

Table of Contents

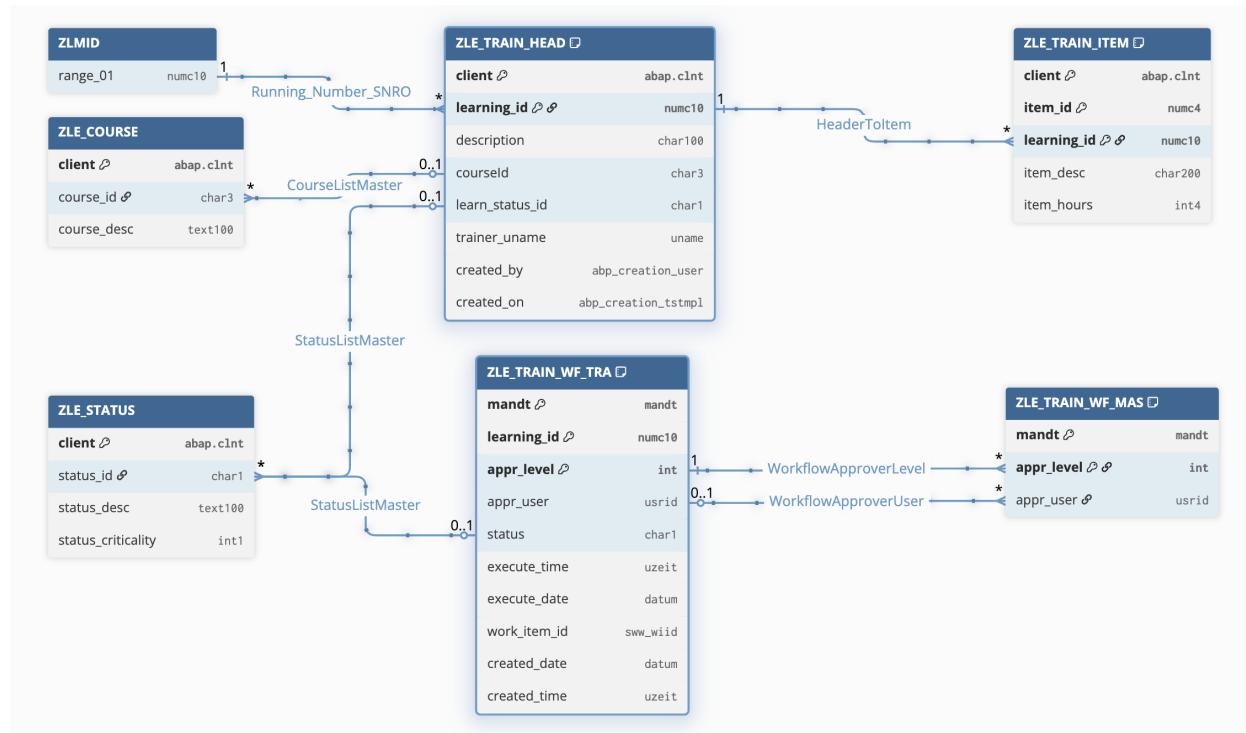
<i>Introduction</i>	2
1. <i>Design the Necessary Custom tables for the Master and Transaction Data</i>	2
1.1. <i>Creation of Transaction Tables</i>	3
1.2. <i>Creation of Master Tables</i>	3
1.3. <i>Creation of SNRO</i>	4
1.4. <i>Creation of Value Help Views</i>	5
2. <i>RAP Managed Scenario</i>	6
2.1. <i>Creation of Base Interface View Entities</i>	6
2.2. Creation of Consumption or Projection View Entities for the Base Interface View Entities:.....	7
2.3. <i>Creation of Meta Data Extensions</i>	8
2.4. <i>Creation of Behavior Definition</i>	9
2.5. <i>Creation of Behavior Implementation Class</i>	11
2.6. <i>Creation of Service Definition</i>	17
2.7. <i>Creation of Service Binding</i>	17
3. <i>Flexible workflow</i>	18
3.1. <i>Creation of Workflow Dictionary Tables</i>	18
3.2. <i>Creation of Structure used in workflow</i>	19
3.3. <i>Creation of Agent Rule</i>	19
3.3.1. Functional Module for Agent Rule	19
3.3.2. Agent Rule Creation in PFAC AC92000024	21
3.4. <i>Creation of Classes for Workflow</i>	24
3.4.1. Creation of Workflow Class and its event ZLE_CL_TRAIN_WORKFLOW	24
3.4.2. Creation of Class Definition Application base ZCL_TRAIN_DEF_APPL_BASE	34
3.4.3. Creation of Class Run Application base ZCL_TRAIN_RUN_APPL_BASE.....	35
3.5. <i>Creation of Custom Decision Task to be used in Workflow TS92000178</i>	39
3.6. <i>Creation of Workflow Template WS92000123</i>	43
3.7. <i>Configuration of Flexible Workflow</i>	57
4. <i>Integrate the Workflow in RAP Application</i>	61
4.1. <i>Behavior Definition</i>	61
4.2. <i>Meta Data Extension</i>	61
4.3. <i>Class Implementation</i>	62
4.4. <i>Behavior Definition Consumption View</i>	63
5. <i>BAS: Create and Deploy Fiori Application</i>	63
5.1. <i>Creation of Fiori Application</i>	63
5.2. <i>Deploy Fiori Application</i>	65
6. <i>Add Fiori Application in Launchpad</i>	66
6.1. <i>Create Catalogue</i>	66
6.2. <i>Create Group</i>	67
7. <i>Visualization Configuration for Tasks</i>	70
8. <i>Task Decision Button Configuration</i>	72
9. <i>Testing the Application</i>	74
9.1. <i>Approve Scenario</i>	74
9.2. <i>Reject Scenario</i>	80

Introduction

Let us see how to develop an ABAP RAP application using Managed Scenario for Custom Sample Training Management process.

- 1.Create the Necessary Custom Dictionary Objects for the Master Data, Transaction Data and logs for Workflow
- 2.Create a Custom SAP RAP Application using Managed Scenario
- 3.Create a Custom Flexible Workflow and its related Agent rules, Tasks and Classes
- 4.Integration of Workflow with the RAP Application

1. Design the Necessary Custom tables for the Master and Transaction Data



ZLE_TRAIN_HEAD - Header Transaction Table

ZLE_TRAIN_ITEM - Item Transaction Table

ZLE_COURSE - Course List Master Table

ZLE_STATUS - Status List Master Table

ZLE_TRAIN_WF_MAS - Workflow Approver List Master Table

ZLE_TRAIN_WF_TRA - Workflow Transaction log Table for each learning ID

1.1. Creation of Transaction Tables

Create Dictionary Table: Header Table: **ZLE TRAIN HEAD**

```

@EndUserText.label : 'Training Header Table'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #RESTRICTED
define table zle_train_head {
  key client    : abap.clnt not null;
  key learning_id : numc10 not null;
  description   : abap.char(100);
  course_id    : abap.char(3);
  learn_status_id : abap.char(1);
  trainer_uname : usrid;
  created_by   : abp_creation_user;
  created_on    : abp_creation_tstmp;
}

```

Create Dictionary Table: Header Table: **ZLE TRAIN ITEM**

```

@EndUserText.label : 'Training Item Table'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #RESTRICTED
define table zle_train_item {
  key client    : abap.cln1 not null;
  key item_id   : numc4 not null;
  key learning_id : numc10 not null;
  item_desc    : abap.char(200);
  item_hours   : abap.int4;
}

```

1.2. Creation of Master Tables

Create Dictionary Table: Master Course Table: **ZLE_COURSE**

```
@EndUserText.label : 'Course Master Table'  
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE  
@AbapCatalog.tableCategory : #TRANSPARENT  
@AbapCatalog.deliveryClass : #A  
@AbapCatalog.dataMaintenance : #ALLOWED  
define table zle_course {  
    key client : abap.clnt not null;  
    key course_id : char3 not null;  
    course_desc : text100 not null;  
}
```

Add the Following Course List Data

Search in Table	ZLE_COURSE	Course Master Table										
Number of Hits	4											
Runtime	0	Maximum No. of Hits 500										
Insert Column <input type="text"/>                 												
<table border="1"> <tr> <td><input checked="" type="checkbox"/></td> <td>3-Byte fi... Text</td> </tr> <tr> <td>ABA</td> <td>ABAP</td> </tr> <tr> <td>AMP</td> <td>ABAP Managed Database Procedures</td> </tr> <tr> <td>CDS</td> <td>Core Data Service</td> </tr> <tr> <td>FIO</td> <td>Fiori and UI5</td> </tr> </table>			<input checked="" type="checkbox"/>	3-Byte fi... Text	ABA	ABAP	AMP	ABAP Managed Database Procedures	CDS	Core Data Service	FIO	Fiori and UI5
<input checked="" type="checkbox"/>	3-Byte fi... Text											
ABA	ABAP											
AMP	ABAP Managed Database Procedures											
CDS	Core Data Service											
FIO	Fiori and UI5											

Create Dictionary Table: Master Course Table: **ZLE STATUS**

@EndUserText.label : 'Training Status'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE

```

@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #ALLOWED
define table zle_status {
  key client      : abap.clnt not null;
  key status_id   : char1 not null;
  status_desc     : text100 not null;
  status_criticality : abap.int1;
}

```

Add the following Status Data

Search in Table	ZLE_STATUS	Training Status
Number of Hits	4	
Runtime	0	Maximum No. of Hits 500
Insert Column		
Single-Character Fl... Text	STATUS_CRITICALI...	
A	Approved	3
D	Draft	0
I	In progress	2
R	Rejected	1

1.3. *Creation of SNRO*

Creation Of SNRO Object: ZLMID

Number Range Object ZLMID: Display

ShortTxt	Training ID
Long Txt	Training ID
Pack.	Z1537_TRAIN

Intervals **Customizing**

Number Length Domain	NUMC10
% Warning	70.0

To-business year flag
 No rolling
 Prefix

Maintain Interval :

Edit Intervals: Training ID, Object ZLMID				
Number Range No.	From No.	To Number	NR Status	External
01	1000000000	1999999999	0	<input type="checkbox"/>

1.4. Creation of Value Help Views

Creation of Value Help Views for the Master Tables to be used as F4 Help:

Value Help view Entity for Course table ZLE_COURSE: ZLE_COURSE_VH

```

@AbapCatalog.viewEnhancementCategory: [#NONE]
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Training Course Value Help'
@Metadata.ignorePropagatedAnnotations: true
@ObjectModel.usageType: {
    serviceQuality: #X,
    sizeCategory: #S,
    dataClass: #MIXED
}
define view entity ZLE_COURSE_VH
as select from zle_course
{
    @EndUserText.label: 'Id'
    key course_id as CourseId,
    @EndUserText.label: 'Description'
    course_desc as CourseDesc
}

```

Value Help view Entity for Course table ZLE_STATUS: ZLE_STATUS_VH

```

@AbapCatalog.viewEnhancementCategory: [#NONE]
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Status Value Help'
@Metadata.ignorePropagatedAnnotations: true
@ObjectModel.usageType: {
    serviceQuality: #X,
    sizeCategory: #S,
    dataClass: #MIXED
}
define view entity ZLE_STATUS_VH
as select from zle_status
{
    @EndUserText.label: 'Id'
    key status_id as StatusId,
    @EndUserText.label: 'Status'
    status_desc as StatusDesc,
    status_criticality as StatusCriticality
}

```

Value Help view Entity for Trainer Master: ZLE_TRAINER_VH

We will form this value help view based on the HR available PERNR, using table PA0105 and PA0001

```

@AbapCatalog.viewEnhancementCategory: [#NONE]
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Training : Trainer Value help'
@Metadata.ignorePropagatedAnnotations: true
@ObjectModel.usageType: {
    serviceQuality: #X,
    sizeCategory: #S,
    dataClass: #MIXED
}
define view entity ZLE_TRAINER_VH
as select from pa0105 as a
    inner join pa0001 as b on b.pernr = a.pernr
        and b.endda >= $session.system_date
        and b.begda <= $session.system_date
{
    key a.usrid as UserId,
    a.pernr as PersonnalNumber,
    b.ename as Name,
    b.plans as JobPosition
}

```

```

}
where
  a.subty = '0001'
  and a.endda >= $session.system_date
  and a.begda <= $session.system_date

```

2. RAP Managed Scenario

Let us develop the RAP Managed Scenario Application that allow employee to raise a request. In this Application the following are used

- **Early Numbering** for Header and Item to fetch the key number from SNRO object
- **Determinations** to set default value for status on create
- **Validations** to set the mandatory fields for Header and Item
- **Instance Features** to control the visibility of the Update, Delete and Send Approval buttons
- **Custom Action Button** for Send Approval to trigger workflow

2.1. Creation of Base Interface View Entities

Root View entity for table ZLE_TRAIN_HEAD: **ZBI_TRAIN_HEAD**

```

@AbapCatalog.viewEnhancementCategory: [#NONE]
@AccessControl.authorizationCheck: #NOT_REQUIRED
@endUserText.label: 'BI : Training Header'
@Metadata.ignorePropagatedAnnotations: true
@ObjectModel.usageType: {
  serviceQuality: #X,
  sizeCategory: #S,
  dataClass: #MIXED
}
define root view entity zbi_train_head
as select from zle_train_head
//Compositions Links
composition[1..*] of zbi_train_item as _TrainItem

//Associations Links
association[1..1] to ZLE_COURSE_VH as _CourseMaster on $projection.CourseId = CourseMaster.CourseId
association[1..1] to ZLE_STATUS_VH as _StatusMaster on $projection.LearnStatusId = StatusMaster.StatusId
association[1..1] to ZLE_TRAINER_VH as _UserMaster on $projection.TrainerName = UserMaster.UserId
{
  key learning_id as LearningId,
  description as Description,
  course_id as CourseId,
  learn_status_id as LearnStatusId,
  StatusMaster.StatusCriticality as StatusCriticality,
  trainer_uname as TrainerUname,
  @Semantics.user.createdBy: true
  created_by as CreatedBy,
  @Semantics.systemDateTime.createdAt: true
  created_on as CreatedOn,
}

//Master Texts
CourseMaster.CourseDesc as CourseDescription,
StatusMaster.StatusDesc as StatusDescription,
UserMaster.Name as TrainerName,
TrainItem,
CourseMaster,
StatusMaster,
UserMaster
}
```

View entity for table ZLE_TRAIN_ITEM: ZBI_TRAIN_ITEM

```

@AbapCatalog.viewEnhancementCategory: [#NONE]
@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'BI : Training Item'
@Metadata.ignorePropagatedAnnotations: true
@ObjectModel.usageType: {
    serviceQuality: #X,
    sizeCategory: #S,
    dataClass: #MIXED
}
define view entity zbi_train_item
as select from zle_train_item
///Association Links
association to parent zbi_train_head as _TrainHead on $projection.LearningId = _TrainHead.LearningId
{
    key item_id as ItemId,
    key learning_id as LearningId,
    item_desc as ItemDesc,
    item_hours as ItemHours,
    //Association
    _TrainHead
}

```

2.2. Creation of Consumption or Projection View Entities for the Base Interface View Entities:

Root Consumption View entity ZBI_TRAIN_HEAD: ZCO_TRAIN_HEAD

```

@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Training : Head Consumption View'
@Metadata.ignorePropagatedAnnotations: true
@Metadata.allowExtensions: true
define root view entity ZCO_TRAIN_HEAD
provider contract transactional_query
as projection on zbi_train_head
{
    key LearningId,
    Description,
    @ObjectModel.text.element: ['CourseDescription']
    CourseId,
    @ObjectModel.text.element: ['StatusDescription']
    LearnStatusId,
    StatusCriticality,
    @ObjectModel.text.element: ['TrainerName']
    TrainerUname,
    CreatedBy,
    CreatedOn,
    CourseDescription,
    StatusDescription,
    TrainerName,
    /* Associations */
    _TrainItem :redirected to composition child ZCO_TRAIN_ITEM,
    _CourseMaster,
    _StatusMaster,
    _UserMaster
}

```

Consumption View entity ZBI_TRAIN_ITEM: ZCO_TRAIN_ITEM

```

@AccessControl.authorizationCheck: #NOT_REQUIRED
@EndUserText.label: 'Training : Item Consumption View'
@Metadata.ignorePropagatedAnnotations: true
@Metadata.allowExtensions: true

```

```
define view entity ZCO_TRAIN_ITEM as projection on zbi_train_item
{
    key ItemId,
    key LearningId,
    ItemDesc,
    ItemHours,
    /* Associations */
    _TrainHead :redirected to parent ZCO_TRAIN_HEAD
}
```

2.3. Creation of Meta Data Extensions

Meta Data Extensions are created for Consumption View Entities here let us create for View entity ZCO_TRAIN_HEAD: **ZMDE_TRAIN_HEAD**

```
@Metadata.layer: #CORE
@UI: { headerInfo: {
    typeName: 'Training',
    typeNamePlural: 'Trainings',
    title: { type:#STANDARD,
        label:'Training',
        value:'LearningId' }

},
presentationVariant: [ { sortOrder: [ { by: 'LearningId', direction: '#DESC' } ] } ]
@Search.searchable: true
annotate entity ZCO_TRAIN_HEAD with
{
//UI Facets
@UI.facet: [ { id:'Learning', purpose : #STANDARD, type :#IDENTIFICATION_REFERENCE, position : 10, label: 'Learning'},
{ id:'Items', purpose : #STANDARD, type :#LINEITEM_REFERENCE, position : 20, label: 'Items', targetElement: '_TrainItem'} ]

@Search.defaultSearchElement: true
@Search.fuzzinessThreshold: 0.5
@UI.identification: [ { position: 10 } ]
@UI.lineItem: [ { position: 10 } ]
@endUserText.label: 'Learning Id'
LearningId;

@Search.defaultSearchElement: true
@Search.fuzzinessThreshold: 0.7
@UI.identification: [ { position: 20 } ]
@UI.lineItem: [ { position: 20 } ]
@endUserText.label: 'Description'
Description;

@UI.identification: [ { position: 30 } ]
@UI.lineItem: [ { position: 30 } ]
@endUserText.label: 'Course'
@Consumption.valueHelpDefinition: [ { entity:{ name: 'ZLE_COURSE_VH', element: 'CourseId' } } ]
@UI.selectionField: [ { position : 30 } ]
Course;
@UI.identification: [ { position: 40, criticality: 'StatusCriticality' } ]
@UI.lineItem: [ { position: 40, criticality: 'StatusCriticality' },
    { type : #FOR_ACTION, dataAction: 'sendApproval', label: 'Send Approval' } ]
@endUserText.label: 'Learning Status'
@Consumption.valueHelpDefinition: [ { entity:{ name: 'ZLE_STATUS_VH', element: 'StatusId' } } ]
@UI.selectionField: [ { position : 40 } ]

LearnStatusId;
@UI.identification: [ { position: 50 } ]
@UI.lineItem: [ { position: 50 } ]
@Consumption.valueHelpDefinition: [ { entity:{ name: 'ZLE_TRAINER_VH', element: 'UserId' } } ]
@endUserText.label: 'Trainer Name'
TrainerUname;
@UI.hidden: true
CreatedBy;
@UI.hidden: true
CreatedOn;
```

```
}
```

Meta Data Extensions for View entity ZCO_TRAIN_ITEM: **ZMDE_TRAIN_ITEM**

```
@Metadata.layer: #CORE
@UI:{ headerInfo:{  
    typeName: 'Item',  
    typeNamePlural: 'Items',  
    title: { type:#STANDARD,  
        label:'Item' ,  
        value:'ItemId' }  
  
},  
presentationVariant: [{ sortOrder: [{ by: 'LearningId',direction: #DESC }] }]  
annotate entity ZCO_TRAIN_ITEM with  
{  
    @UI.facet: [{ id:'Item', purpose : #STANDARD, type :#IDENTIFICATION_REFERENCE, position : 10, label: 'Item'}]  
    @UI.identification: [ { position: 10 } ]  
    @UI.lineItem: [ { position: 10 } ]  
    @EndUserText.label: 'Item Id'  
    ItemId;  
    @UI.hidden: true  
    LearningId;  
    @UI.identification: [ { position: 20 } ]  
    @UI.lineItem: [ { position: 20 } ]  
    @EndUserText.label: 'Description'  
    ItemDesc;  
    @UI.identification: [ { position: 30 } ]  
    @UI.lineItem: [ { position: 30 } ]  
    @EndUserText.label: 'Hours'  
    ItemHours;  
}
```

2.4. Creation of Behavior Definition

Behavior Definition for the Base Interface Root view Entity **ZBI_TRAIN_HEAD** **Implementation Type: Managed**

Here we need to declare the Early numbering, Determinations, Validations, Action the same methods will be implemented in Behavior implementation

```
managed implementation in class zbp_bi_train_head unique;  
strict ( 2 );  
  
define behavior for zbi_train_head alias train_head  
persistent table ZLE_TRAIN_HEAD  
lock master  
authorization master ( instance )  
//etag master <field_name>  
early numbering  
{  
    //Read only Fields  
    field ( readonly ) LearningId,LearnStatusId,CreatedBy,CreatedOn;  
    //Mandatory Fields  
    field ( mandatory ) CourseId,TrainerUname;  
  
    //Determinations  
    determination setStatusDraft on modify { create; }  
  
    //Validations  
    validation mandatoryCourseId on save { create; field CourseId; }  
    validation mandatoryTrainerName on save { create; field TrainerUname; }  
  
    //Actions  
    action ( features : instance ) sendApproval result [1] $self;
```

```

create;
update (features : instance );
delete (features : instance );
association _TrainItem { create (features : instance ); }

//Mappings Fields for the Header Table
mapping for ZLE_TRAIN_HEAD {
    LearningId = learning_id;
    Description = description;
    CourseId = course_id;
    LearnStatusId = learn_status_id;
    TrainerUname = trainer_uname;
    CreatedBy = created_by;
    CreatedOn = created_on;
}

define behavior for zbi_train_item alias train_item
persistent table ZLE_TRAIN_ITEM
lock dependent by _TrainHead
authorization dependent by _TrainHead
//etag master <field_name>
early numbering
{
    //Read Only Fields
    field ( readonly ) ItemId,LearningId;
    //Mandatory Fields
    field ( mandatory ) ItemDesc,ItemHours;

    //Validations
    validation MandatoryItemDesc on save { create; field ItemDesc; }
    validation MandatoryItemHours on save { create; field ItemHours; }

    update(features : instance );
    delete(features : instance );
    association _TrainHead;
}

//Mappings Fields for the Item Table
mapping for ZLE_TRAIN_ITEM {
    ItemId = item_id;
    LearningId = learning_id;
    ItemDesc = item_desc;
    ItemHours = item_hours;
}
}

```

Behavior Definition for the Consumption Root view Entity **ZCO_TRAIN_HEAD**

Implementation Type: Projection

projection;
strict (2);

```

define behavior for ZCO_TRAIN_HEAD alias train_head
{
    use create;
    use update;
    use delete;
    use action sendApproval;

    use association _TrainItem { create; }

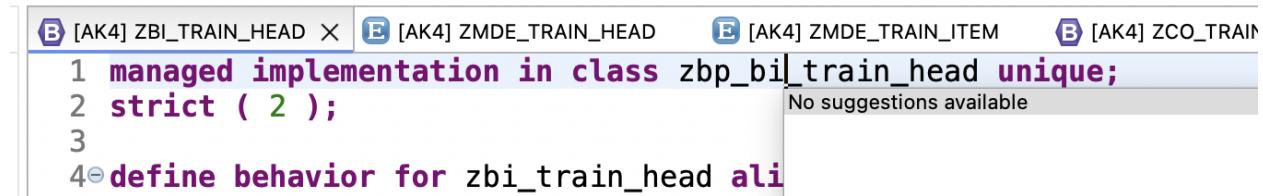
define behavior for ZCO_TRAIN_ITEM alias train_item
{
    use update;
    use delete;
}

```

```
use association _TrainHead;
{
```

2.5. Creation of Behavior Implementation Class

Here I have created one class for both Header and item but we can also create separate classes accordingly. In eclipse the shortcut will be place the cursor on the behavior definition and give Ctrl + 1 which will show you suggestion to create class.



The screenshot shows the Eclipse IDE interface with several tabs at the top: [AK4] ZBI_TRAIN_HEAD, [AK4] ZMDE_TRAIN_HEAD, [AK4] ZMDE_TRAIN_ITEM, and [AK4] ZCO_TRAIN. A code editor window displays the following ABAP code:

```
1 managed implementation in class zbp_bil_train_head unique;
2 strict ( 2 );
3
4 define behavior for zbi_train_head ali
```

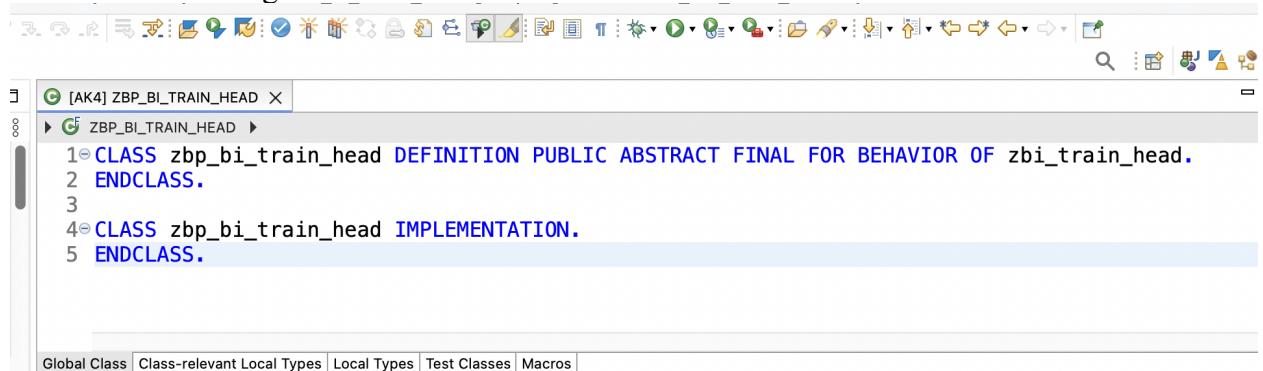
A tooltip "No suggestions available" is visible on the right side of the code editor.

