DUMPSARENA

Implementing a SQL Data Warehouse

Microsoft 70-767

Version Demo

Total Demo Questions: 10

Total Premium Questions: 147

Buy Premium PDF

https://dumpsarena.co

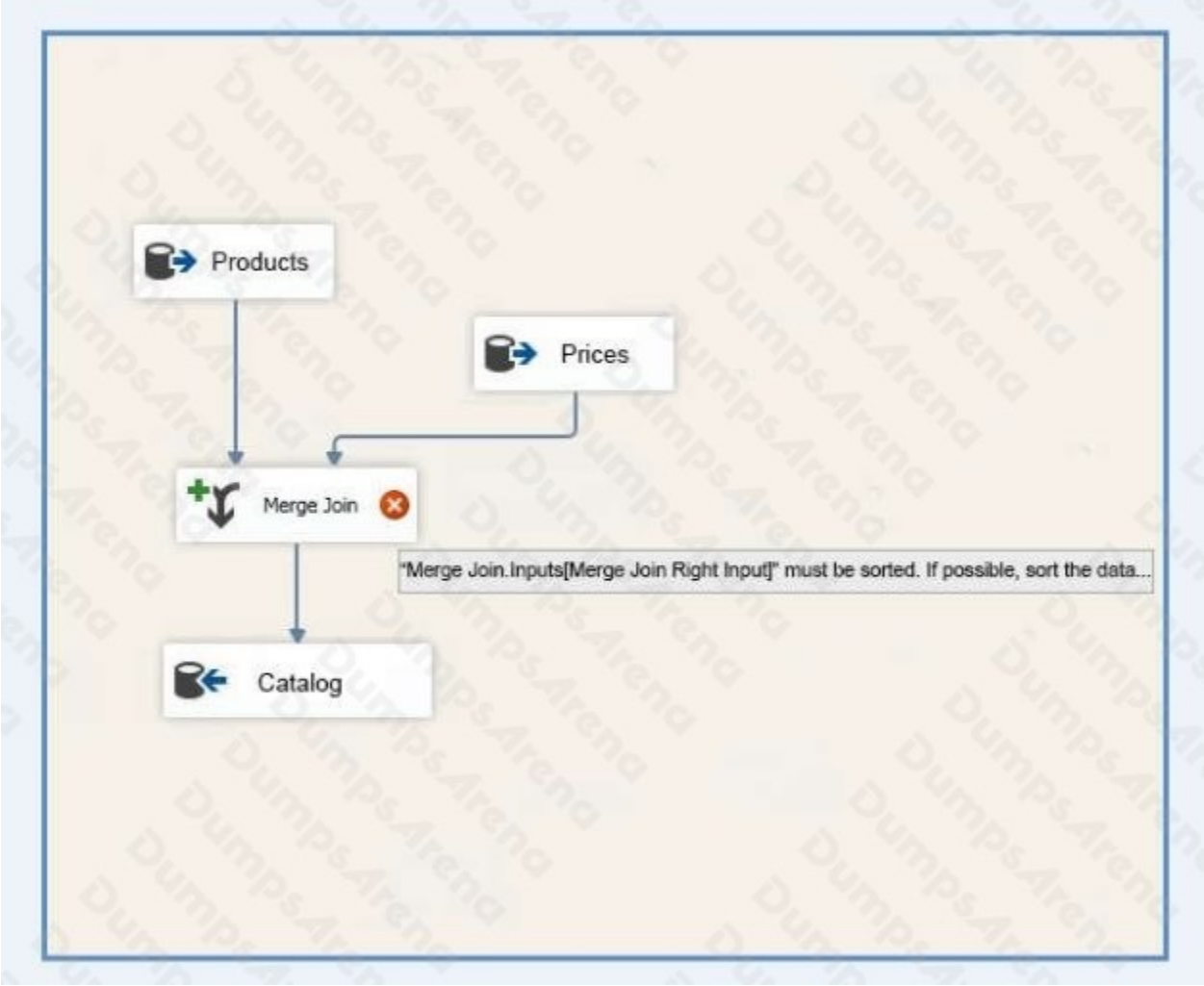
sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

QUESTION NO: 1 - (HOTSPOT)

