TCODES:

SE11

SE37

SEGW

/IWFND/MAINT\_SERVICE

Z-Tables are created in SE11 :

1. ZLFA1\_MASTER - Vendor Master Table
2. ZEKKO\_HEADER - Purchase Order Header Table
3. ZEKPO\_ITEM - Purchase Order Item Table

Creating Project:

By using ‘SEGW’ tcode we can create our project.

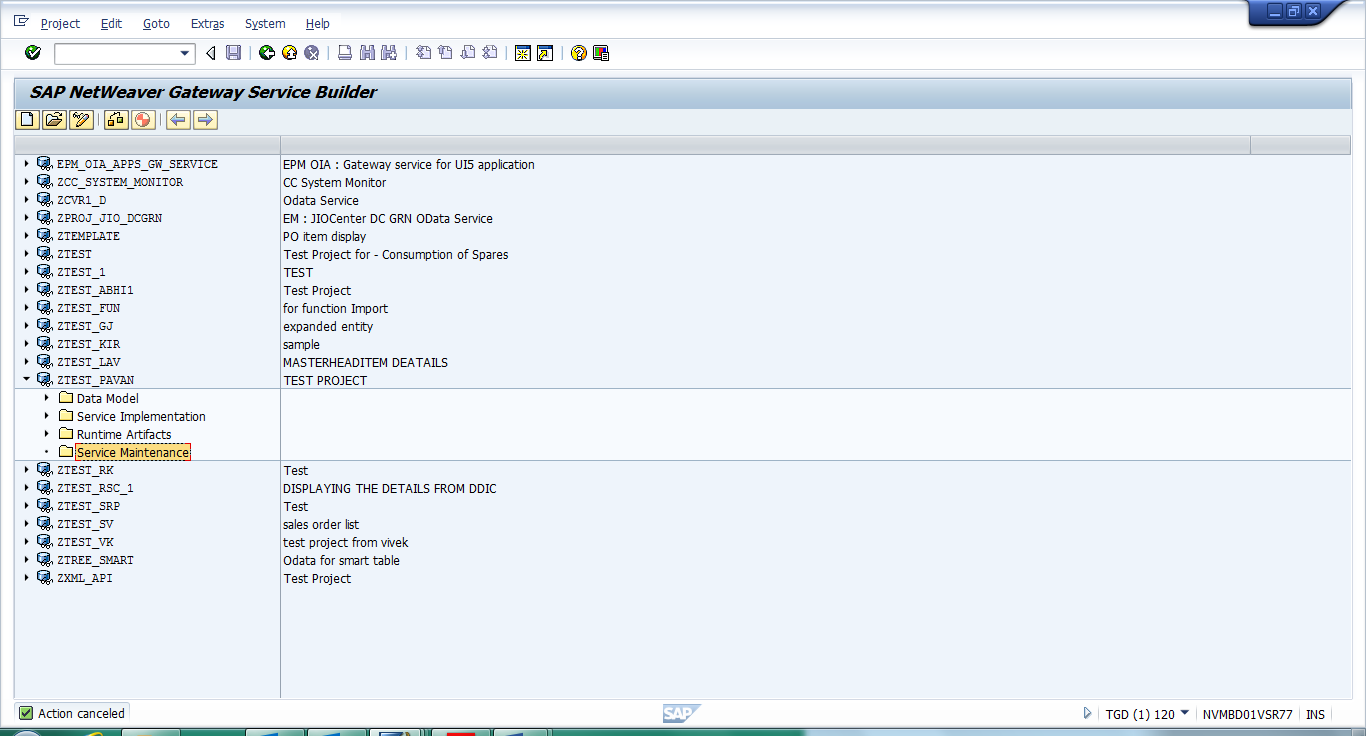
Execute ‘SEGW’ tcode.

Click on ‘CREATE’ in Application Tool Bar.

Then Give Project name: ZTEST\_PAVAN and Description: Test Project.



Click on Local Object.

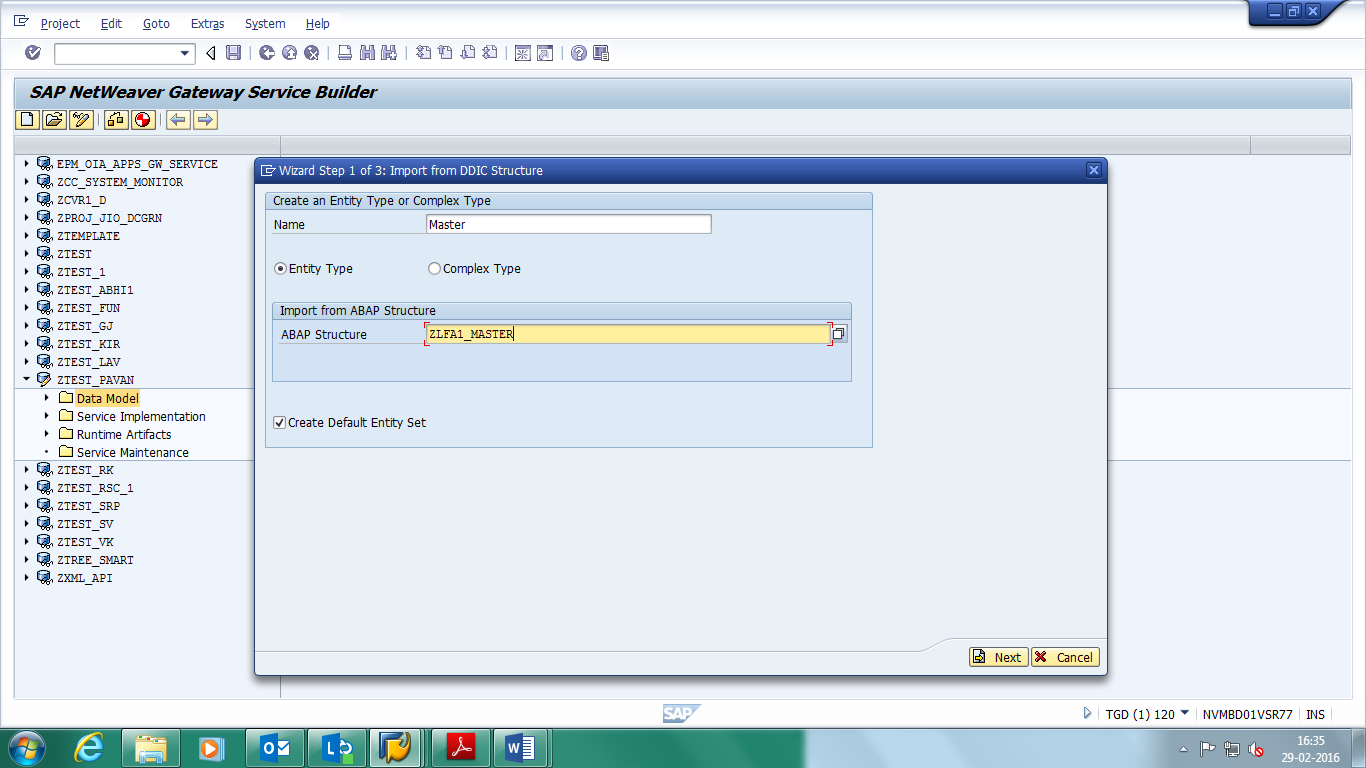


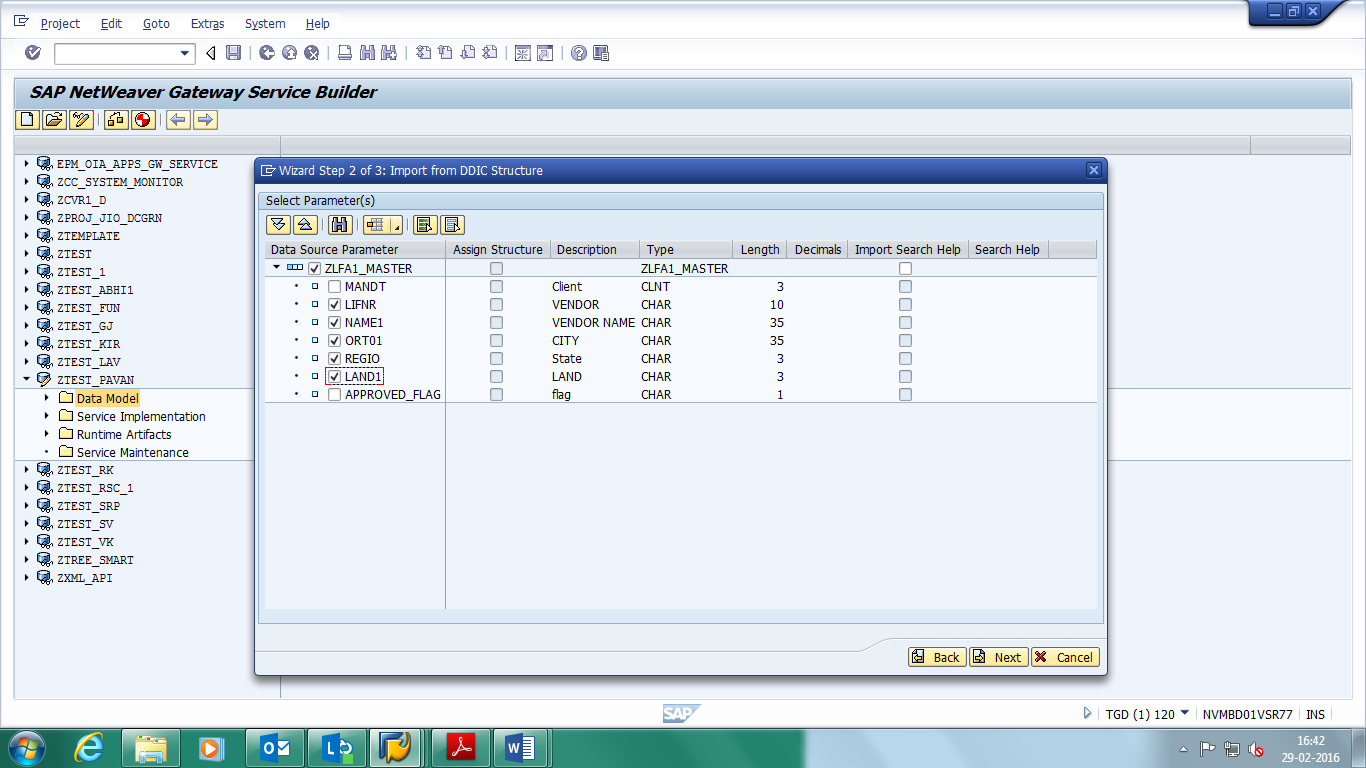
**Creating ENTITY TYPE by Using DDIC Structure:**

**Master Entity Type:-**

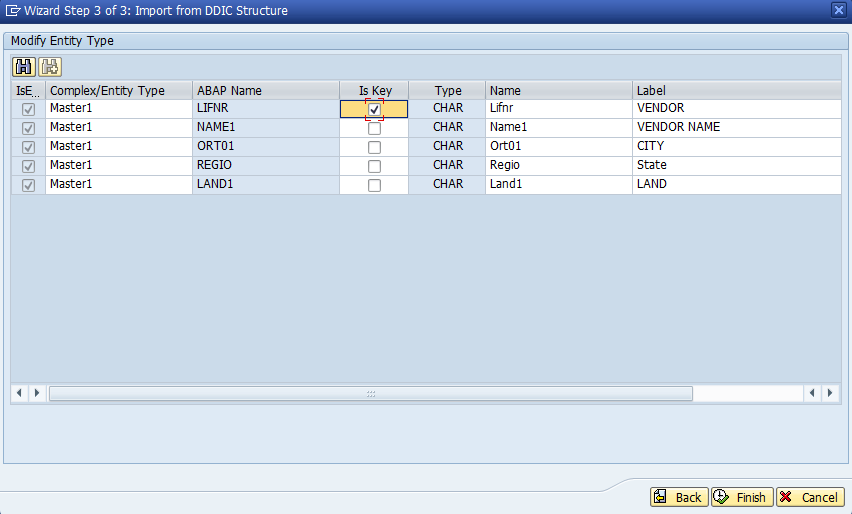
Expand The Project ‘ZTEST\_PAVAN’.

