MB-820$^{Q\&As}$

Microsoft Dynamics 365 Business Central Developer

Pass Microsoft MB-820 Exam with 100% Guarantee

QUESTION 1

You are cleaning up sandbox environments for a company.

The company requires data to be cleared from the environments each time an extension is published.

You need to configure the launch.json file.

Which schemaUpdateMode property should you set?

A. ForceUpgrade

B. ForceSync

C. Synchronize

D. Recreate

Correct Answer: D

In the context of cleaning up sandbox environments for a company where data needs to be cleared each time an extension is published, the schemaUpdateMode property in the launch.json file should be set to Recreate (D). Setting this property to Recreate ensures that every time the extension is published, the existing tables and data are dropped, and then the tables are recreated based on the current extension\\'s schema. This mode is particularly useful in development and testing environments where you need a clean slate for testing each version of the extension without the remnants of previous data affecting the outcomes. It\\'s important to use this setting cautiously, as it results in the loss of all existing data in the tables defined by the extension, which is suitable for a sandbox environment but not for production environments.

QUESTION 2

A company has a Business Central online environment.

You are exporting a file from a client by using the DownloadFromStream method.

You need to create an InStream data type to send the data

Which solution should you use?

A. Use OeateInStream method from codeunit "Temp Blob".

B. Use OeateInStream method for BLOB field of "TempBlob" table.

C. Use CreateInStream method for File type variable.

Correct Answer: A

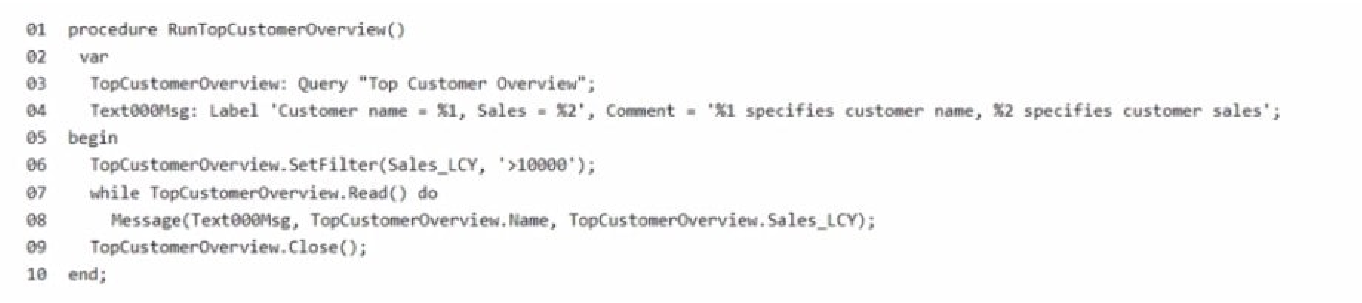
When exporting a file from a client using the DownloadFromStream method in a Business Central online environment, you need to create an InStream data type to send the data. The solution is to use the CreateInStream method from codeunit "Temp Blob" (A). The Temp Blob codeunit provides temporary storage of BLOBs (Binary Large Objects) and is commonly used for handling files and streams in Business Central. By using the CreateInStream method on a Temp Blob, you create an InStream that can then be used with the DownloadFromStream method to send the file data to the client. This approach is efficient for file handling and transfer in Business Central, especially in scenarios involving data export or file manipulation.

QUESTION 3

HOTSPOT

You create a query that contains a procedure to display the top customers.

The procedure breaks at runtime.



