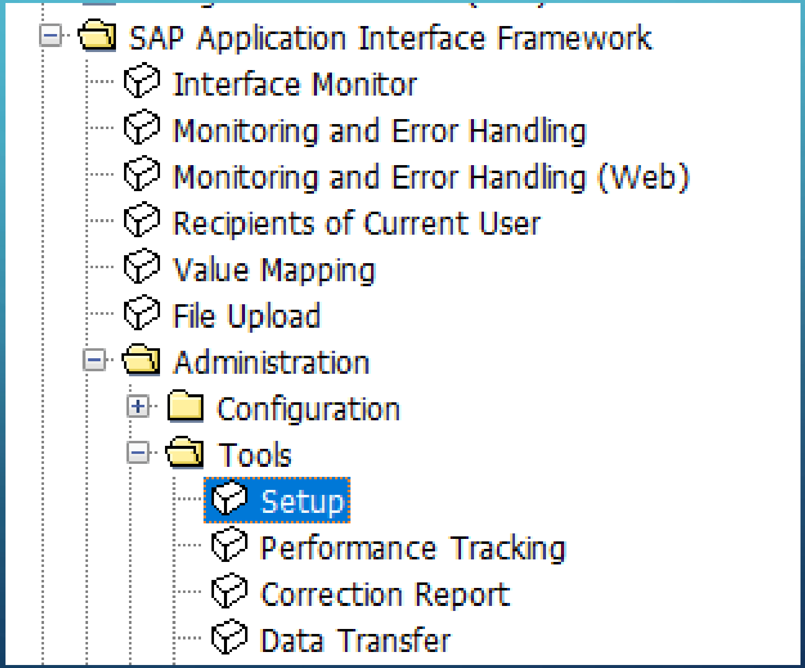
AIF (APPLICATION PROGRAMMING INTERFACE)

PART 1

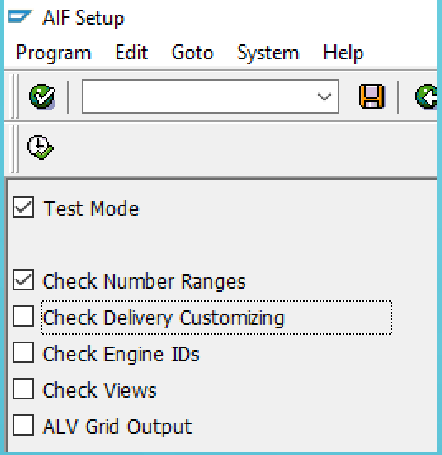
MY TRAINING PROJECTS BY AIF TUTORIALS FROM SAPCODES.COM(HTTPS://SAPCODES.COM/AIF-APPLICATION-PROGRAMMING-INTERFACE/)

1. AIF Set Up

SAP AIF PROVIDES TRANSACTION TX- /AIF/SETUP TO CHECK AND CREATE REQUIRED NUMBER RANGES AND OTHER THINGS. BEFORE DOING ANY PROCESSING IN AIF, THIS MUST BE CHECKED AND CREATED FIRST. FROM THE ARE MENU /AIF/ , SELECT TX- /AIF/SETUP



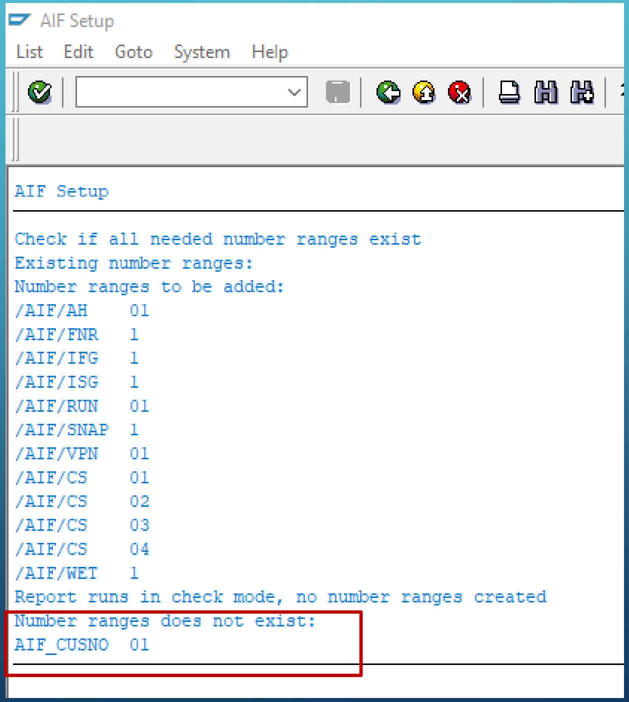
THE PROGRAM CAN BE RUN IN TEST MODE/REAL MODE TO CHECK NUMBER OF THINGS. IN THIS POST WILL CHECK THE AIF NUMBER RANGES. SELECT THE TEST MODE AND SELECT CHECK NUMBER RANGES AND EXECUTE.



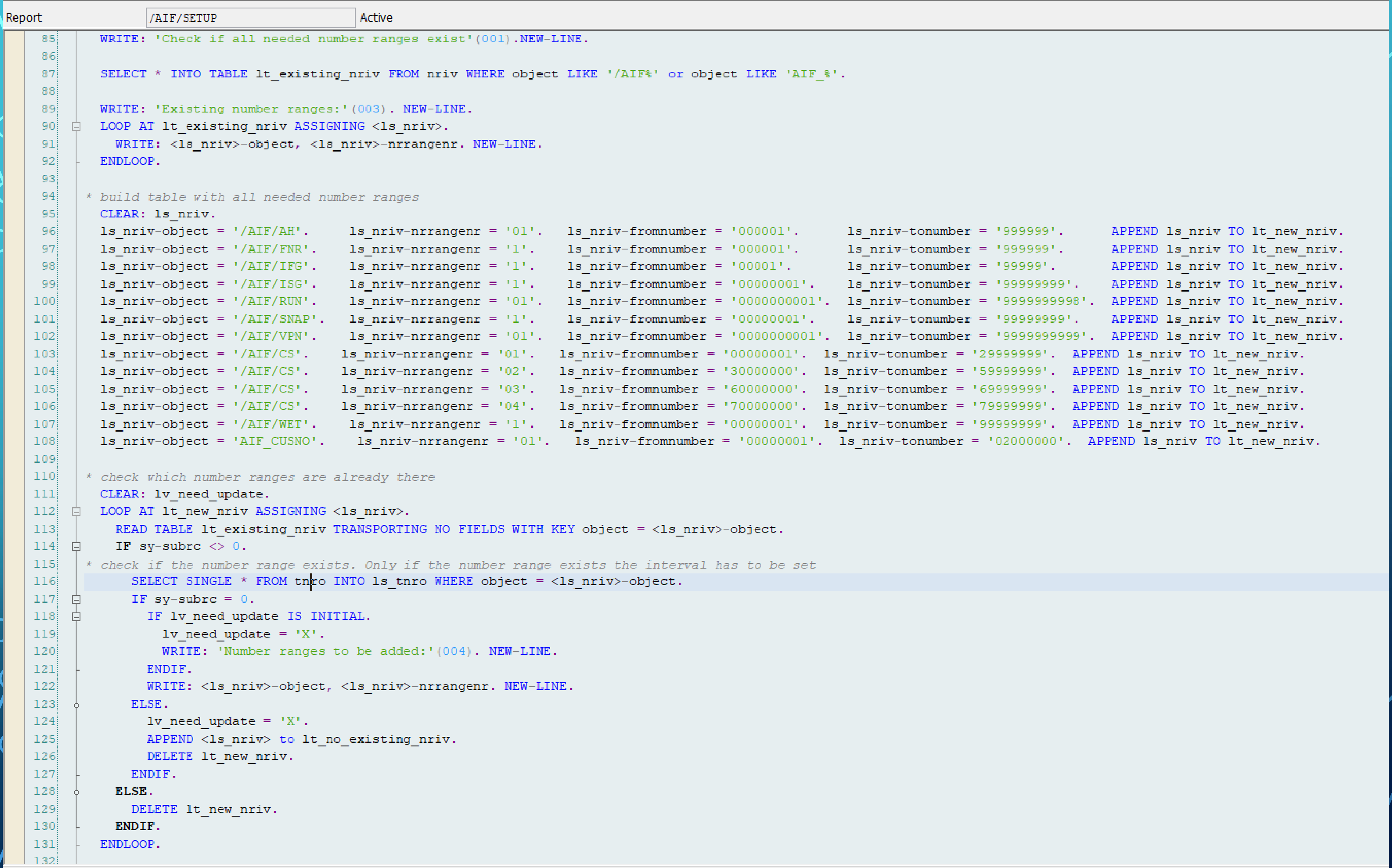
☑ Test Mode

☑ Check Number Ranges
☐ Check Delivery Customizing
☐ Check Engine IDs
☐ Check Views
☐ ALV Grid Output

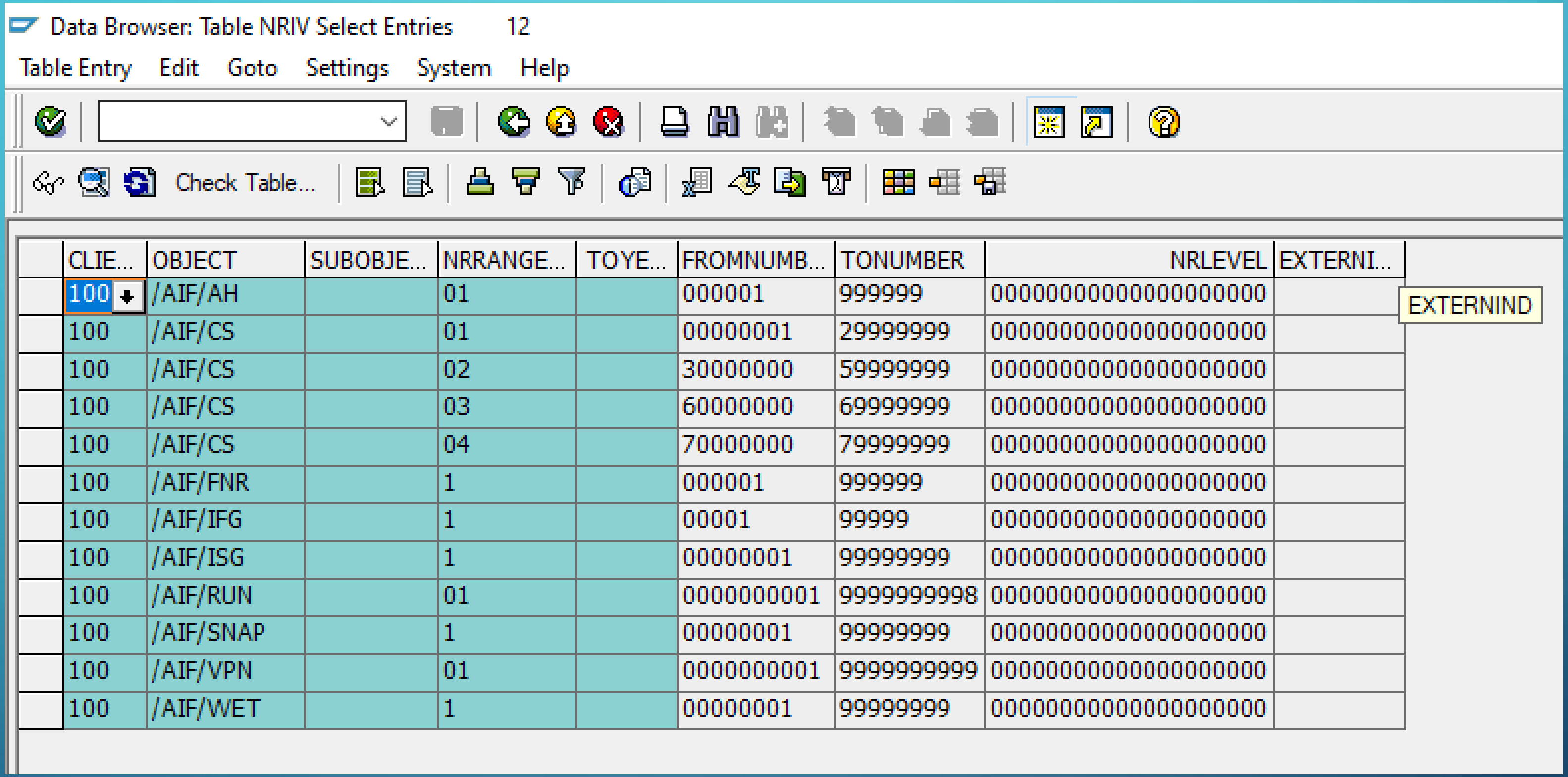
The program result shows all the number ranges already created in the system and if not created for few, then this program can be run (without test mode) to create the number ranges in the system.



```
AIF Setup

Check if all needed number ranges exist
Existing number ranges:
Number ranges to be added:
/AIF/AH    01
/AIF/FNR   1
/AIF/IFG   1
/AIF/ISG   1
/AIF/RUN   01
/AIF/SNAP  1
/AIF/VPN   01
/AIF/CS    01
/AIF/CS    02
/AIF/CS    03
/AIF/CS    04
/AIF/WET   1
Report runs in check mode, no number ranges created
Number ranges does not exist:
AIF_CUSNO  01
```

