DUMPSARENA

Implementing Data Engineering Solutions Using Microsoft Fabric

Microsoft DP-700

Version Demo

Total Demo Questions: 10

Total Premium Questions: 104

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

Topic Break Down

Topic	No. of Questions
Topic 1, Contoso, Ltd	9
Topic 2, Litware, Inc	5
Topic 3, Misc. Questions Set	90
Total	104

QUESTION NO: 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 Fabric eventstream that loads data into a table named Bike_Location in a KQL database. The table contains the following columns:

BikepointID

Street

Neighbourhood

No_Bikes

No_Empty_Docks

Timestamp

You need to apply transformation and filter logic to prepare the data for consumption. The solution must return data for a neighbourhood named Sands End when No_Bikes is at least 15. The results must be ordered by No_Bikes in ascending order. Solution: You use the following code segment:

```
bike_location
| filter Neighbourhood == "Sands End" and No_Bikes >= 15
| order by No_Bikes
| project BikepointID, Street, Neighbourhood, No_Bikes, No_Empty_Docks, Timestamp
```

Does this meet the goal?

A. Yes

B. no

ANSWER: B

Explanation:

No, this doesn't fully meet the goal because the query uses the wrong KQL operator for sorting. In Kusto Query Language, you sort results with **sort by**, not **order by**. So even though the filter logic is correct (Neighbourhood is "Sands End" and No_Bikes is at least 15), the query would fail or not run as written due to that invalid keyword.

To fix it, replace **| order by No_Bikes** with **| sort by No_Bikes asc**. After that change, the results will be properly sorted in ascending order, and the projection list is fine for returning just the columns you want.

References: https://learn.microsoft.com/en-us/kusto/query/sort-operator and https://learn.microsoft.com/en-us/kusto/query/where-operator

QUESTION NO: 2 - (HOTSPOT)

HOTSPOT

You have a Fabric workspace.

You are debugging a statement and discover the following issues:

Sometimes, the statement fails to return all the expected rows.

The PurchaseDate output column is NOT in the expected format of mmm dd, yy. You need to resolve the issues. The solution must ensure that the data types of the results are retained. The results can contain blank cells.

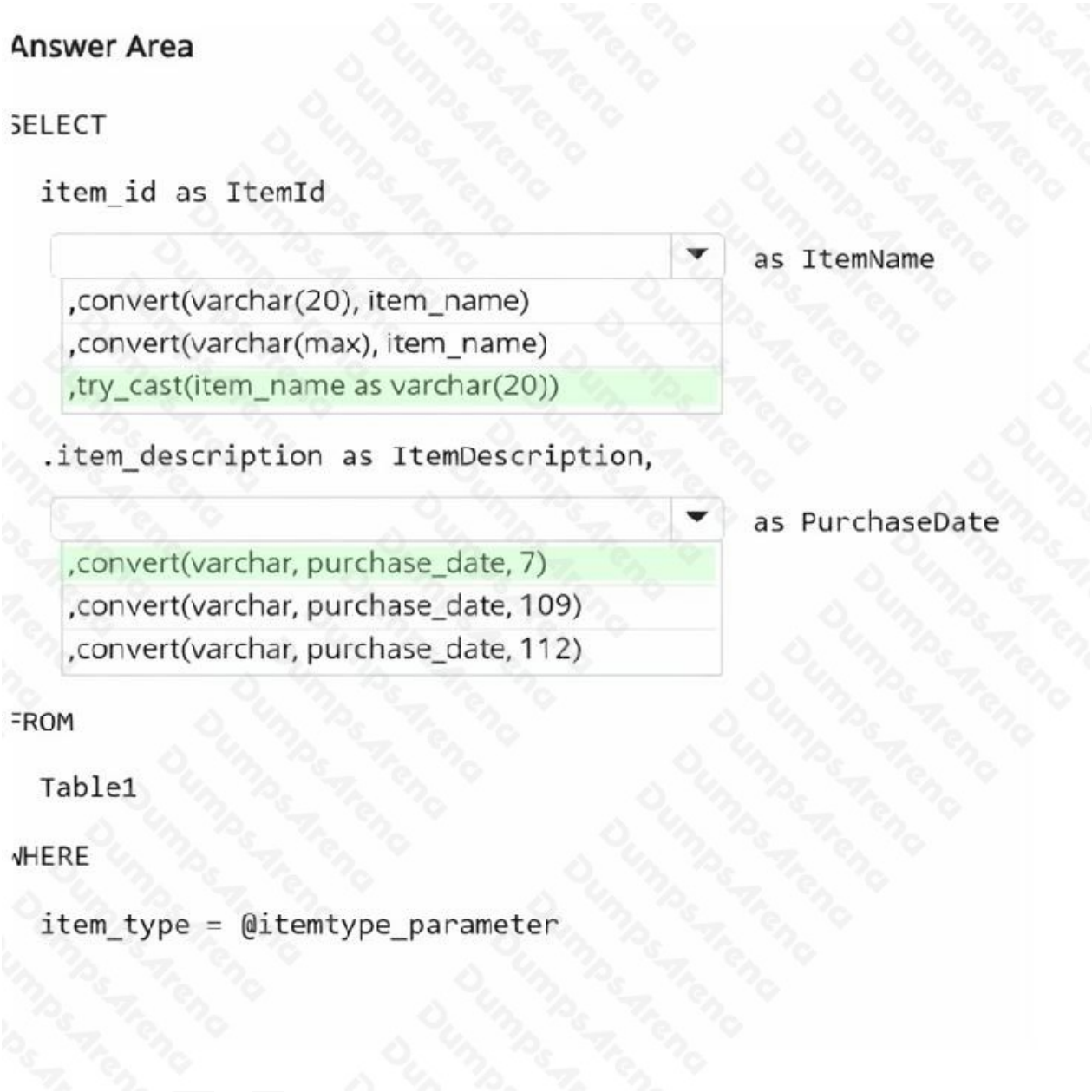
How should you complete the statement? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Answer Area

```
SELECT
  item_id as ItemId
  [ ,convert(varchar(20), item_name) | ,convert(varchar(max), item_name) | ,try_cast(item_name as varchar(20)) ] as ItemName
  .item_description as ItemDescription,
  [ ,convert(varchar, purchase_date, 7) | ,convert(varchar, purchase_date, 109) | ,convert(varchar, purchase_date, 112) ] as PurchaseDate
FROM
  Table1
WHERE
  item_type = @itemtype_parameter
```

