Contents

1 INF99X: Sample Course

- **Download Latest Student Handbook and AllFiles Content**
- **Are you a MCT?** - Have a look at our GitHub User Guide for MCTs
- **Need to manually build the lab instructions?** - Instructions are available in the MicrosoftLearning/Docker-Build repository

1.1 What are we doing?

- To support this course, we will need to make frequent updates to the course content to keep it current with the Azure services used in the course. We are publishing the lab instructions and lab files on GitHub to allow for open contributions between the course authors and MCTs to keep the content current with changes in the Azure platform.
- We hope that this brings a sense of collaboration to the labs like we've never had before - when Azure changes and you find it first during a live delivery, go ahead and make an enhancement right in the lab source. Help your fellow MCTs.

1.2 How should I use these files relative to the released MOC files?

- The instructor handbook and PowerPoints are still going to be your primary source for teaching the course content.
- These files on GitHub are designed to be used in conjunction with the student handbook, but are in GitHub as a central repository so MCTs and course authors can have a shared source for the latest lab files.
- It will be recommended that for every delivery, trainers check GitHub for any changes that may have been made to support the latest Azure services, and get the latest files for their delivery.

1.3 What about changes to the student handbook?

- We will review the student handbook on a quarterly basis and update through the normal MOC release channels as needed.

1.4 How do I contribute?

- Any MCT can submit a pull request to the code or content in the GitHub repro, Microsoft and the course author will triage and include content and lab code changes as needed.
- You can submit bugs, changes, improvement and ideas. Find a new Azure feature before we have? Submit a new demo!

1.5 Notes

1.5.1 Classroom Materials

1.6 It is strongly recommended that MCTs and Partners access these materials and in turn, provide them separately to students. Pointing students directly to GitHub to access Lab steps as part of an ongoing class will require them to access yet another UI as part of the course, contributing to a confusing experience for the student. An explanation to the student regarding why they are receiving separate Lab instructions can highlight the nature of an always-changing cloud-based interface and platform. Microsoft Learning support for accessing files on GitHub and support for navigation of the GitHub site is limited to MCTs teaching this course only.

1.7 title: Online Hosted Instructions permalink: index.html layout: home

2 Content Directory

Hyperlinks to each of the lab exercises and demos are listed below.

2.1 Labs

{% assign labs = site.pages | where_exp:”page”, ”page.url contains ’/Instructions/Labs’” %} | Module | Lab | | --- | --- | {% for activity in labs %}| {{ activity.lab.module }} | [{{ activity.lab.title }}{% if activity.lab.type %} - {{ activity.lab.type }}{% endif %}](/home/ll/Azure_clone/Azure_new/MB-500-Microsoft-Dynamics-365-Finance-and-Operations-Apps-Developer/{{ site.github.url }}{{ activity.url }}) | {% endfor %}

2.2 Demos

2.3 {% assign demos = site.pages | where_exp:"page", "page.url contains '/Instructions/Demos'" %} | Module | Demo | | --- | --- | {% for activity in demos %}| {{ activity.demo.module }} | [{{ activity.demo.title }}](/home/ll/Azure_clone/Azure_new/MB-500-Microsoft-Dynamics-365-Finance-and-Operations-Apps-Developer/{{ site.github.url }}{{ activity.url }}) | {% endfor %}

2.4 lab: title: 'Exercise 01: Create a data entity and import data' module: 'Module 10: Data migration'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

3 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

4 Scenario

In this lab, Asset Insurance data will be imported from a csv file into Dynamics 365 Finance and Operations. You need to develop a data entity to import the asset insurance data.

5 Lab solution

5.1 Prerequisite

- To execute this exercise you need to have a Dynamics 365 Finance and Operations apps development environment with access to *Visual Studio*
- Import the following lab exercise as a prerequisite of this lab (for import instructions, please see appendix)
- MB500TrainingProject.axpp
- MB500TrainingProjectDataModel.axpp
- MB500TrainingProjectUI.axpp
- MB500TrainingProjectCode.axpp
- MB500TrainingProjectExtension.axpp

5.2 Task #1: Import Asset insurance data

1. Right click on the Training solution under Solution Explorer.
2. Click on Add > New project and create a new project **MB500TrainingProjectDataManagement**. Set the new project's **Model** property to **MB500Training**.
3. Right-click on the Project **MB500TrainingProjectDataManagement** and go to Add > New Item
4. Select Data Entity under Data Model to create a new data entity **MBTAssetInsuranceEntity**
5. A wizard will popup, where you enter the following and select **Next**.
    1. Primary datasource: MBTAssetInsurance
    2. Enable public API: Checked
    3. Public entity name: AssetInsurance
    4. Public collection name: AssetInsurances
    5. Enable data management capabilities: Checked

6. Staging table: MBTAssetInsuranceStaging
7. View privilege: MBTAssetInsuranceEntityView
8. Maintain privilege: MBTAssetInsuranceEntityMaintain

6. Make sure all the fields are checked, except DataAreaId.
7. Select **Finish**.
8. A new folder Data Entities will be created under project **MB500TrainingProjectDataManagement** in the Solution Explorer with the new data entity **MBTAssetInsuranceEntity** within it.
9. Another folder Tables will be created under project **MB500TrainingProjectDataManagement** in the Solution Explorer with the new table **MBTAssetInsuranceStaging** within it.
10. Another folder Security Privileges will be created under project **MB500TrainingProjectDataManagement** in the Solution Explorer with two new security privileges **MBTAssetInsuranceEntityView** and **MBTAssetInsuranceEntityMaintain** within it.
11. Save and build the project **MB500TrainingProjectDataManagement**

Output

1. Open the Dynamics 365 Finance and Operations apps portal from your browser
2. Open **Procurement and sourcing > Setup > Vendors > Vendor Groups**
3. Add a new record if not already there:
    1. Vendor Group: 100
    2. Description: Insurance Vendor
4. Open **Procurement and sourcing > Setup > Procurement and sourcing parameters**
5. In the **General** tab, you will find a field Insurance Vendor Group; select 100 in that field.
6. Open **Procurement and sourcing > Vendors > All vendors**
7. Add four new vendors (if not already there) with Vendor group 100.
    1. Vendor Code: INS0001, Name: Insurance vendor 001
    2. Vendor Code: INS0002, Name: Insurance vendor 002
    3. Vendor Code: INS0003, Name: Insurance vendor 003
    4. Vendor Code: INS0004, Name: Insurance vendor 004
8. Navigate to the menu **Organization Administration** and execute **Organization Administration > Periodic > Load Number Sequence**
9. At the end of the execution, you will get a message "Number sequence loaded"
10. Navigate to the module **Fixed Assets** and open **Fixed assets > Setup > Fixed assets parameters**
11. Go to the **Number sequences** tab and you will find a new reference Asset Insurance ID. In the Number Sequence code field select Fixe_17
12. Save the record
13. Open **Data Management** *workspace* and select **Framework parameters**
14. In the Entity settings tab, click on the **Refresh entity list** button
15. In the **Data Management** workspace, open **Data entities**
16. Search for the **MBTAssetInsuranceEntity** and select **Modify target mapping** in the action menu
17. Ensure the mapping between Staging and Target is correct
18. In the Data Management workspace, open Import
19. Enter the following data:
    1. Group name: Asset Insurance Import
    2. Data project operation type: Import

3. Select **+Add file**
4. Entity name: MBTAssetInsuranceEntity
5. Source data format: CSV
6. Default refresh type: Full push only
7. Select button **Upload and add** and select *assetInsurance.csv* from the resources – it may be provided to you or you can download from https://aka.ms/mb500labresources
8. Select the **Close** button

20. In the Selected entities line, select **View map**
21. Select the **Mapping details** tab and select the Auto-generated checkbox in the ASSETINSURANCEID line.
22. Click on Save and close this form
23. Click on the **Import** button in the action pane
24. Go back to the Data management workspace and monitor the status of the import process

Appendix

Project Import instructions:

- The lab resources may be found at https://aka.ms/mb500labresources
- To import a project, navigate to Visual Studio and select **Dynamics 365 > Import Project**
- In the File name field, select the file name with path of the .axpp file that you want to import
- For the first import, of MB500TrainingProject, select **New solution**. For subsequent ones, select the **Current solution** option.
- Select the **OK** button. You may safely override if prompted.

5.3 lab: title: 'Exercise 01: Development Environment Configuration' module: 'Module 01: Overview and architecture'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

6 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

7 Scenario

Create an Excel workbook template to integrate Vendor Group data using Microsoft Dynamics Office Add-in. Add a new Vendor group in the Excel workbook and sync the vendor group with Finance and Operations through OData Endpoint.

8 Lab solution

8.1 Task #1: Create template

1. You may need to run AdminUserProvisioning; follow the lab machine's instructions.
2. Navigate to the browser interface. Sign in with the credentials provided to you.
    1. If you use Edge, it might give a warning; you can use **Advanced** and **Continue** to safely bypass.

2. If you get an error that a service is not running, restart the browser first, then the machine if necessary.

3. Navigate to **Modules** > **Common** > **Common** > **Office Integration** > **Excel Workbook Designer**. Note that most navigation is via **Modules**, so this is not typically specified.
4. **Search** for **VendorGroup** in the filter.
5. From the list of available fields **select** the fields **Vendor group**, **Description**, and **Terms of payment** and move them to the selected field box by selecting the right Arrow.
6. Select the **Create workbook** button on the action pane.
7. Select the **Download** button in the **Save To panel** on the right side.
8. Download the file by selecting **Save As** and store it in the **Downloads** folder.
9. Navigate to **Common** > **Office Integration** > **Document templates.**
10. Select **New**.
11. On the right panel in the **Upload template** section select the **browse** button and select the file downloaded previously (if you used the default name, it's DynamicsWorkbook).
12. Enter **CustomVendorGroup** in the **Template name** field.
13. Select **OK**.
14. Click **Save**.

8.2 Task #2: Open in Excel

1. Navigate to **Procurement and sourcing** > **Setup** > **Vendors** > **Vendor groups**.
2. If you select **Open in Microsoft Office** > **Open in Excel**, you will find the new template, **CustomVendorGroup** that you had uploaded.
3. **Select CustomVendorGroup** and download the Excel template.
4. **Save** and then **Open** the downloaded **Excel, Allow** it if needed**, Close** activation, and Select **Enable editing**. Trust this add-in. Sign in (using your same credentials, if asked).
5. All the existing data of the Vendor group table will appear in the Excel spreadsheet.
6. Enter a new record.
7. **Enter 100** in the field **Vendor group**, **Insurance Vendor** in the **Description** field and **Net10** in the **Terms of payment** field.
8. Select the **Publish** button in the **Microsoft Dynamics Office Add-in app**.
9. **Open** the **Vendor group form** to check the new record is added.

8.3 lab: title: 'Exercise 01: Requirements discussion' module: 'Module 02: Solution design'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

9 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

10 Scenario

In this exercise, you will be given some requirements. You need to provide a solution design using Dynamics 365 Finance and Operations. This is a discussion and not hands-on.

11 List of requirements

11.1 Functional requirements

1. There is a requirement to capture Insurance information for the fixed assets.
2. One single fixed asset can have multiple insurances as different components of the fixed asset can be insured by different insurance companies.
3. Insurance companies will be considered as vendor in Dynamics 365 Finance and Operations
4. All financial postings for the insurance vendors will be booked in the same ledger account.
5. There should be multiple periods for paying the premium; like – monthly, quarterly, half-yearly and yearly.
6. Based on the defined premium amount, premium records will be generated indicating due dates for payment.
7. Other information to be captured related to the insurances are Policy Number, Policy date, Insured value, Policy expiry date (one year from the policy date), insurance agent and policy description.
8. An auto-generated transaction number should be allocated for each insurance for the internal tracking of the company.