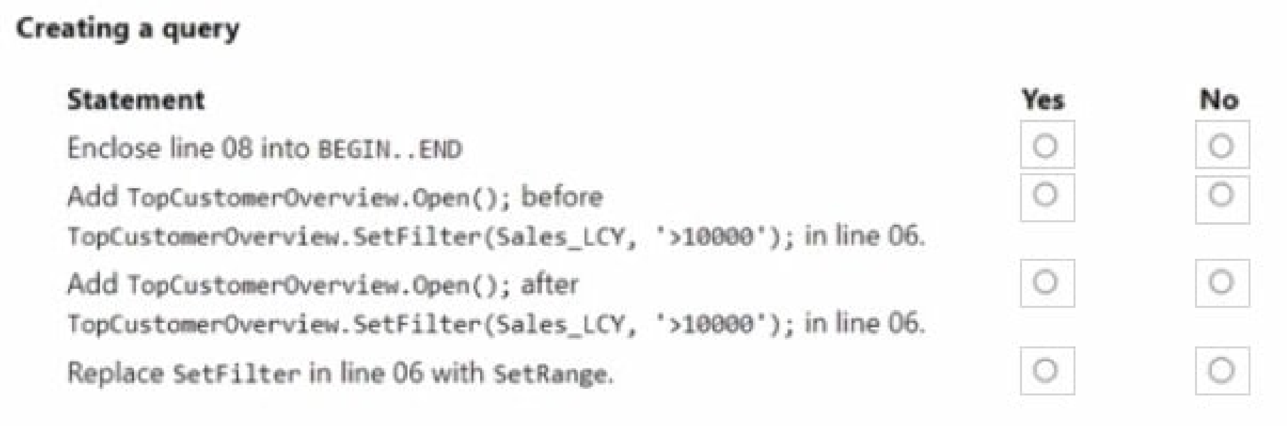
```
01  procedure RunTopCustomerOverview()
02   var
03     TopCustomerOverview: Query "Top Customer Overview";
04     Text000Msg: Label 'Customer name = %1, Sales = %2', Comment = '%1 specifies customer name, %2 specifies customer sales';
05  begin
06     TopCustomerOverview.SetFilter(Sales_LCY, '>10000');
07     while TopCustomerOverview.Read() do
08       Message(Text000Msg, TopCustomerOverview.Name, TopCustomerOverview.Sales_LCY);
09     TopCustomerOverview.Close();
10  end;
```

You need to fix the code.

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

NOTE: Each correct selection is worth one point.

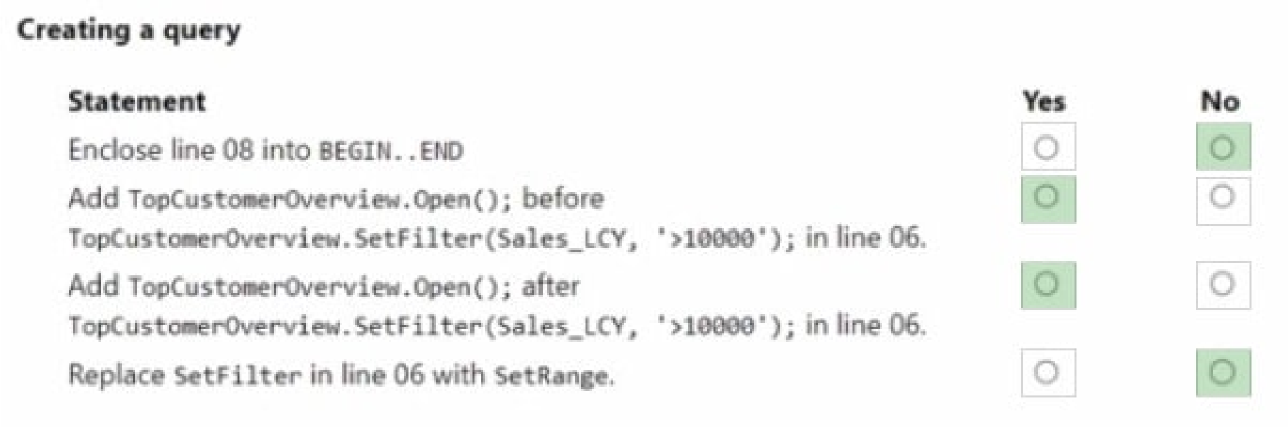
Hot Area:



Creating a query

Statement	Yes	No
Enclose line 08 into BEGIN..END	○	○
Add TopCustomerOverview.Open(); before TopCustomerOverview.SetFilter(Sales_LCY, '>10000'); in line 06.	○	○
Add TopCustomerOverview.Open(); after TopCustomerOverview.SetFilter(Sales_LCY, '>10000'); in line 06.	○	○
Replace SetFilter in line 06 with SetRange.	○	○

Correct Answer:



Creating a query

Statement	Yes	No
Enclose line 08 into BEGIN..END	○	[✓ ○]
Add TopCustomerOverview.Open(); before TopCustomerOverview.SetFilter(Sales_LCY, '>10000'); in line 06.	[✓ ○]	○
Add TopCustomerOverview.Open(); after TopCustomerOverview.SetFilter(Sales_LCY, '>10000'); in line 06.	[✓ ○]	○
Replace SetFilter in line 06 with SetRange.	○	[✓ ○]