ANSWER:



Explanation:

ItemName,try_cast(item_name as varchar(20))

PurchaseDate,convert(varchar, purchase_date, 7)

1. Why TRY_CAST?

The prompt mentions that the statement sometimes fails to return all expected rows. In SQL, if a standard CAST or CONVERT encounters a value it cannot transform (e.g., trying to cast "ABC" to an integer), the entire query fails.

2. Why CONVERT with Style 7?

The requirement specifies the date must be in the format mmm dd, yy (e.g., "Feb 13, 26").

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 Fabric eventstream that loads data into a table named Bike_Location in a KQL database. The table contains the following columns:

BikepointID

Street

Neighbourhood

No_Bikes

No_Empty_Docks

Timestamp

You need to apply transformation and filter logic to prepare the data for consumption. The solution must return data for a neighbourhood named Sands End when No_Bikes is at least 15. The results must be ordered by No_Bikes in ascending order.

Solution: You use the following code segment:

```
bike_location
| filter Neighbourhood == "Sands End" and No_Bikes >= 15
| sort by No_Bikes asc
| project BikepointID, Street, Neighbourhood, No_Bikes, No_Empty_Docks, Timestamp
```

Does this meet the goal?

A. Yes

B. no

ANSWER: B

Explanation:

This solution is close, but it doesn't fully meet the requirement as written. The question says you must include rows where `No_Bikes` is *at least 15*, which means `No_Bikes >= 15`. The query uses `No_Bikes > 15`, so it would incorrectly drop any rows where `No_Bikes` is exactly 15.

Also, the sorting line shown is not valid KQL syntax. In KQL you'd normally write something like `| sort by No_Bikes asc`. The provided `sort by No_Bikes_asc` looks like a column name, not the sort direction, so it won't reliably produce the required ascending order.

If you change the filter to `>= 15` and sort to `| sort by No_Bikes asc`, then it would match the goal. For reference on correct KQL filtering and sorting syntax, see https://learn.microsoft.com/en-us/azure/data-explorer/kusto/query/where-operator and https://learn.microsoft.com/en-us/azure/data-explorer/kusto/query/sort-operator.

QUESTION NO: 4 - (HOTSPOT)

HOTSPOT

You have a Fabric workspace that contains a warehouse named Warehouse!. Warehousel contains a table named DimCustomers. DimCustomers contains the following columns:

CustomerName

CustomerlD BirthDate

Email

You need to configure security to meet the following requirements:

BirthDate in DimCustomer must be masked and display 1900-01-01.