Select The Data Model and Right Click on it -> IMPORT -> DDIC Structure.

Provide Entity Name: ‘Master’, ABAP Structure Name: ‘ZLFA1\_MASTER’ Click on Next.

Select our Required Fields and Click on Next.

Select the check Box Is Key and Click on Finish.



Click on Finish.

Note:- Entity Sets are Automatically Created at the Time of When We Creating Entity Types.

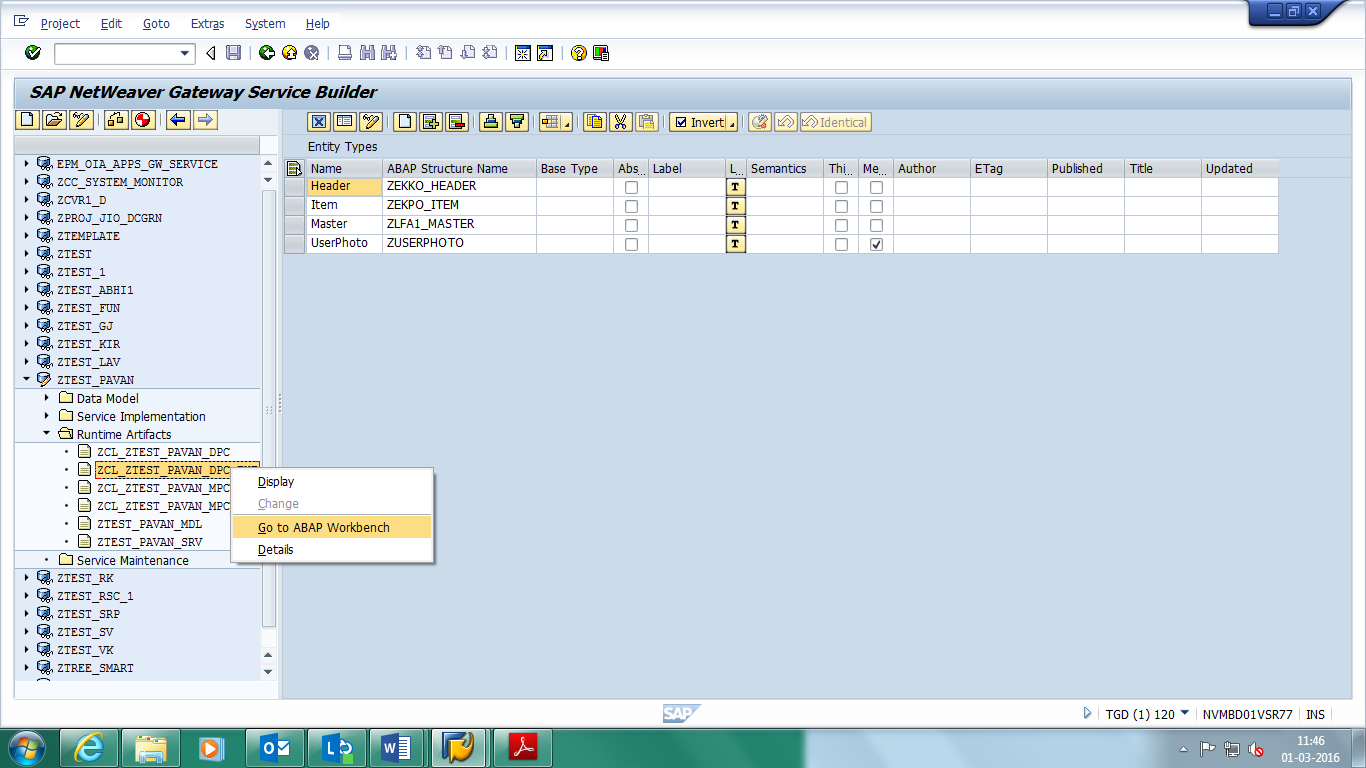
Save Project, Click on Check Project Consistency and Click on Generate Runtime Objects.

Then Automatically Four Classes will generate.

1. ZCL\_ZTEST\_PAVAN\_DPC
2. ZCL\_ZTEST\_PAVAN\_DPC\_EXT
3. ZCL\_ZTEST\_PAVAN\_MPC
4. ZCL\_ZTEST\_PAVAN\_MPC\_EXT

To Implement Methods Expand the Runtime Artifacts.

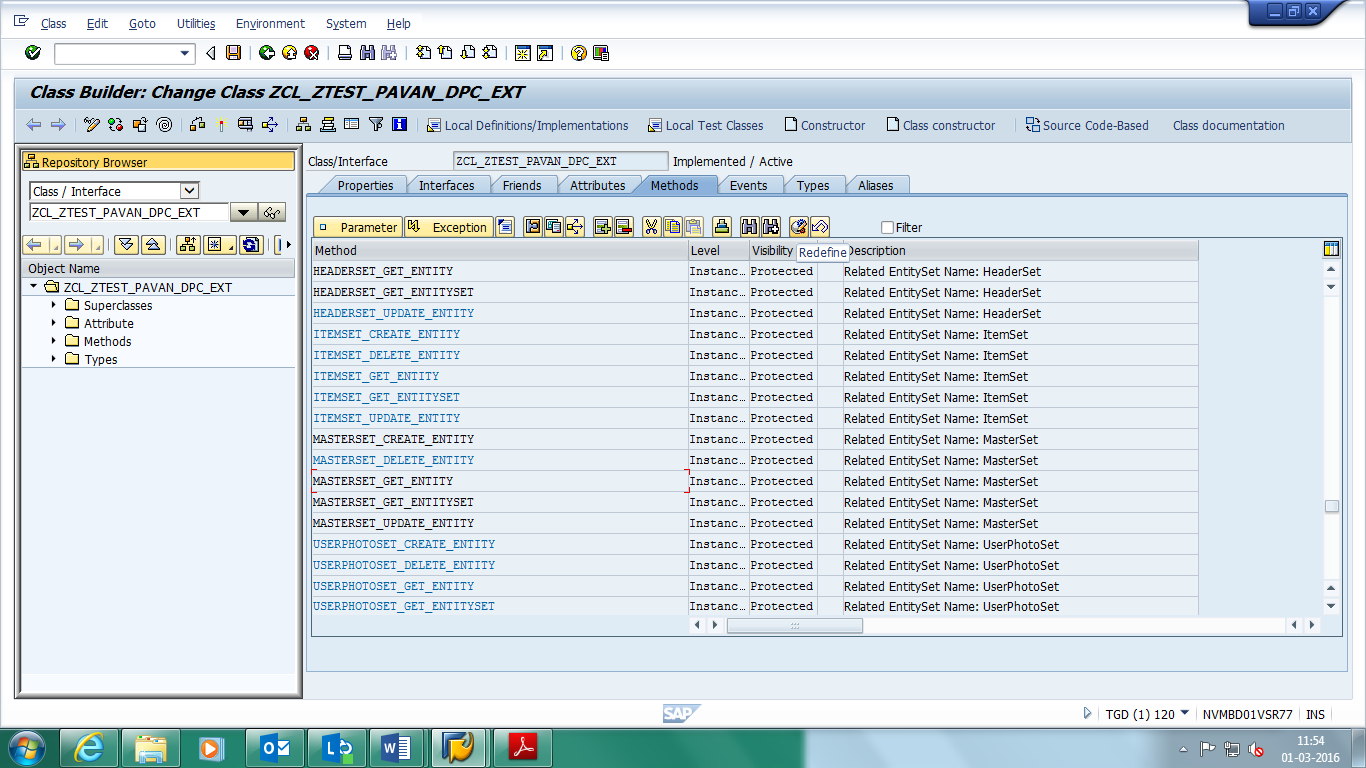
Select the ‘ZCL\_ZTEST\_PAVAN\_DPC\_EXT’ -> Right Click on it -> Go to Abap Work Bench



Select the Method which one we Want to Implement

-> Click on Edit Mode

-> Click on Redefine



**GET\_ENTITY Method:-**This is the method for the read operation for an entity / entry.

**Parameters Used in GET\_ENTITY:**

**IT\_KEY\_TAB:** This is an Input Parameter Which Contains Two Fields Name and Value. it is Used to Represents the keys of the First Segment.

Type /IWBEP/T\_MGW\_NAME\_VALUE\_PAIR

**ER\_ENTITY:** This is Output Parameter Which Returns Reference to the Application Specific Structure Which Contains the Data and it is Used to Display the Output in HTTP Response.