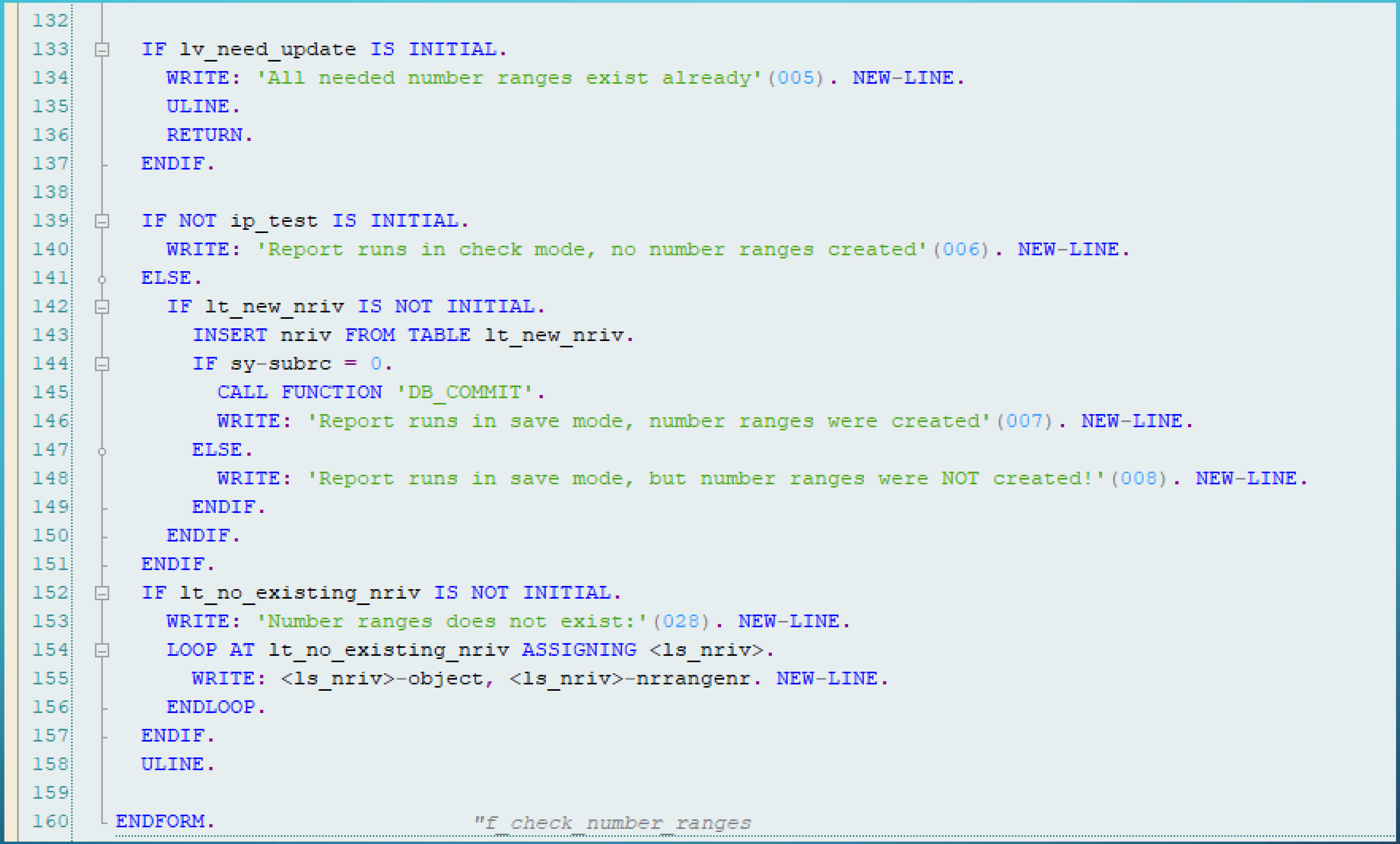


```
  WRITE: 'Check if all needed number ranges exist'(001).NEW-LINE.

  SELECT * INTO TABLE lt_existing_nriv FROM nriv WHERE object LIKE '/AIF%' or object LIKE 'AIF_%'.

  WRITE: 'Existing number ranges:'(003). NEW-LINE.
  LOOP AT lt_existing_nriv ASSIGNING <ls_nriv>.
    WRITE: <ls_nriv>-object, <ls_nriv>-nrrangenr. NEW-LINE.
  ENDLOOP.

* build table with all needed number ranges
  CLEAR: ls_nriv.
  ls_nriv-object = '/AIF/AH'.     ls_nriv-nrrangenr = '01'.   ls_nriv-fromnumber = '000001'.      ls_nriv-tonumber = '999999'.      APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/FNR'.    ls_nriv-nrrangenr = '1'.    ls_nriv-fromnumber = '000001'.      ls_nriv-tonumber = '999999'.      APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/IFG'.    ls_nriv-nrrangenr = '1'.    ls_nriv-fromnumber = '00001'.       ls_nriv-tonumber = '99999'.       APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/ISG'.    ls_nriv-nrrangenr = '1'.    ls_nriv-fromnumber = '00000001'.    ls_nriv-tonumber = '99999999'.    APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/RUN'.    ls_nriv-nrrangenr = '01'.   ls_nriv-fromnumber = '0000000001'.  ls_nriv-tonumber = '9999999998'.  APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/SNAP'.   ls_nriv-nrrangenr = '1'.    ls_nriv-fromnumber = '00000001'.    ls_nriv-tonumber = '99999999'.    APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/VPN'.    ls_nriv-nrrangenr = '01'.   ls_nriv-fromnumber = '0000000001'.  ls_nriv-tonumber = '9999999999'.  APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/CS'.    ls_nriv-nrrangenr = '01'.   ls_nriv-fromnumber = '00000001'.  ls_nriv-tonumber = '29999999'.  APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/CS'.    ls_nriv-nrrangenr = '02'.   ls_nriv-fromnumber = '30000000'.  ls_nriv-tonumber = '59999999'.  APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/CS'.    ls_nriv-nrrangenr = '03'.   ls_nriv-fromnumber = '60000000'.  ls_nriv-tonumber = '69999999'.  APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/CS'.    ls_nriv-nrrangenr = '04'.   ls_nriv-fromnumber = '70000000'.  ls_nriv-tonumber = '79999999'.  APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = '/AIF/WET'.    ls_nriv-nrrangenr = '1'.   ls_nriv-fromnumber = '00000001'.  ls_nriv-tonumber = '99999999'.  APPEND ls_nriv TO lt_new_nriv.
  ls_nriv-object = 'AIF_CUSNO'.    ls_nriv-nrrangenr = '01'.   ls_nriv-fromnumber = '00000001'.  ls_nriv-tonumber = '02000000'.  APPEND ls_nriv TO lt_new_nriv.

* check which number ranges are already there
  CLEAR: lv_need_update.
  LOOP AT lt_new_nriv ASSIGNING <ls_nriv>.
    READ TABLE lt_existing_nriv TRANSPORTING NO FIELDS WITH KEY object = <ls_nriv>-object.
    IF sy-subrc <> 0.
* check if the number range exists. Only if the number range exists the interval has to be set
      SELECT SINGLE * FROM tnro INTO ls_tnro WHERE object = <ls_nriv>-object.
      IF sy-subrc = 0.
        IF lv_need_update IS INITIAL.
          lv_need_update = 'X'.
          WRITE: 'Number ranges to be added:'(004). NEW-LINE.
        ENDIF.
        WRITE: <ls_nriv>-object, <ls_nriv>-nrrangenr. NEW-LINE.
      ELSE.
        lv_need_update = 'X'.
        APPEND <ls_nriv> to lt_no_existing_nriv.
        DELETE lt_new_nriv.
      ENDIF.
    ELSE.
      DELETE lt_new_nriv.
    ENDIF.
  ENDLOOP.
```



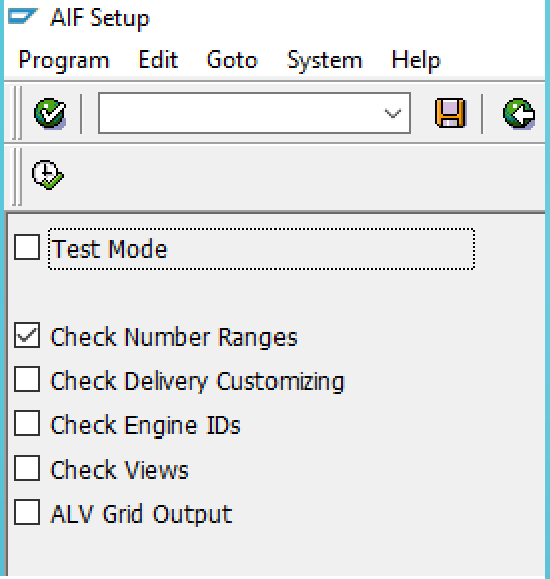
Data Browser: Table NRIV Select Entries 12

CLIE...	OBJECT	SUBOBJE...	NRRANGE...	TOYE...	FROMNUMB...	TONUMBER	NRLEVEL	EXTERNI...
100	/AIF/AH		01		000001	999999	00000000000000000000	
100	/AIF/CS		01		00000001	29999999	00000000000000000000	
100	/AIF/CS		02		30000000	59999999	00000000000000000000	
100	/AIF/CS		03		60000000	69999999	00000000000000000000	
100	/AIF/CS		04		70000000	79999999	00000000000000000000	
100	/AIF/FNR		1		000001	999999	00000000000000000000	
100	/AIF/IFG		1		00001	99999	00000000000000000000	
100	/AIF/ISG		1		00000001	99999999	00000000000000000000	
100	/AIF/RUN		01		0000000001	9999999998	00000000000000000000	
100	/AIF/SNAP		1		00000001	99999999	00000000000000000000	
100	/AIF/VPN		01		0000000001	9999999999	00000000000000000000	
100	/AIF/WET		1		00000001	99999999	00000000000000000000	



