➢ Vendor: Microsoft

➢ Exam Code: 70-761

➢ Exam Name: Querying Data with Transact-SQL

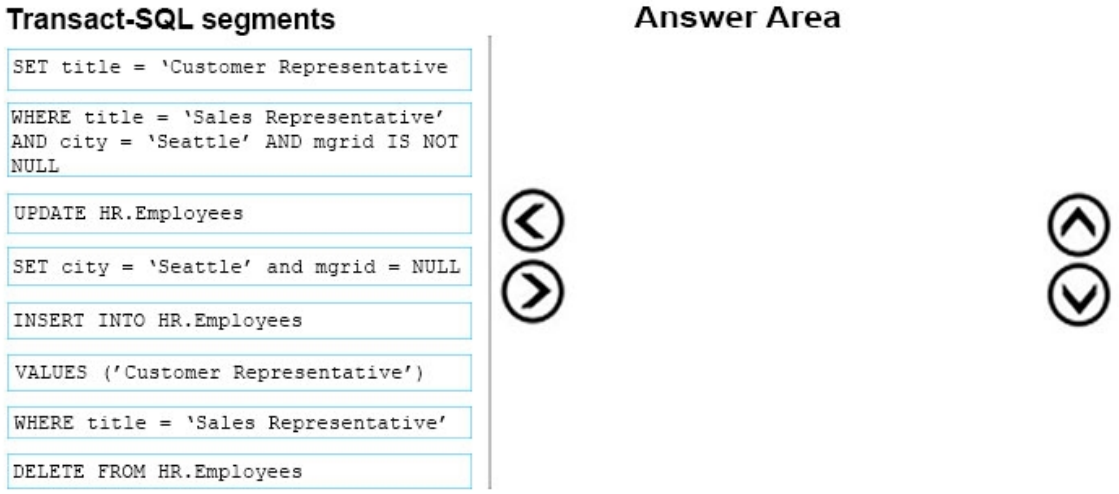
➢ Question 21 – Question 30

QUESTION 21

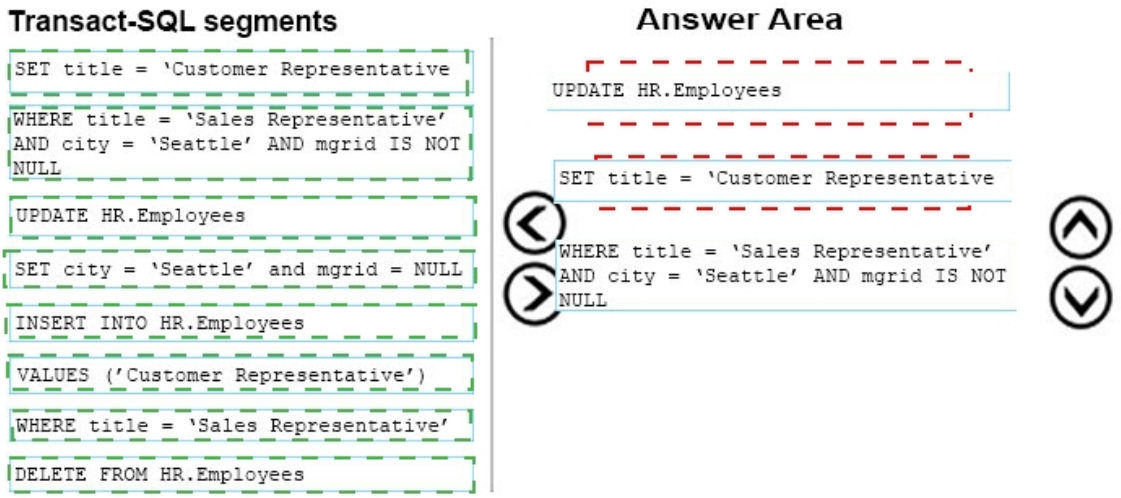
Drag and Drop Question

You have a table named HR.Employees as shown in the exhibit. (Click the exhibit button.)

You need to write a query that will change the value of the job title column to Customer Representative for any employee who lives in Seattle and has a job title of Sales Representative. If the employee does not have a manager defined, you must not change the title. 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.