11.2 Non-functional requirements

1. The company has a branch in a remote location, who does not have access to the asset insurance feature. They will send a regular CSV file with Insurance data, which should be imported in Dynamics 365 Finance and Operations application.
2. The accountant of the company should execute the process to generate the premium once the insurance data is entered or imported. At times, the browser rendering the Dynamics 365 Finance and Operations application hangs, if the volume of insurance data to be processed is significantly high. This may hamper the work of the accountant.

11.3 Integration requirements

1. If a vendor is identified to provide insurance for a fixed asset, a mail should automatically go to the vendor once record is inserted into the system.

11.4 Reporting requirements

1. Dynamics 365 Finance and Operations has a Workspace for Fixed Asset Management. A tile should be added in that workspace which will show a BI report displaying total premium paid by the company based on several criteria like – dates, insurance vendor, fixed asset location etc.

11.5 lab: title: 'Exercise 01: Configuring Visual Studio and new Package/Model/Project creation' module: 'Module 03: Developer tools'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

12 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

13 Scenario

1. In this lab, you will configure the Visual Studio 2015 for Dynamics 365 Finance and Operations apps development work.
2. In this lab, you will create a new Package and Model. Then you will create a Visual Studio Project, where you will develop an executable class.

14 Prerequisite

To execute this exercise, you need to have a Dynamics 365 Finance and Operations apps development environment with access to Visual Studio.

15 Lab solution

15.1 Task #1: Setup Visual Studio 2015

1. **Right click** on the **Visual Studio 2015** icon on the desktop and **Run as administrator**.
2. Select **Dynamics 365 > Options**.
3. Select **Dynamics 365 > Projects** and make sure these checkboxes are selected:
    1. **Organize projects by element type**
    2. **Synchronize database on build for newly created project**
4. Select **Dynamics 365 > Best Practices** and check the following, all beginning with "Microsoft.Dynamics.AX.Framework.":
    1. **CustomizationAnalysisRules**
    2. **DataAccessRules**
    3. **DataMethodRules**
    4. **DeprecatedElementsRules**
    5. **Services.OData.Rules**
5. Navigate up to **Text Editor > All Languages > General**.
6. Select the **Line numbers** check box (twice, if necessary, until a checkmark appears).
7. Select **OK**.

15.2 Task #2: Create new Package and Model

1. Navigate to **Dynamics 365 > Model Management > Create model**.
2. Enter **MB500Training** in the **Model name** field, **Microsoft** in the **Model publisher field**, and **MB-500 Training** in the **Model display name** field, and select **Next**.
3. Select **Create new package**, and select **Next**.
4. Check the **following** packages, then select **Next**.
    1. ApplicationPlatform (default)
    2. ApplicationSuite
    3. ApplicationFoundation
    4. ApplicationWorkspaces
    5. ContactPerson
    6. Directory

7. PersonnelCore

5. The **Create new project** and **Make this my default model for new projects** checkboxes should be **checked**, and select **Finish**.
6. In a moment, a new project dialogue box will appear.
7. Select **Finance Operations**.
8. Enter **MB500TrainingProject** in the **Name** field.
9. Enter **MB500TrainingSolution** in the **Solution name** field, and select **OK**.
10. A new solution and project will be created under Solution Explorer.
11. **Right click** on the project, which is the node below the solution.
12. Go to **Add > New Item**.
13. Under **Code**, select **Runnable Class (Job).**
14. Enter **MBTHelloWorld** in the **Name** field, and select **Add**.
15. A new folder Classes will be created under MB500TrainingProject and a new Class MBTHelloWorld is created under the Classes folder.
16. Open **MBTHelloWorld Class** in the editor if not already open.
17. Within the **main method** (as shown) add the info line in the below:

```
public static void main(Args _args)
{
    info("Hello World");
}
```

18. Select the **Save** icon.
19. **Right click** on the **MB500TrainingProject**.
20. Select **Properties**.
21. Select the value **MBTHelloWorld** in the **Startup Object** field.
22. Enter **USMF** in the **Company** field.
23. Check that the value in the **Synchronize Database on Build** field is **True**.
24. Set the new project's **Model** property to **MB500Training**.
25. Select **Apply** and **OK**.
26. **Right click** on the **MB500TrainingProject** and select **Build**. Check the output in the pane.
27. Select the **Start** button on the Action pane.
28. The Finance and Operations apps instance will open in your browser with an Infolog of **Hello World** after a moment.
29. Close the browser.

15.3 lab: title: 'Exercise 01: Create tables' module: 'Module 04: AOT Elements'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

16 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

17 Scenario

1. In this lab you will create new tables AssetInsurance and InsurancePremium to capture the insurance record for fixed assets. You will create all required indexes and relations in those tables.
2. You will also add a new field in the standard table PurchParameters.

18 Prerequisite

- To execute this exercise you need to have a Dynamics 365 Finance and Operations apps development environment with access to Visual Studio
- Import the following lab exercise as a prerequisite of this lab (for import instructions, please see appendix)
- MB500TrainingProject.axpp

19 Lab solution

19.1 Task #1: Create a new Base Enumeration

1. Right click on the Visual Studio solution under Solution Explorer.
2. Click on Add -> New project and create a new project MB500TrainingProjectDataModel
3. Right click on the project **MB500TrainingProject**DataModel.
4. Select **Add > New Item**.
5. Select **Data Types >Base Enum**.
6. In the **Name** field, enter **MBTPeriod**.
7. Select **Add**.
8. A new folder will be created in Solution Explorer called Base Enums with a new Enum MBTPeriod within it.
9. Open **MBTPeriod** in the editor if not already opened, by double clicking it.
10. **Right click** on the **MBTPeriod** in the editor and select **Properties**.
11. The Properties window typically appears in the lower right. In the **Label** enter **Premium payment**. You can drag the window bigger.
12. Right click on the **MBTPeriod** in the editor.
13. Select **New Element**.
14. A new element will be created called BaseEnumValue1.
15. Right click on **BaseEnumValue1**.
16. In the **Name** property value enter **None**.
17. In the **Label** enter **Not Applicable**.
18. In the same way add the following four new Enum elements:

Name	Label
Monthly	**Monthly**
Quarterly	**Quarterly**
HalfYearly	**Half Yearly**
Annually	**Annually**

19.2 Task #2: Create a new EDT String

1. Right click on the project **MB500TrainingProjectDataModel**.
2. Select **Add > New Item**.

3. Select **Data Types > EDT String**.
4. In the **Name** field, enter **MBTAssetInsuranceID**.
5. Select **Add.**
6. A new folder will be created in Solution Explorer called EDT Strings with a new EDTString, MBTAssetInsuranceID within it.
7. Open **MBTAssetInsuranceID** in the editor if not already opened.
8. Right click on the **MBTAssetInsuranceID** in the editor.
9. Select **Properties**.
10. In the **Label** enter **Asset insurance ID**.

19.3 Task #3: Create a Table MBTAssetInsurance and add fields

1. Right click on the project **MB500TrainingProjectDataModel**.
2. Click on **Add > New Item**.
3. Select **Data Model > Table**.
4. In the **Name** field, enter **MBTAssetInsurance**.
5. Select **Add**.
6. A new folder will be created in Solution Explorer called Tables with a new table, MBTAssetInsurance within it.
7. Open the table **MBTAssetInsurance** in the editor if not already opened.
8. In the Action pane, select **View > Application Explorer**.
9. In the Application Explorer, **search** for **AssetID** within **AOT > Data Types > Extended Data Types**.
10. Drag **AssetID** and drop it on the **Fields** node of the table **MBTAssetInsurance** in the editor.
11. Drag **MBTAssetInsuranceID** from the Solution Explorer and drop it on the **Fields** node of the table **MBTAssetInsurance** in the editor.
12. Go to the **Name** property of the field **MBTAssetInsuranceID** and **remove** the **MBT prefix** from the name (this is best practice).
13. In the Application Explorer, **search** for **VendAccount** within **AOT > Data Types > Extended Data Types**.
14. Drag **VendAccount** and **drop** it on the **Fields** node of the table **MBTAssetInsurance** in the editor.
15. In the Application Explorer, **search** for **TransDate** within **AOT > Data Types > Extended Data Types**.
16. **Drag TransDate** and **drop** it on the **Fields** node of the table **MBTAssetInsurance** in the editor.
17. Go to the **Name** property of the field **TransDate** and enter **InsurancePolicyDate**.
18. In the **Label** property enter **Insurance Policy Date**.
19. In the Application Explorer, **search** for **AssetPolicyExpiration** within **AOT > Data Types > Extended Data Types**.
20. **Drag AssetPolicyExpiration** and **drop** it on the **Fields** node of the table **MBTAssetInsurance** in the editor.
21. Go to the **Name** property of the field **AssetPolicyExpiration** and enter **PolicyExpiration**.
22. In the **Label** property enter **Policy Expiry Date**.
23. In the Application Explorer, **search** for **AssetInsurancePolicyNum** within **AOT > Data Types > Extended Data Types**.
24. **Drag AssetInsurancePolicyNum** and **drop** it on the **Fields** node of the table **MBTAssetInsurance** in the editor.

25. Go to the **Name** property of the field **AssetInsurancePolicyNum** and enter **InsurancePolicyNum**.
26. In the **Label** property enter **Policy number**.
27. In the Application Explorer, **search** for **AssetInsuredValue** within **AOT > Data Types > Extended Data Types**.
28. **Drag AssetInsuredValue** and **drop** it on the **Fields** node of the table **MBTAssetInsurance** in the editor.
29. Go to the **Name** property of the field **AssetInsuredValue** and enter **InsuredValue**.
30. In the **Label** property enter **Insured value**.
31. In the Application Explorer, **search** for **AssetPolicyAmount** within **AOT > Data Types > Extended Data Types**.
32. **Drag AssetPolicyAmount** and **drop** it on the **Fields** node of the table **MBTAssetInsurance** in the editor.
33. Go to the **Name** property of the field **AssetPolicyAmount** and enter **PremiumAmount**.
34. In the **Label** property enter **Premium amount**.
35. From the Solution Explorer **drag MBTPeriod** and **drop** it on the **Fields** node of the table **MBTAssetInsurance**.
36. Go to the **Name** property of the field **MBTPeriod** and **remove** the **MBT prefix** from the name.
37. In the Application Explorer, **search** for **AssetInsuranceAgent** within **AOT > Data Types > Extended Data Types**.
38. **Drag AssetInsuranceAgent** and **drop** it on the Fields node of the table **MBTAssetInsurance** in the editor.
39. Go to the **Name** property of the field **AssetInsuranceAgent** and enter **InsuranceAgent**.
40. In the **Label** property enter **Insurance Agent**.
41. In the Application Explorer, **search** for **Description** within **AOT > Data Types > Extended Data Types**.
42. **Drag Description** and **drop** it on the **Fields** node of the table **MBTAssetInsurance** in the editor.
43. Go to the **Name** property of the field **Description** and enter **PolicyDescription**.
44. In the Label property enter **Policy description**.
45. Select the **Save All** icon and verify that the table compiles.