TYPE ZCL\_ZTEST\_PAVAN\_MPC=>TS\_MASTER

**Get\_Entity Code for MASTERSET\_GET\_ENTITY:-**

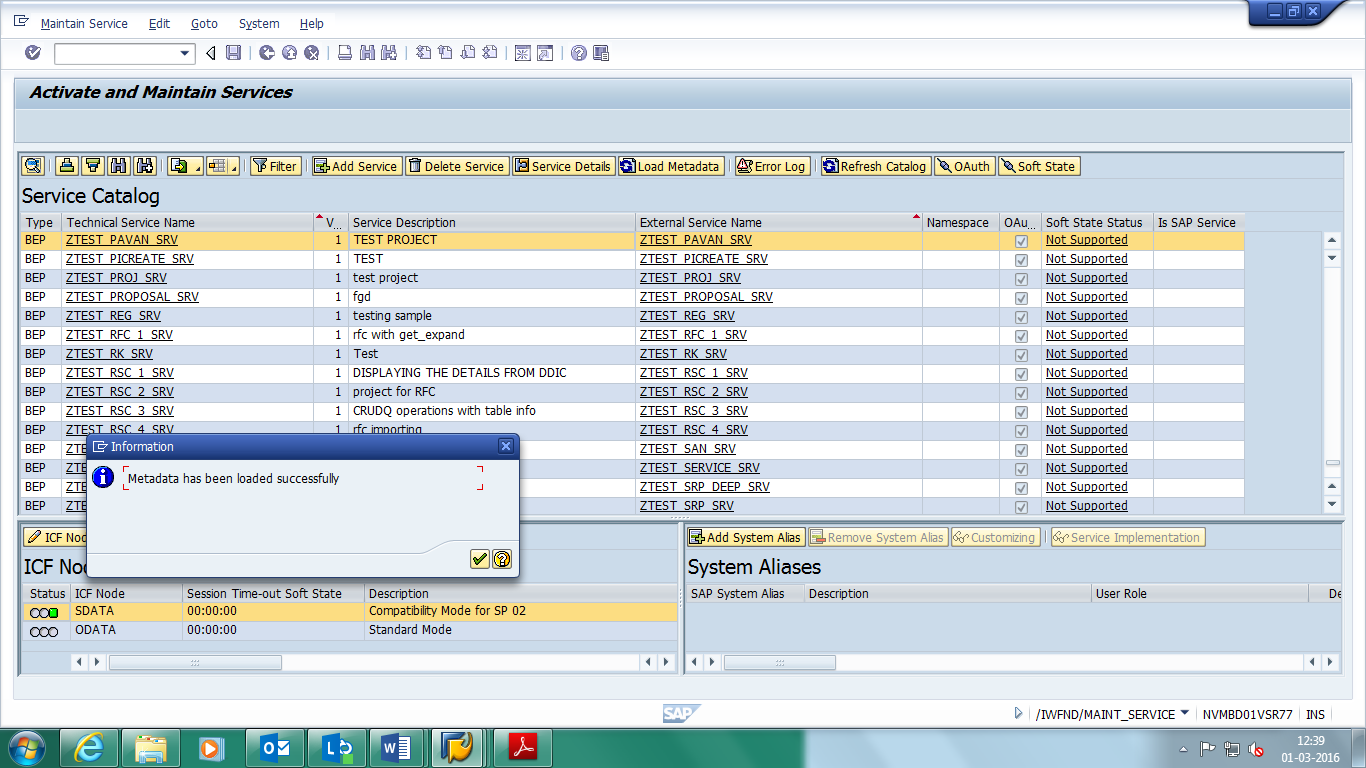
  METHOD masterset\_get\_entity.  
    DATA: ls\_master    TYPE zlfa1\_master,  
          ls\_key\_tab   LIKE LINE OF it\_key\_tab,  
          lv\_lifnr     TYPE lifnr,  
          ls\_return    TYPE bapiret2,  
          lo\_container TYPE REF TO /iwbep/if\_message\_container.  
  
    READ TABLE it\_key\_tab INTO ls\_key\_tab WITH KEY  
                               name = 'VendorAccountNumber'.  
    IF sy-subrc = 0.  
      lv\_lifnr = ls\_key\_tab-value.  
      UNPACK lv\_lifnr TO lv\_lifnr.  
    ENDIF.  
  
    SELECT SINGLE \* FROM zlfa1\_master INTO ls\_master WHERE lifnr = lv\_lifnr.  
  
    IF sy-subrc EQ 0.  
  
      er\_entity-lifnr = ls\_master-lifnr.  
      er\_entity-name1 = ls\_master-name1.  
      er\_entity-ort01 = ls\_master-ort01.  
      er\_entity-regio = ls\_master-regio.  
      er\_entity-land1 = ls\_master-land1.  
  
    ELSE.  
  
      ls\_return-type = 'E'.  
      ls\_return-message = 'No Data Found'.  
  
      lo\_container = me->mo\_context->get\_message\_container( ).  
  
      CALL METHOD lo\_container->add\_message  
        EXPORTING  
          iv\_msg\_type   = ls\_return-type  
          iv\_msg\_id     = ls\_return-id  
          iv\_msg\_number = ls\_return-number  
          iv\_msg\_text   = ls\_return-message.  
  
      RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
        EXPORTING  
          message\_container = lo\_container.  
    ENDIF.  
  
  ENDMETHOD.

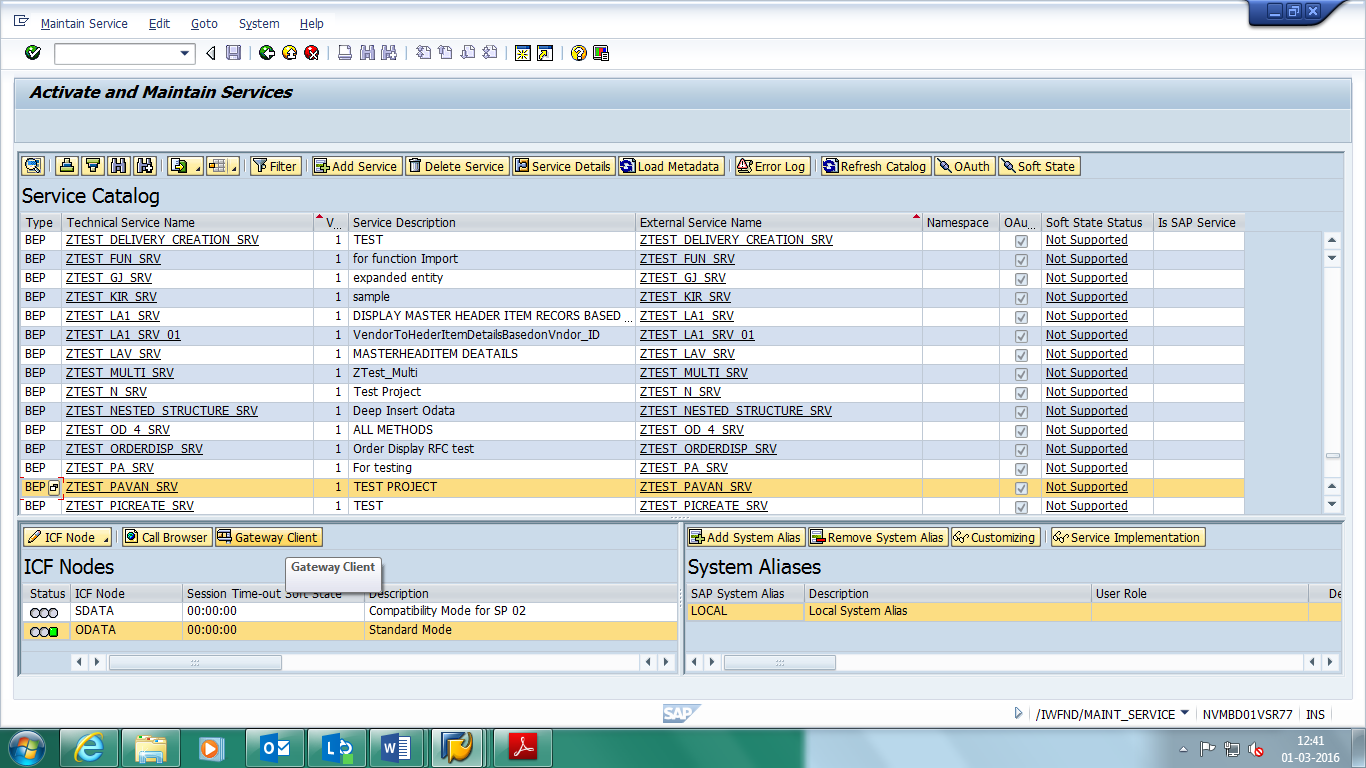
Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