Class name: **ZBP_BI_TRAIN_HEAD**

Following which are declared in the Behavior definition is implemented here:

- >**Mandatory and Read Only Fields for Header**
- >**Early Numbering for Header and Item**
- >**Determinations to set default values**
- >**Instance Features to Enable the Buttons based on Status**
- >**Custom Action Button to Trigger workflow**

Global Class: Auto generated

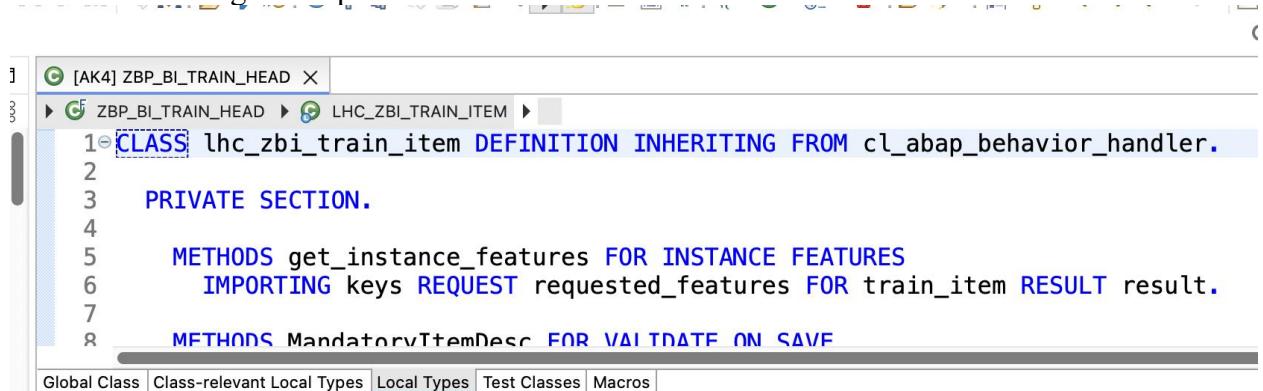


The screenshot shows the Eclipse IDE interface with the Global Class tab selected. The code editor displays the following ABAP code:

```
1 CLASS zbp_bil_train_head DEFINITION PUBLIC ABSTRACT FINAL FOR BEHAVIOR OF zbi_train_head.
2 ENDCLASS.
3
4 CLASS zbp_bil_train_head IMPLEMENTATION.
5 ENDCLASS.
```

The status bar at the bottom shows tabs for Global Class, Class-relevant Local Types, Local Types, Test Classes, and Macros.

Local Types: Declarations will be auto created but we need add code to implementation, this is where the core logics are present



The screenshot shows the Eclipse IDE interface with the Local Types tab selected. The code editor displays the following ABAP code:

```
1 CLASS lhc_zbi_train_item DEFINITION INHERITING FROM cl_abap_behavior_handler.
2
3 PRIVATE SECTION.
4
5 METHODS get_instance_features FOR INSTANCE FEATURES
6   IMPORTING keys REQUEST requested_features FOR train_item RESULT result.
7
8 METHODS MandatoryItemDesc FOR VALIDATE ON SAVE
```

The status bar at the bottom shows tabs for Global Class, Class-relevant Local Types, Local Types, Test Classes, and Macros.

Code for the Local type class:

```
*****
* Definition for the Item Entity *
*****
CLASS lhc_zbi_train_item DEFINITION INHERITING FROM cl_abap_behavior_handler.
PRIVATE SECTION.

METHODS get_instance_features FOR INSTANCE FEATURES
IMPORTING keys REQUEST requested_features FOR train_item RESULT result.

METHODS MandatoryItemDesc FOR VALIDATE ON SAVE
IMPORTING keys FOR train_item~MandatoryItemDesc.

METHODS MandatoryItemHours FOR VALIDATE ON SAVE
IMPORTING keys FOR train_item~MandatoryItemHours.

ENDCLASS.
*****
* Implementation for the Item Entity *
*****
CLASS lhc_zbi_train_item IMPLEMENTATION.

* Method 1: Instance Feature to Enable Update and Delete Button for Item Entity
METHOD get_instance_features.

READ ENTITIES OF zbi_train_head IN LOCAL MODE
ENTITY train_head BY \_TrainItem
FIELDS ( learningid itemid )
WITH CORRESPONDING #( keys )
RESULT DATA(lt_items).

READ ENTITY IN LOCAL MODE zbi_train_item
BY \_TrainHead
FIELDS ( learningid learnstatusid )
WITH CORRESPONDING #( lt_items )
RESULT DATA(lt_heads).

result = VALUE #( FOR ls_head IN lt_heads
FOR ls_item IN lt_items WHERE ( learningid = ls_head-learningid )
( %key = ls_item-%key
%features-%update = COND #( WHEN ls_head-learnstatusid = 'D'
THEN if_abap_behv=>fc-o-enabled
ELSE if_abap_behv=>fc-o-disabled )

%features-%delete = COND #( WHEN ls_head-learnstatusid = 'D'
THEN if_abap_behv=>fc-o-enabled
ELSE if_abap_behv=>fc-o-disabled ) ) ).

ENDMETHOD.

* Method 2: Mandatory Check for Item Description field in Item Entity
METHOD MandatoryItemDesc.
READ ENTITY IN LOCAL MODE zbi_train_item
FIELDS ( itemdesc )
WITH CORRESPONDING #( keys )
RESULT DATA(lt_item).
LOOP AT lt_item ASSIGNING FIELD-SYMBOL(<fs_item>).
IF <fs_item>-itemdesc IS INITIAL.
APPEND VALUE #( %tky = <fs_item>-%tky ) TO failed-train_item.
APPEND VALUE #( %tky = <fs_item>-%tky
%element-itemdesc = if_abap_behv=>mk-on
%msg = new_message_with_text( severity = if_abap_behv_message=>severity-error
text = 'Description is Mandatory' ) ) TO reported-train_item.

ENDIF.
```

```

ENDLOOP.
ENDMETHOD.

* Method 3: Mandatory Check for Item Hours field in Item Entity
METHOD MandatoryItemHours.
READ ENTITY IN LOCAL MODE zbi_train_item
FIELDS ( itemdesc )
WITH CORRESPONDING #( keys )
RESULT DATA(lt_item).
LOOP AT lt_item ASSIGNING FIELD-SYMBOL(<fs_item>).
IF <fs_item>-itemhours IS INITIAL.
APPEND VALUE #( %tky = <fs_item>-%tky ) TO failed-train_item.
APPEND VALUE #( %tky      = <fs_item>-%tky
               %element-itemhours = if abap behv=>mk-on
               %msg                = new_message_with_text( severity = if abap behv_message=>severity-error
                                                 text   = 'Hours is Mandatory' ) ) TO reported-train_item.
ENDIF.
ENDLOOP.
ENDMETHOD.

ENDCLASS.

*****
* Definition for the Header Entity *
*****
CLASS lhc_train_head DEFINITION INHERITING FROM cl_abap_behavior_handler.
PRIVATE SECTION.

METHODS get_instance_authorizations FOR INSTANCE AUTHORIZATION
IMPORTING keys REQUEST requested_authorizations FOR train_head RESULT result.

METHODS get_instance_features FOR INSTANCE FEATURES
IMPORTING keys REQUEST requested_features FOR train_head RESULT result.

METHODS setstatusdraft FOR DETERMINE ON MODIFY
IMPORTING keys FOR train_head~setstatusdraft.

METHODS mandatorycourseid FOR VALIDATE ON SAVE
IMPORTING keys FOR train_head~mandatorycourseid.

METHODS mandatorytrainername FOR VALIDATE ON SAVE
IMPORTING keys FOR train_head~mandatorytrainername.

METHODS sendapproval FOR MODIFY
IMPORTING keys FOR ACTION train_head~sendapproval RESULT result.

METHODS earlynumbering_cba_trainitem FOR NUMBERING
IMPORTING entities FOR CREATE train_head~trainitem.

METHODS earlynumbering_create FOR NUMBERING
IMPORTING entities FOR CREATE train_head.

ENDCLASS.

*****
* Implementation for the Header Entity *
*****
CLASS lhc_train_head IMPLEMENTATION.

* Method 1:Instance Authorizations -> In this example it is not used
METHOD get_instance_authorizations.
ENDMETHOD.

* Method 2:Early Numbering for Training Header
METHOD earlynumbering_create.

"Method to Handle Header Number Range
DATA(lt_entites)=entities.
DELETE lt_entites WHERE learningid IS NOT INITIAL.

```

```

"Get the Number Range
TRY.
cl_numberrange_runtime=>number_get(
  EXPORTING
    nr_range_nr      = '01'
    object          = 'ZLMID'
    quantity        = CONV #( lines( lt_entites ) )
  IMPORTING
    number          = DATA(lv_latest_num)
    returncode      = DATA(lv_ret_code)
    returned_quantity = DATA(lv_ret_quan)
).
CATCH cx_nr_object_not_found INTO DATA(lo_exp_object).
CATCH cx_number_ranges INTO DATA(lo_exp_number).
"Exception or Error Handling
LOOP AT lt_entites INTO DATA(ls_entites).
  APPEND VALUE #( %cid = ls_entites-%cid
    %key = ls_entites-%key )
    TO failed-train_head.

  APPEND VALUE #( %cid = ls_entites-%cid
    %key = ls_entites-%key
    %msg = lo_exp_number )
    TO reported-train_head.
ENDLOOP.
EXIT.
ENDTRY.
ASSERT lv_ret_quan = lines( lt_entites ).
DATA(lv_curr_num) = CONV zlm_head-learning_id( lv_latest_num - lv_ret_quan ).
```

```

lv_curr_num = lv_curr_num + 1.
CLEAR ls_entites.
LOOP AT lt_entites INTO ls_entites.
"Number Range Assignment
  APPEND VALUE #( %cid = ls_entites-%cid
    learningid = lv_curr_num )
    TO mapped-train_head.
ENDLOOP.
ENDMETHOD.
```

* Method 3: Early Numbering for Item
METHOD earlynumbering_cba_Trainitem.

```

DATA : lv_max_item TYPE zle_train_item-item_id.

READ ENTITIES OF zbi_train_head IN LOCAL MODE
ENTITY train_head BY \_TrainItem
FROM CORRESPONDING #( entities )
LINK DATA(lt_link).

LOOP AT entities ASSIGNING FIELD-SYMBOL(<ls_group_entity>) GROUP BY <ls_group_entity>-learningid.
"Link Data will be empty for creating first record
lv_max_item = REDUCE #( INIT lv_max = CONV zle_train_item-item_id( '0' )
  FOR ls_link IN lt_link USING KEY entity WHERE ( source-learningid = <ls_group_entity>-learningid )
  NEXT lv_max = COND zle_train_item-item_id( WHEN lv_max < ls_link-target-itemid
    THEN ls_link-target-itemid
    ELSE lv_max ) ).

"Increment to 10 and Assignment to item so that the item numbering will be 10,20,30 etc..
LOOP AT entities ASSIGNING FIELD-SYMBOL(<ls_entities>)
  USING KEY entity
  WHERE learningid = <ls_group_entity>-learningid.
LOOP AT <ls_entities>-%target ASSIGNING FIELD-SYMBOL(<ls_item>).
  APPEND CORRESPONDING #( <ls_item> ) TO mapped-train_item ASSIGNING FIELD-SYMBOL(<ls_new_item>).
  IF <ls_item>-itemid IS INITIAL.
    lv_max_item += 10.
    <ls_new_item>-itemid = lv_max_item.
  ENDIF.
ENDLOOP.
ENDLOOP.
ENDLOOP.
```

ENDMETHOD.

* Method 4:Instance features to control update, Send Approval, Delete Buttons

```
METHOD get_instance_features.
  READ ENTITIES OF zbi_train_head IN LOCAL MODE
    ENTITY train_head
    FIELDS ( learnstatusid )
    WITH CORRESPONDING #( keys )
    RESULT DATA(lt_head).

  result = VALUE #(
    FOR ls_head IN lt_head (
      %tky = ls_head-%tky
      %features-%update = COND #(
        WHEN ls_head-learnstatusid = 'D'
        THEN if_abap_behv=>fc-o-enabled
        ELSE if_abap_behv=>fc-o-disabled )
      %features-%action-sendApproval = COND #(
        WHEN ls_head-learnstatusid = 'D'
        THEN if_abap_behv=>fc-o-enabled
        ELSE if_abap_behv=>fc-o-disabled )
      %features-%delete = COND #(
        WHEN ls_head-learnstatusid = 'D'
        THEN if_abap_behv=>fc-o-enabled
        ELSE if_abap_behv=>fc-o-disabled )
      %features-%assoc-TrainItem = COND #(
        WHEN ls_head-learnstatusid = 'D'
        THEN if_abap_behv=>fc-o-enabled
        ELSE if_abap_behv=>fc-o-disabled ) )
    ))).
```

ENDMETHOD.

* Method 5:Set Status as Draft(D) for newly created Records by Default

```
METHOD setStatusDraft.
```

```
READ ENTITIES OF zbi_train_head IN LOCAL MODE
  ENTITY train_head
  FIELDS ( learnstatusid )
  WITH CORRESPONDING #( keys )
  RESULT DATA(lt_head).
```

```
DELETE lt_head WHERE learnstatusid IS NOT INITIAL.
CHECK lt_head IS NOT INITIAL.
```

```
MODIFY ENTITIES OF zbi_train_head IN LOCAL MODE
  ENTITY train_head
  UPDATE FIELDS ( learnstatusid )
  WITH VALUE #(
    FOR ls_head IN lt_head
    (%tky = ls_head-%tky
    learnstatusid = 'D') ).
```

ENDMETHOD.

* Method 6:Validation and Mandatory check for Course ID
METHOD mandatoryCourseId.

```
READ ENTITIES OF zbi_train_head IN LOCAL MODE
  ENTITY train_head
  FIELDS ( courseid )
  WITH CORRESPONDING #( keys )
  RESULT DATA(lt_heads).
```

```
SELECT course_id FROM zle_course INTO TABLE @DATA(lt_course_master).
```

```
LOOP AT lt_heads ASSIGNING FIELD-SYMBOL(<fs_head>).
```

""Invalidate State Messages

```
APPEND VALUE #(%tky = <fs_head>-%tky
%state_area = 'MANDATORY_COURSEID') TO reported-train_head.
```

```
IF <fs_head>-courseid IS INITIAL.
```

```
APPEND VALUE #(%tky = <fs_head>-%tky) TO failed-train_head.
```

```
APPEND VALUE #(%tky = <fs_head>-%tky
```

```
%state_area = 'MANDATORY_COURSEID'
```

```
%element-courseid = if_abap_behv=>mk-on
```

```
%omsg = new_message_with_text(severity = if_abap_behv message=>severity-error
```

```
text = 'Course ID is Mandatory' ) ) TO reported-train_head.
```

```

ELSEIF <fs_head>-courseid IS NOT INITIAL AND NOT line_exists( lt_course_master[ course_id = <fs_head>-courseid ]).
APPEND VALUE #( %tky      = <fs_head>-%tky
               %state_area = 'MANDATORY_COURSEID'
               %element-courseid = if_abap_behv=>mk-on
               %msg        = new_message_with_text( severity = if_abap_behv_message=>severity-error
                                         text   = 'Invalid Course ID' ) ) TO reported-train_head.
ENDIF.

ENDLOOP.

ENDMETHOD.

* Method 7:Validation and Mandatory Check for Trainer ID
METHOD mandatoryTrainerName.
READ ENTITIES OF zbi_train_head IN LOCAL MODE
ENTITY train_head
FIELDS ( traineruname )
WITH CORRESPONDING #( keys )
RESULT DATA(lt_heads).

SELECT * FROM zle_trainer_vh INTO TABLE @DATA(lt_users).

LOOP AT lt_heads ASSIGNING FIELD-SYMBOL(<fs_head>).
  ""Invalidate State Messages
APPEND VALUE #( %tky      = <fs_head>-%tky
               %state_area = 'MANDATORY_USERNAME' ) TO reported-train_head.

IF <fs_head>-traineruname IS INITIAL.
  APPEND VALUE #( %tky = <fs_head>-%tky ) TO failed-train_head.
APPEND VALUE #( %tky = <fs_head>-%tky
               %state_area = 'MANDATORY_USERNAME'
               %element-traineruname = if_abap_behv=>mk-on
               %msg = new_message_with_text( severity = if_abap_behv_message=>severity-error
                                         text   = 'Trainer User Name is Mandatory' ) ) TO reported-train_head.

ELSEIF <fs_head>-traineruname IS NOT INITIAL AND NOT line_exists( lt_users[ userid = <fs_head>-traineruname ] ).
APPEND VALUE #( %tky = <fs_head>-%tky
               %state_area = 'MANDATORY_USERNAME'
               %element-traineruname = if_abap_behv=>mk-on
               %msg = new_message_with_text( severity = if_abap_behv_message=>severity-error
                                         text   = 'Invalid Trainer User Name' ) ) TO reported-train_head.
ENDIF.

ENDLOOP.

ENDMETHOD.

* Method 8:Action to send the approval -> Workflow is Triggered
METHOD sendApproval.
  ""->Read the Selected Data
READ ENTITIES OF zbi_train_head IN LOCAL MODE
ENTITY train_head BY \_TrainItem
ALL FIELDS
WITH CORRESPONDING #( keys )
RESULT DATA(lt_items).

READ ENTITY IN LOCAL MODE zbi_train_item
BY \_TrainHead
FIELDS ( learningid learnstatusid )
WITH CORRESPONDING #( lt_items )
RESULT DATA(lt_heads).

DATA(ls_head)=VALUE #( lt_heads[ 1 ] OPTIONAL ).
  ""->Only Learning ID is Enough for us in the structure
DATA(ls_learning)=VALUE zle_train_head( learning_id = ls_head-LearningId ).

  ""->Trigger Workflow -> *DO NOT USE COMMIT in any place as it is not allowed*
DATA :lo_wf TYPE REF TO zle_cl_train_workflow.
CREATE OBJECT lo_wf
EXPORTING

```

```

lv_key_number = ls_learning-learning_id.

CALL METHOD lo_wf->trigger_start_wf
EXPORTING
learning_det = ls_learning.

""->Update the status back in entity
MODIFY ENTITIES OF zbi_train_head IN LOCAL MODE
ENTITY train_head
UPDATE FIELDS ( LearnStatusId )
WITH VALUE #( FOR ls_keys IN keys ( %tky = ls_keys-%tky
LearnStatusId = 'T' )).

READ ENTITIES OF zbi_train_head IN LOCAL MODE
ENTITY train_head
ALL FIELDS WITH CORRESPONDING #( keys )
RESULT DATA(lt_head_r)
FAILED DATA(lt_failed).

result = VALUE #( FOR ls_head_r IN lt_head_r ( %tky = ls_head_r-%tky
%param = ls_head_r )).

```

ENDMETHOD.

ENDCLASS.

2.6. Creation of Service Definition

Create Service Definition for Consumption View: ZCO_TRAIN_HEAD:

ZSDEF_TRAIN_HEAD

```

@EndUserText.label: 'Service Definition for Training'
define service ZSDEF_TRAIN_HEAD {
    expose ZCO_TRAIN_HEAD;
    expose ZCO_TRAIN_ITEM;
}

```

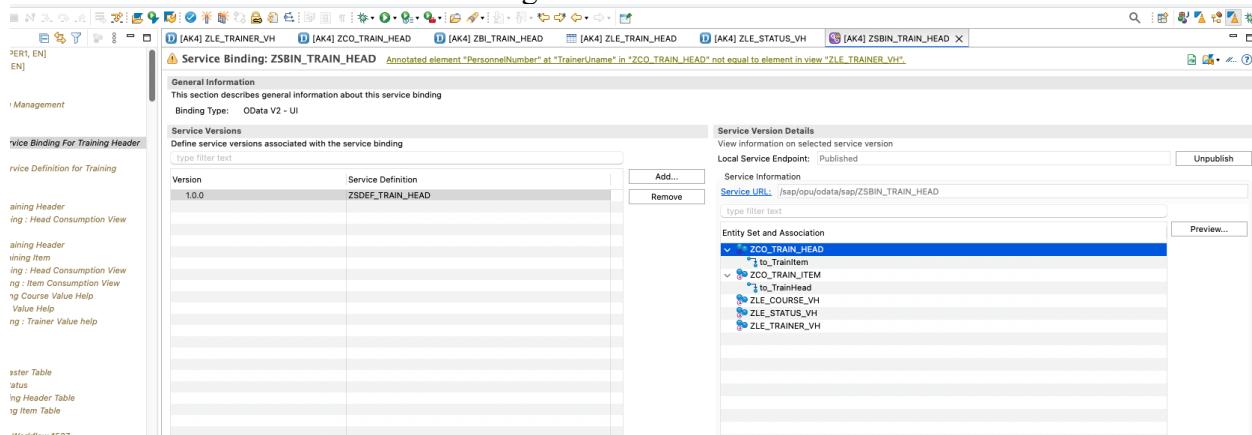
2.7. Creation of Service Binding

Create Service Binding for Service Definition: ZSDEF_TRAIN_HEAD:

ZSBIN_TRAIN_HEAD

Binding Type: ODATA V2 - UI

Activate and Publish the Service Binding:



3. Flexible workflow

3.1. Creation of Workflow Dictionary Tables

Create Dictionary Master Table to store the number of levels for workflow:

ZLE_TRAIN_WF_MAS

```
@EndUserText.label : 'Workflow Master Table'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #ALLOWED
define table zle_train_wf_mas {
  key mandt   : mandt not null;
  key appr_level : int4 not null;
  appr_user   : usrid;
}
```

Let's Maintain Two Levels and corresponding users:

Note You can maintain any level you want and also instead of using USRID you can use Position as well

IN...	Created by
1	K1537
2	K1537

Create a Dictionary Transaction Table to store the Approval log against the Training ID and status for each approval: **ZLE_TRAIN_WF_TRA**

```
@EndUserText.label : 'Training Workflow for approval'
@AbapCatalog.enhancement.category : #NOT_EXTENSIBLE
@AbapCatalog.tableCategory : #TRANSPARENT
@AbapCatalog.deliveryClass : #A
@AbapCatalog.dataMaintenance : #ALLOWED
define table zle_train_wf_tra {
  key mandt   : mandt not null;
  key learning_id : numc10 not null;
  key appr_level : int4 not null;
  appr_user   : usrid;
  status      : zle_de_train_status;
  execute_time : uzeit;
  execute_date : datum;
```

```

work_item_id : sww_wiid;
created_date : datum;
created_time : uzeit;
}

```

3.2. Creation of Structure used in workflow

Here you can use the own structure but here since the scenario is simple, we are using the already created Dictionary Table which is the header table to store the records:

ZLE_TRAIN_HEAD

The screenshot shows the SAP Dictionary View for the 'ZLE_TRAIN_HEAD' table. The table has 8 fields:

Field	Key	Init...	Data element	Data Type	Length	Deci...	Coordinate	Short Description
CLIENT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>		CLNT	3	0	0	
LEARNING_ID	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	NUMC10	NUMC	10	0	0	0 Numeric Character Field, Length 10
DESCRIPTION	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	100	0	0	
COURSE_ID	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	3	0	0	
LEARN_STATUS_ID	<input type="checkbox"/>	<input type="checkbox"/>		CHAR	1	0	0	
TRAINER_UNAME	<input type="checkbox"/>	<input type="checkbox"/>	USRID	CHAR	12	0	0	0 User who created the record
CREATED_BY	<input type="checkbox"/>	<input type="checkbox"/>	ABP_CREATION_USR	CHAR	12	0	0	0 Created By User
CREATED_ON	<input type="checkbox"/>	<input type="checkbox"/>	ABP_CREATION_TS	DEC	21	7	0	0 Creation Date Time

3.3. Creation of Agent Rule

3.3.1. Functional Module for Agent Rule

Workflows will have agents to whom the task is sent and it can be achieved in various ways, here we will use the Agent rule.