19.4 Task #4: Create New Field groups for MBTAssetInsurance Table

1. Right click on the **Field groups** node of the **MBTAssetInsurance** Table and add a **New Group**.
2. Enter **Identification** as the **Name** and **Label** of the **Group**.
3. Drag the following fields from the **Fields** node to the **Identification Field group**.
    1. **AssetInsuranceId**
    2. **VendAccount**
    3. **InsuranceAgent**
4. Right click on the **Field groups** node of the **MBTAssetInsurance** Table and **add** a **New Group**.
5. Enter **Description** as the **Name** and **Label** of the Group.
6. Drag the field **PolicyDescription** from the **Fields** node to the **Description Field group**.
7. Right click on the **Field** groups node of the **MBTAssetInsurance** Table and add a **New Group**.
8. Enter **Policy** as the **Name** and **Label** of the **Group**.
9. Drag the following fields from the **Fields** node to the **Policy Field group**.
    1. **InsurancePolicyDate**

2. **PolicyExpiration**
3. **InsurancePolicyNum**
4. **InsuredValue**
5. **PremiumAmount**
6. **Period**

10. Save all.

19.5 Task #5: Create New Indexes for MBTAssetInsurance Table

1. **Right click** on the **Indexes** node of the **MBTAssetInsurance** Table and select **New Index**.
2. In the **Name** field of the **Index** property pane enter **AssetInsuranceIdIdx**.
3. Set **Alternate Key** to **Yes**.
4. **Drag** the **AssetInsuranceId** field from the **Fields** node and **drop** it under the index **AssetInsuranceIdIdx**.

19.6 Task #6: Create New Relations for MBTAssetInsurance Table

1. Under the **Relations** node, create a **New Relation** named **AssetId**.
2. Change the **property** values of the **AssetId** Relation as follows:

Property Name	Value
Related Table	**AssetTable**
Cardinality	**ZeroMore**
Related Table Cardinality	**ZeroOne**
Relationship Type	**Association**
On Delete	**Restricted**

3. Right click on the **AssetId** relation and select **New > Normal**.
4. In the property window, select **Field** value as **AssetId** and **Related Field** value as **AssetId**.
5. Under the **Relations** node, create a **New Relation** named **InsuranceVendor**.
6. Change the property values of the **InsuranceVendor** Relation as follows:

Property Name	Value
Related Table	**VendTable**
Cardinality	**ZeroMore**
Related Table Cardinality	**ZeroOne**
Relationship Type	**Association**
On Delete	**Restricted**

7. **Right click** on the **InsuranceVendor** relation and select **New > Normal**.
8. In the properties, select **Field** value as **VendAccount** and **Related Field** value as **AccountNum**.
9. Save all.

19.7 Task #7: Create Table MBTInsurancePremium and Add Fields

1. **Right click** on the project **MB500TrainingProjectDataModel**.
2. Select **Add > New Item**.
3. Select **Data Model > Table**.
4. In the **Name** field, enter **MBTInsurancePremium**.
5. Select **Add**.

6. The existing folder Tables will have a new table MBTInsurancePremium within it.
7. Open the table **MBTInsurancePremium** in the editor if not already opened.
8. Drag **MBTAssetInsuranceID** from the Solution Explorer and drop it on the **Fields** node of the table **MBTAssetInsurance** in the editor.
9. Go to the **Name** property of the field **MBTAssetInsuranceID** and remove the **MBT prefix** from the name.
10. In the Action pane, select **View > Application Explorer** if not already open.
11. In the Application Explorer, search for **Counter** within **AOT > Data Types > Extended Data Types**.
12. Drag **Counter** and drop it on the **Fields** node of the table **MBTInsurancePremium** in the editor.
13. In the **Name** property of the field **Description** enter **LineNum**.
14. In the **Label** property enter **Line number**.
15. In the Application Explorer, search for **AssetPolicyAmount** within **AOT > Data Types > Extended Data Types**.
16. Drag **AssetPolicyAmount** and drop it on the **Fields** node of the table **MBTInsurancePremium** in the editor.
17. In the **Name** property of the field **Description** enter **PremiumAmount**.
18. In the **Label** property enter **Premium Amount**.
19. In the Application Explorer, **search** for **TransDate** within **AOT > Data Types > Extended Data Types** and drag it to the **Fields** node of the table **MBTInsurancePremium** in the editor.
20. In the **Name** property of the field **Description** enter **PremiumDueDate**.
21. In the **Label** property enter **Premium due date**.
22. In the Application Explorer, search for **TransDate** within **AOT > Data Types > Extended Data Types**, and drag it to the **Fields** node of the table **MBTInsurancePremium** in the editor.
23. Go to the **Name** property of the field Description and enter **PaymentDate**.
24. In the **Label** property enter **Payment Date**.

19.8 Task #8: Create New Field groups for MBTInsurancePremium Table

1. **Right click** on the **Field groups** node of the **MBTInsurancePremium** Table and add a **New Group**.
2. Enter **InsurancePremium** as the **Name** and **Label** of the **Group**.
3. **Drag** the following **fields** from the **Fields** node to the **InsurancePremium** Field group.
    1. **AssetInsuranceId**
    2. **LineNum**
    3. **PremiumAmount**
    4. **PremiumDueDate**
    5. **PaymentDate**

19.9 Task #9: Create a new Index for the MBTInsurancePremium Table

1. Right click on the **Indexes** node of the **MBTInsurancePremium** Table and add a **New Index**.
2. In the **Name** field of the **Index** property pane enter **InsurancePremiumIdx**.
3. Drag the **AssetInsuranceId** field from the **Fields** node and drop it under the index **InsurancePremiumIdx**.

19.10 Task #10: Create a New Relation for the MBTInsurancePremium Table

1. **Drag** the **LineNum** field from the **Fields** node and drop it under the index **InsurancePremiumIdx**.
2. Under the **Relations** node, **create** a new **Relation** named **AssetInsurance**.
3. **Change** the **property** values of the **AssetInsurance** Relation as follows:

Property Name
Related Table
Cardinality
Related Table Cardinality
Relationship Type
On Delete
Right click the relation, and select a new Normal one. **MBTInsurancePremium.AssetInsuranceId == MBTAssetIn**

19.11 Task #11: Add a new field in the PurchParameters Table

1. Search for the **PurchParameters** table from the Application Explorer Data Model Tables, right click on it, and select **Create extension**
2. A new folder **Table Extensions** will be created in the project **MB500TrainingProjectDataModel** under the Solution Explorer with the PurchParameters extension table in it
3. Open the **PurchParameters** extension table in the designer by double clicking it
4. Add a new string field in the extension table named **MBTInsuranceVendGroup**
5. Enter **VendGroupId** in the Extended Data Type and **Insurance Vendor Group** in the Label property of the MBTInsuranceVendGroup field
6. Open the field group **DefaultValues** and add MBTInsuranceVendGroup in that field group
7. Open the relations node and add a new relation InsuranceVendorGroup

Property Name
Related Table
Cardinality
Related Table Cardinality
Relationship Type
On Delete
Right click the relation, and select a new Normal one. PurchParameters.MB500Training. MBTInsuranceVendGroup == Ve

8. Save all and verify that the project builds.

Appendix

Project Import instructions:

- The lab resources may be found at https://aka.ms/mb500labresources
- To import a project, navigate to Visual Studio and select **Dynamics 365** > **Import Project**
- In the File name field, select the file name with path of the .axpp file that you want to import
- For the first import, of MB500TrainingProject, select **New solution**. For subsequent ones, select the **Current solution** option.
- Select the **OK** button. You may safely override if prompted.

19.12 lab: title: 'Exercise 01: User interface development' module: 'Module 04: AOT Elements'

20 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

User Interface Development

21 Scenario

A new Form is needed to capture the Asset Insurance information and premium details. Add a new button in the Asset Table from where the Asset Insurance form can be opened. Make the insurance tab of the Asset table form un-editable

22 Prerequisite

- To execute this exercise you need to have a Dynamics 365 Finance and Operations apps development environment with access to Visual Studio
- Import the following lab exercises as a prerequisite of this lab (for import instructions, please see appendix)
  - MB500TrainingProject.axpp
  - MB500TrainingProjectDataModel.axpp

23 Lab solution

23.1 Task #1: Add a new form MBTAssetInsuranceDetails

1. Right click on the Visual Studio solution under Solution Explorer.
2. Click on Add -> New project and create a new project **MB500TrainingProjectUI**. Set the new project's **Model** property to **MB500Training**.
3. In your project, add a New Item in User Interface of type Form, and Name it MBTAssetInsuranceDetails
4. **Add** these tables in the **Data Sources** node of the form:
   1. **MBTAssetInsurance.**
   2. **MBTInsurancePremium**
5. **Update** this property of the **MBTAssetInsurance** datasource
   1. **Index: AssetInsuranceIdIdx**
6. **Update** these properties of the **MBTInsurancePremium** datasource
   1. **Index: InsurancePremiumIdx**
   2. **Join Source: MBTAssetInsurance**
7. Right click on the **Design | Pattern** node and select **Apply Pattern > Simple List and Details > List Grid.**
8. Right click on the **Design | Pattern** node and select **New > Action Pane.**
9. In the Action pane property rename the **Name** property to **AssetInsuranceActionPane.**
10. Go to **Design | Pattern** node and select **New > Group.**
11. Go to **Design | Pattern** node and select **New > Group** a second time.
12. Go to **Design | Pattern** node and select **New > Tab.**
13. In the first Group control, there will not be any suffix, like "| Pattern: <unspecified>".
14. Rename this group to **NavigationGroup.**
15. In the second Group control, there will be a suffix, like "| Pattern: <unspecified>".
16. Rename this group to **DetailsGroup.**

17. Rename the tab to **DetailsTab.**
18. Right click on the group **NavigationGroup** and select **New > QuickFilter.**
19. Rename the **QuickFilter** control to **QuickFilterControl.**
20. In the **Properties** pane of the **QuickFilterControl** enter the following properties**:**

Property	Value
Target Control	Grid
Default Column	MBTAssetInsurance_AssetInsuranceId

21. Right click on the group **NavigationGroup** and select **New > Grid.**
22. Rename the **Grid control** to **Grid.**
23. In the **Properties** pane of the **Grid** control under **NavigationGroup** enter the following **properties:**

Property	Value
Data Source	MBTAssetInsurance

24. Drag these **fields** from the **MBTAssetInsurance** data source and drop on the **Grid** control under **NavigationGroup**
    1. **AssetInsuranceId**
    2. **InsuredValue**
25. Right click on the group **DetailsGroup** and select **Apply Pattern > Fields and Field Groups.**
26. Drag the **Identification** field group from the MBTAssetInsurance datasource and drop it on the group **DetailsGroup**
27. In the **Properties** pane of the **Identification** group control under **DetailsGroup** enter the following **properties:**

Property	Value
Style	Card

28. **Right click** on the **DetailsTab** and **add** new **Tabpages** with these names:
    1. **General**
    2. **Notes**
    3. **Premium**
29. For the **General** tabpage apply the pattern **Fields** and **Field Groups.**
30. In the **Properties** pane of the **General TabPage** control under **DetailsTab** enter the following **properties:**

Property	Value
Caption	Policy

31. Drag the **Policy** Field group from **Data source > MBTAssetInsurance > Field groups** and **drop** under the **General** tabpage**.**
32. In the **Properties** pane of the **Policy** Group control under **DetailsTab > General tabpage** enter the following **properties:**

Property	Value
Frame Type	None

33. For the **Notes** Tab Page apply the pattern **Fields** and **Field Groups.**

34. Drag the **Description** Field group from **Data source > MBTAssetInsurance > Field groups** and drop under the **Notes** Tab Page**.**

35. In the **Properties** pane of the **Notes tabPage** control under **DetailsTab** enter the following **properties:**

Property	Value
Caption	Description

36. In the **Properties** pane of the **Description** Group control under **DetailsTab > Notes tabpage** enter the following **properties:**

Property	Value
Frame Type	None

37. For the **Premium tabpage** apply the pattern **Toolbar** and **List.**

38. Right click on the **Premium** Tab Page and add a new **Action pane**, named **PremiumActionPane**.

39. Add a new **Button Group** to the action pane, named **AddDeleteButtonGroup** under **PremiumActionPane** with the following **properties**:

Property	Value
Name	AddDeleteButtonGroup
Data Source	MBTInsurancePremium
Arrange Method	Vertical