Enclose line 08 into BEGIN .. END = NO Add TopCustomerOverview.Open(); before = YES TopCustomerOverview.SetFilter(Sales_LCY, \\'>10000\\'); in line 06. Add TopCustomerOverview.Open(); after TopCustomerOverview.SetFilter(Sales_LCY, \\'>10000\\'); in line 06. = YES Replace SetFilter in line 06 with SetRange. = NO

The code provided has a runtime error because the query TopCustomerOverview must be opened before it can be read from. Therefore, TopCustomerOverview.Open(); should be added before trying to read from the query, which is not present in the code. Enclosing line 08 into a BEGIN .. END block is unnecessary because it is a single statement, and AL does not require a BEGIN .. END block for single statements within trigger or procedure bodies. TopCustomerOverview.SetFilter(Sales_LCY, \\'>10000\\'); is a correct method to set a filter for the query, and using SetRange instead is not necessary unless the requirement is specifically to set a range of values, which is not indicated in the procedure\\'s description. In summary, for the procedure to run correctly, the query must be opened after setting the filter and before attempting to read from it. The SetFilter method is correct for the intended operation, and there\\'s no requirement to use SetRange or to enclose the Message call in a BEGIN .. END block.

QUESTION 4

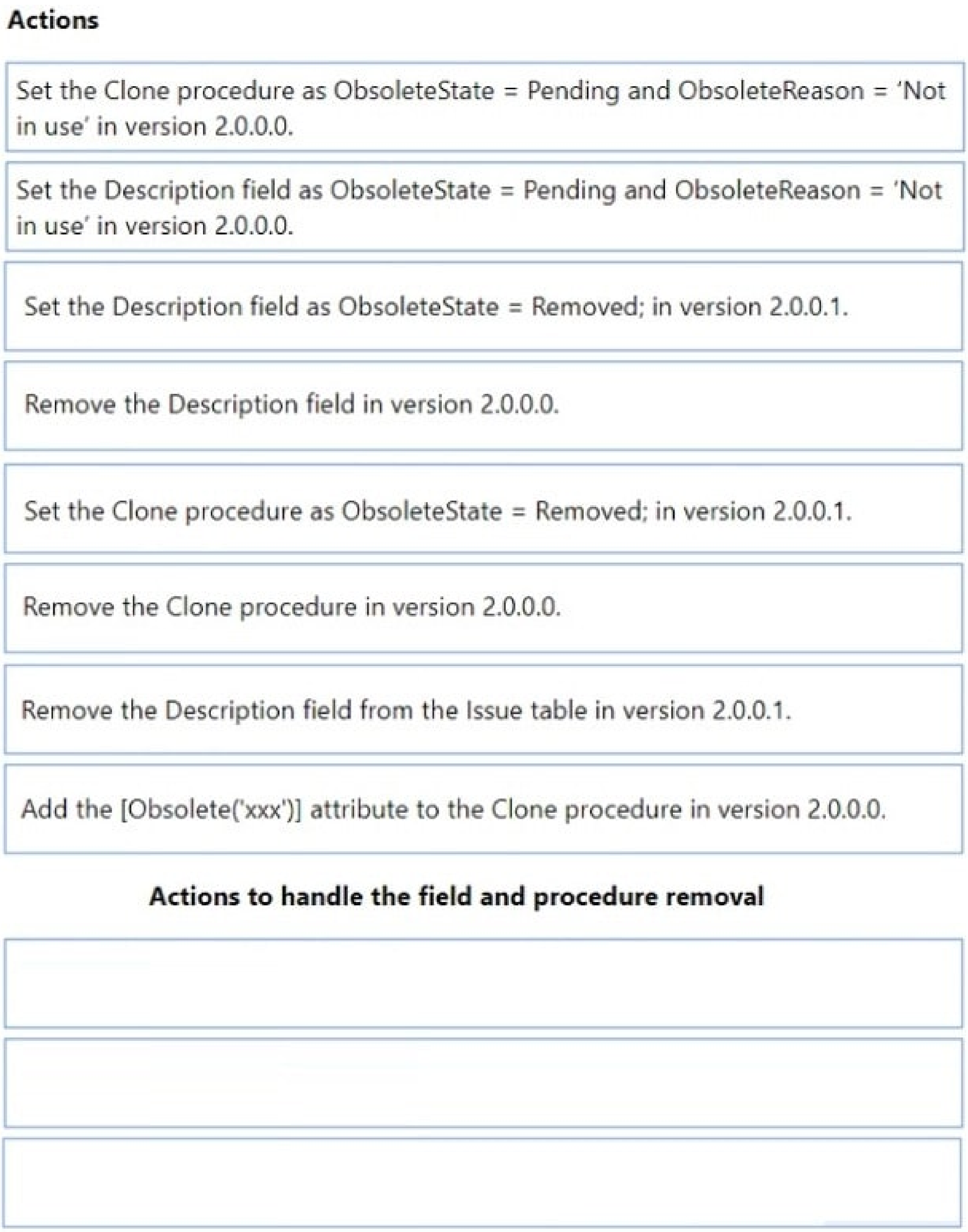
DRAG DROP

You need to handle the removal of the Description field and the Clone procedure without breaking other extensions.

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.

NOTE: More than one order of answer choices is correct. You will receive credit for any of the correct orders you select.

Select and Place:



Actions

Set the Clone procedure as ObsoleteState = Pending and ObsoleteReason = 'Not in use' in version 2.0.0.0.

Set the Description field as ObsoleteState = Pending and ObsoleteReason = 'Not in use' in version 2.0.0.0.

Set the Description field as ObsoleteState = Removed; in version 2.0.0.1.

Remove the Description field in version 2.0.0.0.

Set the Clone procedure as ObsoleteState = Removed; in version 2.0.0.1.

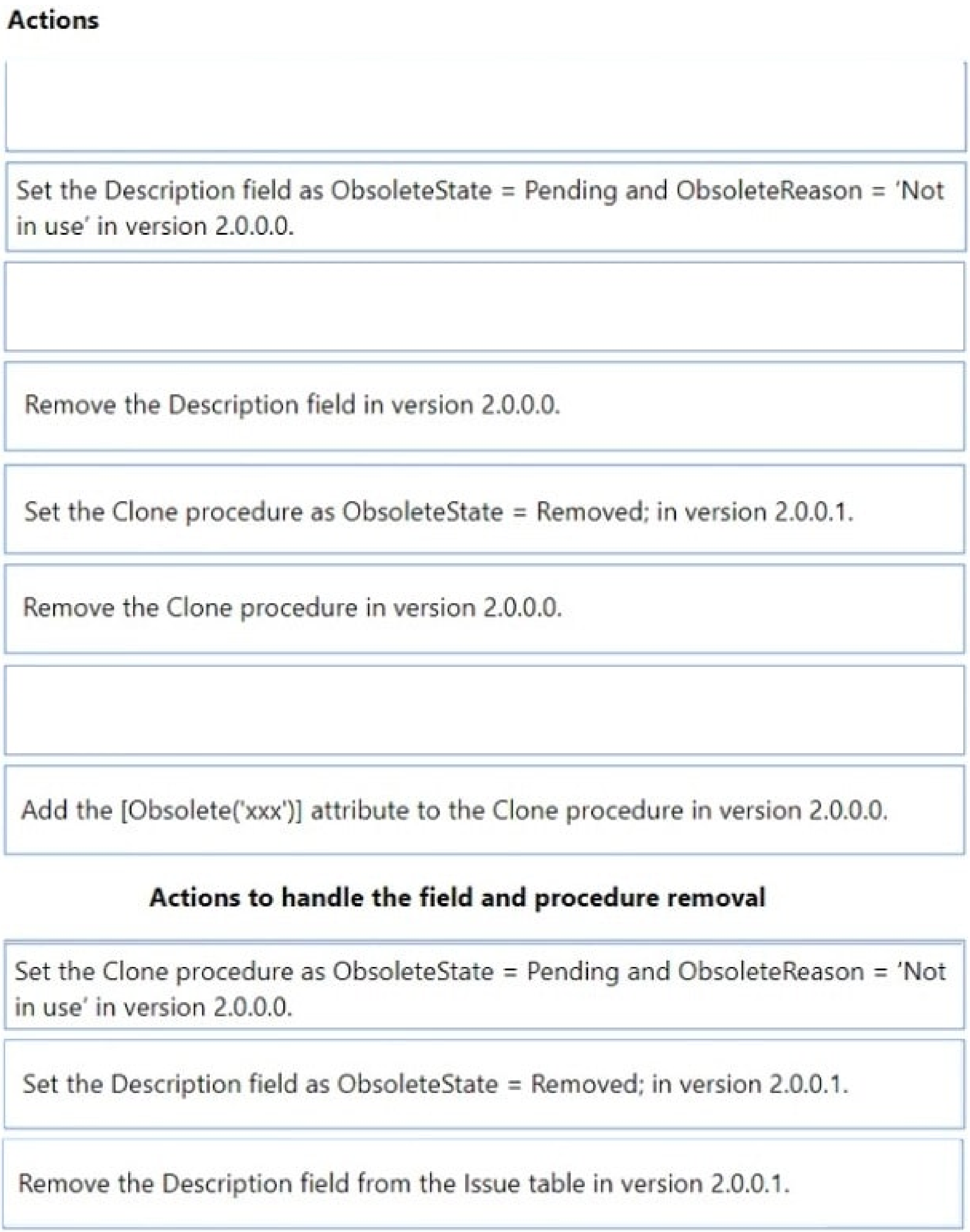
Remove the Clone procedure in version 2.0.0.0.