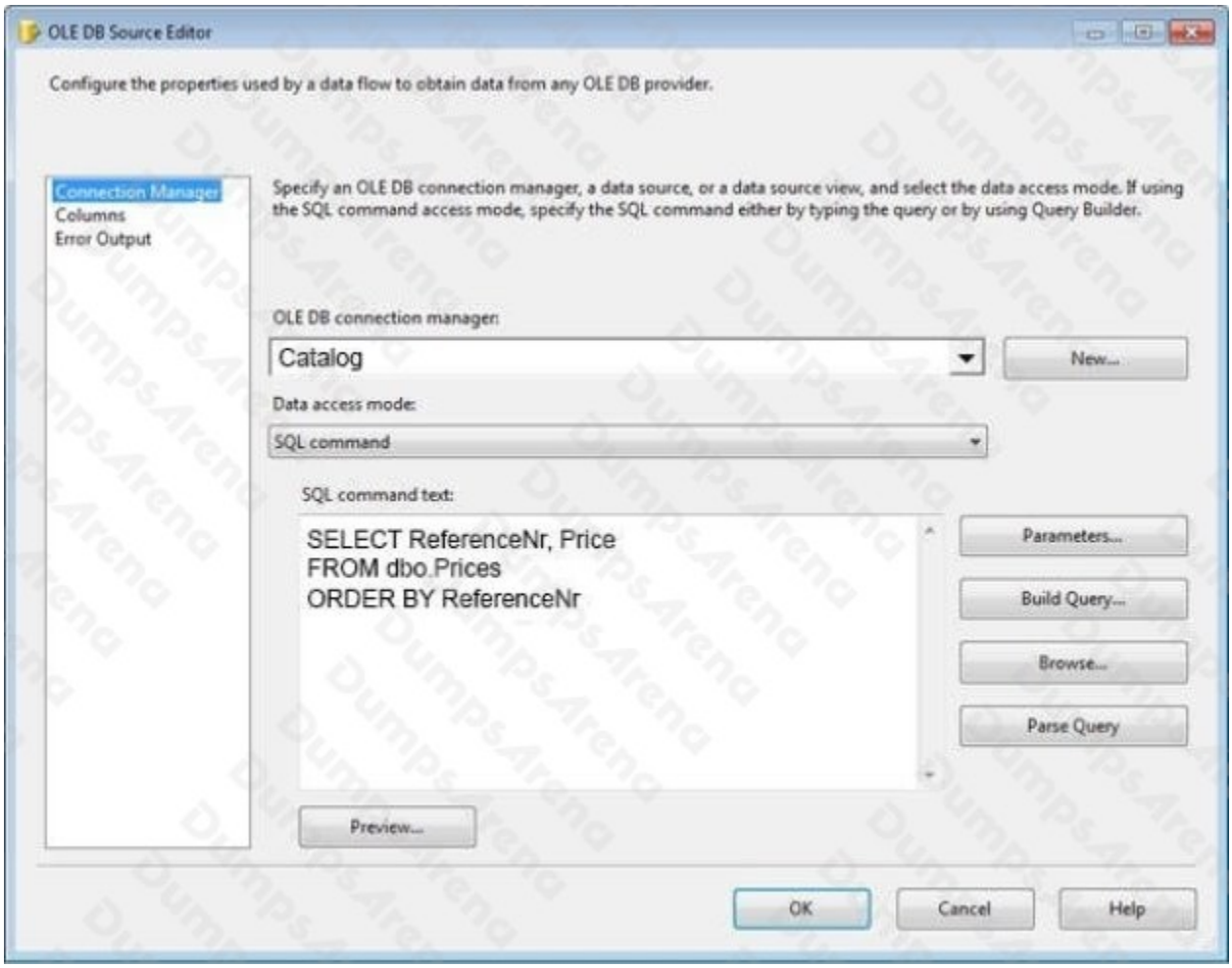
HOTSPOT

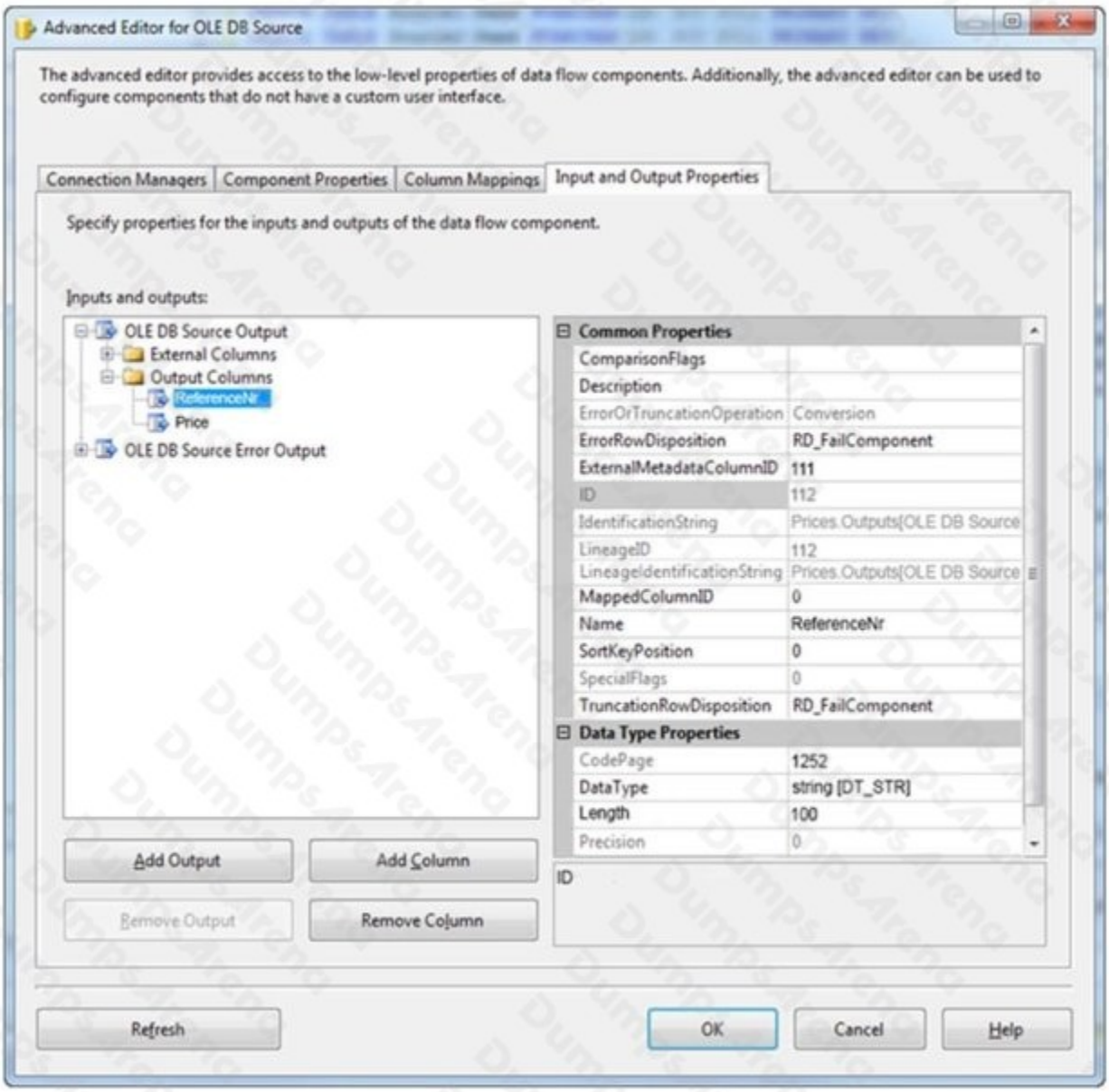
You create a Microsoft SQL Server Integration Services (SSIS) package as shown in the SSIS Package exhibit. (Click the Exhibit button.)



The package uses data from the Products table and the Prices table. Properties of the Prices source are shown in the OLE DB Source Editor exhibit

(Click the Exhibit Button.) and the Advanced Editor for Prices exhibit (Click the Exhibit button.)


OLE DB Source Editor
Configure the properties used by a data flow to obtain data from any OLE DB provider.
Connection Manager
Columns
Error Output
Specify an OLE DB connection manager, a data source, or a data source view, and select the data access mode. If using the SQL command access mode, specify the SQL command either by typing the query or by using Query Builder.
OLE DB connection manager:
Catalog
New...
Data access mode:
SQL command
SQL command text:
SELECT ReferenceNr, Price
FROM dbo.Prices
ORDER BY ReferenceNr
Parameters...
Build Query...
Browse...
Parse Query
Preview...
OK
Cancel
Help