-> Select our Service

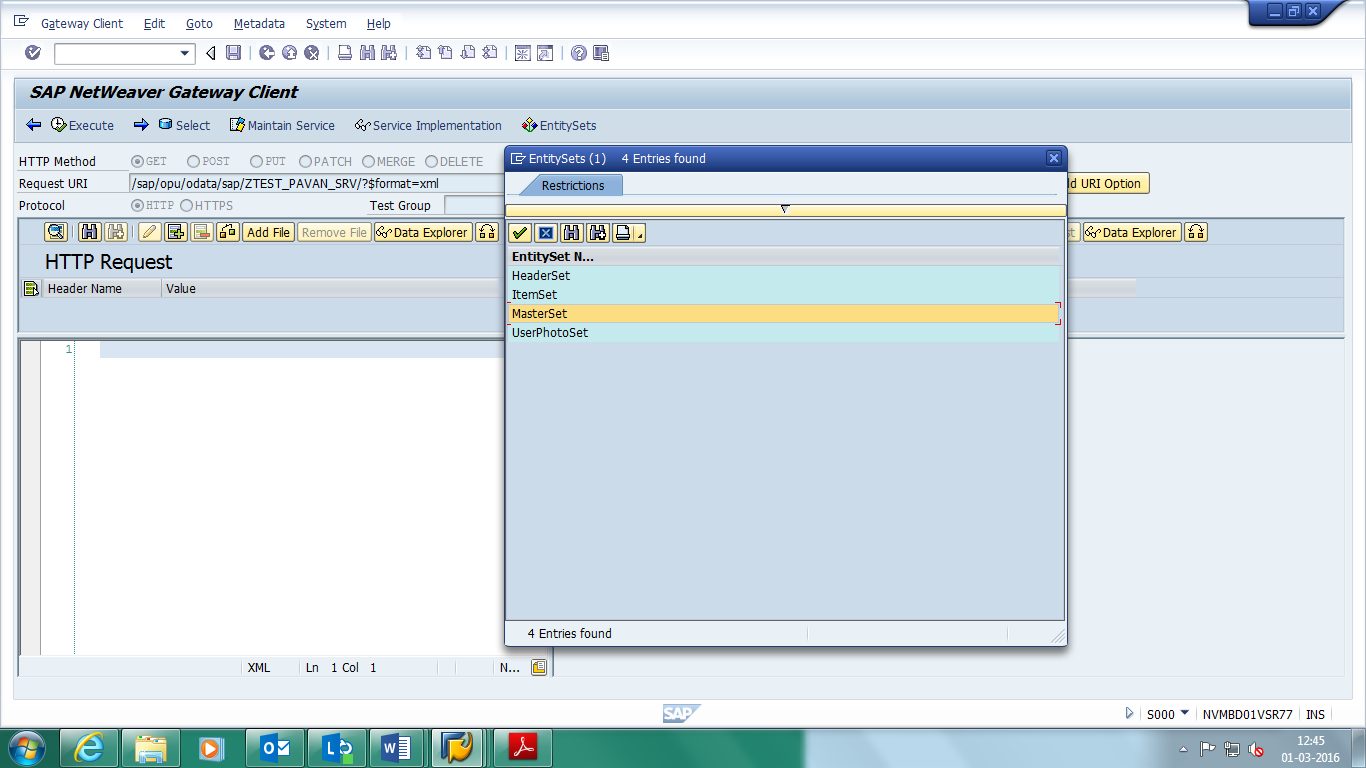
-> Click on Load Metadata



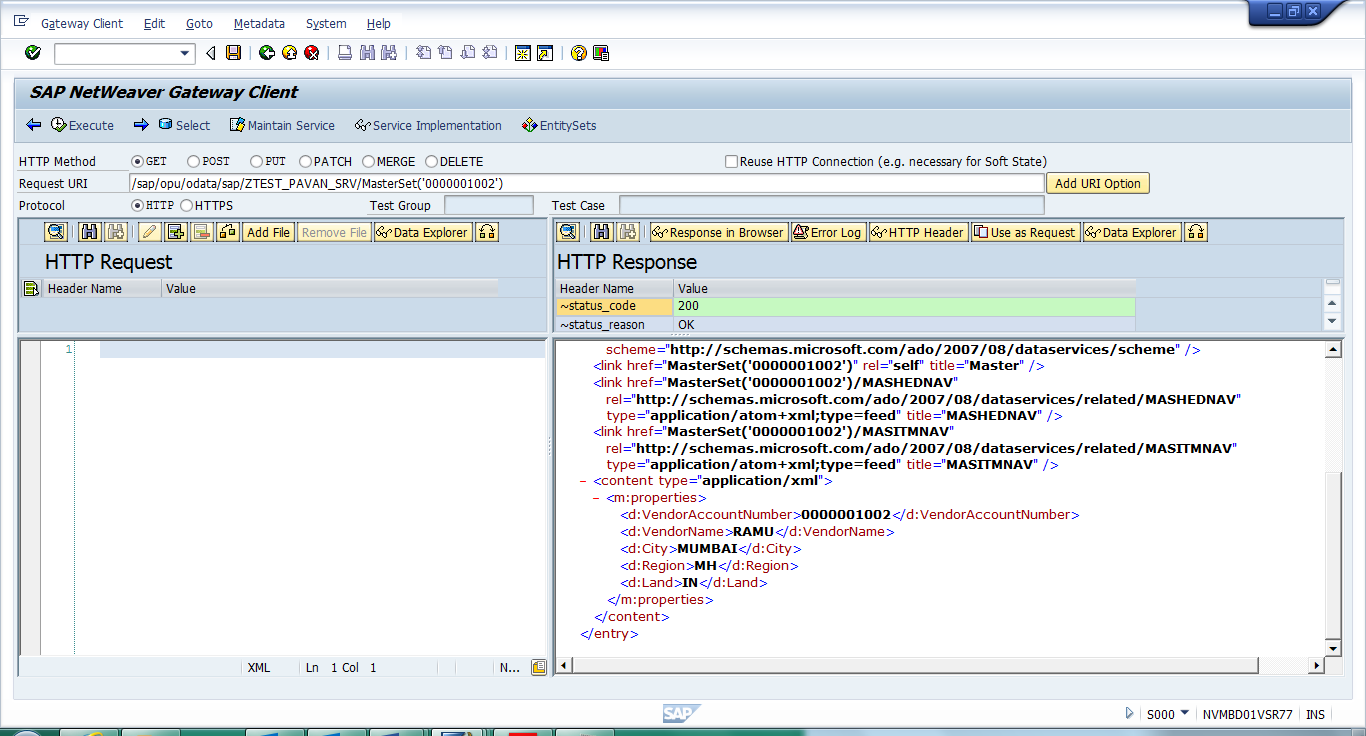
-> Click on Gateway Client

-> Click on EntitySets and Select Entityset Name

-> Select the HTTP Method ‘Get’



**URL:** /sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/MasterSet('0000001002')



**GET\_ENTITYSET Method:-** This Method Can be Used for a Read/Query Operation for an Expanded Set of Entities. This Method for a (query) Operation Returns a List of Entities and so Allows to do Filtering, Sorting, Free Text Search, etc.

**Parameters Used in GET\_ENTITYSET:**

**IT\_FILTER\_SELECT\_OPTIONS:** This is Input Parameter and Used for filter parameters as an ABAP Select-options Table for the Filter Parameter.

Type /IWBEP/T\_MGW\_SELECT\_OPTION

**ER\_ENTITYSET:** This is Output Parameter Which Returns Reference to the Application Specific Structure Which Contains the Data and it is Used to Display the Output in HTTP Response.

TYPE ZCL\_ZTEST\_PAVAN\_MPC=>TT\_MASTER

**Get\_Entity Code for MASTERSET\_GET\_ENTITYSET:-**

method MASTERSET\_GET\_ENTITYSET.  
  
 DATA: LS\_MASHED TYPE ZCL\_ZTEST\_PAVAN\_MPC=>ts\_master,  
          LS\_LFA1   TYPE ZLFA1\_MASTER,  
          LT\_LFA1   TYPE STANDARD TABLE OF ZLFA1\_MASTER,  
          ls\_filter       TYPE /iwbep/s\_mgw\_select\_option,  
          ls\_filter\_range TYPE /iwbep/s\_cod\_select\_option,  
          lo\_container type REF TO /iwbep/if\_message\_container,  
          ls\_return type bapiret2.  
  
    LOOP AT IT\_FILTER\_SELECT\_OPTIONS INTO ls\_filter.  
  
    CASE ls\_filter-property.  
      WHEN 'VendorAccountNumber'.  
    READ TABLE ls\_filter-SELECT\_OPTIONS INTO ls\_filter\_range INDEX 1.  
    IF sy-subrc = 0.  
      SELECT \* FROM ZLFA1\_MASTER INTO TABLE LT\_LFA1 WHERE lifnr = LS\_FILTER\_RANGE-LOW.  
    ENDIF.  
    ENDCASE.  
    ENDLOOP.  
  
    if lt\_lfa1 is NOT INITIAL.  
  
        LOOP AT LT\_LFA1 INTO LS\_LFA1.  
        LS\_MASHED-LIFNR = LS\_LFA1-LIFNR.  
        LS\_MASHED-NAME1 = LS\_LFA1-NAME1.  
        LS\_MASHED-ORT01 = LS\_LFA1-ORT01.  
        LS\_MASHED-REGIO = LS\_LFA1-REGIO.  
        LS\_MASHED-LAND1 = LS\_LFA1-LAND1.  
        APPEND LS\_MASHED TO ET\_ENTITYSET.  
        ENDLOOP.  
  
     else.  
  
      ls\_return-type = 'E'.  
      ls\_return-message = 'No Data Found'.  
  
      lo\_container = me->mo\_context->get\_message\_container( ).  
  
      CALL METHOD lo\_container->add\_message  
        EXPORTING  
          iv\_msg\_type   = ls\_return-type  
          iv\_msg\_id     = ls\_return-id  
          iv\_msg\_number = ls\_return-number  
          iv\_msg\_text   = ls\_return-message.  
  
      RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
        EXPORTING  
          message\_container = lo\_container.  
    ENDIF.  
  
  endmethod.

Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

-> Select our Service

-> Click on Load Metadata

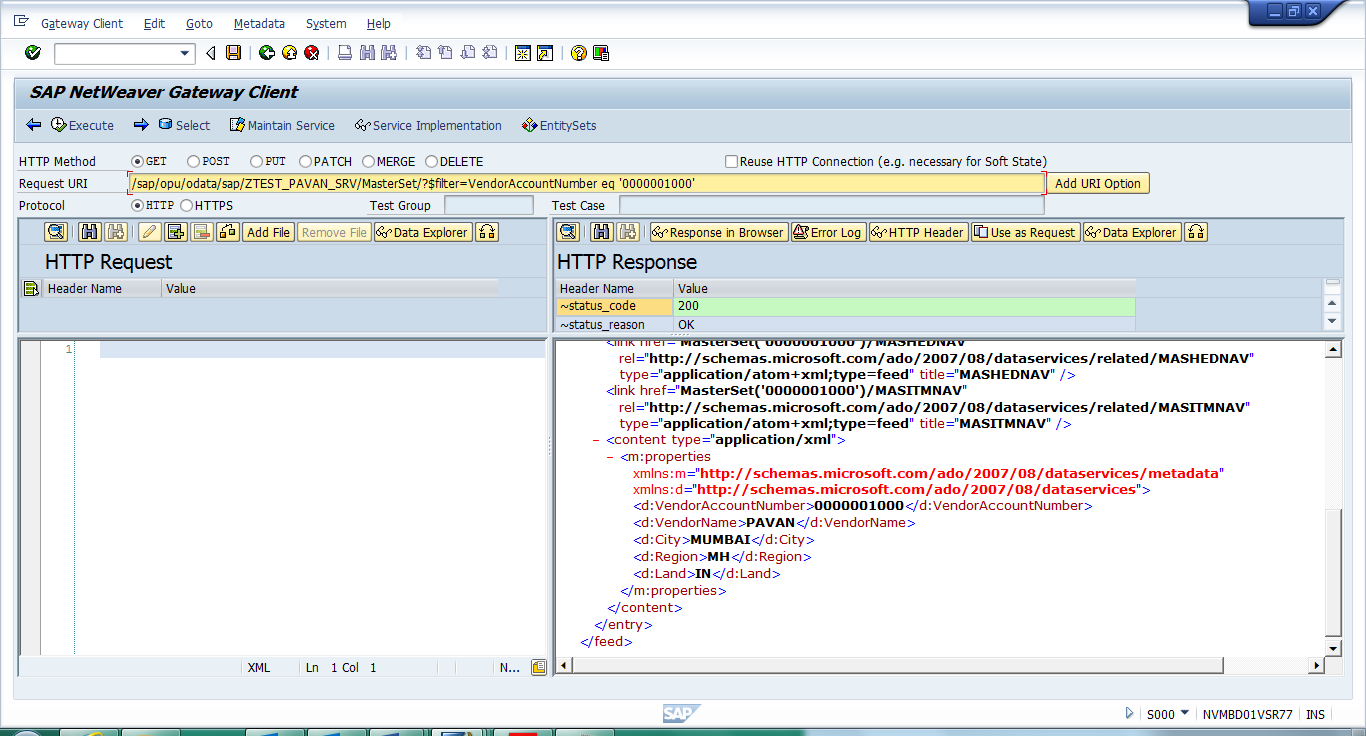
-> Click on Gateway Client

-> Click on EntitySets and Select Entityset Name

-> Select the HTTP Method ‘Get’

-> Write the Payload in HTTP Request

**URL:**/sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/MasterSet?$filter=VendorAccountNumber eq '0000001000'



**CREATE\_ENTITY Method:-**This is the method for the Create operation for an entity / entry.

**Parameters Used in CREATE\_ENTITY:**

**IO\_DATA\_PROVIDER:** The interface to deserialize the payload into the application specific structure.

Type /IWBEP/IF\_MGW\_ENTRY\_PROVIDER

**ER\_ENTITY:** This is Output Parameter Which Returns Reference to the Application Specific Structure Which Contains the Data and it is Used to Display the Output in HTTP Response.

TYPE ZCL\_ZTEST\_PAVAN\_MPC=>TS\_MASTER

**Create\_Entity Code for MASTERSET\_CREATE\_ENTITY:-**

method MASTERSET\_CREATE\_ENTITY.  
  
  
    DATA: ls\_master    TYPE zcl\_ztest\_pavan\_mpc=>ts\_master,  
          ls\_lfa1      TYPE zlfa1\_master,  
          ls\_return    TYPE bapiret2,  
          lt\_return    TYPE bapiret2\_t,  
          lo\_container TYPE REF TO /iwbep/if\_message\_container,  
          lw\_lheader   TYPE ihttpnvp,  
          json         TYPE string,  
          lo\_message   TYPE REF TO zcl\_fiori\_format\_message\_json.  
  
  
    io\_data\_provider->read\_entry\_data( IMPORTING es\_data = ls\_master ).  
  
    ls\_lfa1-lifnr = ls\_master-lifnr.  
    ls\_lfa1-name1 = ls\_master-name1.  
    ls\_lfa1-ort01 = ls\_master-ort01.  
    ls\_lfa1-regio = ls\_master-regio.  
    ls\_lfa1-land1 = ls\_master-land1.  
  
