70-464$^{Q\&As}$

Developing Microsoft SQL Server Databases

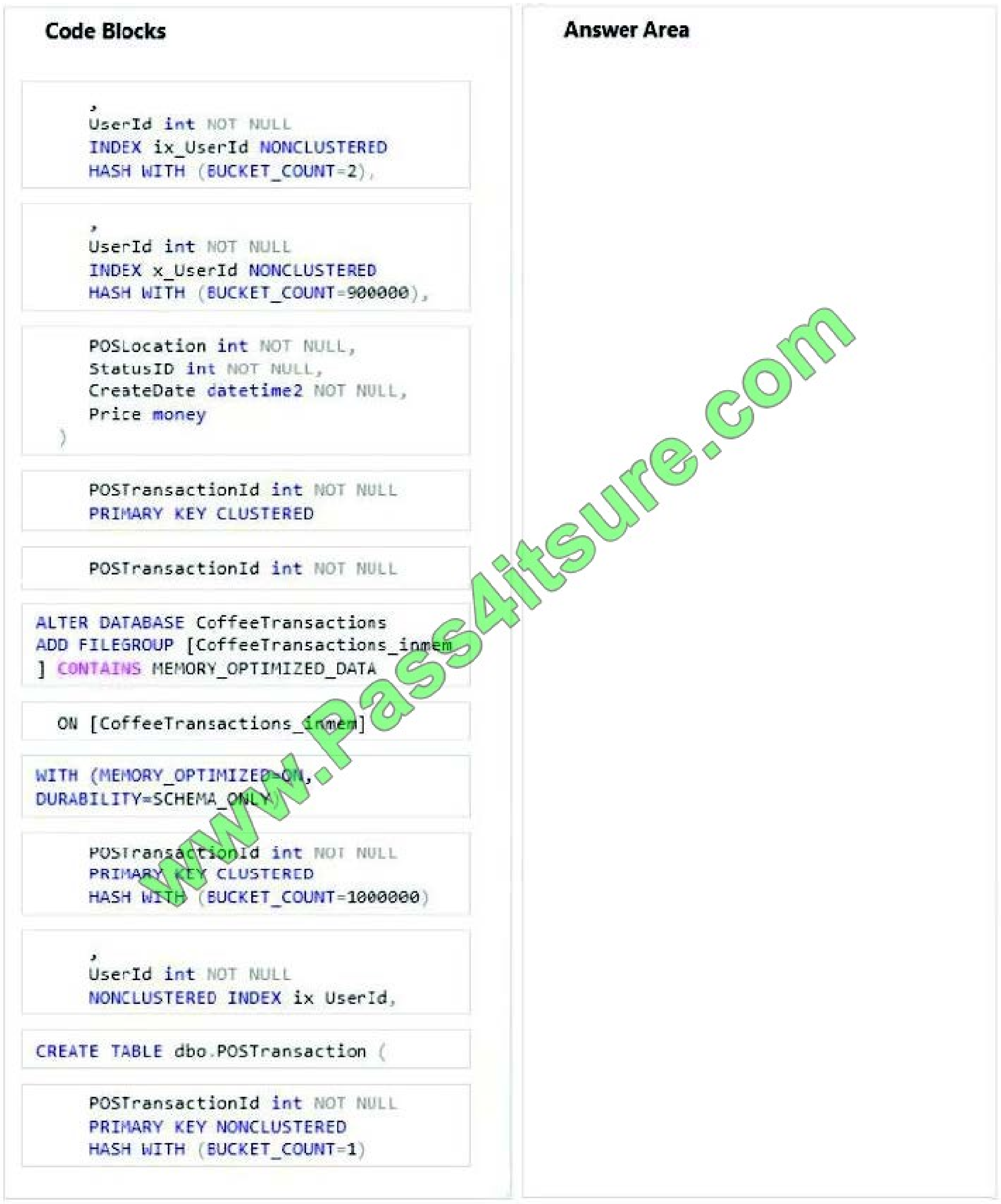
Pass Microsoft 70-464 Exam with 100% Guarantee

QUESTION 1

You need to redesign the system to meet the scalability requirements of the application.

