Microsoft.Premium.70-761.by.VCEplus.82q

☑**VCEplus**
Free Practice Exam - IT Certifications

Exam Code: 70-761
Exam Name: Querying Data with Transact-SQL
Certification Provider: Microsoft
Corresponding Certification: MCSA: SQL 2016
Website: www.vceplus.com
Free Exam: https://vceplus.com/exam-70-761/

Questions & Answers Exam Engine is rigorously checked before being put up for sale. We make sure there is nothing irrelevant in 70-761 exam products and you get latest questions. We strive to deliver the best 70-761 exam product for top grades in your first attempt.

VCE to PDF Converter : https://vceplus.com/vce-to-pdf/
Facebook: https://www.facebook.com/VCE.For.All.VN/
Twitter : https://twitter.com/VCE_Plus
Google+ : https://plus.google.com/+Vcepluscom
LinkedIn : https://www.linkedin.com/company/vceplus

Exam A

QUESTION 1
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You create a table named Products by running the following Transact-SQL statement:

```
CREATE TABLE Products (
    ProductID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
    ProductName nvarchar(100) NULL,
    UnitPrice decimal(18, 2) NOT NULL,
    UnitsInStock int NOT NULL,
    UnitsOnOrder int NULL
)
```

You have the following stored procedure:

```
CREATE PROCEDURE InsertProduct
    @ProductName nvarchar(100),
    @UnitPrice decimal(18,2),
    @UnitsInStock int,
    @UnitsOnOrder int
AS
BEGIN
    INSERT INTO Products(ProductName,ProductPrice,ProductsInStock,ProductsOnOrder)
    VALUES (@ProductName,@UnitPrice,@UnitsInStock,@UnitsOnOrder)
END
```

You need to modify the stored procedure to meet the following new requirements:
-Insert product records as a single unit of work.
-Return error number 51000 when a product fails to insert into the database.
-If a product record insert operation fails, the product information must not be permanently written to the database
Solution: You run the following Transact-SQL statement:

```
ALTER PROCEDURE InsertProduct
@ProductName nvarchar(100),
@UnitPrice decimal(18,2),
@UnitsInStock int,
@UnitsOnOrder int
AS
BEGIN
    SET XACT_ABORT ON
    BEGIN TRY
        BEGIN TRANSACTION
            INSERT INTO Products(ProductName,ProductPrice,ProductsInStock,ProductsOnOrder)
            VALUES (@ProductName,@UnitPrice,@UnitsInStock,@UnitsOnOrder)
        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        IF XACT_STATE() <> 0 ROLLBACK TRANSACTION
        THROW 51000, 'The product could not be created.', 1
    END CATCH
END
```

Does the solution meet the goal?

A. Yes

B. No

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
With X_ABORT ON the INSERT INTO statement and the transaction will be rolled back when an error is raised, it would then not be possible to ROLLBACK it again in the IF XACT_STATE() <> O ROLLACK TRANSACTION statmen.
Note: A transaction is correctly defined forthe INSERT INTO ..VALUES statement, and if there is an error in the transaction it will be caught ant he transaction will be rolled back, finally an error 51000 will be raised.
Note: When SET XACT_ABORT is ON, if a Transact-SQL statement raises a run-time error, the entire transaction is terminated and rolled back.
XACT_STATE is a scalar function that reports the user transaction state of a current running request. XACT_STATE indicates whether the request has an active user transaction, and whether the transaction is capable of being committed.
The states of XACT_STATE are:
0There is no active user transaction for the current request.
1The current request has an active user transaction. The request can perform any actions, including writing data and committing the transaction.
2The current request has an active user transaction, but an error has occurred that has caused the transaction to be classified as an uncommittable transaction.
References:
https://msdn.microsoft.com/en-us/library/ms188792.aspx
https://msdn.microsoft.com/en-us/library/ms189797.aspx

QUESTION 2
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You create a table named Products by running the following Transact-SQL statement:

```
CREATE TABLE Products (
    ProductID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
    ProductName nvarchar(100) NULL,
    UnitPrice decimal(18, 2) NOT NULL,
    UnitsInStock int NOT NULL,
    UnitsOnOrder int NULL
)
```

You have the following stored procedure:

```
CREATE PROCEDURE InsertProduct
    @ProductName nvarchar(100),
    @UnitPrice decimal(18,2),
    @UnitsInStock int,
    @UnitsOnOrder int
AS
BEGIN
    INSERT INTO Products(ProductName,ProductPrice,ProductsInStock,ProductsOnOrder)
    VALUES (@ProductName,@UnitPrice,@UnitsInStock,@UnitsOnOrder)
END
```

You need to modify the stored procedure to meet the following new requirements:
-Insert product records as a single unit of work.
-Return error number 51000 when a product fails to insert into the database.
-If a product record insert operation fails, the product information must not be permanently written to the database
Solution: You run the following Transact-SQL statement:

```
ALTER PROCEDURE InsertProduct
@ProductName nvarchar(100),
@UnitPrice decimal(18,2),
@UnitsInStock int,
@UnitsOnOrder int
AS
BEGIN
    BEGIN TRY
        BEGIN TRANSACTION
            INSERT INTO Products(ProductName,ProductPrice,ProductsInStock,ProductsOnOrder)
            VALUES (@ProductName,@UnitPrice,@UnitsInStock,@UnitsOnOrder)
        COMMIT TRANSACTION
    END TRY
    BEGIN CATCH
        IF @@TRANCOUNT > 0 ROLLBACK TRANSACTION
            IF @@ERROR = 51000
                THROW
    END CATCH
END
```

Does the solution meet the goal?

A. Yes

B. No

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
A transaction is correctly defined for the INSERT INTO ..VALUES statement, and if there is an error in the transaction it will be caught ant he transaction will be rolled back. However, error number 51000 will not be returned, as it is only used in an IF @ERROR = 51000 statement.
Note: @@TRANCOUNT returns the number of BEGIN TRANSACTION statements that have occurred on the current connection.
References: https://msdn.microsoft.com/en-us/library/ms187967.aspx

QUESTION 3
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You create a table named Products by running the following Transact-SQL statement:

```
CREATE TABLE Products (
    ProductID int IDENTITY(1,1) NOT NULL PRIMARY KEY,
    ProductName nvarchar(100) NULL,
    UnitPrice decimal(18, 2) NOT NULL,
    UnitsInStock int NOT NULL,
    UnitsOnOrder int NULL
)
```

You have the following stored procedure:

```
CREATE PROCEDURE InsertProduct
    @ProductName nvarchar(100),
    @UnitPrice decimal(18,2),
    @UnitsInStock int,
    @UnitsOnOrder int
AS
BEGIN
    INSERT INTO Products(ProductName,ProductPrice,ProductsInStock,ProductsOnOrder)
    VALUES (@ProductName,@UnitPrice,@UnitsInStock,@UnitsOnOrder)
END
```

You need to modify the stored procedure to meet the following new requirements:
-Insert product records as a single unit of work.
-Return error number 51000 when a product fails to insert into the database.