First create a Function Module for the rule with the following as parameters and activate it with the following source code

The screenshot shows the SAP Function Builder for the 'ZLE_TRAINING_AGENT_RULE' function module. It has 2 parameters:

Parameter Name	Typing	Associated Type	Optional	Short text	Long Text
ACTOR_TAB	LIKE	SWHACTOR	<input type="checkbox"/>	Rule Resolution Result	Create
AC_CONTAINER	LIKE	SWCONT	<input type="checkbox"/>	Instance Structure	Create
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		
			<input type="checkbox"/>		

The code reads the Rule container and fetch the agent user value from the Workflow transaction table:

The screenshot shows the SAP ABAP source code editor with the following details:

- Title Bar:** Function module ZLE_TRAINING_AGENT_RULE Active
- Toolbar:** Attributes, Import, Export, Changing, Tables, Exceptions, Source code
- Code Content:**

```

1  FUNCTION ZLE_TRAINING_AGENT_RULE.
2  *-----------------------------------------------------------------
3  *"**Local Interface:
4  *"  TABLES
5  *"      ACTOR_TAB STRUCTURE  SWHACTOR
6  *"      AC_CONTAINER STRUCTURE  SWCONT
7  *"-*
8  include <cntain>.
9
10  "-->Read the Inputs from the container
11  data(lv_learn) = VALUE num10( ac_container[ element = 'LEARNID' ]-value OPTIONAL ). 
12  data(lv_step) = VALUE int4( ac_container[ element = 'CURRSTEP' ]-value OPTIONAL ). 
13
14  "-->Read the user id from log table and pass it to agent
15  select SINGLE * FROM ZLE_TRAIN_WF_TRA where learning_id = @lv_learn
16    AND appr_level = @lv_step INTO @DATA(lwa).
17  if sy-subrc IS INITIAL.
18    APPEND VALUE #( otype = 'US' objid = lwa-APPR_USER ) to actor_tab.
19  endif.
20
21  ENDFUNCTION.
```

Full Code for the function Module

```

function zle_training_agent_rule
tables
  actor_tab like swhactor
  ac_container like swcont.

"-->It is Important to include this container
include <cntain>.

"-->Read the Inputs from the container
data(lv_learn)=VALUE num10( ac_container[ element = 'LEARNID' ]-value OPTIONAL ). 
data(lv_step)=VALUE int4( ac_container[ element = 'CURRSTEP' ]-value OPTIONAL ). 

"-->Read the user id from log table and pass it to agent
select SINGLE * FROM ZLE_TRAIN_WF_TRA where learning_id = @lv_learn
  AND appr_level = @lv_step INTO @DATA(lwa).
if sy-subrc IS INITIAL.
  APPEND VALUE #( otype = 'US' objid = lwa-APPR_USER ) to actor_tab.
endif.

ENDFUNCTION.
```

3.3.2. Agent Rule Creation in PFAC AC92000024

Go to TCODE PFAC: Click on create new

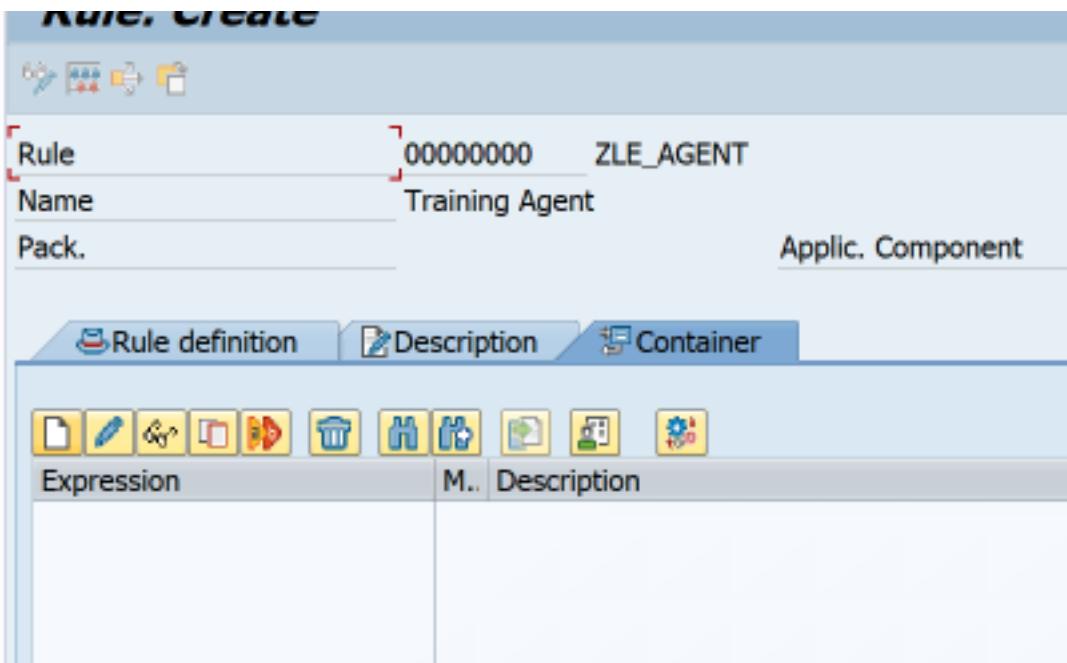


Enter the Abbreviation, Name, Category, and provide the function Module

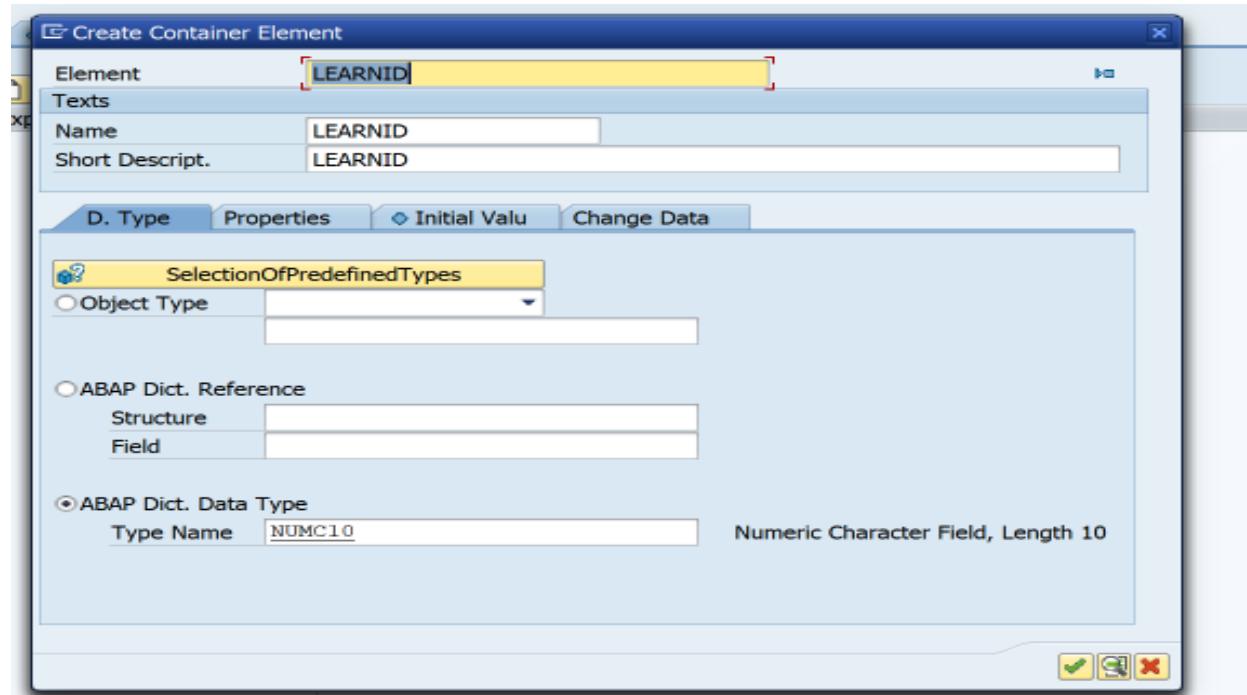
Basic data	
Abbr.	ZLE_AGENT
Name	Training Agent

Rule definition	
Category	F Agent Determination: Function to be Executed
Function Module	ZLE_TRAINING_AGENT_RULE
<input type="checkbox"/> Terminate If Rule Resolution W/o Result	

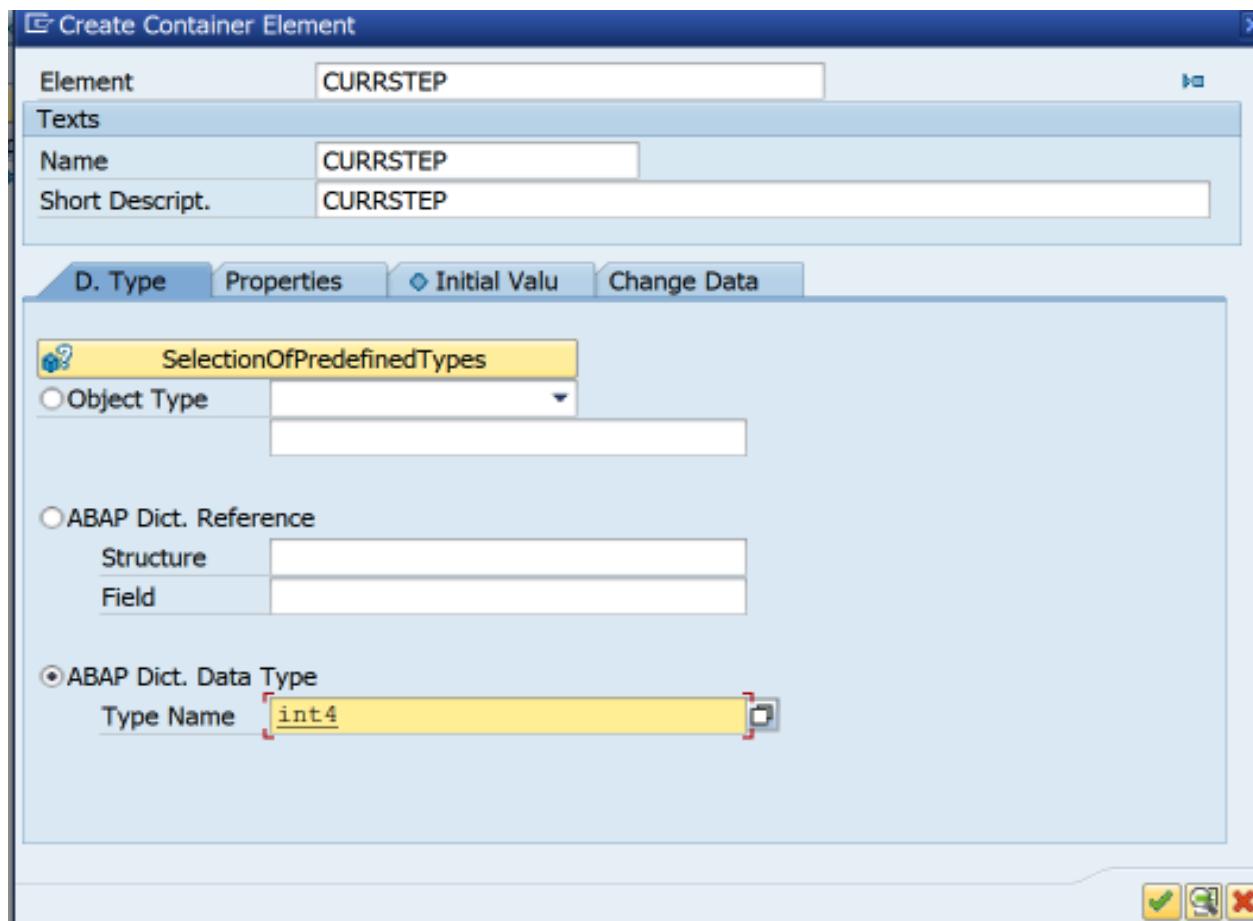
Go to the Container Tab and Create two Container Elements:



Element 1: LEARNID



Element 2: CURRSTEP



Click on Save to generate a new Number:

Rule: Change

Rule 92000024 ZLE_AGENT

Name Training Agent

Pack. Z1537_TRAIN Applic. Component

Expression	M.	Description	Initial value
• LEARNID		LEARNID	< Not Set >
• CURRSTEP		CURRSTEP	< Not Set >

3.4. Creation of Classes for Workflow

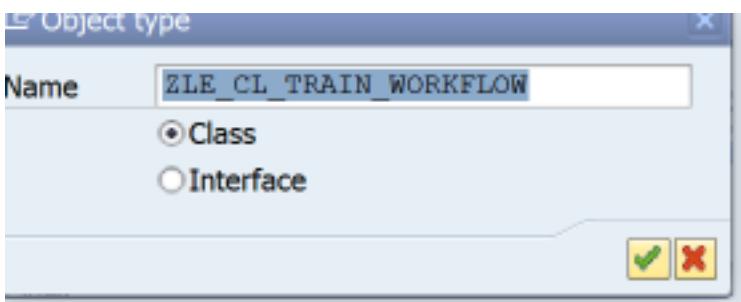
To create a workflow, we need 3 classes

- A Class and its event which will trigger the workflow
- A Class for Definition Application Base for Flexible Workflow Control
- A Class for Runtime Application Base for Flexible Workflow Control

3.4.1. Creation of Workflow Class and its event ZLE_CL_TRAIN_WORKFLOW

You can use the GUI approach or directly create it in ABAP eclipse

Go to TCODE: SE24: Create a new Class



Go to Interface tab and give IF_WORKFLOW automatically two new interfaces are added

Interface	Abstract	Final	Modeler	Description
BI_OBJECT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Business Instance
BI_PERSISTENT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Persistent Business Instance
IF_WORKFLOW	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	Business Workflow
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Go to attributes and add the following Key Number for workflow instance: **KEY_NUMBER** and **GS_OBJ**

Attribute	Level	Visibility	K R...	Typing	Associated Type	Description
KEY_NUMBER	Instance Attribute	Public	<input type="checkbox"/>	Type	ZLE_TRAIN_HEAD-LEARNING_ID	
GS_OBJ	Instance Attribute	Public	<input type="checkbox"/>	Type	SIBFLPOR	
			<input type="checkbox"/>	Type		
			<input type="checkbox"/>	Type		

Go to the Event tabs and add a New Event: **START_WF**

Type	Visibility	Description
Instance Event	Public	Trigger the Workflow

Select the Event and click on parameters and add the following as the Parameter: **LEARNING**

Parameters	O...	Typing	Associated Type	Default Value
LEARNING		<input type="checkbox"/> Type	ZLE_TRAIN_HEAD	
		<input type="checkbox"/> Type		
		<input type="checkbox"/> Type		

Creation of Constructor : Click on create constructor button which will add the constructor method

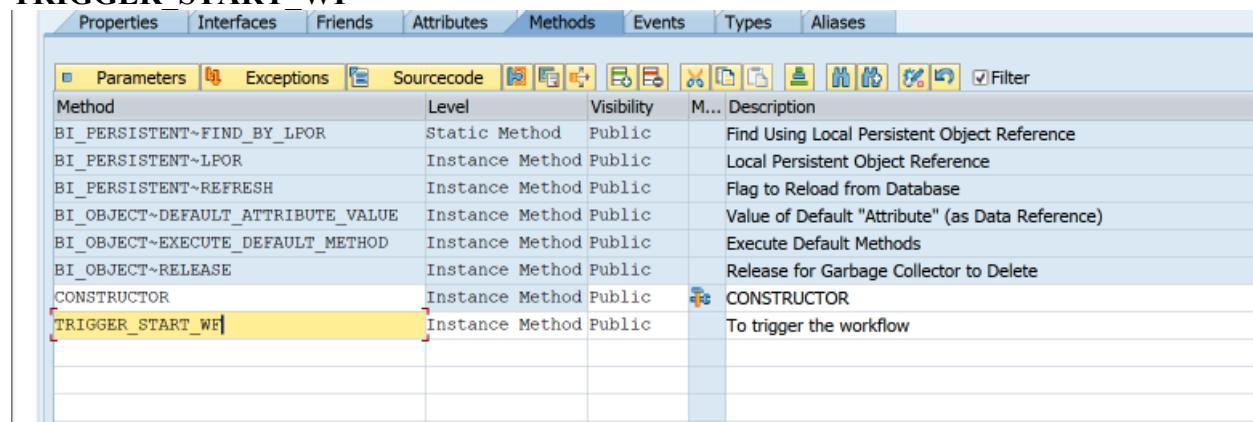
Select the Constructor and click on Parameters tab to add parameter for constructor

Goto Parameters	Level	Visibility	M...	Description
BI_PERSISTENT~FIND_BY_LPOR	Static Method	Public		Find Using Local Persistent Object Reference
BI_PERSISTENT~LPOR	Instance Method	Public		Local Persistent Object Reference
BI_PERSISTENT~REFRESH	Instance Method	Public		Flag to Reload from Database
BI_OBJECT~DEFAULT_ATTRIBUTE_VALUE	Instance Method	Public		Value of Default "Attribute" (as Data Reference)
BI_OBJECT~EXECUTE_DEFAULT_METHOD	Instance Method	Public		Execute Default Methods
BI_OBJECT~RELEASE	Instance Method	Public		Release for Garbage Collector to Delete
CONSTRUCTOR	Instance Method	Public		CONSTRUCTOR

Add the LV_KEY_NUMBER as Parameter to constructor

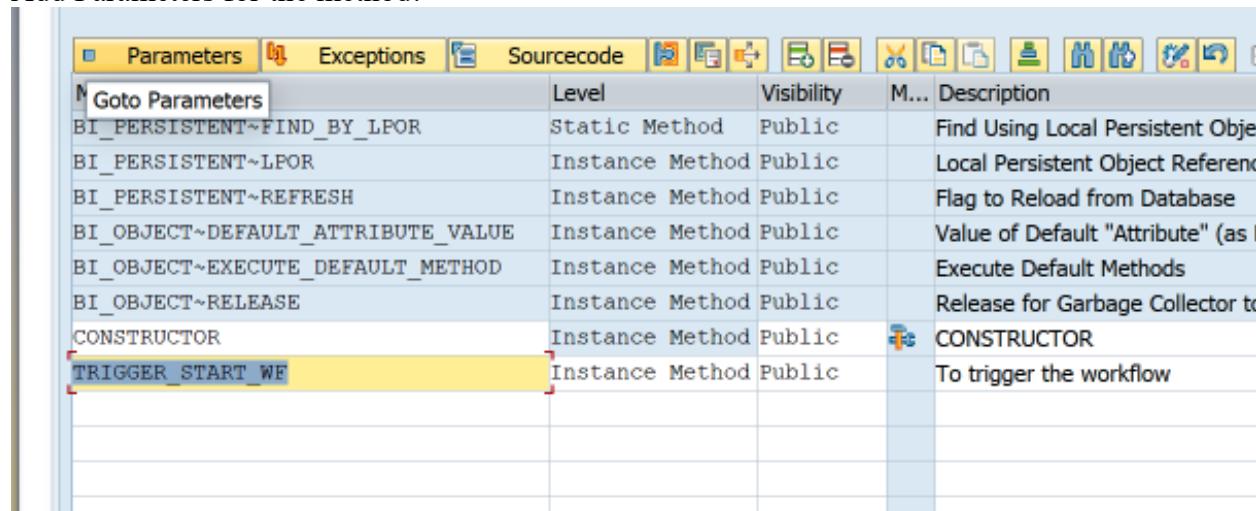
Parameter	P...	O...	Typing Method	Associated Type	Default Value
LV_KEY_NUMBER		<input type="checkbox"/> Type		ZLE_TRAIN_HEAD-LEARNING_ID	
		<input type="checkbox"/> Type			

Go back to Method and add a method to store the Approver log table and trigger the workflow:
TRIGGER_START_WF



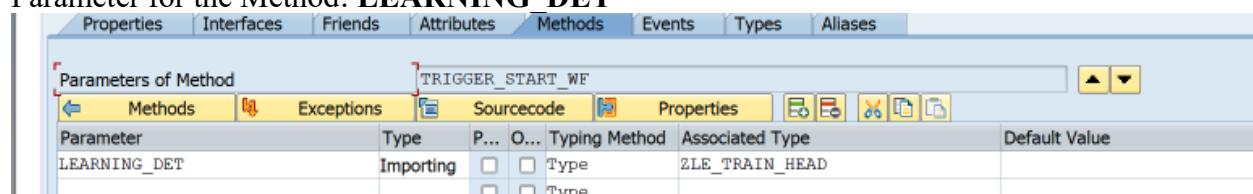
Method	Level	Visibility	M...	Description
BI_PERSISTENT~FIND_BY_LPOR	Static Method	Public		Find Using Local Persistent Object Reference
BI_PERSISTENT~LPOR	Instance Method	Public		Local Persistent Object Reference
BI_PERSISTENT~REFRESH	Instance Method	Public		Flag to Reload from Database
BI_OBJECT~DEFAULT_ATTRIBUTE_VALUE	Instance Method	Public		Value of Default "Attribute" (as Data Reference)
BI_OBJECT~EXECUTE_DEFAULT_METHOD	Instance Method	Public		Execute Default Methods
BI_OBJECT~RELEASE	Instance Method	Public		Release for Garbage Collector to Delete
CONSTRUCTOR	Instance Method	Public		CONSTRUCTOR
TRIGGER_START_WF	Instance Method	Public		To trigger the workflow

Add Parameters for the method:



Parameter	Level	Visibility	M...	Description
Goto Parameters	Static Method	Public		Find Using Local Persistent Obj
BI_PERSISTENT~FIND_BY_LPOR	Instance Method	Public		Local Persistent Object Referenc
BI_PERSISTENT~REFRESH	Instance Method	Public		Flag to Reload from Database
BI_OBJECT~DEFAULT_ATTRIBUTE_VALUE	Instance Method	Public		Value of Default "Attribute" (as D
BI_OBJECT~EXECUTE_DEFAULT_METHOD	Instance Method	Public		Execute Default Methods
BI_OBJECT~RELEASE	Instance Method	Public		Release for Garbage Collector to
CONSTRUCTOR	Instance Method	Public		CONSTRUCTOR
TRIGGER_START_WF	Instance Method	Public		To trigger the workflow

Parameter for the Method: **LEARNING_DET**



Parameters of Method		TRIGGER_START_WF			
Parameter	Type	P... O...	Typing Method	Associated Type	Default Value
LEARNING_DET	Importing	<input type="checkbox"/> <input type="checkbox"/> Type		ZLE_TRAIN_HEAD	

Add the following Code in the following methods:

Class Builder Class ZLE_CL_TRAIN_WORKFLOW Change

Method BI_PERSISTENT~FIND_BY_LPOR inactive (revised)

```
1 method BI_PERSISTENT~FIND_BY_LPOR.  
2     create OBJECT result type zle_cl_train_workflow  
3     EXPORTING  
4         lv_key_number = conv #( lpof(10) ).  
5     endmethod.
```

Class Builder Class ZLE_CL_TRAIN_WORKFLOW Change

Method BI_PERSISTENT~LPOR inactive

```
1 method BI_PERSISTENT~LPOR.  
2     result = me->gs_obj.  
3 endmethod.
```

Class Builder Class ZLE_CL_TRAIN_WORKFLOW Change

Pattern Pretty Printer Signature Public Section Protect

Ty.	Parameter	Typing	Description
lv_key_number	TYPE ZLE_TRAIN_HEAD-LEARNING_ID	Numeric Character Field, Length 10	

Method CONSTRUCTOR inactive

```

1 method CONSTRUCTOR.
2     gs_obj-catid = 'CL'.
3     gs_obj-typeid = 'ZLE_CL_TRAIN_WORKFLOW'.
4     gs_obj-instid = lv_key_number.
5 endmethod.

```

Method TRIGGER_START_WF active

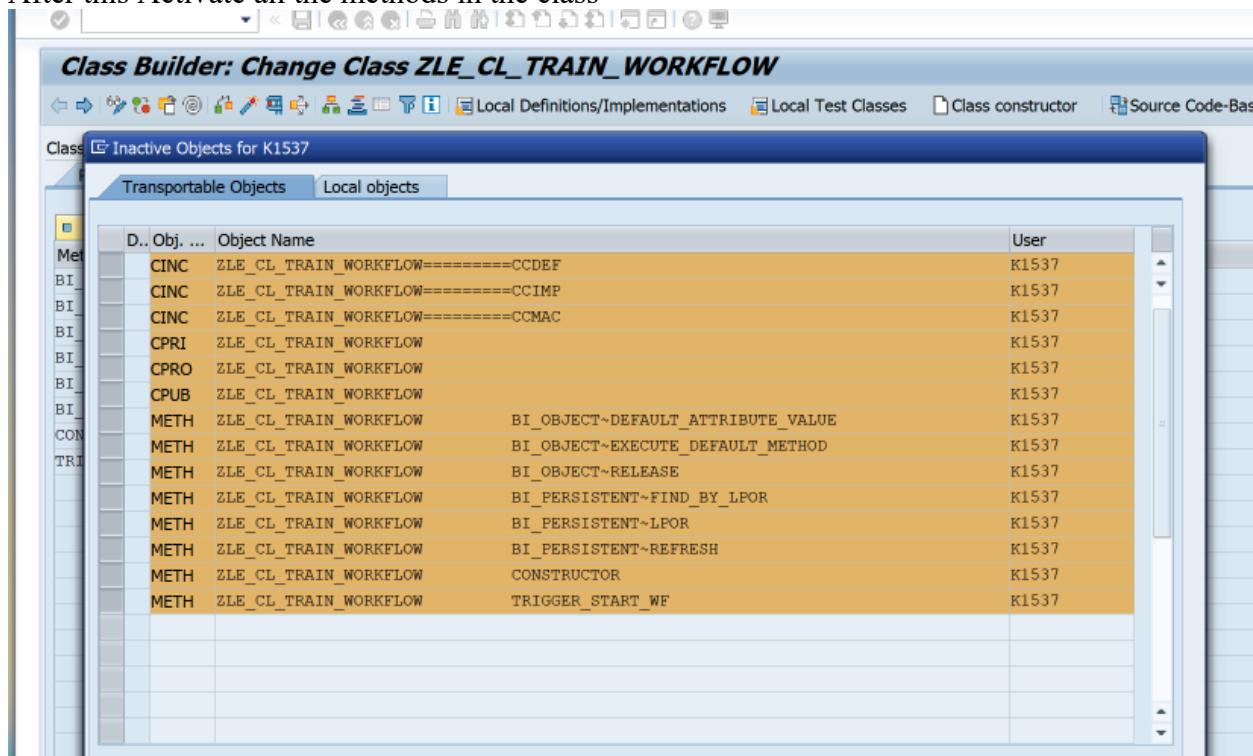
```

1 method TRIGGER_START_WF.
2     ""->Get the Approval Levels from master
3     DATA : lt_appr_tran TYPE TABLE OF zle_train_wf_tra.
4     select * FROM zle_train_wf_mas INTO TABLE @DATA(lt_mast).
5     if sy-subrc IS INITIAL.
6         lt_appr_tran = VALUE #( for lwa in lt_mast (
7             LEARNING_ID = learning_det-learning_id
8             APPR_LEVEL = lwa-appr_level
9             APPR_USER = lwa-appr_user
10            ) ).
11        ""->Store it in the table
12        MODIFY zle_train_wf_tra FROM TABLE lt_appr_tran.
13    endif.
14    ""->Get the Event Container
15    CALL METHOD cl_swf_evt_event=>get_event_container
16        EXPORTING
17            im_objcateg = cl_swf_evt_event=>mc_objcateg_cl
18            im_objtype = 'ZLE_CL_TRAIN_WORKFLOW'
19            im_event = 'START_WF'
20        RECEIVING
21            re_reference = data(lr_event_parameters).
22    try.
23        ""->Set Input Values to the Event Parameter
24        CALL METHOD lr_event_parameters->set
25            EXPORTING
26                name = conv #( 'LEARNING' ) "#EC NOTEXT
27                value = learning_det.
28        catch cx_swf_cnt_cont_access_denied INTO DATA(lo_ex1).
29        catch cx_swf_cnt_elem_def_invalid INTO DATA(lo_ex2).
30        catch cx_swf_cnt_unit_type_conflict INTO DATA(lo_ex3).
31        catch cx_swf_cnt_container INTO DATA(lo_ex4).
32    endtry.
33    ""->Trigger the Workflow
34    TRY.
35        CALL METHOD cl_swf_evt_event=>raise
36            EXPORTING
37                im_objcateg = cl_swf_evt_event=>mc_objcateg_cl
38                im_objtype = 'ZLE_CL_TRAIN_WORKFLOW'
39                im_event = 'START_WF'
40                im_objkey = conv #( learning_det-learning_id )
41                im_event_container = lr_event_parameters.

```

Scope: \METHOD TRIGGER_START_WF\TRY

After this Activate all the methods in the class



3.4.1.1. Full Code ZLE_CL_TRAIN_WORKFLOW

```
class ZLE_CL_TRAIN_WORKFLOW definition  
public  
final  
create public .
```

public section.

interfaces BI_OBJECT .
interfaces BI_PERSISTENT .
interfaces IF_WORKFLOW .

types:
BEGIN OF Ty_RECEPIENTS,
 user_id TYPE sysid,
END OF ty_recipients.

types:
tt_recipients **TYPE TABLE** of ty_recipients **WITH DEFAULT KEY**

data KEY_NUMBER type ZLE_TRAIN_HEAD-LEARNING_ID .
data GS_QBL type SIREL_POR .

```
events START_WF  
    exporting  
        1. (LEARNING) :> ZLF_TRAIN_HEAD
```

methods CONSTRUCTOR

!LV_KEY_NUMBER type ZLE_TRAIN_HEAD

methods TRIGGER_START_WF
importing
 HEARING_DATE .. ZLE_TRAIN_HEAD

```
!LEARNING_DET type ZLE_TRAIN_HEAD .  
class-methods SEND_EMAIL
```

importing
!IM_IT_USERID type TT_RECEPIENTS

```

!IM_LEARNING type ZLE_TRAIN_HEAD
!IM_EMAIL_TYPE type TEXT10
!IM_REJECTION_NOTE type STRING optional .
protected section.
private section.
ENDCLASS.

