Exam Questions 70-761
Querying Data with Transact-SQL (beta)
NEW 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 have a database that contains a single table named tblVehicleRegistration. The table is defined as follows:

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VehicleId</td>
<td>int</td>
<td>the primary key for the table</td>
</tr>
<tr>
<td>RegistrationNumber</td>
<td>varchar(5)</td>
<td>a vehicle registration number that contains only letters and numbers</td>
</tr>
<tr>
<td>RegistrationDate</td>
<td>date</td>
<td>the vehicle registration date</td>
</tr>
<tr>
<td>UserId</td>
<td>int</td>
<td>an identifier for the vehicle owner</td>
</tr>
</tbody>
</table>

You run the following query:
```
SELECT UserId FROM tblVehicleRegistration
WHERE RegistrationNumber = '20012'
AND RegistrationDate > '2016-01-01'
```

The query output window displays the following error message: "Conversion failed when converting the varchar value ‘AB012’ to data type int.”

You need to resolve the error.
Solution: You modify the Transact-SQL statement as follows:
```
SELECT UserId FROM tblVehicleRegistration
WHERE RegistrationNumber = CAST('20012' AS varchar(5))
AND RegistrationDate > '2016-01-01'
```

Does the solution meet the goal?
A. Yes
B. No

Answer: B

Explanation:

NEW 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 are building a stored procedure that will be used by hundreds of users concurrently.
You need to store rows that will be processed later by the stored procedure. The object that stores the rows must meet the following requirements:
- Be indexable
- Contain up-to-date statistics
- Be able to scale between 10 and 100,000 rows

The solution must prevent users from accessing one another’s data. Solution: You create a table variable in the stored procedure.

Does this meet the goal?
A. Yes
B. No

Answer: B

NEW QUESTION 3
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. Option 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

A. Option A
NEW 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 Products by running the following Transact-SQL statement:

```sql
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:

```sql
CREATE PROCEDURE InsertProduct  
    @ProductName nvarchar(100),  
    @UnitPrice decimal(18,2),  
    @UnitsInStock int,  
    @UnitsOnOrder int  
AS  
BEGIN  
    INSERT INTO Products(ProductName, UnitPrice, UnitsInStock, UnitsOnOrder)  
    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:

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

Does the solution meet the goal?

A. Yes
B. No
NEW QUESTION 5
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.

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

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

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

<table>
<thead>
<tr>
<th>Column</th>
<th>Data type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CityID</td>
<td>int</td>
<td>primary key</td>
</tr>
<tr>
<td>LatestRecordedPopulation</td>
<td>bigint</td>
<td>null values are permitted</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Column</th>
<th>Data type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CustomerCategoryId</td>
<td>int</td>
<td>primary key</td>
</tr>
<tr>
<td>CustomerCategoryName</td>
<td>nvarchar(50)</td>
<td>does not allow null values</td>
</tr>
</tbody>
</table>

The marketing department is performing an analysis of how discount affect credit limits. They need to know the average credit limit per standard discount percentage for customers whose standard discount percentage is between zero and four. You need to create a query that returns the data for the analysis.

How should you complete the Transact-SQL statement? To answer, drag the appropriate Transact-SQL segments to the correct locations. Each Transact-SQL segments may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.

Answer: B

Explanation:
A transaction is correctly defined for the INSERT INTO ..VALUES statement, and if there is an error in the transaction it will be caught and the 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.
A. Mastered
B. Not Mastered

Answer: A

Explanation:
Box 1: 0, 1, 2, 3, 4
Pivot example:
-- Pivot table with one row and five columns
SELECT 'AverageCost' AS Cost_Sorted_By_Production_Days, [0], [1], [2], [3], [4]  
FROM 
(SELECT DaysToManufacture, StandardCost FROM Production.Product) AS SourceTable PIVOT 
(  
AVG(StandardCost)  
FOR DaysToManufacture IN ([0], [1], [2], [3], [4])  
) AS PivotTable;  
Box 2: [CreditLimit]
Box 3: PIVOT
You can use the PIVOT and UNPIVOT relational operators to change a table-valued expression into another table. PIVOT rotates a table-valued expression by turning the unique values from one column in the expression into multiple columns in the output, and performs aggregations where they are required on any remaining column values that are wanted in the final output.
Box 4: 0, 1, 2, 3, 4
The IN clause determines whether a specified value matches any value in a subquery or a list. Syntax: test_expression [ NOT ] IN ( subquery | expression [ ,...n ] )
Where expression[ ,... n ] is a list of expressions to test for a match. All expressions must be of the same type as test_expression. References: https://technet.microsoft.com/en-us/library/ms177410(v=sql.105).aspx

