70-762$^{Q\&As}$

Developing SQL Databases

Pass Microsoft 70-762 Exam with 100% Guarantee

Following Questions and Answers are all new published by Microsoft Official Exam Center

QUESTION 1

You have multiple stored procedures inside a transaction.

You need to ensure that all the data modified by the transaction is rolled back if a stored procedure causes a deadlock or times out.

What should you do?

A. Use the NOLOCK option.

B. Execute the DBCC UPDATEUSAGE statement.

C. Use the max worker threads option.

D. Use a table-valued parameter.

E. Set SET ALLOW_SNAPSHOT_ISOLATION to ON.

F. Set SET XACT_ABORT to ON.

G. Execute the ALTER TABLE T1 SET (LOCK_ESCALATION = AUTO); statement.

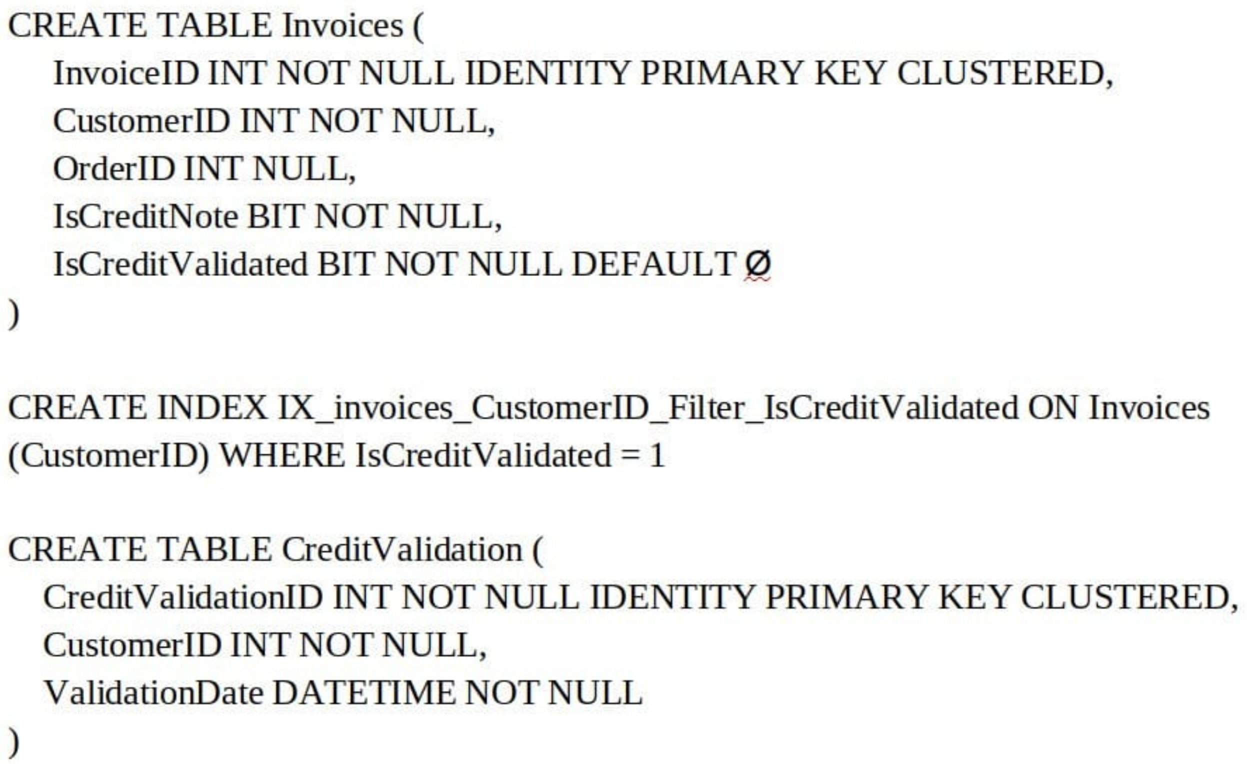
H. Use the OUTPUT parameters.

