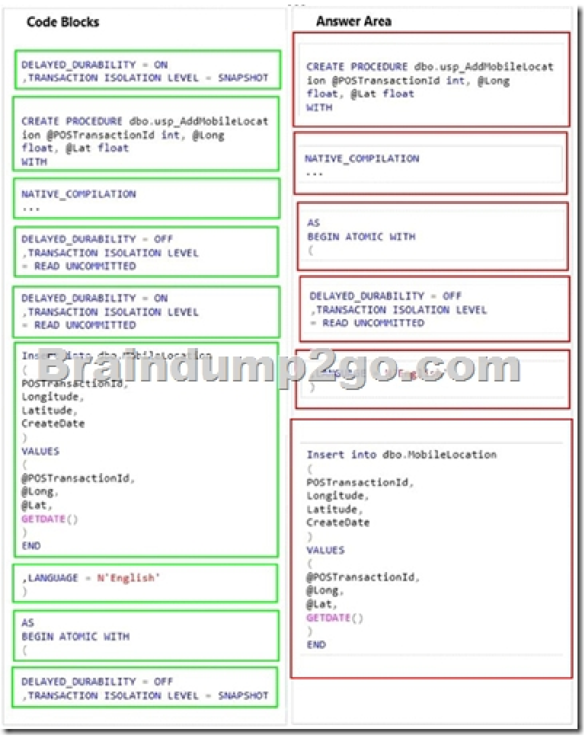
```
,LANGUAGE = N'English'
)
```

```
AS
BEGIN ATOMIC WITH
(
```

```
DELAYED_DURABILITY = OFF
,TRANSACTION ISOLATION LEVEL = SNAPSHOT
```

Answer Area

Answer:



Code Blocks

```
DELAYED_DURABILITY = ON
,TRANSACTION ISOLATION LEVEL = SNAPSHOT
```

```
CREATE PROCEDURE dbo.usp_AddMobileLocat
ion @POSTransactionId int, @Long
float, @Lat float
WITH
```

```
NATIVE_COMPILATION
...
```

```
DELAYED_DURABILITY = OFF
,TRANSACTION ISOLATION LEVEL
= READ UNCOMMITTED
```

```
DELAYED_DURABILITY = ON
,TRANSACTION ISOLATION LEVEL
= READ UNCOMMITTED
```

```
Insert into dbo.MobileLocation
(
POSTransactionId,
Longitude,
Latitude,
CreateDate
)
VALUES
(
@POSTransactionId,
@Long,
@Lat,
GETDATE()
)
END
```

```
,LANGUAGE = N'English'
)
```

```
AS
BEGIN ATOMIC WITH
(
```

```
DELAYED_DURABILITY = OFF
,TRANSACTION ISOLATION LEVEL = SNAPSHOT
```

Answer Area

```
CREATE PROCEDURE dbo.usp_AddMobileLocat
ion @POSTransactionId int, @Long
float, @Lat float
WITH
```

```
NATIVE_COMPILATION
...
```

```
AS
BEGIN ATOMIC WITH
(
```

```
DELAYED_DURABILITY = OFF
,TRANSACTION ISOLATION LEVEL
= READ UNCOMMITTED
```

```
,LANGUAGE = N'English'
)
```

```
Insert into dbo.MobileLocation
(
POSTransactionId,
Longitude,
Latitude,
CreateDate
)
VALUES
(
@POSTransactionId,
@Long,
@Lat,
GETDATE()
)
END
```