Develop the solution by selecting and arranging the required code blocks in the correct order. You may not need all of the code blocks.

Select and Place:



Code Blocks

```
    ,
    UserId int NOT NULL
    INDEX ix_UserId NONCLUSTERED
    HASH WITH (BUCKET_COUNT=2),
```

```
    ,
    UserId int NOT NULL
    INDEX x_UserId NONCLUSTERED
    HASH WITH (BUCKET_COUNT=900000),
```

```
    POSLocation int NOT NULL,
    StatusID int NOT NULL,
    CreateDate datetime2 NOT NULL,
    Price money
  )
```

```
    POSTransactionId int NOT NULL
    PRIMARY KEY CLUSTERED
```

```
    POSTransactionId int NOT NULL
```

```
ALTER DATABASE CoffeeTransactions
ADD FILEGROUP [CoffeeTransactions_inmem
] CONTAINS MEMORY_OPTIMIZED_DATA
```

```
  ON [CoffeeTransactions_inmem]
```

```
WITH (MEMORY_OPTIMIZED=ON,
DURABILITY=SCHEMA_ONLY)
```

```
    POSTransactionId int NOT NULL
    PRIMARY KEY CLUSTERED
    HASH WITH (BUCKET_COUNT=1000000)
```

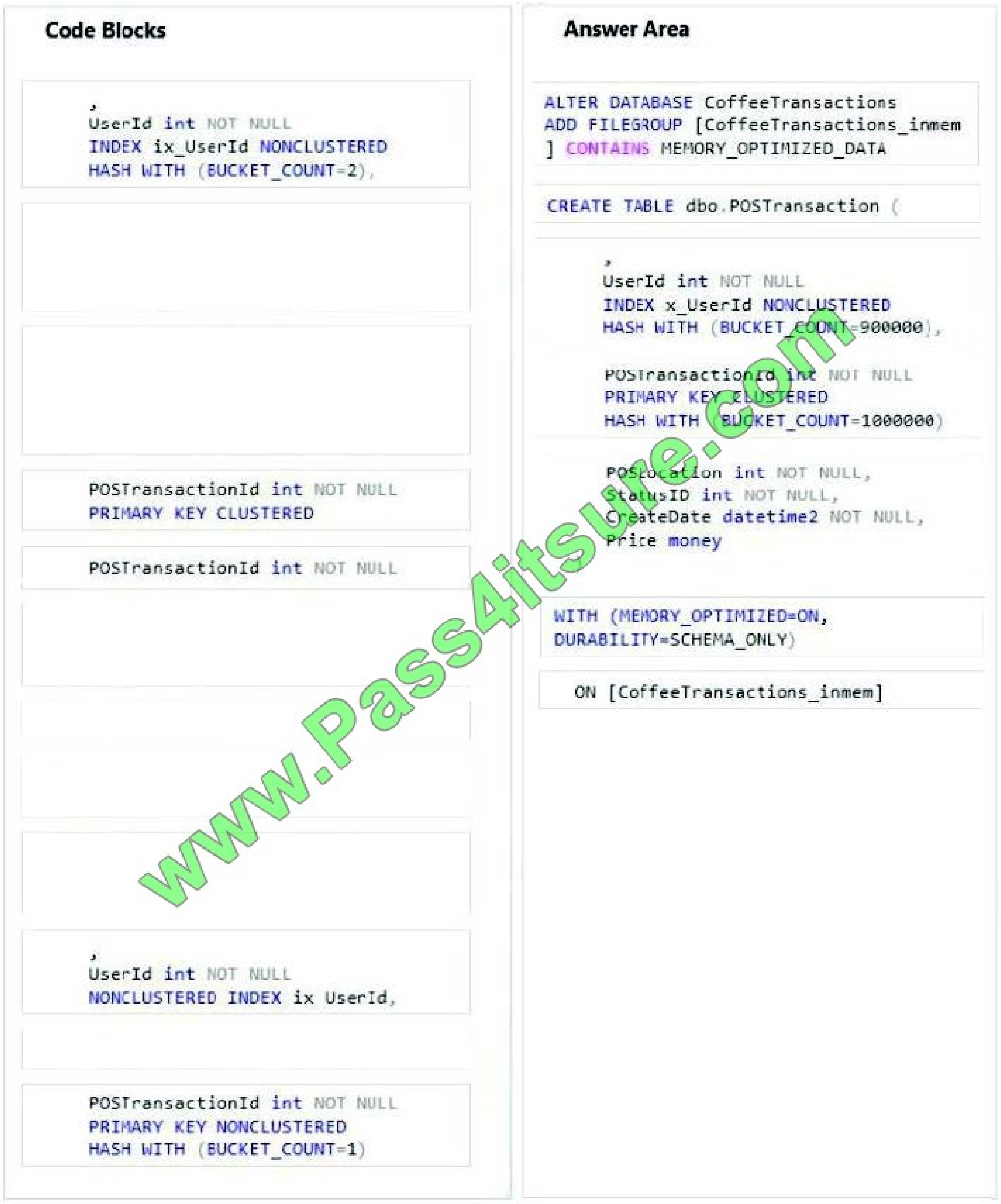
```
    ,
    UserId int NOT NULL
    NONCLUSTERED INDEX ix UserId,
```

```
CREATE TABLE dbo.POSTransaction (
```

```
    POSTransactionId int NOT NULL
    PRIMARY KEY NONCLUSTERED
    HASH WITH (BUCKET_COUNT=1)
```