-If a product record insert operation fails, the product information must not be permanently written to the database
Solution: You run the following Transact-SQL statement:

```
ALTER PROCEDURE InsertProduct
@ProductName nvarchar(100),
@UnitPrice decimal(18,2),
@UnitsInStock int,
@UnitsOnOrder int
AS
BEGIN
    BEGIN TRY
        INSERT INTO Products(ProductName,ProductPrice,ProductsInStock,ProductsOnOrder)
            VALUES (@ProductName,@UnitPrice,@UnitsInStock,@UnitsOnOrder)
    END TRY
    BEGIN CATCH
        THROW 51000, 'The product could not be created.', 1
        END CATCH
END
```

Does the solution meet the goal?

A. Yes
B. No

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
If the INSERT INTO statement raises an error, the statement will be caught and an error 51000 will be thrown. In this case no records will have been inserted.
Note:
You can implement error handling for the INSERT statement by specifying the statement in a TRY...CATCH construct.
If an INSERT statement violates a constraint or rule, or if it has a value incompatible with the data type of the column, the statement fails and an error message is returned.
References: https://msdn.microsoft.com/en-us/library/ms174335.aspx

QUESTION 4
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.
You create a table named Customer by running the following Transact-SQL statement:

```
CREATE TABLE Customer (
    CustomerID int IDENTITY(1,1) PRIMARY KEY,
    FirstName varchar(50) NULL,
    LastName varchar(50) NOT NULL,
    DateOfBirth date NOT NULL,
    CreditLimit money CHECK (CreditLimit < 10000),
    TownID int NULL REFERENCES dbo.Town(TownID),
    CreatedDate datetime DEFAULT(Getdate())
)
```

You must insert the following data into the Customer table:

Record	First name	Last name	Date of Birth	Credit limit	Town ID	Created date
Record 1	Yvonne	McKay	1984-05-25	9,000	no town details	current date and time
Record 2	Jossef	Goldberg	1995-06-03	5,500	no town details	current date and time

You need to ensure that both records are inserted or neither record is inserted
Solution: You run the following Transact-SQL statement:

```
INSERT INTO Customer (FirstName, LastName, DateOfBirth, CreditLimit, CreatedDate)
VALUES ('Yvonne', 'McKay', '1984-05-25', 9000, GETDATE())
INSERT INTO Customer (FirstName, LastName, DateOfBirth, CreditLimit, CreatedDate)
VALUES ('Jossef', 'Goldberg', '1995-06-03', 5500, GETDATE())
GO
```

Does the solution meet the goal?

A. Yes

B. No

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
As there are two separate INSERT INTO statements we cannot ensure that both or neither records is inserted

QUESTION 5
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique
solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have
a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the

review screen.
You create a table named Customer by running the following Transact-SQL statement:

```
CREATE TABLE Customer (
    CustomerID int IDENTITY(1,1) PRIMARY KEY,
    FirstName varchar(50) NULL,
    LastName varchar(50) NOT NULL,
    DateOfBirth date NOT NULL,
    CreditLimit money CHECK (CreditLimit < 10000),
    TownID int NULL REFERENCES dbo.Town(TownID),
    CreatedDate datetime DEFAULT(Getdate())
)
```

You must insert the following data into the Customer table:

Record	First name	Last name	Date of Birth	Credit limit	Town ID	Created date
Record 1	Yvonne	McKay	1984-05-25	9,000	no town details	current date and time
Record 2	Jossef	Goldberg	1995-06-03	5,500	no town details	current date and time

You need to ensure that both records are inserted or neither record is inserted
Solution: You run the following Transact-SQL statement:

```
INSERT INTO Customer (FirstName, LastName, DateOfBirth, CreditLimit, TownID, CreatedDate)
VALUES ('Yvonne', 'McKay', '1984-05-25', 9000, NULL, GETDATE())
INSERT INTO Customer (FirstName, LastName, DateOfBirth, CreditLimit, TownID, CreatedDate)
VALUES ('Jossef', 'Goldberg', '1995-06-03', 5500, NULL, GETDATE())
GO
```

Does the solution meet the goal?

A. Yes
B. No

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
As there are two separate INSERT INTO statements we cannot ensure that both or neither records is inserted

QUESTION 6
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique
solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have

a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You create a table named Customer by running the following Transact-SQL statement:

```
CREATE TABLE Customer (
    CustomerID int IDENTITY(1,1) PRIMARY KEY,
    FirstName varchar(50) NULL,
    LastName varchar(50) NOT NULL,
    DateOfBirth date NOT NULL,
    CreditLimit money CHECK (CreditLimit < 10000),
    TownID int NULL REFERENCES dbo.Town(TownID),
    CreatedDate datetime DEFAULT(Getdate())
)
```

You must insert the following data into the Customer table:

Record	First name	Last name	Date of Birth	Credit limit	Town ID	Created date
Record 1	Yvonne	McKay	1984-05-25	9,000	no town details	current date and time
Record 2	Jossef	Goldberg	1995-06-03	5,500	no town details	current date and time

You need to ensure that both records are inserted or neither record is inserted
Solution: You run the following Transact-SQL statement:

```
INSERT INTO dbo.Customer (FirstName, LastName, DateOfBirth, CreditLimit)
VALUES ('Yvonne', 'McKay', '1984-05-25', 9000), ("Jossef', 'Goldberg', '1995-06-03', 5500)
```

Does the solution meet the goal?

A. Yes
B. No

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
With the INSERT INTO. VALUES statement we can insert both values with just one statement. This ensures that both records or neither is inserted.
References: https://msdn.microsoft.com/en-us/library/ms174335.aspx

QUESTION 7
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique

solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have a database that tracks orders and deliveries for customers in North America. The database contains the following tables:

Sales. Customers

Column	Data type	Notes
CustomerID	int	primary key
CustomerCategoryID	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow new values
StandardDiscountPercentage	int	does not allow new values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow new values
DeliveryLocation	geography	does not allow new values
PhoneNumber	nvarchar(20)	does not allow new values data is formatted as follows: 425-555-0187

Application.Cities

Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Sales.CustomerCategories

Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

The company's development team is designing a customer directory application. The application must list customers by the area code of their phone number. The area code is defined as the first three characters of the phone number.