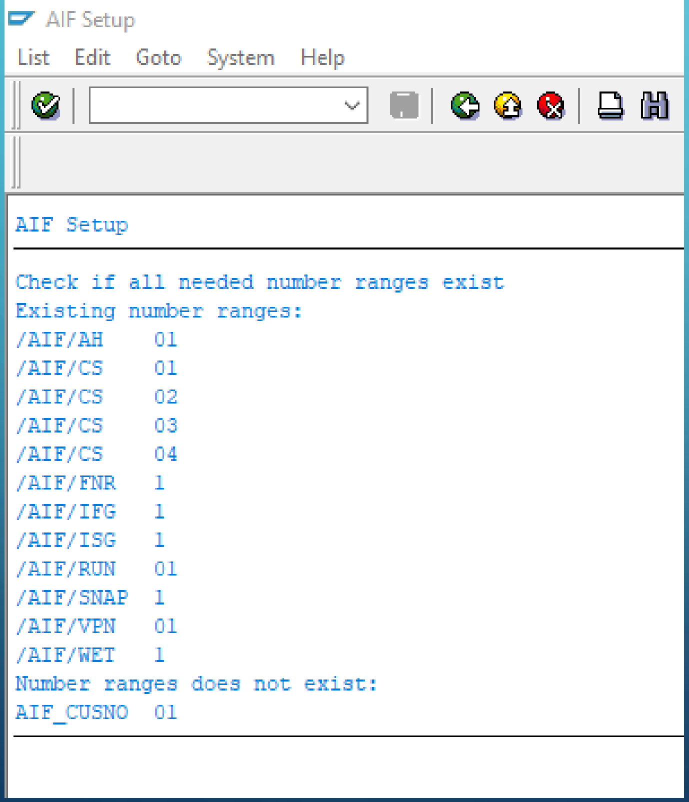
```

  IF lv_need_update IS INITIAL.
    WRITE: 'All needed number ranges exist already'(005). NEW-LINE.
    ULINE.
    RETURN.
  ENDIF.

  IF NOT ip_test IS INITIAL.
    WRITE: 'Report runs in check mode, no number ranges created'(006). NEW-LINE.
  ELSE.
    IF lt_new_nriv IS NOT INITIAL.
      INSERT nriv FROM TABLE lt_new_nriv.
      IF sy-subrc = 0.
        CALL FUNCTION 'DB_COMMIT'.
        WRITE: 'Report runs in save mode, number ranges were created'(007). NEW-LINE.
      ELSE.
        WRITE: 'Report runs in save mode, but number ranges were NOT created!'(008). NEW-LINE.
      ENDIF.
    ENDIF.
  ENDIF.
  IF lt_no_existing_nriv IS NOT INITIAL.
    WRITE: 'Number ranges does not exist:'(028). NEW-LINE.
    LOOP AT lt_no_existing_nriv ASSIGNING <ls_nriv>.
      WRITE: <ls_nriv>-object, <ls_nriv>-nrrangenr. NEW-LINE.
    ENDLOOP.
  ENDIF.
  ULINE.

ENDFORM.                    "f_check_number_ranges
```

The report details: it really creates number ranges which not yet created in the system except of ranges that don’t exist in the table TNRO.



□ Test Mode

☑ Check Number Ranges
□ Check Delivery Customizing
□ Check Engine IDs
□ Check Views
□ ALV Grid Output



```
AIF Setup

Check if all needed number ranges exist
Existing number ranges:
/AIF/AH    01
/AIF/CS    01
/AIF/CS    02
/AIF/CS    03
/AIF/CS    04
/AIF/FNR   1
/AIF/IFG   1
/AIF/ISG   1
/AIF/RUN   01
/AIF/SNAP  1
/AIF/VPN   01
/AIF/WET   1
Number ranges does not exist:
AIF_CUSNO  01
```

2. AIF INTRO & PROCESSING FIRST MESSAGE IN AIF

THE SAP APPLICATION INTERFACE FRAMEWORK (AIF) ENABLES TO DEVELOP AND MONITOR INTERFACES AS WELL AS EXECUTE ERROR HANDLING IN A SINGLE FRAMEWORK .

IT IS MOSTLY USEFUL IN A COMPLEX HETEROGENEOUS SYSTEM LANDSCAPE WITH SAP PI SYSTEM. BUSINESS USER(NOT TECHNICAL USER) CAN PERFORM ERROR MONITORING AND THE ERROR HANDLING.

AIF IS MOSTLY USEFUL WHEN DATA TRANSFER HAPPENS BETWEEN DIFFERENT SAP SYSTEMS.

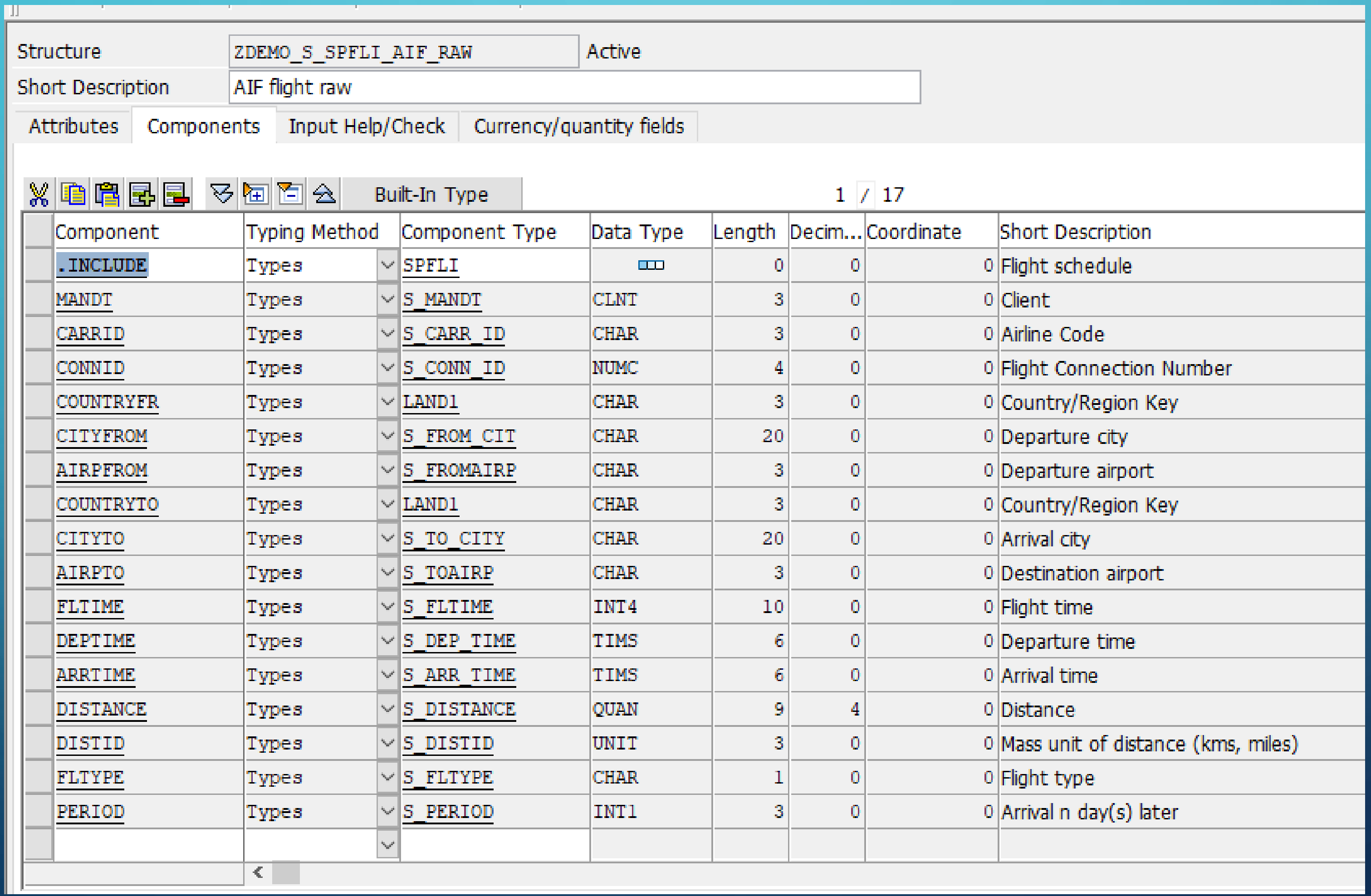
FOR THE BELOW POST USE CASE- CONSIDER WE HAVE TWO SYSTEMS

1. SOURCE SYSTEM WHICH SENDS THE DATA
2. TARGET SYSTEM WHICH RECEIVES THE DATA VIA AIF AND PROCESS IT THEN

SO AS RECEIVED MESSAGE TO BE PROCESSED IN TARGET SYSTEM VIA AIF, THEN CERTAIN AIF CUSTOMIZING NEEDED.

BELOW STEP EXPLAINS AIF CUSTOMIZING STEPS NEEDED IN TARGET SYSTEM.

CREATE DDIC STRUCTURE (THIS IS THE STRUCTURAL FORMAT TARGET SYSTEM RECEIVES THE DATA)

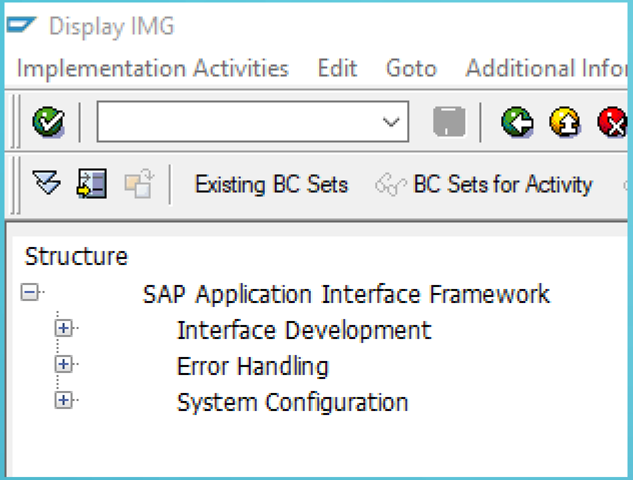


Structure: ZDEMO_S_SPFLI_AIF_RAW Active

Short Description: AIF flight raw

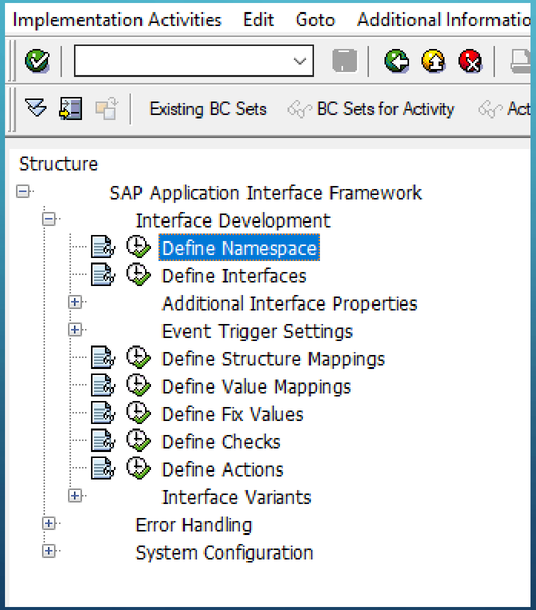
Component	Typing Method	Component Type	Data Type	Length	Decim...	Coordinate	Short Description
.INCLUDE	Types	SPFLI	▭	0	0	0	Flight schedule
MANDT	Types	S_MANDT	CLNT	3	0	0	Client
CARRID	Types	S_CARR_ID	CHAR	3	0	0	Airline Code
CONNID	Types	S_CONN_ID	NUMC	4	0	0	Flight Connection Number
COUNTRYFR	Types	LAND1	CHAR	3	0	0	Country/Region Key
CITYFROM	Types	S_FROM_CIT	CHAR	20	0	0	Departure city
AIRPFROM	Types	S_FROMAIRP	CHAR	3	0	0	Departure airport
COUNTRYTO	Types	LAND1	CHAR	3	0	0	Country/Region Key
CITYTO	Types	S_TO_CITY	CHAR	20	0	0	Arrival city
AIRPTO	Types	S_TOAIRP	CHAR	3	0	0	Destination airport
FLTIME	Types	S_FLTIME	INT4	10	0	0	Flight time
DEPTIME	Types	S_DEP_TIME	TIMS	6	0	0	Departure time
ARRTIME	Types	S_ARR_TIME	TIMS	6	0	0	Arrival time
DISTANCE	Types	S_DISTANCE	QUAN	9	4	0	Distance
DISTID	Types	S_DISTID	UNIT	3	0	0	Mass unit of distance (kms, miles)
FLTYPE	Types	S_FLTYPE	CHAR	1	0	0	Flight type
PERIOD	Types	S_PERIOD	INT1	3	0	0	Arrival n day(s) later