Answer:



Explanation:
https://msdn.microsoft.com/en-us/library/ms177523.aspx

QUESTION 22
Hotspot Question
You have the following Transact-SQL query:

```
SELECT
     City.CityID,
     City.CityName,
     TranslateName(Nearby.CityName) AS NearbyCity
FROM Cities AS City
CROSS APPLY NearbyCities(City.CityID) AS Nearby
```

What type of functions are used in the query? To answer, select the appropriate options in the answer area. NOTE: Each correct selection is worth one point.

Answer Area

Function	Type
TranslateName	Scalar / Table-Valued / System / Aggregate
NearbyCities	Scalar / Table-Valued / System / Aggregate

Answer:

Answer Area

Function	Type
TranslateName	Scalar / Table-Valued / System / Aggregate
NearbyCities	Scalar / Table-Valued / System / Aggregate

Explanation:
Box 1: Scalar
The return value of a function can either be a scalar (single) value or a table.
Box 2: Table-Valued
The APPLY operator allows you to invoke a table-valued function for each row returned by an outer table expression of a query. The table-valued function acts as the right input and the outer table expression acts as the left input. The right input is evaluated for each row from the left input and the rows produced are combined for the final output. The list of columns produced by the APPLY

operator is the set of columns in the left input followed by the list of columns returned by the right input.

References:

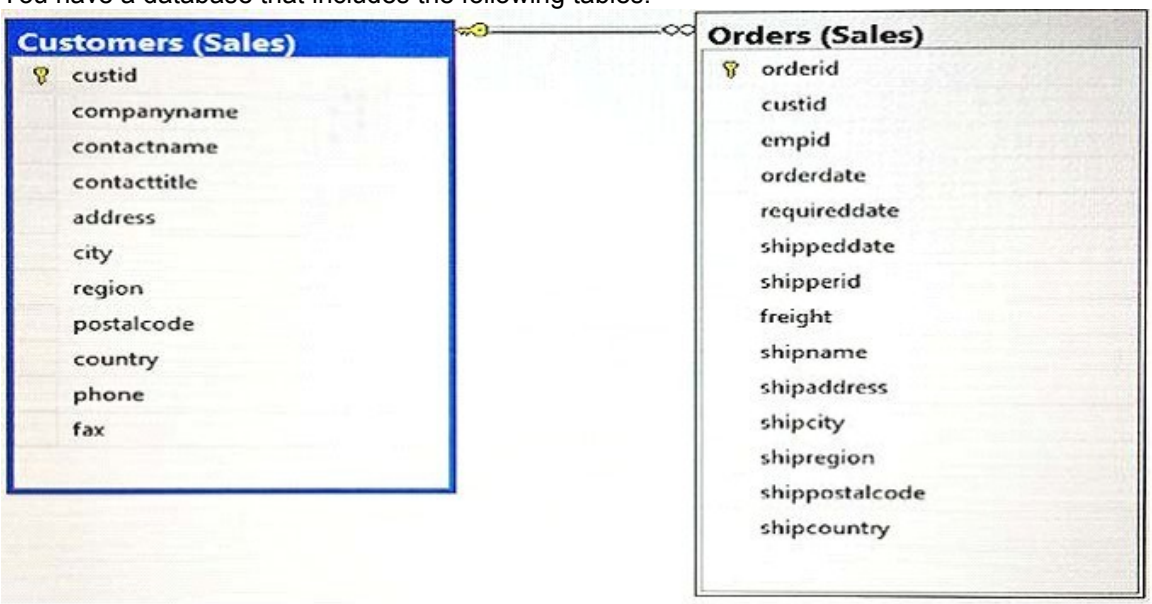
https://msdn.microsoft.com/en-us/library/ms186755.aspx

https://technet.microsoft.com/en-us/library/ms175156(v=sql.105).aspx

QUESTION 23

Drag and Drop Question

You have a database that includes the following tables:



You need to create a list of all customer IDs and the date of the last order that each customer placed. If the customer has not placed any orders, you must return the date January 1, 1900. The column names must be CustomerID and LastOrderDate. Which four 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.