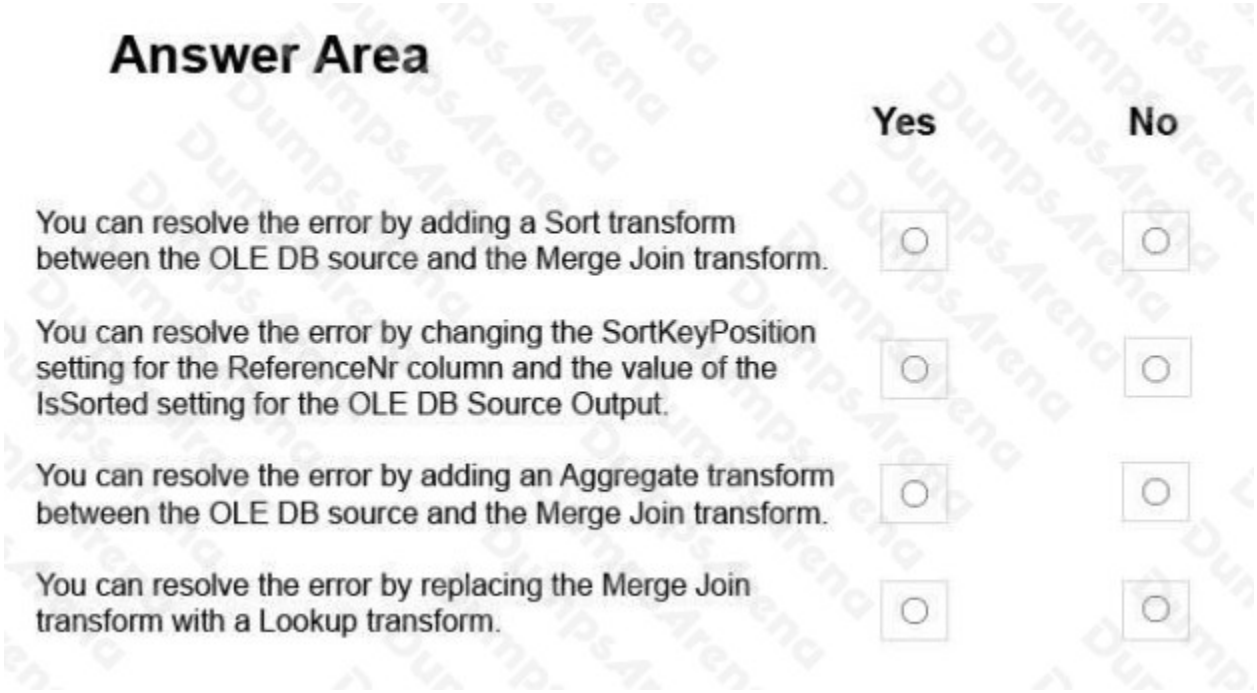
You join the Products and Prices tables by using the ReferenceNr column.

You need to resolve the error with the package.

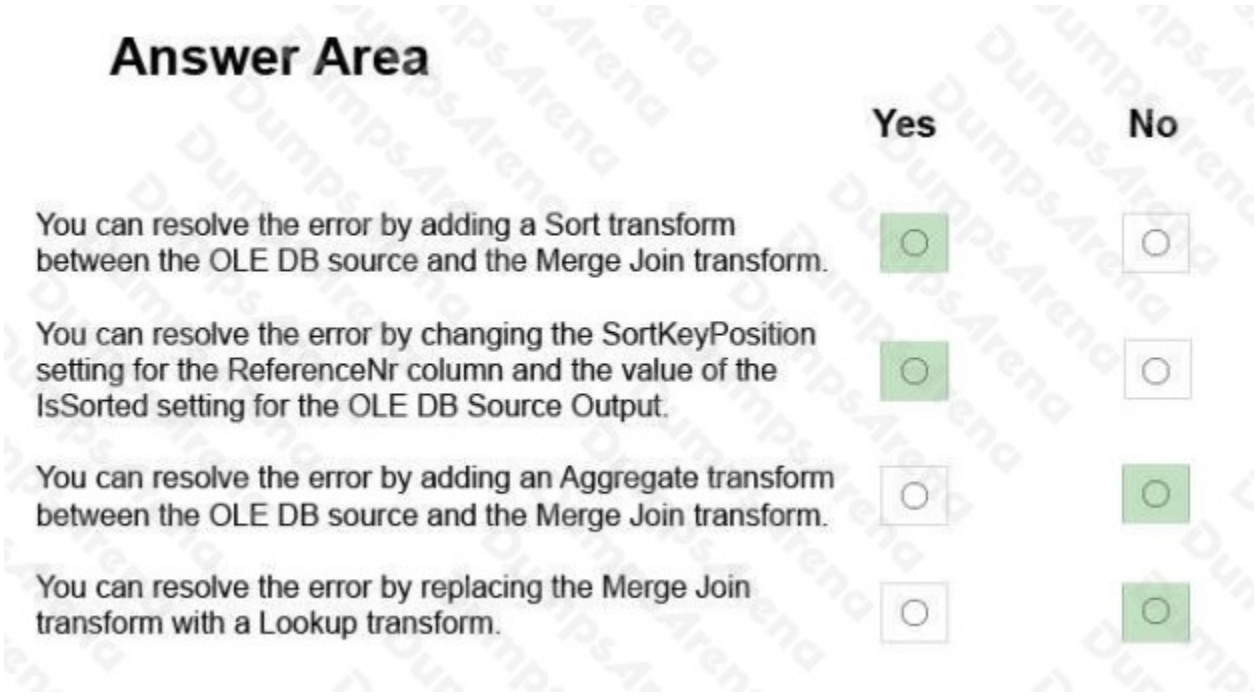
For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Hot Area:



ANSWER:



Explanation:

There are two important sort properties that must be set for the source or upstream transformation that supplies data to the Merge and Merge Join transformations:

The Merge Join Transformation requires sorted data for its inputs.

- The IsSorted property of the output that indicates whether the data has been sorted. This property must be set to True.

- The SortKeyPosition property of output columns that indicates whether a column is sorted, the column's sort order, and the sequence in which multiple columns are sorted. This property must be set for each column of sorted data.

If you do not use a Sort transformation to sort the data, you must set these sort properties manually on the source or the upstream transformation.

References: https://docs.microsoft.com/en-us/sql/integration-services/data-flow/transformations/sort-data-for-the-merge-and-merge-join-transformations

QUESTION NO: 2 - (DRAG DROP)

DRAG DROP

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 have a Microsoft SQL Server data warehouse instance that supports several client applications.

The data warehouse includes the following tables: Dimension.SalesTerritory, Dimension.Customer, Dimension.Date, Fact.Ticket, and Fact.Order. The Dimension.SalesTerritory and Dimension.Customer tables are frequently updated. The Fact.Order table is optimized for weekly reporting, but the company wants to change it to daily. The Fact.Order table is loaded by using an ETL process. Indexes have been added to the table over time, but the presence of these indexes slows data loading.

All data in the data warehouse is stored on a shared SAN. All tables are in a database named DB1. You have a second database named DB2 that contains copies of production data for a development environment. The data warehouse has grown and the cost of storage has increased. Data older than one year is accessed infrequently and is considered historical.

You have the following requirements:

- Implement table partitioning to improve the manageability of the data warehouse and to avoid the need to repopulate all transactional data each night. Use a partitioning strategy that is as granular as possible. - Partition the Fact.Order table and retain a total of seven years of data.

- Partition the Fact.Ticket table and retain seven years of data. At the end of each month, the partition structure must apply a sliding window strategy to ensure that a new partition is available for the upcoming month, and that the oldest month of data is archived and removed.

- Optimize data loading for the Dimension.SalesTerritory, Dimension.Customer, and Dimension.Date tables.

- Incrementally load all tables in the database and ensure that all incremental changes are processed.

- Maximize the performance during the data loading process for the Fact.Order partition.

- Ensure that historical data remains online and available for querying.