40. Add a **Command Button** for adding new records under **AddDeleteButtonGroup** with the following **properties:**

Property	Value
Name	AddButton
Command	New
Text	Add
Normal Image	New

41. **Add** a **Command Button** for deleting records under **AddDeleteButtonGroup** with the following **properties:**

Property	Value
Name	RemoveButton
Command	DeleteRecord
Text	Remove
Normal Image	Delete
Save Record	No

42. **Right click** on the **Premium** Tab Page and **add** a new **Grid** with the following **properties**:

Property	Value
Name	PremiumGrid
Data Source	MBTInsurancePremium

43. In the **Properties** pane of the **Premium tabPage** control under **DetailsTab** enter the following **properties:**

Property	Value
Caption	Premium

44. Drag the **InsurancePremium** Field group from **Data source > MBTInsurancePremium > Field groups** and drop under **PremiumGrid** control.
45. In the **Properties** pane of the **Design | Pattern** node enter the following **properties:**

Property	Value
Caption	Asset Insurance
Data Source	MBTAssetInsurance

46. In the **Properties** pane of the **AssetInsuranceActionPane** control enter the following property**:**

Property	Value
Height Mode	Auto

23.2 Task #2: Add a new Display Menu Item MBTAssetInsuranceDisplay

1. In your project MB500TrainingProjectUI in the Solution Explorer, add a New Item in Data Interface of type Display Menu Item, and Name it MBTAssetInsuranceDisplay
2. In the **Properties** pane of **MBTAssetInsuranceDisplay** enter the following **properties:**

Property	Value
Object	MBTAssetInsuranceDetails
Label	Insurance Details

23.3 Task #3: Add a new Menu in the Asset Table form

1. Find the **AssetTable** form in Application Explorer under **AOT > User Interface > Forms**.
2. Right click on **AssetTable** form and select **Create extension**
3. A new folder Form Extensions will be created under the MB500TrainingProjectUI project in Solution Explorer with a new extension object AssetTable.MB500Training
4. Open **AssetTable.MB500Training** in the designer
5. Open **ActionPane > FixedAsset** and **create** a **New Button Group MBTInsuranceButtonGroup**.
6. In the **Properties** pane of **MBTInsuranceButtonGroup** enter the following property**:**

Property	Value
Caption	Asset insurance

7. **Drag** the **Display Menu Item MBTAssetInsuranceDisplay** from your project and drop it under **MBTInsuranceButtonGroup**
8. Go to the newly created **MBTAssetInsuranceDisplay Menu Item** Button under Button Group **MBTInsuranceButtonGroup** and enter the following property**:**

Property	Value
Data Source	AssetTable

23.4 Task #4: Change Asset Table control property

1. **Open AssetTable.MB500Training** in the designer from the project MB500TrainingProjectUI under Solution Explorer.
2. Search for **Design > Tab > TabPageDetails > TabHeader > InsuranceTab.**
3. For the following two **groups**, **change** the **Allow Edit** property to **No**
    1. **Insurance**
    2. **Policy**
4. Save and build your project MB500TrainingProjectUI.

Output

1. Open Dynamics 365 Finance and Operations apps from the browser
2. Change the Legal entity to "USMF"
3. Open Fixed assets > Fixed assets > Fixed assets
4. Click on the **Insurance Details** menu button from action pane Fixed asset > Asset Insurance
5. Select the **New** button in the action pane and create a new record (Currently, the form has no validation. Hence, you can enter data as per your wish)
6. Click on the Premium tab and insert the premium records by clicking the **Add** button (Currently, the grid has no validation. Hence, you can enter data as per your wish)

Appendix

Project Import instructions:

- The lab resources may be found at https://aka.ms/mb500labresources
- To import a project, navigate to Visual Studio and select **Dynamics 365 > Import Project**
- In the File name field, select the file name with path of the .axpp file that you want to import
- For the first import, of MB500TrainingProject, select **New solution**. For subsequent ones, select the **Current solution** option.
- Select the **OK** button. You may safely override if prompted.

23.5 lab: title: 'Exercise 01: Lookup filter' module: 'Module 05: Code development and testing'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

24 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

25 Scenario

In this exercise, you will develop code to create a custom lookup on the Vendor field of the Asset Insurance form. The lookup will filter vendor records based on Vendor group. A vendor group field is added in Asset Parameters table and form, which will denote Insurance vendor only. The lookup in the vendor field will display those vendors, whose vendor group is defined in the Asset parameters.

26 Lab solution

26.1 Prerequisite

- To execute this exercise you need to have a Dynamics 365 Finance and Operations apps development environment with access to Visual Studio
- Import the following lab exercises as a prerequisite of this lab (for import instructions, please see appendix)
- MB500TrainingProject.axpp
- MB500TrainingProjectDataModel.axpp
- MB500TrainingProjectUI.axpp

26.2 Task #1: Vendor Lookup

1. **Right click** on the solution **MB500TrainingSolution**.
2. Go to **Add > New Project**.
3. Select **Dynamics 365 > Finance Operations**.
4. In the **Name** field, enter **MB500TrainingProjectCoding**.
5. Select **OK**. Set the new project's **Model** property to **MB500Training**.
6. In the Solution Explorer **right click** on the project **MB500TrainingProjectCoding**.
7. Go to **Add > New Item**.
8. Go to **Code > Class**.
9. In the **Name** field enter **MBTAssetInsuranceFormEventHandler**.
10. Select **Add**.
11. In the Classes folder, a new class **MBTAssetInsuranceFormEventHandler** will be added.
12. Find the form **MBTAssetInsuranceDetails** in Solution Explorer (it's in the UI project) and double click it to bring it into the designer
13. Find the control **Identification_VendAccount** under Design > DetailsGroup > Identification in the form **MBTAssetInsuranceDetails**
14. Expand the Events node of the field **Identification_VendAccount** and right-click on the *OnLookup* event to click on the **Copy event handler method.**
15. Open the newly created class **MBTAssetInsuranceFormEventHandler** and paste the event signature for the lookup method
16. The following method should be added in the class

    MBTAssetInsuranceFormEventHandler

    ```
    */// \<summary\>*
    *///*
    */// \</summary\>*
    */// \<param name="sender"\>\</param\>*
    */// \<param name="e"\>\</param\>*
    [FormControlEventHandler(formControlStr(MBTAssetInsuranceDetails, Identification_VendAccount), Forn
    public static void Identification_VendAccount_OnLookup(FormControl sender, FormControlEventArgs e)
    {
    }
    ```

17. Add the following code in the lookup method:

    ```
    FormRun callerFormRun;
    Query query = new Query();
    QueryBuildDataSource queryBuildDataSource;
    QueryBuildRange queryBuildRange;
    MBTAssetInsurance assetInsurance=sender.dataSourceObject().cursor();
    SysTableLookup sysTableLookup=SysTableLookup::newParameters(tableNum(VendTable), sender);
    ```

```
sysTableLookup.addLookupField(fieldNum(VendTable, AccountNum));
sysTableLookup.addLookupField(fieldNum(VendTable, Party));
queryBuildDataSource = query.addDataSource(tableNum(VendTable));
queryBuildRange = queryBuildDataSource.addRange(fieldNum(VendTable, VendGroup));
queryBuildRange.value(PurchParameters::find().MBTInsuranceVendGroup);
sysTableLookup.parmQuery(query);
sysTableLookup.performFormLookup();

FormControlCancelableSuperEventArgs formControlCancelSuper = e as
FormControlCancelableSuperEventArgs;
formControlCancelSuper.CancelSuperCall();
```

18. **Save** the Class and build the project.

Output

1. Navigate to your **Finance and Operations apps** environment in your browser.
2. Open **Procurement and sourcing** > **Setup** > **Vendors** > **Vendor Groups**
3. Add a new record:
    1. Vendor Group: 100
    2. Description: Insurance Vendor
4. Open **Procurement and sourcing** > **Setup** > **Procurement and sourcing parameters**
5. In the General tab, you will find a field Insurance Vendor Group; select 100 in that field.
6. Open **Procurement and sourcing** > **Vendors** > **All vendors**
7. Add two new vendors with Vendor group 100.
    1. Vendor account: INS0001, Name: Insurance vendor 001
    2. Vendor account: INS0002, Name: Insurance vendor 002
8. Open **Fixed assets** > **Fixed assets** > **Fixed assets**.
9. In the **Action pane**, go to **Fixed asset** > **Asset Insurance** > **Insurance Details**.
10. Select **New**.
11. Enter a value in the **Asset Insurance ID** field.
12. Check the lookup of the **Vendor Account** field. You will only find two vendors (INS0001 & INS0002) with vendor group 100.

Appendix

Project Import instructions:

- The lab resources may be found at https://aka.ms/mb500labresources
- To import a project, navigate to Visual Studio and select **Dynamics 365** > **Import Project**
- In the File name field, select the file name with path of the .axpp file that you want to import
- For the first import, of MB500TrainingProject, select **New solution**. For subsequent ones, select the **Current solution** option.
- Select the **OK** button. You may safely override if prompted.

26.3 lab: title: 'Exercise 02: Generate data' module: 'Module 05: Code development and testing'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

27 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

28 Scenario

In this exercise, you will develop code to generate premium data for each insurance record. Premium calculation will be based on the data entered in the insurance record.

29 Lab solution

29.1 Prerequisite

- To execute this exercise you need to have a Dynamics 365 Finance and Operations apps development environment with access to *Visual Studio*
- Import the following lab exercises as a prerequisite of this lab *(for import instructions, please see appendix)*
- MB500TrainingProject.axpp
- MB500TrainingProjectDataModel.axpp
- MB500TrainingProjectUI.axpp

29.2 Task #1: Task title

1. In the solution **MB500TrainingSolution,** search for project **MB500TrainingProjectCoding.**
2. If you do not find the project **MB500TrainingProjectCoding:**
    1. Right click on the solution **MB500TrainingSolution**.
    2. Go to **Add > New Project**.
    3. Select **Dynamics 365 > Finance and Operations**.
    4. In the **Name** field, enter **MB500TrainingProjectCoding**.
    5. **Click Add**. Set the new project's **Model** property to **MB500Training**.
3. In the Solution Explorer **right click** on the project **MB500TrainingProjectCoding**.
4. Go to **Add > New Item**.
5. Go to **Code > Runnable Class (Job)**.
6. In the **Name** field enter **MBTGeneratePremiums** and select **Add**.
7. In the Classes folder, a new class MBTGeneratePremiums will be added.
8. It will create a method by default with the following signature:
9. **Enter** the following code in the **main method**:

```
MBTAssetInsurance assetInsurance;
MBTInsurancePremium insurancePremium;
assetInsurance = \_args.record();
int cnt= 0;
switch(assetInsurance.Period)
{
    case MBTPeriod::Monthly:
        cnt=12;
        break;
    case MBTPeriod::Quarterly:
        cnt=4;
        break;
```

```
        case MBTPeriod::HalfYearly:
            cnt=2;
            break;
        case MBTPeriod::Annually:
            cnt=1;
            break;
        case MBTPeriod::None:
            cnt=0;
            break;
    }
    delete_from insurancePremium where insurancePremium.AssetInsuranceId ==
    assetInsurance.AssetInsuranceId;
    for (int i = 0; i \< cnt; i++)
    {
        insurancePremium.clear();
        insurancePremium.AssetInsuranceId = assetInsurance.AssetInsuranceId;
        insurancePremium.LineNum = i+1;
        insurancePremium.PremiumAmount = assetInsurance.PremiumAmount;
        insurancePremium.PremiumDueDate =
        MBTGeneratePremiums::getDueDate(assetInsurance.InsurancePolicyDate,(12/cnt)\i);
        insurancePremium.insert();
    }

    FormDataSource fds = \_args.record().dataSource();
    fds.research(true);
    fds.refresh();
```

10. **Create** another **method** within the same Class as follows:

```
    static date getDueDate(date \_PolicyDate, int \_noOfMnt)
    {
        date premiumDate = \_PolicyDate;
        for(int i = 0; i\<_noOfMnt; i++)
        {
            premiumDate = nextMth(premiumDate);
        }
        return premiumDate;
    }
```

11. **Save** the Class.
12. In the Solution Explorer **right click** on the project **MB500TrainingProjectCoding**.
13. Go to **Add > New Item**.
14. Go to **User Interface > Action Menu Item**.
15. In the **Name** field enter **MBTInsurancePremiumGenerate** and select **Add.**
16. A new folder will be created called Action Menu Items with a new Action Menu Item **MBTInsurancePremiumGenerate** within it.
17. Right click the menu item in the design canvas and select **Properties**.
18. Enter **Class** in **Object Type** field.
19. Enter **MBTGeneratePremium** in the **Object field**.
20. Enter **Generate premium** in the **Label** field.
21. **Right click** on **MB500TrainingProjectCoding** and **click Build**.
22. **Click** on **MB500TrainingProjectCode**.
23. **Open** Application Explorer.
24. Go to **AOT > User Interface > Forms**.
25. **Search** and select **MBTAssestInsuranceDetails**.