EXECUTE TX- /AIF/CUST

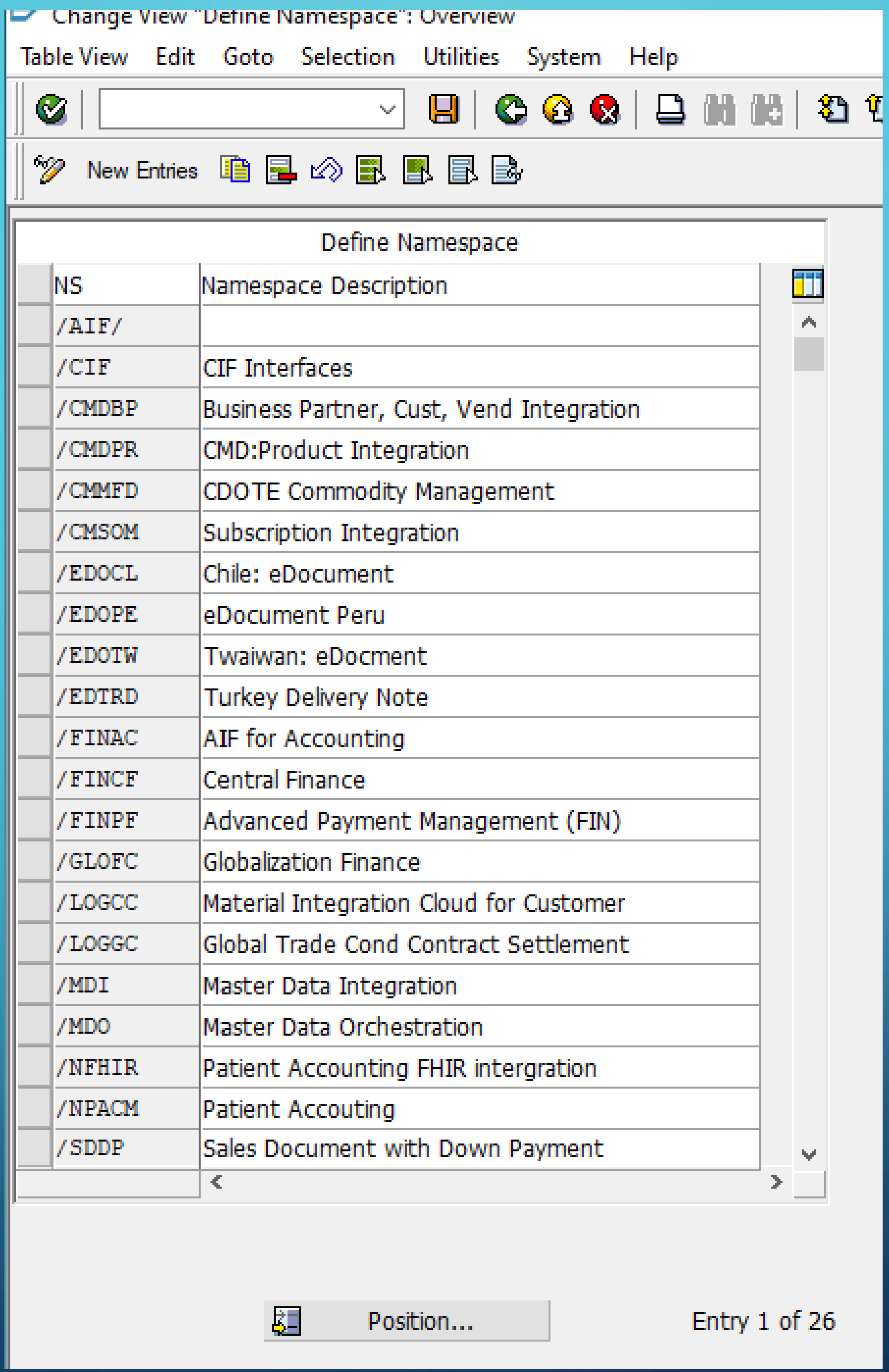


Structure
- SAP Application Interface Framework
  - Interface Development
  - Error Handling
  - System Configuration

Expand the IMG tree and choose option- Define Namespace.



Structure
- SAP Application Interface Framework
  - Interface Development
    - Define Namespace
    - Define Interfaces
    - Additional Interface Properties
    - Event Trigger Settings
    - Define Structure Mappings
    - Define Value Mappings
    - Define Fix Values
    - Define Checks
    - Define Actions
    - Interface Variants
  - Error Handling
  - System Configuration



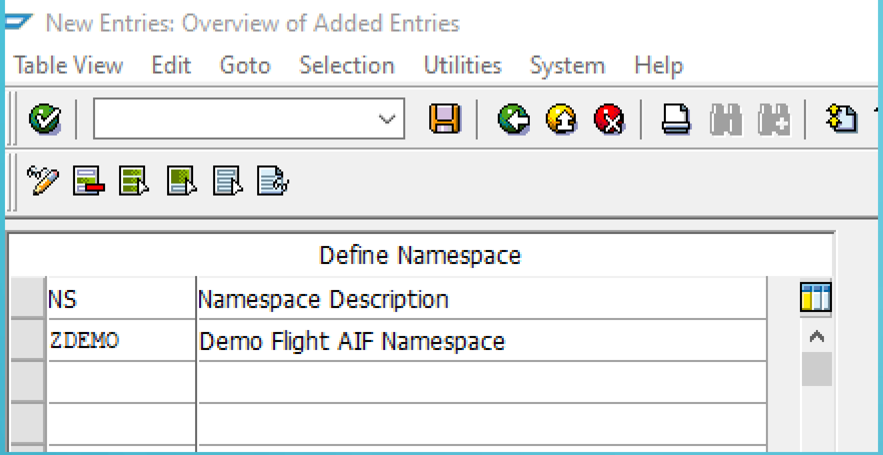
Change View "Define Namespace": Overview

Define Namespace

NS	Namespace Description
/AIF/	
/CIF	CIF Interfaces
/CMDBP	Business Partner, Cust, Vend Integration
/CMDPR	CMD:Product Integration
/CMMFD	CDOTE Commodity Management
/CMSOM	Subscription Integration
/EDOCL	Chile: eDocument
/EDOPE	eDocument Peru
/EDOTW	Twaiwan: eDocment
/EDTRD	Turkey Delivery Note
/FINAC	AIF for Accounting
/FINCF	Central Finance
/FINPF	Advanced Payment Management (FIN)
/GLOFC	Globalization Finance
/LOGCC	Material Integration Cloud for Customer
/LOGGC	Global Trade Cond Contract Settlement
/MDI	Master Data Integration
/MDO	Master Data Orchestration
/NFHIR	Patient Accounting FHIR intergration
/NPACM	Patient Accouting
/SDDP	Sales Document with Down Payment

Entry 1 of 26

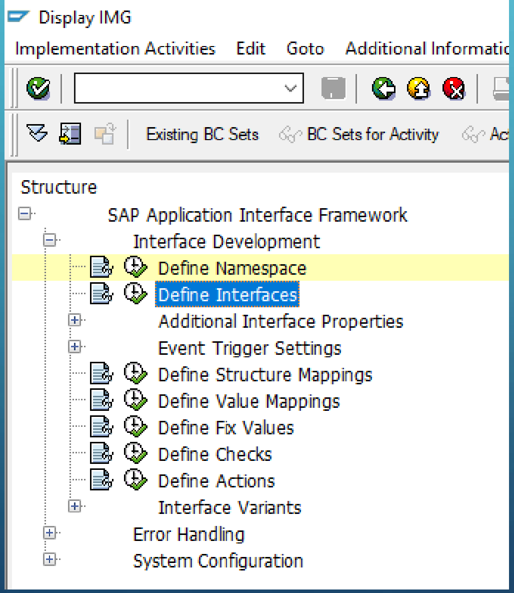
Create a namespace ‘ZDEMO’ and save it.

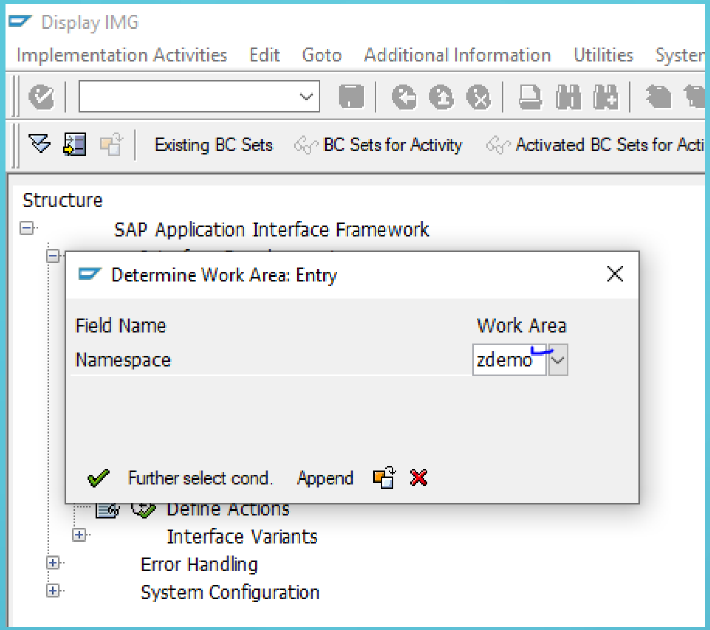


Define Namespace

NS	Namespace Description
ZDEMO	Demo Flight AIF Namespace

Next step is to create Interface. Choose define Interface option.