The main page of the application will be based on an indexed view that contains the area and phone number for all customers.
You need to return the area code from the PhoneNumber field.
Solution: You run the following Transact-SQL statement:

```
CREATE FUNCTION AreaCode (
    @phoneNumber nvarchar(20)
)
RETURNS
TABLE
WITH SCHEMABINDING
AS
RETURN (
    SELECT TOP 1 @phoneNumber as PhoneNumber, VALUE as AreaCode
    FROM STRING_SPLIT(@phoneNumber, '-')
)
```

Does the solution meet the goal?

A. Yes

B. No

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
As the result of the function will be used in an indexed view we should use schemabinding
References: https://sqlstudies.com/2014/08/06/schemabinding-what-why/

QUESTION 8
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique
solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have
a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the
review screen.
You have a database that tracks orders and deliveries for customers in North America. The database contains the following tables:
Sales.Customers

Column	Data type	Notes
CustomerID	int	primary key
CustomerCategoryID	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow new values
StandardDiscountPercentage	int	does not allow new values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow new values
DeliveryLocation	geography	does not allow new values
PhoneNumber	nvarchar(20)	does not allow new values data is formatted as follows: 425-555-0187

Application.Cities

Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Sales.CustomerCategories

Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

The company's development team is designing a customer directory application. The application must list customers by the area code of their phone number. The area code is defined as the first three characters of the phone number.
The main page of the application will be based on an indexed view that contains the area and phone number for all customers.
You need to return the area code from the PhoneNumber field.
Solution: You run the following Transact-SQL statement:

```
CREATE FUNCTION AreaCode (
     @phoneNumber nvarchar(20)
)
RETURNS nvarchar(10)
AS
BEGIN
     DECLARE @areaCode nvarchar(max)
     SELECT TOP 1 @areaCode = VALUE FROM STRING_SPLIT(@phoneNumber, '-')
     RETURN @areaCode
END
```

Does the solution meet the goal?

A. Yes

B. No

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
As the result of the function will be used in an indexed view we should use schemabinding
References: https://sqlstudies.com/2014/08/06/schemabinding-what-why/

QUESTION 9
Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique
solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have
a correct solution.
After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the
review screen.
You have a database that tracks orders and deliveries for customers in North America. The database contains the following tables:
Sales.Customers

Column	Data type	Notes
CustomerID	int	primary key
CustomerCategoryID	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow new values
StandardDiscountPercentage	int	does not allow new values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow new values
DeliveryLocation	geography	does not allow new values
PhoneNumber	nvarchar(20)	does not allow new values data is formatted as follows: 425-555-0187

Application.Cities

Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Sales.CustomerCategories

Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

The company's development team is designing a customer directory application. The application must list customers by the area code of their phone number. The area code is defined as the first three characters of the phone number.

The main page of the application will be based on an indexed view that contains the area and phone number for all customers.

You need to return the area code from the PhoneNumber field.

Solution: You run the following Transact-SQL statement:

```
CREATE FUNCTION AreaCode (
    @phoneNumber nvarchar(20)
)
RETURNS nvarchar(10)
WITH SCHEMABINDING
AS
BEGIN
    DECLARE @areaCode nvarchar(max)
    SELECT @areaCode = value FROM STRING_SPLIT(@phoneNumber, '-')
    RETURN @areaCode
END
```

Does the solution meet the goal?

A. Yes
B. No

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
The variable max, in the line DECLARE @areaCode nvarchar(max), is not defined

QUESTION 10
Note: This question is part of a series of questions that use the same scenario. For your convenience, the scenario is repeated in each
question. Each question presents a different goal and answer choices, but the text of the scenario is exactly the same in each
question in this series.
You query a database that includes two tables: Project and Task. The Project table includes the following columns:

Column name	Data type	Notes
ProjectId	int	This is a unique identifier for a project.
ProjectName	varchar(100)	
StartTime	datetime2(7)	
EndTime	datetime2(7)	A null value indicates the project is not finished yet.
UserId	int	Identifies the owner of the project.

Column name	Data type	Notes
TaskId	int	This is a unique identifier for a task.
TaskName	varchar(100)	A nonclustered index exists for this column.
ParentTaskId	int	Each task may or may not have a parent task.
ProjectId	int	A null value indicates the task is not assigned to a specific project.
StartTime	datetime2(7)	
EndTime	datetime2(7)	A null value indicates the task is not completed yet.
UserId	int	Identifies the owner of the task.

You plan to run the following query to update tasks that are not yet started:

```
UPDATE Task SET StartTime = GETDATE() WHERE StartTime IS NULL
```

You need to return the total count of tasks that are impacted by this UPDATE operation, but are not associated with a project
What set of Transact-SQL statements should you run?

A.
```
DECLARE @startedTasks TABLE(ProjectId int)
UPDATE Task SET StartTime = GETDATE() OUTPUT deleted.ProjectId INTO @startedTasks WHERE StartTime is NULL
SELECT COUNT(*) FROM @startedTasks WHERE ProjectId IS NOT NULL
```

B.
```
DECLARE @startedTasks TABLE(TaskId int, ProjectId int)
UPDATE Task SET StartTime = GETDATE() OUTPUT deleted.TaskId, deleted.ProjectId INTO @startedTasks
WHERE StartTime is NULL
SELECT COUNT(*) FROM @startedTasks WHERE ProjectId IS NULL
```

C.

```
DECLARE @startedTasks TABLE(TaskId int)
UPDATE Task SET StartTime = GETDATE() OUTPUT inserted.TaskId, INTO @startedTasks WHERE StartTime is NULL
SELECT COUNT(*) FROM @startedTasks WHERE TaskId IS NOT NULL
```

D.

```
DECLARE @startedTasks TABLE(TaskId int)
UPDATE Task SET StartTime = GETDATE() OUTPUT deleted.TaskId, INTO @startedTasks WHERE StartTime is NULL
SELECT COUNT(*) FROM @startedTasks WHERE TaskId IS NOT NULL
```

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
The WHERE clause of the third line should be WHERE ProjectID IS NULL, as we want to count the tasks that are not associated with
a project.

QUESTION 11
You need to create an indexed view that requires logic statements to manipulate the data that the view displays
Which two database objects should you use? Each correct answer presents a complete solution.

A. a user-defined table-valued function

B. a CRL function

C. a stored procedure

D. a user-defined scalar function

Correct Answer: AC
Section: (none)
Explanation

Explanation/Reference:
Explanation
Incorrect Answers:
B: You can create a database object inside an instance of 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
common language runtime include aggregate functions, functions, stored procedures, triggers, and types.

