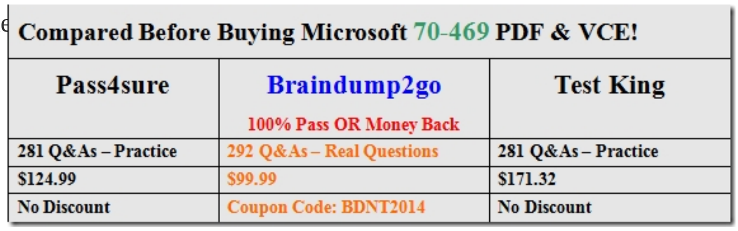
Braindump2go 70-469 Dumps 2015 Free Download (61-70)

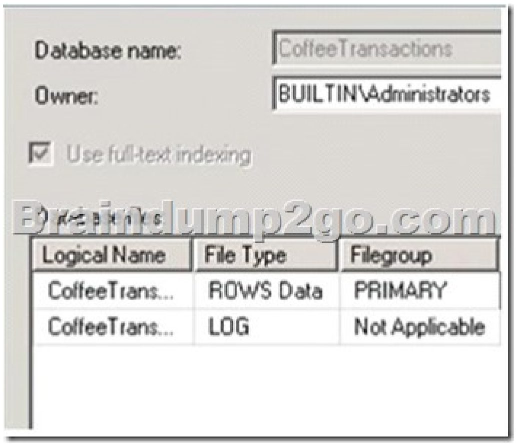
2015 Microsoft 70-469 Certification Exam is coming! Getting a Laest 70-469 Practice Test is very important for an Exam Candiate! Braindump2go New Updated 70-469 Exam Questions Well Formatted in PDF and VCE versions, providing you convenience and excellence both at the same time! Free Questions and Answer are provided Following: Vendor: MicrosoftExam Code: 70-469Exam Name: Recertification for MCSE: Data PlatformKeywords: 70-469 Exam Dumps,70-469 Practice Tests,70-469 Practice Exams,70-469 Exam Questions,70-469 PDF,70-469 VCE Free,70-469 Book,70-469 E-Book,70-469 Study Guide,70-469 Braindump,70-469 Prep Guide



Compared Before Buying Microsoft 70-469 PDF & VCE!

Pass4sure	Braindump2go 100% Pass OR Money Back	Test King
281 Q&As – Practice	292 Q&As – Real Questions	281 Q&As – Practice
$124.99	$99.99	$171.32
No Discount	Coupon Code: BDNT2014	No Discount

QUESTION 61You need to ensure that usp_AddXMLOrder can be used to validate the XML input from the retailers.Which parameters should you add to usp_AddXMLOrder on line 04 and line 05? (Each correct answer presents part of the solution. Choose all that apply.) A. @schema varbinary(100).B. @items varchar(max).C. @schema sysname.D. @items varbinary(max).E. @items xml.F. @schema xml. Answer: CE QUESTION 62You need to implement a solution that addresses the index monitoring requirements.What should you do? A. Schedule a SQL Server Agent job that saves data from the dynamic management views to a table in the database.B. Create a SQL Server Audit that saves data to a log file, and then create a SQL Server Audit Specification that gathers data from the DATABASE_OPERATION group.C. Create a performance monitor Data Collector Set (DCS) that monitors the SQL Server counters.D. Schedule a SQL Server Profiler trace, and then save the trace data to a table in the database. Answer: A QUESTION 63You need to implement a solution that solves the performance issues of usp_GetOrdersAndItems.Which statements should you execute? A. CREATE INDEX IX_Orders_Active ON Orders(ShipDate, DeliveryDate, Amount)B. CREATE INDEX IX_Orders_Active ON Orders(DeliveryDate) INCLUDE(Amount) WHERE ShipDate IS NULLC. CREATE INDEX IX_Orders_Active ON Orders(DeliveryDate, Amount) WHERE ShipDateIS NULLD. CREATE INDEX IX_Orders_Active ON Orders(ShipDate, DeliveryDate) INCLUDE(Amount) Answer: B QUESTION 64You need to modify usp.GetOrdersAndItems to ensure that an order is NOT retrieved by usp_GetOrdersAndItems while the order is being updated.What should you add to usp.GetOrdersAndItems? A. Add WITH (NOLOCK) to the end of line 47.B. Add SET TRANSACTION ISOLATION LEVEL READ COMMITTED to line 44.C. Add SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED to line 44.D. Add WITH (READPAST) to the end of line 47. Answer: B QUESTION 65You need to ensure that a new execution plan is used by usp_GetOrdersByProduct each time the stored procedure runs.What should you do? A. Execute sp_help usp_GetOrdersByProductB. Add WITH (FORCESEEK) to line 69 in usp.GetOrdersByProduct.C. Add WITH RECOMPILE to line 64 in usp.GetOrdersByProduct.D. Execute sp_recompile usp.GetOrdersByProduct'. Answer: B Case Study 7 - Fourth Coffee (Question 66 - Question 86)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 datareaderwriter 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 Support InfrastructureThe proposed database for consolidating micropayments and transactions is called CoffeeTransactions. The database is hosted on a SQL Server 2014 Enterprise Edition instance and has the following file structures:



Database name: CoffeeTransactions

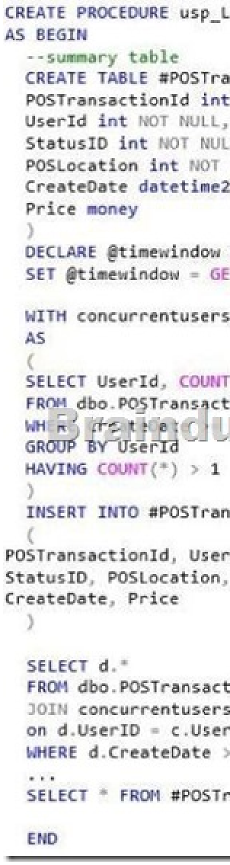
Owner: BUILTIN\Administrators

☑ Use full-text indexing

Database files:

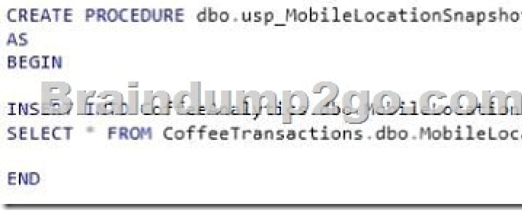
Logical Name	File Type	Filegroup
CoffeeTrans...	ROWS Data	PRIMARY
CoffeeTrans...	LOG	Not Applicable

Business RequirementsGeneral Application Solution RequirementsThe database infrastructure must support a phased global rollout of the micropayment application and consolidation.The consolidated micropayment and point of sales database will be into a CoffeeTransactions database. The infrastructure also will include a new CoffeeAnalytics database for reporting on content from CoffeeTransactions.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 named usp_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 current user 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.- Detect micropayments that are flagged with a StatusId value that is greater than 3 and that occurred within the last minute.The CoffeeAnalytics database will combine imports of the POSTransaction and MobileLocation 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 of a 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 in order 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 within first 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 UserActivity table 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 micropayments queries 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 CoffeeTransactions 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:



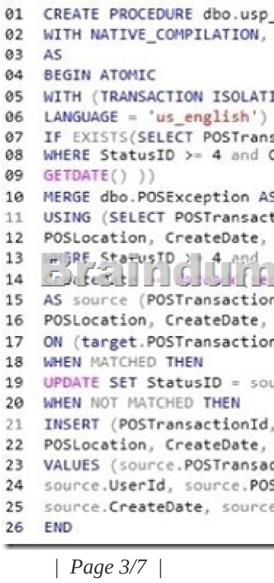
```
CREATE PROCEDURE usp_L
AS BEGIN
  --summary table
  CREATE TABLE #POSTra
  POSTransactionId int
  UserId int NOT NULL,
  StatusID int NOT NUL
  POSLocation int NOT
  CreateDate datetime2
  Price money
  )
  DECLARE @timewindow
  SET @timewindow = GE

  WITH concurrentusers
  AS
  (
  SELECT UserId, COUNT
  FROM dbo.POSTransact
  WHERE CreateDate
  GROUP BY UserId
  HAVING COUNT(*) > 1
  )
  INSERT INTO #POSTran
  (
POSTransactionId, User
StatusID, POSLocation,
CreateDate, Price
  )

  SELECT d.*
  FROM dbo.POSTransact
  JOIN concurrentusers
  on d.UserID = c.User
  WHERE d.CreateDate >
  ...
  SELECT * FROM #POSTr

  END
```

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



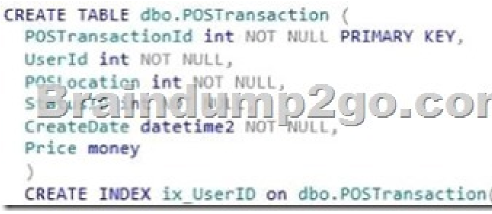
```
CREATE PROCEDURE dbo.usp_MobileLocationSnapsho
AS
BEGIN

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

END
```