Determine Work Area: Entry

Field Name	Work Area
Namespace	zdemo

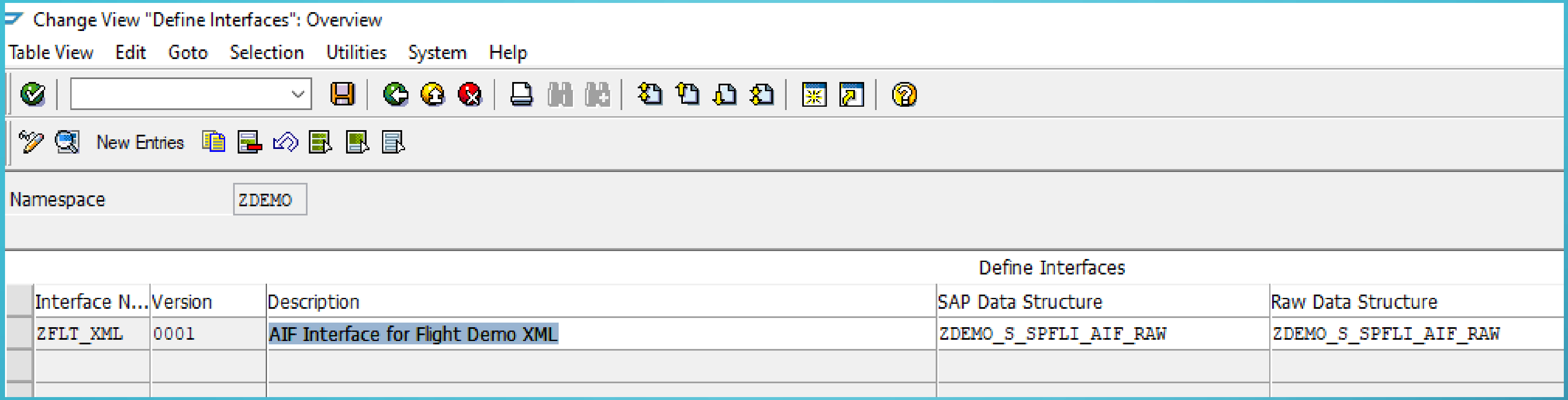
Choose New Entries button.

Provide interface name as- ZFLT_XML(in AIF message can be processed in different ways like XML, Proxy etc , the below demo processes the AIF message as XML, so accordingly the interface name is adopted) and interface version as 00001.

provide the structure name we have created in the very first step as SAP Data Structure and RAW Data Structure and select the Move Corresponding structure.

For this demo, the RAW and SAP structure are same but in normal business case these two structures are different. RAW structure is what is received in the target system and then it is converted to SAP structure as per different structure mapping rules in AIF which is one of the strength of AIF. Save and go back.s

For the namespace one interface is defined.

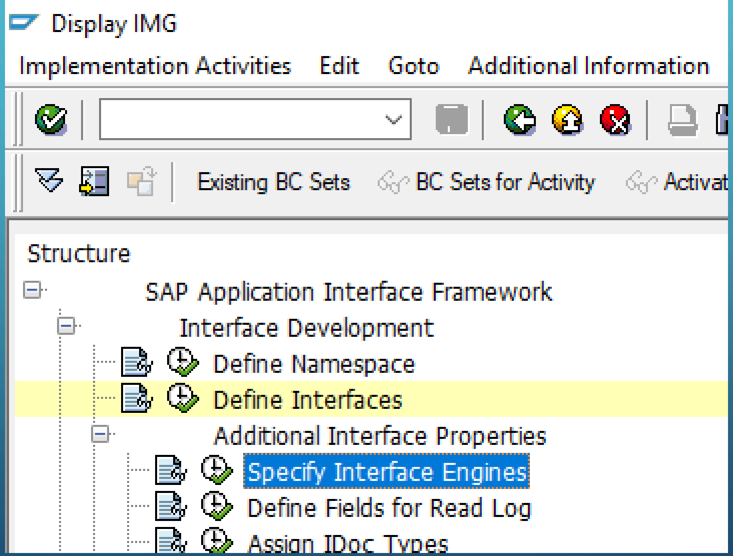

Change View "Define Interfaces": Overview
Table View Edit Goto Selection Utilities System Help
New Entries

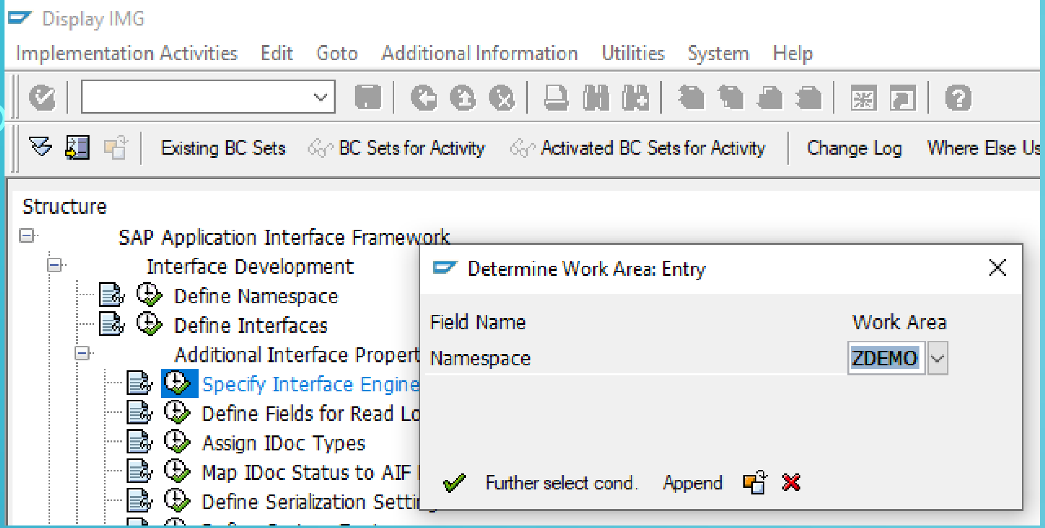
Namespace ZDEMO

Define Interfaces

Interface N...	Version	Description	SAP Data Structure	Raw Data Structure
ZFLT_XML	0001	AIF Interface for Flight Demo XML	ZDEMO_S_SPFLI_AIF_RAW	ZDEMO_S_SPFLI_AIF_RAW

Now we have to specify which AIF processing technique to be used like XML or proxy so on. Choose option- Specify Interface Engines.


Display IMG
Implementation Activities Edit Goto Additional Information
Existing BC Sets
BC Sets for Activity
Activat
Structure
SAP Application Interface Framework
Interface Development
Define Namespace
Define Interfaces
Additional Interface Properties
Specify Interface Engines
Define Fields for Read Log
Assign IDoc Types


Display IMG
Implementation Activities Edit Goto Additional Information Utilities System Help
Existing BC Sets
BC Sets for Activity
Activated BC Sets for Activity
Change Log
Where Else Us

Structure

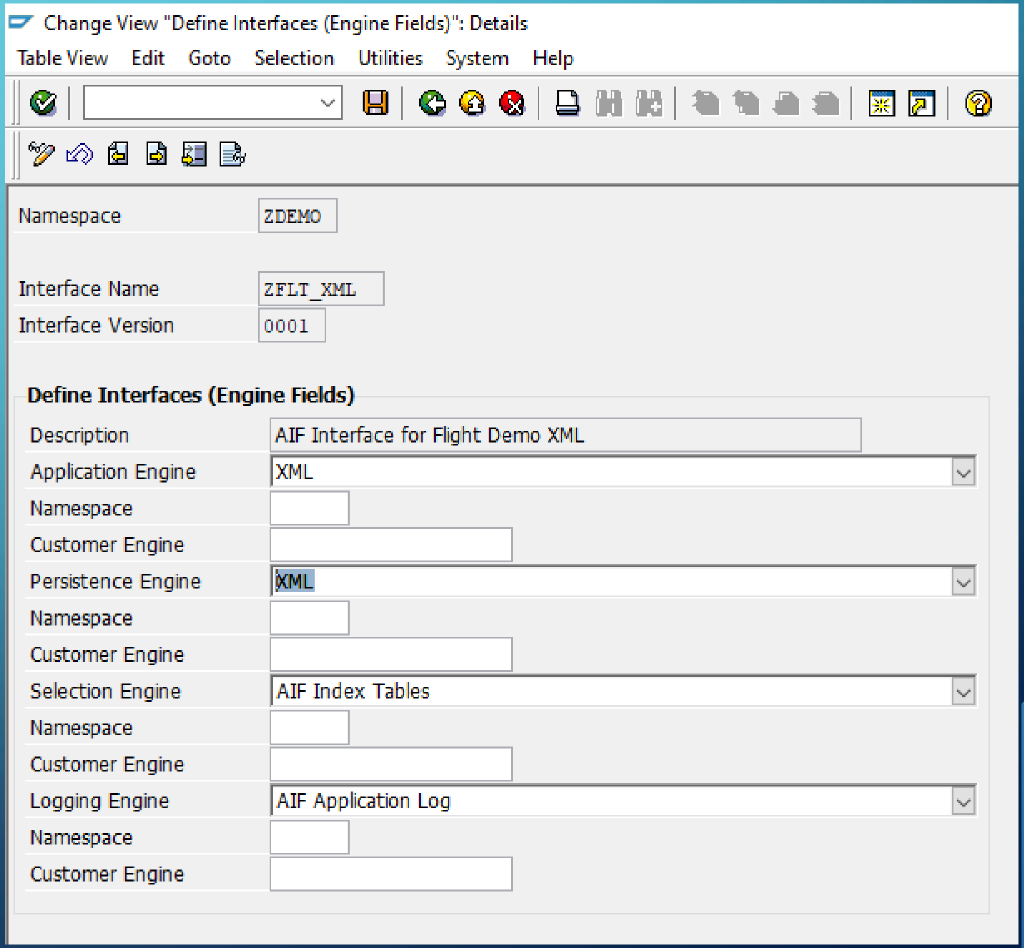
- SAP Application Interface Framework
  - Interface Development
    - Define Namespace
    - Define Interfaces
    - Additional Interface Propert
      - Specify Interface Engine
      - Define Fields for Read Lo
      - Assign IDoc Types
      - Map IDoc Status to AIF
      - Define Serialization Setti

Determine Work Area: Entry

Field Name	Work Area
Namespace	ZDEMO

Further select cond.
Append

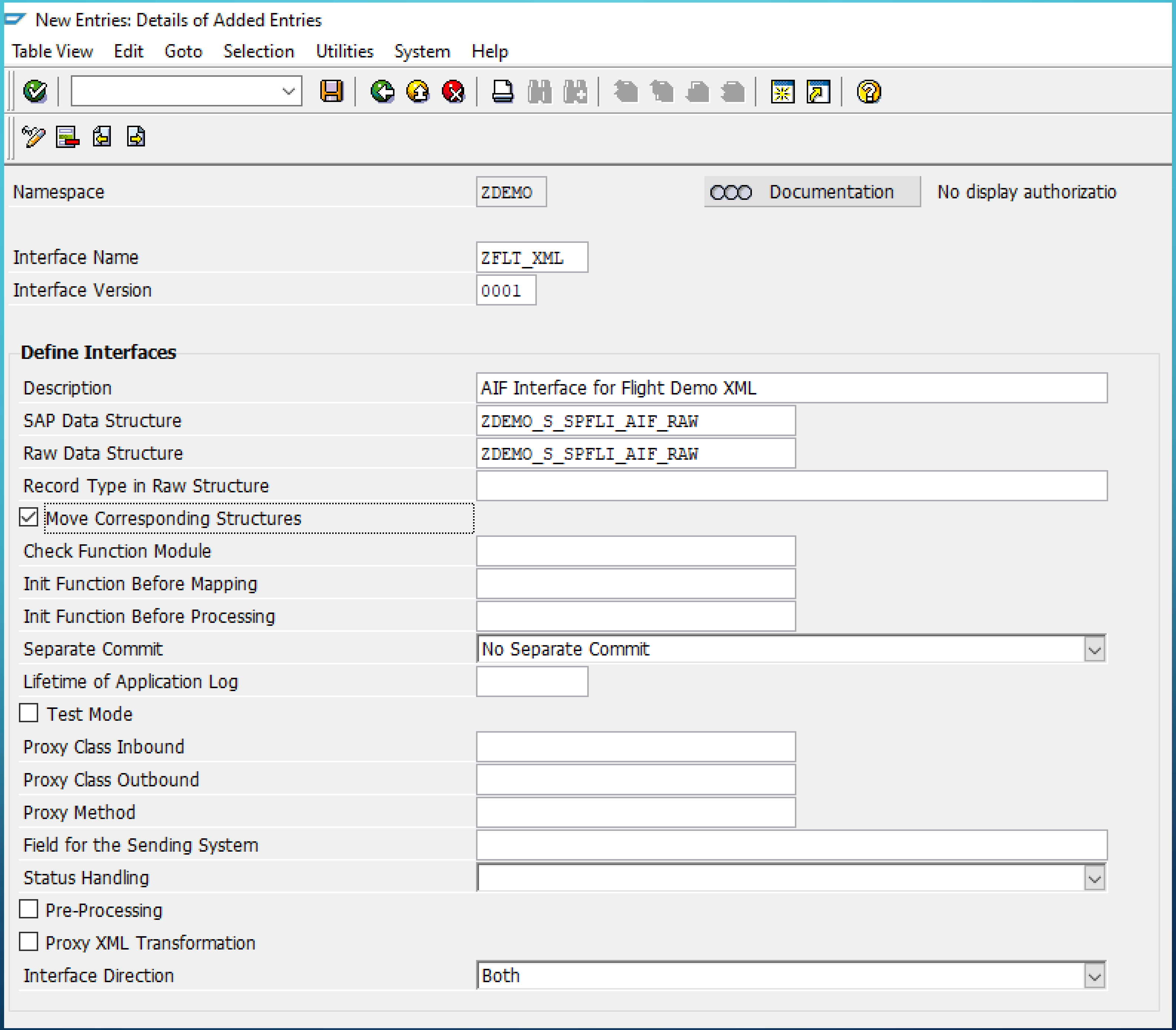
Choose XML for application engine and persistent engine. This means the received messages in target system AIF will be stored as XML message.