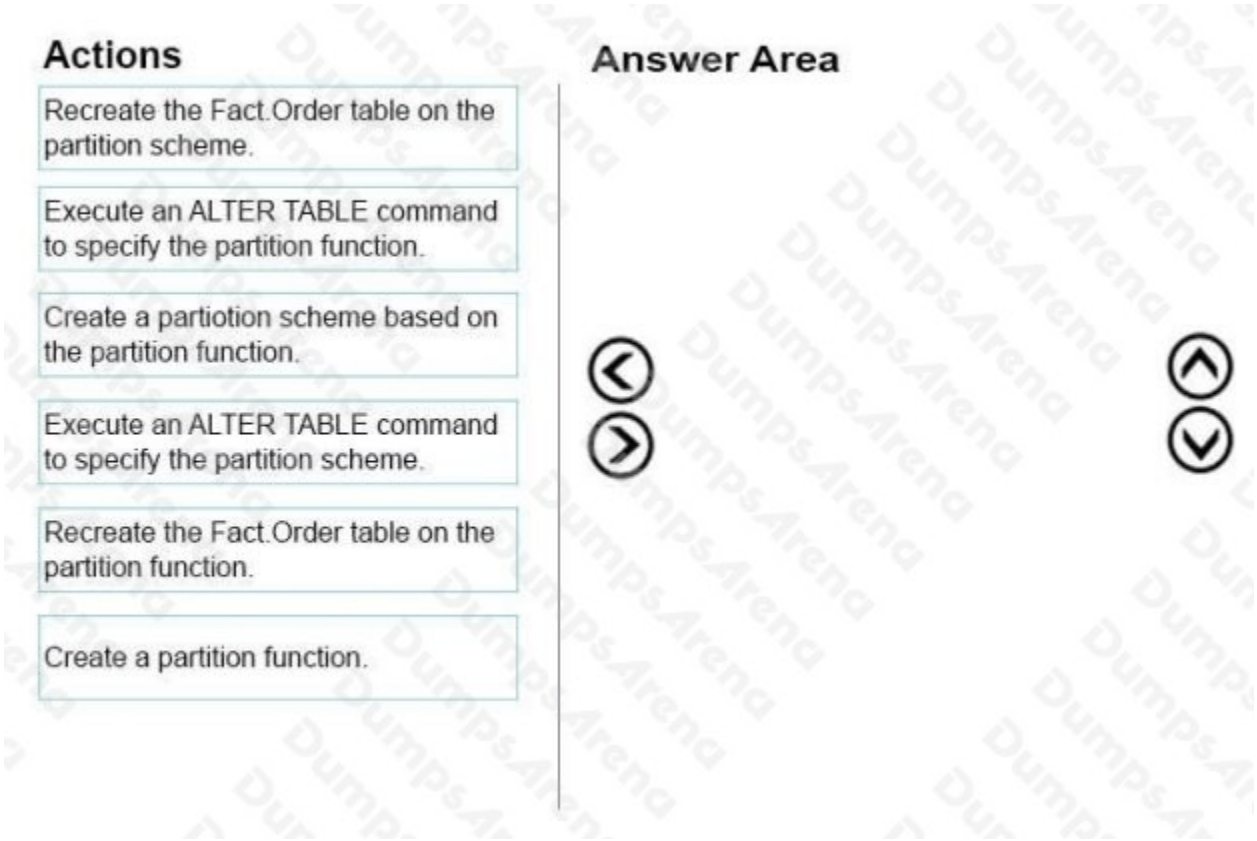
- Reduce ongoing storage costs while maintaining query performance for current data.

You are not permitted to make changes to the client applications.

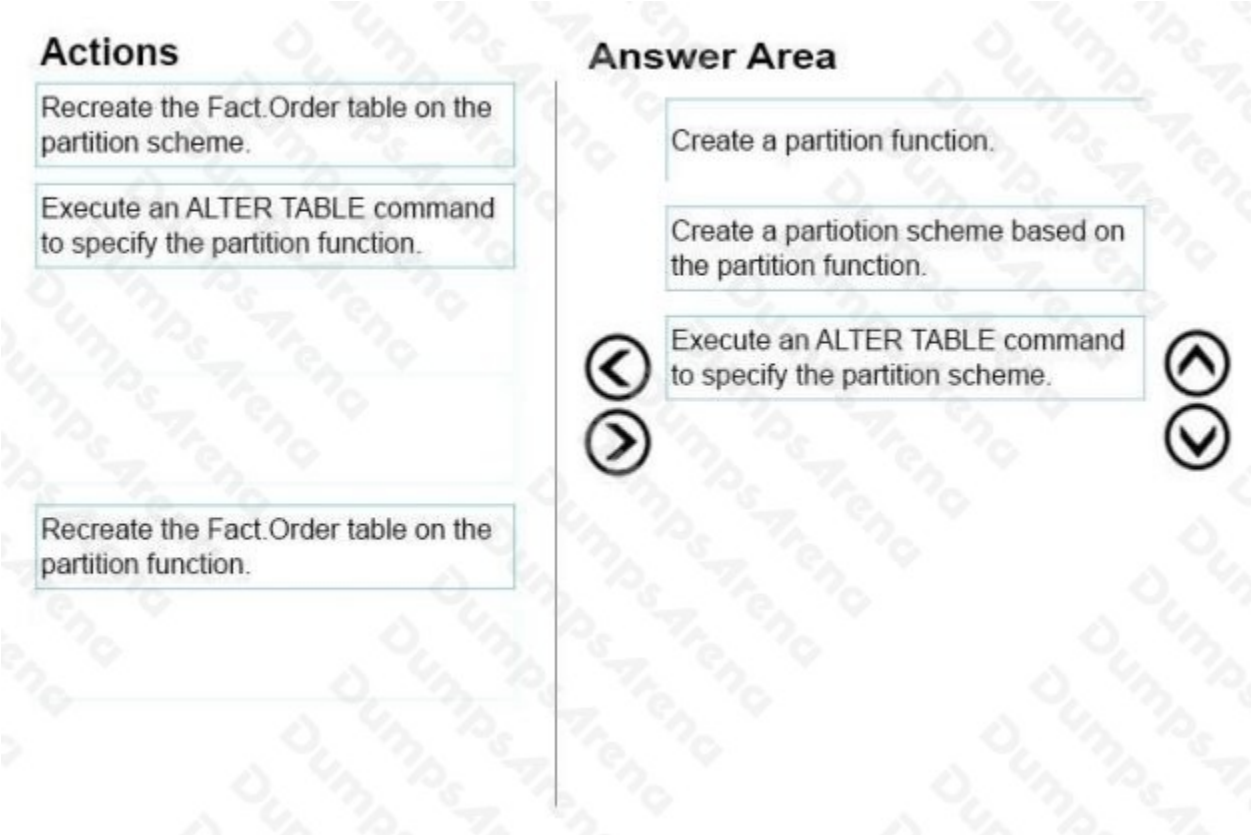
You need to configure the Fact.Order table.

Which three actions should you perform in sequence? To answer, move the appropriate actions from the list of actions to the answer area and arrange them in the correct order.

Select and Place:



ANSWER:



Explanation:

From scenario: Partition the Fact.Order table and retain a total of seven years of data. Maximize the performance during the data loading process for the Fact.Order partition.

Step 1: Create a partition function.

Using CREATE PARTITION FUNCTION is the first step in creating a partitioned table or index.

Step 2: Create a partition scheme based on the partition function.