Transact-SQL segments

```
GROUP BY c.custid
```

```
FROM sales.Customers AS c INNER
JOIN sales.Orders AS o
```

```
ON c.orderid = o.orderid
```

```
SELECT c.custid AS CustomerID,
MAX(o.orderdate) AS LastOrderDate
```

```
FROM sales.Customers AS c LEFT
OUTER JOIN sales.Orders AS o
```

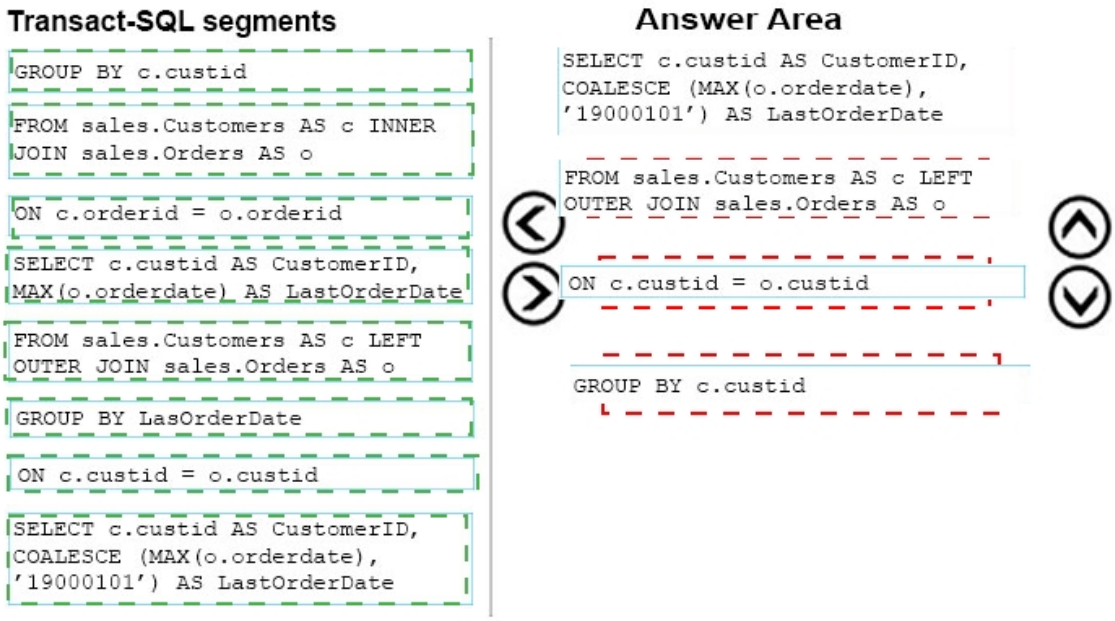
```
GROUP BY LasOrderDate
```

```
ON c.custid = o.custid
```

```
SELECT c.custid AS CustomerID,
COALESCE (MAX(o.orderdate),
'19000101') AS LastOrderDate
```

Answer Area

Answer:



Explanation:

Box 1: SELECT…COALESCE...

The COALESCE function evaluates the arguments in order and returns the current value of the first expression that initially does not evaluate to NULL.

Box 2: LEFT OUTER JOIN…

The LEFT JOIN (LEFT OUTER JOIN) keyword returns all rows from the left table (table1), with the matching rows in the right table (table2). The result is NULL in the right side when there is no match. A customer might have no orders so the right table must be allowed have a NULL value.

Box 3: ON c.custid = o.custid

We JOIN on the custID column, which is available in both tables.

Box 4: GROUP BY c.custid

References:

https://technet.microsoft.com/en-us/library/ms189499(v=sql.110).aspx

http://www.w3schools.com/sql/sql_join_left.asp

QUESTION 24

Hotspot Question

You run the following Transact-SQL statement:

```
CREATE TABLE Sales.Customers(
     custid int IDENTITY(1,1) NOT NULL,
     companyname nvarchar(50) NULL,
     contacttitle nvarchar(30) NOT NULL,
     address nvarchar(60) NOT NULL,
     postalcode nvarchar(10) NOT NULL,
     region nvarchar(15) NULL,
     phone nvarchar(24) NOT NULL,
     fax nvarchar(24) NULL,
) ON PPRIMARY
```

You need to ensure that you can insert data into the table. What are the characteristics of the data? To answer, select the appropriate options in the answer area.