Change View "Define Interfaces (Engine Fields)": Details
Table View Edit Goto Selection Utilities System Help

Namespace	ZDEMO
Interface Name	ZFLT_XML
Interface Version	0001

Define Interfaces (Engine Fields)

Description	AIF Interface for Flight Demo XML
Application Engine	XML
Namespace	
Customer Engine	
Persistence Engine	XML
Namespace	
Customer Engine	
Selection Engine	AIF Index Tables
Namespace	
Customer Engine	
Logging Engine	AIF Application Log
Namespace	
Customer Engine	



Namespace	ZDEMO	No display authorizatio

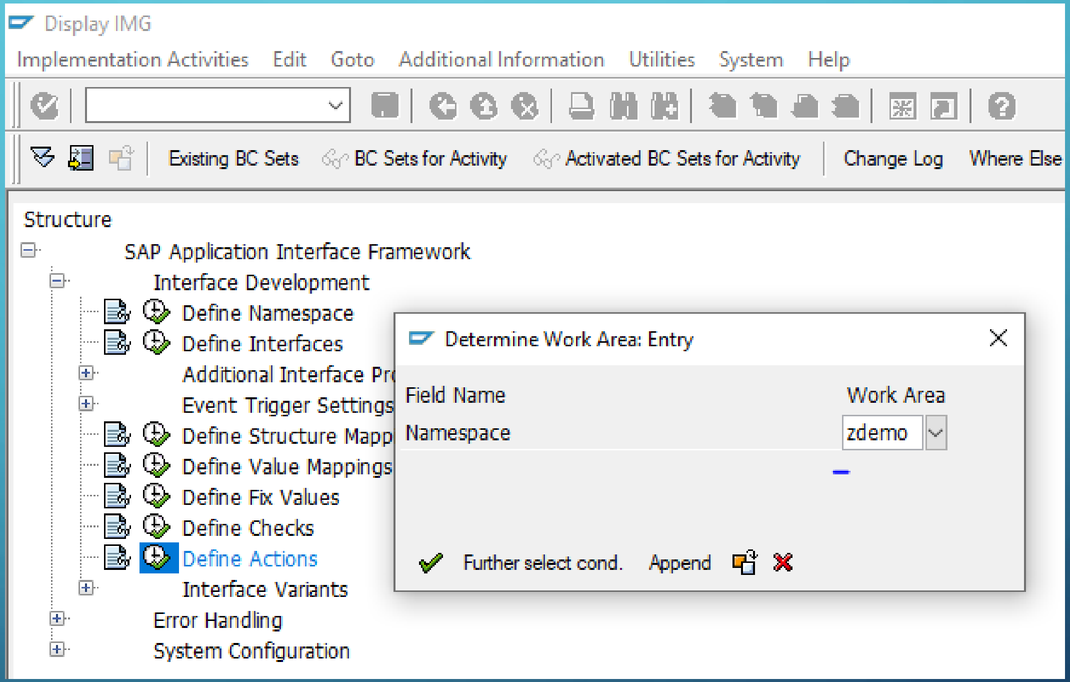
Interface Name	ZFLT_XML
Interface Version	0001

Define Interfaces

Description	AIF Interface for Flight Demo XML
SAP Data Structure	ZDEMO_S_SPFLI_AIF_RAW
Raw Data Structure	ZDEMO_S_SPFLI_AIF_RAW
Record Type in Raw Structure	
☑ Move Corresponding Structures	
Check Function Module	
Init Function Before Mapping	
Init Function Before Processing	
Separate Commit	No Separate Commit
Lifetime of Application Log	
☐ Test Mode	
Proxy Class Inbound	
Proxy Class Outbound	
Proxy Method	
Field for the Sending System	
Status Handling	
☐ Pre-Processing	
☐ Proxy XML Transformation	
Interface Direction	Both

ONCE THE MESSAGE IS RECEIVED BY THE AIF, THEN IT IS FURTHER PROCESSED. THE ACTIONS DEFINE THE PROCESSING STEPS. EACH ACTION – HAS MULTIPLE STEPS AND IN EACH STEP A FUNCTION MODULE CAN BE SPECIFIED.

Choose Define Actions option.


Display IMG
Implementation Activities Edit Goto Additional Information Utilities System Help
Existing BC Sets
BC Sets for Activity
Activated BC Sets for Activity
Change Log
Where Else

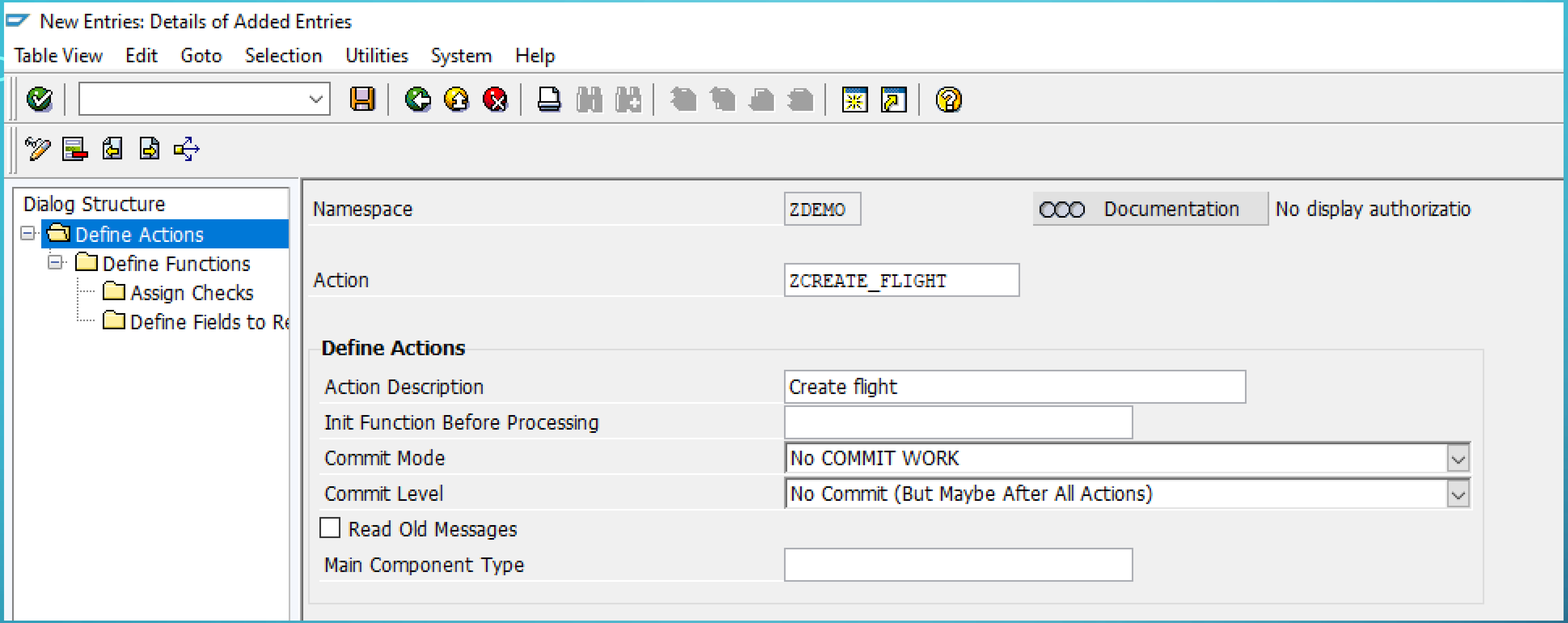
Structure

- SAP Application Interface Framework
  - Interface Development
    - Define Namespace
    - Define Interfaces
    - Additional Interface Pr
    - Event Trigger Settings
    - Define Structure Mapp
    - Define Value Mappings
    - Define Fix Values
    - Define Checks
    - Define Actions
    - Interface Variants
  - Error Handling
  - System Configuration

Determine Work Area: Entry

Field Name	Work Area
Namespace	zdemo

s



Namespace	ZDEMO
Action	ZCREATE_FLIGHT

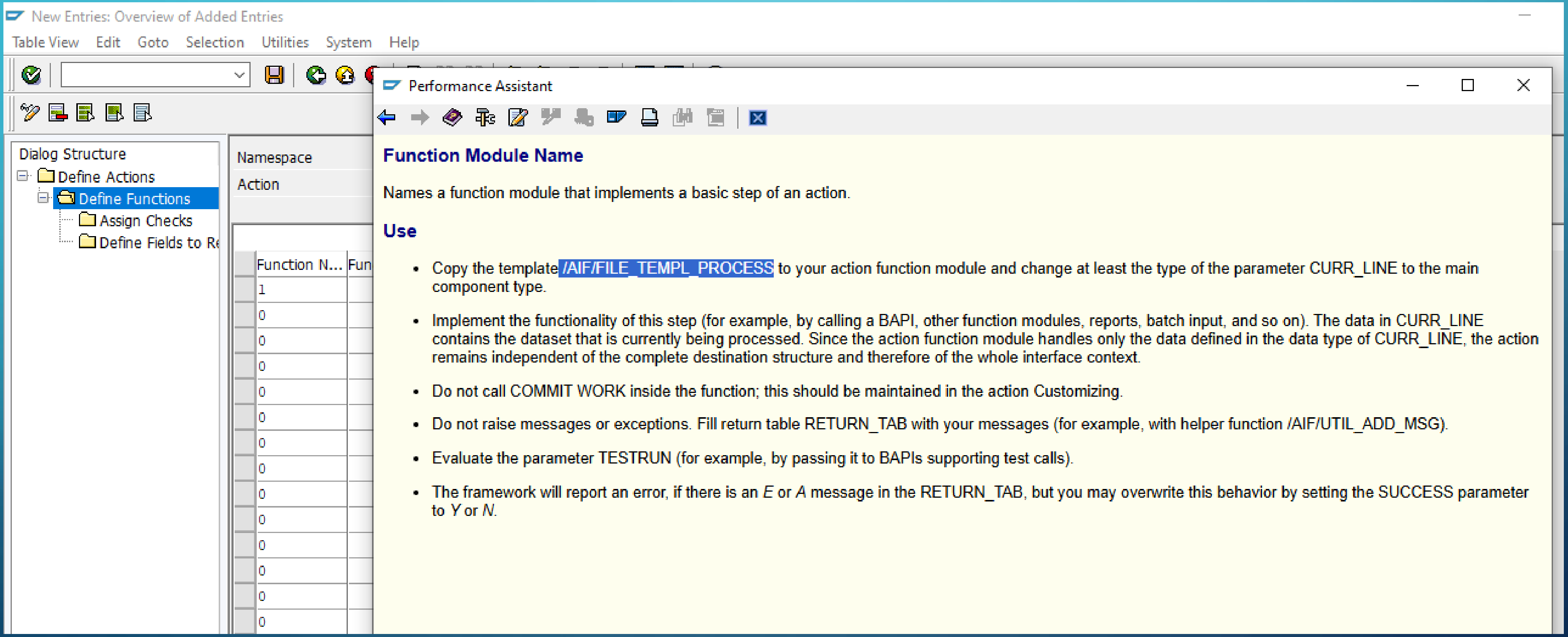
No display authorizatio

Define Actions

Action Description	Create flight
Init Function Before Processing	
Commit Mode	No COMMIT WORK
Commit Level	No Commit (But Maybe After All Actions)
☐ Read Old Messages	
Main Component Type	

Provide action name and action description and then choose Define Functions from left side panel.