To migrate SQL Server partition definitions to SQL Data Warehouse simply:

- Eliminate the SQL Server partition scheme.
- Add the partition function definition to your CREATE TABLE.

Step 3: Execute an ALTER TABLE command to specify the partition function.

References: https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-tables-partition

QUESTION NO: 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 have a Microsoft Azure SQL Data Warehouse instance that must be available six months a day for reporting.

You need to pause the compute resources when the instance is not being used.

Solution: You use the Azure portal.

Does the solution meet the goal?

A. Yes

B. No

ANSWER: A

Explanation:

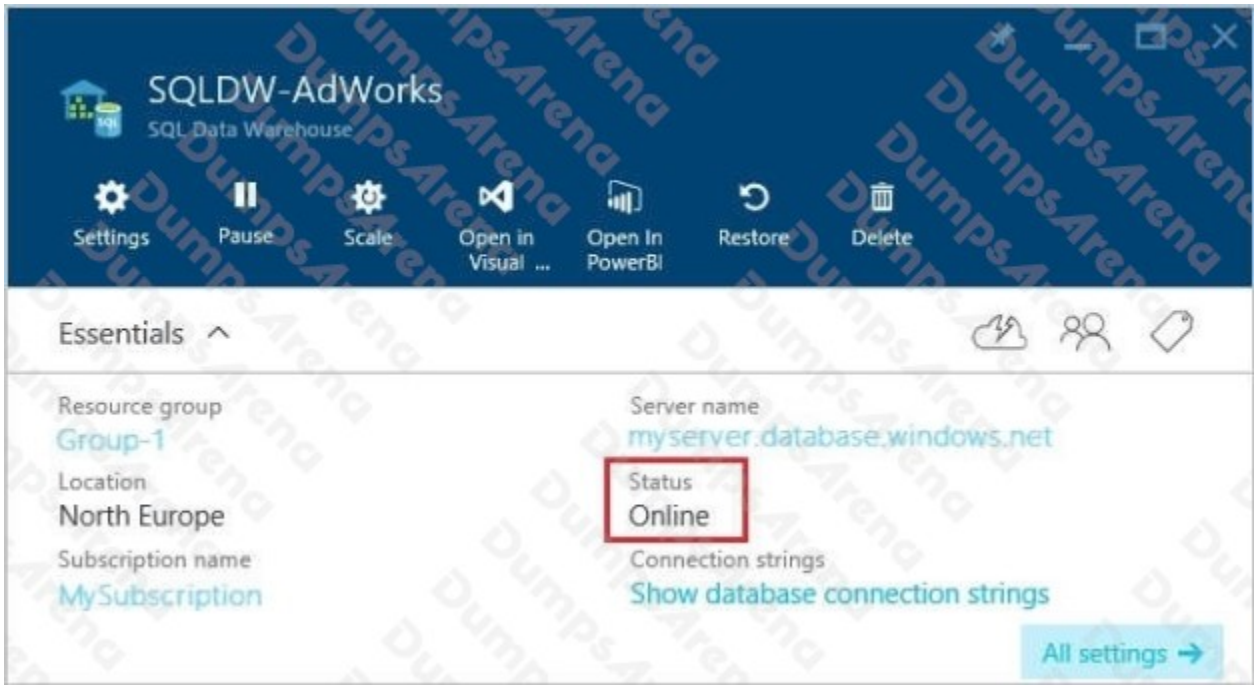
To pause a SQL Data Warehouse database, use any of these individual methods.

Pause compute with Azure portal

Pause compute with PowerShell Pause compute with REST APIs

Note: To pause a database:

1. Open the Azure portal and open your database. Notice that the Status is Online.



2. To suspend compute and memory resources, click Pause, and then a confirmation message appears. Click yes to confirm or no to cancel.

References: https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-manage-compute-overview

https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-manage-compute-portal#pause-compute-bk

QUESTION NO: 4

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 the series.

Start of repeated scenario

You have a Microsoft SQL Server data warehouse instance that supports several client applications.

The data warehouse includes the following tables: Dimension.SalesTerritory, Dimension.Customer, Dimension.Date, Fact.Ticket, and Fact.Order. The Dimension.SalesTerritory and Dimension.Customer tables are frequently updated. The Fact.Order table is optimized for weekly reporting, but the company wants to change it to daily. The Fact.Order table is loaded by using an ETL process. Indexes have been added to the table over time, but the presence of these indexes slows data loading.

All tables are in a database named DB1. You have a second database named DB2 that contains copies of production data for a development environment. The data warehouse has grown and the cost of storage has increased. Data older than one year is accessed infrequently and is considered historical.

The following requirements must be met:

- Implement table partitioning to improve the manageability of the data warehouse and to avoid the need to repopulate all transactional data each night. Use a partitioning strategy that is as granular as possible. - Partition the Fact.Order table and retain a total of seven years of data.

- Partition the Fact.Ticket table and retain seven years of data. At the end of each month, the partition structure must apply a sliding window strategy to ensure that a new partition is available for the upcoming month, and that the oldest month of data is archived and removed.

- Optimize data loading for the Dimension.SalesTerritory, Dimension.Customer, and Dimension.Date tables.

- Incrementally load all tables in the database and ensure that all incremental changes are processed.

- Maximize the performance during the data loading process for the Fact.Order partition.

- Ensure that historical data remains online and available for querying.

- Reduce ongoing storage costs while maintaining query performance for current data.

You are not permitted to make changes to the client applications.

End of repeated scenario

You need to implement the data partitioning strategy.

How should you partition the Fact.Order table?

A. Create 17,520 partitions.

B. Use a granularity of one day.

C. Use a granularity of one month.

D. Create 1,460 partitions.

ANSWER: B

Explanation:

We create on partition for each day, which means that a granularity of one day is used.

Note: If we calculate the partitions that are needed, we get: 7 years times 365 days is 2,555. Make that 2,557 to provide for leap years.

From scenario: Partition the Fact.Order table and retain a total of seven years of data.

The Fact.Order table is optimized for weekly reporting, but the company wants to change it to daily. Maximize the performance during the data loading process for the Fact.Order partition.

Reference: https://docs.microsoft.com/en-us/azure/sql-data-warehouse/sql-data-warehouse-tables-partition

QUESTION NO: 5 - (DRAG DROP)

DRAG DROP

You are developing a Microsoft SQL Server Integration Services (SSIS) package to incrementally load new and changed records from a data source.