26. **Right click** on **MBTAssestInsuranceDetails** and **click Add to project**.
27. Make sure MBTAssestInsuranceDetails form is added under the project MB500TrainingProjectCoding.
28. **Open** the form **MBTAssetInsuranceDetails**.
29. Go to **Design > AssetInsuranceActionPane**.
30. **Right click** on **AssetInsuranceActionPane**.
31. **Create** a new **ButtonGroup**, **ActionButtonGroup**.
32. **Drag MBTInsurancePremiumGenerate** from the folder **Action Menu Items** and **drop** it under the **ActionButtonGroup**, Button Group.
33. Go to the **property** of **MBTInsurancePremiumGenerate** under **ActionButtonGroup**.
34. Enter **MBTAssetInsurance** in the **Data Source** property.
35. Save and build the project **MB500TrainingProjectCoding**

Output

1. Navigate to the Dynamics 365 Finance and Operations apps environment in the browser.
2. Open **Fixed assets > Fixed assets > Fixed assets**.
3. In the **Action pane**, go to **Fixed asset > Asset Insurance > Insurance Details**.
4. Select **New**.
5. Enter **12345** in the (upper) **Asset Insurance ID** field.
6. Enter **1001** in the **Vendor Account** field.
7. Enter **1111** in the **Policy Number** field.
8. Enter **10000** in the **Insured Value** field.
9. Enter **1/1/2021** in the **Insurance Policy Date** field.
10. Enter **100** in the **Premium Amount** field.
11. Select **Save**.
12. Select **Generate Premium**.
13. The premium value should be generated under the Premium tab.

Appendix

Project Import instructions:

- The lab resources may be found at https://aka.ms/mb500labresources
- To import a project, navigate to Visual Studio and select **Dynamics 365 > Import Project**
- In the File name field, select the file name with path of the .axpp file that you want to import
- For the first import, of MB500TrainingProject, select **New solution**. For subsequent ones, select the **Current solution** option.
- Select the **OK** button. You may safely override if prompted.

29.3 lab: title: 'Exercise 03: Perform validation' module: 'Module 05: Code development and testing'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

30 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

31 Scenario

In this exercise, you will develop code to add a validation in the table MBTAssetInsurance.

32 Lab solution

32.1 Prerequisite

- To execute this exercise you need to have a Dynamics 365 Finance and Operations apps development environment with access to Visual Studio
- Import the following lab exercises as a prerequisite of this lab (for import instructions, please see appendix)
- MB500TrainingProject.axpp
- MB500TrainingProjectDataModel.axpp
- MB500TrainingProjectUI.axpp

32.2 Task #1: Table validation

1. Find the project **MB500TrainingProjectCoding** in the Solution Explorer.
2. If you do not find the project:
    1. Right click on the solution **MB500TrainingSolution**.
    2. Go to **Add > New Project**.
    3. Select **Dynamics 365 > Finance and Operations**.
    4. In the **Name** field, enter **MB500TrainingProjectCoding**.
    5. Select **Add**. Set the new project's **Model** property to **MB500Training**.
3. In the Solution Explorer **right click** on the project **MB500TrainingProjectCoding**.
4. Go to **Add > New Item**.
5. Go to **Code > Class**.
6. In the **Name** field enter **MBTAssetInsuranceTableEventHandler** and select **Add**.
7. In the Classes folder, a new class **MBTAssetInsuranceTableEventHandler** will be added.
8. In the Solution Explorer, find and open table **MBTAssetInsurance** from the project **MB500TrainingProjectDataM**
9. Expand the Events node and find the event *onValidatedWrite*
10. Right click the event *onValidatedWrite* and click on **Copy event handler method.**
11. Open class **MBTAssetInsuranceTableEventHandler** and paste the copied event handler signature within the brackets. It will look like this:

```
/// \<summary\>
///
/// \</summary\>
/// \<param name="sender"\>\</param\>
/// \<param name="e"\>\</param\>
[DataEventHandler(tableStr(MBTAssetInsurance), DataEventType::ValidatedWrite)]
public static void MBTAssetInsurance_onValidatedWrite(Common sender, DataEventArgs e)
{
}
```

12. **Enter** the following code within the MBTAssetInsurance_onValidatedWrite **method**:

```
ValidateEventArgs validateArgs = e as ValidateEventArgs;
MBTAssetInsurance assetInsurance = sender as MBTAssetInsurance;
boolean result = validateArgs.parmValidateResult();
if (assetInsurance.Period == MBTPeriod::None)
{
   result = checkFailed("Enter the Premium Payment Period");
    validateArgs.parmValidateResult(result);
}
```

13. Save the class and build the project **MB500TrainingProjectCoding**

33 Output

1. Navigate to your Dynamics 365 Finance and Operations environment from browser.
2. Open **Fixed assets > Fixed assets > Fixed assets**.
3. In the **Action pane**, navigate to **Fixed asset > Asset Insurance > Insurance Details**.
4. Select **New**.
5. Enter **12345** in the **Asset Insurance ID** field.
6. Enter **1001** in the **Vendor Account** field.
7. Enter **1111** in the **Policy Number** field.
8. Enter **10000** in the **Insured Value** field.
9. Enter **1/1/2021** in the **Insurance Policy Date** field.
10. Enter **100** in the **Premium Amount** field.
11. In the **Premium Payment** field select *Not Applicable*.
12. An error message should appear when you try to save the record.

Appendix

Project Import instructions:

- The lab resources may be found at https://aka.ms/mb500labresources
- To import a project, navigate to Visual Studio and select **Dynamics 365 > Import Project**
- In the File name field, select the file name with path of the .axpp file that you want to import
- For the first import, of MB500TrainingProject, select **New solution**. For subsequent ones, select the **Current solution** option.
- Select the **OK** button. You may safely override if prompted.

33.1 lab: title: 'Exercise 01: Extend the number sequence framework' module: 'Module 06: Frameworks'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

34 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

35 Scenario

In this lab, number sequence framework is extended to create a new number sequence for the asset insurance transactions under Fixed Asset.

36 Lab solution

36.1 Prerequisite

- To execute this exercise you need to have a Dynamics 365 Finance and Operations apps development environment with access to Visual Studio
- Import the following lab exercises as a prerequisite of this lab (for import instructions, please see appendix)
- MB500TrainingProject.axpp
- MB500TrainingProjectDataModel.axpp
- MB500TrainingProjectUI.axpp
- MB500TrainingProjectCode.axpp

36.2 Task #1: Asset insurance number sequence

1. Right click on the Visual Studio solution under Solution Explorer.
2. Click on Add > New project and create a new project **MB500TrainingProjectExtension**. Set the new project's **Model** property to **MB500Training**.

   To create a new number sequence, first you need to add the number sequence in the LoadModule() method of the list of number sequences specific to the module. We are planning to create a new number sequence in the Fixed Asset module; hence we need to create a Chain of Control for the class NumberSeqModuleAsset.
3. Right-click on the Project **MB500TrainingProjectExtension** and create a new class **NumberSeqModuleAsset_Extension**
4. This should be the signature of the class (rather than the default):

```
[ExtensionOf(classStr(NumberSeqModuleAsset))]
final class NumberSeqModuleAsset_Extension
{
}
```

5. A new chain of control should be created for the method LoadModule() to add the new number sequence for the Asset insurance transaction. This will be the code:

```
protected void loadModule()
{
    next loadModule();
    NumberSeqDatatype datatype =    NumberSeqDatatype::construct();
    //Setup Asset Insurance Number
    datatype.parmDatatypeId(extendedtypenum(MBTAssetInsuranceId));
    datatype.parmReferenceHelp("Asset Insurance Id");
    datatype.parmWizardIsContinuous(false);
    datatype.parmWizardIsManual(NoYes::No);
    datatype.parmWizardIsChangeDownAllowed(NoYes::No);
    datatype.parmWizardIsChangeUpAllowed(NoYes::No);
    datatype.parmSortField(8);
    datatype.addParameterType(NumberSeqParameterType::DataArea, true, false);
    this.create(datatype);
}
```

   The AssetParameters class is extended to add a new method to create reference of the asset insurance number sequence.
6. Right click on the project **MB500TrainingProjectExtension** and click on Add > New item to create a new class **AssetParameters_Extension** with the following signature

```
[ExtensionOf(tableStr(AssetParameters))]
final class AssetParameters_Extension
{
}
```

7. Create a new method **numRefMBTAssetInsuranceId** in the class to refer to the Asset Insurance transaction id.

   You need to call the number sequence framework when a new record is added in the Asset Insurance table

```
public static NumberSequenceReference numRefMBTAssetInsuranceId()
{
    return NumberSeqReference::findReference(extendedTypeNum(MBTAssetInsuranceId));
}
```

8. Right click on the project **MB500TrainingProjectExtension** and click on Add > New item to create a new class **MBTAssetInsuranceTableEventHandler2**
9. Open table **MBTAssetInsurance** from Solution Explorer and expand the Events node.
10. Right click the *onInitializedRecord* and click on **Copy event handler method**.
11. Paste the event handler method within the newly created class **MBTAssetInsuranceTableEventHandler2**
12. This will be the signature of the class: (there should be no backslashes)

```
/// \<summary\>
///
/// \</summary\>
/// \<param name="sender"\>\</param\>
/// \<param name="e"\>\</param\>
[DataEventHandler(tableStr(MBTAssetInsurance),DataEventType::InitializedRecord)]
public static void MBTAssetInsurance_onInitializedRecord(Common sender,DataEventArgs e)
{
}
```

13. Add the following code within the class (noting that you already have the first two and last one lines):

```
public static void MBTAssetInsurance_onInitializedRecord(Common sender,
DataEventArgs e)
{
    MBTAssetInsurance assetInsurance = sender as MBTAssetInsurance;
    NumberSeq NumSeq;
    NumSeq = NumberSeq::newGetNum(AssetParameters::numRefMBTAssetInsuranceId(),true);
    assetInsurance.AssetInsuranceId = NumSeq.num();
}
```

   Create a job to reload all the number sequence modules.