Provide Function number as 1 and then press F1 to get the help. Here we have to provide a Function module which will be called by the run time execution of AIF. The FM has a specific format. The F1 help provide a template FM which can be copied and code adopted according to business process.



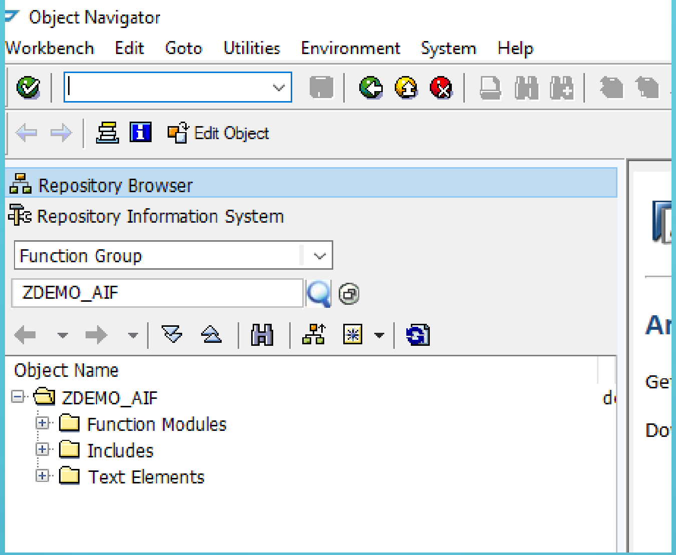
Function Module Name

Names a function module that implements a basic step of an action.

Use

- Copy the template /AIF/FILE_TEMPL_PROCESS to your action function module and change at least the type of the parameter CURR_LINE to the main component type.
- Implement the functionality of this step (for example, by calling a BAPI, other function modules, reports, batch input, and so on). The data in CURR_LINE contains the dataset that is currently being processed. Since the action function module handles only the data defined in the data type of CURR_LINE, the action remains independent of the complete destination structure and therefore of the whole interface context.
- Do not call COMMIT WORK inside the function; this should be maintained in the action Customizing.
- Do not raise messages or exceptions. Fill return table RETURN_TAB with your messages (for example, with helper function /AIF/UTIL_ADD_MSG).
- Evaluate the parameter TESTRUN (for example, by passing it to BAPIs supporting test calls).
- The framework will report an error, if there is an *E* or *A* message in the RETURN_TAB, but you may overwrite this behavior by setting the SUCCESS parameter to *Y* or *N*.

BEFORE CREATING A FM CREATE A FUNCTION GROUP.

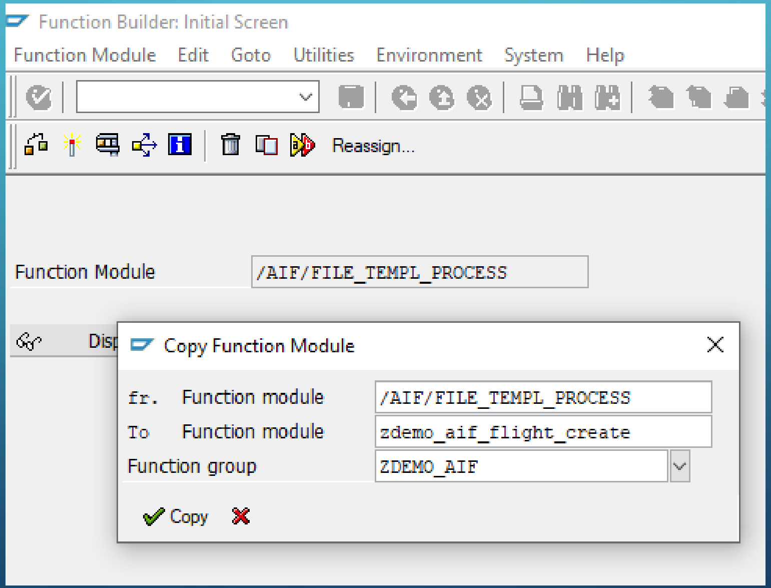


Function Group

ZDEMO_AIF

ZDEMO_AIF

Go to Tx- SE37 and Copy the FM- and create a new FM.



Function Builder: Initial Screen
Function Module Edit Goto Utilities Environment System Help
Reassign...
Copy Function Module
Copy

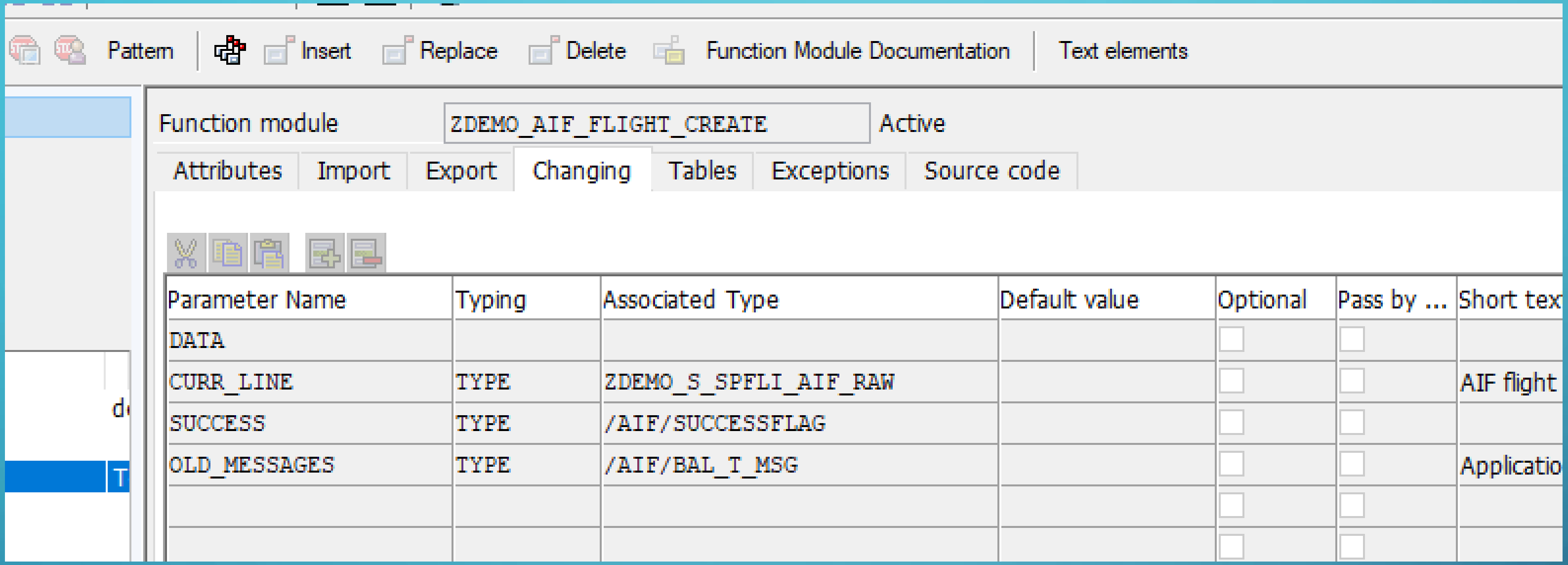
Function Module /AIF/FILE_TEMPL_PROCESS

fr. Function module /AIF/FILE_TEMPL_PROCESS

To Function module zdemo_aif_flight_create

Function group ZDEMO_AIF

IN CHANGING SECTION – FOR THE PARAMETER- CURR_LIEN PROVIDE THE ASSOCIATED TYPE AS OUR SAP STRUCTURE TYPE.



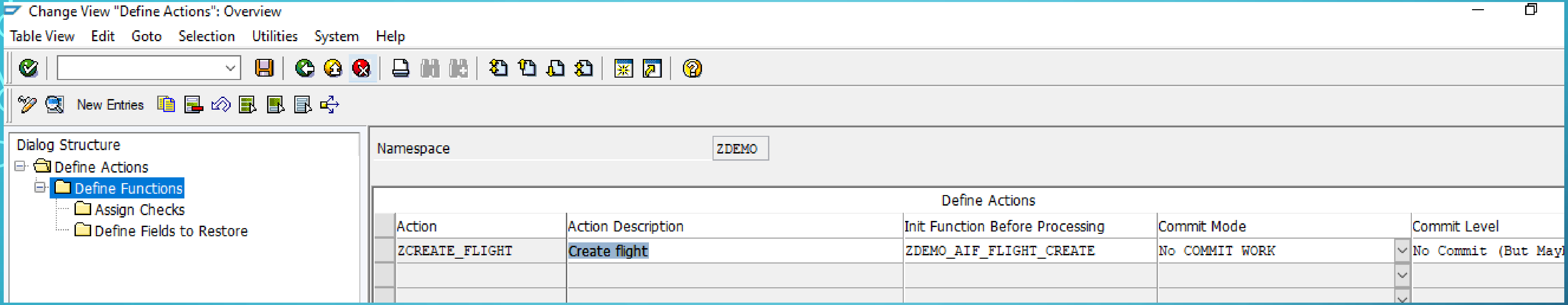
Function module ZDEMO_AIF_FLIGHT_CREATE Active

Parameter Name	Typing	Associated Type	Default value	Optional	Pass by ...	Short tex
DATA				□	□	
CURR_LINE	TYPE	ZDEMO_S_SPFLI_AIF_RAW		□	□	AIF flight
SUCCESS	TYPE	/AIF/SUCCESSFLAG		□	□	
OLD_MESSAGES	TYPE	/AIF/BAL_T_MSG		□	□	Applicatio
				□	□	
				□	□	

For FLIGHT demo, the below demo code is provided. The FM has a parameter called TESTRUN which is set when you test message in AIF for trouble shooting. This is kine some simulation mode and make sure that if TESTRUN is set no data base update happens.

```
IF CURR_LINE IS NOT INITIAL.
  SELECT SINGLE * FROM SPFLI INTO @DATA(LS_SPFLI)
    WHERE CARRID = @CURR_LINE-CARRID AND CONNID = @CURR_LINE-CONNID.
  IF SY-SUBRC IS NOT INITIAL.
    IF TESTRUN NE ABAP_TRUE .
      INSERT INTO SPFLI VALUES CURR_LINE.
    ENDIF.
    APPEND INITIAL LINE TO RETURN_TAB ASSIGNING FIELD-SYMBOL(<FS_RET>).
    <FS_RET>-ID = 'SAPABAPDEMOS'.
    <FS_RET>-NUMBER = '888'.
    <FS_RET>-TYPE = 'S'.
    <FS_RET>-MESSAGE_V1 = 'Record inserted successfully'.
  ELSE.
    APPEND INITIAL LINE TO RETURN_TAB ASSIGNING <FS_RET>.
    <FS_RET>-ID = 'SAPABAPDEMOS'.
    <FS_RET>-NUMBER = '888'.
    <FS_RET>-TYPE = 'E'.
    <FS_RET>-MESSAGE_V1 = 'Record already present'.
  ENDIF.
 ELSE.
  APPEND INITIAL LINE TO RETURN_TAB ASSIGNING <FS_RET>.
  <FS_RET>-ID = 'SAPABAPDEMOS'.
  <FS_RET>-NUMBER = '888'.
  <FS_RET>-TYPE = 'E'.
  <FS_RET>-MESSAGE_V1 = 'Empty information'.
 ENDIF.
```

One the action FM is ready then mention the FM name in the AIF action. Save and go back.