Remove the Description field from the Issue table in version 2.0.0.1.

Add the [Obsolete('xxx')] attribute to the Clone procedure in version 2.0.0.0.

Actions to handle the field and procedure removal

Correct Answer:



Actions

Set the Description field as ObsoleteState = Pending and ObsoleteReason = 'Not in use' in version 2.0.0.0.

Remove the Description field in version 2.0.0.0.

Set the Clone procedure as ObsoleteState = Removed; in version 2.0.0.1.

Remove the Clone procedure in version 2.0.0.0.

Add the [Obsolete('xxx')] attribute to the Clone procedure in version 2.0.0.0.

Actions to handle the field and procedure removal

Set the Clone procedure as ObsoleteState = Pending and ObsoleteReason = 'Not in use' in version 2.0.0.0.

Set the Description field as ObsoleteState = Removed; in version 2.0.0.1.

Remove the Description field from the Issue table in version 2.0.0.1.

Step 1: Set the Clone procedure as ObsoleteState = Pending and ObsoleteReason = \\'Not in use\\' in version 2.0.0.0
ObsoleteState Property Marks whether the object will be deprecated.

Property Value

*

No

Not obsolete. This is the normal/default setting.

*

Pending

Will become obsolete in a future version.

Syntax

ObsoleteState = Pending;

Remarks

By coding against this property, you can use this property as a way to communicate through code to other developers which objects and elements will become obsolete over time and those which are already obsolete, enabling them to adjust

their application code accordingly.

Note

For all elements, except for Tables and Table fields, setting ObsoleteState = Removed will throw Compiler Error AL0169 because after an appropriate warning state of Pending, these elements can be deleted.

Note

When developing using Dynamics NAV Development Environment (C/SIDE), you do not get warnings or errors when you compile objects that reference table objects, fields, or keys that are marked as Pending or Removed. ObsoleteState

property is only detected by the AL compiler, which will return warnings for references to elements marked as Pending and errors for references to elements marked as Removed.

Step 2: Set the Description field as ObsoleteState = Removed in version 2.0.01

Step 3: Remove the Description field from the issue table in version 2.0.0.1

Note: ObsoleteReason Property

Specifies why the object has been marked as Pending in the ObsoleteState property.

Syntax

ObsoleteReason = \\'Not Needed\\';

Remarks

Use this property to inform developers about an object or element that will become obsolete in time or is already obsolete. Use the ObsoleteTag Property to specify additional information which can be valuable to other developers.

Scenario:

The Issue Management process must be split into two extensions:

-

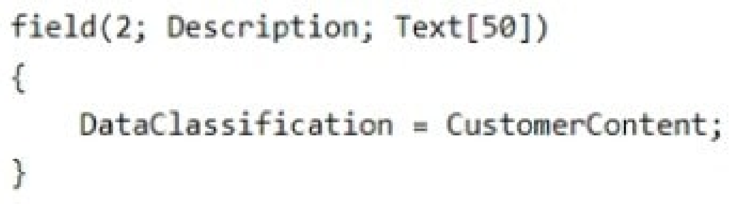
ISSUE BASE: main extension

-

ISSUE EXT: second extension with dependency from ISSUE BASE

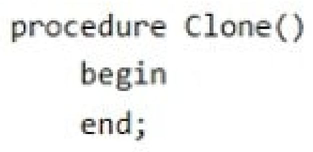
In the version 1.0.0.0 of the ISSUE BASE extension, you plan to create an Issue table that contains a global Decimal variable named IssueTotal.