QUESTION 12
SIMULATION

You work for an organization that monitors seismic activity around volcanos. You have a table named GroundSensors. The table stored data collected from seismic sensors. It includes the columns describes in the following table:

Name	Data Type	Notes
SensorID	int	primary key
Location	geography	do not allow null values
Tremor	int	do not allow null values
NormalizedReading	float	allow null values

The database also contains a scalar value function named NearestMountain that returns the name of the mountain that is nearest to the sensor.
You need to create a query that shows the average of the normalized readings from the sensors for each mountain. The query must
meet the following requirements:
-Include the average normalized readings and nearest mountain name.
-Exclude sensors for which no normalized reading exists.
-Exclude those sensors with value of zero for tremor.
Construct the query using the following guidelines:
-Use one part names to reference tables, columns and functions.
-Do not use parentheses unless required.
-Do not use aliases for column names and table names.
-Do not surround object names with square brackets.

Keywords

ADD	EXIT	PROC
ALL	EXTERNAL	PROCEDURE
ALTER	FETCH	PUBLIC
AND	FILE	RAISERROR
ANY	FILLFACTOR	READ
AS	FORFOREIGN	READTEXT
ASC	FREETEXT	RECONFIGURE
AUTHORIZATION	FREETEXTTABLE	REFERENCES
BACKUP	FROM	REPLICATION
BEGIN	FULL	RESTORE
BETWEEN	FUNCTION	RESTRICT
BREAK	GOTO	RETURN
BROWSE	GRANT	REVERT
BULK	GROUP	REVOKE
BY	HAVING	RIGHT
CASCADE	HOLDLOCK	ROLLBACK
CASE	IDENTITY	ROWCOUNT
CHECK	IDENTITY_INSERT	ROWGUIDCOL
CHECKPOINT	IDENTITYCOL	RULE
CLOSE	IF	SAVE
CLUSTERED	IN	SCHEMA
COALESCE	INDEX	SECURITYAUDIT
COLLATE	INNER	SELECT
COLUMN	INSERT	SEMANTICKEYPHRASETABLE
COMMIT	INTERSECT	SEMANTICSIMILARITYDETAILSTABLE

COMPUTE	INTO	SEMANTICSIMILARITYTABLE
CONCAT	IS	SESSION_USER
CONSTRAINT	JOIN	SET
CONTAINS	KEY	SETUSER
CONTAINSTABLE	KILL	SHUTDOWN
CONTINUE	LEFT	SOME
CONVERT	LIKE	STATISTICS
CREATE	LINENO	SYSTEM_USER
CROSS	LOAD	TABLE
CURRENT	MERGE	TABLESAMPLE
CURRENT_DATE	NATIONAL	TEXTSIZE
CURRENT_TIME	NOCHECK	THEN
CURRENT_TIMESTAMP	NONCLUSTERED	TO
CURENT_USER	NOT	TOP
CURSOR	NULL	TRAN
DATABASE	NULLIF	TRANSACTION
DBCC	OF	TRIGGER
DEALLOCATE	OFF	TRUNCATE
DECLARE	OFFSETS	TRY_CONVERT
DEFAULT	ON	TSEQUAL
DELETE	OPEN	UNION
DENY	OPENDATASOURCE	UNIQUE
DESC	OPENQUERY	UNPIVOT
DISK	OPENROWSET	UPDATE
DISTINCT	OPENXML	UPDATETEXT
DISTRIBUTED	OPTION	USE
DOUBLE	OR	USER

DUMP	OUTER	VARYING
ELSE	OVER	VIEW
END	PERCENT	WAITFOR
ERRLVL	PIVOT	WHEN
ESCAPE	PLAN	WHERE
ESCEPT	PRECISION	WHILE
EXEC	PRIMARY	WITH
EXECUTE	PRINT	WITHIN GROUP
EXISTS		WRITETEXT

Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the code that has been provided as well as below it.

```
1  select
```

Use the Check Syntax button to verify your work. Any syntax or spelling errors will be reported by line and character position.

Your Response:

type here

A. "See solution below"

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
SELECT Average(NormalizedReading), NearestMountain(SensorID)
FROM GroundSensors
GROUP BY NearestMountain(SensorID)
WHERE TREMOR IS NOT 0 AND NormalizedReading IS NOT NULL

Ex:
GROUP BY is a SELECT statement clause that divides the query result into groups of rows, usually for the purpose of performing one or more aggregations on each group. The SELECT statement returns one row per group.
Reference: https://msdn.microsoft.com/en-us/library/ms177673.aspx

QUESTION 13
SIMULATION

You create a table named Sales.Orders by running the following Transact-SQL statement:

```
CREATE TABLE Sales.Orders (
    OrderID int NOT NULL,
    OrderDate date NULL,
    ShippedDate date NULL,
    Status varchar(10),
    CONSTRAINT PK_ORDERS PRIMARY KEY CLUSTERED
)
```

You need to write a query that meets the following requirements:
-removes orders from the table that were placed before January 1,2012
-uses the date format of YYYYMMDD
-ensures that the order has been shipped before deleting the record
Construct the query using the following guidelines:
-use one-part column names and two-part table names
-do not use functions
-do not surround object names with square brackets
-do not use variables
-do not use aliases for column names and table names

Keywords

ADD	EXIT	PROC
ALL	EXTERNAL	PROCEDURE
ALTER	FETCH	PUBLIC
AND	FILE	RAISERROR
ANY	FILLFACTOR	READ
AS	FORFOREIGN	READTEXT
ASC	FREETEXT	RECONFIGURE
AUTHORIZATION	FREETEXTTABLE	REFERENCES
BACKUP	FROM	REPLICATION
BEGIN	FULL	RESTORE
BETWEEN	FUNCTION	RESTRICT
BREAK	GOTO	RETURN
BROWSE	GRANT	REVERT
BULK	GROUP	REVOKE
BY	HAVING	RIGHT
CASCADE	HOLDLOCK	ROLLBACK
CASE	IDENTITY	ROWCOUNT
CHECK	IDENTITY_INSERT	ROWGUIDCOL
CHECKPOINT	IDENTITYCOL	RULE
CLOSE	IF	SAVE
CLUSTERED	IN	SCHEMA
COALESCE	INDEX	SECURITYAUDIT
COLLATE	INNER	SELECT
COLUMN	INSERT	SEMANTICKEYPHRASETABLE
COMMIT	INTERSECT	SEMANTICSIMILARITYDETAILSTABLE