    INSERT zlfa1\_master FROM ls\_lfa1.  
    IF sy-subrc EQ 0.  
      COMMIT WORK.  
  
      ls\_return-type = 'S'.  
      ls\_return-message = 'Data Updated Successfully'.  
      APPEND ls\_return TO lt\_return.  
  
      CREATE OBJECT lo\_message.  
      CALL METHOD lo\_message->message\_format  
        EXPORTING  
          im\_returnmsg = lt\_return  
        IMPORTING  
          ex\_json      = json.  
  
      lw\_lheader-name = 'sap-message'.  
      lw\_lheader-value = json.  
      set\_header( is\_header = lw\_lheader ).  
  
    ELSE.  
      ls\_return-type = 'E'.  
      ls\_return-message = 'Error in updating data '.  
  
      lo\_container = me->mo\_context->get\_message\_container( ).  
  
      CALL METHOD lo\_container->add\_message  
        EXPORTING  
          iv\_msg\_type   = ls\_return-type  
          iv\_msg\_id     = ls\_return-id  
          iv\_msg\_number = ls\_return-number  
          iv\_msg\_text   = ls\_return-message.  
  
      RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
        EXPORTING  
          message\_container = lo\_container.  
    ENDIF.  
  
    er\_entity = ls\_master.  
  endmethod.

Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

-> Select our Service

-> Click on Load Metadata

-> Click on Gateway Client

-> Click on EntitySets and Select Entityset Name

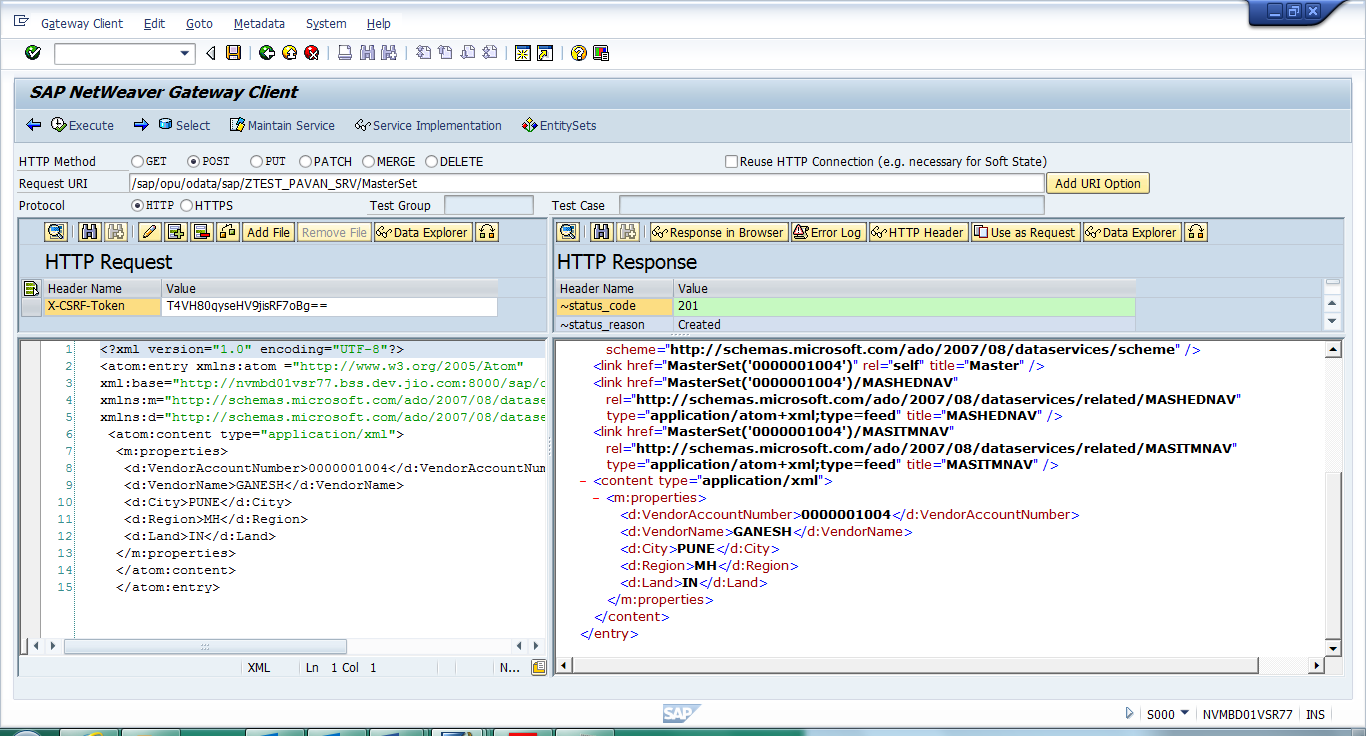
-> Select the HTTP Method ‘Post’

-> Write the Payload in HTTP Request

**URL:** /sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/MasterSet

**Payload:**

<?xml version="1.0" encoding="UTF-8"?>  
<atom:entry xmlns:atom ="http://www.w3.org/2005/Atom"  
xml:base="http://nvmbd01vsr77.bss.dev.jio.com:8000/sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/"  
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"  
xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices">  
 <atom:content type="application/xml">  
  <m:properties>  
   <d:VendorAccountNumber>0000001004</d:VendorAccountNumber>  
   <d:VendorName>MANESH</d:VendorName>  
   <d:City>MUM</d:City>  
   <d:State>MP</d:State>  
   <d:Country>IN</d:Country>  
  </m:properties>  
  </atom:content>  
  </atom:entry>



**UPDATE\_ENTITY Method:-**This is the method for the Update operation for an entity / entry.

**Parameters Used in UPDATE\_ENTITY:**

**IO\_DATA\_PROVIDER:** The interface to deserialize the payload into the application specific structure.

Type /IWBEP/IF\_MGW\_ENTRY\_PROVIDER

**IT\_KEY\_TAB:** This is an Input Parameter Which Contains Two Fields Name and Value. it is Used to Represents the keys of the First Segment.

Type /IWBEP/T\_MGW\_NAME\_VALUE\_PAIR

**Update\_Entity Code for MASTERSET\_UPDATE\_ENTITY:-**

  METHOD masterset\_update\_entity.  
    DATA: ls\_master    TYPE zcl\_ztest\_pavan\_mpc=>ts\_master,  
          ls\_lfa1      TYPE zlfa1\_master,  
          ls\_return    TYPE bapiret2,  
          lt\_return    TYPE bapiret2\_t,  
          lv\_lifnr     TYPE lifnr,  
          lv\_varkey    TYPE vim\_enqkey,  
          ls\_key\_tab   TYPE /iwbep/s\_mgw\_name\_value\_pair,  
          lo\_container TYPE REF TO /iwbep/if\_message\_container,  
          lw\_lheader   TYPE ihttpnvp,  
          json         TYPE string,  
          lo\_message   TYPE REF TO zcl\_fiori\_format\_message\_json.  
  
  
    io\_data\_provider->read\_entry\_data( IMPORTING es\_data = ls\_master ).  
  
    READ TABLE it\_key\_tab INTO ls\_key\_tab WITH KEY  
                               name = 'VendorAccountNumber'.  
    IF sy-subrc = 0.  
      lv\_lifnr = ls\_key\_tab-value.  
    ENDIF.  
  
    SELECT SINGLE \* FROM zlfa1\_master INTO ls\_lfa1 WHERE lifnr = lv\_lifnr.  
    IF sy-subrc EQ 0.  
  
      ls\_lfa1-name1 = ls\_master-name1.  
      ls\_lfa1-ort01 = ls\_master-ort01.  
      ls\_lfa1-regio = ls\_master-regio.  
      ls\_lfa1-land1 = ls\_master-land1.  
  
      CONCATENATE sy-mandt lv\_lifnr INTO lv\_varkey.  
  
      CALL FUNCTION 'ENQUEUE\_E\_TABLE'  
        EXPORTING  
          mode\_rstable = 'E'  
          tabname      = 'ZLFA1\_MASTER'  
          varkey       = lv\_varkey  
          \_scope       = '3'.  
  
  
      UPDATE zlfa1\_master FROM ls\_lfa1.  
      IF sy-subrc EQ 0.  
        COMMIT WORK.  
  
        ls\_return-type = 'S'.  
        ls\_return-message = 'Data Updated Successfully'.  
        APPEND ls\_return TO lt\_return.  
  
        CREATE OBJECT lo\_message.  
        CALL METHOD lo\_message->message\_format  
          EXPORTING  
            im\_returnmsg = lt\_return  
          IMPORTING  
            ex\_json      = json.  
  
        lw\_lheader-name = 'sap-message'.  
        lw\_lheader-value = json.  
        set\_header( is\_header = lw\_lheader ).  
  
      ELSE.  
        ls\_return-type = 'E'.  
        ls\_return-message = 'Error in updating data '.  
  
        lo\_container = me->mo\_context->get\_message\_container( ).  
  
        CALL METHOD lo\_container->add\_message  
          EXPORTING  
            iv\_msg\_type   = ls\_return-type  
            iv\_msg\_id     = ls\_return-id  
            iv\_msg\_number = ls\_return-number  
            iv\_msg\_text   = ls\_return-message.  
  
        RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
          EXPORTING  
            message\_container = lo\_container.  
      ENDIF.  
  
        CALL FUNCTION 'DEQUEUE\_E\_TABLE'  
          EXPORTING  
            mode\_rstable = 'E'  
            tabname      = 'ZLFA1\_MASTER'  
            varkey       = lv\_varkey  
            \_scope       = '3'.  
  
    ELSE.  
      ls\_return-type = 'E'.  
      ls\_return-message = 'Invalid Vendor Num'.  
  
      lo\_container = me->mo\_context->get\_message\_container( ).  
  
      CALL METHOD lo\_container->add\_message  
        EXPORTING  
          iv\_msg\_type   = ls\_return-type  
          iv\_msg\_id     = ls\_return-id  
          iv\_msg\_number = ls\_return-number  
          iv\_msg\_text   = ls\_return-message.  
  
      RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
        EXPORTING  
          message\_container = lo\_container.  
    ENDIF.  
  
  ENDMETHOD.

Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

-> Select our Service

-> Click on Load Metadata

-> Click on Gateway Client

-> Click on EntitySets and Select Entityset Name

-> Select the HTTP Method ‘Put’

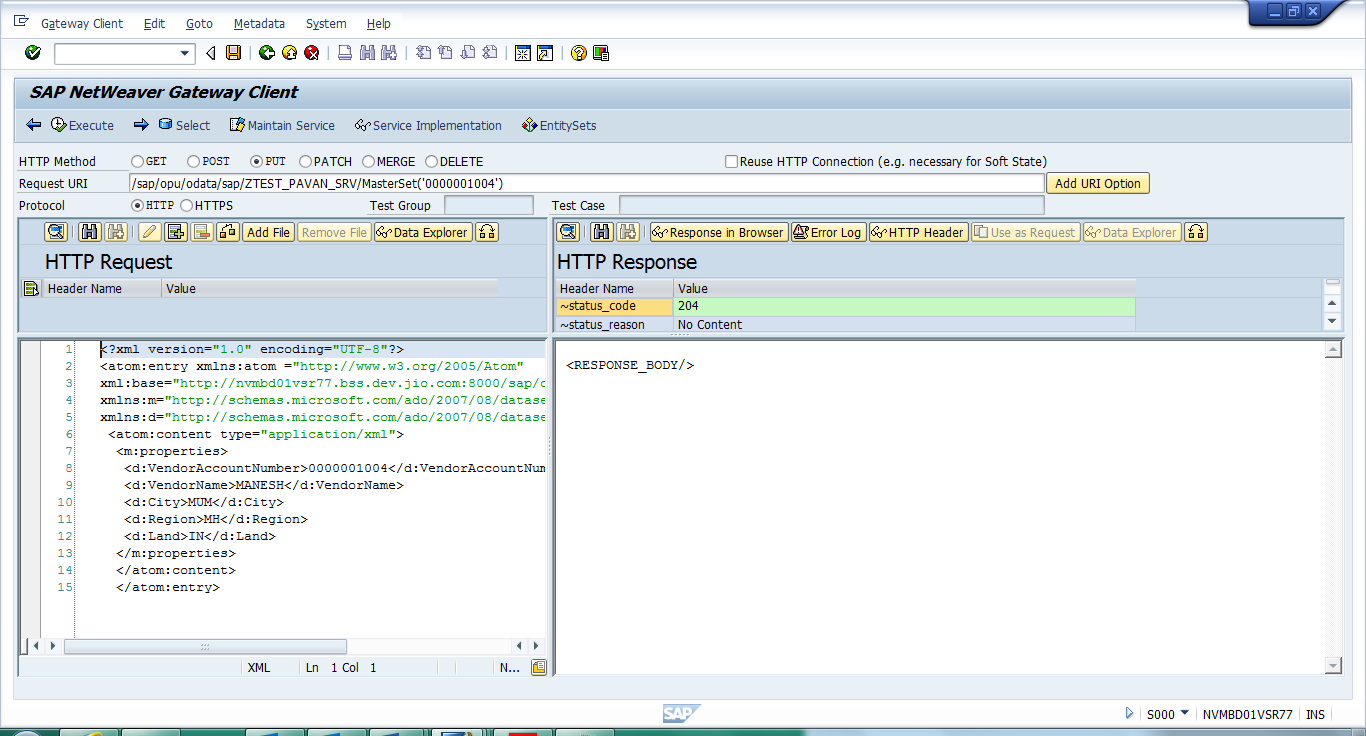
-> Write the Payload in HTTP Request

**URL:**

/sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/MasterSet(VendorAccountNumber='0000001004')

**Payload:**

<?xml version="1.0" encoding="UTF-8"?>  
<atom:entry xmlns:atom ="http://www.w3.org/2005/Atom"  
xml:base="http://nvmbd01vsr77.bss.dev.jio.com:8000/sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/"  
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"  
xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices">  
 <atom:content type="application/xml">  
  <m:properties>  
   <d:VendorAccountNumber>0000001004</d:VendorAccountNumber>  
   <d:VendorName>MANESH</d:VendorName>  
   <d:City>MUM</d:City>  
   <d:State>MP</d:State>  
   <d:Country>IN</d:Country>  
  </m:properties>  
  </atom:content>  
  </atom:entry>



**DELETE\_ENTITY Method:-**This is the method for the Update operation for an entity / entry.

**Parameters Used in DELETE\_ENTITY:**

**IT\_KEY\_TAB:** This is an Input Parameter Which Contains Two Fields Name and Value. it is Used to Represents the keys of the First Segment.

Type /IWBEP/T\_MGW\_NAME\_VALUE\_PAIR

  method MASTERSET\_DELETE\_ENTITY.  
    DATA: ls\_master    TYPE zcl\_ztest\_pavan\_mpc=>ts\_master,  
          ls\_lfa1      TYPE zlfa1\_master,  
          ls\_return    TYPE bapiret2,  
          lt\_return    TYPE bapiret2\_t,  
          lv\_lifnr     TYPE lifnr,  
          ls\_key\_tab   TYPE /iwbep/s\_mgw\_name\_value\_pair,  
          lo\_container TYPE REF TO /iwbep/if\_message\_container,  
          lw\_lheader   TYPE ihttpnvp,  
          json         TYPE string,  
          lo\_message   TYPE REF TO zcl\_fiori\_format\_message\_json.  
  
  
    READ TABLE it\_key\_tab INTO ls\_key\_tab WITH KEY  
                               name = 'VendorAccountNumber'.  
    IF sy-subrc = 0.  
      lv\_lifnr = ls\_key\_tab-value.  
    ENDIF.  
  
       DELETE FROM zlfa1\_master WHERE lifnr = lv\_lifnr.  
      IF sy-subrc EQ 0.  
        COMMIT WORK.  
        ls\_return-type = 'S'.  
        ls\_return-message = 'Data Deleted Successfully'.  
        APPEND ls\_return TO lt\_return.  
  
        CREATE OBJECT lo\_message.  
        CALL METHOD lo\_message->message\_format  
          EXPORTING  
            im\_returnmsg = lt\_return  
          IMPORTING  
            ex\_json      = json.  
  
        lw\_lheader-name = 'sap-message'.  
        lw\_lheader-value = json.  
        set\_header( is\_header = lw\_lheader ).  
  
      ELSE.  
        ls\_return-type = 'E'.  
        ls\_return-message = 'Error in deleting data '.  
  
        lo\_container = me->mo\_context->get\_message\_container( ).  
  
        CALL METHOD lo\_container->add\_message  
          EXPORTING  
            iv\_msg\_type   = ls\_return-type  
            iv\_msg\_id     = ls\_return-id  
            iv\_msg\_number = ls\_return-number  
            iv\_msg\_text   = ls\_return-message.  
  
        RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
          EXPORTING  
            message\_container = lo\_container.  
      ENDIF.  
  
  endmethod.

Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

-> Select our Service

-> Click on Load Metadata

-> Click on Gateway Client

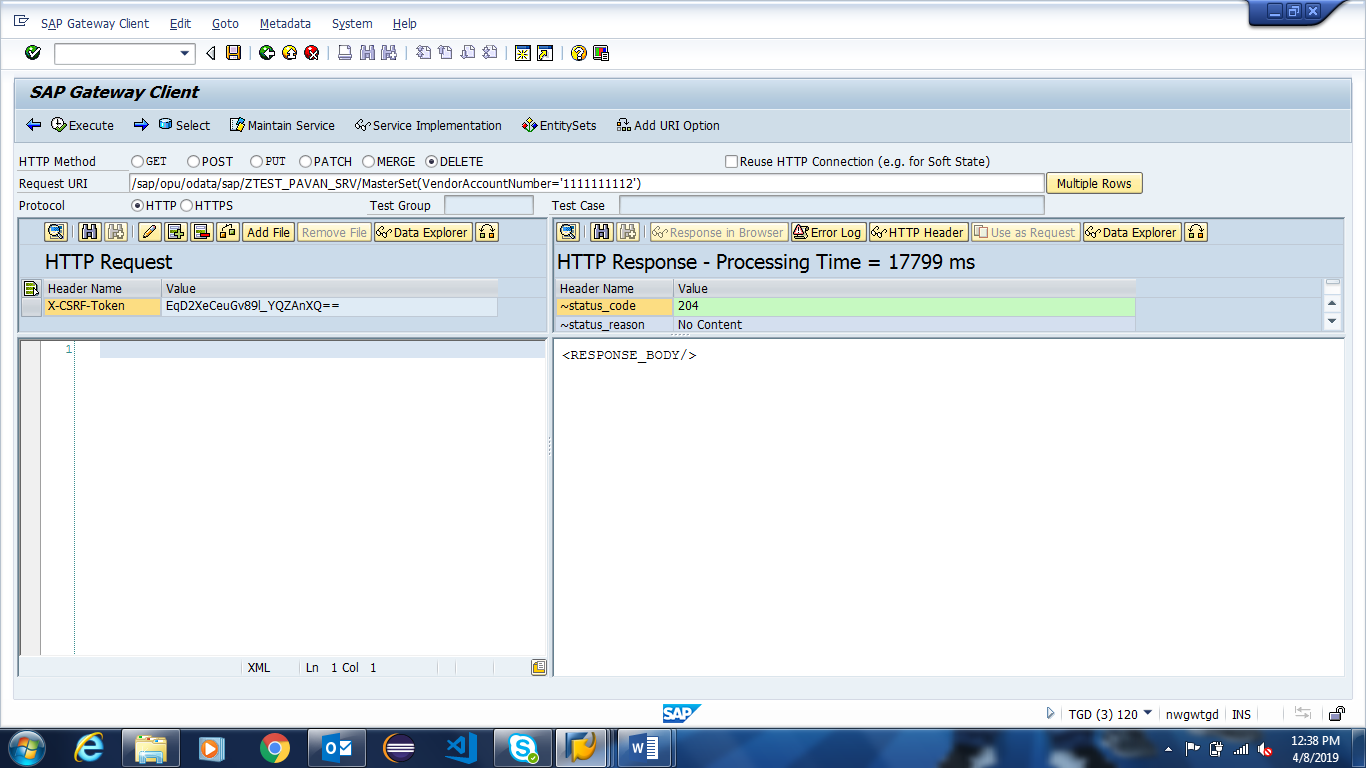
-> Click on EntitySets and Select Entityset Name

-> Select the HTTP Method ‘Delete’

-> Write the Payload in HTTP Request

**URL:**

/sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/MasterSet(VendorAccountNumber='0000001004')