```

CLASS ZLE_CL_TRAIN_WORKFLOW IMPLEMENTATION.

method BI_OBJECT~DEFAULT_ATTRIBUTE_VALUE.

endmethod.

method BI_OBJECT~EXECUTE_DEFAULT_METHOD.

endmethod.

method BI_OBJECT~RELEASE.

endmethod.

method BI_PERSISTENT~FIND_BY_LPOR.

create OBJECT result type zle_cl_train_workflow
EXPORTING

lv_key_number = conv #(lpor(10)).
endmethod.

method BI_PERSISTENT~LPOR.

result = me->gs_obj.
endmethod.

method BI_PERSISTENT~REFRESH.

endmethod.

method CONSTRUCTOR.

gs_obj-catid = 'CL'.
gs_obj-typeid = 'ZLE_CL_TRAIN_WORKFLOW'.
gs_obj-instid = lv_key_number.
endmethod.

method TRIGGER_START_WF.

""->Get the Approval Levels from master
DATA : lt_appr_tran TYPE TABLE OF zle_train_wf_tr.
select * FROM zle_train_wf_mas INTO TABLE @DATA(lt_mast).
if sy-subrc IS INITIAL.
lt_appr_tran = VALUE #(for lwa in lt_mast (
LEARNING_ID = learning_det-learning_id
APPR_LEVEL = lwa-appr_level
APPR_USER = lwa-appr_user
)).
MODIFY zle_train_wf_tr from TABLE lt_appr_tran.
endif.

CALL METHOD cl_swf_evt_event=>get_event_container
EXPORTING
im_objcateg = cl_swf_evt_event=>mc_objcateg cl
im_objtype = 'ZLE_CL_TRAIN_WORKFLOW'

```

im_event = 'START_WF'
RECEIVING
re_reference = data(lr_event_parameters).

try.
CALL METHOD lr_event_parameters->set
EXPORTING
name = conv #( 'LEARNING' ) "#EC NOTEXT
value = learning_det.
catch cx_swf_cnt cont access denied INTO DATA(lo_ex1).
catch cx_swf_cnt_elem_def_invalid INTO DATA(lo_ex2).
catch cx_swf_cnt_unit_type_conflict INTO DATA(lo_ex3).
catch cx_swf_cnt_container INTO DATA(lo_ex4).
endtry.
** TRY.
CALL METHOD cl_swf_evt_event=>raise
EXPORTING
im_objcateg = cl_swf_evt_event=>mc_objcateg_cl
im_objtype = 'ZLE_CL_TRAIN_WORKFLOW'
im_event = 'START_WF'
im_objkey = conv #( learning_det-learning_id )
im_event_container = lr_event_parameters.

CATCH cx_swf_evt_invalid_objtype.
CATCH cx_swf_evt_invalid_event.
ENDTRY.

endmethod.

method SEND_EMAIL.
"""->Common Method to send Custom Email during the execution of workflow
"""->Constants Declaration
CONSTANTS : c_doc_type TYPE so_obj_tp VALUE 'HTM',
c_sub_0001 TYPE pa0105-subty VALUE '0001',
c_sub_0010 TYPE pa0105-subty VALUE '0010'.

"""->Class Definition
CLASS cl_bcs DEFINITION LOAD.

"""->Object Declaration
DATA: lo_send_request TYPE REF TO cl_bcs VALUE IS INITIAL,
lo_document TYPE REF TO cl_document_bcs VALUE IS INITIAL,
lo_recipient TYPE REF TO if_recipient_bcs VALUE IS INITIAL,
lo_sender TYPE REF TO if_sender_bcs VALUE IS INITIAL.

"""->Internal Table Declaration
DATA: lt_message_body TYPE bcsy_text VALUE IS INITIAL.

"""->Work Area Declaration
DATA: ls_message_ody TYPE bcsy_text,
ls_body_msg TYPE solisti1.

"""->Variable Declaration
DATA: lv_message_text TYPE string,
lv_send TYPE adr6-smtp_addr,
lv_subject TYPE so_obj_des.

CHECK im_it_userid IS NOT INITIAL.

TRY.
lo_send_request = cl_bcs=>create_persistent().
CATCH cx_send_req_bcs INTO DATA(lo_excep).
ENDTRY.

"""->Get Full Name
IF im_it_userid IS NOT INITIAL.
SELECT b~usrnid_long,
c~ename
FROM pa0105 AS a

```

```

INNER JOIN pa0105 AS b
ON b~pernr = a~pernr
AND b~subty = @c_sub_0010
AND b~begda <= @sy-datum
AND b~endda >= @sy-datum
INNER JOIN pa0001 AS c
ON c~pernr = a~pernr
AND c~begda <= @sy-datum
AND c~endda >= @sy-datum
FOR ALL ENTRIES IN @im_it_userid
WHERE a-usrid = @im_it_userid-user_id
AND a~subty = @c_sub_0001
AND a~begda <= @sy-datum
AND a~endda >= @sy-datum
INTO TABLE @DATA(lt_recipient).
ENDIF.

delete lt_recipient WHERE usrid long IS INITIAL.
sort lt_recipient ASCENDING BY ename.
DELETE ADJACENT DUPLICATES FROM lt_recipient COMPARING ename.

""->Mail Subject
CASE im_email_type.
when 'C'. ""Creation of Task
lv_subject = |Please Approve Training Request({ im_learning-learning_id }).|
when 'A'. "" Approval of Task
lv_subject = |Training Request ({ im_learning-learning_id }) approved|.
when 'R'. ""Rejection of Request
lv_subject = |Training Request ({ im_learning-learning_id }) Rejected|.
ENDCASE.

""->Message Body
APPEND '<html>' TO lt_message_body.
APPEND '<body><p>' TO lt_message_body.
""->Address Recipients with Names
APPEND |Dear &nbsp;| TO lt_message_body.
LOOP AT lt_recipient INTO DATA(ls_recipient).
IF sy-tabix EQ 1.
APPEND ls_recipient-ename TO lt_message_body.
ELSE.
APPEND |,{ ls_recipient-ename }| TO lt_message_body.
ENDIF.
TRY.
lo_recipient = cl_cam_address_bcs=>create_internet_address( ls_recipient-usrid_long ).  

lo_send_request=>add_recipient(EXPORTING i_recipient = lo_recipient  

i_express = abap_true ).  

CATCH cx_send_req_bcs INTO DATA(lo_send_exception).  

CATCH cx_address_bcs INTO DATA(lo_address_exception).
ENDTRY.
ENDLOOP.
APPEND |</br>| TO lt_message_body.
CASE im_email_type.
WHEN 'C'. ""->Creation
APPEND |Training Request { im_learning-learning_id } is sent for your Approval| TO lt_message_body.
WHEN 'A'. ""->Approved
APPEND |Training Request { im_learning-learning_id } is Approved.| TO lt_message_body.
WHEN 'R'. ""->Rejected
APPEND |Training Request { im_learning-learning_id } is Rejected.<br>| TO lt_message_body.
Append |Rejection Note : { im_rejection_note } <br>| to lt_message_body.
ENDCASE.
APPEND '</p></body></html>' TO lt_message_body.

TRY.
lo_document = cl_document_bcs=>create_document( i_type = c_doc_type
i_text = lt_message_body
i_subject = lv_subject ).  

CATCH cx_document_bcs INTO DATA(lo_doc_exception).
ENDTRY.

TRY.
""->Prepare Sender

```

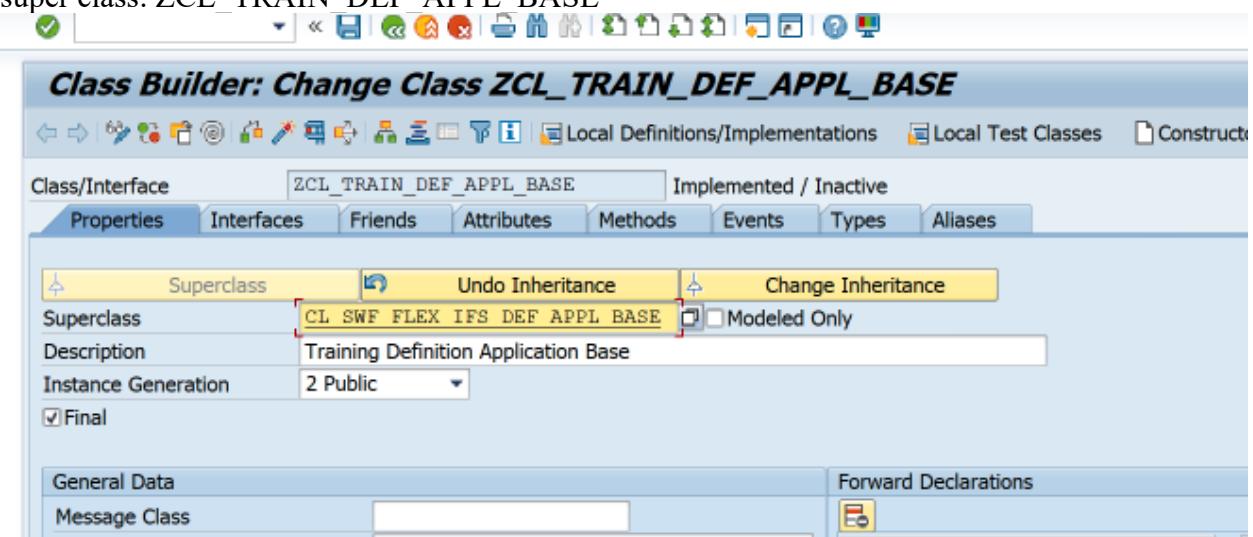
```

lo_sender = cl_sapuser_bcs->create('K1537'). "" Can use your user id
*   lo_recipient = cl_cam_address_bcs->create_internet_address( lv_send ).
CATCH cx_address_bcs INTO DATA(lo_address_bcs).
ENDTRY.
TRY.
  lo_send_request->set_sender(EXPORTING i_sender = lo_sender).
  lo_send_request->set_document( lo_document ).
  lo_send_request->send().
CATCH cx_send_req_bcs INTO lo_send_exception.
CATCH cx_address_bcs INTO lo_address_exception.
ENDTRY.

endmethod.
ENDCLASS.
```

3.4.2. Creation of Class Definition Application base ZCL_TRAIN_DEF_APPL_BASE

Create a new Z Class by giving the standard class CL_SWF_FLEX_IFS_DEF_APPL_BASE as super class: ZCL_TRAIN_DEF_APPL_BASE



This class will not have any code, just activate it.

3.4.2.1. Full Code ZCL_TRAIN_DEF_APPL_BASE

```

class ZCL_TRAIN_DEF_APPL_BASE definition
public
  inheriting from CL_SWF_FLEX_IFS_DEF_APPL_BASE
  final
  create public .

public section.
protected section.
private section.
ENDCLASS.
```

```

CLASS ZCL_TRAIN_DEF_APPL_BASE IMPLEMENTATION.
ENDCLASS.
```

3.4.3. Creation of Class Run time Application base ZCL_TRAIN_RUN_APPL_BASE

You have to redefine the methods into our Z class, which will be triggered during the workflow runtime.

Here we will use the following methods

->IF_SWF_FLEX_IFS_RUN_APPL_STEP~ON_CREATION_CALLBACK

This method gets called at creation of each task, this is used to write the current approver level in the workflow and send email for the task receiver

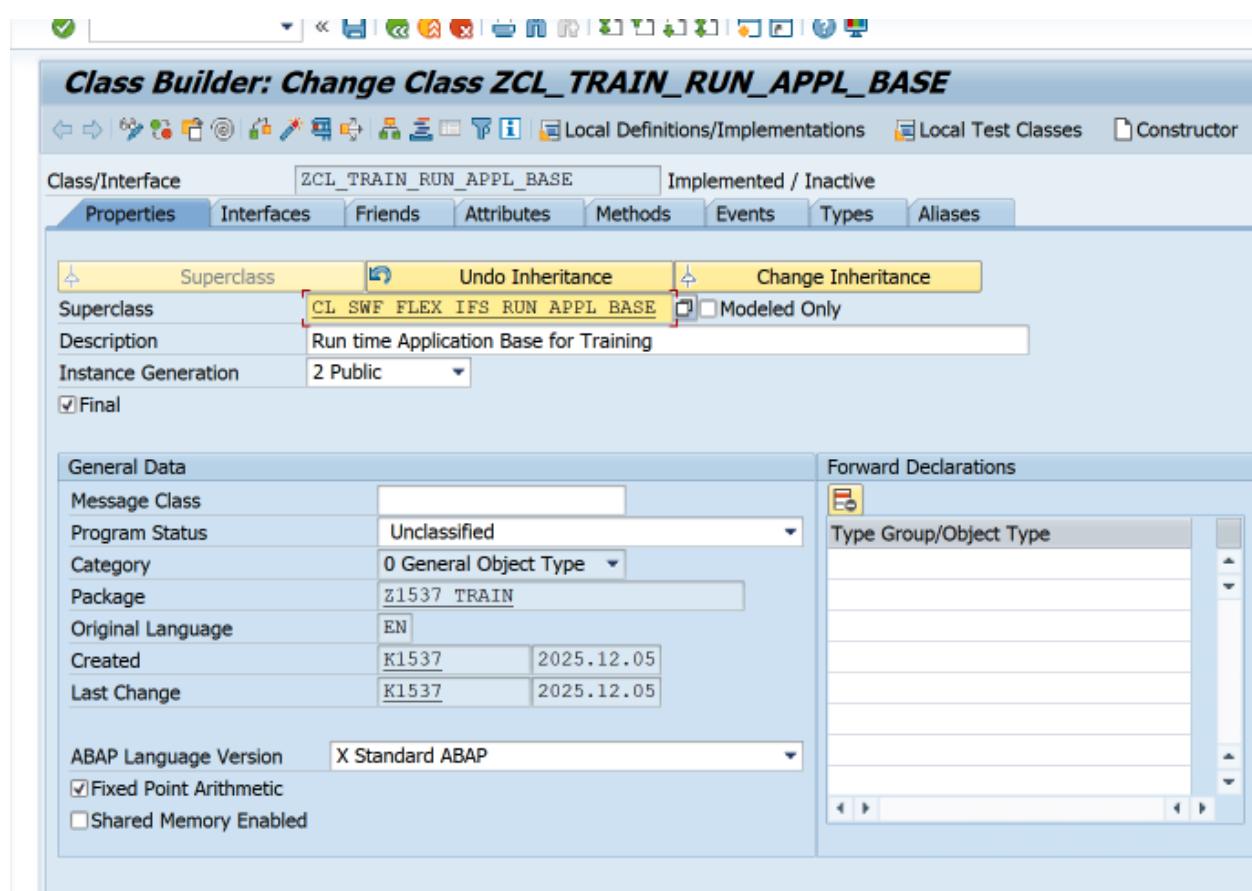
-> IF_SWF_FLEX_IFS_RUN_APPL_STEP~AFTER_COMPLETION_CALLBACK

This method gets called at completion of each task and is used to store the decision of each step in the ZTABLE

IF_SWF_FLEX_IFS_RUN_APPL~RESULT_CALLBACK

This method gets called at completion of workflow with the decision

Create a new Z Class by giving the standard class CL_SWF_FLEX_IFS_RUN_APPL_BASE as super class: ZCL_TRAIN_RUN_APPL_BASE



Save and activate the class.

3.4.3.1. Full Code ZCL_TRAIN_RUN_APPL_BASE

```

class ZCL_TRAIN_RUN_APPL_BASE definition
public
inheriting from CL_SWF_FLEX_IFS_RUN_APPL_BASE
final
create public .

public section.

methods IF_SWF_FLEX_IFS_RUN_APPL_STEP~AFTER_COMPLETION_CALLBACK
    redefinition .
methods IF_SWF_FLEX_IFS_RUN_APPL_STEP~ON_CREATION_CALLBACK
    redefinition .
methods IF_SWF_FLEX_IFS_RUN_APPL~RESULT_CALLBACK
    redefinition .
protected section.
private section.
ENDCLASS.

```

CLASS ZCL_TRAIN_RUN_APPL_BASE IMPLEMENTATION.

method IF_SWF_FLEX_IFS_RUN_APPL_STEP~AFTER_COMPLETION_CALLBACK.
 *This Method is called after the Completion of every task

DATA: lwa_learn TYPE zle_train_head,
 lv_curr_step TYPE int4.

DATA(lo_wf_cont) = io_context->get_workflow_container().
 DATA(lt_curr_act) = io_current_activity->get_execution_results().
 DATA(ls_curr_act) = VALUE #(lt_curr_act[1] OPTIONAL).

DATA(lv_user) = ls_curr_act->processor->objid.

TRY.

CALL METHOD lo_wf_cont->get
 EXPORTING
 name = CONV #('LEARNING')
 IMPORTING
 value = lwa_learn .

CALL METHOD lo_wf_cont->get
 EXPORTING
 name = CONV #('CurrentLevel')
 IMPORTING
 value = lv_curr_step.

DATA(lv_status) = COND #(when ls_curr_act->result = 'Approve' then 'A' else 'R').

""->Update Workflow Log table
 update ZLE_TRAIN_WF_TRA SET execute_date = sy-datum
 execute_time = sy-uzeit
 status = lv_status
 WHERE learning_id = lwa_learn-learning_id
 AND appr_level = lv_curr_step.

if lv_status eq 'R'.
 ""->On Reject Delete the further Approvals from table
 delete from ZLE_TRAIN_WF_TRA WHERE learning_id = lwa_learn-learning_id
 AND appr_level > lv_curr_step.
 endif.

""->Set the incremented data of Current Level to Workflow

```

lv_curr_step = lv_curr_step + 1.
CALL METHOD lo_wf_cont->set
EXPORTING
    name = CONV #( 'CurrentLevel' )
    value = lv_curr_step.

CATCH cx_swf_cnt_elem_not_found.
CATCH cx_swf_cnt_elem_type_conflict.
CATCH cx_swf_cnt_unit_type_conflict.
CATCH cx_swf_cnt_container.
ENDTRY.

endmethod.

method IF_SWF_FLEX_IFS_RUN_APPL_STEP~ON_CREATION_CALLBACK.
""->This Method gets triggered When a task is created in workflow
DATA: ls_learn TYPE zle_train_head,
    lv_curr_step TYPE int4.
DATA(lo_wf_cont) = io_context->get_workflow_container( ).
DATA(lt_curr_act) = io_current_activity->get_execution_results( ).
DATA(ls_curr_act) = VALUE #( lt_curr_act[ wi_stat = 'READY' wi_type = 'W' ] OPTIONAL ).
""->User to Process the task
DATA(lv_user) = ls_curr_act->processor_objid.

TRY.
""->Read the data from Container
CALL METHOD lo_wf_cont->get
EXPORTING
    name = CONV #( 'LEARNING' )
IMPORTING
    value = ls_learn.

""->Read the data Current Level from Workflow
CALL METHOD lo_wf_cont->get
EXPORTING
    name = CONV #( 'CurrentLevel' )
IMPORTING
    value = lv_curr_step.

""->Update Workflow Log table
update ZLE_TRAIN_WF_TRA SET created_date = sy-datum
    created_time = sy-uzzeit
    status = 'P'
    work_item_id = ls_curr_act-wi_id
    WHERE learning_id = ls_learn-learning_id
        AND appr_level = lv_curr_step.

CATCH cx_swf_cnt_elem_not_found.
CATCH cx_swf_cnt_elem_type_conflict.
CATCH cx_swf_cnt_unit_type_conflict.
CATCH cx_swf_cnt_container.
ENDTRY.

CHECK lv_user IS NOT INITIAL.
""->Send Email for approver
ZLE_CL_TRAIN_WORKFLOW=>send_email( im_it_userid = VALUE #( ( user_id = lv_user ) )
    im_learning = ls_learn
    im_email_type = 'C' ).
```

endmethod.

method IF_SWF_FLEX_IFS_RUN_APPL~RESULT_CALLBACK.

```

""">Data Declaration
DATA : lv_user TYPE uname,
       lv_rejnote TYPE string.

""">Internal Table Declaration
DATA : lt_note TYPE swf_utl_porb_tab,
       lt_objcont TYPE TABLE OF solisti1,
       lt_recep TYPE ZCL_ZLM_LEARN_WF=>tt_recipients.

""">Get the Initiator from Container
DATA(lo_wf cont)=io_context->get_workflow_container( ).
DATA(ls_lm_learn)=VALUE zle_train_head( ).

TRY.
  CALL METHOD lo_wf cont->get
    EXPORTING
      name = CONV #('WF_INITIATOR')
    IMPORTING
      value = lv_user.

  CALL METHOD lo_wf_cont->get
    EXPORTING
      name = CONV #('LEARNING')
    IMPORTING
      value = ls_lm_learn.

  CALL METHOD lo_wf_cont->get
    EXPORTING
      name = CONV #('ATTACH_OBJECTS')
    IMPORTING
      value = lt_note.

  DATA(lv_objkey)=VALUE sofolenti1-doc_id( lt_note[ 1 ]-instid OPTIONAL ).

  CATCH cx_swf_cnt_elem_not_found.
  CATCH cx_swf_cnt_elem_type_conflict.
  CATCH cx_swf_cnt_unit_type_conflict.
  CATCH cx_swf_cnt_container.
ENDTRY.

SHIFT lv_user LEFT DELETING LEADING 'US'.
CONDENSE lv_user.

DATA(lv_decision)=io_result->get_result()-result.

IF lv_decision = 'Approve'.
  UPDATE zle_train_head SET learn_status_id = 'A'
  WHERE learning_id = ls_lm_learn-learning_id.

  "->Send Email for Approval
  SELECT SINGLE * FROM zle_train_head
    INTO @DATA(ls_head)
    WHERE learning_id = @ls_lm_learn-learning_id.

  IF ls_head-trainer_uname IS NOT INITIAL.
    APPEND VALUE #( user_id = ls_head-trainer_uname ) TO lt_recep.
  ENDIF.

  APPEND VALUE #( user_id = lv_user ) TO lt_recep.
  ZLE_CL_TRAIN_WORKFLOW=>send_email( im_it_userid = lt_recep
    im_learning = ls_lm_learn
    im_email_type = 'A' ).

ELSE.
  UPDATE zle_train_head SET learn_status_id = 'R' WHERE learning_id = ls_lm_learn-learning_id.

  CHECK lv_objkey IS NOT INITIAL.

  CALL FUNCTION 'SO_DOCUMENT_READ_API1'
    EXPORTING
      document_id      = lv_objkey
    TABLES

```

```

object_content      = lt_objcont
EXCEPTIONS
document_id_not_exist = 1
operation_no_authorization = 2
x_error            = 3
OTHERS              = 4.
IF sy-subrc <> 0.
* Implement suitable error handling here
ENDIF.

LOOP AT lt_objcont INTO DATA(ls_objcontent).
lv_rejnote = COND string( WHEN lv_rejnote IS INITIAL
    THEN ls_objcontent-line
    ELSE ||{ lv_rejnote } { ls_objcontent-line }|| ).
ENDLOOP.

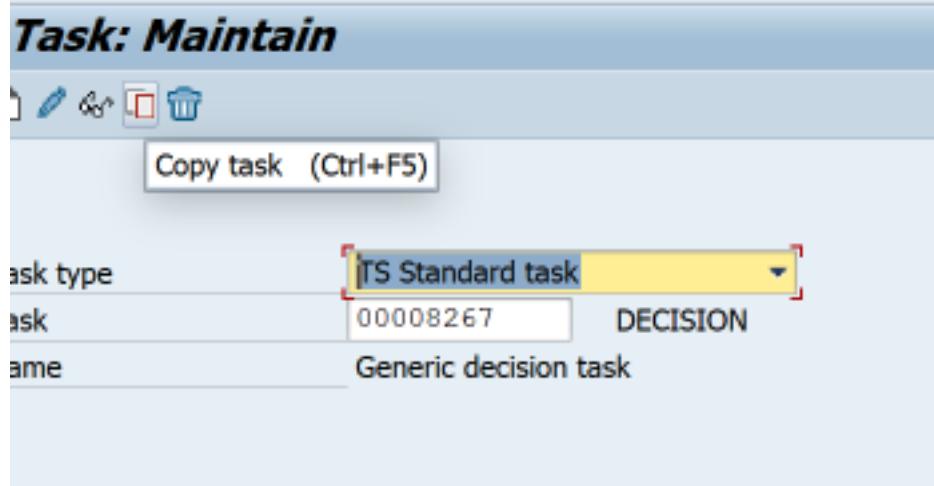
""->Send Email for Rejection
ZLE CL_TRAIN_WORKFLOW=>send_email( im_it_userid = VALUE #(( user_id = lv_user ))
    im_learning = ls_lm_learn
    im_email_type = 'R'
    im_rejection_note = lv_rejnote ).
ENDIF.

endmethod.
ENDCLASS.

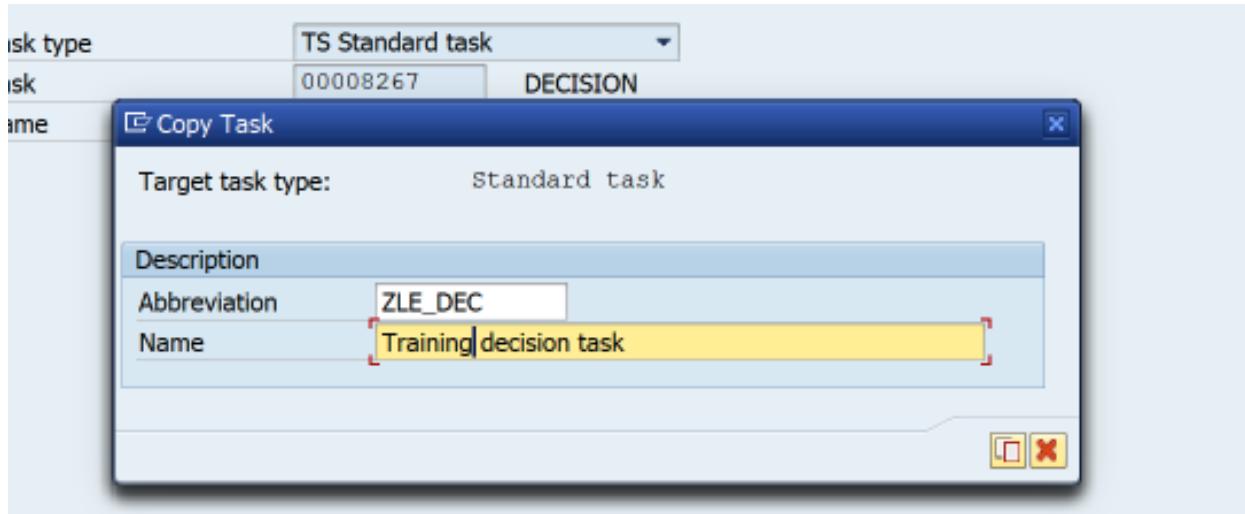
```

3.5. Creation of Custom Decision Task to be used in Workflow TS92000178

Go to TCODE PFTC: Give the standard task number TS00008267 and copy it into a Z task



Give the necessary name and Click Copy



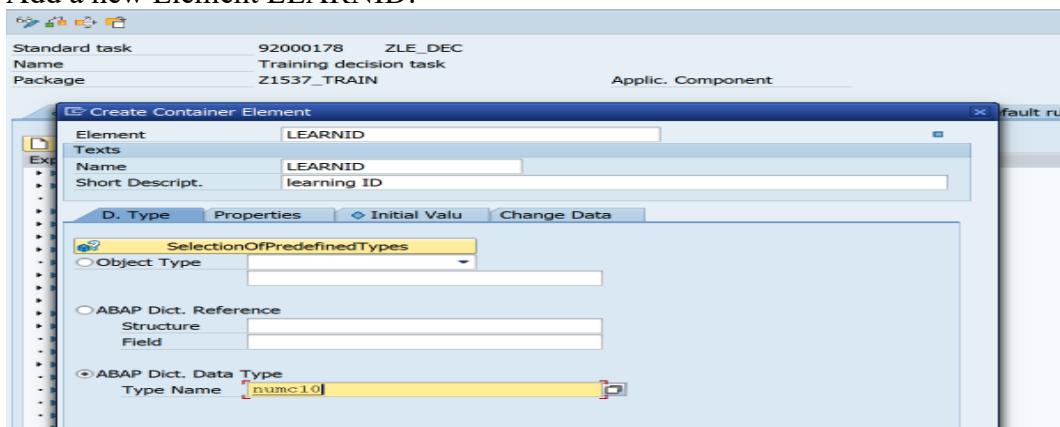
A new number for the task will be generated: TS92000178

We need the learning ID inside the task container hence create one element in it.