Email in DimCustomer must be masked and display only the first leading character and the last five characters.

How should you complete the statement? To answer, select the appropriate options in the answer area. NOTE: Each correct selection is worth one point.

Answer Area

ANSWER:

Explanation:

'default()'

'email()'

Dynamic Data Masking (DDM) allows you to protect sensitive data at the column level while ensuring it remains usable for analysts.

1. BirthDate Masking: default()

The requirement is to display 1900-01-01 for the BirthDate column.

2. Email Masking: email()

The requirement is to display only the first character and the last five characters of the email address.

QUESTION NO: 5

You need to recommend a solution for handling old files. The solution must meet the technical requirements. What should you include in the recommendation?

A. a data pipeline that includes a Copy data activity

B. a notebook that runs the VACUUM command

C. a notebook that runs the OPTIMIZE command

D. a data pipeline that includes a Delete data activity

ANSWER: B

Explanation:

If the goal is to deal with “old files” in a Delta table (for example, outdated parquet files left behind after updates/merges), the right tool is **VACUUM**. VACUUM removes old, unreferenced data files that are no longer needed for the table’s current state, which is exactly what you typically mean by cleaning up old files and controlling storage growth.

OPTIMIZE is different: it rewrites many small files into fewer larger ones to improve query performance, but it doesn’t remove old versions. And pipeline activities like Copy data or Delete data aren’t the normal or safest way to clean up Delta’s historical files, because Delta’s transaction log controls what’s valid vs. obsolete.

So the best recommendation is a notebook that runs the **VACUUM** command (often scheduled), using an appropriate retention period to avoid breaking time travel requirements. References: https://learn.microsoft.com/en-us/azure/databricks/delta/vacuum and https://learn.microsoft.com/en-us/fabric/data-engineering/lakehouse-delta-table

QUESTION NO: 6

You have a Fabric workspace named Workspace1 that contains a lakehouse named Lakehouse1. Lakehouse1 contains the following tables:

Orders

Customer

Employee

The Employee table contains Personally Identifiable Information (PII).

A data engineer is building a workflow that requires writing data to the Customer table, however, the user does NOT have the elevated permissions required to view the contents of the Employee table. You need to ensure that the data engineer can write data to the Customer table without reading data from the Employee table.

Which three actions should you perform? Each correct answer presents part of the solution. NOTE: Each correct selection is worth one point.

A. Share Lakehouse1 with the data engineer.

B. Assign the data engineer the Contributor role for Workspace2.

C. Assign the data engineer the Viewer role for Workspace2.

D. Assign the data engineer the Contributor role for Workspace1.

E. Migrate the Employee table from Lakehouse1 to Lakehouse2.

F. Create a new workspace named Workspace2 that contains a new lakehouse named Lakehouse2.

G. Assign the data engineer the Viewer role for Workspace1.

ANSWER: A D E

Explanation:

In Fabric, permissions are mainly applied at the workspace and item level, not per-table inside a lakehouse. So if someone can open the lakehouse with enough rights to write data, they can usually also see the tables in that same lakehouse. That’s why the cleanest way to stop accidental access to PII is to separate it.

First, share Lakehouse1 with the data engineer so they can actually use the lakehouse item. Then, give them the Contributor role on Workspace1 so they can run jobs and write data into the Customer table (Viewer wouldn’t be enough for writing).

Finally, move the sensitive Employee table out of Lakehouse1 into a different lakehouse (Lakehouse2). With the PII table physically separated, you can keep the engineer's access limited to Lakehouse1 while locking down Lakehouse2 to only the people who should see Employee data.

References: https://learn.microsoft.com/en-us/fabric/get-started/roles-workspaces https://learn.microsoft.com/en-us/fabric/data-engineering/lakehouse-overview

QUESTION NO: 7

You have a Fabric workspace that contains a lakehouse named Lakehouse1. Data is ingested into Lakehouse1 as one flat table. The table contains the following columns.

Name	Description
TransactionID	Contains a unique ID for each transaction
Date	Contains the date of a transaction
ProductID	Contains a unique ID for each product
ProductColor	Contains a descriptive attribute that describes the color of each product
ProductName	Contains a unique name for each product
SalesAmount	Contains the sales amount of a transaction

You plan to load the data into a dimensional model and implement a star schema. From the original flat table, you create two tables named FactSales and DimProduct. You will track changes in DimProduct.

You need to prepare the data.