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



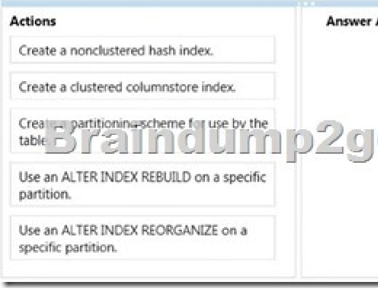
```
CREATE PROCEDURE dbo.usp_
WITH NATIVE_COMPILATION,
AS
BEGIN ATOMIC
WITH (TRANSACTION ISOLATI
LANGUAGE = 'us_english')
IF EXISTS(SELECT POSTrans
WHERE StatusID >= 4 and C
GETDATE() ))
MERGE dbo.POSException AS
USING (SELECT POSTransact
POSLocation, CreateDate,
WHERE StatusID >= 4 and
[illegible]
AS source (POSTransaction
POSLocation, CreateDate,
ON (target.POSTransaction
WHEN MATCHED THEN
UPDATE SET StatusID = sou
WHEN NOT MATCHED THEN
INSERT (POSTransactionId,
POSLocation, CreateDate,
VALUES (source.POSTransac
source.UserId, source.POS
source.CreateDate, source
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,
  S[illegible]
  CreateDate datetime2 NOT NULL,
  Price money
  )
  CREATE INDEX ix_UserID on dbo.POSTransaction(
```

QUESTION 66You 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 will need index maintenance.* The index maintenance strategy for the UserActivity table must provide the optimal structure for both maintainability and query performance. QUESTION 67Drag and Drop QuestionsYou need to design the UserActivity table.Which three steps should you perform in sequence? To answer, move the appropriate three actions from the list of actions to the answer area and arrange them in the correct order.



Actions

Create a nonclustered hash index.

Create a clustered columnstore index.

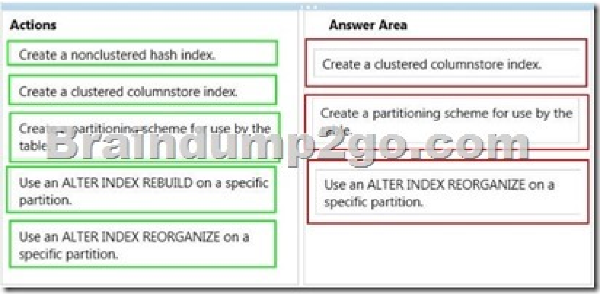
Create a partitioning scheme for use by the table.

Use an ALTER INDEX REBUILD on a specific partition.

Use an ALTER INDEX REORGANIZE on a specific partition.

Answer A

Answer:



Actions

Create a nonclustered hash index.

Create a clustered columnstore index.

Create a partitioning scheme for use by the table.

Use an ALTER INDEX REBUILD on a specific partition.

Use an ALTER INDEX REORGANIZE on a specific partition.

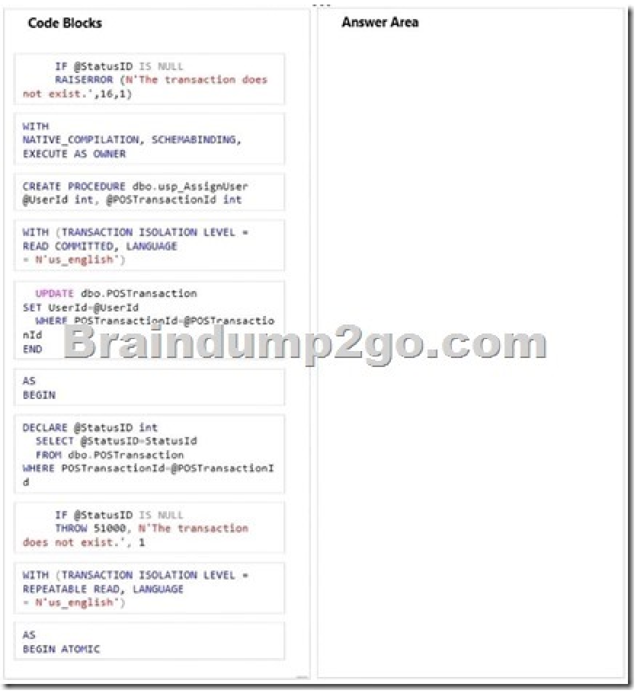
Answer Area

Create a clustered columnstore index.

Create a partitioning scheme for use by the table.

Use an ALTER INDEX REORGANIZE on a specific partition.