Answer Area

Column input constraint	Column name
Values cannot be entered into this column	custid fax postalcode region
A value must be inserted into this column	custid fax postalcode region
Data entry into this column is optional	custid fax postalcode region

Answer:

Answer Area

Column input constraint	Column name
Values cannot be entered into this column	custid / fax / postalcode / region
A value must be inserted into this column	custid / fax / postalcode / region
Data entry into this column is optional	custid / fax / postalcode / region

Explanation:
Box 1: custid
IDENTITY indicates that the new column is an identity column. When a new row is added to the table, the Database Engine provides a unique, incremental value for the column. Identity columns are typically used with PRIMARY KEY constraints to serve as the unique row identifier for the table.
Box 2: postalcode
postalcode is declared as NOT NULL, which means that a value must be inserted.
Box 3: region
Fax is also a correct answer. Both these two columns are declared as NULL, which means that data entry is optional.
References: https://msdn.microsoft.com/en-us/library/ms174979.aspx

QUESTION 25
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
ALL
ALTER
AND
ANY
AS
ASC
AUTHORIZATION
BACKUP
BEGIN
BETWEEN
BREAK
BROWSE
BULK
BY
CASCADE
CASE
CHECK
CHECKPOINT
CLOSE
CLUSTERED
COALESCE
COLLATE
COLUMN
COMMIT
COMPUTE
CONCAT
CONSTRAINT
CONTAINS
CONTAINSTABLE
CONTINUE
CONVERT
CREATE
CROSS
CURRENT
CURRENT_DATE
CURRENT_TIME
CURRENT_TIMESTAMP
CURENT_USER
CURSOR
DATABASE
DBCC
DEALLOCATE
DECLARE
DEFAULT
DELETE
DENY
DESC
DISK
DISTINCT
DISTRIBUTED
DOUBLE
DROP
DUMP
ELSE
END
ERRLVL
ESCAPE
ESCEPT
EXEC
EXECUTE
EXISTS
EXIT
EXTERNAL
FETCH
FILE
FILLFACTOR
FORFOREIGN
FREETEXT
FREETEXTTABLE
FROM
FULL
FUNCTION
GOTO
GRANT
GROUP
HAVING
HOLDLOCK
IDENTITY
IDENTITY_INSERT
IDENTITYCOL
IF
IN
INDEX
INNER
INSERT
INTERSECT
INTO
IS
JOIN
KEY
KILL
LEFT
LIKE
LINENO
LOAD
MERGE
NATIONAL
NOCHECK
NONCLUSTERED
NOT
NULL
NULLIF
OF
OFF
OFFSETS
ON
OPEN
OPENDATASOURCE
OPENQUERY
OPENROWSET
OPENXML
OPTION
OR
ORDER
OUTER
OVER
PERCENT
PIVOT
PLAN
PRECISION
PRIMARY
PRINT
PROC
PROCEDURE
PUBLIC
RAISERROR
READ
READTEXT
RECONFIGURE
REFERENCES
REPLICATION
RESTORE
RESTRICT
RETURN
REVERT
REVOKE
RIGHT
ROLLBACK
ROWCOUNT
ROWGUIDCOL
RULE
SAVE
SCHEMA
SECURITYAUDIT
SELECT
SEMANTICKEYPHRASETABLE
SEMANTICSIMILARITYDETAILSTABLE
SEMANTICSIMILARITYTABLE
SESSION_USER
SET
SETUSER
SHUTDOWN
SOME
STATISTICS
SYSTEM_USER
TABLE
TABLESAMPLE
TEXTSIZE
THEN
TO
TOP
TRAN
TRANSACTION
TRIGGER
TRUNCATE
TRY_CONVERT
TSEQUAL
UNION
UNIQUE
UNPIVOT
UPDATE
UPDATETEXT
USE
USER
VALUES
VARYING
VIEW
WAITFOR
WHEN
WHERE
WHILE
WITH
WITHIN GROUP
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.