COMPUTE	INTO	SEMANTICSIMILARITYTABLE
CONCAT	IS	SESSION_USER
CONSTRAINT	JOIN	SET
CONTAINS	KEY	SETUSER
CONTAINSTABLE	KILL	SHUTDOWN
CONTINUE	LEFT	SOME
CONVERT	LIKE	STATISTICS
CREATE	LINENO	SYSTEM_USER
CROSS	LOAD	TABLE
CURRENT	MERGE	TABLESAMPLE
CURRENT_DATE	NATIONAL	TEXTSIZE
CURRENT_TIME	NOCHECK	THEN
CURRENT_TIMESTAMP	NONCLUSTERED	TO
CURENT_USER	NOT	TOP
CURSOR	NULL	TRAN
DATABASE	NULLIF	TRANSACTION
DBCC	OF	TRIGGER
DEALLOCATE	OFF	TRUNCATE
DECLARE	OFFSETS	TRY_CONVERT
DEFAULT	ON	TSEQUAL
DELETE	OPEN	UNION
DENY	OPENDATASOURCE	UNIQUE
DESC	OPENQUERY	UNPIVOT
DISK	OPENROWSET	UPDATE
DISTINCT	OPENXML	UPDATETEXT
DISTRIBUTED	OPTION	USE
DOUBLE	OR	USER

DUMP	OUTER	VARYING
ELSE	OVER	VIEW
END	PERCENT	WAITFOR
ERRLVL	PIVOT	WHEN
ESCAPE	PLAN	WHERE
ESCEPT	PRECISION	WHILE
EXEC	PRIMARY	WITH
EXECUTE	PRINT	WITHIN GROUP
EXISTS		WRITETEXT

Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the code that has been provided as well as below it.

```
1  DELETE
```

Use the Check Syntax button to verify your work. Any syntax or spelling errors will be reported by line and character position.

Your Response:

type here

A. See solution below

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
DELETE FROM Sales.Orders
WHERE OrderDate < '2012-01-01' AND ShippedDate NOT NULL
References:
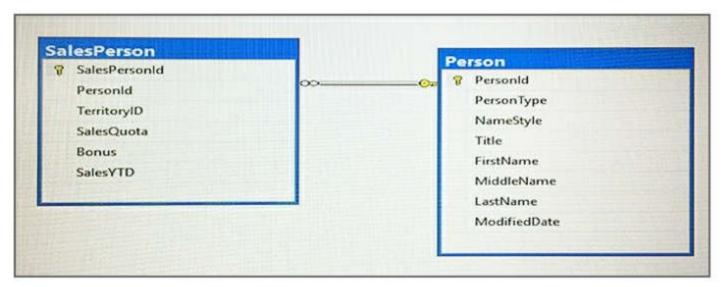https://msdn.microsoft.com/en-us/librarv/ms189835.aspx
https://msdn.microsoft.com/en-us/librarv/bb630352.aspx

QUESTION 14
SIMULATION
You have a database that contains the following tables.

You need to create a query that lists the lowest-performing salespersons based on the current year-to-date sales period. The query must meet the following requirements:
-Return a column named Fullname that includes the salesperson FirstName, a space, and then LastName.
-Include the current year-to-date sales for each salesperson.
-Display only data for the three salespersons with the lowest year-to-year sales values.
-Exclude salespersons that have no value for TerritoryID.
Construct the query using the following guidelines:
-Use the first letter of a table name as the table alias.
-Use two-part column names.
-Do not surround object names with square brackets.
-Do not use implicit joins.
-Use only single quotes for literal text.
-Use aliases only if required.

Keywords

ADD	EXIT	PROC
ALL	EXTERNAL	PROCEDURE
ALTER	FETCH	PUBLIC
AND	FILE	RAISERROR
ANY	FILLFACTOR	READ
AS	FORFOREIGN	READTEXT
ASC	FREETEXT	RECONFIGURE
AUTHORIZATION	FREETEXTTABLE	REFERENCES
BACKUP	FROM	REPLICATION
BEGIN	FULL	RESTORE
BETWEEN	FUNCTION	RESTRICT
BREAK	GOTO	RETURN
BROWSE	GRANT	REVERT
BULK	GROUP	REVOKE
BY	HAVING	RIGHT
CASCADE	HOLDLOCK	ROLLBACK
CASE	IDENTITY	ROWCOUNT
CHECK	IDENTITY_INSERT	ROWGUIDCOL
CHECKPOINT	IDENTITYCOL	RULE
CLOSE	IF	SAVE
CLUSTERED	IN	SCHEMA
COALESCE	INDEX	SECURITYAUDIT
COLLATE	INNER	SELECT
COLUMN	INSERT	SEMANTICKEYPHRASETABLE
COMMIT	INTERSECT	SEMANTICSIMILARITYDETAILSTABLE

COMPUTE	INTO	SEMANTICSIMILARITYTABLE
CONCAT	IS	SESSION_USER
CONSTRAINT	JOIN	SET
CONTAINS	KEY	SETUSER
CONTAINSTABLE	KILL	SHUTDOWN
CONTINUE	LEFT	SOME
CONVERT	LIKE	STATISTICS
CREATE	LINENO	SYSTEM_USER
CROSS	LOAD	TABLE
CURRENT	MERGE	TABLESAMPLE
CURRENT_DATE	NATIONAL	TEXTSIZE
CURRENT_TIME	NOCHECK	THEN
CURRENT_TIMESTAMP	NONCLUSTERED	TO
CURENT_USER	NOT	TOP
CURSOR	NULL	TRAN
DATABASE	NULLIF	TRANSACTION
DBCC	OF	TRIGGER
DEALLOCATE	OFF	TRUNCATE
DECLARE	OFFSETS	TRY_CONVERT
DEFAULT	ON	TSEQUAL
DELETE	OPEN	UNION
DENY	OPENDATASOURCE	UNIQUE
DESC	OPENQUERY	UNPIVOT
DISK	OPENROWSET	UPDATE
DISTINCT	OPENXML	UPDATETEXT
DISTRIBUTED	OPTION	USE
DOUBLE	OR	USER

DUMP	OUTER	VARYING
ELSE	OVER	VIEW
END	PERCENT	WAITFOR
ERRLVL	PIVOT	WHEN
ESCAPE	PLAN	WHERE
ESCEPT	PRECISION	WHILE
EXEC	PRIMARY	WITH
EXECUTE	PRINT	WITHIN GROUP
EXISTS		WRITETEXT

Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the code that has been provided as well as below it.

```
1  SELECT
2  FROM Person AS P INNER JOIN SalesPerson AS S
3  ON P.PersonID = S.SalesPersonID
4  WHERE
```

Use the Check Syntax button to verify your work. Any syntax or spelling errors will be reported by line and character position.