In the version 1.0.0.0 of the ISSUE BASE extension, you plan to define a table named Issue Category with a Description field defined as follows:



```
field(2; Description; Text[50])
{
    DataClassification = CustomerContent;
}
```

The Issue table defined in ISSUE BASE extension contains a Clone procedure defined as follows:



```
procedure Clone()
    begin
    end;
```

In the ISSUE EXT extension, you create a tableextension object of the Issue table.

The tableextension object of the Issue table must access the IssueTotal: Decimal variable.

After weeks of usage, you discover that you must remove the Description field and the Clone procedure because they are no longer required.

Reference:

https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/properties/devenv-obsoletestate-property

https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/properties/devenv-obsoletereason-property

QUESTION 5

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question set 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 on the review screen.

A company creates a Business Central app and a table named MyTable to store records when sales orders are posted.

Users report the following issues:

1.

The users receive permission errors related to MyTable.

2.

Users are no longer able to post sales orders since installing the new app.

3.

The users cannot access the list page created in MyTable.

You need to resolve the user issues without creating new permission sets. You must use the principle of least privilege.

Solution: Decorate the event subscriber used for inserting data in MyTable by entering (lnherentPermissions(PermissionOb]ectType:TableData. Database:MyTable. \\'R\\')]

Does the solution meet the goal?

A. Yes

B. No

Correct Answer: A

Using InherentPermissions in an event subscriber with the specified syntax could potentially resolve the permission issues related to MyTable, provided that the permissions specified (in this case, \\'R\\' for Read) align with the minimum necessary for the users to perform their tasks. This approach allows the app to grant permissions dynamically based on the context of the event subscriber, which in this case is involved with inserting data into MyTable. By granting Read permission at the event level, it ensures that users have the necessary permissions to interact with MyTable in the context of the operations facilitated by the event subscriber, without needing to alter existing permission sets or grant broader permissions than necessary. This solution adheres to the principle of least privilege by ensuring that permissions are granted only within the narrow scope needed for specific operations, thereby potentially resolving the reported user issues in a secure and controlled manner.

QUESTION 6

You need to determine why the debugger does not start correctly.

What is the cause of the problem?

A. The "userId" parameter must have the GUID of the user specified, not the username.

B. The "breakOnNext" parameter is not set to "WebServiceClient".

C. The "userId" parameter is specified, and the next user session that is specified in the "breakOnNext" parameter is snapshot debugged.

D. The "executionContext" parameter is not set to "Debug".

Correct Answer: A

Initialize a snapshot debugging session

You can start a snapshot by creating a snapshot configuration file in Visual Studio Code.

Choose whether to run the session on a cloud service or locally. The configuration file contains the following information.

*

userId The GUID of the user who initiated the process to start snapshot debugging. For on-premises, this can also be the user name in user password authentication scenarios. The user must be able to start, or have a session type opened that is specified in the breakOnNext parameter.

*

Etc.

Scenario: Debugging problems A user of the ISSUE BASE extension in Business Central reports a problem.

You discover that the debugging is not triggering.

Incorrect:

Not B: Example (attach to a web client session)

The following example illustrates a configuration for a local server, where you want to debug a web client session.

...

```
{
 "name": "My attach to local server",
 "type": "al",
 "request": "attach",
 "server": "https://localhost",
 "serverInstance": "BC200",
 "authentication": "Windows",
 "breakOnError": true,
 "breakOnRecordWrite": false,
 "enableSqlInformationDebugger": true,
 "enableLongRunningSqlStatements": true,
 "longRunningSqlStatementsThreshold": 500,
 "numberOfSqlStatements": 10,
"breakOnNext": "WebClient"

} ...
```

Reference: https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-attach-debug-next https://learn.microsoft.com/en-us/dynamics365/business-central/dev-itpro/developer/devenv-snapshot-debugging