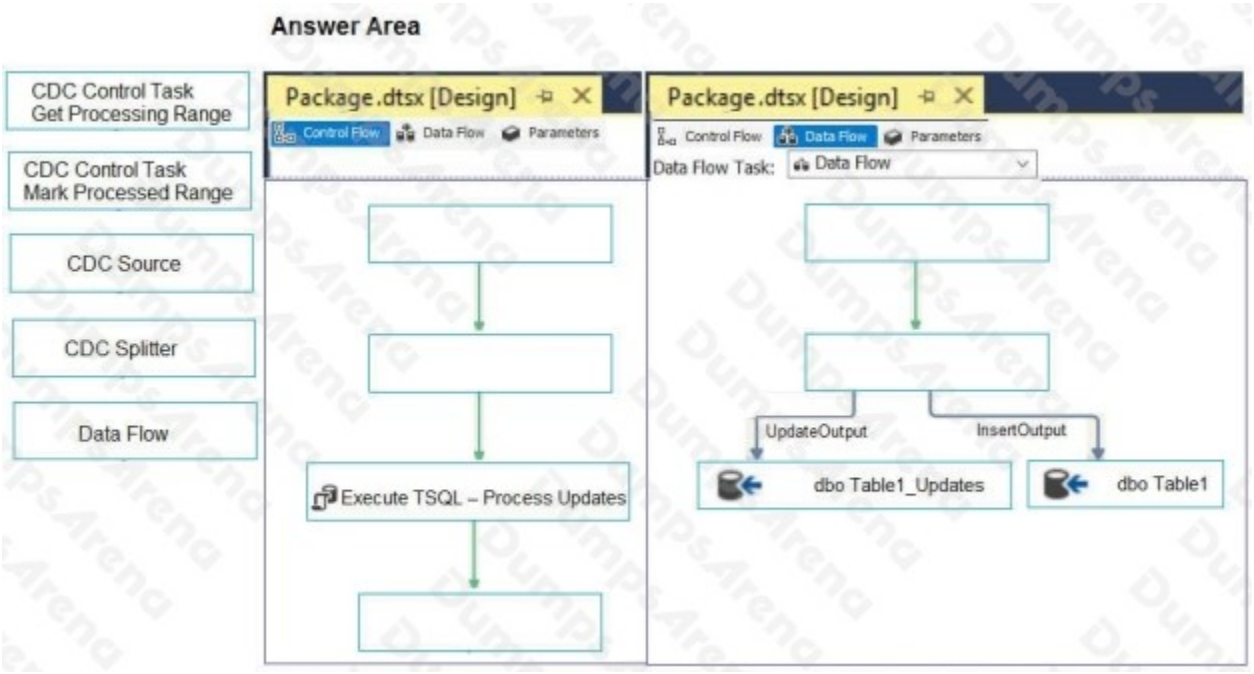
The SSIS package must load new records into Table1 and updated records into Table1_Updates. After loading records, the package must call a Transact-SQL statement to process updated rows according to existing business logic.

You need to complete the design of the SSIS package.

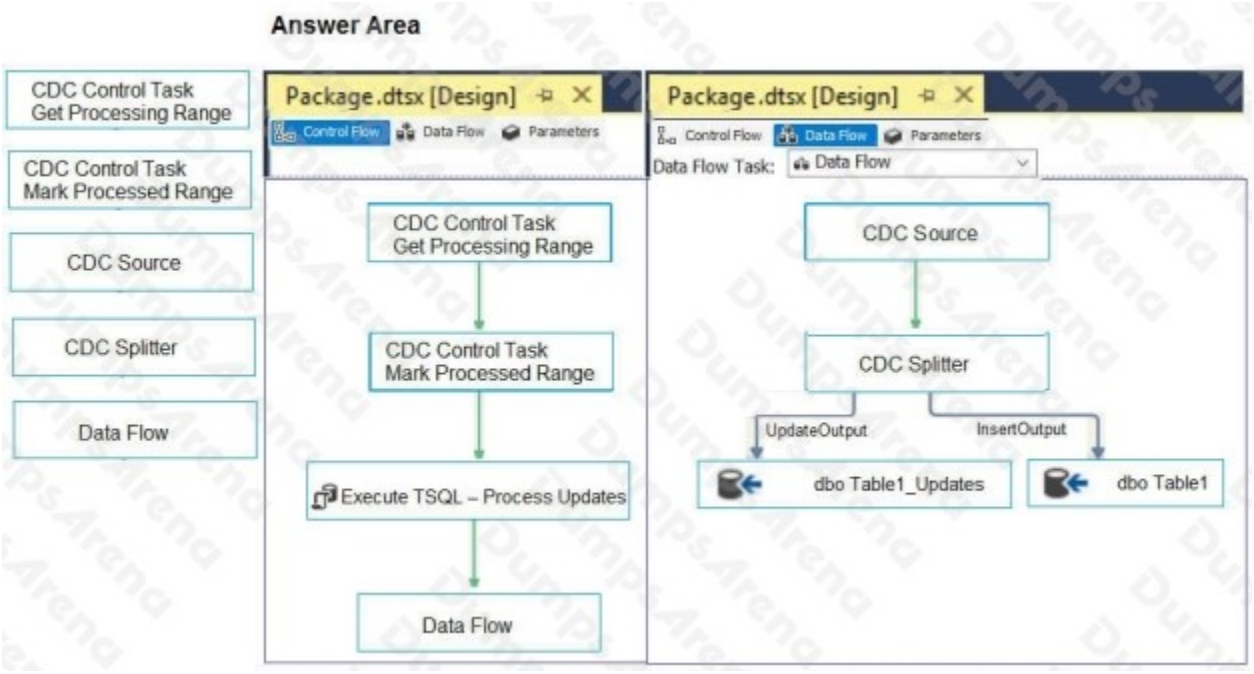
Which tasks should you use? To answer, drag the appropriate SSIS objects to the correct targets. Each SSIS object may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.

NOTE: Each correct selection is worth one point.

Select and Place:



ANSWER:



Explanation:

Step 1: CDC Control Task Get Processing Range

Step 2: Mark Processed Range

Step 3: Data Flow

The Data Flow task encapsulates the data flow engine that moves data between sources and destinations, and lets the user transform, clean, and modify data as it is moved. Addition of a Data Flow task to a package control flow makes it possible for the package to extract, transform, and load data.

Step 4: CDC Source

The CDC source reads a range of change data from SQL Server 2017 change tables and delivers the changes downstream to other SSIS component.

Step 5: CDC Splitter

The CDC splitter splits a single flow of change rows from a CDC source data flow into different data flows for Insert, Update and Delete operations.

References:

https://docs.microsoft.com/en-us/sql/integration-services/control-flow/cdc-control-task https://docs.microsoft.com/en-us/sql/integration-services/control-flow/data-flow-task

https://docs.microsoft.com/en-us/sql/integration-services/data-flow/cdc-splitter?view=sql-server-2017

QUESTION NO: 6

You have a data warehouse that contains a fact table named Table1 and a Product table named Dim1. Dim1 is configured as shown in the following table.

Column name	Column data type
ProductID	Integer identity
ProductKey	Char(10)
Name	Varchar(50)
Color	Varchar(20)
Weight	Decimal (13, 1)

You are adding a second OLTP system to the data warehouse as a new fact table named Table2. The Product table of the OLTP system is configured as shown in the following table

Column name	Column data type
ProductIdentifier	Char (8)
ProductName	Varchar(35)
SalesUnit	varchar(25)
Weight	Decimal(19,2)

You need to modify Dim1 to ensure that the table can be used for both fact tables.

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

NOTE: Each correct selection is worth one point.