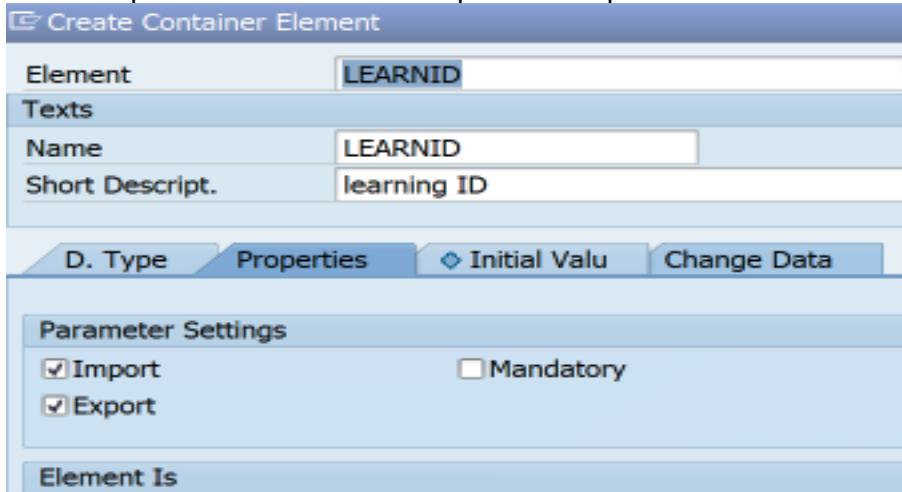
Click on Edit and go to the Container table and click create button which is present below Basic Data

Expression	M.	Description	Initial value
_Adhoc_Objects		Ad Hoc Objects of Workflow Instance	< Not Set >

Add a new Element LEARNID:

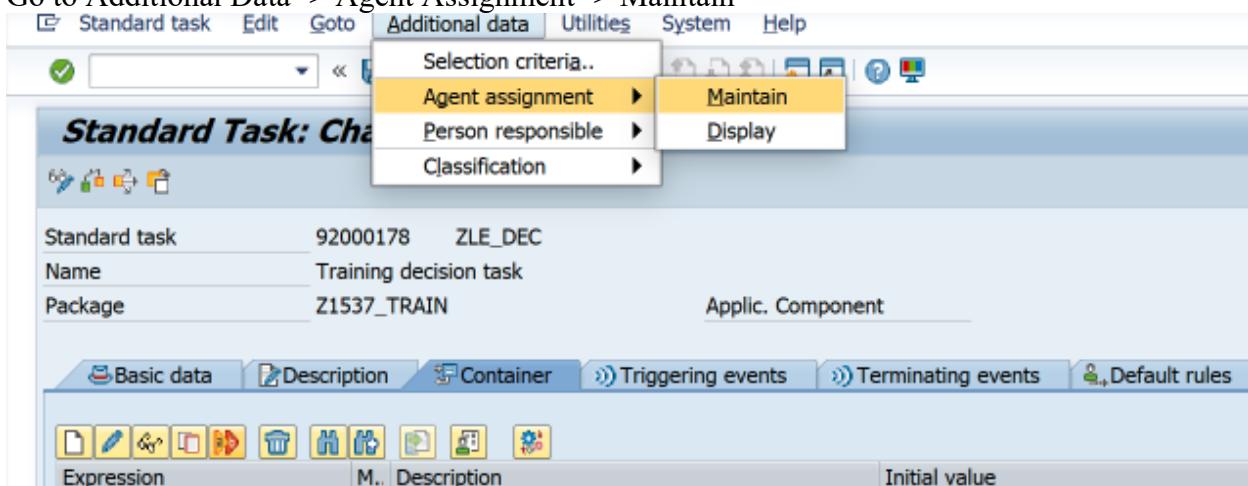


Go to Properties Tab and Select Import and Export:

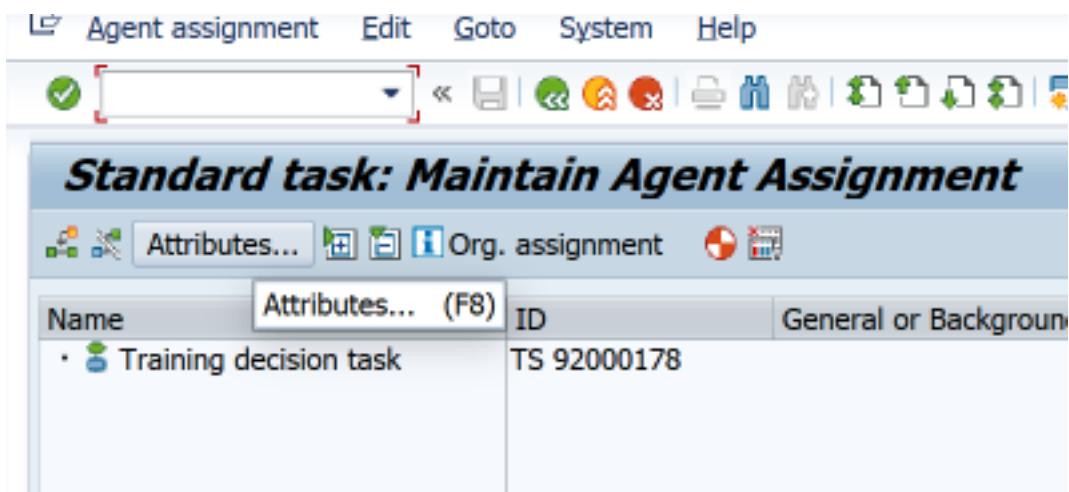


Make the task as General task:

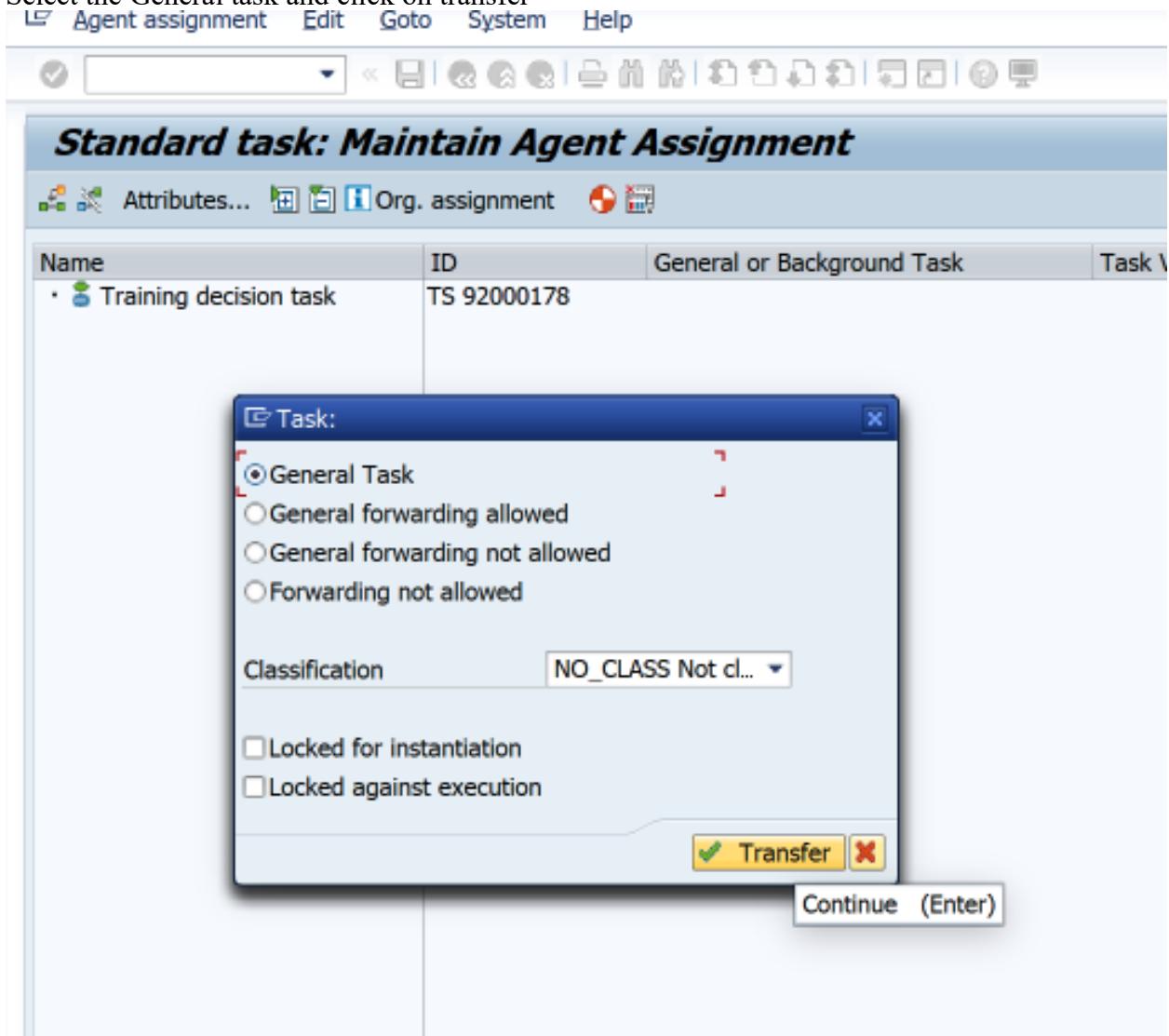
Go to Additional Data -> Agent Assignment -> Maintain



Click on attributes



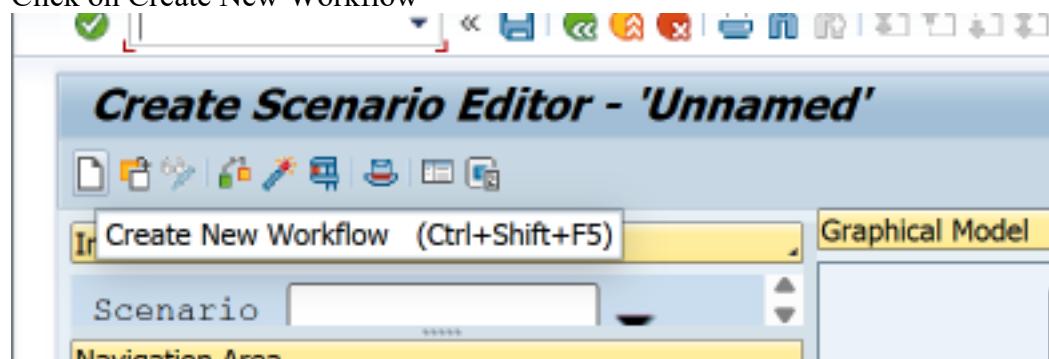
Select the General task and click on transfer



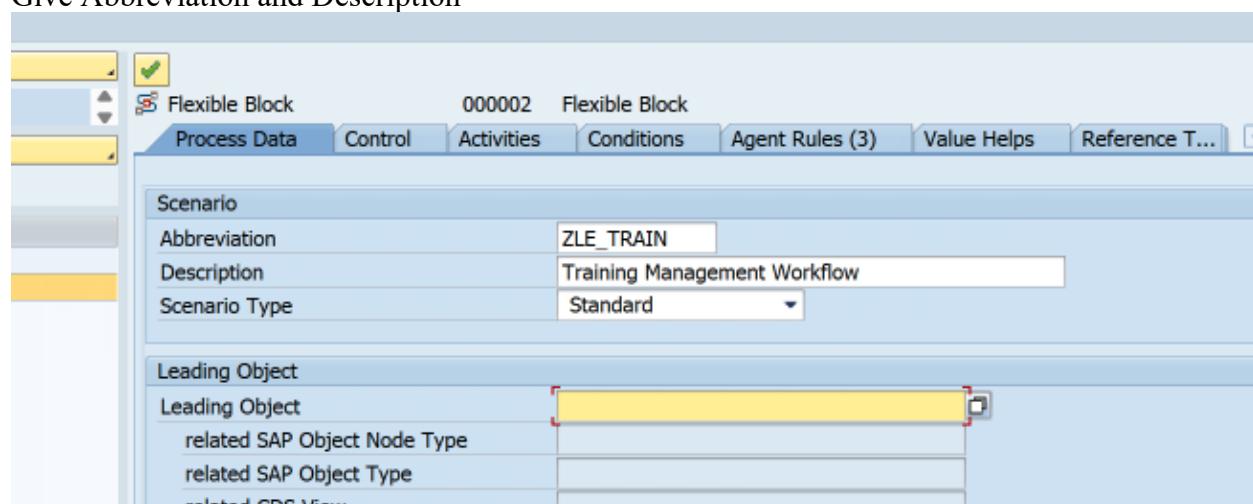
3.6. Creation of Workflow Template WS92000123

Got to TCODE SWDD_SCENARIO:

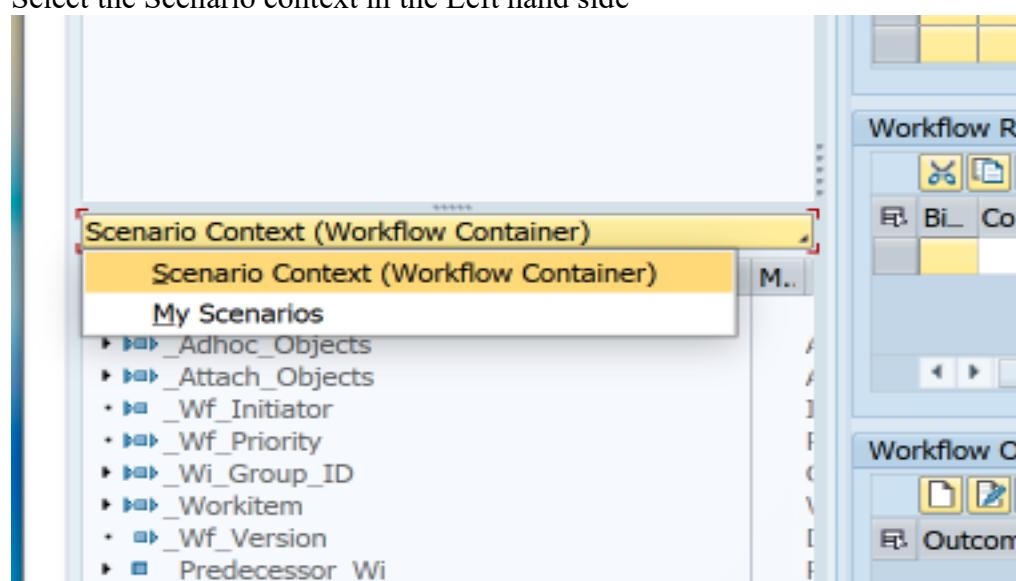
Click on Create New Workflow



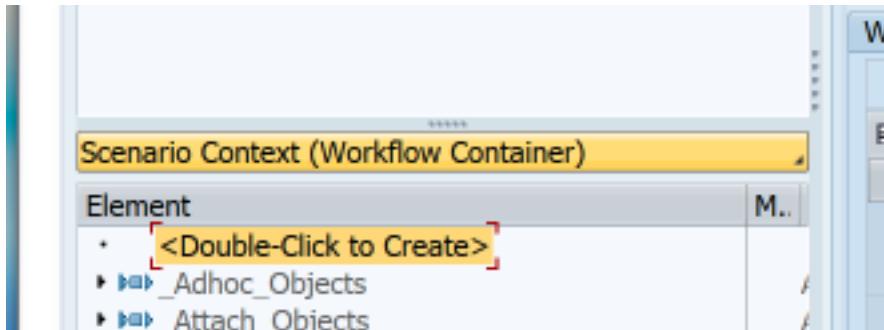
Give Abbreviation and Description



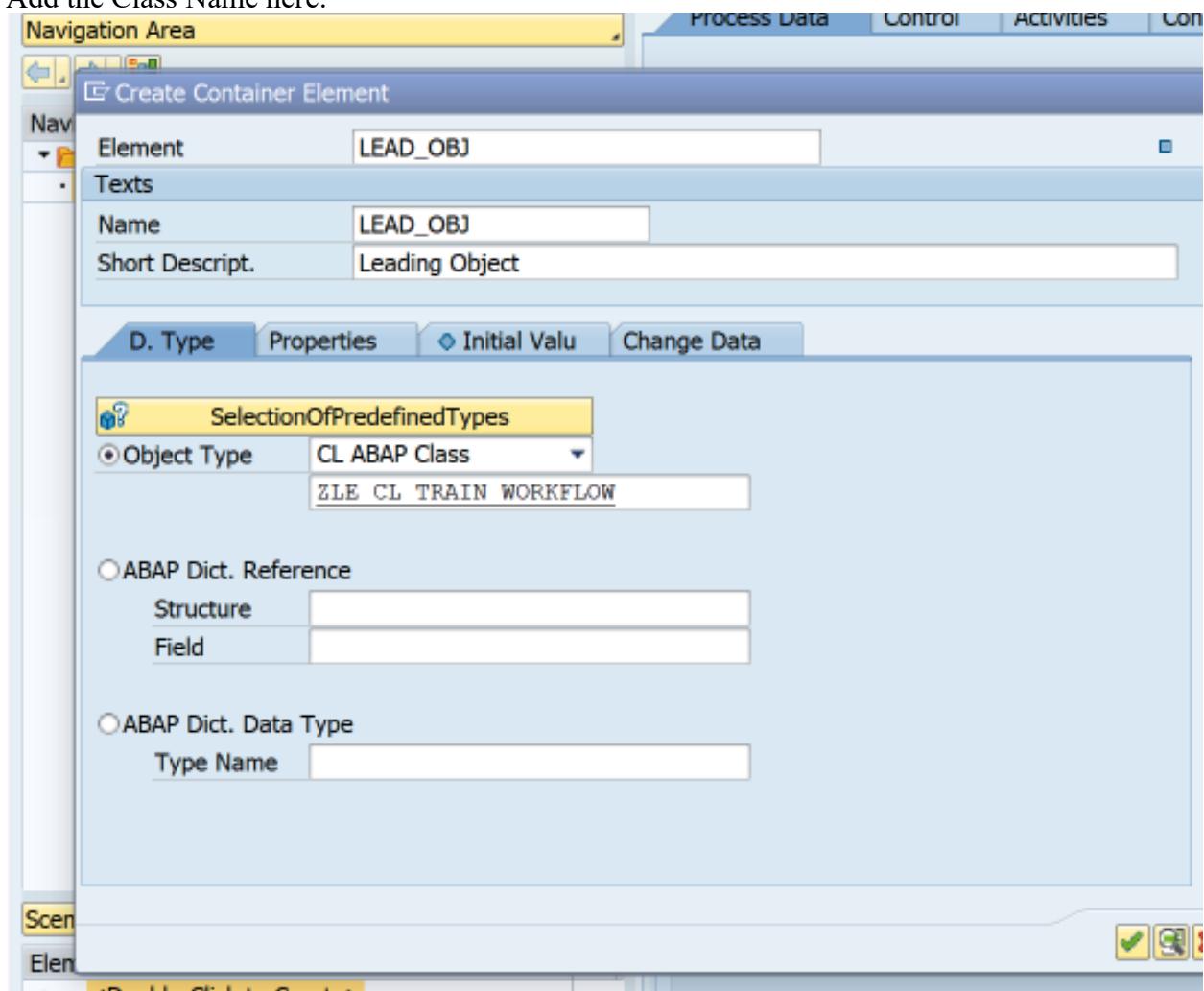
Select the Scenario context in the Left hand side



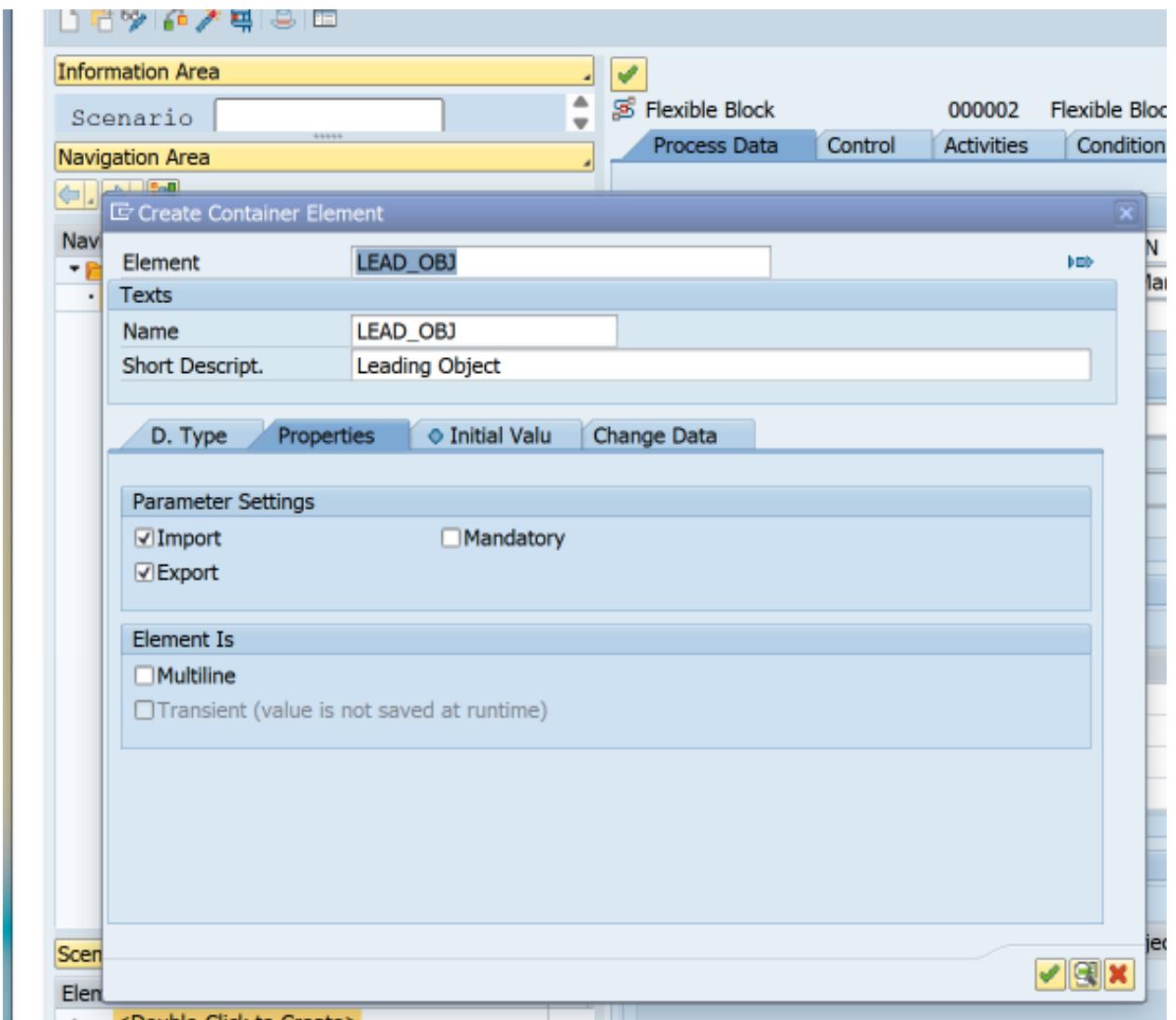
Create a new Container element called LEAD_OBJ for your workflow class using the Event Class we created above
 Double Click to Create:



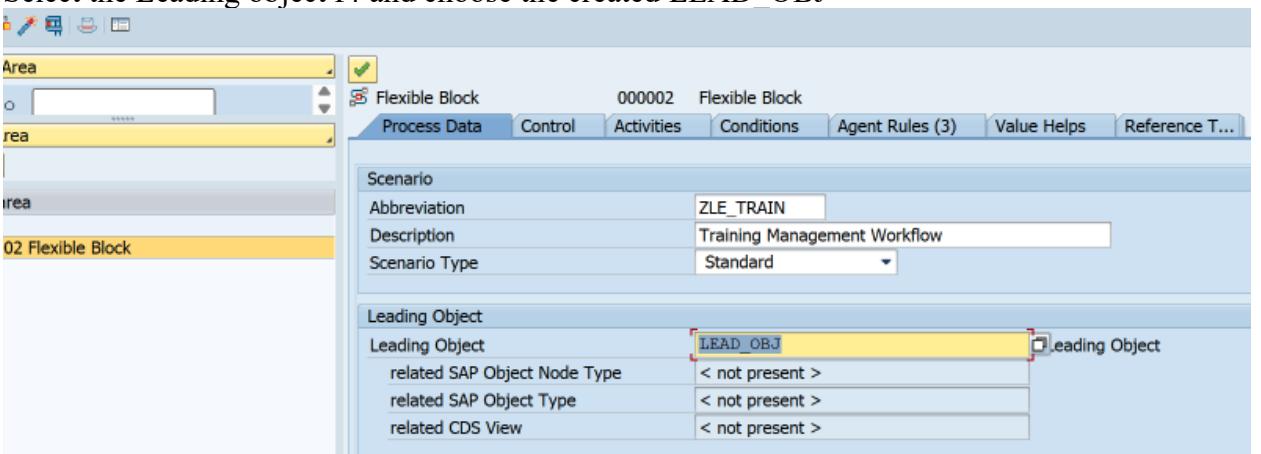
Add the Class Name here:



Go to Properties and Give import, export and select tick mark to confirm



Select the Leading object f4 and choose the created LEAD_OBJ



Create a new Container Object again for Training Input: LEARNING

Double Click to Create:

The screenshot shows a list of elements under 'Scenario Context (Workflow Container)'. One element, '<Double-Click to Create>', is highlighted with a red box.