Your Response:

type here

>

A. See solution below

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
SELECT TOP(3) FirstName + " + LastName AS Fullname, S.SalesYTD
FROM Person as P INNER JOIN Salesperson AS S
ON P.PersonID = S.SalesPersonID
WHERE S.TerritoryID IS NOT NULL
ORDER BY S.SalesYTD

On ordering: ASC | DESC
Specifies that the values in the specified column should be sorted in ascending or descending order. ASC sorts from the lowest value to highest value. DESC sorts from highest value to lowest value. ASC is the default sort order. Null values are treated as the lowest
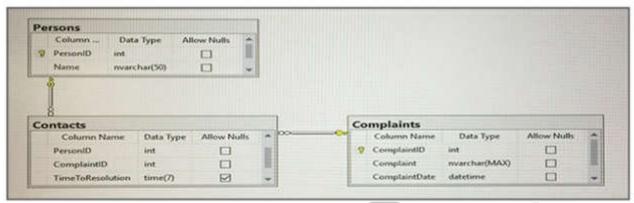
possible values.
References: https://msdn.microsoft.com/en-us/library/ms189463.aspx

QUESTION 15
SIMULATION
You have a database that contains the following tables.

You need to create a query that lists all complaints from the Complaints table, and the name of the person handling the complaints if a person is assigned. The ComplaintID must be displayed first, followed by the person name.
Construct the query using the following guidelines:
-Use two-part column names.
-Use one-part table names.
-Do not use aliases for column names or table names.
-Do not use Transact-SQL functions.
-Do not use implicit joins.
-Do not surround object names with square brackets.

Keywords

ADD	EXIT	PROC
ALL	EXTERNAL	PROCEDURE
ALTER	FETCH	PUBLIC
AND	FILE	RAISERROR
ANY	FILLFACTOR	READ
AS	FORFOREIGN	READTEXT
ASC	FREETEXT	RECONFIGURE
AUTHORIZATION	FREETEXTTABLE	REFERENCES
BACKUP	FROM	REPLICATION
BEGIN	FULL	RESTORE
BETWEEN	FUNCTION	RESTRICT
BREAK	GOTO	RETURN
BROWSE	GRANT	REVERT
BULK	GROUP	REVOKE
BY	HAVING	RIGHT
CASCADE	HOLDLOCK	ROLLBACK
CASE	IDENTITY	ROWCOUNT
CHECK	IDENTITY_INSERT	ROWGUIDCOL
CHECKPOINT	IDENTITYCOL	RULE
CLOSE	IF	SAVE
CLUSTERED	IN	SCHEMA
COALESCE	INDEX	SECURITYAUDIT
COLLATE	INNER	SELECT
COLUMN	INSERT	SEMANTICKEYPHRASETABLE
COMMIT	INTERSECT	SEMANTICSIMILARITYDETAILSTABLE

COMPUTE	INTO	SEMANTICSIMILARITYTABLE
CONCAT	IS	SESSION_USER
CONSTRAINT	JOIN	SET
CONTAINS	KEY	SETUSER
CONTAINSTABLE	KILL	SHUTDOWN
CONTINUE	LEFT	SOME
CONVERT	LIKE	STATISTICS
CREATE	LINENO	SYSTEM_USER
CROSS	LOAD	TABLE
CURRENT	MERGE	TABLESAMPLE
CURRENT_DATE	NATIONAL	TEXTSIZE
CURRENT_TIME	NOCHECK	THEN
CURRENT_TIMESTAMP	NONCLUSTERED	TO
CURENT_USER	NOT	TOP
CURSOR	NULL	TRAN
DATABASE	NULLIF	TRANSACTION
DBCC	OF	TRIGGER
DEALLOCATE	OFF	TRUNCATE
DECLARE	OFFSETS	TRY_CONVERT
DEFAULT	ON	TSEQUAL
DELETE	OPEN	UNION
DENY	OPENDATASOURCE	UNIQUE
DESC	OPENQUERY	UNPIVOT
DISK	OPENROWSET	UPDATE
DISTINCT	OPENXML	UPDATETEXT
DISTRIBUTED	OPTION	USE
DOUBLE	OR	USER

```
DUMP          OUTER           VARYING
ELSE          OVER            VIEW
END           PERCENT         WAITFOR
ERRLVL        PIVOT           WHEN
ESCAPE        PLAN            WHERE
ESCEPT        PRECISION       WHILE
EXEC          PRIMARY         WITH
EXECUTE       PRINT           WITHIN GROUP
EXISTS                        WRITETEXT
```

```
1  SELECT  Complaints.ComplaintId,
2  FROM
3  JOIN
4  JOIN
```

Part of the correct Transact-SQL has been provided in the answer area below. Enter the code in the answer area that resolves the problem and meets the stated goals or requirements. You can add code within the codethat has been provided as well as below it.
Use the Check Syntax button to verify your work. Any syntax or spelling errors will be reported by line and character position

A. See solution below

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
SELECT Complaints.ComlaintID, Persons.Name
FROM Persons
JOIN Contacts
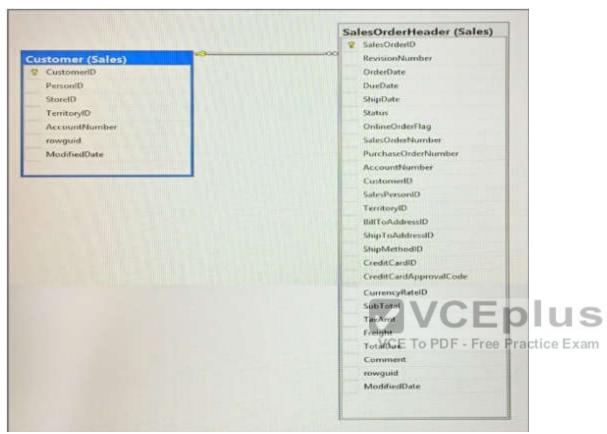ON Persons. PersonID=Contacts.PersonID
JOIN Complaints
ON Contacts. ComplaintID=Complaints. Complaint ID
References: https://technet.microsoft.com/en-us/library/ms190014(v=sql.105).aspx

QUESTION 16
You have a database that includes the tables shown in the exhibit. (Click the exhibit button.)

You need to create a list of all customers, the order ID for the last order that the customer placed, and the date that the order was placed. For customers who have not placed orders, you must substitute a zero for the order ID and 01/01/1990 for the date.
Which Transact-SQL statement should you run?