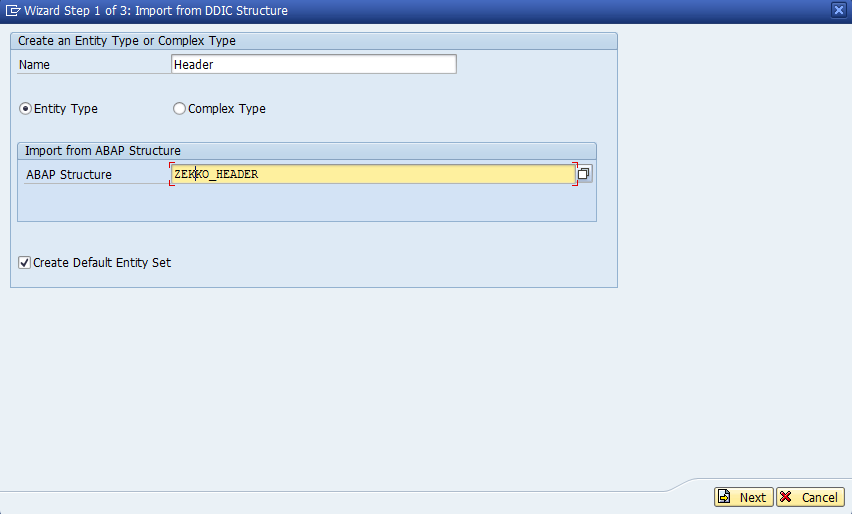


**Header Entity Type:-**

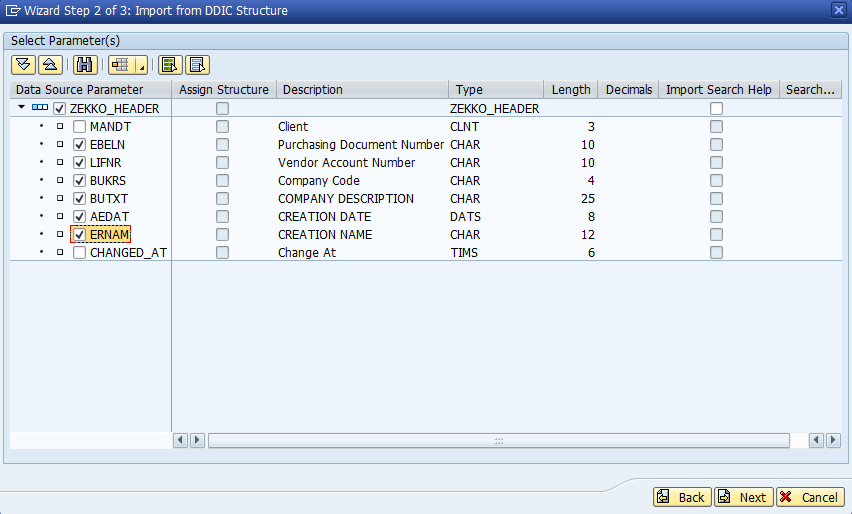
Expand The Project ‘ZTEST\_PAVAN’.

Select The Data Model and Right Click on it -> IMPORT -> DDIC Structure.

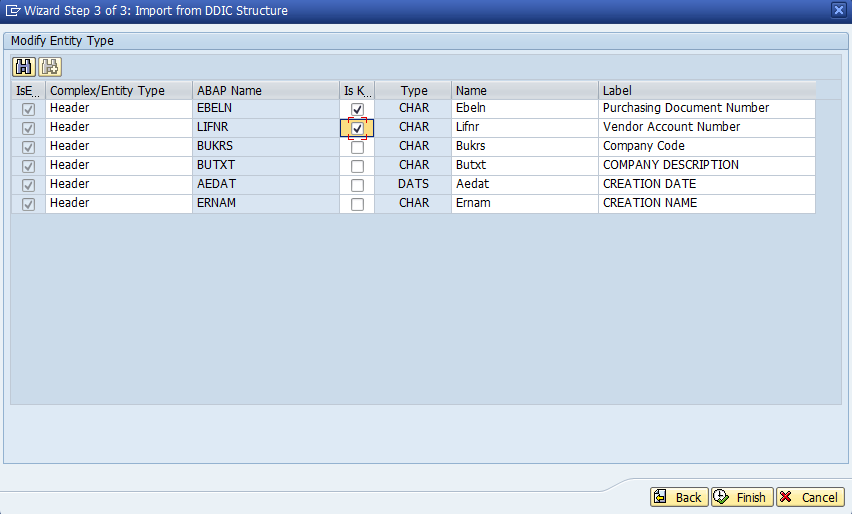
Provide Entity Name: ‘Header’, ABAP Structure Name: ‘ZEKKO\_HEADER’ Click on Next.



Select our Required Fields and Click on Next.



Select the check Box Is Key and Click on Finish.

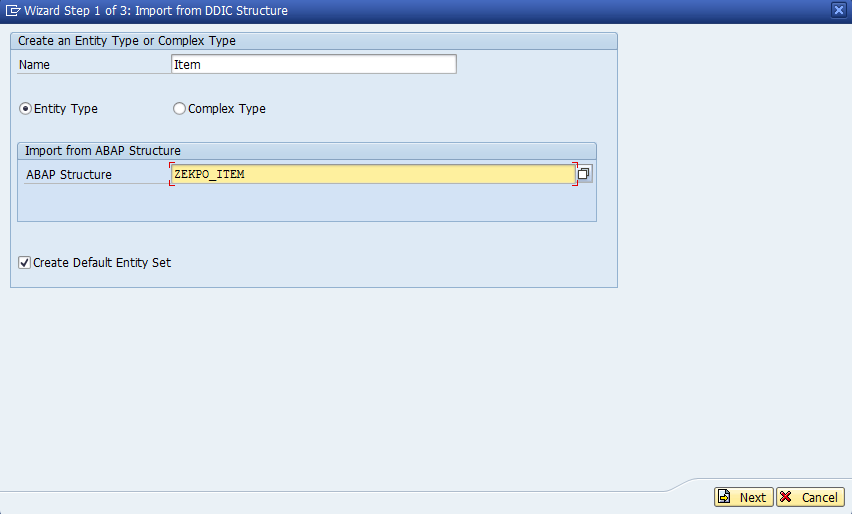


**Item Entity Type:-**

Expand The Project ‘ZTEST\_PAVAN’.

Select The Data Model and Right Click on it -> IMPORT -> DDIC Structure.

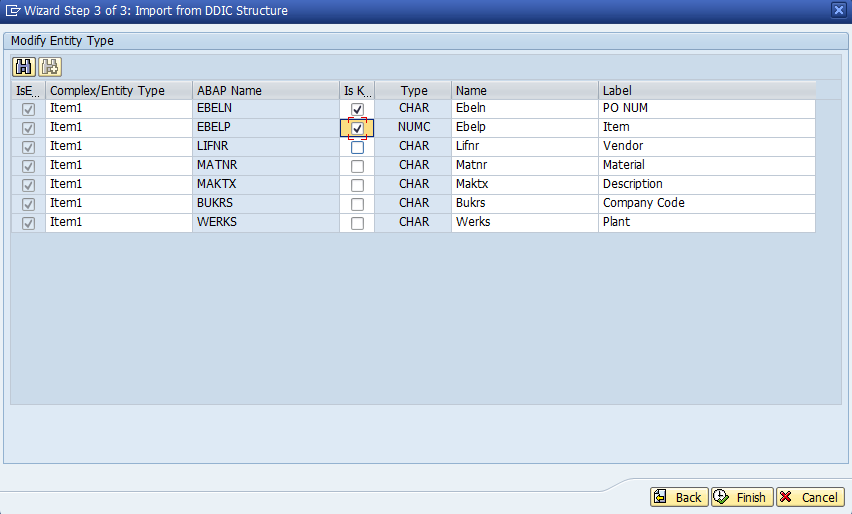
Provide Entity Name: ‘Item’, ABAP Structure Name: ‘ZEKPO\_ITEM’ Click on Next.



Select our Required Fields and Click on Next.



Select the check Box Is Key and Click on Finish.



**Creating ASSOCIATIONS:-**

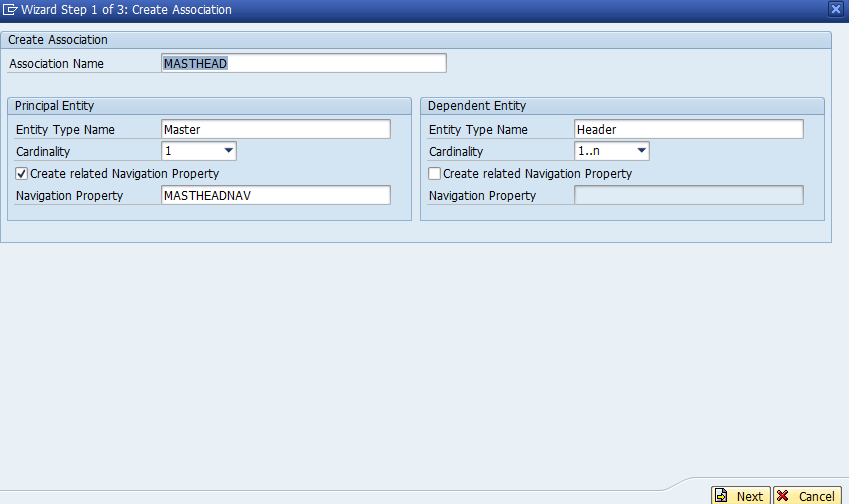
**MASTHEAD Association:**

Expand The Project ‘ZTEST\_PAVAN’.

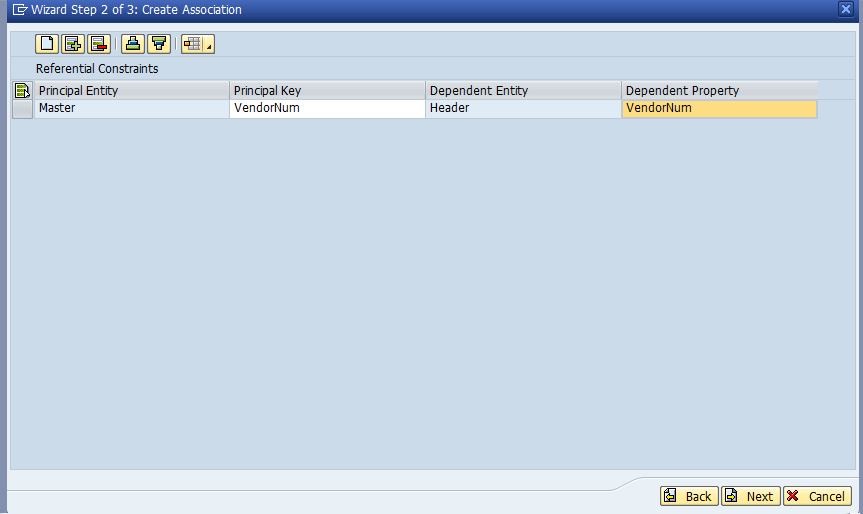
Expand The Project ‘Data Model’.

Select The Associations and Right Click on it -> Create.

Provide Association Name: ‘MASTHEAD’, Principal Entity Name: ‘Master’, Dependent Entity Name: ‘Header’, Navigation Name: ‘MASTHEADNAV’ and Click on Next.



Give The Dependent Property ‘VendorNum’.



Click on Next.



Click on Finish.

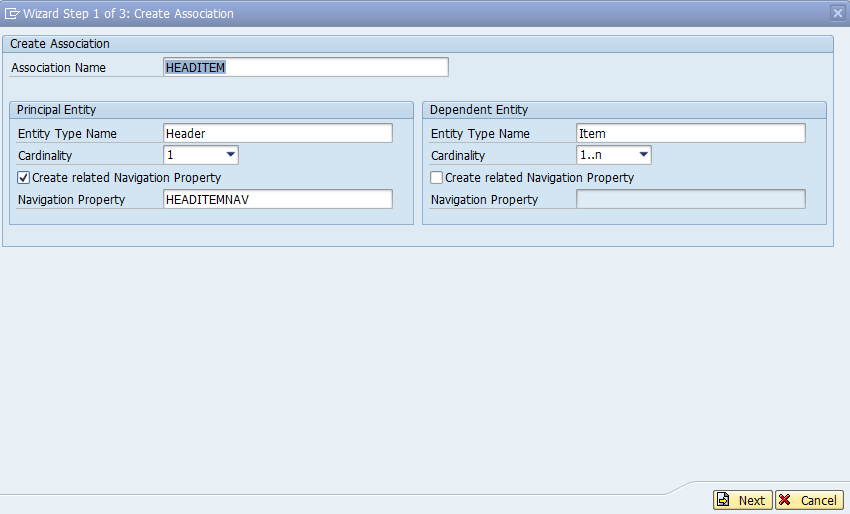
**HEADITEM Association:**

Expand The Project ‘ZTEST\_PAVAN’.