NEW 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 are creating indexes in a data warehouse.
You have a dimension table named Table1 that has 10,000 rows. The rows are used to generate several reports. The reports join a column that is the primary key. The execution plan contains bookmark lookups for Table1. You discover that the reports run slower than expected.
You need to reduce the amount of time it takes to run the reports. Solution: You create a hash index on the primary key column. Does this meet the goal?
A. Yes
B. No

Answer: B

NEW QUESTION 7
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.
Start of repeated scenario
You have a database that contains the tables shown in the exhibit. (Click the Exhibit button.)
You review the Employee table and make the following observations:
- Every record has a value in the ManagerID except for the Chief Executive Officer (CEO).
- The FirstName and MiddleName columns contain null values for some records.
- The valid values for the Title column are Sales Representative manager, and CEO. You review the SalesSummary table and make the following observations:
  - The ProductCode column contains two parts: The first five digits represent a product code, and the last seven digits represent the unit price. The unit price uses the following pattern: ####.##.
  - You observe that for many records, the unit price portion of the ProductCode column contains values.
  - The RegionCode column contains NULL for some records.
  - Sales data is only recorded for sales representatives.

You are developing a series of reports and procedures to support the business. Details for each report or procedure follow.

Sales Summary report: This report aggregates data by year and quarter. The report must resemble the following table.

<table>
<thead>
<tr>
<th>SalesYear</th>
<th>SalesQuarter</th>
<th>YearSalesAmount</th>
<th>QuarterSalesAmount</th>
</tr>
</thead>
<tbody>
<tr>
<td>2015</td>
<td>1</td>
<td>2000.00</td>
<td>1000.00</td>
</tr>
<tr>
<td>2015</td>
<td>2</td>
<td>2000.00</td>
<td>500.00</td>
</tr>
<tr>
<td>2015</td>
<td>3</td>
<td>2000.00</td>
<td>250.00</td>
</tr>
<tr>
<td>2016</td>
<td>4</td>
<td>2000.00</td>
<td>250.00</td>
</tr>
<tr>
<td>2016</td>
<td>1</td>
<td>3500.00</td>
<td>500.00</td>
</tr>
<tr>
<td>2016</td>
<td>2</td>
<td>3500.00</td>
<td>1000.00</td>
</tr>
</tbody>
</table>

Sales Manager report: This report lists each sales manager and the total sales amount for all employees that report to the sales manager.

Sales by Region report: This report lists the total sales amount by employee by employee and by region. The report must include the following columns: EmployeeCode, MiddleName, LastName, RegionCode, and SalesAmount. If MiddleName is NULL, FirstName must be displayed. If both FirstName and MiddleName have null values, the word Unknown must be displayed. If RegionCode is NULL, the word Unknown must be displayed.

Report1: This report joins data from SalesSummary with the Employee table and other tables. You plan to create an object to support Report1. The object has the following requirements:
- be joinable with the SELECT statement that supplies data for the report
- can be used multiple times with the SELECT statement for the report
- be usable only with the SELECT statement for the report
- not be saved as a permanent object

Report2: This report joins data from SalesSummary with the Employee table and other tables. You plan to create an object to support Report1. The object has the following requirements:

Sales Hierarchy report. This report aggregates rows, creates subtotal rows, and super-aggregates rows over the SalesAmount column in a single result-set. The report uses SaleYear, SaleQuarter, and SaleMonth as a hierarchy. The result set must not contain a grand total or cross-tabulation aggregate rows.

Current Price Stored Procedure: This stored procedure must return the unit price for a product when a product code is supplied. The unit price must include a dollar sign at the beginning. In addition, the unit price must contain a comma every three digits to the left of the decimal point, and must display two digits to the left of the decimal point. The stored procedure must not throw errors, even if the product code contains invalid data.

End of Repeated Scenario

You need to create a query to return the data for the Sales Summary report.