A. Modify the data type of the Weight column in Dim1 to decimal (19, 2).

B. Add the SalesUnit column to Dim1.

C. Modify the data type of the Name column in Dim1 to varchar (85).

D. Drop the ProductKey column from Dim1 and replace the column with the ProductIdentifier column.

E. Drop the Color column from Dim1.

F. Modify the data type of the ProductKey column in Dim1 to char (18).

ANSWER: A D

QUESTION NO: 7 - (HOTSPOT)

HOTSPOT

You have a database that includes a table named dbo.Sales. The table contains two billion rows. You created the table by running the following Transact-SQL statement:

```
CREATE TABLE dbo.Sales (
    SaleId BIGINT PRIMARY KEY,
    StoreId INT,
    EmployeeId INT,
    SaleAmount MONEY,
    TaxAmount MONEY,
    SubTotalAmount MONEY,
    LineItems XML,
    Refund BIT,
    SaleDate DATE,
    SaleTime TIME
)
```

You run the following queries against the dbo.Sales. All the queries perform poorly.

Query name	Query text
Query1	SELECT StoreId, SUM(SaleAmount) SaleTotal, SUM(TaxAmount) TaxTotal FROM dbo.Sales WHERE SaleDate BETWEEN '1/1/2015' AND '1/1/2016' GROUP BY StoreId
Query2	SELECT StoreId, datepart(hh, SaleTime) SaleHour, count(*) FROM dbo.Sales WHERE SaleDate = convert(varchar(10), getdate()-1, 111) GROUP BY StoredId, datepart(hh, SaleTime)
Query3	SELECT SaleId, StoreId, EmployeeId, SaleAmount FROM dbo.Sales WHERE Refund = 1 AND SaleDate = convert(varchar(10), getdate()-1, 111)

The ETL process that populates the table uses bulk insert to load 10 million rows each day. The process currently takes six hours to load the records.

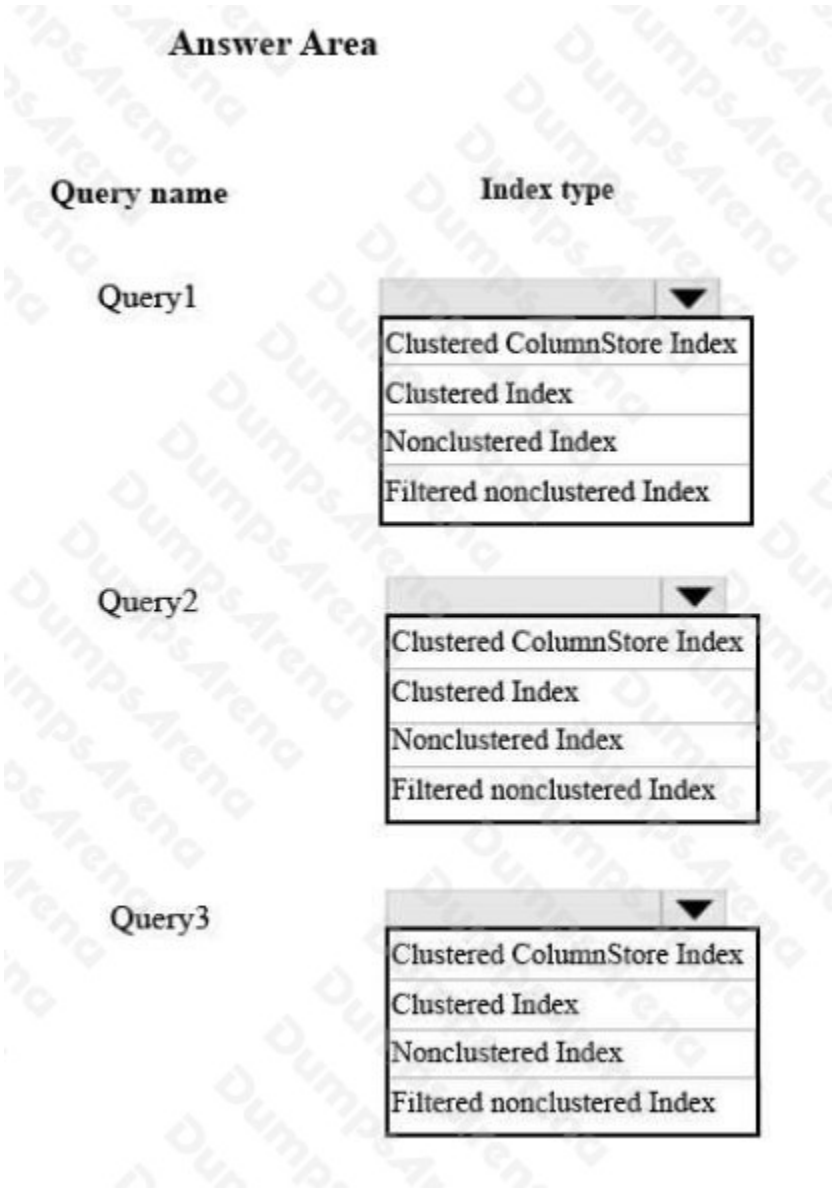
The value of the Refund column is equal to 1 for only 0.01 percent of the rows in the table. For all other rows, the value of the Refund column is equal to 0.

You need to maximize the performance of queries and the ETL process.

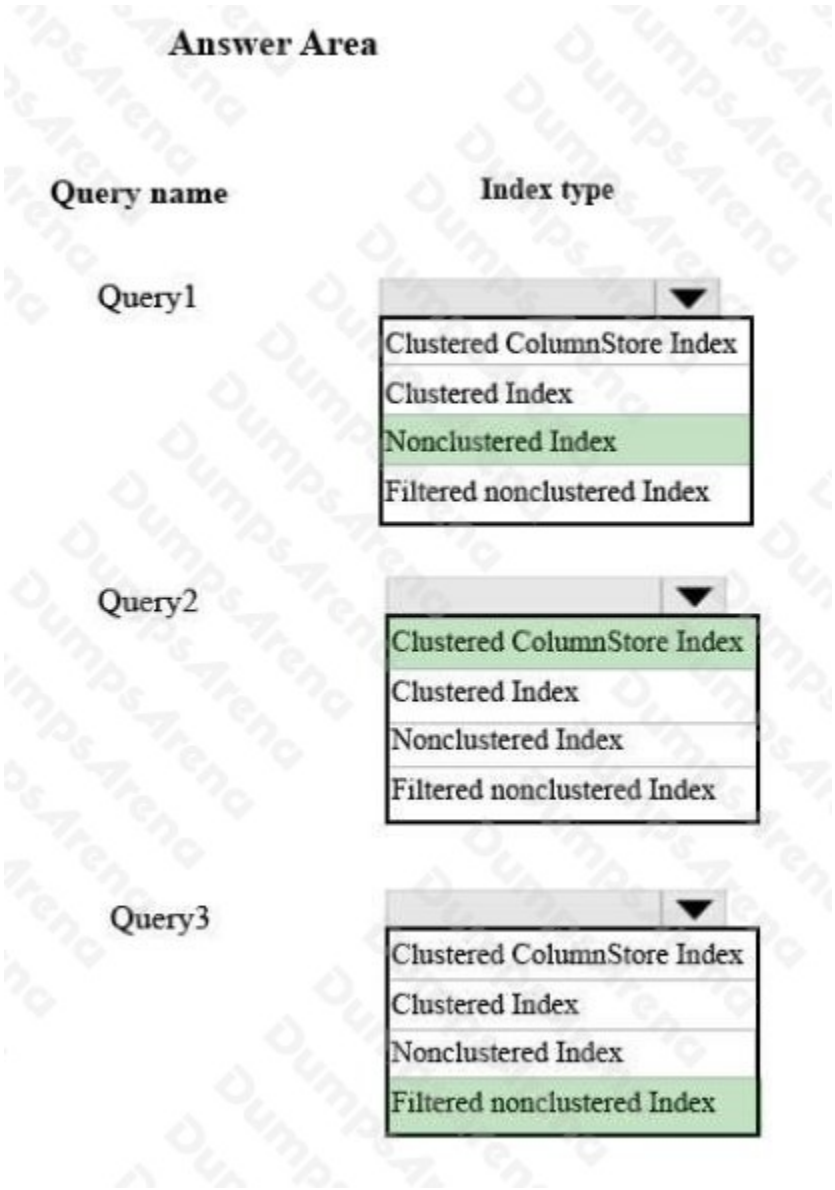
Which index type should you use for each query? To answer, select the appropriate index types in the answer area.