Answer Area

Correct Answer:



Code Blocks

```
,
UserId int NOT NULL
INDEX ix_UserId NONCLUSTERED
HASH WITH (BUCKET_COUNT=2),
```

```
POSTransactionId int NOT NULL
PRIMARY KEY CLUSTERED
```

```
POSTransactionId int NOT NULL
```

```
,
UserId int NOT NULL
NONCLUSTERED INDEX ix UserId,
```

```
POSTransactionId int NOT NULL
PRIMARY KEY NONCLUSTERED
HASH WITH (BUCKET_COUNT=1)
```

Answer Area

```
ALTER DATABASE CoffeeTransactions
ADD FILEGROUP [CoffeeTransactions_inmem
] CONTAINS MEMORY_OPTIMIZED_DATA
```

```
CREATE TABLE dbo.POSTransaction (
```

```
,
UserId int NOT NULL
INDEX x_UserId NONCLUSTERED
HASH WITH (BUCKET_COUNT=900000),

POSTransactionId int NOT NULL
PRIMARY KEY CLUSTERED
HASH WITH (BUCKET_COUNT=1000000)

POSLocation int NOT NULL,
StatusID int NOT NULL,
CreateDate datetime2 NOT NULL,
Price money
```

```
WITH (MEMORY_OPTIMIZED=ON,
DURABILITY=SCHEMA_ONLY)
```

```
ON [CoffeeTransactions_inmem]
```

Note:

*

MEMORY_OPTIMIZED_DATA

First create a memory-optimized data filegroup and add a container to the filegroup.

Then create a memory-optimized table.

*

You must specify a value for the BUCKET_COUNT parameter when you create the memory- optimized table. In most cases the bucket count should be between

1 and 2 times the number of distinct values in the index key.