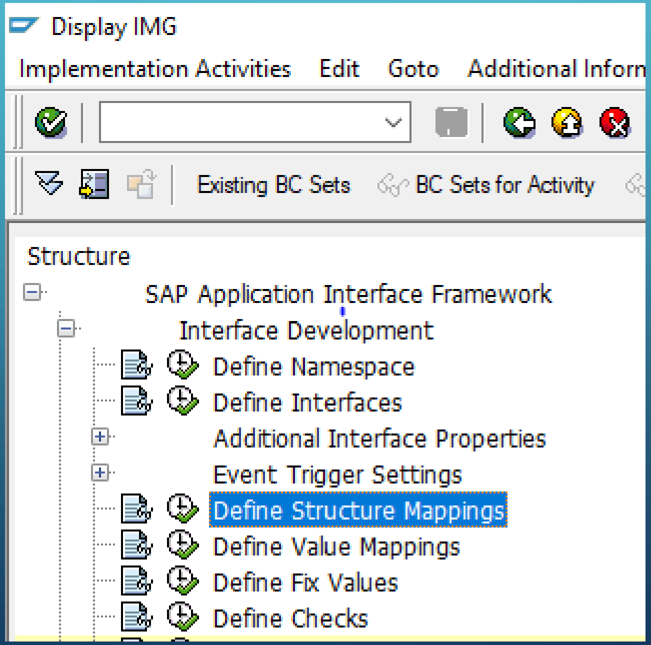


Namespace ZDEMO

Define Actions

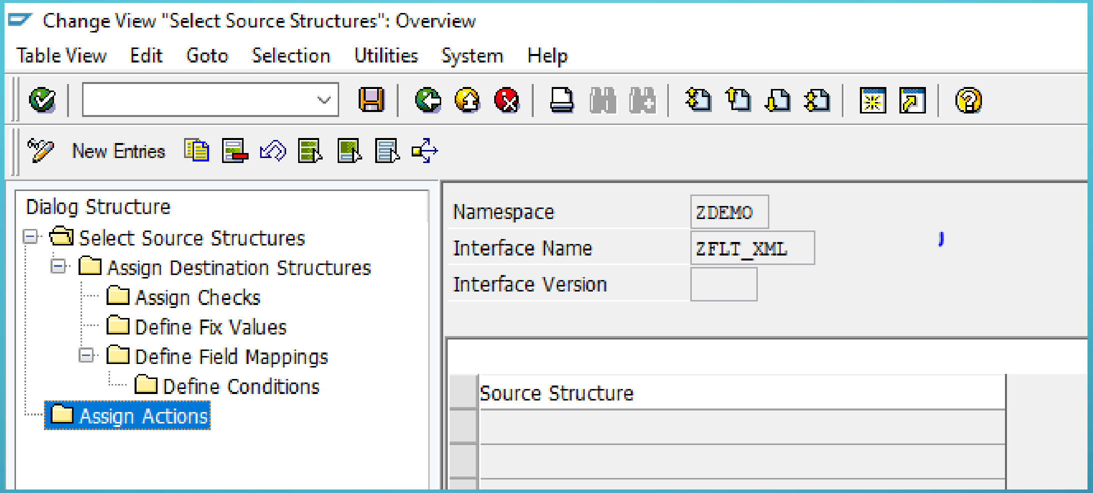
Action	Action Description	Init Function Before Processing	Commit Mode	Commit Level
ZCREATE_FLIGHT	Create flight	ZDEMO_AIF_FLIGHT_CREATE	No COMMIT WORK	No Commit (But May

The actions are defined on the Namespace level and now ot has to be assigned to the AIF Interface. Choose the option Define Structure Mapping.



Provide the namespace, interface and version and continue.

In interface Structure Mapping – here a lot of things can be done like mappings etc. But for this case we don't need any thing. Now choose Assign Actions option from left side panel.


Change View "Select Source Structures": Overview

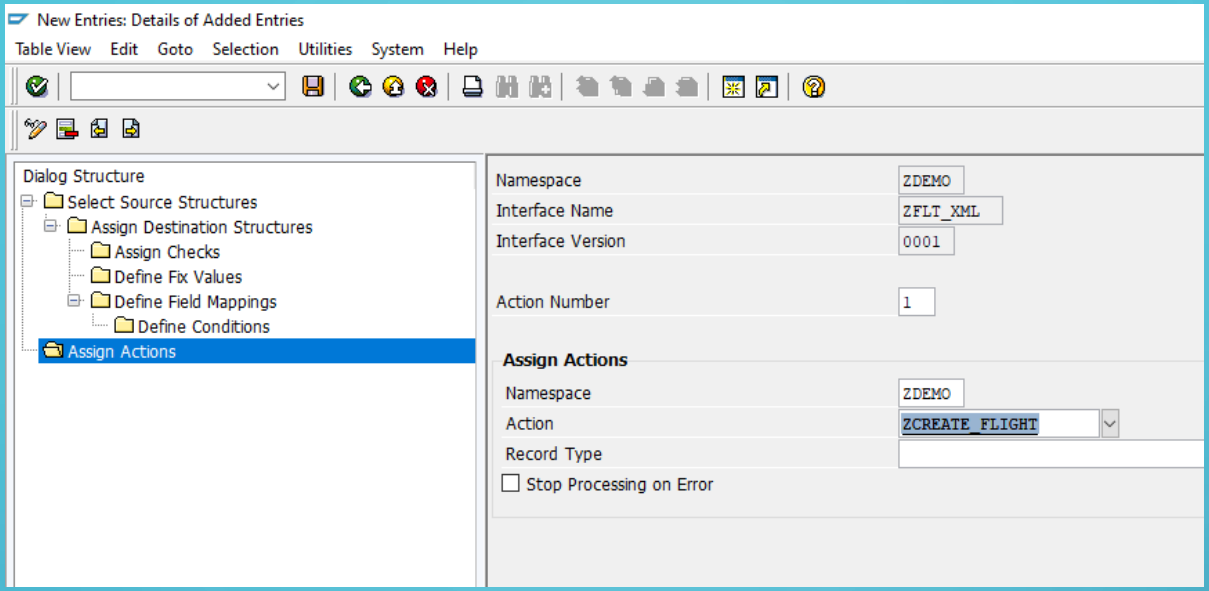
Dialog Structure
- Select Source Structures
  - Assign Destination Structures
    - Assign Checks
    - Define Fix Values
    - Define Field Mappings
      - Define Conditions
- Assign Actions

Namespace	ZDEMO
Interface Name	ZFLT_XML
Interface Version	

Source Structure

Choose New Entries option.

Provide the action number and choose namespace and the action name. Save and go back.


New Entries: Details of Added Entries
Table View Edit Goto Selection Utilities System Help

Dialog Structure

- Select Source Structures
  - Assign Destination Structures
    - Assign Checks
    - Define Fix Values
    - Define Field Mappings
      - Define Conditions
- Assign Actions

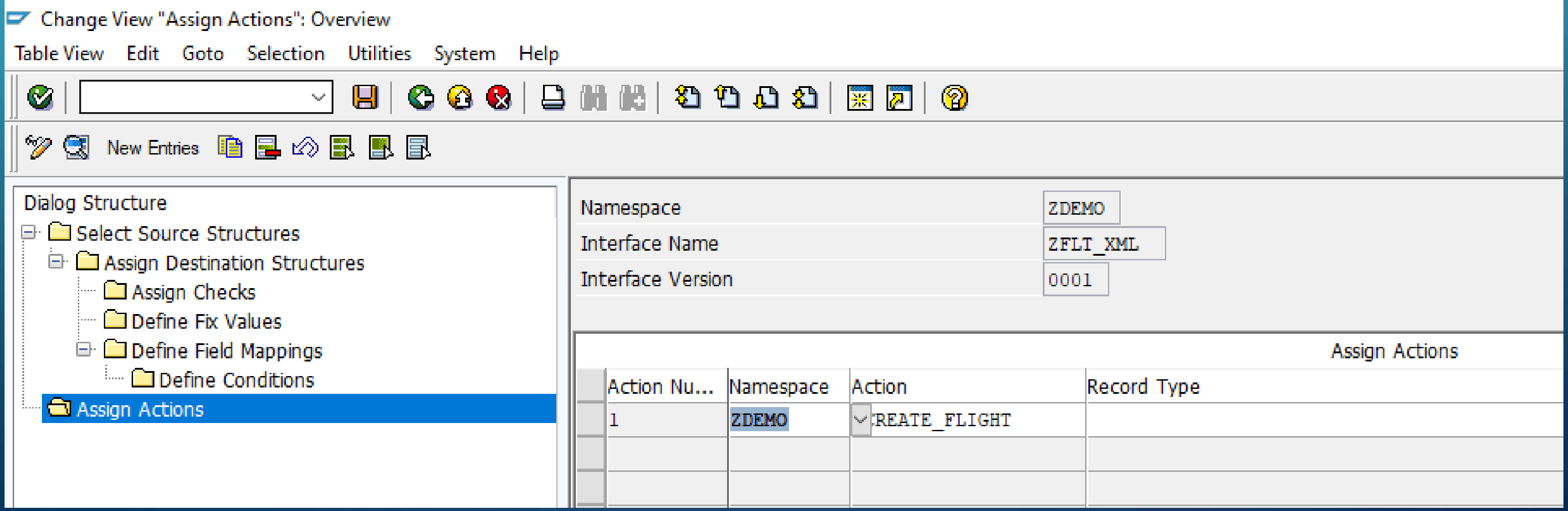
Namespace	ZDEMO
Interface Name	ZFLT_XML
Interface Version	0001
Action Number	1

Assign Actions

Namespace	ZDEMO
Action	ZCREATE_FLIGHT
Record Type	

☐ Stop Processing on Error

Now an action is assigned to the interface.


Change View "Assign Actions": Overview
Table View Edit Goto Selection Utilities System Help
New Entries

Dialog Structure

- Select Source Structures
  - Assign Destination Structures
    - Assign Checks
    - Define Fix Values
    - Define Field Mappings
      - Define Conditions
- Assign Actions

Namespace	ZDEMO
Interface Name	ZFLT_XML
Interface Version	0001

Assign Actions

Action Nu...	Namespace	Action	Record Type
1	ZDEMO	REATE_FLIGHT	