14. Right click on the project **MB500TrainingProjectExtension** and click on Add > New item to create a new Runnable class (job) **MBTNumberSeqLoadAll**
15. The class is created with a default *main* method as follows:

```
class MBTNumberSeqLoadAll
{
    /// \<summary\>
    /// Runs the class with the specified arguments.
    /// \</summary\>
    /// \<param name = "_args"\>The specified arguments.\</param\>
    public static void main(Args \_args)
    {
    }
}
```

16. Add these two lines of code in the *main* method:

```
public static void main(Args \_args)
{
    NumberSeqApplicationModule::loadAll();
    info("Number sequence loaded");
}
```

17. Right click on the project **MB500TrainingProjectExtension** and click on Add > New
18. Select Action Menu Item within User Interface and create a new action menu item **MBTNumberSeqLoadAll**
19. A new folder Action Menu Items will be created in the project **MB500TrainingProjectExtension** with the action menu item **MBTNumberSeqLoadAll** in it.
20. Open the action menu item **MBTNumberSeqLoadAll** in the designer and right click on it to open the properties pane
21. Change the following properties of the action menu item **MBTNumberSeqLoadAll**
    1. **Label**: Load number sequence
    2. **Object type**: Class
    3. **Object**: MBTNumberSeqLoadAll
22. In the application explorer, search for menu **OrganizationAdministration**
23. Right click on the menu **OrganizationAdministration** and click on Create extension
24. A new folder Menu Extensions will be created in the project **MB500TrainingProjectExtension** with the menu extension **OrganizationAdministration.MB500Training** in it.
25. Open **OrganizationAdministration.MB500Training** in the designer
26. Drag action menu item **MBTNumberSeqLoadAll** from Solution Explorer and drop it to the *Periodic* folder in **OrganizationAdministration.MB500Training**
27. Save and build the project **MB500TrainingProjectExtension**

37 Lab solution

1. Navigate to the Dynamics 365 Finance and Operations apps instance in the browser and execute **Organization Administration > Periodic > Load Number Sequence**
2. At the end of the execution, you will get a message “Number sequence loaded”
3. Navigate to Dynamics 365 Finance and Operations instance and open **Fixed assets > Setup > Fixed assets parameters**
4. Go to the **Number sequences** tab and you will find a new reference Asset Insurance ID. In the Number Sequence code field select Fixe_17
5. **Save** the record
6. Go to the Dynamics 365 Finance and Operations apps instance and open **Fixed assets > Fixed assets > Fixed assets**
7. Open the Asset insurance form by selecting on the action pane **Fixed asset > Asset Insurance > Insurance Details**
8. Select the **New** button in the action pane.
9. A new record will be created with Asset Insurance ID auto-updated from the Number sequence framework.

Appendix

Project Import instructions:

- The lab resources may be found at https://aka.ms/mb500labresources
- To import a project, navigate to Visual Studio and select **Dynamics 365 > Import Project**
- In the File name field, select the file name with path of the .axpp file that you want to import

- For the first import, of MB500TrainingProject, select **New solution**. For subsequent ones, select the **Current solution** option.
- Select the **OK** button. You may safely override if prompted.

37.1 lab: title: 'Exercise 01: Develop a BI report' module: 'Module 07: Reporting'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

38 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

39 Scenario

In this lab, a BI report is developed based on the premium amount paid for different fixed assets.

40 Lab solution

40.1 Prerequisite

- To execute this exercise you need to have a Dynamics 365 Finance and Operations apps development environment with access to Visual Studio
- Import the following lab exercise as a prerequisite of this lab (for import instructions, please see appendix)
- MB500TrainingProject.axpp
- MB500TrainingProjectDataModel.axpp
- MB500TrainingProjectUI.axpp
- MB500TrainingProjectCode.axpp
- MB500TrainingProjectExtension.axpp
- MB500TrainingProjectDataManagement.axpp

40.2 Task #1: Asset insurance advanced BI report

1. Right click on the Training solution under Solution Explorer.
2. Select **Add** > **New project** and create a new project **MB500TrainingProjectReporting**
3. Right click the project and select **Properties** to set the **Model** to **MB-500 Training,** and the **Company** to **USMF.** Set the new project's **Model** property to **MB500Training**. Select **Apply** and **OK**

   MBTAssetInsurance is the table that captures information of the insurances for each fixed asset. But this table doesn't contain any field capturing yearly premium amount. Hence we will create a view with a computed field showing yearly premium amount. This field will be used in the BI reports.
4. Right-click on the Project **MB500TrainingProjectReporting** and go to Add > New Item
5. Select View under Data Model to create a new View **MBTAssetInsuranceView**
6. In the view **MBTAssetInsuranceView**, under the View Metadata node add a new Data source **MBTAssetInsurance** with table **MBTAssetInsurance**
7. Right click on the **MBTAssetInsurance** data source and verify the Dynamic Fields property is Yes

8. Drag all fields under View Metadata > Data sources > MBTAssetInsurance > Fields node and drop them to the Fields node under the root of the view, except these fields (if you don't see them, right click on the solution and Build)
    1. DataAreaId
    2. Partition
    3. RecId
    4. RecVersion
9. Right click on the Fields node under root of the view and add a new Real Computed Column named **TotalPremium**
10. Go to the Methods node under the root of the view and add a **New Method totalPremium**
11. Add the following code in the **totalPremium** method (replacing the private void method). *Note: This code is multiplying the premium amount based on the period selected to calculate the total premium.*

```
private static str totalPremium()
{
    const str cntDataSourceName = "MBTAssetInsurance";
    const str cntFieldPremiumAmount = "PremiumAmount";
    const str cntFieldPeriod = "Period";
    str totalPremium;
    str premiumAmount;
    str period;
    DictView dictViewAssetInsurance;
    dictViewAssetInsurance = new DictView(tableNum(MBTAssetInsuranceView));
    premiumAmount = dictViewAssetInsurance.computedColumnString (cntDataSourceName, cntFieldPremiu
    period = dictViewAssetInsurance.computedColumnString (cntDataSourceName, cntFieldPeriod, Fieldl
    totalPremium = premiumAmount + "\* iif(" + period + " = 0, 0, iif(" + period + " = 1, 12, iif(
    return totalPremium;
}
```

12. Save. Select the **TotalPremium** field under the Fields node of the view and change the **View Method** property to **totalPremium**
13. Save and build the project **MB500TrainingProjectReporting.** If it doesn't build, then right click the entire solution and **build** it. Right click the DataModel project and **synchronize** it, and then the Reporting project and **synchronize** it
14. Find the view **MBTAssetInsuranceView** under the Solution Explorer > **MB500TrainingProjectReporting** project
15. Right click on the view **MBTAssetInsuranceView** and select **Open table browser**
16. The table browser of the view **MBTAssetInsuranceView** will open with values populated in the **TotalPremium** field. *Create an aggregate dimension for AssetTable, which will enable drill down of Insurance premium data based on Asset group, location etc.*
17. Right-click on the Project **MB500TrainingProjectReporting** and select **Add** > **New Item**
18. Select Aggregate Dimensions under Analytics to create a new aggregate dimension **MBTAssetDimension**
19. Open the property pane of the root node of **MBTAssetDimension**
20. Change the Table property of the aggregate dimension **MBTAssetDimension** to **AssetTable**
21. Expand the Attributes node, which should have the following fields already added by default. In case these are not added, right click on the Attributes node and select **New Dimension Attribute** to add them. *Create an aggregate measurement for the view created from the Asset Insurance table. Two measures will be created from this view; like – Total premium amount and total count of insurance policies.*
    1. MBTAssetDimension (Dimension field Reference: DataAreaId, AssetId)
    2. AssetGroup
    3. Name

4. NameAlias
5. Location
6. MajorType

22. Right-click on the Project **MB500TrainingProjectReporting** and select **Add** > **New Item**
23. Select Aggregate Measurements under Analytics to create a new aggregate measurement **MBTAssetInsuranceMeasurement**
24. Open the property pane of the root node of **MBTAssetInsuranceMeasurement**
25. Change the Label property of the aggregate measurement **MBTAssetInsuranceMeasurement** to **Asset insurance**.
26. Select the default MeasureGroup and in the property pane change the Table property to **MBTAssetInsuranceView**
27. The Name property of the MeasureGroup should also change to **MBTAssetInsuranceView**
28. **Save all**.
29. Go to the Dimensions node of the MeasureGroup **MBTAssetInsuranceView**, which should have two default values, as follows:
    1. **Company**: Expand the company node till the end of nodes and ensure you find the following: BICompanyView.id == MBTAssetInsuranceView.DataAreaId
    2. **Date_**: Expand the Date_ node till end of nodes and update as following: BIDateDimensionValue.Date == MBTAssetInsuranceView.InsurancePolicyDate
30. Drag the aggregate dimension **MBTAssetDimension** from the Solution Explorer and drop it on the Dimensions node of the measureGroup **MBTAssetInsuranceView**
31. Expand the MBTAssetDimension under the Dimensions node of the measureGroup **MBTAssetInsuranceView** till end of nodes and ensure as following:

    AssetTable.AssetId == MBTAssetInsuranceView.AssetId
32. Go to the Measures node of measureGroup **MBTAssetInsuranceView** and add **New Measure**s with property values as in the table below. *Create a key performance indicator for the measurement created above, which can be show the total premium amount based on given criteria*

#	Name	Field	Default Aggregate	Label
I	TotalPremium	TotalPremium	Sum	Total premium
II	PolicyCount	AssetInsuranceId	Count	Policy count

33. Right-click on the Project **MB500TrainingProjectReporting** and select **Add** > **New Item**
34. Select Key Performance Indicator under Analytics to create a new key performance indicator (KPI) **MBTAssetInsuranceKPI**
35. Open the property pane of the *root* node of **MBTAssetInsuranceKPI** and change them as such:
    1. Label: Asset insurance
    2. Measurement: **MBTAssetInsuranceMeasurement**
    3. Refresh Frequency: AsFastAsPermissible
36. Select the property pane of the **Goal** node of the KPI **MBTAssetInsuranceKPI** and change the Value property to 1000
37. Select the property pane of the **Value** node of the KPI **MBTAssetInsuranceKPI** and change the Measure Group property to **MBTAssetInsuranceView** and the Measure property to **TotalPremium.** *Create a tile, which can display the KPI on a Workspace*
38. Right-click on the Project **MB500TrainingProjectReporting** and select **Add** > **New Item**
39. Select Tile under User Interface to create a new Tile **MBTAssetInsuranceTile**
40. Open the properties pane of **MBTAssetInsuranceTile**

41. Change the Type property of the Tile **MBTAssetInsuranceTile** to KPI.
42. The KPI property of the Tile **MBTAssetInsuranceTile** will be enabled and change it to **MBTAssetInsuranceKPI.** *Create an extension of the workspace AssetWorkspace. In the Tile section of the workspace add the new tile, which is created in the previous section*
43. In the (View >) Application Explorer, search for the form **AssetWorkspace**.
44. Right click on **AssetWorkspace** and select **Create Extension**
45. A new folder **Form Extensions** will be created in the project MB500TrainingProjectReporting under Solution Explorer with a new object AssetWorkspace.MB500Training within it.
46. In AssetWorkspace.MB500Training, create a **New Tile Button** under Design > ControlTab > MyWork > PanoramaTab > SummarySection
47. Set the properties of the Tile Button as follows
    1. Name: MBTAssetInsurance
    2. Tile: MBTAssetInsuranceTile
    3. Tile Display: Auto
48. Save and build the project **MB500TrainingProjectReporting**

Output

Please note that the module 10 exercise is a prerequisite to testing this exercise