Which three columns should you include in the DimProduct table? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. Date

B. ProductName

C. ProductColor

D. TransactionID

E. SalesAmount

F. ProductID

ANSWER: B C F

Explanation:

In a star schema, the fact table (FactSales) holds the measurable transaction data, while the dimension table (DimProduct) holds the descriptive details you want to slice and filter by. So things like **SalesAmount**, **Date**, and **TransactionID** belong with the sales event in the fact table, not in the product dimension.

For **DimProduct**, you want a stable product key plus the product attributes that can change over time (since you're tracking changes). **ProductID** is the natural identifier you'll use to relate sales to the product, and attributes like **ProductName** and **ProductColor** are exactly the kind of descriptive fields that belong in a product dimension and are commonly tracked as slowly changing attributes.

This setup keeps the model clean: FactSales stays lean and numeric, and DimProduct carries the "what is the product?" context for reporting and historical tracking. For more on star schemas and dimension vs. fact tables, see https://learn.microsoft.com/en-us/power-bi/guidance/star-schema and https://learn.microsoft.com/en-us/power-bi/transform-model/desktop-dimensional-modeling.

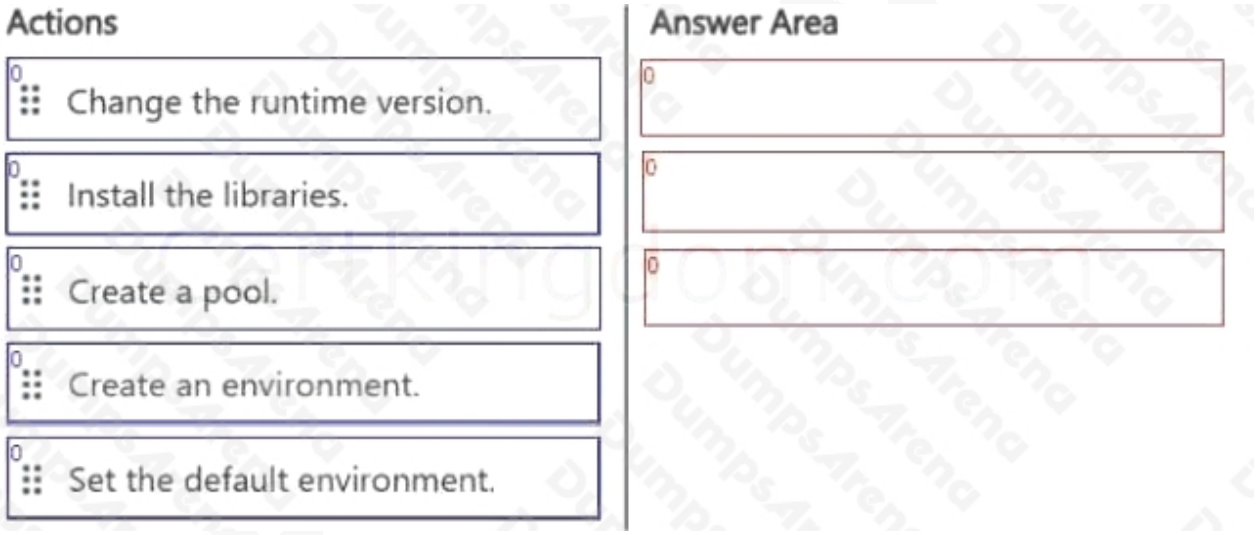
QUESTION NO: 8 - (DRAG DROP)