QUESTION 68You need to implement security for the restore and audit process. What should you do? A. Grant the COFFECORPAuditors group ALTER ANY CONNECTION and SELECT ALL USER SECURABLES permissions. Grant the COFFECORPStoreAgent group ALTER ANY CONNECTION and IMPERSONATE ANY LOGIN permissions.B. Grant the COFFECORPAuditors group CONNECT ANY DATABASE and IMPERSONATE ANY LOGIN permissions. Grant the COFFECORPStoreAgent groupCONNECT ANY DATABASE and SELECT ALL USER SECURABLES permissions.C. Grant the COFFECORPAuditors group ALTER ANY CONNECTION and IMPERSONATE ANY LOGIN permissions. Grant the COFFECORPStoreAgent group ALTER ANY CONNECTION and SELECT ALL USER SECURABLES permissions.D. Grant the COFFECORPAuditors group CONNECT ANY DATABASE and SELECT ALL USER SECURABLES permissions. Grant the COFFECORPStoreAgent group CONNECT ANY DATABASE and IMPERSONATE ANY LOGIN permissions. Answer: A QUESTION 69Drag and Drop QuestionsYou need to create the usp.AssignUser stored procedure.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

```
     IF @StatusID IS NULL
     RAISERROR (N'The transaction does
not exist.',16,1)
```

```
WITH
NATIVE_COMPILATION, SCHEMABINDING,
EXECUTE AS OWNER
```

```
CREATE PROCEDURE dbo.usp_AssignUser
@UserId int, @POSTransactionId int
```

```
WITH (TRANSACTION ISOLATION LEVEL =
READ COMMITTED, LANGUAGE
= N'us_english')
```

```
  UPDATE dbo.POSTransaction
SET UserId=@UserId
  WHERE POSTransactionId=@POSTransactio
nId
END
```

```
AS
BEGIN
```

```
DECLARE @StatusID int
  SELECT @StatusID=StatusId
  FROM dbo.POSTransaction
WHERE POSTransactionId=@POSTransactionI
d
```

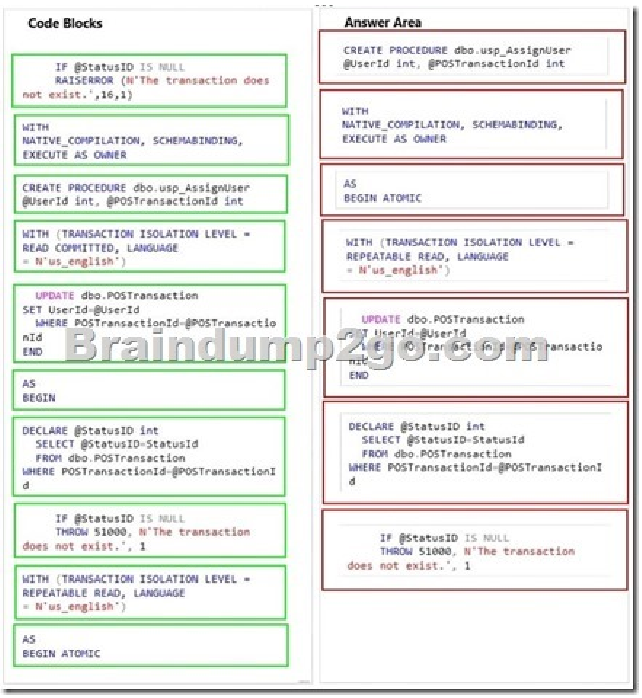
```
     IF @StatusID IS NULL
     THROW 51000, N'The transaction
does not exist.', 1
```

```
WITH (TRANSACTION ISOLATION LEVEL =
REPEATABLE READ, LANGUAGE
= N'us_english')
```

```
AS
BEGIN ATOMIC
```

Answer Area

Answer:



Code Blocks

```
     IF @StatusID IS NULL
     RAISERROR (N'The transaction does
not exist.',16,1)
```

```
WITH
NATIVE_COMPILATION, SCHEMABINDING,
EXECUTE AS OWNER
```

```
CREATE PROCEDURE dbo.usp_AssignUser
@UserId int, @POSTransactionId int
```

```
WITH (TRANSACTION ISOLATION LEVEL =
READ COMMITTED, LANGUAGE
= N'us_english')
```

```
  UPDATE dbo.POSTransaction
SET UserId=@UserId
  WHERE POSTransactionId=@POSTransactio
nId
END
```

```
AS
BEGIN
```

```
DECLARE @StatusID int
  SELECT @StatusID=StatusId
  FROM dbo.POSTransaction
WHERE POSTransactionId=@POSTransactionI
d
```

```
     IF @StatusID IS NULL
     THROW 51000, N'The transaction
does not exist.', 1
```

```
WITH (TRANSACTION ISOLATION LEVEL =
REPEATABLE READ, LANGUAGE
= N'us_english')
```

```
AS
BEGIN ATOMIC
```