- + <Double-Click to Create>
- + _Adhoc_Objects
- + _Attach_Objects
- + _Wf_Initiator
- + _Wf_Priority
- + _Wi_Group_ID
- + _Workitem
- + _Wf_Version
- + _Predecessor_Wi
- + LEAD_OBJ

Add the Element

The screenshot shows the 'Create Container Element' dialog box. The 'Element' field is set to 'LEARNING'. The 'D. Type' tab is selected, showing the 'SelectionOfPredefinedTypes' section. The 'Object Type' radio button is selected, and the dropdown menu shows 'BO BOR Object Type'. The 'ABAP Dict. Reference' section is also visible, showing fields for 'Structure' and 'Field'. The 'ABAP Dict. Data Type' section is selected, and the 'Type Name' field contains 'ZLE TRAIN HEAD'.

Go to Properties Tab : Give import and export

Create Container Element

Element	LEARNING
Texts	
Name	LEARNING
Short Descript.	Learning

D. Type Properties Initial Value Change Data

Parameter Settings

Import Mandatory
 Export

Element Is

Multiline
 Transient (value is not saved at runtime)

Create one more container element to store the current level using the above approach again

Create Container Element

Element	CurrentLevel
Texts	
Name	CurrentLevel
Short Descript.	CurrentLevel

D. Type Properties Initial Value Change Data

SelectionOfPredefinedTypes

Object Type BO BOR Object Type

ABAP Dict. Reference
Structure
Field

ABAP Dict. Data Type
Type Name int4

Go to Properties : Select import and export

The screenshot shows the SAP RAP Properties dialog for an element named 'CurrentLevel'. The 'Element' field is highlighted with a yellow background. The 'Texts' section contains 'Name: CurrentLevel' and 'Short Descript.: CurrentLevel'. Below the tabs 'D. Type', 'Properties', and 'Initial Valu' (which is selected), the 'Parameter Settings' section shows 'Import' checked and 'Mandatory' unchecked. Under 'Element Is', there is a dropdown menu.

Set initial value as 1 -> for every task generation is it incremented to identify the current level

The screenshot shows the SAP RAP Properties dialog for an element named 'CurrentLevel'. The 'Element' field is highlighted with a yellow background. The 'Texts' section contains 'Name: CurrentLevel' and 'Short Descript.: CurrentLevel'. Below the tabs 'D. Type', 'Properties', and 'Initial Valu' (which is selected), the 'Initial Valu' field is highlighted with a yellow background and contains the value '1'. The 'CurrentLev' part of the label is also highlighted with a yellow background.

Go back to Process Data and Give the Class and Event in the Workflow Start events and save the workflow but do not activate it. A new WS number will be generated: **WS92000123**

Ac...	Bi...	Co...	Category	Object Type	Event of the object
			CL	ZLE_CL_TRAIN_WORKFLOW	START_WF

Click on Bindings to do the bindings

Ac...	Bi...	Co...	Category	Object Type	Event of the object
			CL	ZLE_CL_TRAIN_WORKFLOW	START_WF

It is important to get the Learning Details from the event into workflow container

The screenshot shows the SAP Change Binding For Workflow dialog. It has three main sections:

- Event:** A tree view showing various system fields and a Container node. Under Container, there are fields like _EVT_OBJECT, _EVT_OBJTYPE, _EVT_NAME, _EVT_OBJKEY, _EVT_CREATOR, _EVT_RECEIVER_ID, _EVT_CREATION_DATE, _EVT_CREATION_TIME, _TCODE, _SOURCE_SYSID, _SOURCE_MANDT, and _EVT_RECEIVER_MODE, along with a LEARNING node.
- Description:** A column next to the Event fields providing descriptions.
- Workflow:** A tree view showing workflow objects. Under Workflow, there are Adhoc_Objects, Attach_Objects, Wf_Initiator, Wf_Priority, Wi_Group_ID, Workitem, LEAD_OBJ, and LEARNING.
- P.. Description:** A column next to the Workflow objects providing descriptions.

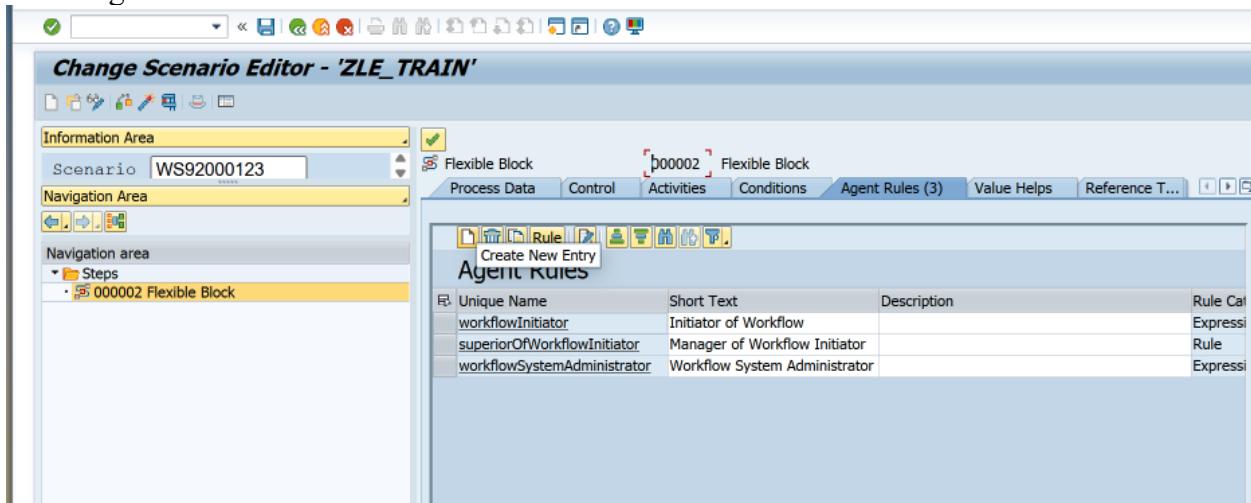
Below the main table, there is a toolbar with icons for generating automatic binding and other functions. The "Bindin" button is highlighted.

Go to the Control Tab Provide the Runtime and Definition class names that we created above

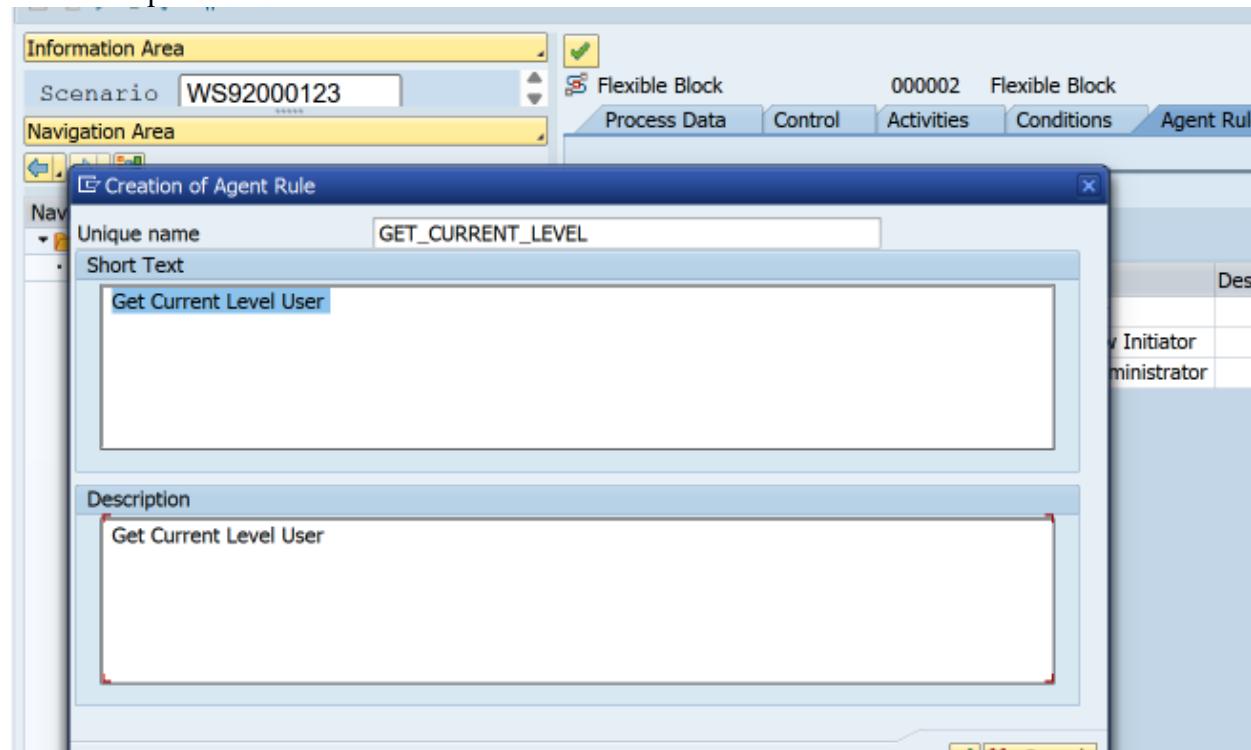
The screenshot shows the SAP Change Scenario Editor for scenario WS92000123. The navigation area shows a step named "000002 Flexible Block". The main area is the Control tab for this step, which includes the following sections:

- Flexible Block:** Shows the step name "000002 Flexible Block".
- Process Data:** Contains tabs for Activities, Conditions, Agent Rules (3), Value Helps, and Reference T...
- Definition Data:** Set to "ZCL_TRAIN_DEF_APPL_BASE".
- Runtime Data:** Set to "ZCL_TRAIN_RUN_APPL_BASE".
- Properties:** Includes checkboxes for Confirmation Obligatory, Allow Definition of Parallel Task Processing, and Enable Collaborative Attachment and Comment Handling.
- Pre-delivered Content (Process Description):** Buttons for Display, Import, Export, and Delete.

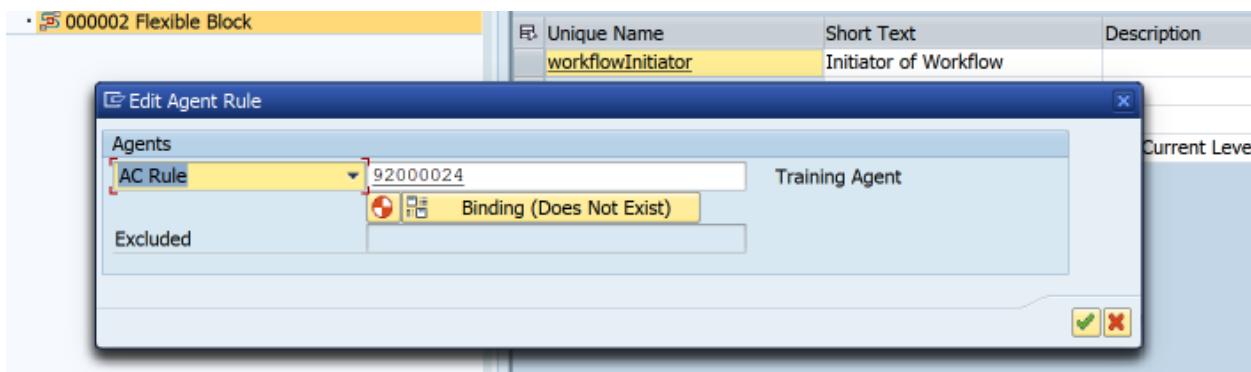
Go to Agent Rules click on create new rule:



Give a unique name



Select Rule in agents and give the Agent Rule number which we created above:



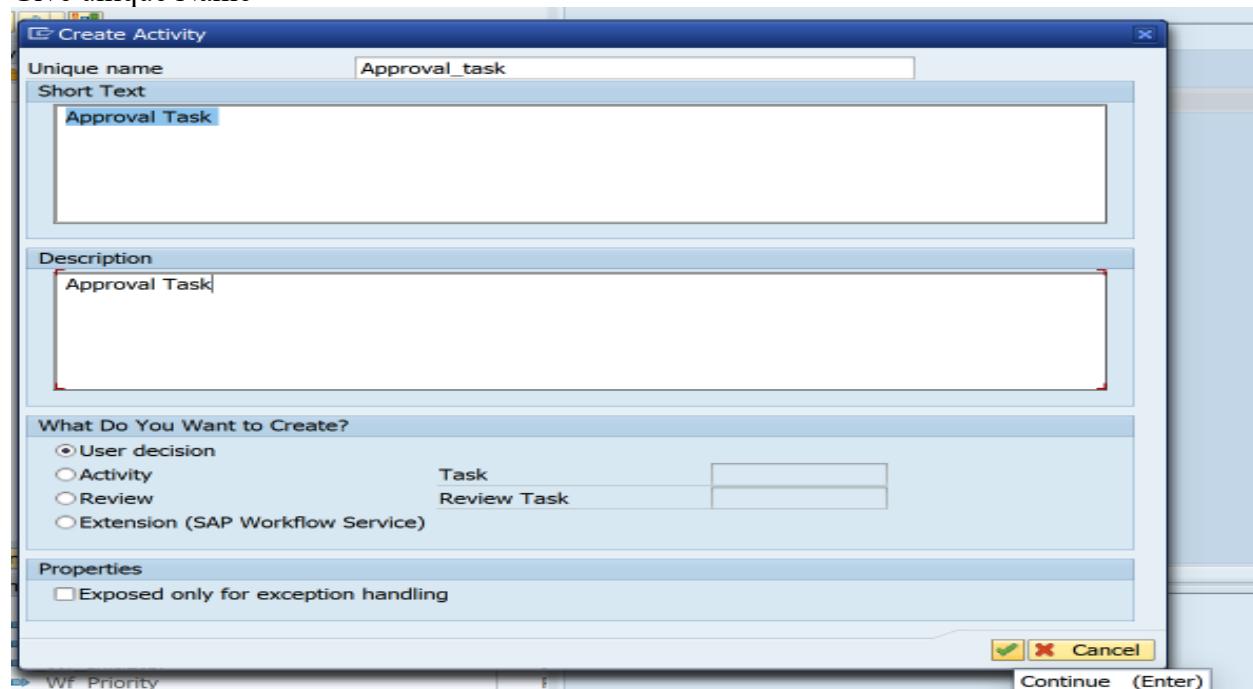
Click on the binding and map the Learning id and Current step:

Workflow 'Training Management Workflow'	Description
• _Wf_Version	Definition Version of this Workflow Insta
► _Predecessor_Wi	Previous Work Item
► LEAD_OBJ	Leading Object
- LEARNING	Learning
• CLIENT	CLIENT
• LEARNING_ID	Numeric Character Field, Length 10
• DESCRIPTION	DESCRIPTION
• COURSE_ID	COURSE_ID
• LEARN_STATUS_ID	LEARN_STATUS_ID
• TRAINER_UNAME	User who created the record

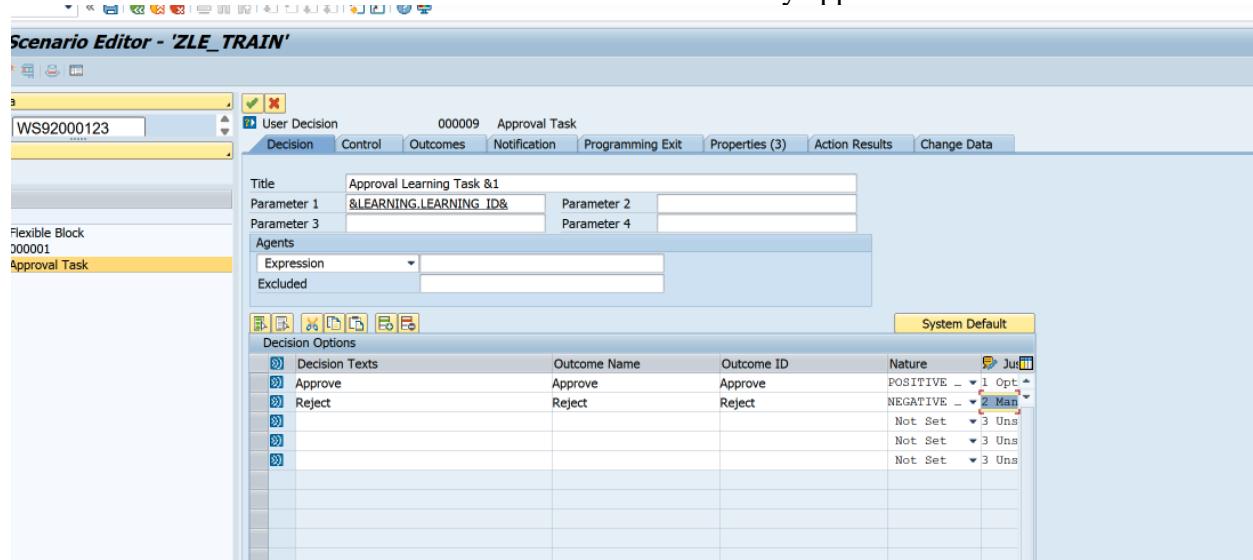
Rule 'Training Agent'	P..	Description
► System Fields		
► Container		
• LEARNID	LEARNID	
• CURRSTEP	CURRSTEP	
► _RULE_RESULT	Result	

Go to the Activities Tab and add the Approval step .This task is of type Decision Step

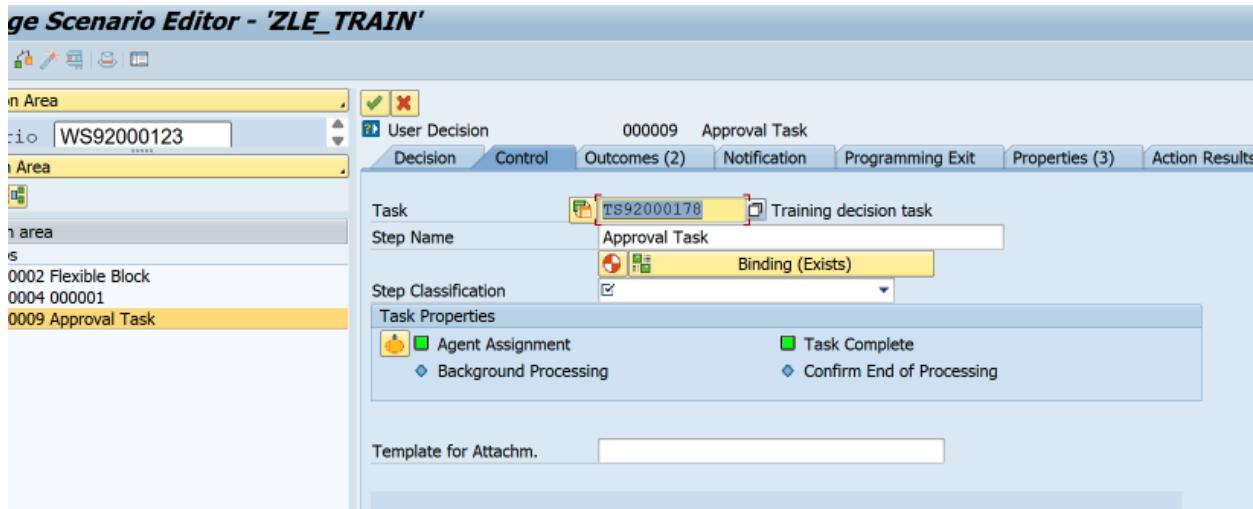
Give unique Name



Give the necessary Title with parameters, decisions and its nature and justifications as mentioned below. These are Decisions will be button or actions taken by approver



Go to Control and replace the standard task with Custom task we created before:



Click on the binding pass the learning id into the task and get the decision note from the task.

Workflow 'Training Management Workflow'	Description	Step 'Training decision task'	Description
↳ _Wi_Group_ID	Grouping Characteristic for Workflow Instance	↳ _Container	↳ Ad Hoc Objects of Workflow
↳ _Workitem	Workflow Instance	↳ _Adhoc_Objects	↳ Attachments of Workflow
↳ _Wf_Version	Definition Version of this Workflow Instance	↳ _Attach_Objects	↳ Actual Agent of Workflow
↳ _Predecessor_Wi	Previous Work Item	↳ _Wi_Actual_Agent	↳ Grouping Characteristic for Step Instance
↳ LEAD_OBJ	Leading Object	↳ _Rule_Result	↳ Result of Agent Determination
↳ LEARNING	Learning	↳ _Method_Objects	↳ Secondary Method Object
↳ CLIENT	CLIENT	↳ Extended	↳ Extended user decision
↳ LEARNING_ID	Numeric Character Field, Length 10	↳ Note_Reference	↳ SOFM template for notes
↳ DESCRIPTION	DESCRIPTION		
↳ COURSE_ID	COURSE_ID		

Save do a syntax check and activate the workflow:

Then we have to do the event type linkage to link the class event to the workflow
 Open a new session and go to TCODE SWE2 for event type linkage: map the workflow number with the class and event.

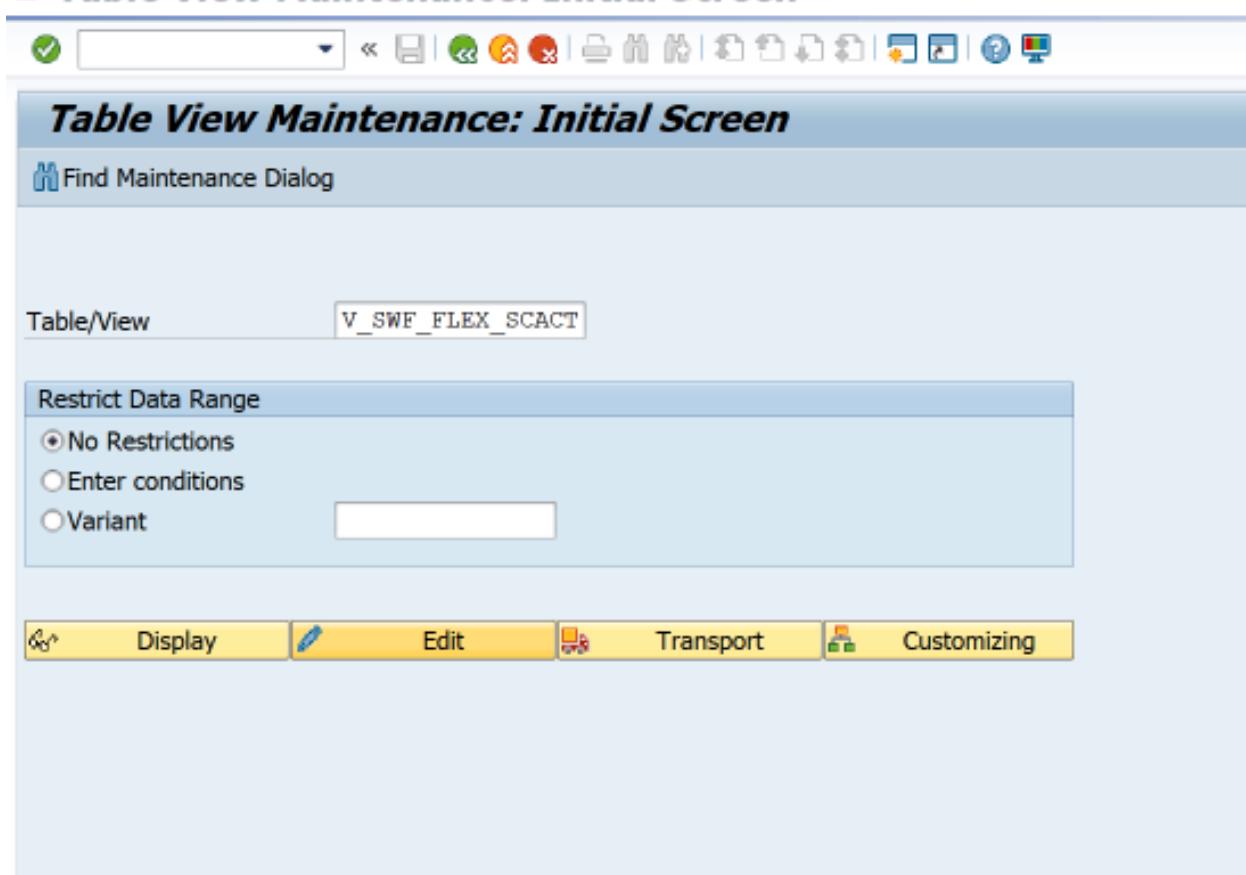
Make sure the Linkage Activated is selected

And click on add new entries and give the following and save it:

Change View "Event Type Linkages": Details of Selected Set

Object Category	CL ABAP Class
Object Type	ZLE_CL_TRAIN_WORKFLOW
Event	CREATED
Receiver Type	WS92000123
Linkage Setting (Event Receiver)	
Receiver Call	Function Module
Receiver Function Module	SWW_WI_CREATE_VIA_EVENT_IBF
Check Function Module	
Receiver Type Function Module	
Destination of Receiver	
Event delivery	Using tRFC (Default)
<input checked="" type="checkbox"/> Linkage Activated <input type="checkbox"/> Enable Event Queue	
Behavior Upon Error Feedback	0 System defaults
Receiver Status	0 No errors

Now go to the TCODE SM31: and edit this view V_SWF_FLEX_SCACT



Your workflow number will be there as active, if not add it using create new entries

WS92000116	<input type="checkbox"/>	ABAPER1	2025.09.08 03:25:42
WS92000117	<input type="checkbox"/>	ABAPER1	2025.09.07 17:51:03
WS92000123	<input checked="" type="checkbox"/>	K1537	2025.12.05 12:30:15

3.7. Configuration of Flexible Workflow

Let us Configure the workflow in the Manage workflow Fiori Application:

Open Fiori Launchpad and click on the Manage workflows (use Fiori app library for the necessary roles to enable the app for your role) app

Select the created workflow from the dropdown:

Click on Create new button to configure a new workflow:

Give workflow name and description:

Type	Name	Recipients	Step Conditions

Click on create in the steps, we can add the decisions steps we created in workflow template any number of times we want, this makes the workflow as flexible

Click on Create button in steps

Training Management Workflow / Training Workflow

Header Properties Start Conditions Steps

Steps

Workflow Steps

Type	Name	Recipients	Step Conditions
No data			

Create Delete ^ v

Give Level 1 as inputs and in the role give the created agent role in the workflow Template

Training Management Workflow / Training Workflow / Level 1

Approval Task

Header Recipients Deadlines Exception Handling

Step Name:
Level 1

Step Type:
Approval Task

Recipients

Role:
Get Current Level User

Step to be completed by:
 One of the recipients
 All of the recipients

Deadlines

Time Constraints

Time	Action
There are no deadlines defined.	