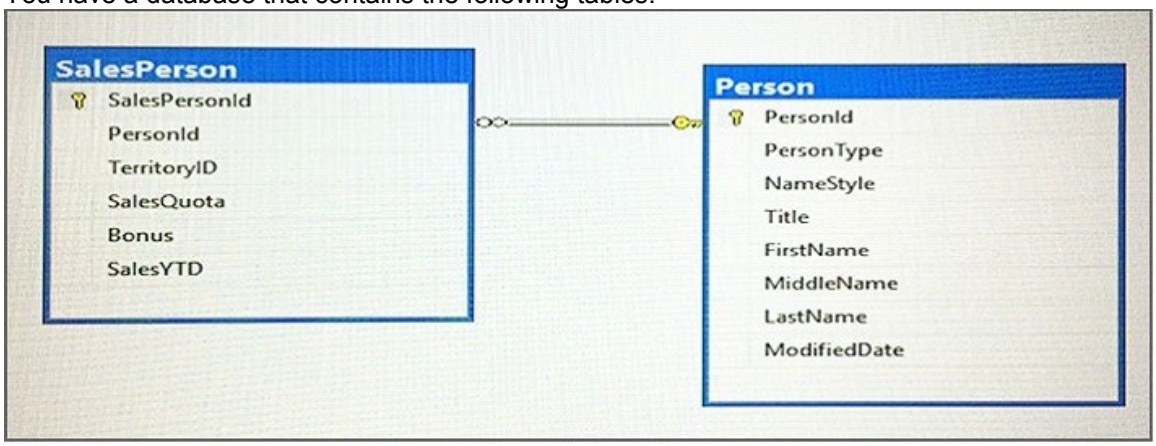
```
1 DELETE
```

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

Answer: Pending

QUESTION 26

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
ALL
ALTER
AND
ANY
AS
ASC
AUTHORIZATION
BACKUP
BEGIN
BETWEEN
BREAK
BROWSE
BULK
BY
CASCADE
CASE
CHECK
CHECKPOINT
CLOSE
CLUSTERED
COALESCE
COLLATE
COLUMN
COMMIT
COMPUTE
CONCAT
CONSTRAINT
CONTAINS
CONTAINSTABLE
CONTINUE
CONVERT
CREATE
CROSS
CURRENT
CURRENT_DATE
CURRENT_TIME
CURRENT_TIMESTAMP
CURENT_USER
CURSOR
DATABASE
DBCC
DEALLOCATE
DECLARE
DEFAULT
DELETE
DENY
DESC
DISK
DISTINCT
DISTRIBUTED
DOUBLE
DROP
DUMP
ELSE
END
ERRLVL
ESCAPE
ESCEPT
EXEC
EXECUTE
EXISTS
EXIT
EXTERNAL
FETCH
FILE
FILLFACTOR
FORFOREIGN
FREETEXT
FREETEXTTABLE
FROM
FULL
FUNCTION
GOTO
GRANT
GROUP
HAVING
HOLDLOCK
IDENTITY
IDENTITY_INSERT
IDENTITYCOL
IF
IN
INDEX
INNER
INSERT
INTERSECT
INTO
IS
JOIN
KEY
KILL
LEFT
LIKE
LINENO
LOAD
MERGE
NATIONAL
NOCHECK
NONCLUSTERED
NOT
NULL
NULLIF
OF
OFF
OFFSETS
ON
OPEN
OPENDATASOURCE
OPENQUERY
OPENROWSET
OPENXML
OPTION
OR
ORDER
OUTER
OVER
PERCENT
PIVOT
PLAN
PRECISION
PRIMARY
PRINT
PROC
PROCEDURE
PUBLIC
RAISERROR
READ
READTEXT
RECONFIGURE
REFERENCES
REPLICATION
RESTORE
RESTRICT
RETURN
REVERT
REVOKE
RIGHT
ROLLBACK
ROWCOUNT
ROWGUIDCOL
RULE
SAVE
SCHEMA
SECURITYAUDIT
SELECT
SEMANTICKEYPHRASETABLE
SEMANTICSIMILARITYDETAILSTABLE
SEMANTICSIMILARITYTABLE
SESSION_USER
SET
SETUSER
SHUTDOWN
SOME
STATISTICS
SYSTEM_USER
TABLE
TABLESAMPLE
TEXTSIZE
THEN
TO
TOP
TRAN
TRANSACTION
TRIGGER
TRUNCATE
TRY_CONVERT
TSEQUAL
UNION
UNIQUE
UNPIVOT
UPDATE
UPDATETEXT
USE
USER
VALUES
VARYING
VIEW
WAITFOR
WHEN
WHERE
WHILE
WITH
WITHIN GROUP
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.