DRAG DROP

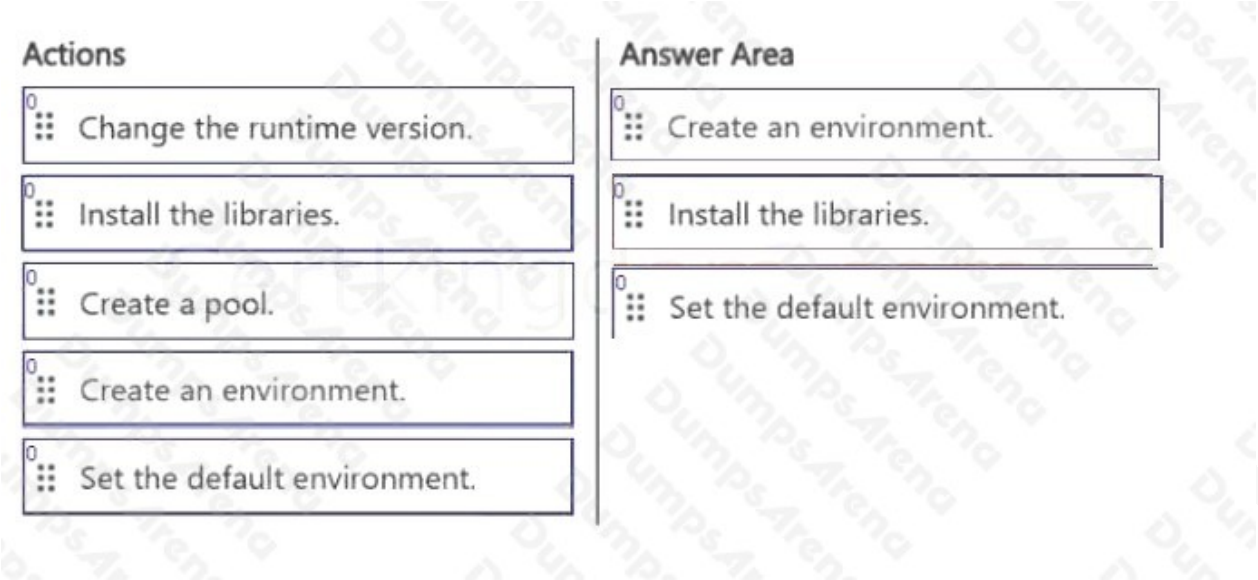
Your company has a team of developers. The team creates Python libraries of reusable code that is used to transform data.

You create a Fabric workspace name Workspace1 that will be used to develop extract, transform, and load (ETL) solutions by using notebooks.

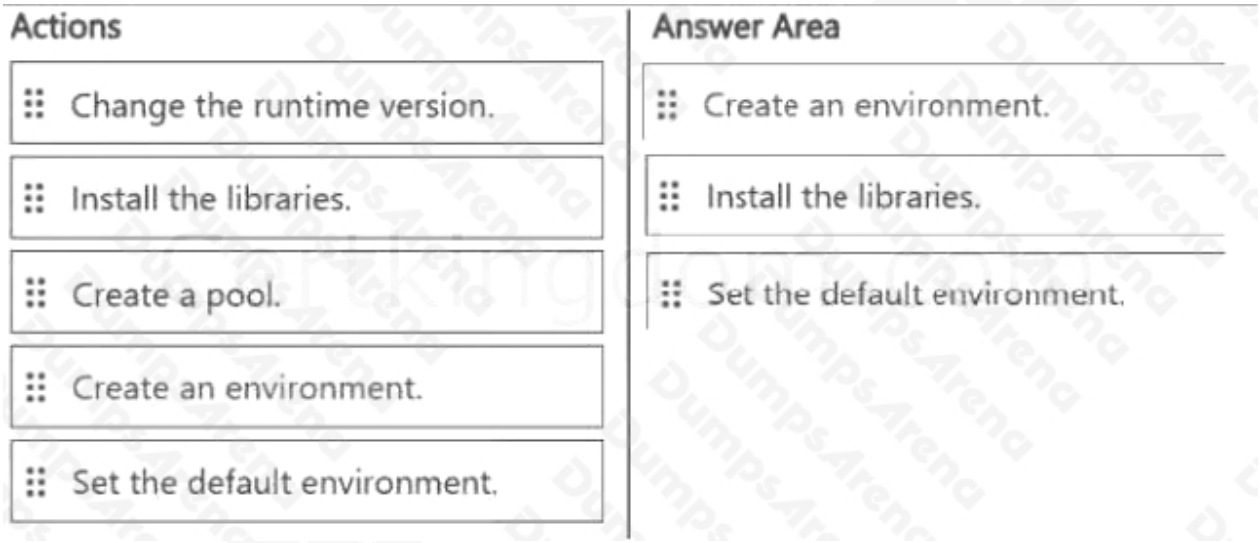
You need to ensure that the libraries are available by default to new notebooks in Workspace1. Which three actions should you perform in sequence? To answer, move the appropriate actions from the list of actions to the answer area and arrange them in the correct order.



ANSWER:



Explanation:



This workflow leverages the Fabric Environment item, which is the centralized way to manage Spark runtime settings and libraries at a workspace level.

1. Create an environment

In Microsoft Fabric, an Environment item serves as a container for your specific Spark configurations, including compute settings and public/private libraries. You must first create this dedicated environment item within your workspace to hold the developers' reusable code.

2. Install the libraries

Once the environment is created, you must upload and install the custom Python libraries into it.

3. Set the default environment

To ensure these libraries are available by default to new notebooks, you must navigate to the Workspace settings.