Correct Answer: F

When SET XACT_ABORT is ON, if a Transact-SQL statement raises a run-time error, the entire transaction is terminated and rolled back. References: https://docs.microsoft.com/en-us/sql/t-sql/statements/set-xact-abort-transact-sql?view=sql-server-2017

QUESTION 2

You are optimizing the performance of a batch update process. You have tables and indexes that were created by running the following Transact-SQL statements:

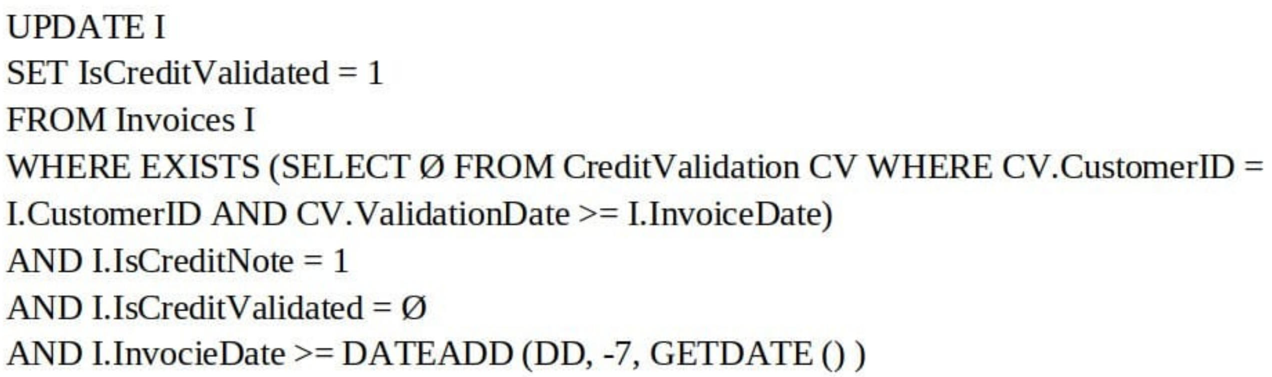


```
CREATE TABLE Invoices (
    InvoiceID INT NOT NULL IDENTITY PRIMARY KEY CLUSTERED,
    CustomerID INT NOT NULL,
    OrderID INT NULL,
    IsCreditNote BIT NOT NULL,
    IsCreditValidated BIT NOT NULL DEFAULT Ø
)

CREATE INDEX IX_invoices_CustomerID_Filter_IsCreditValidated ON Invoices
(CustomerID) WHERE IsCreditValidated = 1

CREATE TABLE CreditValidation (
   CreditValidationID INT NOT NULL IDENTITY PRIMARY KEY CLUSTERED,
   CustomerID INT NOT NULL,
   ValidationDate DATETIME NOT NULL
)
```

The following query runs nightly to update the isCreditValidated field:



```
UPDATE I
SET IsCreditValidated = 1
FROM Invoices I
WHERE EXISTS (SELECT Ø FROM CreditValidation CV WHERE CV.CustomerID =
I.CustomerID AND CV.ValidationDate >= I.InvoiceDate)
AND I.IsCreditNote = 1
AND I.IsCreditValidated = Ø
AND I.InvocieDate >= DATEADD (DD, -7, GETDATE () )
```

You review the database and make the following observations:

Most of the IsCreditValidated values in the Invoices table are set to a value of 1.

There are many unique InvoiceDate values.

The CreditValidation table does not have an index.

Statistics for the index IX_invoices_CustomerID_Filter_IsCreditValidated indicate there are no individual seeks but multiple individual updates.

You need to ensure that any indexes added can be used by the update query. If the IX_invoices_CustomerId_Filter_IsCreditValidated index cannot be used by the query, it must be removed. Otherwise, the query must be modified to use with the index.

Which three actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. Add a filtered nonclustered index to Invoices on InvoiceDate that selects where IsCreditNote= 1 and IsCreditValidated = 0.