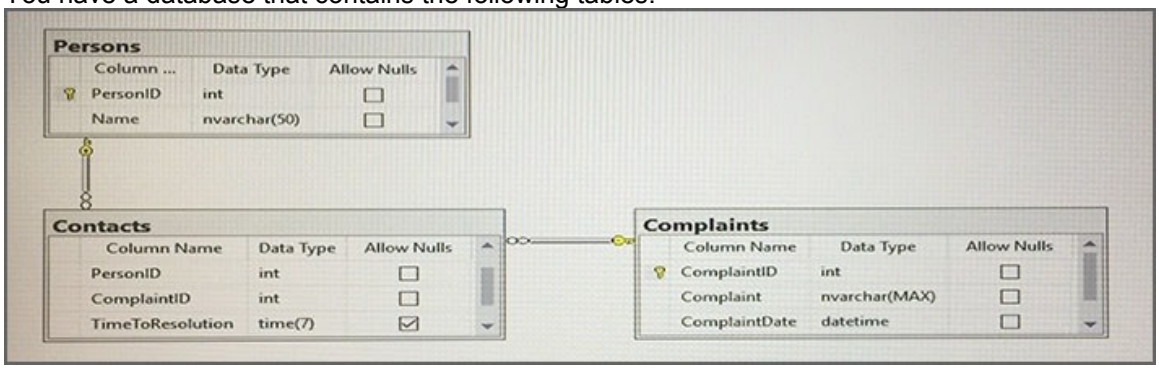
```
1 SELECT
2 FROM Person AS P INNER JOIN SalesPerson AS S
3 ON P.PersonID = S.SalesPersonID
4 WHERE
```

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

Answer: Pending

QUESTION 27

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.

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.

Keywords

ADD
ALL
ALTER
AND
ANY
AS
ASC
AUTHORIZATION
BACKUP
BEGIN
BETWEEN
BREAK
BROWSE
BULK
BY
CASCADE
CASE
CHECK
CHECKPOINT
CLOSE
CLUSTERED
COALESCE
COLLATE
COLUMN
COMMIT
COMPUTE
CONCAT
CONSTRAINT
CONTAINS
CONTAINSTABLE
CONTINUE
CONVERT
CREATE
CROSS
CURRENT
CURRENT_DATE
CURRENT_TIME
CURRENT_TIMESTAMP
CURENT_USER
CURSOR
DATABASE
DBCC
DEALLOCATE
DECLARE
DEFAULT
DELETE
DENY
DESC
DISK
DISTINCT
DISTRIBUTED
DOUBLE
DROP
DUMP
ELSE
END
ERRLVL
ESCAPE
ESCEPT
EXEC
EXECUTE
EXISTS
EXIT
EXTERNAL
FETCH
FILE
FILLFACTOR
FORFOREIGN
FREETEXT
FREETEXTTABLE
FROM
FULL
FUNCTION
GOTO
GRANT
GROUP
HAVING
HOLDLOCK
IDENTITY
IDENTITY_INSERT
IDENTITYCOL
IF
IN
INDEX
INNER
INSERT
INTERSECT
INTO
IS
JOIN
KEY
KILL
LEFT
LIKE
LINENO
LOAD
MERGE
NATIONAL
NOCHECK
NONCLUSTERED
NOT
NULL
NULLIF
OF
OFF
OFFSETS
ON
OPEN
OPENDATASOURCE
OPENQUERY
OPENROWSET
OPENXML
OPTION
OR
ORDER
OUTER
OVER
PERCENT
PIVOT
PLAN
PRECISION
PRIMARY
PRINT
PROC
PROCEDURE
PUBLIC
RAISERROR
READ
READTEXT
RECONFIGURE
REFERENCES
REPLICATION
RESTORE
RESTRICT
RETURN
REVERT
REVOKE
RIGHT
ROLLBACK
ROWCOUNT
ROWGUIDCOL
RULE
SAVE
SCHEMA
SECURITYAUDIT
SELECT
SEMANTICKEYPHRASETABLE
SEMANTICSIMILARITYDETAILSTABLE
SEMANTICSIMILARITYTABLE
SESSION_USER
SET
SETUSER
SHUTDOWN
SOME
STATISTICS
SYSTEM_USER
TABLE
TABLESAMPLE
TEXTSIZE
THEN
TO
TOP
TRAN
TRANSACTION
TRIGGER
TRUNCATE
TRY_CONVERT
TSEQUAL
UNION
UNIQUE
UNPIVOT
UPDATE
UPDATETEXT
USE
USER
VALUES
VARYING
VIEW
WAITFOR
WHEN
WHERE
WHILE
WITH
WITHIN GROUP
WRITETEXT