*

Example:

-- create a durable (data will be persisted) memory-optimized table -- two of the columns are indexed

CREATE TABLE dbo.ShoppingCart (

ShoppingCartId INT IDENTITY(1,1) PRIMARY KEY NONCLUSTERED, UserId INT NOT NULL INDEX ix_UserId NONCLUSTERED HASH WITH

(BUCKET_COUNT=1000000),

CreatedDate DATETIME2 NOT NULL,

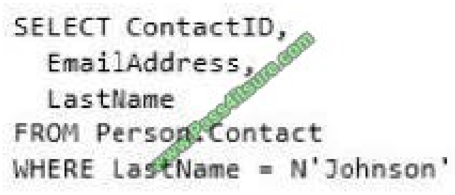
TotalPrice MONEY

) WITH (MEMORY_OPTIMIZED=ON)

GO

QUESTION 2

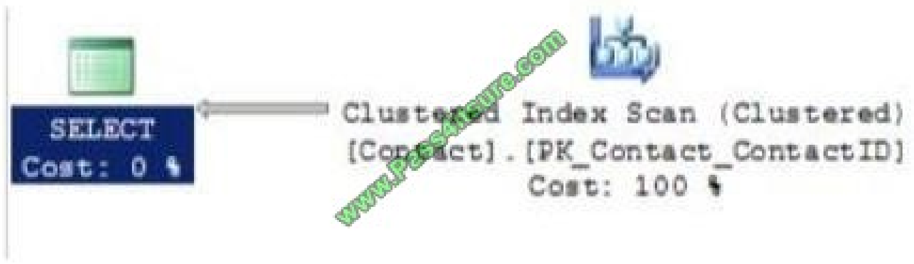
You have the following query on a disk-based table:



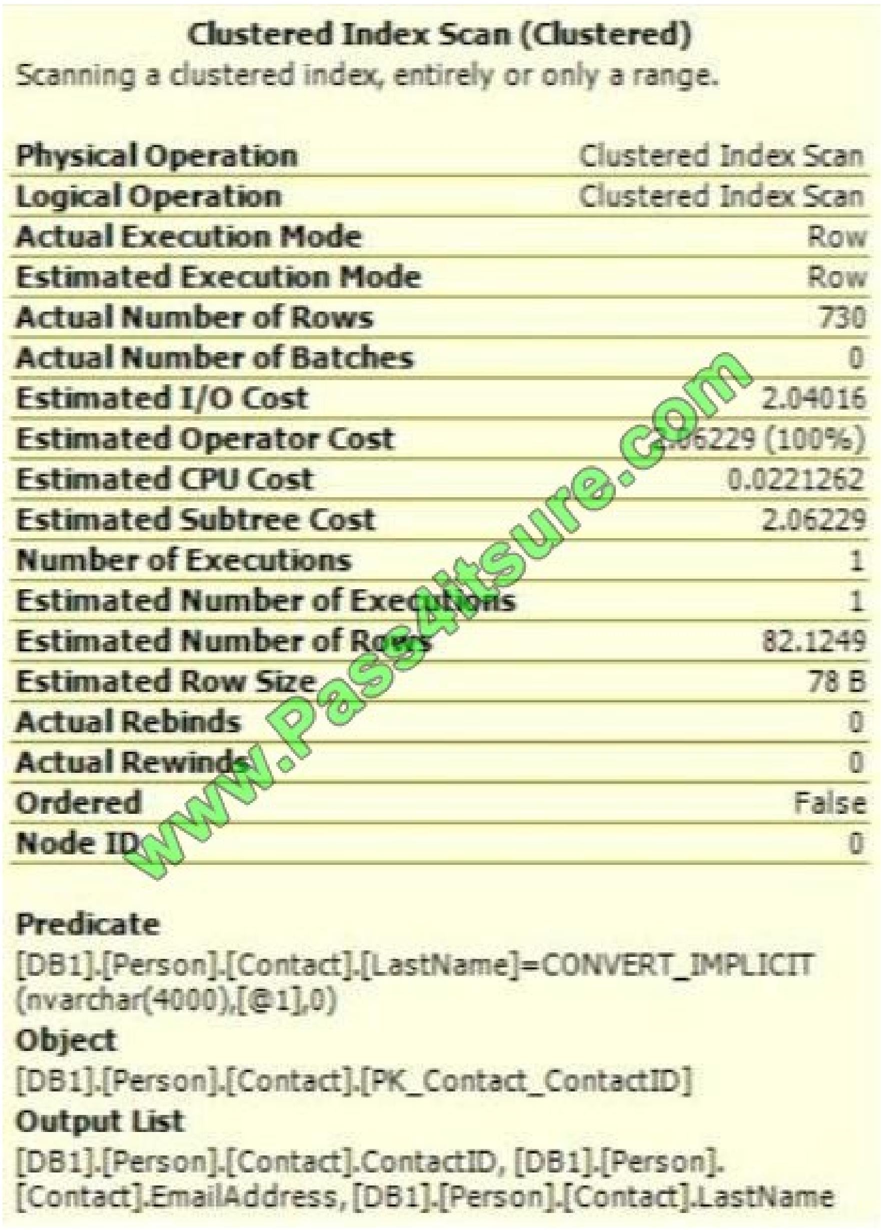
```
SELECT ContactID,
   EmailAddress,
   LastName
FROM Person.Contact
WHERE LastName = N'Johnson'
```