Which three Transact-SQL segments should you use to develop the solution? To answer, move the appropriate Transact-SQL segments from the list of Transact-SQL segments to the answer area and arrange them in the correct order.

---

Guaranteed success with Our exam guides

visit - https://www.certshared.com
A. Mastered
B. Not Mastered

Answer: A

Explanation:
Use two CTE expressions, one for salesYear and one for SalesQuarter, and combine them with a SELECT statement.

Note: A common table expression (CTE) can be thought of as a temporary result set that is defined within the execution scope of a single SELECT, INSERT, UPDATE, DELETE, or CREATE VIEW statement. A CTE is similar to a derived table in that it is not stored as an object and lasts only for the duration of the query.

NEW 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 includes the tables shown in the exhibit (Click the Exhibit button.)

You need to create a Transact-SQL query that returns the following information:
* the customer number
* the customer contact name
* the date the order was placed, with a name of DateofOrder
* a column named Salesperson, formatted with the employee first name, a space, and the employee last name
* orders for customers where the employee identifier equals 4
The output must be sorted by order date, with the newest orders first. The solution must return only the most recent order for each customer. Solution: You run the following Transact-SQL statement:

```
SELECT c.custid, contactname, MAX(orderdate) AS DateofOrder, e.firstname + ' ' + e.lastname AS Salesperson
FROM Sales.Customers AS c
INNER JOIN Sales.Orders AS o ON c.custid = o.custid
INNER JOIN HR.Employee AS e ON o.empid = e.empid
GROUP BY c.custid, contactname, firstname, lastname, o.empid
HAVING o.empid = 4
ORDER BY DateofOrder DESC
```

Does the solution meet the goal?

A. Yes
B. No

Answer: B

Explanation:
We should use a WHERE clause, not a HAVING clause. The HAVING clause would refer to aggregate data.

NEW QUESTION 9
You have a database that contains the following tables: tblRoles, tblUsers, and tblUsersInRoles. The table tblRoles is defined as follows.

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Nullable</th>
<th>Primary key</th>
</tr>
</thead>
<tbody>
<tr>
<td>RoleID</td>
<td>int</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>RoleName</td>
<td>varchar(20)</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

You have a function named ufnGetRoleActiveUsers that was created by running the following Transact-SQL statement:
You need to list all roles and their corresponding active users. The query must return the RoleId, RoleName, and UserName columns. If a role has no active users, a NULL value should be returned as the UserName for that role. How should you complete the Transact-SQL statement? To answer, select the appropriate Transact-SQL segments in the answer area.

**Answer area**

```
SELECT *
FROM
    tblRoles
    CROSS JOIN
    tblUsersInRoles
    CROSS APPLY
        ufnGetRoleActiveUsers(RoleId)
    OUTER APPLY
        tblUsers
            WHERE tblUsersInRoles.RoleId = ufnGetRoleActiveUsers(RoleId) AND tblUsers.InActive = 1
```

A. Mastered  
B. Not Mastered  

**Answer:** A  

**Explanation:**

**Answer area**

```
SELECT *
FROM
    tblRoles
    CROSS JOIN
    tblUsersInRoles
    CROSS APPLY
        ufnGetRoleActiveUsers(RoleId)
    OUTER APPLY
        tblUsers
            WHERE tblUsersInRoles.RoleId = ufnGetRoleActiveUsers(RoleId) AND tblUsers.InActive = 1
```

**NEW QUESTION 10**

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 are building a stored procedure that will be used by hundreds of users concurrently.
You need to store rows that will be processed later by the stored procedure. The object that stores the rows must meet the following requirements:

- Be indexable
- Contain up-to-date statistics
- Be able to scale between 10 and 100,000 rows

The solution must prevent users from accessing one another’s data. Solution: You create a global temporary table in the stored procedure. Does this meet the goal?