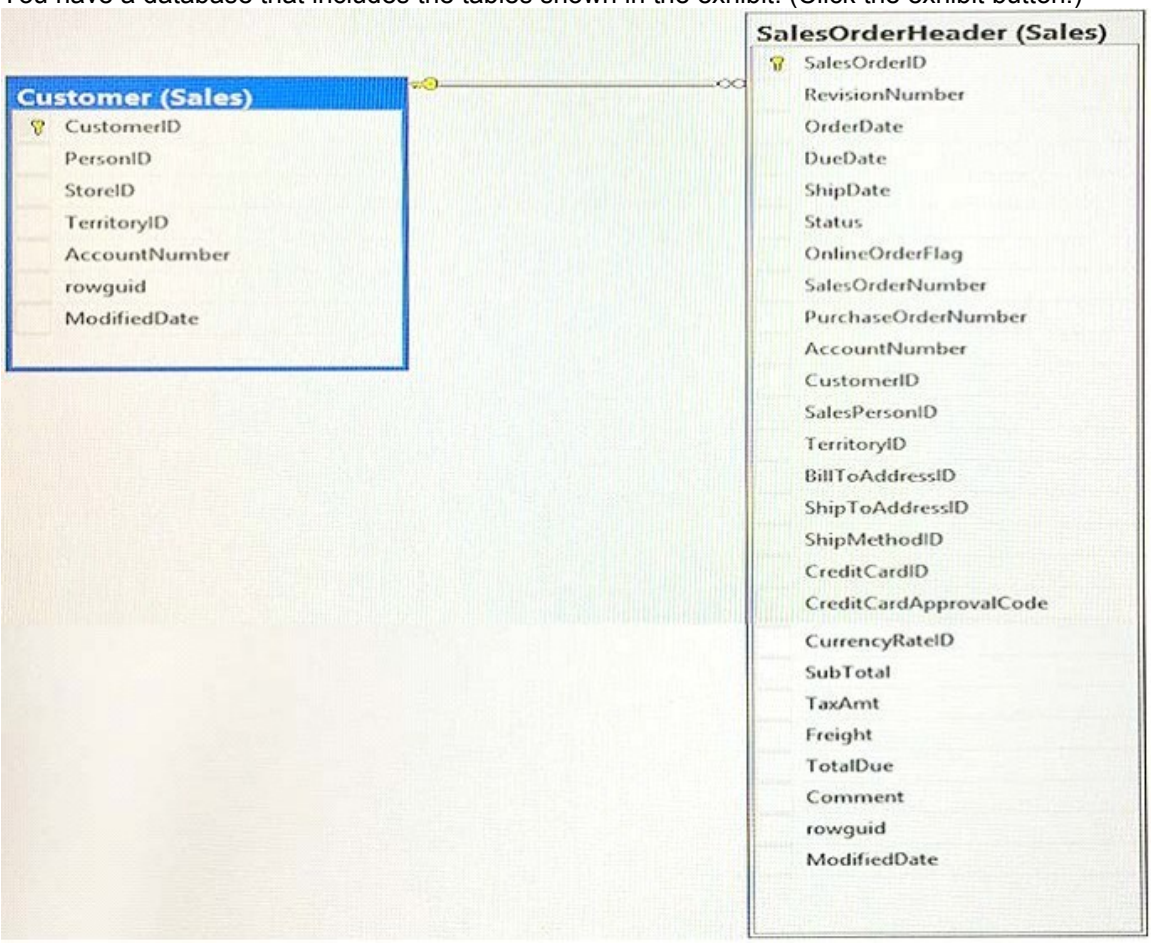
```
1 SELECT Complaints.ComplaintId,
2 FROM
3 JOIN
4 JOIN
```

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

Answer: Pending

QUESTION 28

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 O 1/01/1990 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
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: A
Explanation:
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 29
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	

Customer Audit:

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
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: D
Explanation:
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.

QUESTION 30

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.