1. Right click the DataManagement project and Synchronize to the database
2. Open the Dynamics 365 Finance and Operations portal in your browser
3. Open **Procurement and sourcing > Setup > Vendors > Vendor Groups**
4. Add a new record (if not already there from a previous exercise):
    1. Vendor Group: **100**
    2. Description: **Insurance vendors**
5. Open **Procurement and sourcing > Setup > Procurement and sourcing parameters**
6. In the General tab, you will find a field Insurance Vendor Group; select 100 in that field.
7. Open **Procurement and sourcing > Vendors > All vendors**
8. Add four new vendors with **Group** 100 (if not already there).
    1. Vendor Code: INS0001, Name: Insurance vendor 001
    2. Vendor Code: INS0002, Name: Insurance vendor 002
    3. Vendor Code: INS0003, Name: Insurance vendor 003
    4. Vendor Code: INS0004, Name: Insurance vendor 004
9. Navigate to the menu **Organization Administration** and execute its **Periodic > Load Number Sequence**
10. At the end of the execution, you will get a message “Number sequence loaded”
11. Navigate to the module **Fixed Assets** and open its **Setup > Fixed assets parameters**
12. Go to the Number sequences tab and you will find a new reference Asset Insurance ID. In the Number Sequence code field select **Fixe_17**
13. **Save** the record
14. Open the **Data Management** workspace and select the **Framework parameters** tile
15. In the **Entity settings** tab, select the **Refresh entity list** button
16. In the **Data Management** workspace, open **Data Entities**
17. When the entity list is refreshed, search for the **Asset Insurance** entity and select **Modify target mapping** in the action menu

18. Ensure the mapping between Staging and Target is correct
19. In the **Data Management** workspace, open **Import**
20. Enter the following data:
    1. Group name: Asset insurance import
    2. Data project operation type: Import
    3. Click on **+Add file**
    4. Entity name: Asset Insurance
    5. Source data format: CSV
    6. Default refresh type: Full push only
    7. Click on button **Upload and add** and select *assetInsurance.csv* (you may download this from https://aka.ms/mb500labresources)
    8. Select the **Close** button
21. In the **Selected entities** line, select **View map**
22. Navigate to the **Mapping details** tab and select the **Auto-generated** checkbox in the ASSETINSURANCEID line.
23. Select **Save** and close this form
24. Select the **Import** button in the action pane
25. Monitor the status of the import process
26. Open the *workspace* **Fixed asset management**
27. You will find a new Tile added in the Tiles section of the Workspace with label Asset Insurance
28. Open the Tile and explore the Asset Insurance BI report

Appendix

Project Import instructions:

- The lab resources may be found at https://aka.ms/mb500labresources
- To import a project, navigate to Visual Studio and select **Dynamics 365 > Import Project**
- In the File name field, select the file name with path of the .axpp file that you want to import
- For the first import, of MB500TrainingProject, select **New solution**. For subsequent ones, select the **Current solution** option.
- Select the **OK** button. You may safely override if prompted.

40.3 lab: title: 'Exercise 01: Create a custom business event' module: 'Module 08: Integration'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

41 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

42 Scenario

In this lab, a custom business event will be created that can capture the event when a new insurance is added for a fixed asset. It can also capture the Vendor email, so that email can be sent to the vendor.

43 Lab solution

43.1 Prerequisite

- To execute this exercise you need to have a Dynamics 365 Finance and Operations *apps* development environment with access to *Visual Studio*
- Import the following lab exercise as a prerequisite of this lab (for import instructions, please see appendix)
- MB500TrainingProject.axpp
- MB500TrainingProjectDataModel.axpp
- MB500TrainingProjectUI.axpp

43.2 Task #1: Create a custom business event

1. Right click on the Training solution under Solution Explorer.
2. Click on Add > New project and create a new project **MB500TrainingProjectBusinessEvent**. Set the new project's **Model** property to **MB500Training**.

   A new method is added in the VendTable table, which will return the email address of the vendor
3. Right-click on the Project **MB500TrainingProjectBusinessEvent** and go to Add > New Item
4. Select Class under Code to create a new class **MBTVendTableTable_Extension**
5. Change the signature for the class **MBTVendTableTable_Extension** to:

```
[ExtensionOf(tableStr(VendTable))]
final class MBTVendTableTable_Extension
{
}
```

6. Add a new method getEmail() to find the Email of the vendor. This is the code:

```
public Email getEmail()
{
    DirPartyLocation dirPartyLocation;
    LogisticsElectronicAddress logisticsElectronicAddress;
    LogisticsElectronicAddressRole logisticsElectronicAddressRole;
    LogisticsLocationRole logisticsLocationRole;
    Email emailValue;
    while select dirPartyLocation
        where dirPartyLocation.party == this.Party
    {
        while select logisticsElectronicAddress
            where logisticsElectronicAddress.Location == dirPartyLocation.Location
            && logisticsElectronicAddress.Type == LogisticsElectronicAddressMethodType::Email
        {
            while select logisticsElectronicAddressRole
                where logisticsElectronicAddressRole.ElectronicAddress == logisticsElectronicAddres
            join logisticsLocationRole
                where logisticsLocationRole.RecId == logisticsElectronicAddressRole.LocationRole
            {
                if(!emailValue)
                {
                    emailValue = logisticsElectronicAddress.Locator;
                }
            }
        }
    }
    return emailValue;
}
```

A new data contract is added that will capture the data to be sent while the insert event will be triggered.

7. Right-click on the Project **MB500TrainingProjectBusinessEvent** and go to Add > New Item
8. Select Class under Code to create a new class **MBTAssetInsuranceBusinessEventContract**
9. Use the following signature for the class **MBTAssetInsuranceBusinessEventContract:**

```
*/// \<summary\>*
*/// The data contract for a \<c\>AssetInsurancePostedBusinessEvent\</c\>.*
*/// \</summary\>*
[DataContract]
public final class MBTAssetInsuranceBusinessEventContract extends
BusinessEventsContract
{
}
```

10. Add the member variables in the contract class **MBTAssetInsuranceBusinessEventContract**.

```
private AssetId AssetId;
private MBTAssetInsuranceId AssetInsuranceId;
private VendAccount VendAccount;
private Email VendEmail;
private String10 InsurancePolicyDate;
private AssetInsurancePolicyNum AssetInsurancePolicyNum;
private AssetInsuredValue AssetInsuredValue;
private AssetPolicyAmount PremiumAmount;
private MBTPeriod PaymentPeriod;
private AssetInsuranceAgent InsuranceAgent;
private LegalEntityDataAreaId legalEntity;
```

11. Add the default constructor in the contract class **MBTAssetInsuranceBusinessEventContract**.

```
private void new()
{
}
```

12. Add parm methods for all the member variables

```
[DataMember('AssetId'), BusinessEventsDataMember("Asset ID")]
public AssetId parmAssetId(AssetId \_assetId = AssetId)
{
AssetId = \_assetId;
return AssetId;
}

[DataMember('AssetInsuranceId'), BusinessEventsDataMember("Asset Insurance ID")]
public MBTAssetInsuranceId parmAssetInsuranceId(MBTAssetInsuranceId \_assetInsuranceId = AssetInsu
{
AssetInsuranceId = \_assetInsuranceId;
return AssetInsuranceId;
}

[DataMember('VendAccount'), BusinessEventsDataMember("Vendor Account")]
public VendAccount parmVendAccount(VendAccount \_vendAccount = VendAccount)
{
VendAccount = \_vendAccount;
return VendAccount;
}

[DataMember('VendEmail'), BusinessEventsDataMember("Vendor Email")]
public Email parmVendEmail(Email \_vendEmail = VendEmail)
{
VendEmail = \_vendEmail;
return VendEmail;
}
```

```
[DataMember('InsurancePolicyDate'), BusinessEventsDataMember("Insurance Policy Date")]
public String10 parmPolicyDate(String10 \_insurancePolicyDate = InsurancePolicyDate)
{
InsurancePolicyDate = \_insurancePolicyDate;
return InsurancePolicyDate;
}

[DataMember('AssetInsurancePolicyNum'), BusinessEventsDataMember("Policy Number")]
public AssetInsurancePolicyNum parmPolicyNum(AssetInsurancePolicyNum \_policyNum = AssetInsurancePo
{
AssetInsurancePolicyNum = \_policyNum;
return AssetInsurancePolicyNum;
}

[DataMember('AssetInsuredValue'), BusinessEventsDataMember("Insured Value")]
public AssetInsuredValue parmInsuredValue(AssetInsuredValue \_InsuredValue = AssetInsuredValue)
{
AssetInsuredValue = \_InsuredValue;
return AssetInsuredValue;
}

[DataMember('PremiumAmount'), BusinessEventsDataMember("Premium Amount")]
public AssetPolicyAmount parmPremiumAmount(AssetPolicyAmount \_premiumAmount = PremiumAmount)
{
PremiumAmount = \_premiumAmount;
return PremiumAmount;
}

[DataMember('PaymentPeriod'), BusinessEventsDataMember("Payment Period")]
public MBTPeriod parmPaymentPeriod(MBTPeriod \_paymentPeriod = PaymentPeriod)
{
PaymentPeriod = \_paymentPeriod;
return PaymentPeriod;
}

[DataMember('InsuranceAgent'), BusinessEventsDataMember("Insurance Agent")]
public AssetInsuranceAgent parmInsuranceAgent(AssetInsuranceAgent \_insuranceAgent = InsuranceAgen
{
InsuranceAgent = \_insuranceAgent;
return InsuranceAgent;
}

[DataMember('LegalEntity'), BusinessEventsDataMember("Legal Entity")]
public LegalEntityDataAreaId parmLegalEntity(LegalEntityDataAreaId \_legalEntity = legalEntity)
{
legalEntity = \_legalEntity;
return legalEntity;
}
```

13. Add a new method **initialize** to initialize the value of the member variables of the contract class **MBTAssetInsuranceBusinessEventContract**

```
private void initialize(MBTAssetInsurance \_assetInsurance)
{
    AssetId = \_assetInsurance.AssetId;
    AssetInsuranceId = \_assetInsurance.AssetInsuranceId;
    VendAccount = \_assetInsurance.VendAccount;
    VendEmail = VendTable::find(_assetInsurance.VendAccount).getEmail();
    InsurancePolicyDate = date2Str(_assetInsurance.InsurancePolicyDate, 213, DateDay::Digits2, Dat
    AssetInsurancePolicyNum = \_assetInsurance.InsurancePolicyNum;
    AssetInsuredValue = \_assetInsurance.InsuredValue;
    PremiumAmount = \_assetInsurance.PremiumAmount;
```

```
    PaymentPeriod = \_assetInsurance.Period;
    InsuranceAgent = \_assetInsurance.InsuranceAgent;
}
```

14. Add a new method **newAssetInsurance** in the contract class **MBTAssetInsuranceBusinessEvent-Contract** as follows:

```
public static MBTAssetInsuranceBusinessEventContract newAssetInsurance(MBTAssetInsurance \_assetIns
{
    var contract = new MBTAssetInsuranceBusinessEventContract();
    contract.initialize(_assetInsurance);
    return contract;
}
```

A new business event class is added that will capture the insert event on the Asset Insurance table.