NOTE: Each correct selection is worth one point.

Hot Area:


Answer Area
Query name
Index type
Query1
Clustered ColumnStore Index
Clustered Index
Nonclustered Index
Filtered nonclustered Index
Query2
Clustered ColumnStore Index
Clustered Index
Nonclustered Index
Filtered nonclustered Index
Query3
Clustered ColumnStore Index
Clustered Index
Nonclustered Index
Filtered nonclustered Index

ANSWER:



Explanation:

Query1: Nonclustered Index

The query include a date range.

If you have included columns in your index, then the leaf level page of your non-clustered index contains the columns as defined in the nonclustered index the clustering key column(s) all those additional columns as defined in your INCLUDE statement.

Query2: Clustered columnstore index

Columnstore index is a new type of index introduced in SQL Server 2012. It is a column-based non-clustered index geared toward increasing query performance for workloads that involve large amounts of data, typically found in data warehouse fact tables.

Query3: Filtered nonclustered index

* When a column only has a small number of relevant values for queries, you can create a filtered index on the subset of values. For example, when the values in a column are mostly NULL and the query selects only from the non-NULL values, you can create a filtered index for the non-NULL data rows. The resulting index will be smaller and cost less to maintain than a full-table nonclustered index defined on the same key columns.

When a table has heterogeneous data rows, you can create a filtered index for one or more categories of data. This can improve the performance of queries on these data rows by narrowing the focus of a query to a specific area of the table. Again, the resulting index will be smaller and cost less to maintain than a full-table nonclustered index.

References:

https://docs.microsoft.com/en-us/sql/relational-databases/indexes/create-filtered-indexes https://logicalread.com/sql-server-columnstore-index-w02/#.XRo06egzaUk

QUESTION NO: 8

You manage the user accounts in master data Services (MDS).

You need to assign a user access to the MDS data and functions.

Which two components must you assign? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. file share permissions

B. model object permissions

C. functional area permissions

D. SQL Database permissions

ANSWER: B C

Explanation:

B: In Master Data Services, assign permissions to model objects when you need to give a user or group access to data in the Explorer functional area of Master Data Manager, or when you need to make a user or group an administrator.

C: Assign functional area permission to grant users or groups access to the functional areas of Master Data Manager.

To assign functional area permissions

1. In Master Data Manager, click User and Group Permissions.

2. On the Users or Groups page, select the row for the user or group that you want to edit.

3. Click Edit selected user.

4. Click the Functions tab.

5. Click Edit.

6. Click a functional area and click the Add arrow.

7. When you are done, click Save.

References:

https://docs.microsoft.com/en-us/sql/master-data-services/assign-model-object-permissions-master-data-services
https://docs.microsoft.com/en-us/sql/master-data-services/assign-functional-area-permissions-master-data-services

QUESTION NO: 9

Your company manufactures several types of products.

The company has a production tracking application that stores the following data about the products:

- The production date
- The cost of production
- The names of the products
- The amount of waste created
- The number of products produced
- The name of the facility where the products are produced

You are designing a data warehouse for the data. You add a Date dimension.

You need to ensure that you can create a composite primary key for the fact table.

Which two columns should you add to the new dimension tables? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. Cost of Production

B. Amount Produced

C. Waste Amount

D. Product Name

E. Facility Name

ANSWER: D E

Explanation:

Both the Product Name and the Facility Name are unique.

QUESTION NO: 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 have a Microsoft SQL Server data warehouse instance that supports several client applications.

The data warehouse includes the following tables: Dimension.SalesTerritory, Dimension.Customer, Dimension.Date, Fact.Ticket, and Fact.Order. The Dimension.SalesTerritory and Dimension.Customer tables are frequently updated. The Fact.Order table is optimized for weekly reporting, but the company wants to change it to daily. The Fact.Order table is loaded by using an ETL process. Indexes have been added to the table over time, but the presence of these indexes slows data loading.

All data in the data warehouse is stored on a shared SAN. All tables are in a database named DB1. You have a second database named DB2 that contains copies of production data for a development environment. The data warehouse has grown and the cost of storage has increased. Data older than one year is accessed infrequently and is considered historical.

You have the following requirements:

- Implement table partitioning to improve the manageability of the data warehouse and to avoid the need to repopulate all transactional data each night. Use a partitioning strategy that is as granular as possible.
- Partition the Fact.Order table and retain a total of seven years of data.

- Partition the Fact.Ticket table and retain seven years of data. At the end of each month, the partition structure must apply a sliding window strategy to ensure that a new partition is available for the upcoming month, and that the oldest month of data is archived and removed.

- Optimize data loading for the Dimension.SalesTerritory, Dimension.Customer, and Dimension.Date tables.

- Incrementally load all tables in the database and ensure that all incremental changes are processed.

- Maximize the performance during the data loading process for the Fact.Order partition.

- Ensure that historical data remains online and available for querying.

- Reduce ongoing storage costs while maintaining query performance for current data.

You are not permitted to make changes to the client applications.

You need to optimize the storage for the data warehouse.

What change should you make?

A. Partition the Fact.Order table, and move historical data to new filegroups on lower-cost storage.

B. Create new tables on lower-cost storage, move the historical data to the new tables, and then shrink the database.

C. Remove the historical data from the database to leave available space for new data.

D. Move historical data to new tables on lower-cost storage.

E. Implement row compression for the Fact.Order table.

F. Move the index for the Fact.Order table to lower-cost storage.

ANSWER: A

Explanation:

Create the load staging table in the same filegroup as the partition you are loading.

Create the unload staging table in the same filegroup as the partition you are deleting.

From scenario: The data warehouse has grown and the cost of storage has increased. Data older than one year is accessed infrequently and is considered historical.

References: https://blogs.msdn.microsoft.com/sqlcat/2013/09/16/top-10-best-practices-for-building-a-large-scale-relational-data-warehouse/