B. Rewrite the update query so that the condition for IsCreditValidated = 0 precedes the condition for IsCreditNote = 1.

C. Create a nonclustered index for invoices in IsCreditValidated, InvoiceDate with an include statement using IsCreditNote and CustomerID.

D. Add a nonclustered index for CreditValidation on CustomerID.

E. Drop the IX_invoices_CustomerId_Filter_IsCreditValidatedIndex.

Correct Answer: ABE

A filtered index is an optimized nonclustered index especially suited to cover queries that select from a well-defined subset of data. It uses a filter predicate to index a portion of rows in the table. A well-designed filtered index can improve query performance as well as reduce index maintenance and storage costs compared with full-table indexes.

References: https://docs.microsoft.com/en-us/sql/relational-databases/indexes/create-filtered-indexes

QUESTION 3

Note: this question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in the series.

Information and details provided in a question apply only to that question.

You are developing an application to track customer sales.

You need to create a database object that meets the following requirements:

Launch when table data is modified.

Evaluate the state a table before and after a data modification and take action based on the difference.

Prevent malicious or incorrect table data operations.

Prevent changes that violate referential integrity by cancelling the attempted data modification.

Run managed code packaged in an assembly that is created in the Microsoft.NET Framework and located into Microsoft SQL Server.

What should you create?

A. extended procedure

B. CLR procedure

C. user-defined procedure

D. DDL trigger

E. scalar-valued function

F. table-valued function

Correct Answer: B

You can create a database object inside SQL Server that is programmed in an assembly created in the Microsoft .NET Framework common language runtime (CLR). Database objects that can leverage the rich programming model provided

by the CLR include DML triggers, DDL triggers, stored procedures, functions, aggregate functions, and types.

Creating a CLR trigger (DML or DDL) in SQL Server involves the following steps:

Define the trigger as a class in a .NETFramework-supported language. For more information about how to program triggers in the CLR, see CLR Triggers. Then, compile the class to build an assembly in the .NET Framework using the

appropriate language compiler.

Register the assembly in SQL Server using the CREATE ASSEMBLY statement. For more information about assemblies in SQL Server, see Assemblies (Database Engine).

Create the trigger that references the registered assembly.

References: https://msdn.microsoft.com/en-us/library/ms179562.aspx

QUESTION 4

Note: This question is part of a series of questions that present the same scenario. Each question in this series contains a unique solution. Determine whether the solution meets the stated goals.

You are developing a new application that uses a stored procedure. The stored procedure inserts thousands of records as a single batch into the Employees table.

Users report that the application response time has worsened since the stored procedure was updated. You examine disk-related performance counters for the Microsoft SQL Server instance and observe several high values that include a

disk performance issue. You examine wait statistics and observe an unusually high WRITELOG value.

You need to improve the application response time.

Solution: You add a unique clustered index to the Employees table.

Does the solution meet the goal?

A. Yes

B. No

Correct Answer: A

References: https://msdn.microsoft.com/en-us/library/ms190457.aspx

QUESTION 5

Note: This question is part of a series of questions that use the same or similar answer choices. An answer choice may be correct for more than one question in the series. Each question is independent of the other questions in this series.

Information and details provided in a question apply only to that question.

You have a Microsoft SQL Server database named DB1 that contains the following tables:

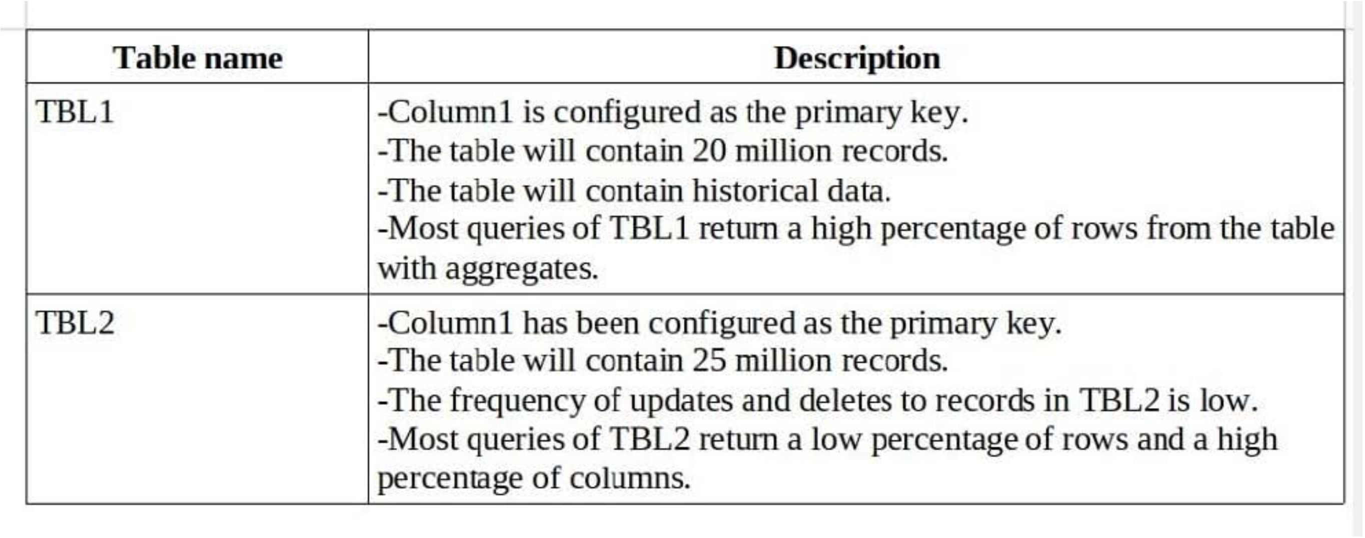


Table name	Description
TBL1	-Column1 is configured as the primary key. -The table will contain 20 million records. -The table will contain historical data. -Most queries of TBL1 return a high percentage of rows from the table with aggregates.
TBL2	-Column1 has been configured as the primary key. -The table will contain 25 million records. -The frequency of updates and deletes to records in TBL2 is low. -Most queries of TBL2 return a low percentage of rows and a high percentage of columns.

There are no foreign key relationships between TBL1 and TBL2.

You need to minimize the amount of time required for queries that use data from TB1 and TBL2 to return data.

What should you do?

A. Create clustered indexes on TBL1 and TBL2.

B. Create a clustered index on TBL1. Create a nonclustered index on TBL2 and add the most frequently queried column as included columns.

C. Create a nonclustered index on TBL2 only.

D. Create UNIQUE constraints on both TBL1 and TBL2. Create a partitioned view that combines columns from TBL1 and TBL2.

E. Drop existing indexes on TBL1 and then create a clustered columnstore index. Create a nonclustered columnstore index on TBL1. Create a nonclustered index on TBL2.

F. Drop existing indexes on TBL1 and then create a clustered columnstore index. Create a nonclustered columnstore index on TBL1.Make no changes to TBL2.

G. Create CHECK constraints on both TBL1 and TBL2. Create a partitioned view that combines columns from TBL1 and TBL2.

H. Create an indexed view that combines columns from TBL1 and TBL2.

Correct Answer: G

A partitioned view is a view defined by a UNION ALL of member tables structured in the same way, but stored separately as multiple tables in either the same instance of SQL Server or in a group of autonomous instances of SQL Server

servers, called federated database servers.

Conditions for Creating Partitioned Views Include:

The select list

All columns in the member tables should be selected in the column list of the view definition.

The columns in the same ordinal position of each select list should be of the same type, including collations. It is not sufficient for the columns to be implicitly convertible types, as is generally the case for UNION.

Also, at least one column (for example) must appear in all the select lists in the same ordinal position. This should be defined in a way that the member tables T1, ..., Tn have CHECK constraints C1, ..., Cn defined on ,

respectively.

References:

https://docs.microsoft.com/en-us/sql/t-sql/statements/create-view-transact-sql

Latest 70-762 Dumps 70-762 PDF Dumps 70-762 Study Guide