15. Right-click on the Project **MB500TrainingProjectBusinessEvent** and go to Add > New Item
16. Select Class under Code to create a new class **MBTAssetInsurancePostedBusinessEvent**
17. Use the following signature for the class **MBTAssetInsurancePostedBusinessEvent:**

```
*/// \<summary\>*
*/// Asset Insurance posted business event.*
*/// \</summary\>*
[BusinessEvents(classStr(MBTAssetInsuranceBusinessEventContract), 'Asset Insurance', 'Asset Insura
public final class MBTAssetInsurancePostedBusinessEvent extends BusinessEventsBase
{
}
```

18. Add the member variable in the class **MBTAssetInsurancePostedBusinessEvent**.

```
MBTAssetInsurance AssetInsurance;
```

19. Add the following method in the class **MBTAssetInsurancePostedBusinessEvent:**

```
[Wrappable(true), Replaceable(true)]
public BusinessEventsContract buildContract()
{
    return MBTAssetInsuranceBusinessEventContract::newAssetInsurance(AssetInsurance);
}
```

20. Add the default constructor in the class **MBTAssetInsurancePostedBusinessEvent**:

```
private void new()
{
}
```

21. Add a parm method for the member variable in the class **MBTAssetInsurancePostedBusinessEvent.**

```
private MBTAssetInsurance parmAssetInsurance(MBTAssetInsurance \_assetInsurance = AssetInsurance)
{
    AssetInsurance = \_assetInsurance;
    return AssetInsurance;
}
```

22. Add the following new method to trigger the business event in the class **MBTAssetInsurancePosted-BusinessEvent**

Create an extension of the Asset Insurance table and add a chain of command of the insert() method to capture the insert trigger.

```
*/// \<summary\>*
*/// Creates a \<c\>AssetInsurancePostedBusinessEvent\</c\> from a \<c\>MBTAssetInsurance\</c\> re
*/// \</summary\>*
*/// \<param name = "_assetInsurance"\> A \<c\>MBTAssetInsurance\</c\> record.\</param\>*
*/// \<returns\>A \<c\>MBTAssetInsurancePostedBusinessEvent\</c\>.\</returns\>*
public static MBTAssetInsurancePostedBusinessEvent newAssetInsurance(MBTAssetInsurance \_assetInsu
{
    MBTAssetInsurancePostedBusinessEvent businessEvent = new MBTAssetInsurancePostedBusinessEvent(
```

```
        businessEvent.parmAssetInsurance(_assetInsurance);
        return businessEvent;
    }
```

23. Right-click on the Project **MB500TrainingProjectBusinessEvent** and select Add > New Item
24. Select Class under Code to create a new class **MBTAssetInsuranceTable_Extension**
25. Use the following signature for the class **MBTAssetInsuranceTable_Extension:**

```
[ExtensionOf(tableStr(MBTAssetInsurance))]
final class MBTAssetInsuranceTable_Extension
{
}
```

26. Add the following code within the brackets to create a chain of command for the insert() method:

```
public void insert()
{
    next insert();
    if (BusinessEventsConfigurationReader::isBusinessEventEnabled(classStr(MBTAssetInsurancePosted
    {
        MBTAssetInsurancePostedBusinessEvent::newAssetInsurance(this).send();
    }
}
```

27. Save and Build the project **MB500TrainingProjectBusinessEvent**

Output

1. Navigate to Dynamics 365 Finance and Operations apps in your browser
2. Navigate to **System administration** > **Setup** > **Business events** > **Business events catalog**
3. Select **Manage** > **Rebuild business event catalog**
4. You will find a new business event is created named **MBTAssetInsurancePostedBusinessEvent** under the **Fixed Asset** category

Appendix

Project Import instructions:

- The lab resources may be found at https://aka.ms/mb500labresources
- To import a project, navigate to Visual Studio and select **Dynamics 365** > **Import Project**
- In the File name field, select the file name with path of the .axpp file that you want to import
- For the first import, of MB500TrainingProject, select **New solution**. For subsequent ones, select the **Current solution** option.
- Select the **OK** button. You may safely override if prompted.

43.3 lab: title: 'Exercise 01: Demonstrate asynchronous vs. synchronous processing' module: 'Module 09: Security and performance'

MB-500: Microsoft Dynamics 365: Finance and Operations Apps Developer

44 Change Record

Version	Date	Change
1.0	30 Apr 2021	Initial release

45 Scenario

In this lab, a premium is generated for all the insurance records in one button click. Task #1 is generating the premium records using synchronous processing. Task #2 generates the same records using asynchronous processing.

46 Lab solution

46.1 Prerequisite

- To execute this exercise you need to have a Dynamics 365 Finance and Operations apps development environment with access to Visual Studio
- Import the following lab exercise as a prerequisite of this lab (for import instructions, please see appendix)
- MB500TrainingProject.axpp
- MB500TrainingProjectDataModel.axpp
- MB500TrainingProjectUI.axpp
- MB500TrainingProjectCode.axpp
- MB500TrainingProjectExtension
- MB500TrainingProjectDataManagement

46.2 Task #1: Synchronous process

1. Create a new Finance Operations project within the training solution called **MB500TrainingProjectAsync**. Set the new project's **Model** property to **MB500Training**.
2. **Create** a new **Runnable class MBTAllPremiumGenerate**
3. **Add** the following new **method** in **MBTAllPremiumGenerate.**

```
static date getDueDate(date \_PolicyDate, int \_noOfMnt)
{
    date premiumDate = \_PolicyDate;
    for(int i = 0; i\<_noOfMnt; i++)
    {
        premiumDate = nextMth(premiumDate);
    }
    return premiumDate;
}
```

4. **Add** the following new **method** in **MBTAllPremiumGenerate.**

```
static void generateAllPremium(boolean \_asynch = true)
{
    MBTAssetInsurance assetInsurance;
    MBTInsurancePremium insurancePremium;
    int cnt= 0;
    while select assetInsurance
    {
        switch(assetInsurance.Period)
        {
            case MBTPeriod::Monthly:
                cnt=12;
                break;
            case MBTPeriod::Quarterly:
                cnt=4;
                break;
            case MBTPeriod::HalfYearly:
                cnt=2;
                break;
            case MBTPeriod::Annually:
```

```
                cnt=1;
                break;
            case MBTPeriod::None:
                cnt=0;
                break;
        }
        delete_from insurancePremium where insurancePremium.AssetInsuranceId == assetInsurance.Ass
        for (int i = 0; i \< cnt; i++)
        {
            insurancePremium.clear();
            insurancePremium.AssetInsuranceId = assetInsurance.AssetInsuranceId;
            insurancePremium.LineNum = i+1;
            insurancePremium.PremiumAmount = assetInsurance.PremiumAmount;
            insurancePremium.PremiumDueDate = MBTGeneratePremiums::getDueDate(assetInsurance.Insur
            insurancePremium.insert();
        }
    }
    if(_asynch)
        info("All premiums generated ... asynchronously");
    else
        info("All premiums generated ... synchronously");
}
```

5. Call the previous method from the main method by adding the following line of code to the existing main method:

```
public static void main(Args \_args)
{
    MBTAllPremiumGenerate::generateAllPremium(false);
}
```

6. **Create** a new **Action Menu** Item **MBTAllPremiumGenerate.**
7. **Change** the following **properties:**
    1. **Object Type: Class**
    2. **Object: MBTAllPremiumGenerate**
    3. **Label: Generate all premiums**
8. Search for form AssetTable in Application Explorer
9. Right-click on AssetTable and create an extension. It's OK if this is your second extension of the same form
10. A new folder will be created in Solution Explorer named Form Extensions with the extended form within it
11. Under Design > Action Pane > FixedAsset, there should be a button group MBTInsuranceButtonGroup (if not, add a new button group here)
12. Add a new menu button MBTAllPremiumGenerate under MBTInsuranceButtonGroup (you can drag it from the Solution Explorer)
13. **Change** the following **properties:**
    1. **Menu Item Type: Action**
    2. **Menu Item Name: MBTAllPremiumGenerate**
    3. **Text: Generate all premiums (sync)**

46.3 Task #2: Asynchronous processing

1. In the Solution Explorer, find the folder Form Extensions and the extension of AssetTable form within it, then open it in the design canvas.

2. Under Design > Action Pane > FixedAsset > MBTInsuranceButtonGroup, add a new button **AllPremiumAsync**.
3. Change the **Text** property value to All Premium Generate (Async) for the button **AllPremiumAsync.**
4. Expand the Event node of **AllPremiumAsync** button.
5. Right-click on the *onClicked* event and click on **Copy event handler method**.
6. Right-click on the project MB500TrainingProjectAsync and add a new class AssetTableFormEventHandler
7. Paste the event handler within the class. This method should be added in the class:

```
*/// \<summary\>*
*///*
*/// \</summary\>*
*/// \<param name="sender"\>\</param\>*
*/// \<param name="e"\>\</param\>*
[FormControlEventHandler(formControlStr(AssetTable, AllPremiumAsync), FormControlEventType::Clicke
public static void AllPremiumAsync_OnClicked(FormControl sender, FormControlEventArgs e)
{
}
```

8. Add the following piece of code within this method:

```
FormButtonControl callerButton = sender as FormButtonControl;
FormRun form = callerButton.formRun();
form.runAsync(classNum(MBTAllPremiumGenerate), "generateAllPremium", [true],
System.Threading.CancellationToken::None);
```

9. Save and build the project MB500TrainingProjectAsync

Output

Please note that the module 10 exercise is a prerequisite to testing this exercise

1. Navigate to Dynamics 365 Finance and Operations apps in your browser
2. Open **Procurement and sourcing** > **Setup** > **Vendors** > **Vendor Groups**
3. Add a new record if it's not already there:
    1. Vendor Group: 100
    2. Description: Insurance Vendor
4. Open **Procurement and sourcing** > **Setup** > **Procurement and sourcing parameters**
5. In the General tab, you will find a field called Insurance Vendor Group; select 100 in that field.
6. Open **Procurement and sourcing** > **Vendors** > **All vendors**
7. Add four new vendors with Vendor group 100.
    1. Vendor Code: INS0001, Name: Insurance vendor 001
    2. Vendor Code: INS0002, Name: Insurance vendor 002
    3. Vendor Code: INS0003, Name: Insurance vendor 003
    4. Vendor Code: INS0004, Name: Insurance vendor 004
8. Navigate to the menu **Organization Administration** and execute **Organization Administration** > **Periodic** > **Load Number Sequence**
9. At the end of the execution, you will get a message "Number sequence loaded"
10. Navigate to the module Fixed Assets and open **Fixed assets** > **Setup** > **Fixed assets parameters**
11. Go to **Number sequences** tab and you will find a new reference Asset Insurance ID. In the Number Sequence code field select Fixe_17
12. Save the record
13. Open the **Data Management** workspace and click on **Framework parameters**

14. In the Entity settings tab, click on the **Refresh entity list** button
15. In the **Data Management** *workspace*, open **Data Entities.** If the list is being refreshed you may need to wait a bit. Hit the refresh icon near the top right until the **MBTAssetInsuranceEntity** appears.
16. Search for the **MBTAssetInsuranceEntity** and select **Modify target mapping** in the action menu
17. Ensure the mapping between Staging and Target is correct
18. In the Data Management workspace, open Import
19. Enter the following data
    1. Group name: Asset Insurance Import
    2. Data project operation type: Import
    3. Click on **+Add file**
    4. Entity name: **MBTAssetInsuranceEntity**
    5. Source data format: CSV
    6. Default refresh type: Full push only
    7. Click on button **Upload and add** and select *assetInsurance.csv*, which is downloadable from https://aka.ms/mb500labresources
    8. Select the **Close** button
20. In the Selected entities line, click on **View map**
21. Click on the **Mapping details** tab and select the Auto-generated checkbox in the ASSETINSURANCEID line.
22. Click on Save and close this form
23. Click on the **Import** button in the action pane
24. Go back to the Data management workspace and monitor the status of the import process
25. Open the form **Fixed Assets**, under module **Fixed Assets > Fixed Assets**
26. Under the Fixed Asset action pane, find the group Asset Insurance
27. There are two buttons under Asset Insurance:
    1. Generate All Premium
    2. All Premium Generate (Async)

    Functionally both are doing the same job, i.e., generating premium data. But notice the technical differences in the job execution process.

Appendix

Project Import instructions:

- The lab resources may be found at https://aka.ms/mb500labresources
- To import a project, navigate to Visual Studio and select **Dynamics 365 > Import Project**
- In the File name field, select the file name with path of the .axpp file that you want to import
- For the first import, of MB500TrainingProject, select **New solution**. For subsequent ones, select the **Current solution** option.
- Select the **OK** button. You may safely override if prompted. test