Answer Area

```
CREATE PROCEDURE dbo.usp_AssignUser
@UserId int, @POSTransactionId int
```

```
WITH
NATIVE_COMPILATION, SCHEMABINDING,
EXECUTE AS OWNER
```

```
AS
BEGIN ATOMIC
```

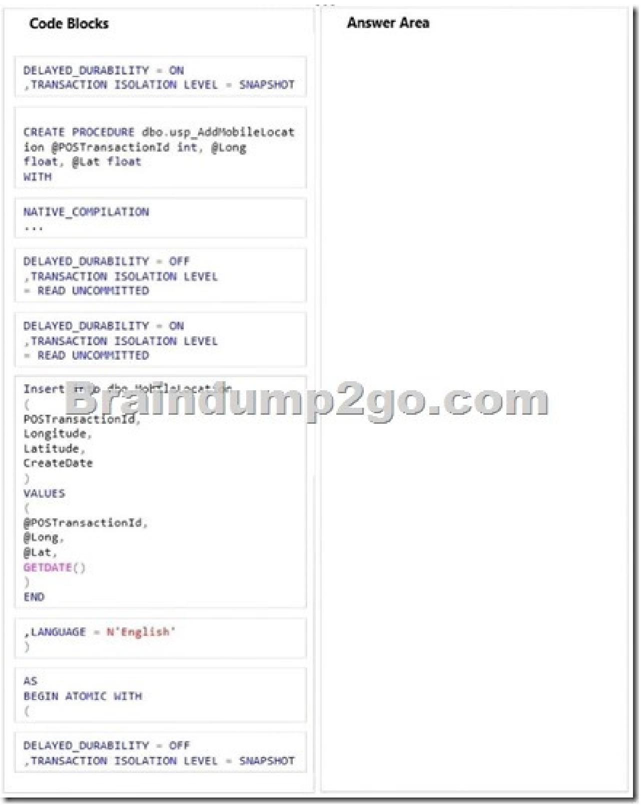
```
WITH (TRANSACTION ISOLATION LEVEL =
REPEATABLE READ, LANGUAGE
= N'us_english')
```

```
  UPDATE dbo.POSTransaction
SET UserId=@UserId
  WHERE POSTransactionId=@POSTransactio
nId
END
```

```
DECLARE @StatusID int
  SELECT @StatusID=StatusId
  FROM dbo.POSTransaction
WHERE POSTransactionId=@POSTransactionI
d
```

```
     IF @StatusID IS NULL
     THROW 51000, N'The transaction
does not exist.', 1
```

QUESTION 70Drag and Drop QuestionsYou 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
```

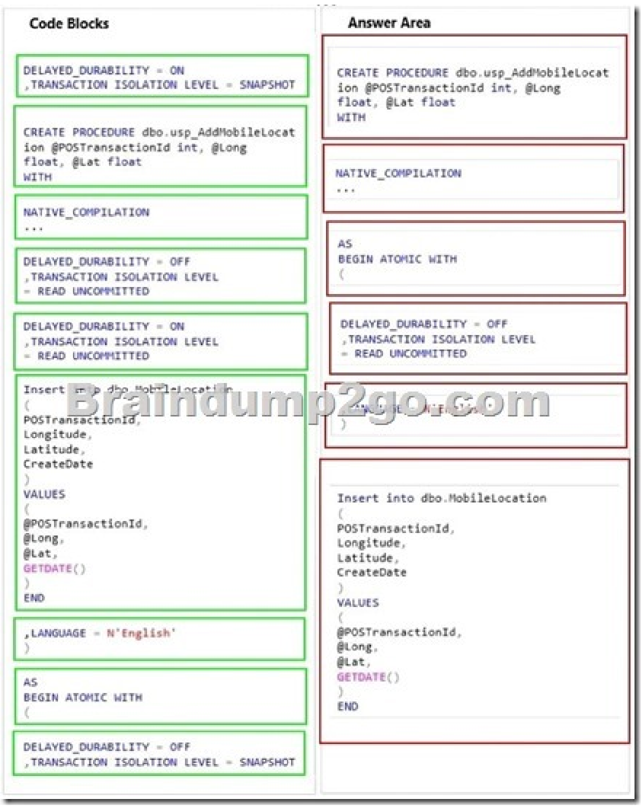
```
,LANGUAGE = N'English'
)
```

```
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
```

http://www.braindump2go.com/70-469.html