QUESTION NO: 9

You have a Fabric workspace that contains an eventhouse and a KQL database named Database1. Database1 has the following:

A table named Table1

A table named Table2

An update policy named Policy1

Policy1 sends data from Table1 to Table2.

The following is a sample of the data in Table2.

Timestamp (datetime)	DeviceId (guid)	StreamData (dynamic)
2024-05-18 12:45:17.16524	81416f30-60a2-4e75-9b19-2a84ea059735	[{ "index": 0, "eventid": "719afca0-be30-4559-bb5e-59feade642f6" }]
2024-05-18 12:45:21.76423	bb664e1e-02aa-4e17-8c8a-116cd4458d52	[{ "index": 0, "eventid": "782222b2-fbcb-43c0-82d6-ecd49a99dbf5 }]
2024-05-18 12:45:23.98642	717bfe7d-0e5d-498f-9f21-e60aaf258056	[{ "index": 0, "eventid": "d5730286-0da4-41f8-8e59-f75e209310a9 }]

Recently, the following actions were performed on Table1:

An additional element named temperature was added to the StreamData column.

The data type of the Timestamp column was changed to date.

The data type of the DeviceId column was changed to string.

You plan to load additional records to Table2.

Which two records will load from Table1 to Table2? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

A)

Timestamp (datetime)	DeviceId (guid)	StreamData (dynamic)
2024-05-18	81416f30-60a2-4e75-9b19-2a84ea059735	[{ "index": 40, "eventid": "729afca2-be30-4559-bb5e-59feade642f3", "temperature": 32 }]

B)

Timestamp (datetime)	DeviceId (guid)	StreamData (dynamic)
2024-05-21	81416f30	[{ "index": 0, "eventid": "719afca0-be30-4559-bb5e-5werade642f6", "temperature": 27 }]

C)

Timestamp (datetime)	DeviceId (guid)	StreamData (dynamic)
2024-05-23	81416f3060a24e759b192a84ea05973532dhdyte3	[{ "index": 0, "eventid": "719afca0-be30-4559-bb5e-59feade642f6" }]

D)

Timestamp (datetime)	DeviceId (guid)	StreamData (dynamic)
2024-05-24	81416f30-60a2-4e75-9b19-2a84ea059735	[{ "index": 0, "eventid": "719afca0-be30-4559-bb5e-59feade642f6" }]

A. Option A

B. Option B

C. Option c

D. Option D

ANSWER: B D

Explanation:

Table2 has a fixed schema: **Timestamp** is **datetime**, **DeviceId** is a **guid**, and **StreamData** is **dynamic**. Even though Table1 was changed (Timestamp to date, DeviceId to string, and StreamData got a new *temperature* field), the update policy can only insert rows into Table2 when the incoming values can still be interpreted as Table2's column types.

Option **B** will load because the Timestamp is still a valid datetime value and the DeviceId value can be treated as a GUID, and StreamData is still valid dynamic JSON (extra fields like *temperature* don't break a dynamic column).

Option **D** will also load for the same reason: Timestamp is datetime, DeviceId is a proper GUID string, and StreamData is valid dynamic JSON. The fact that the JSON uses different property names (like *_index* and *_eventid*) doesn't prevent loading into a dynamic column—it's just stored as-is.

For background on update policies and how data is ingested into target tables, see https://learn.microsoft.com/en-us/azure/data-explorer/kusto/management/update-policy and for dynamic data type behavior see https://learn.microsoft.com/en-us/azure/data-explorer/kusto/query/scalar-data-types/dynamic

QUESTION NO: 10

You have a Fabric warehouse named DW1. DW1 contains a table that stores sales data and is used by multiple sales representatives.

You plan to implement row-level security (RLS).

You need to ensure that the sales representatives can see only their respective data. Which warehouse object do you require to implement RLS?

A. ISTORED PROCEDURE

B. CONSTRAINT

C. SCHEMA

D. FUNCTION

ANSWER: D

Explanation:

In a Fabric warehouse, row-level security is implemented the same general way it is in SQL Server: you create an inline table-valued **function** that returns which rows a user is allowed to see (usually by checking the current user identity and matching it to a column like SalesRepID). That function becomes the "filter" logic for the table.

After that, you attach the function to the target table using a security policy so the filter is automatically applied whenever someone queries the table. A stored procedure or constraint can't automatically filter every query for every user, and a schema is just a container—neither of those enforce per-user row filtering.

Reference: https://learn.microsoft.com/en-us/sql/relational-databases/security/row-level-security?view=sql-server-ver16