A. Yes
B. No

**Answer:** A

**NEW QUESTION 11**

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 contains a single table named tblVehicleRegistration. The table is defined as follows:

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VehicleId</td>
<td>int</td>
<td>the primary key for the table</td>
</tr>
<tr>
<td>RegistrationNumber</td>
<td>varchar(5)</td>
<td>a vehicle registration number that contains only letters and numbers</td>
</tr>
<tr>
<td>RegistrationDate</td>
<td>date</td>
<td>the vehicle registration date</td>
</tr>
<tr>
<td>UserId</td>
<td>int</td>
<td>an identifier for the vehicle owner</td>
</tr>
</tbody>
</table>

You run the following query:

```sql
SELECT UserId FROM tblVehicleRegistration
WHERE RegistrationNumber = 'AB012'
AND RegistrationDate > '2016-01-01'
```

The query output window displays the following error message: “Conversion failed when converting the varchar value ‘AB012’ to data type int.”

You need to resolve the error.

Solution: You modify the Transact-SQL statement as follows:

```sql
SELECT UserId FROM tblVehicleRegistration
WHERE CAST(RegistrationNumber AS int) = 'AB012'
AND RegistrationDate > '2016-01-01'
```

Does the solution meet the goal?

A. Yes
B. No

**Answer:** A

**NEW QUESTION 12**

A company’s sales team is divided in two different regions, North and South. You create tables named SalesNorth and SalesSouth. The SalesNorth table stores sales data from the North region. The SalesSouth table stores sales data from the South region. Both tables use the following structure:

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Allow nulls</th>
</tr>
</thead>
<tbody>
<tr>
<td>region</td>
<td>CHAR(1)</td>
<td>Yes</td>
</tr>
<tr>
<td>salesID</td>
<td>INT</td>
<td>Yes</td>
</tr>
<tr>
<td>customer</td>
<td>VARCHAR(150)</td>
<td>Yes</td>
</tr>
<tr>
<td>amount</td>
<td>MONEY</td>
<td>Yes</td>
</tr>
</tbody>
</table>

You need to create a consolidated result set that includes all records from both tables. Which Transact-SQL statement should you run?

A. SELECT SalesNorth.salesID, SalesNorth.customer, SalesNorth.amount, SalesSouth.salesID, SalesSouth.customer, SalesSouth.amountFROM SalesNorthJOIN SalesSouth ON SalesNorth.salesID = SalesSouth.salesID
B. SELECT SalesNorth.salesID, SalesNorth.customer, SalesNorth.amount, SalesSouth.salesID, SalesSouth.customer, SalesSouth.amountFROM SalesNorthLEFT JOIN SalesSouth ON SalesNorth.salesID = SalesSouth.salesID
C. SELECT salesID, customer, amount FROM SalesNorthUNION ALL SELECT salesID, customer, amount FROM SalesSouth
D. SELECT salesID, customer, amount FROM SalesNorthUNIONSELECT salesID, customer, amountFROM SalesSouth

**Answer:** C

**Explanation:**

NEW QUESTION 13
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 create a table by running the following Transact-SQL statement:

```
CREATE TABLE Customers (
    CustomerID int NOT NULL PRIMARY KEY CLUSTERED,
    FirstName nvarchar(100) NOT NULL,
    LastName nvarchar(100) NOT NULL,
    TaxIdNumber varchar(20) NOT NULL,
    Address nvarchar(1024) NOT NULL,
    AnnualRevenue decimal(19,2) NOT NULL,
    DateCreated datetime2(2) NOT NULL,
    ValidFrom datetime2(2) GENERATED ALWAYS AS ROW START NOT NULL,
    ValidTo datetime2(2) GENERATED ALWAYS AS ROW END NOT NULL,
    PERIOD FOR SYSTEM_TIME(ValidFrom, ValidTo)
)
WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE = CustomersHistory))
```

You need to audit all customer data.
Which Transact-SQL statement should you run?

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E
F. Option F
G. Option G
H. Option H

Answer: B

Explanation:
The FOR SYSTEM_TIME ALL clause returns all the row versions from both the Temporal and History table. Note: A system-versioned temporal table defined through is a new type of user table in SQL Server 2016, here defined on the last line WITH (SYSTEM_VERSIONING = ON…, is designed to keep a full history of data changes and allow easy point in time analysis.

To query temporal data, the SELECT statement FROM<table> clause has a new clause FOR SYSTEM_TIME with five temporal-specific sub-clauses to query data across the current and history tables.


NEW QUESTION 14

You need to recommend a query that returns the number of customers who never completed a transaction. Which query should you recommend?