A.
```
SELECT C.CustomerID, ISNULL(SOH.SalesOrderID, 0) AS OrderID,ISNULL(MAX(OrderDate), '')
FROM Sales.Customer C LEFT OUTER JOIN Sales.SalesOrderHeader SOH
ON C.CustomerID = SOH.CustomerID
GROUP BY C.CustomerID, SOH.SalesOrderID
ORDER BY C.CustomerID
```

B.
```
SELECT C.CustomerID, SOH.SalesOrderID, MAX(OrderDate)
FROM Sales.Customer C INNER JOIN Sales.SalesOrderHeader SOH
ON C.CustomerID = SOH.CustomerID
GROUP BY C.CustomerID, SOH.SalesOrderID
ORDER BY C.CustomerID
```

C.
```
SELECT C.CustomerID, SOH.SalesOrderID, MAX(OrderDate)
FROM Sales.Customer C CROSS JOIN Sales.SalesOrderHeader SOH
ON C.CustomerID = SOH.CustomerID
GROUP BY C.CustomerID, SOH.SalesOrderID
ORDER BY C.CustomerID
```

D.
```
SELECT C.CustomerID, SOH.SalesOrderID, MAX(OrderDate)
FROM Sales.Customer C RIGHT OUTER JOIN Sales.SalesOrderHeader SOH
ON C.CustomerID = SOH.CustomerID
GROUP BY C.CustomerID, SOH.SalesOrderID
ORDER BY C.CustomerID
```

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
ISNULL Syntax: ISNULL (check_expression , replacement_value) author:"Luxemburg, Rosa"
The ISNULL function replaces NULL with the specified replacement value. The value of check_expression is returned if it is not NULL;
otherwise, replacement_value is returned after it is implicitly converted to the type of check_expression.
References: https://msdn.microsoft.com/en-us/library/ms184325.aspx

QUESTION 17
You have a database that contains the following tables:
Customer

Column name	Data type	Nullable	Default value
CustomerId	int	No	Identity property
FirstName	varchar(30)	Yes	
LastName	varchar(30)	No	
CreditLimit	money	No	

CustomerAudit

Column name	Data type	Nullable	Default value
CustomerId	int	No	
DateChanged	datetime	No	GETDATE()
OldCreditLimit	money	No	
NewCreditLimit	money	No	
ChangedBy	varchar(100)	No	SYSTEM_USER

Where the value of the CustomerID column equals 3, you need to update the value of the CreditLimit column to 1000 for the customer.
You must ensure that the change to the record in the Customer table is recorded on the CustomerAudit table.
Which Transact-SQL statement should you run?

A.
```
UPDATE Customer
SET CreditLimit = 1000
WHERE CustomerId = 3
INSERT INTO dbo.CustomerAudit (CustomerId, OldCreditLimit, NewCreditLimit)
SELECT CustomerId, CreditLimit, CreditLimit
FROM Customer
WHERE CustomerId = 3
```

B.
```
UPDATE Customer
SET CreditLimit = 1000
WHERE CustomerId = 3
INSERT INTO dbo.CustomerAudit (CustomerId, OldCreditLimit, NewCreditLimit)
SELECT CustomerId, CreditLimit, CreditLimit
FROM Customer
```

C.
```
UPDATE Customer
SET CreditLimit = 1000
OUTPUT inserted.CustomerId, inserted.CreditLimit, deleted.CreditLimit
INTO CustomerAudit (CustomerId, OldCreditLimit, NewCreditLimit)
WHERE CustomerId = 3
```

D.
```
UPDATE Customer
SET CreditLimit = 1000
OUTPUT inserted.CustomerId, deleted.CreditLimit, inserted.CreditLimit
INTO CustomerAudit (CustomerId, OldCreditLimit, NewCreditLimit)
WHERE CustomerId = 3
```

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
The OUTPUT Clause returns information from, or expressions based on, each row affected by an INSERT, UPDATE, DELETE, or MERGE statement. These results can be returned to the processing application for use in such things as confirmation messages, archiving, and other such application requirements. The results can also be inserted into a table or table variable. Additionally, you can capture the results of an OUTPUT clause in a nested INSERT, UPDATE, DELETE, or MERGE statement, and insert those results into a target table or view.
Note: If the column modified by the .RITE clause is referenced in an OUTPUT clause, the complete value of the column, either the before image in deleted.column_name or the after image in inserted.column_name, is returned to the specified column in the table variable.
Incorrect Answers:
C: The deleted.Creditlimit should be inserted in the second column, the OldCreditLimit column, not the third column.
References: https://msdn.microsoft.com/en-us/library/ms177564.aspx

QUESTION 18
Note: This question is part of a series of questions that use the same scenario. For your convenience, the scenario is repeated in each question. Each question presents a different goal and answer choices, but the text of the scenario is exactly the same in each question on this series.
You have a database that tracks orders and deliveries for customers in North America. System versioning is enabled for all tables.
The database contains the Sales.Customers, Application.Cities, and Sales.CustomerCategories tables.
Details for the Sales.Customers table are shown in the following table:

Column	Data type	Notes
CustomerId	int	primary key
CustomerCategoryId	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow values
StandardDiscountPercentage	int	does not allow values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow values
DeliveryLocation	geography	does not allow values
PhoneNumber	nvarchar(20)	does not allow values
ValidFrom	datetime2(7)	does not allow values, GENERATED ALWAYS AS ROW START
ValidTo	datetime2(7)	does not allow values, GENERATED ALWAYS AS ROW END

Details for the Application.Cities table are shown in the following table:

Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Details for the Sales.CustomerCategories table are shown in the following table:

Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

You need to create a query that meets the following requirements:
-For customers that are not on a credit hold, return the CustomerID and the latest recorded population forthe delivery city that is associated with the customer.
-For customers that are on a credit hold, return the CustomerID and the latest recorded population forthe postal city that is associated with the customer.
Which two Transact-SQL queries will achieve the goal? Each correct answer presents a complete solution.