You discover that the query takes a long time to complete.

The execution plan is shown in the Execution Plan exhibit. (Click the Exhibit button.)


SELECT
Cost: 0 %
Clustered Index Scan (Clustered)
[Contact].[PK_Contact_ContactID]
Cost: 100 %

The index usage is show in the Index Usage exhibit. (Click the Exhibit button.)



Clustered Index Scan (Clustered)
Scanning a clustered index, entirely or only a range.

Physical Operation	Clustered Index Scan
Logical Operation	Clustered Index Scan
Actual Execution Mode	Row
Estimated Execution Mode	Row
Actual Number of Rows	730
Actual Number of Batches	0
Estimated I/O Cost	2.04016
Estimated Operator Cost	2.06229 (100%)
Estimated CPU Cost	0.0221262
Estimated Subtree Cost	2.06229
Number of Executions	1
Estimated Number of Executions	1
Estimated Number of Rows	82.1249
Estimated Row Size	78 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	False
Node ID	0

Predicate
[DB1].[Person].[Contact].[LastName]=CONVERT_IMPLICIT(nvarchar(4000),[@1],0)
Object
[DB1].[Person].[Contact].[PK_Contact_ContactID]
Output List
[DB1].[Person].[Contact].ContactID, [DB1].[Person].[Contact].EmailAddress, [DB1].[Person].[Contact].LastName

You need to reduce the amount of time it takes to complete the query. You must achieve this goal as quickly as possible. What should you do?