Drag and Drop Question

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

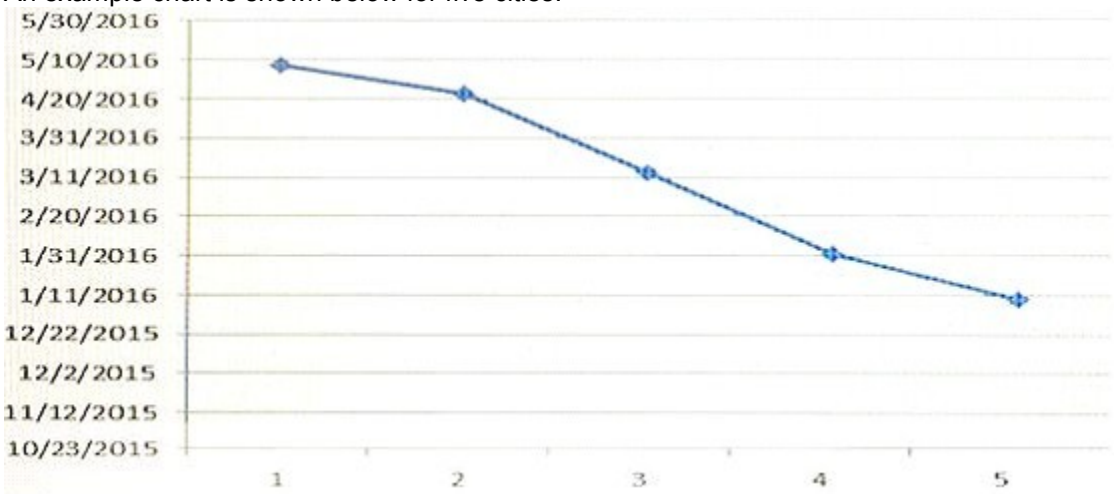
Details for the Sales.CustomerCategories table are shown in the following table:

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

You are creating a report to show when the first customer account was opened in each city. The report contains a line chart with the following characteristics:

- The chart contains a data point for each city, with lines connecting the points.
- The X axis contains the position that the city occupies relative to other cities.
- The Y axis contains the date that the first account in any city was opened.

An example chart is shown below for five cities:



During a sales promotion, customers from various cities open new accounts on the same date. You need to write a query that returns the data for the chart. How should you complete the Transact-SQL statement? To answer, drag the appropriate Transact-SQL segments to the correct locations. Each Transact-SQL segment 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. NOTE: Each correct selection is worth one point.

Transact-SQL segments

- `DENSE_RANK() OVER`
- `RANK() OVER`
- `(ORDER BY MIN(AccountOpenedDate) DESC)`
- `(PARTITION BY CityID ORDER BY min(AccountOpenedDate) DESC)`
- `(ORDER BY AccountOpenedDate DESC)`
- `(PARTITION BY CityID ORDER BY AccountOpenedDate DESC)`
- `GROUP BY CityID`
- `GROUP BY PARTITION`

Answer Area

```
SELECT
    CityID,
    MIN(AccountOpenedDate),
```

[Transact-SQL segment]

[Transact-SQL segment]

```
FROM Application.Citites
INNER JOIN Sales.Customers ON CityID = PostalCityID
```

[Transact-SQL segment]

```
ORDER BY MIN(AccountOpenedDate) DESC
```

Answer:

Transact-SQL segments

```
DENSE_RANK() OVER
RANK() OVER
(ORDER BY MIN(AccountOpenedDate) DESC)
(PARTITION BY CityID ORDER BY min(AccountOpenedDate) DESC)
(ORDER BY AccountOpenedDate DESC)
(PARTITION BY CityID ORDER BY AccountOpenedDate DESC)
GROUP BY CityID
GROUP BY PARTITION
```

Answer Area

```
SELECT
    CityID,
    MIN(AccountOpenedDate),
    RANK() OVER
    (PARTITION BY CityID ORDER BY min(AccountOpenedDate) DESC)
FROM Application.Citites
INNER JOIN Sales.Customers ON CityID = PostalCityID
GROUP BY CityID
ORDER BY MIN(AccountOpenedDate) DESC
```

Explanation:

Box 1: RANK() OVER

RANK returns the rank of each row within the partition of a result set. The rank of a row is one plus the number of ranks that come before the row in question. ROW_NUMBER and RANK are similar. ROW_NUMBER numbers all rows sequentially (for example 1, 2, 3, 4, 5).