A.

```
SELECT CustomerID, LatestRecordedPopulation
FROM Sales.Customers
CROSS JOIN Application.Citites
WHERE (IsOnCreditHold = 0 AND DeliveryCityID = CityID)
OR (IsOnCreditHold = 1 AND PostalCityID = CityID)
```

B.

```
SELECT CustomerID, LatestRecordedPopulation
FROM Sales.Customers
INNER JOIN Application.Citites AS A
ON A.CityID = IIF(IsOnCreditHold = 0, DeliveryCityID, PostalCityID)
```

C.

```
SELECT CustomerID, ISNULL(A.LatestRecordedPopulation, B.LatestRecorded Population)
FROM Sales.Customers
INNER JOIN Application.Citites AS A ON A.CityID = DeliveryCityID
INNER JOIN Application.Citites AS B ON B.CityID = PostalCityID
WHERE IsOnCreditHold = 0
```

D.

```
SELECT CustomerID,LatestRecordedPopulation,
IIF(IsOnCreditHold = 0, DeliveryCityID, PostalCityID) As CityId
FROM Sales.Customers
INNER JOIN Application.Citites AS A ON A.CityID = CityId
```

Correct Answer: A
Section: (none)
Explanation

Explanation/Reference:
Using Cross Joins
A cross join that does not have a WHERE clause produces the Cartesian product of the tables involved in the join. The size of a
Cartesian product result set is the number of rows in the first table multiplied by the number of rows in the second table.
However, if a WHERE clause is added, the cross join behaves as an inner join.
B: You can use the IIF in the ON-statement.
IIF returns one of two values, depending on whether the Boolean expression evaluates to true or false in SQL Server.
References:
https://technet.microsoft.com/en-us/library/ms190690(v=sql.1 05).aspx
https://msdn.microsoft.com/en-us/library/hh213574.aspx

QUESTION 19

Note: This question is part of a series of questions that use the same scenario. For your convenience, the scenario is repeated in each question. Each question presents a different goal and answer choices, but the text of the scenario is exactly the same in each question on this series.
You have a database that tracks orders and deliveries for customers in North America. System versioning is enabled for all tables.
The database contains the Sales.Customers, Application.Cities, and Sales.CustomerCategories tables.
Details for the Sales.Customers table are shown in the following table:

Column	Data type	Notes
CustomerId	int	primary key
CustomerCategoryId	int	foreign key to the Sales.CustomerCategories table
PostalCityID	int	foreign key to the Application.Cities table
DeliveryCityID	int	foreign key to the Application.Cities table
AccountOpenedDate	datetime	does not allow values
StandardDiscountPercentage	int	does not allow values
CreditLimit	decimal(18,2)	null values are permitted
IsOnCreditHold	bit	does not allow values
DeliveryLocation	geography	does not allow values
PhoneNumber	nvarchar(20)	does not allow values
ValidFrom	datetime2(7)	does not allow values, GENERATED ALWAYS AS ROW START
ValidTo	datetime2(7)	does not allow values, GENERATED ALWAYS AS ROW END

Details for the Application.Cities table are shown in the following table:

Column	Data type	Notes
CityID	int	primary key
LatestRecordedPopulation	bigint	null values are permitted

Details for the Sales.CustomerCategories table are shown in the following table:

Column	Data type	Notes
CustomerCategoryID	int	primary key
CustomerCategoryName	nvarchar(50)	does not allow null values

You discover an application bug that impacts customer data for records created on or after January 1,2014. In order to fix the data impacted by the bug, application programmers require a report that contains customer data as it existed on December 31,2013.
You need to provide the query for the report.
Which Transact-SQL statement should you use?

A.
```
DECLARE @sdate DATETIME, @edate DATETIME
SET @sdate = DATEFROMPARTS (2013, 12, 31)
set @edate = DATEADD(d, 1, @sdate)
SELECT * FROM Sales.Customers FOR SYSTEM_TIME ALL
WHERE ValidFrom > @sdate AND ValidTo < @edate
```

B.
```
DECLARE @sdate DATETIME, @edate DATETIME
SET @sdate = DATEFROMPARTS (2013, 12, 31)
set @edate = DATEADD(d, -1, @sdate)
SELECT * FROM Sales.Customers FOR SYSTEM_TIME BETWEEN @sdate AND @edate
```

C.
```
DECLARE @date DATE
SET @date = DATEFROMPARTS (2013, 12, 31)
SELECT * FROM Sales.Customers FOR SYSTEM_TIME AS OF @date
```

D.
```
DECLARE @date DATE
SET @date = DATEFROMPARTS (2013, 12, 31)
SELECT * FROM Sales.Customers WHERE @date BETWEEN ValidFrom AND ValidTo
```

Correct Answer: D
Section: (none)
Explanation

Explanation/Reference:
The datetime datetype defines a date that is combined with a time of day with fractional seconds that is based on a 24-hour clock.
The DATEFROMPARTS function returns a date value for the specified year, month, and day.
Incorrect Answers:
A: ValidFrom should be less (<) than @sdate AND ValidTo should be greater (>) than @edate.
B: We should add a day with DATEADD, not subtract one day.
C: We cannot compare a date to an exact datetime.
References: https://msdn.microsoft.com/en-us/library/ms187819.aspx

QUESTION 20
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 table named Products that contains information about the products that your company sells. The table contains many columns that do not always contain values.
You need to implement an ANSI standard method to convert the NULL values in the query output to the phrase "Not Applicable".
What should you implement?

A. the COALESCE function

B. a view

C. a table-valued function

D. the TRY_PARSE function

E. a stored procedure

F. the ISNULL function

G. a scalar function

H. the TRY.CONVERT function

Correct Answer: F
Section: (none)
Explanation

Explanation/Reference:
The ISNULL function replaces NULL with the specified replacement value
References: https://msdn.microsoft.com/en-us/library/ms184325.aspx

QUESTION 21
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 database that is denormalized. Users make frequent changes to data in a primary table.
You need to ensure that users cannot change the tables directly, and that changes made to the primary table also update any related tables.
What should you implement?

A. the COALESCE function

B. a view

C. a table-valued function

D. the TRY.PARSE function

E. a stored procedure

F. the ISNULL function

G. a scalar function

H. the TRY.CONVERT function

Correct Answer: B
Section: (none)
Explanation

Explanation/Reference:
Using an Indexed View would allow you to keep your base data in properly normalized tables and maintain data-integrity while giving
you the denormalized "view" of that data.
References: http://stackoverflow.com/questions/4789091/updating-redundant-denormalized-data-automatically-in-sql-server

QUESTION 22
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 database that stores sales and order information.
Users must be able to extract information from the tables on an ad hoc basis. They must also be able to reference the extracted information as a single table.
You need to implement a solution that allows users to retrieve the data required, based on variables defined at the time of the query.
What should you implement?

A. the COALESCE function

B. a view

C. a table-valued function

D. the TRY_PARSE function

E. a stored procedure

F. the ISNULL function

G. a scalar function

H. the TRY.CONVERT function

Correct Answer: C
Section: (none)
Explanation

Explanation/Reference:
User-defined functions that return a table data type can be powerful alternatives to views. These functions are referred to as table-
valued functions. A table-valued user-defined function can be used where table or view expressions are allowed in Transact-SQL
queries. While views are limited to a single SELECT statement, user-defined functions can contain additional statements that allow
more powerful logic than is possible in views.
A table-valued user-defined function can also replace stored procedures that return a single result set.
References: https://technet.microsoft.com/en-us/library/ms191165(v=sql.105).aspx