Create **Delete**

Exception Handling

Similar way create second step 2:

The screenshot shows the SAP S/4 HANA On Premise interface for creating a new workflow step. The top navigation bar includes the SAP logo, a back arrow, the text "New Step", and search and help icons. The main title is "Training Management Workflow / Training Workflow /". Below the title, the section "Level 2" is displayed under "Approval Task". A horizontal navigation bar at the top of the form includes tabs for "Header", "Recipients", "Deadlines", and "Exception Handling", with "Header" being the active tab.

Step Name: Level 2

Step Type: Approval Task

Recipients

Role: Get Current Level User

Step to be completed by:

- One of the recipients
- All of the recipients

Deadlines

Time Constraints Create

Time	Action
------	--------

Click on Save and Activate it.

The screenshot shows the SAP Workflow Details interface. At the top, there are several icons and a search bar. Below the header, the title 'Training Management Workflow / Training Workflow' is displayed. A navigation bar with tabs 'Properties', 'Start Conditions', and 'Steps' is visible, with 'Steps' being the active tab. Under the 'Steps' tab, a section titled 'Workflow Steps' lists two steps:

Type	Name	Recipients	Step Conditions
	1. Level 1	Get Current Level User	>
	2. Level 2	Get Current Level User	>

Final screen will look like this

The screenshot shows the SAP Manage Workflows interface. At the top, there is a search bar and a navigation bar with tabs 'Create', 'Copy', 'Deactivate', 'Define Order', and '...'. Below the header, a table displays the list of workflows:

Name	Order	Status	Valid From	Valid To
Training Workflow	1	Active		>

4. Integrate the Workflow in RAP Application

Then we need to integrate the workflow in behavior definition and implementation with a custom action button. This is already present in the above full code for behavior definition and implementation

4.1. Behavior Definition

A New Custom Action in behavior definition

```

B [AK4] ZL_ZLM_HEAD B [AK4] ZBL_TRAIN_HEAD X G [AK4] ZBP_I_ZLM_HEAD G [AK4] ZBP_BL_TRAIN_HEAD E [AK4] ZMDE_TRAIN_HEAD
1 managed implementation in class zbp.bi_train_head unique;
2 strict ( 2 );
3
4 define behavior for zbi_train_head alias train_head
5 persistent table ZLE_TRAIN_HEAD
6 lock master
7 authorization master ( instance )
8 //etag master <field_name>
9 early numbering
10 {
11   //Read only Fields
12   field ( readonly ) LearningId,LearnStatusId,CreatedBy,CreatedOn;
13   //Mandatory Fields
14   field ( mandatory ) CourseId,TrainerUname;
15
16   //Determinations
17   determination setStatusDraft on modify { create; }
18
19   //Validations
20   validation mandatoryCourseId on save { create; field CourseId; }
21   validation mandatoryTrainerName on save { create; field TrainerUname; }
22
23   //Actions
24   action ( features : instance ) sendApproval result [1] $self;
25
26
27   create;
28   update ( features : instance );
29   delete ( features : instance );
30   association TrainItem { create ( features : instance ) : }
```

4.2. Meta Data Extension

Enablement of Data Action in Metadata extension:

```

B [AK4] ZL_ZLM_HEAD B [AK4] ZBL_TRAIN_HEAD G [AK4] ZBP_I_ZLM_HEAD G [AK4] ZBP_BL_TRAIN_HEAD E [AK4] ZMDE_TRAIN_HEAD X G [AK4] ZSBIN_TRAIN
8
9 },
10 presentationVariant: [{ sortOrder: [{ by: 'LearningId', direction: '#DESC' }] } ]
11 @Search.searchable: true
12 annotate entity ZC0_TRAIN_HEAD with
13 {
14   //UI Facets
15   @UI.facet: [{ id:'Learning', purpose : #STANDARD, type :#IDENTIFICATION_REFERENCE, position : 10, lat
16     { id:'Items', purpose : #STANDARD, type :#LINEITEM_REFERENCE, position : 20, label: 'Item
17
18   @Search.defaultSearchElement: true
19   @Search.fuzzinessThreshold: 0.5
20   @UI.identification: [ { position: 10 } ]
21   @UI.lineItem: [ { position: 10 } ]
22   @EndUserText.label: 'Learning Id'
23   LearningId;
24
25   @Search.defaultSearchElement: true
26   @Search.fuzzinessThreshold: 0.7
27   @UI.identification: [ { position: 20 } ]
28   @UI.lineItem: [ { position: 20 } ]
29   @EndUserText.label: 'Description'
30   Description;
31
32   @UI.identification: [ { position: 30 } ]
33   @UI.lineItem: [ { position: 30 } ]
34   @EndUserText.label: 'Course'
35   @Consumption.valueHelpDefinition: [{ entity:{ name: 'ZLE_COURSE_VH', element: 'CourseId' }}]
36   @UI.selectionField: [ { position : 30 }]
37   CourseId;
38   @UI.identification: [ { position: 40, criticality: 'StatusCriticality' }]
39   @UI.lineItem: [ { position: 40, criticality: 'StatusCriticality' },
40     { type : #FOR_ACTION, dataAction: 'sendApproval', label: 'Send Approval' }]
41   @EndUserText.label: 'Learning Status'
42   @Consumption.valueHelpDefinition: [{ entity:{ name: 'ZLE_STATUS_VH', element: 'StatusId' }}]
43   @UI.selectionField: [ { position : 40 }]
44

```

4.3. Class Implementation

Implementation of Custom Action in the behavior implementation class to trigger the workflow

```

B [AK4] ZI_ZLM_HEAD   B [AK4] ZBI_TRAIN_HEAD   C [AK4] ZBP_I_ZLM_HEAD   C [AK4] ZBP_BI_TRAIN_HEAD X E [AK4] ZMDE_TRAIN_HEAD   S [AK4]
D ZBP_BI_TRAIN_HEAD D LHC_TRAIN_HEAD D SENDAPPROVAL

314
315 METHOD sendApproval.
316     ""-->Read the Selected Data
317     READ ENTITIES OF zbi_train_head IN LOCAL MODE
318         ENTITY train_head BY \_TrainItem
319             ALL FIELDS
320             WITH CORRESPONDING #( keys )
321             RESULT DATA(lt_items).
322
323     READ ENTITY IN LOCAL MODE zbi_train_item
324         BY \_TrainHead
325             FIELDS ( learningid learnstatusid )
326             WITH CORRESPONDING #( lt_items )
327             RESULT DATA(lt_heads).
328
329     DATA(ls_head) = VALUE #( lt_heads[ 1 ] OPTIONAL ).
330     ""-->Only Learning ID is Enough for us in the structure
331     DATA(ls_learning) = VALUE zle_train_head( learning_id = ls_head-LearningId ).  

332
333     ""-->Trigger Workflow ***DO NOT USE COMMIT in any place as it is not allowed ***
334     DATA :lo_wf TYPE REF TO zle_cl_train_workflow.
335     CREATE OBJECT lo_wf
336         EXPORTING
337             lv_key_number = ls_learning-learning_id.
338
339     CALL METHOD lo_wf->trigger_start_wf
340         EXPORTING
341             learning_det = ls_learning.  

342
343     ""-->Update the status back
344     MODIFY ENTITIES OF zbi_train_head IN LOCAL MODE
345         ENTITY train_head
346             UPDATE FIELDS ( LearnStatusId )
347                 WITH VALUE #( FOR ls_keys IN keys ( %tky = ls_keys-%tky
348                             LearnStatusId = 'I' ) ).  

349
350     READ ENTITIES OF zbi_train_head IN LOCAL MODE
351         ENTITY train_head
352             ALL FIELDS WITH CORRESPONDING #( keys )
353             RESULT DATA(lt_head_r)
354             FAILED DATA(lt_failed).
355
356     result = VALUE #( FOR ls_head_r IN lt_head_r ( %tky      = ls_head_r-%tky
357                               %param = ls_head_r ) ).  

358
359
360 ENDMETHOD.

```

4.4. Behavior Definition Consumption View

Give the action in the behavior definition of Consumption view ZCO_TRAIN_HEAD

```

1 projection;
2 strict ( 2 );
3
4@define behavior for ZCO_TRAIN_HEAD alias train_head
5 {
6   use create;
7   use update;
8   use delete;
9   use action sendApproval;
0
.1   use association _TrainItem { create; }
.2 }
.3
.4@define behavior for ZCO_TRAIN_ITEM alias train_item
5 {
6   use update;
7   use delete;
8
.9   use association _TrainHead;
0 }
```

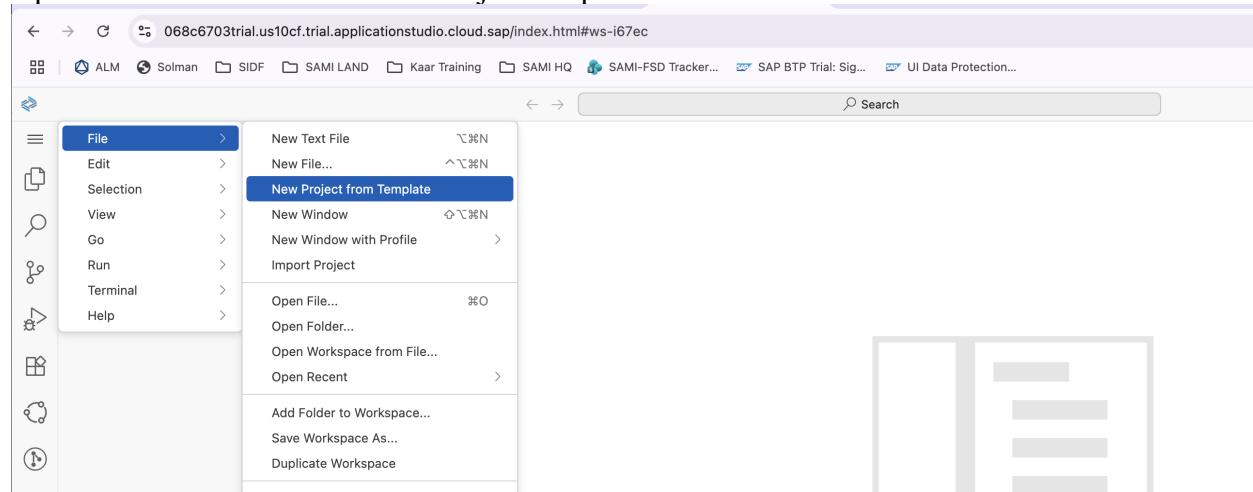
5. BAS: Create and Deploy Fiori Application

5.1. Creation of Fiori Application

Here I am using the Trail version and have the cloud connector ready which is connected to my on-premise system

Let us Create a Fiori Application in Business Application Studio.

Open the BAS and Create a new Project template



Select the Following to Create a new Project with list template

Project From Template X

SAP Fiori generator ⓘ

Project Attributes

Configure the main project attributes.

Select Template and Target Location

SAP Fiori generator

Template Selection

Template: List Report Page

Data Source and Service Selection

Data Source: Connect to a System
System: AK4
Service: ZSBIN_TRAIN_HEAD (1) - OData V2

Entity Selection

Main Entity: ZCO_TRAIN_HEAD
Navigation Entity: to_TrainItem
Table Type: Responsive

Project Attributes

Module Name: zle_emp_training
Application Title: Employee Training
Application Namespace: com.le.train
Description: Employee Training Management
[More...](#)

Module Name ⓘ *

zle_emp_training

Application Title ⓘ

Employee Training

Application Namespace ⓘ

com.le.train

Description ⓘ

Employee Training Management

Project Folder Path *

/home/user/projects

Minimum SAPUI5 Version ⓘ *

1.108.40 (Source system version)

Enable TypeScript

Yes No

Add Deployment Configuration

Yes No

Add SAP Fiori Launchpad Configuration

Back ⏪ Finish ⓘ The generated project will not open in a stand-alone.

Once the project is Created do a preview to check whether the application is working

Trainings (7)				
Learning Id	Description	Course	Learning Status	Trainer Name
1000000006	asdfsdfsdf	ABAP (ABA)	⚠ In progress (I)	Dinesh Kumar (K1537)
1000000005	sadssdsd	ABAP Managed Database Procedures (AMP)	⚠ In progress (I)	Dinesh Kumar (K1537)
1000000004	Test	ABAP (ABA)	⚠ In progress (I)	Test Meka (ABAPER1)
1000000003	test	Fiori and UI5 (FIO)	⚠ In progress (I)	Dinesh Kumar (K1537)
1000000002	New wf	ABAP Managed Database Procedures (AMP)	⚠ In progress (I)	Dinesh Kumar (K1537)
1000000001	second	ABAP (ABA)	⚠ In progress (I)	Dinesh Kumar (K1537)
1000000000	Test1	ABAP (ABA)	⚠ In progress (I)	Dinesh Kumar (K1537)

5.2. Deploy Fiori Application

We need to do the following commands to deploy application to on premise system

```
user: zle_emp_training $ npx fiori add deploy-config
Adding deploy-config to the project.
warn Add:Deploy-config No VSCode Settings file found.
info Add:Deploy-config Using: @sap/fiori:deploy-config
? Please choose the target. ABAP
? Destination AK4 - [REDACTED]
? SAPUI5 ABAP Repository ZLE_EMP_TRAIN
? Deployment Description Employee Training
? Select How You Want to Enter the Package Enter Manually
? Package Z1537_TRAIN
? Select How You Want to Enter the Transport Request Enter Manually
? Transport Request AK4K901804
  force package.json
  create ui5-deploy.yaml

user: zle_emp_training $ npm run deploy

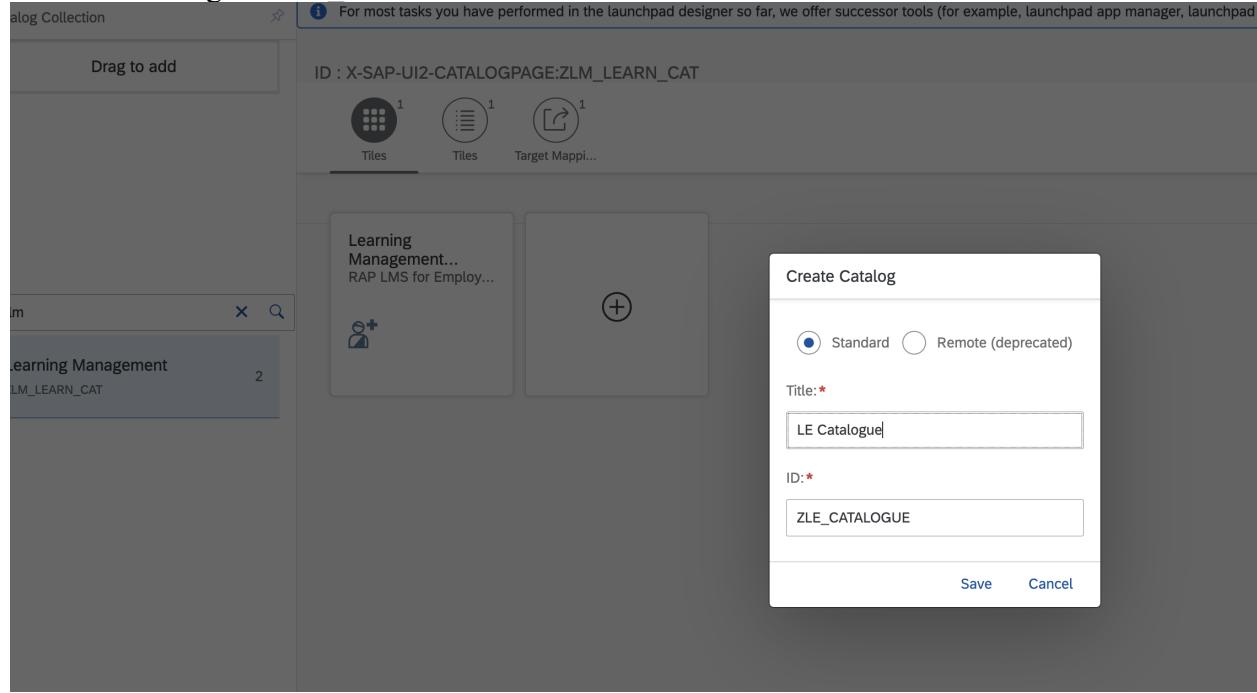
info abap-deploy-task ZLE_EMP_TRAIN Deployment Successful.
info abap-deploy-task ZLE_EMP_TRAIN (Note: As the destination is configure
```

6. Add Fiori Application in Launchpad

6.1. Create Catalogue

Open Fiori Launchpad Customization: /n/ui2/flpd_cust

Create a Catalogue: ZLE CATALOGUE



Create Target Mapping:

Intent		Target													
Semantic Object:	ZLE_SEM_OBJ_TRAIN	Application Type:	SAPUI5 Fiori App												
Action:	manage	Title:	Training Management												
		URL:	/sap/bc/ui5_ui5/sap/zle_emp_train												
		ID:	com.te.train.zleemptraining												
General															
Information: <input type="text"/> Device Types: <input checked="" type="checkbox"/> Desktop <input checked="" type="checkbox"/> Tablet <input checked="" type="checkbox"/> Phone Parameters: <table border="1"> <tr> <th>Name</th> <th>Mandatory</th> <th>Value</th> <th>Is Regular Expression</th> <th>Default Value</th> <th>Target N...</th> </tr> <tr> <td>openMode</td> <td><input type="checkbox"/></td> <td>embedIntoDetailsNestedRouter</td> <td><input type="checkbox"/></td> <td>embedIntoDetailsNestedRouter</td> <td><input type="checkbox"/></td> </tr> </table>				Name	Mandatory	Value	Is Regular Expression	Default Value	Target N...	openMode	<input type="checkbox"/>	embedIntoDetailsNestedRouter	<input type="checkbox"/>	embedIntoDetailsNestedRouter	<input type="checkbox"/>
Name	Mandatory	Value	Is Regular Expression	Default Value	Target N...										
openMode	<input type="checkbox"/>	embedIntoDetailsNestedRouter	<input type="checkbox"/>	embedIntoDetailsNestedRouter	<input type="checkbox"/>										

It is very important to give the Parameters as OpenMode and Value As embedIntoDetailsNestedRouter to enable the detail page of the application inside the common inbox details page in Fiori.

Create a New Tile for the application:

Configure: 'Training Management'

Instance ID: 403CETYKK5UP5C372T2KU4U8

General	Navigation
Title: Training Management	Use semantic object navigation: <input checked="" type="checkbox"/>
Subtitle:	Semantic Object: ZLE_SEM_OBJ_TRAIN
Keywords:	Action: manage
Icon: sap-icon://inbox	Parameters:
Information:	Target URL: #ZLE_SEM_OBJ_TRAIN-manage

Title Actions

6.2. Create Group

Create a Group: ZLE_GRP (Newer way is to use SPACES and PAGES but here since it is a sample I go with older way of groups)

The screenshot shows the SAP Launchpad Designer interface. On the left, there's a sidebar with 'Catalogs' and 'Groups'. Under 'Groups', there's a list of existing groups like 'UI2/ZMA_TEST_PAGE_GERMAN' (2), 'Pricing Configuration' (1), 'Pricing Inbox' (1), etc. In the center, there's a message about successor tools. On the right, a modal window titled 'Create Group' is open. It has fields for 'Title:' (set to 'LE : Training Management') and 'ID:' (set to 'ZLE_GRP'). There's also a checkbox for 'Group personalization' which is checked. At the bottom of the modal are 'Save' and 'Cancel' buttons.

Select the group and inside select the catalog and add the tile

[Add Tile to Group 'LE : Training Management'](#)

LE Catalogue

Training Management

+

Catalogs Groups

Search for groups

LE : Training Management

ID : ZLE_GRP

Show as Tiles

Training Management

+

Show as Links

For most tasks you have performed in the launchpad designer so far, we offer successor tools (for example)

Group Name	Count
LE : Training Management	1
Learning Management	1
Linear Reference Pattern	3
Liquidity Management	5
LiveCache Administration	6

You can either create new role or assign this group and catalogue in existing role: Take support of basis to do this

Now when you log into the Launchpad you can access the application

SAP RAP and Flexible Workflow in S/4 HANA On Premise

The screenshot displays two SAP S/4 HANA On Premise interfaces:

Top Interface (Training Management):

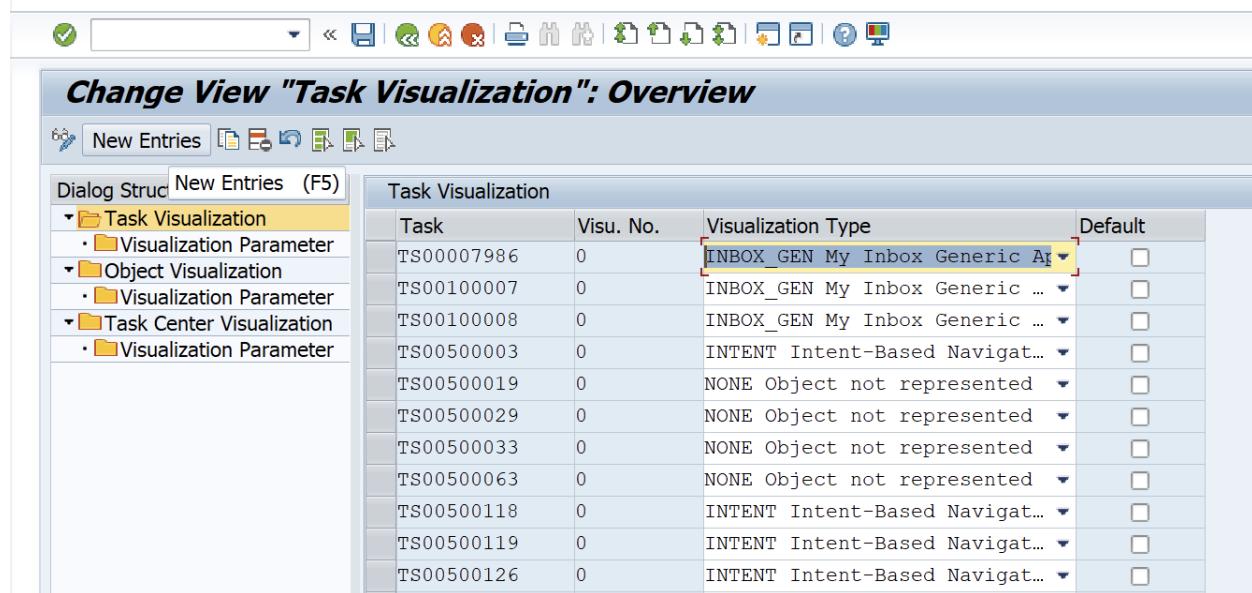
- Header: Training Management, Solman, SIDF, SAMI LAND, Kaar Training, SAMI HQ, SAMI-FSD Tracker..., SAP BTP Trial: Sig..., UI Data Protection...
- Navigation: Home, Inspection Methods Display, Inspection Planning Display, Master Inspection Characteristics Display, Quality Tasks, Results Recording, Sample Manage.
- Main Content:
 - LE : Training Management
 - Training Management (boxed)

Bottom Interface (Employee Training):

- Header: SAP, Employee Training, Standard*.
- Search Bar: Search, Go, Adapt Filters.
- Filter Options: Course: (dropdown), Learning Status: (dropdown).
- Buttons: Send Approval, Create, Delete, Print, Copy, Paste.
- Table Headers: Trainings, Learning Id, Description, Course, Learning Status, Trainer Name.
- Text: To start, set the relevant filters.