QUESTION 7

DRAG DROP

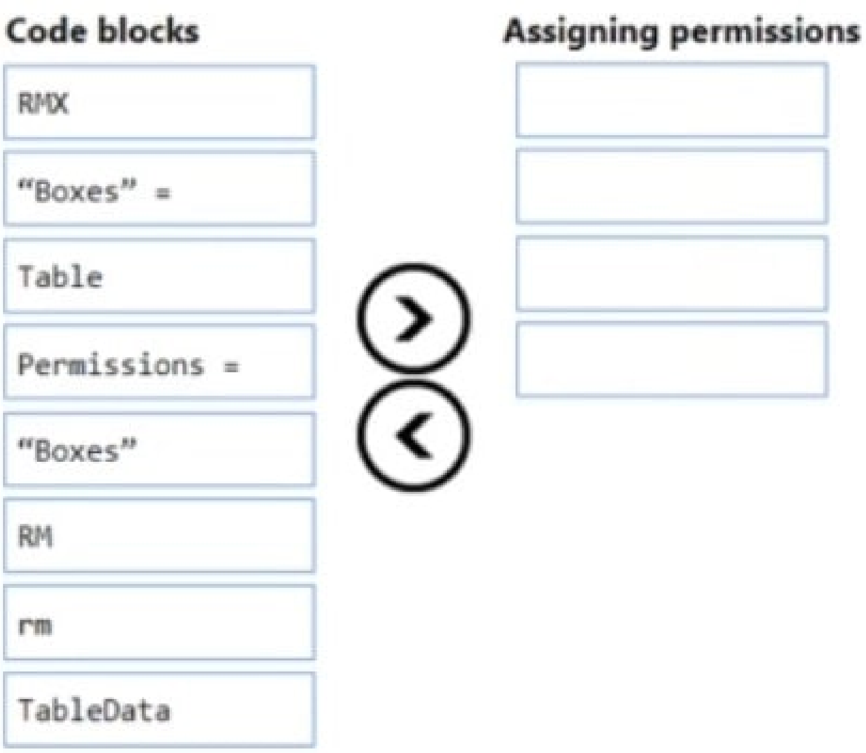
You create a codeunit that works with a table named Boxes. You plan to filter the records and then modify them.

You get an error that you do not have permission to work with the Boxes table.

You need to assign the Indirect permissions for the Boxes table to the codeunit.

Which four code blocks should you use in sequence to assign the correct permission? To answer, move the appropriate code blocks from the list of code blocks to the answer area and arrange them in the correct order.

Select and Place:

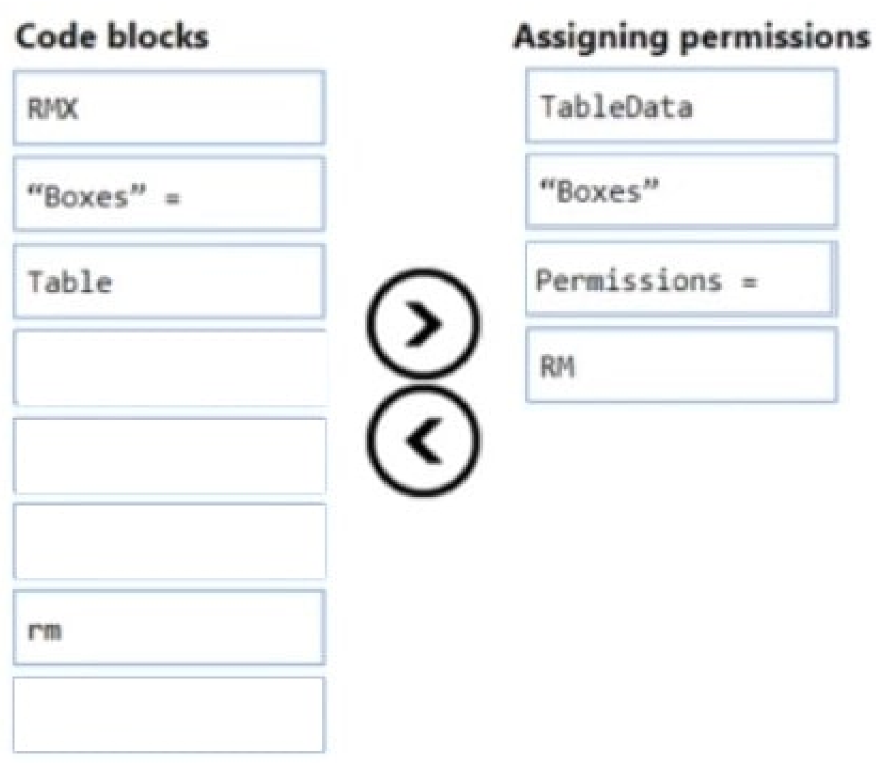


Code blocks

- RMX
- "Boxes" =
- Table
- Permissions =
- "Boxes"
- RM
- rm
- TableData

Assigning permissions

Correct Answer:



Code blocks

RMX

"Boxes" =

Table

rm

Assigning permissions

TableData

"Boxes"

Permissions =

RM

To assign the indirect permissions for the Boxes table to the codeunit, use the following code blocks in sequence: TableData "Boxes" = Permissions RIM Assigning permissions:In Business Central, to assign permissions within a codeunit, you need to specify the table that the permissions apply to, followed by the type of permission. The sequence starts by indicating that we are defining table data permissions (TableData). Then, we specify the table in question ("Boxes" =). After that, we state that we are setting permissions (Permissions). Finally, we assign the RIM permissions, which stands for Read, Insert, and Modify permissions. The Indirect permission allows the codeunit to read, insert, and modify records in the Boxes table indirectly, meaning these operations can be performed by the codeunit when it is called by a user who has direct permissions for these operations.

QUESTION 8

You are developing an app.

You plan to publish the app to Microsoft AppSource.

You need to assign an object range for the app.

Which object range should you use?

A. custom object within the range 50000 to 59999

B. custom object within the range 50000 to 99999

C. divided by countries and use specific a country within the range 100000 to 999999

D. an object range within the range of 7000000 to 74999999 that is requested from Microsoft

E. free object within the standard range 1 to 49999

Correct Answer: D

When developing an app for Microsoft AppSource, it is essential to use an object range that is specifically designated by Microsoft to avoid conflicts with other apps and the base application. The correct object range to use is: An object range within the range of 70000000 to 74999999 that is requested from Microsoft (D): This range is reserved for AppSource apps. Developers need to request this range from Microsoft to ensure that the objects used in their extension do not conflict with those used by other extensions or by the base application. Using this reserved range helps maintain the integrity and compatibility of extensions published on AppSource. It\\'s important to note that the other ranges mentioned (A, B, C, and E) are not suitable for apps intended for AppSource. Ranges 50000 to 59999 and 50000 to 99999 are typically reserved for per-tenant customizations or partner solutions, not for distribution on AppSource. The standard range 1 to 49999 is reserved for the base application objects, and using an object range divided by countries (C) is not a standard practice for AppSource apps.

QUESTION 9

A company uses Business Central.

The company plans to use a translation file in an extension. The extension has a caption that should not be translated.

You need to prevent the caption from being translated.

What should you do?

A. Use the CaptionML property and copy the same caption for each language used.

B. Set the GenerateLockedTranslations feature in the appjson file.

C. Add the Locked = true parameter to the Caption.

D. Delete the Caption property.

E. Copy the same caption for each language in the translation file.

Correct Answer: C

To prevent a caption from being translated in an extension for Microsoft Dynamics 365 Business Central, you should add the Locked = true parameter to the Caption (C). This parameter explicitly marks the caption as locked for translation, ensuring that it remains unchanged across different language versions of the extension. This approach is useful for specific terms, brand names, or other elements within the application that should remain consistent regardless of the user\\'s language settings. Unlike the other options, which involve manual manipulation of the translation file or properties, setting Locked = true directly in the AL code provides a clear, maintainable, and error-proof method to exclude specific captions from the translation process.

QUESTION 10

You plan to write unit test functions to test newly developed functionality in an app.

You must create a test codeunit to write the functions.

You need to select the property to use for the test codeunit.

Which property should you use to ensure that the requirements are fulfilled?

A. SubType

B. Access

C. Description

Correct Answer: A

When creating a test codeunit in Microsoft Dynamics 365 Business Central to write unit test functions, the SubType property (A) of the codeunit should be set to Test. This property is crucial for defining the codeunit\\'s purpose and behavior within the application. By setting the SubType property to Test, you are indicating that the codeunit contains test functions intended to validate the functionality of other parts of the application, such as customizations or new developments. This distinction ensures that the testing framework within Business Central recognizes the codeunit as a container for test functions, allowing it to execute these functions in a testing context, which can include setting up test data, running the tests, and cleaning up after the tests have completed.