A. Reorganize the index.

B. Update statistics.

C. Create an index on LastName.

D. Rebuild the index.

Correct Answer: C

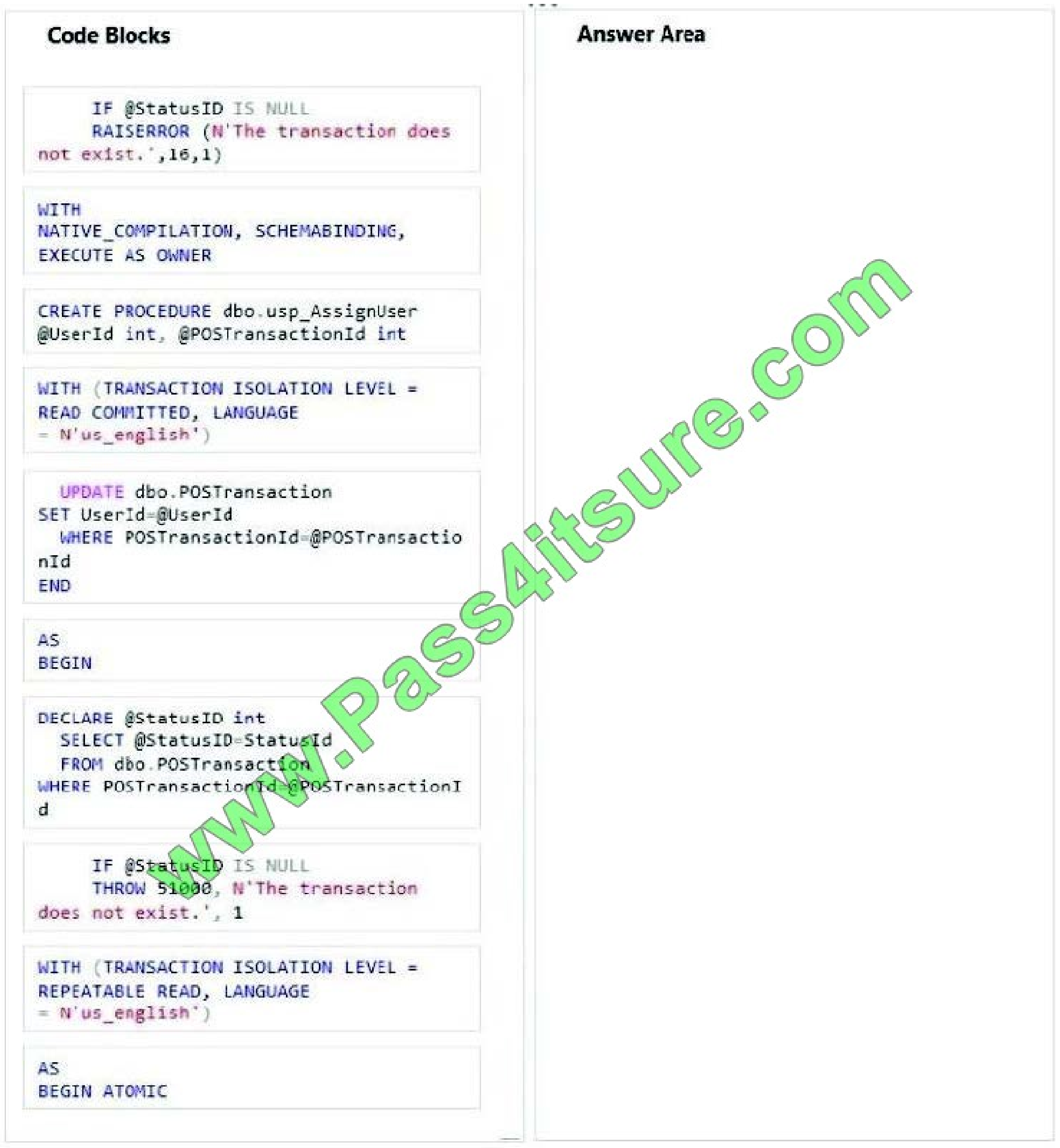
QUESTION 3

You 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.

Select and Place:



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
```

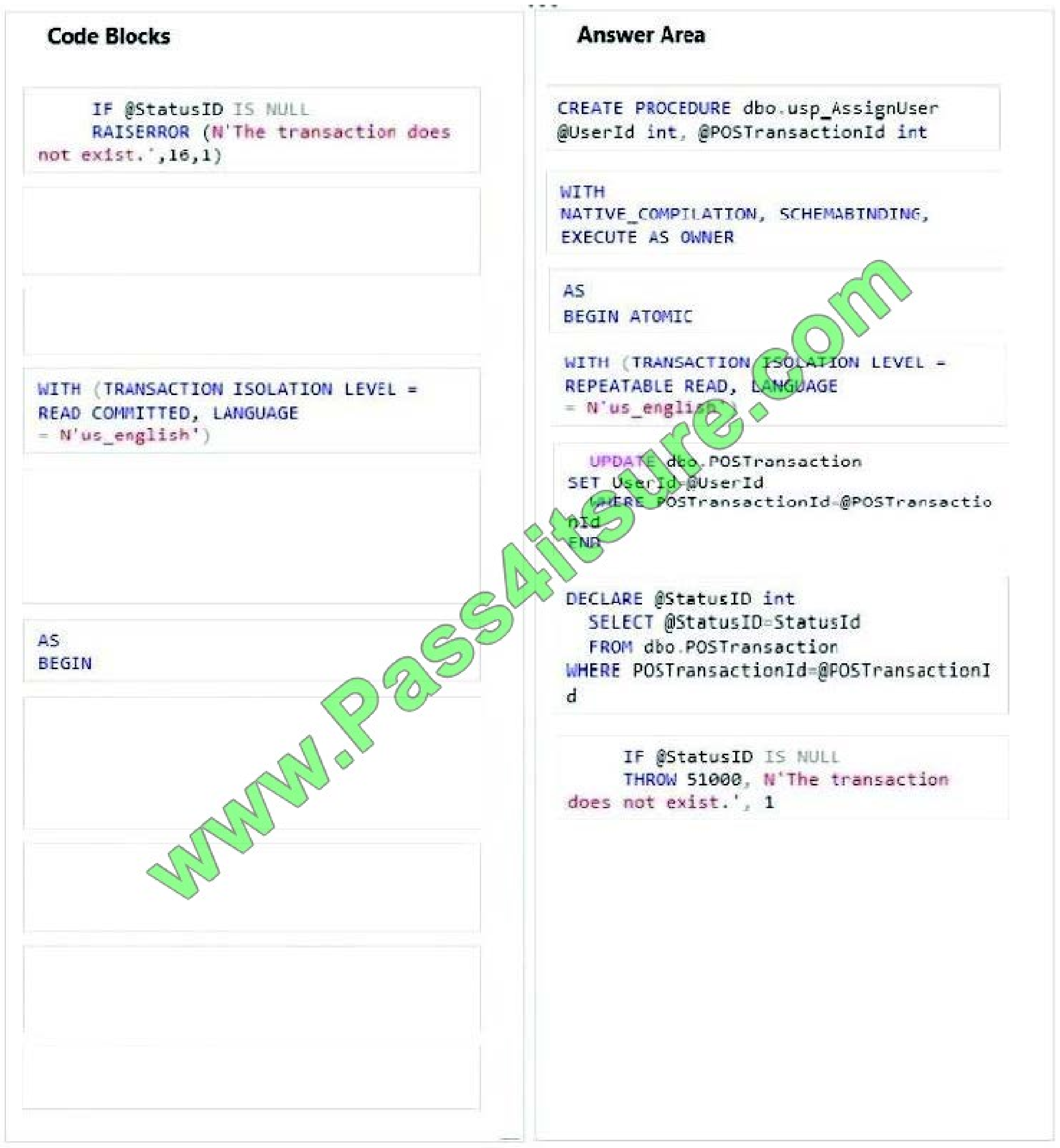
```
     IF @StatusID IS NULL
     THROW 51000, N'The transaction
does not exist.', 1
```

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

```
AS
BEGIN ATOMIC
```

Answer Area

Correct Answer:



Code Blocks

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

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

```
AS
BEGIN
```

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
```

Note:

*

From scenario: 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.

*

Example:

create procedure dbo.OrderInsert(@OrdNo integer, @CustCode nvarchar(5)) with native_compilation, schemabinding, execute as owner as begin atomic with (transaction isolation level = snapshot, language = N\\'English\\')

declare @OrdDate datetime = getdate();

insert into dbo.Ord (OrdNo, CustCode, OrdDate) values (@OrdNo, @CustCode, @OrdDate);

end

go

*

Natively compiled stored procedures are Transact-SQL stored procedures compiled to native code that access memory-optimized tables. Natively compiled stored procedures allow for efficient execution of the queries and business logic in the stored procedure.