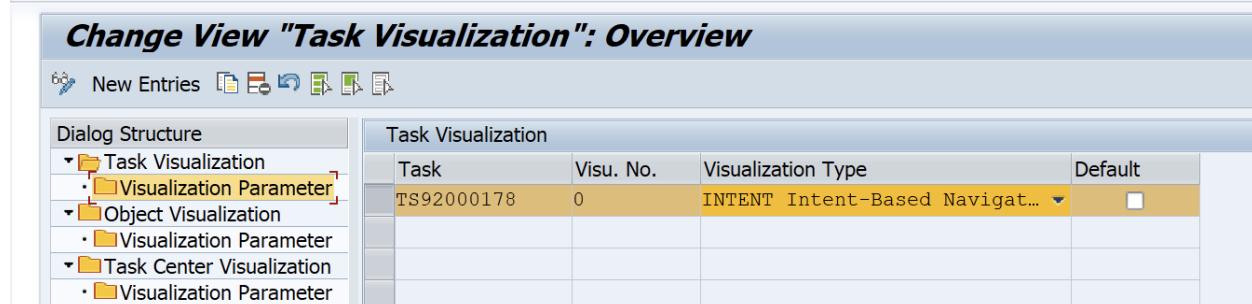
7. Visualization Configuration for Tasks

Configure Visualization parameter to enable the RAP Detail view inside Inbox Detail view.
 Go to TCODE: SWFVISU
 Click on create New Entries:



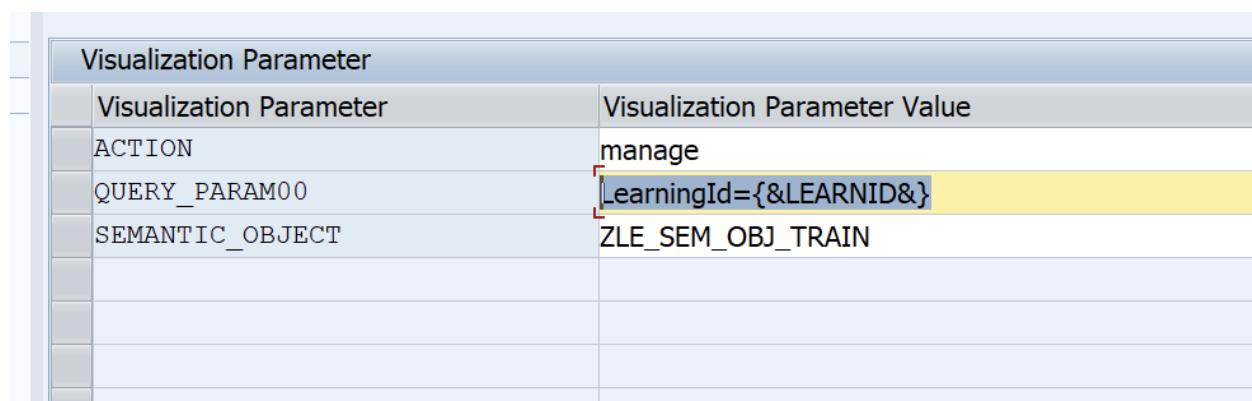
Task	Visu. No.	Visualization Type	Default
TS000007986	0	INBOX_GEN My Inbox Generic App	<input type="checkbox"/>
TS00100007	0	INBOX_GEN My Inbox Generic ...	<input type="checkbox"/>
TS00100008	0	INBOX_GEN My Inbox Generic ...	<input type="checkbox"/>
TS00500003	0	INTENT Intent-Based Navigat...	<input type="checkbox"/>
TS00500019	0	NONE Object not represented	<input type="checkbox"/>
TS00500029	0	NONE Object not represented	<input type="checkbox"/>
TS00500033	0	NONE Object not represented	<input type="checkbox"/>
TS00500063	0	NONE Object not represented	<input type="checkbox"/>
TS00500118	0	INTENT Intent-Based Navigat...	<input type="checkbox"/>
TS00500119	0	INTENT Intent-Based Navigat...	<input type="checkbox"/>
TS00500126	0	INTENT Intent-Based Navigat...	<input type="checkbox"/>

Give the approval task number that we created in PFTC and the following values



Task	Visu. No.	Visualization Type	Default
TS92000178	0	INTENT Intent-Based Navigation	<input type="checkbox"/>

Select the Task number and double click visualization parameter



Visualization Parameter	
Visualization Parameter	Visualization Parameter Value
ACTION	manage
QUERY_PARAM00	LearningId={&LEARNID&}
SEMANTIC_OBJECT	ZLE_SEM_OBJ_TRAIN

SEMANTIC_OBJECT and ACTION is what we created in the Target Mapping during the Fiori launchpad catalogue

QUERY_PARAM00 is the key field LearningId that we have in the Base Interface View and the Assigned values is the container parameter we created in the workflow task binding.

Note: Parameters here are case sensitive be careful when filling it.

The screenshot shows the SAP RAP interface for a 'Standard Task: Display'. The top menu bar includes Standard task, Edit, Goto, Additional data, Utilities, and System. Below the menu is a toolbar with various icons. The main title is 'Standard Task: Display'.

Basic Data:

Standard task	92000178	ZLE_DEC
Name	Training decision task	
Package	Z1537_TRAIN	Appli

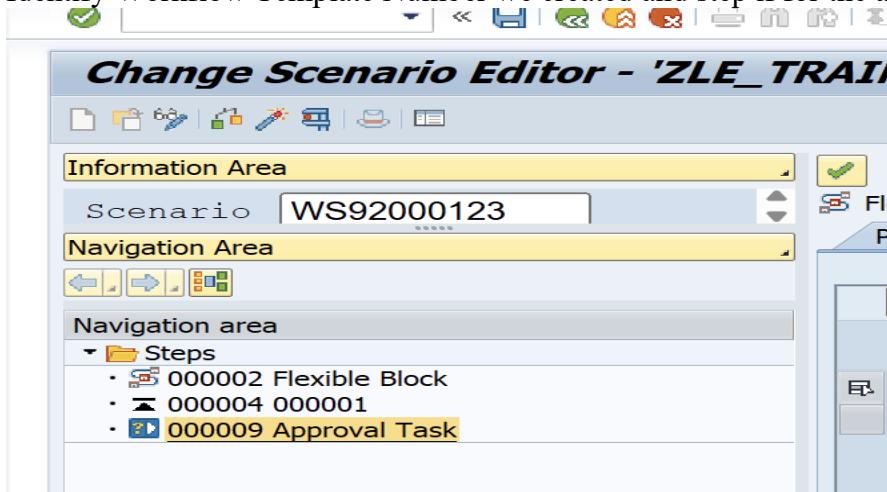
Expression Mapping:

Expression	M..	Description
Ad Hoc Objects		Ad Hoc Objects of Workflow Instance
Attachments		Attachments of Workflow Instance
Agent		Actual Agent of Workflow Activity
Grouping Character		Grouping Characteristic for Workflow
Step		Step Instance
Agents		Result of Agent Determination
Second.Method Object		Secondary Method Objects of Workflow
Ext. user decision		Extended user decision
Template for notes		SOFM template for notes
_Wi_Object_ID		_Wi_Object_ID
Decision_Note		Decision_Note
LEARNID		learning ID
_WI_Result		_WI_Result

Save the values and close

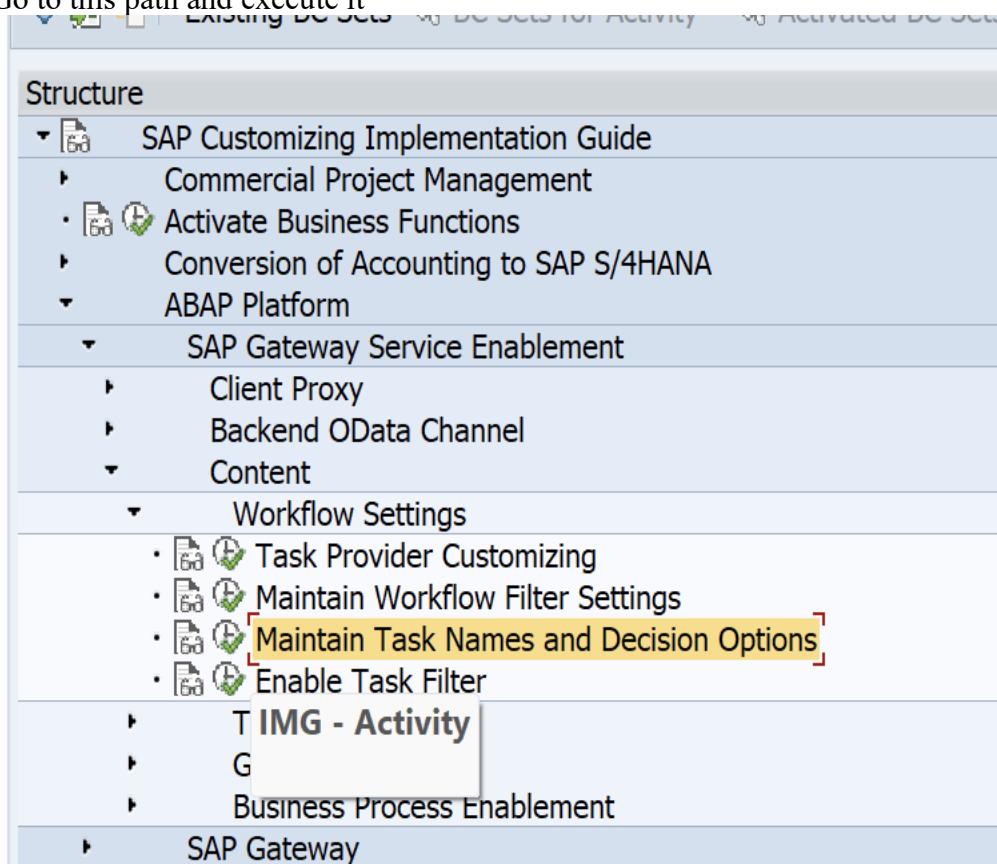
8. Task Decision Button Configuration

Configure Task Decision Buttons to show the Approve and Reject Button in approval task:
Identify Workflow Template Number we created and step id for the approval task.



Go to TCODE SPRO and select SAP Reference IMG

Go to this path and execute it



Give the following Workflow Id And step id, this step id we can take from SWDD_SCENARIO workflow template

Step Name				Step Description
	Workflow ID	Step ID	Icon MIME Repository Path	
	WS92000123	000000009		Training Approval Task

Select this line and double click the Decision Keys:

Step Name				Step Description
	Workflow ID	Step ID	Icon MIME Repository Path	
	WS92000123	000000009		Training Approval Task

Provide Button details:

0001 for Approve and 0002 for Reject with Positive and Negative nature

Decision Keys					
Key	Icon MIME Repository Path	Decision Text	Note	Nature	
0001		Approve	Optional	POSITIVE	
0002		Reject	X Mandatory	NEGATIVE	

Save the entries and close the TCODE:

9. Testing the Application

Now we are all done, let us check the Application

9.1. Approve Scenario

Open application and click on Create Button

The screenshot shows the SAP Employee Training application. At the top, there is a search bar and a 'Go' button. Below the search bar, there are two input fields: 'Course:' and 'Learning Status:', each with a dropdown arrow. To the right of these fields is a 'Send Approval' button, followed by a large blue 'Create' button with the text 'Ctrl+Enter' below it. To the right of the 'Create' button are 'Delete', 'Settings', and 'Print' buttons. The main area is titled 'Trainings' and contains a table with columns: Learning Id, Description, Course, and Learning Status. A message 'No data found.' is displayed at the bottom of the table.

Select the necessary header values using f4 help and give the header inputs and click on create

The screenshot shows the SAP Training application. The title bar says 'Training'. The main area is titled 'Learning' and contains a form with fields: 'Learning Id' (with a value of '-'), 'Course:' (set to 'ABA'), 'Trainer Name:' (set to 'K1537'), 'Description' (set to 'Learning ABAP'), and 'Learning Status' (set to '-'). At the bottom right are buttons for 'Ctrl+S', 'Create' (highlighted), and 'Cancel'.

Create Items:

The screenshot shows the SAP Training application. The title bar says 'Training'. The main area shows a learning item with ID '1000000014'. The details for this item are: 'Learning Status: Draft' and 'Trainer Name: Dinesh Kumar (K1537)'. Below this, there is a table for 'Items' with columns: Item Id, Description, and Hours. A message 'No items available.' is displayed. At the bottom right are buttons for 'Edit', 'Delete', 'Create' (highlighted), 'Ctrl+Enter', 'Delete', 'Settings', and 'Print'.

I have added two items

1000000014

Edit **Delete**

Learning **Items**

Learning Id: 1000000014 Learning Status: Draft

Description: Learning ABAP Trainer Name: Dinesh Kumar (K1537)

Course: ABAP (ABA)

Items (2) Standard

Item Id	Description	Hours	
10	Item 1	2 >	
20	Item 2	4 >	

You can create any number of items in draft state

Standard

Course: Learning Status:

Trainings (2)

Send Approval **Create** Delete

Learning Id	Description	Course	Learning Status	Trainer Name
1000000015	Test 2	Core Data Service (CDS)	Draft (D)	Dinesh Kumar (K1537) >
1000000014	Learning ABAP	ABAP (ABA)	Draft (D)	Dinesh Kumar (K1537) >

Now Select one to enable the send for approval

The screenshot shows the SAP Employee Training application interface. At the top, there's a search bar with 'Search' and a 'Go' button, along with filter options. Below the search bar, there's a table titled 'Trainings (2)' with columns: Learning Id, Description, Course, Learning Status, and Trainer Name. Two entries are listed:

Learning Id	Description	Course	Learning Status	Trainer Name
1000000015	Test 2	Core Data Service (CDS)	Draft (D)	Dinesh Kumar (K1537)
1000000014	Learning ABAP	ABAP (ABA)	In progress (I)	Dinesh Kumar (K1537)

At the bottom right of the table, there are buttons for 'Send Approval', 'Create', 'Delete', and other actions.

Now the status is changed as in progress

This screenshot is identical to the one above, but the status for the second entry ('Learning ABAP') has been updated to 'In progress (I)', indicated by a yellow warning icon.

Workflow Log table is updated:

The screenshot shows the SAP Workflow Log table. The search bar at the top contains 'ZLE_TRAIN_WF_TRA'. The table has columns: Length, Creator Stat., Time, Da..., ID, Date, and Time. There are two rows of data:

Length	Creator Stat.	Time	Da...	ID	Date	Time
1000000014	2 K1537	00:00:00				00:00:00
1000000014	1 K1537 I	00:00:00	4112...	2025.12.06		15:29:20

Email is sent for approver:

The screenshot shows an email message with the subject 'Display Document: Please Approve Training Request(1000000014)'. The message body contains the text 'Please Approve Training Request(1000000014)'. At the bottom, there's a note: 'Dear Dinesh Kumar, Training Request 1000000014 is sent for your Approval'.

Now the Go to the user inbox where we can see the detail page of the RAP Application embedded for the selected Learning ID

SAP RAP and Flexible Workflow in S/4 HANA On Premise

The screenshot shows a SAP RAP interface for a 'Training Approval Task' with ID 1000000014. The task is categorized under 'Approval Learning Task' and is marked as 'Medium'. The 'Learning' tab is selected, showing details like Learning Id: 1000000014, Course: ABAP (ABA), Trainer Name: Dinesh Kumar (K1537), and Learning Status: In progress. Below this, a table lists 'Items (2) Standard' with two entries: Item 10 (Item 1, 2 hours) and Item 20 (Item 2, 4 hours). At the bottom right are buttons for Approve, Reject, Show Log, Show Details, Claim, Forward, Suspend, Open Task, and a copy icon.

Approve the Request to send for second level:

The screenshot shows the same SAP RAP interface for the same task. A modal dialog box titled 'Submit Decision' is open, displaying the message 'You selected "Approve".' and a 'Decision Note' field containing 'Approved Level 1'. There are 'Submit' and 'Cancel' buttons at the bottom of the dialog.

Log table is updated

ZLE_TRAIN_WF_TRA: Display of Entries Found													
Search in Table					ZLE_TRAIN_WF_TRA		Training Workflow for approval						
Number of Hits					2								
Runtime					0		Maximum No. of Hits		500				
Insert Column													
	Length	10	IN...	Creat...	Stat...	Time	Date	ID	Date	Time			
	1000000014	1	K1537	A	15:31:50	2025.12.06	411228	2025.12.06	15:29:20				
	1000000014	2	K1537	I	00:00:00		411229	2025.12.06	15:31:50				

Email is sent for next approver:

Period	Send Status	Sender	Options	Attribs.			
<input checked="" type="checkbox"/> Waiting	<button>Further...</button>						
<input checked="" type="checkbox"/> Errors							
<input checked="" type="checkbox"/> Sent	<input checked="" type="checkbox"/> Transmitted						
<input type="button" value="Refresh"/>							
<input type="button" value="Status Text"/>							
All Send Requests							
Status	Send Method	Doc. Title	Sender	Recipient	Send Date	Send Time	Msg
	By E-Mail	Please Approve Training Request(1000000014)	K1537	TESTEMAIL@GMAIL.COM	2025.12.06	14:01:51	672
	Waiting	Please Approve Training Request(1000000014)	K1537	TESTEMAIL@GMAIL.COM	2025.12.06	13:59:25	672

Now Go to inbox of Level 2 Approver and approve:

The screenshot shows the SAP Training application interface. The top navigation bar includes links for ALM, Solman, SIDF, SAMI LAND, Kaar Training, SAMI HQ, SAMI-FSD Tracker..., SAP BTP Trial: Sig..., UI Data Protection..., and All Bookmarks. The main title is "Training" with a dropdown arrow. Below it, a sub-section titled "Training Approval Task" is displayed.

All Tasks (1)

Approval Learning Task
1000000014
SAP_WFRT Medium

Training Approval Task
1000000014

Learning Items

Learning Id: 1000000014 **Course:** ABAP (ABA)

Description: Learning ABAP **Learning Status:** ⚠ In progress

Trainer Name: Dinesh Kumar (K1537)

Items (2) Standard ▾

Item Id	Description	Hours	
10	Item 1	2 >	
20	Item 2	4 >	

Action Buttons: Delete, Standard ▾, More ▾

Bottom Navigation: ⏪ ⏩ ⏴ ⏵, Approve, Reject, Show Log, Show Details, Claim, Forward, Suspend, Open Task, More ▾

Log table is updated and Header table is Update on completion:

The screenshot shows a SAP RAP interface with a search bar for 'ZLE_TRAIN_WF_TRA' and a message 'Training Workflow for approval'. Below it, there are fields for 'Number of Hits' (2) and 'Runtime' (0). A yellow box highlights the 'Maximum No. of Hits' field containing '500'. Below this, an 'Insert Column' toolbar is shown above a table with two rows of data.

Length 10 IN...	Creat...	Stat...	Time	Date	ID	Date	Time
1000000014	1	K1537	A	15:31:50	2025.12.06	411228	2025.12.06 15:29:20
1000000014	2	K1537	A	15:34:57	2025.12.06	411229	2025.12.06 15:31:50

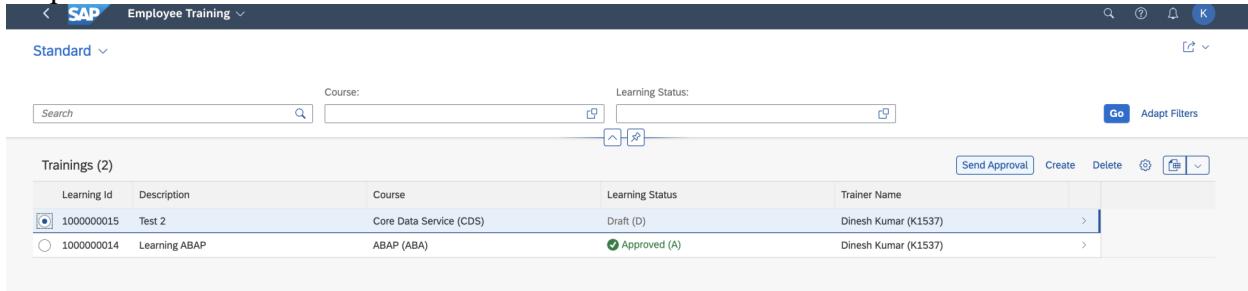
The screenshot shows the SAP Employee Training application. It displays a list of 'Trainings (2)' with columns: Learning Id, Description, Course, Learning Status, and Trainer Name. The first row (Learning Id 1000000015) is marked as 'Draft (D)' and assigned to 'Dinesh Kumar (K1537)'. The second row (Learning Id 1000000014) is marked as 'Approved (A)' and also assigned to 'Dinesh Kumar (K1537)'. Buttons for 'Send Approval', 'Create', 'Delete', and a refresh icon are visible.

Email is sent for Initiator:

The screenshot shows an email message titled 'Display Document: Training Request (1000000014) approved'. The message body contains the text: 'Training Request (1000000014) approved' and 'Dear Dinesh Kumar, Training Request 1000000014 is Approved.' The message was created by 'SAP_WFR' and timestamped 'on 2025.12.06 15:34:57'.

9.2. Reject Scenario

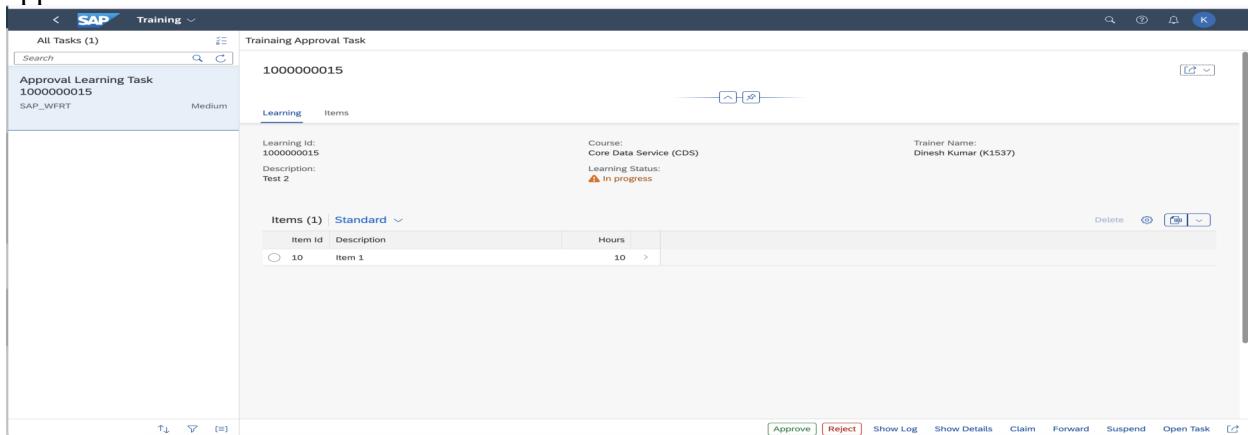
Let us send the other request and reject it in first level itself
Request is sent



The screenshot shows the SAP Employee Training application. In the 'Trainings' section, there are two entries:

- Learning Id: 1000000015, Description: Test 2, Course: Core Data Service (CDS), Learning Status: Draft (D), Trainer Name: Dinesh Kumar (K1537)
- Learning Id: 1000000014, Description: Learning ABAP, Course: ABAP (ABA), Learning Status: Approved (A), Trainer Name: Dinesh Kumar (K1537)

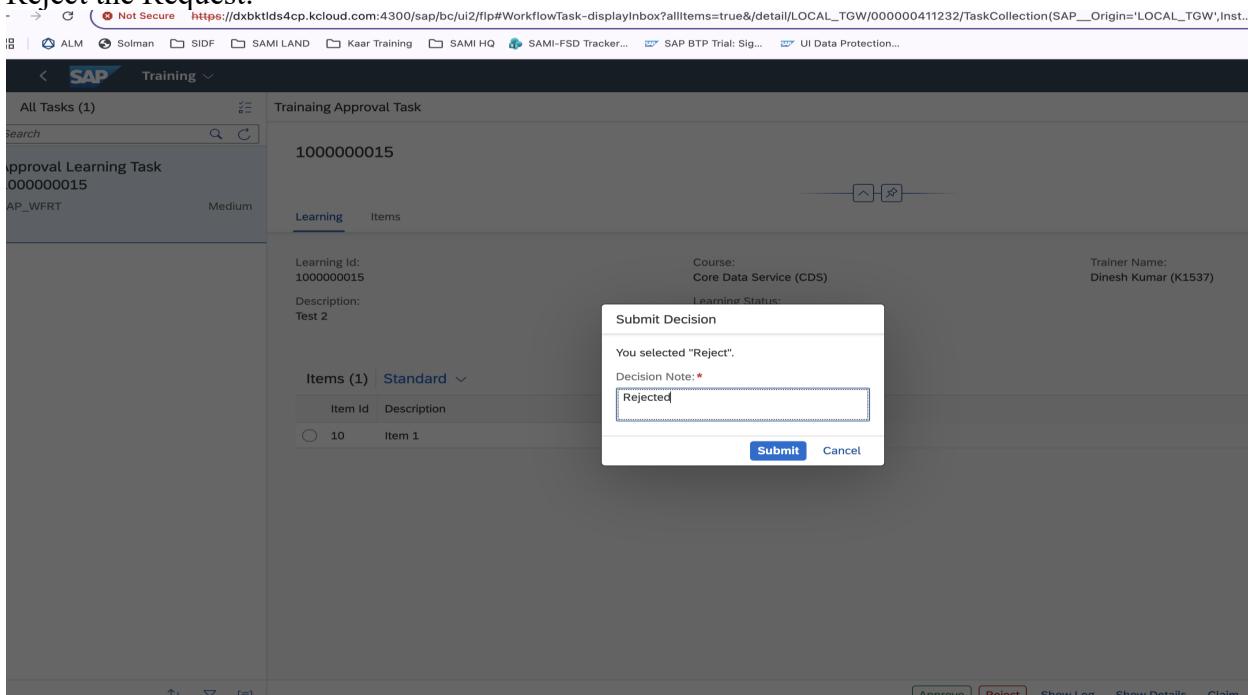
Same Workflow log Tables will be updated and also the Emails are sent. Now go to inbox of approver 1



The screenshot shows the SAP Training application. An approval task for learning item 1000000015 is displayed. The item details are:

- Learning Id: 1000000015
- Description: Test 2
- Course: Core Data Service (CDS)
- Learning Status: In progress
- Trainer Name: Dinesh Kumar (K1537)

Reject the Request:



The screenshot shows the SAP Training application. A modal dialog titled 'Submit Decision' is open, showing the message: "You selected 'Reject'. Decision Note: * Rejected". The 'Submit' button is visible at the bottom.

Now Workflow will be completed in level one itself and level 2 will be ignored

ZLE_TRAIN_WF_TRA: Display of Entries Found									
Search in Table		ZLE_TRAIN_WF_TRA		Training Workflow for approval					
Number of Hits		3		Runtime		0		Maximum No. of Hits	
Insert Column									
Length 10 IN...	Creat...	Stat...	Time	Date	ID	Date	Time		
1000000014	1	K1537	A	15:31:50	2025.12.06	411228	2025.12.06	15:29:20	
1000000014	2	K1537	A	15:34:57	2025.12.06	411229	2025.12.06	15:31:50	
1000000015	1	K1537	R	15:39:21	2025.12.06	411232	2025.12.06	15:37:40	

Email is sent to initiator on rejection:

All Send Requests							
Status	Send Method	Doc. Title	Sender	Recipient	Send Date	Send Time	Msg
Waiting / E-Mail	By E-Mail	Training Request (1000000015) Rejected	K1537	TESTEMAIL@GMAIL.COM	2025.12.06	14:09:21	672
Waiting / E-Mail	By E-Mail	Please Approve Training Request(1000000015)	K1537	TESTEMAIL@GMAIL.COM	2025.12.06	14:07:40	672

Status is updated on table:

Trainings (2)					Send Approval	Create	Delete	Go	Adapt Filters
Learning Id	Description	Course	Learning Status	Trainer Name					
1000000015	Test 2	Core Data Service (CDS)	Rejected (R)	Dinesh Kumar (K1537)					
1000000014	Learning ABAP	ABAP (ABA)	Approved (A)	Dinesh Kumar (K1537)					