A

```
SELECT COUNT(Cust.CustomerID)
FROM Sales.Customers Cust
  LEFT JOIN Sales.CustomerTransactions Trans
    ON Cust.CustomerID = Trans.CustomerID
WHERE Trans.CustomerTransactionID IS NULL;
```

B

```
SELECT COUNT(CustomerID)
FROM Sales.Customers Cust
  LEFT JOIN Sales.CustomerTransactions Trans
    ON Cust.CustomerID = Trans.CustomerID
WHERE Trans.CustomerTransactionID IS NULL;
```

C

```
SELECT COUNT(CustomerID)
FROM Sales.Customers Cust
  LEFT JOIN Sales.CustomerTransactions Trans
    ON Cust.CustomerID = Trans.CustomerID
```

D

```
SELECT COUNT(CustomerID)
FROM Sales.Customers Cust
  INNER JOIN Sales.CustomerTransactions Trans
    ON Cust.CustomerID = Trans.CustomerID
WHERE Trans.CustomerTransactionID IS NULL;
```
NEW QUESTION 15
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) NOT 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:

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

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

Answer: B

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

NEW QUESTION 16
You have a table named HumanResources.Employee. You configure the table to use a default history table that contains 10 years of data.

You need to write a query that retrieves the values of the BusinessEntityID and JobTitle fields. You must retrieve all historical data up to January 1, 2017 where the value of the BusinessEntityID column equals 4.

Which four Transact-SQL segments should you use to develop the solution? To answer, move the appropriate Transact-SQL segments to the answer area and arrange them in the correct order.
A. Mastered
B. Not Mastered

**Answer:** A

**Explanation:**

**References:**

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

<table>
<thead>
<tr>
<th>Column</th>
<th>Data type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CityID</td>
<td>int</td>
<td>primary key</td>
</tr>
<tr>
<td>LatestRecordedPopulation</td>
<td>bigint</td>
<td>null values are permitted</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Column</th>
<th>Data type</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CustomerCategoryId</td>
<td>int</td>
<td>primary key</td>
</tr>
<tr>
<td>CustomerCategoryName</td>
<td>nvarchar(50)</td>
<td>does not allow null values</td>
</tr>
</tbody>
</table>

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 for the delivery city that is associated with the customer.
- For customers that are on a credit hold, return the CustomerID and the latest recorded population for the 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.Cities
WHERE (IsOnCreditHold = 0 AND DeliveryCityID = CityID)
OR (IsOnCreditHold = 1 AND PostalCityID = CityID)
```

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

C
```
SELECT CustomerID, ISNULL(A.LatestRecordedPopulation, B.LatestRecordedPopulation)
FROM Sales.Customers
INNER JOIN Application.Cities AS A ON A.CityID = DeliveryCityID
INNER JOIN Application.Cities 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.Cities AS A ON A.CityID = CityId
```
A. Option A  
B. Option B  
C. Option C  
D. Option D

Answer: AB

Explanation:  
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:  

NEW QUESTION 18  
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.

You have a database that contains a single table named tblVehicleRegistration. The table is defined as follows:

<table>
<thead>
<tr>
<th>Column name</th>
<th>Data type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>VehicleId</td>
<td>int</td>
<td>the primary key for the table</td>
</tr>
<tr>
<td>RegistrationNumber</td>
<td>varchar(5)</td>
<td>a vehicle registration number that contains only letters and numbers</td>
</tr>
<tr>
<td>RegistrationDate</td>
<td>date</td>
<td>the vehicle registration date</td>
</tr>
<tr>
<td>UserId</td>
<td>int</td>
<td>an identifier for the vehicle owner</td>
</tr>
</tbody>
</table>

You run the following query:

```
SELECT RegistrationNumber, RegistrationDate FROM tblVehicleRegistration
WHERE RegistrationNumber = '20012' AND RegistrationDate > '2016-01-01'
```

The query output window displays the following error message: “Conversion failed when converting the varchar value ‘AB012’ to data type int.” You need to resolve the error.

Solution: You modify the Transact-SQL statement as follows:

```
SELECT RegistrationNumber, RegistrationDate FROM tblVehicleRegistration
WHERE RegistrationNumber = '20012'
AND RegistrationDate > '2016-01-01'
```

Does the solution meet the goal?

A. Yes  
B. No

Answer: B

NEW QUESTION 19  
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 create a table by running the following Transact-SQL statement:

```
CREATE TABLE Customers (  
    CustomerID int NOT NULL PRIMARY KEY CLUSTERED,  
    FirstName nvarchar(100) NOT NULL,  
    LastName nvarchar(100) NOT NULL,  
    TaxIdNumber varchar(20) NOT NULL,  
    Address nvarchar(1024) NOT NULL,  
    AnnualRevenue decimal(19,2) NOT NULL,  
    DateCreated datetime2(2) NOT NULL,  
    ValidFrom datetime2(2) GENERATED ALWAYS AS ROW START NOT NULL,  
    ValidTo datetime2(2) GENERATED ALWAYS AS ROW END NOT NULL,  
    PERIOD FOR SYSTEM_TIME (ValidFrom, ValidTo)  
)  
WITH (SYSTEM_VERSIONING = ON (HISTORY_TABLE = CustomersHistory))
```

You need to return normalized data for all customers that were added in the year 2014. Which Transact-SQL statement should you run?
A

```sql
SELECT FirstName, LastName, SUM(AnnualRevenue)
FROM Customers
GROUP BY GROUPING SETS((FirstName, LastName, AnnualRevenue),())
ORDER BY FirstName, LastName, AnnualRevenue
```

B

```sql
SELECT FirstName, LastName, Address
FROM Customers
FOR SYSTEM_TIME ALL ORDER BY ValidFrom
```

C

```sql
SELECT c.CustomerID, c.FirstName, c.LastName, c.Address, c.ValidFrom, c.ValidTo
FROM Customers AS c
ORDER BY c.CustomerID
FOR JSON AUTO, ROOT('Customers')
```

D

```sql
SELECT * FROM (SELECT CustomerID, FirstName, LastName, Address, AnnualRevenue, DateCreated
FROM Customers) AS Customers PIVOT(AVG(AnnualRevenue)
FOR DateCreated IN([2014]')) AS PivotCustomers
ORDER BY LastName, FirstName
```

E

```sql
SELECT CustomerID, AVG(AnnualRevenue)
AS AverageAnnualRevenue, FirstName, LastName, Address, DateCreated
FROM Customers WHERE YEAR(DateCreated) >= 2014
GROUP BY CustomerID, FirstName, LastName, Address, DateCreated
```

F

```sql
SELECT c.CustomerID, c.FirstName, c.LastName, c.Address, c.ValidFrom, c.ValidTo
FROM Customers AS c ORDER BY c.CustomerID
FOR XML PATH ('CustomerData'), root ('Customers')
```

G

```sql
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, ValidFrom, ValidTo
FROM Customers FOR SYSTEM_TIME
BETWEEN '2014-01-01 00:00:00.000000' AND '2015-01-01 00:00:00.000000'
```

H

```sql
SELECT CustomerID, FirstName, LastName, TaxIdNumber, Address, ValidFrom, ValidTo
FROM Customers
WHERE DateCreated
BETWEEN '20140101' AND '20141231'
```
NEW QUESTION 20
You deploy a hybrid Active Directory Domain Services (AD DS) and Azure Active Directory (Azure AD) environment. You need to implement a synchronization process that meets the following requirements:
• Monitor latency trends for synchronization processes.
• Configure automated email notifications when data synchronization to Azure AD is unsuccessful.
• Monitor operational metrics, performance metrics, and status of on-premises domain controllers.
• Monitor replication status of on-premises domain controllers.
• Monitor connections and receive automated alerts when abnormalities occur.
Which technologies should you use? To answer, drag the appropriate technologies to the correct tasks. Each technology may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.

A. Mastered
B. Not Mastered

Answer: A

Explanation:

NEW QUESTION 21
......
Thank You for Trying Our Product

We offer two products:

1st - We have Practice Tests Software with Actual Exam Questions

2nd - Questions and Answers in PDF Format

70-761 Practice Exam Features:

* 70-761 Questions and Answers Updated Frequently
* 70-761 Practice Questions Verified by Expert Senior Certified Staff
* 70-761 Most Realistic Questions that Guarantee you a Pass on Your First Try
* 70-761 Practice Test Questions in Multiple Choice Formats and Updates for 1 Year

100% Actual & Verified — Instant Download, Please Click
Order The 70-761 Practice Test Here