*

READ COMITTED versus REPEATABLE READ Read committed is an isolation level that guarantees that any data read was committed at the moment is read. It simply restricts the reader from seeing any intermediate, uncommitted, \\'dirty\\' read. IT makes no promise whatsoever that if the transaction re-issues the read, will find the Same data, data is free to change after it was read.

Repeatable read is a higher isolation level, that in addition to the guarantees of the read committed level, it also guarantees that any data read cannot change, if the transaction reads the same data again, it will find the previously read data in place, unchanged, and available to read.

*

Both RAISERROR and THROW statements are used to raise an error in Sql Server. The journey of RAISERROR started from Sql Server 7.0, where as the journey of THROW statement has just began with Sql Server 2012. obviously, Microsoft suggesting us to start using THROW statement instead of RAISERROR.

*

Explicit transactions. The user starts the transaction through an explicit BEGIN TRAN or BEGIN ATOMIC. The transaction is completed following the corresponding COMMIT and ROLLBACK or END (in the case of an atomic block).

THROW statement seems to be simple and easy to use than RAISERROR.

QUESTION 4

You discover a sudden increase in processor utilization on a server that has SQL Server installed.

You need to correlate server performance and database activity for an extended time period.

Which two tools should you use? Each correct answer presents part of the solution.

A. Activity Monitor

B. Performance Monitor

C. SQL Server Profiler

D. sp_who2

E. SQL Server Extended Events

Correct Answer: BE

B: The Performance Monitor side, we have a few SQL Server monitoring tools AKA counters that can be used when troubleshooting CPU performance. The following counters are simple and easy to use:

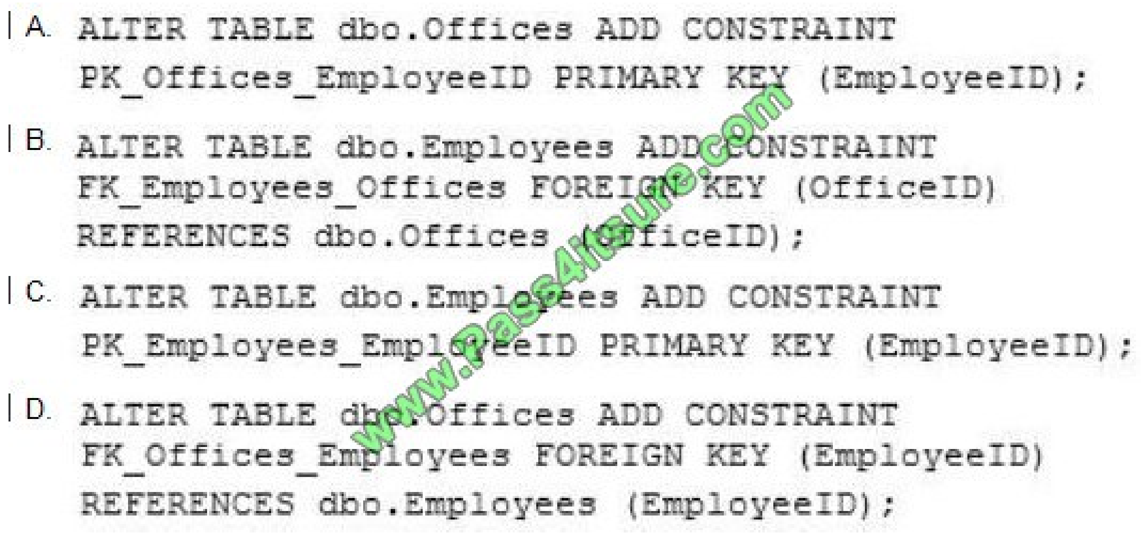
Processor % Processor Time ==

References: https://www.sqlshack.com/sql-server-monitoring-tool-for-cpu-performance/

QUESTION 5

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

http://msdn.microsoft.com/en-us/library/ms189049.aspx

Submit A Ticket