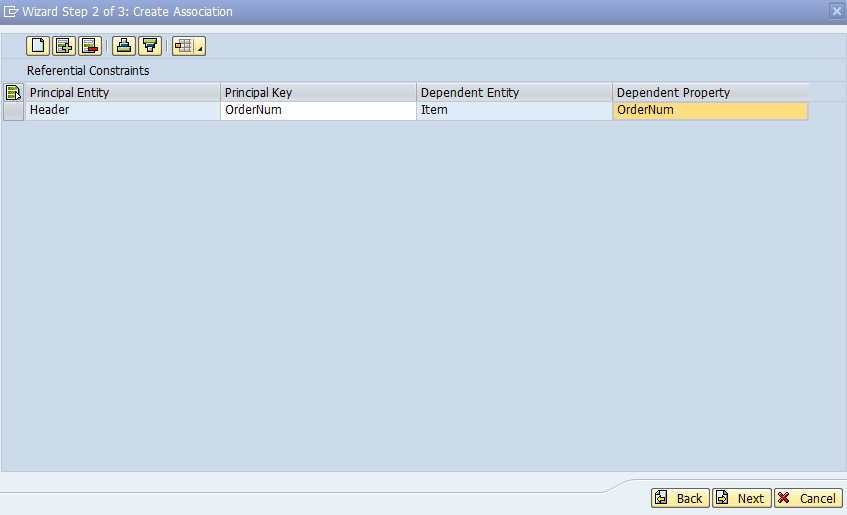
Expand The Project ‘Data Model’.

Select The Associations and Right Click on it -> Create.

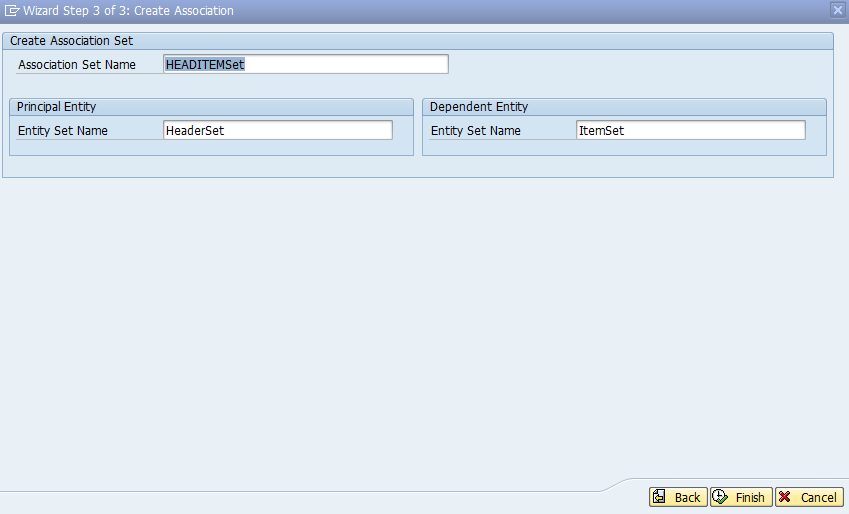
Provide Association Name: ‘HEADITEM’, Principal Entity Name: ‘Header’, Dependent Entity Name: ‘Item’, Navigation Name: ‘HEADITEMNAV’ and Click on Next.



Give The Dependent Property ‘OrderNum’.



Click on Next.



Click on Finish.

Note:- Entity Sets and Association Sets are Automatically Created at the Time of When We Creating Entity Types and Associations.

**CREATE\_DEEP\_ENTITY Method:**

**Parameters Used in CREATE\_DEEP\_ENTITY:**

**ER\_DEEP\_ENTITY:** The returned reference to the application specific structure which contains the data.

**IO\_DATA\_PROVIDER:** The interface to deserialize the payload into the application specific structure. Type /IWBEP/IF\_MGW\_ENTRY\_PROVIDER

**CREATE\_DEEP\_ENTITY Code:**

    METHOD /iwbep/if\_mgw\_appl\_srv\_runtime~create\_deep\_entity.  
  
    TYPES: BEGIN OF ty\_heditm.  
            INCLUDE TYPE zcl\_ztest\_pavan\_mpc=>ts\_header.  
    TYPES: headitemnav TYPE STANDARD TABLE OF zcl\_ztest\_pavan\_mpc=>ts\_item WITH NON-UNIQUE DEFAULT KEY,  
           END OF ty\_heditm.  
  
  
    TYPES: BEGIN OF ty\_mashed.  
            INCLUDE TYPE zcl\_ztest\_pavan\_mpc=>ts\_master.  
    TYPES: mastheadnav TYPE STANDARD TABLE OF ty\_heditm WITH NON-UNIQUE DEFAULT KEY,  
           END OF ty\_mashed.  
  
    DATA: wa\_mashed TYPE ty\_mashed,  
          wa\_lfa1   TYPE zlfa1\_master,  
          wa\_ekko   TYPE zekko\_header,  
          wa\_ekpo   TYPE zekpo\_item,  
          it\_ekko   TYPE TABLE OF zekko\_header,  
          it\_ekpo   TYPE TABLE OF zekpo\_item,  
          wa\_head   TYPE ty\_heditm,  
          wa\_item   TYPE zcl\_ztest\_pavan\_mpc=>ts\_item.  
  
      DATA   :   lv\_flag    TYPE abap\_bool VALUE space,  
             lw\_lheader TYPE ihttpnvp,  
             json       TYPE string,  
             lo\_message TYPE REF TO zcl\_fiori\_format\_message\_json.  
  
  CONSTANTS: lc\_e TYPE char01 VALUE 'E',  
             lc\_w TYPE char01 VALUE 'W',  
             lc\_i TYPE char01 VALUE 'I',  
             lc\_s TYPE char01 VALUE 'S'.  
  
  DATA    :  lt\_return    TYPE STANDARD TABLE OF bapiret2,  
             lw\_return    TYPE bapiret2,  
             lo\_container TYPE REF TO /iwbep/if\_message\_container.  
  
    io\_data\_provider->read\_entry\_data( IMPORTING es\_data = wa\_mashed ).  
  
    IF  wa\_mashed IS NOT INITIAL.  
      wa\_lfa1-lifnr = wa\_mashed-lifnr.  
      wa\_lfa1-name1 = wa\_mashed-name1.  
      wa\_lfa1-ort01 = wa\_mashed-ort01.  
      wa\_lfa1-regio = wa\_mashed-regio.  
      wa\_lfa1-land1 = wa\_mashed-land1.  
  
      LOOP AT wa\_mashed-mastheadnav INTO wa\_head.  
        wa\_ekko-ebeln = wa\_head-ebeln.  
        wa\_ekko-lifnr = wa\_head-lifnr.  
        wa\_ekko-bukrs = wa\_head-bukrs.  
        wa\_ekko-butxt = wa\_head-butxt.  
        wa\_ekko-aedat = wa\_head-aedat.  
        wa\_ekko-ernam = wa\_head-ernam.  
  
        LOOP AT wa\_head-headitemnav INTO wa\_item.  
          wa\_ekpo-ebeln = wa\_item-ebeln.  
          wa\_ekpo-ebelp = wa\_item-ebelp.  
          wa\_ekpo-matnr = wa\_item-matnr.  
          wa\_ekpo-maktx = wa\_item-maktx.  
          wa\_ekpo-bukrs = wa\_head-bukrs.  
          wa\_ekpo-werks = wa\_item-werks.  
          APPEND wa\_ekpo TO it\_ekpo.  
          CLEAR: wa\_ekpo, wa\_item.  
        ENDLOOP.  
  
        APPEND wa\_ekko TO it\_ekko.  
        CLEAR: wa\_ekko, wa\_head.  
      ENDLOOP.  
    ENDIF.  
  
    INSERT zlfa1\_master FROM wa\_lfa1.  
    IF sy-subrc EQ 0.  
  
      COMMIT WORK.  
      lw\_return-type = 'S'.  
      lw\_return-message = 'Data Updated Successfully in Master Table'.  
      APPEND lw\_return TO lt\_return.  
      CLEAR lw\_return.  
  
    ELSE.  
  
      lw\_return-type = 'E'.  
      lw\_return-message = 'Error updating data in Master Table'.  
      APPEND lw\_return TO lt\_return.  
      CLEAR lw\_return.  
  
    ENDIF.  
    INSERT zekko\_header FROM TABLE it\_ekko.  
    IF sy-subrc EQ 0.  
  
      COMMIT WORK.  
      lW\_return-type = 'S'.  
      lw\_return-message = 'Data Updated Successfully in Header Table'.  
      APPEND lw\_return TO lt\_return.  
      CLEAR lw\_return.  
  
    ELSE.  
  
      lw\_return-type = 'E'.  
      lw\_return-message = 'Error updating data in Header Table'.  
      APPEND lw\_return TO lt\_return.  
      CLEAR lw\_return.  
  
    ENDIF.  
    INSERT zekpo\_item FROM TABLE it\_ekpo.  
    IF sy-subrc EQ 0.  
  
      COMMIT WORK.  
      lw\_return-type = 'S'.  
      lw\_return-message = 'Data Updated Successfully in Item Table'.  
      APPEND lw\_return TO lt\_return.  
      CLEAR lw\_return.  
  
    ELSE.  
  
      lw\_return-type = 'E'.  
      lw\_return-message = 'Error updating data in Item Table'.  
      APPEND lw\_return TO lt\_return.  
      CLEAR lw\_return.  
  
    ENDIF.  
  
     IF lt\_return IS NOT INITIAL.  
      CLEAR: lw\_return, lv\_flag.  
      LOOP AT lt\_return INTO lw\_return WHERE type = lc\_s  
                                          OR type = lc\_i  
                                          OR type = lc\_w.  
        lv\_flag = abap\_true.  
        EXIT.  
      ENDLOOP.  
  
      IF lv\_flag IS INITIAL.  
        lo\_container = me->mo\_context->get\_message\_container( ).  
        CLEAR lw\_return.  
        LOOP AT lt\_return INTO lw\_return.  
          CALL METHOD lo\_container->add\_message  
            EXPORTING  
              iv\_msg\_type   = lw\_return-type  
              iv\_msg\_id     = lw\_return-id  
              iv\_msg\_number = lw\_return-number  
              iv\_msg\_text   = lw\_return-message  
              iv\_msg\_v1     = lw\_return-message\_v1  
              iv\_msg\_v2     = lw\_return-message\_v2  
              iv\_msg\_v3     = lw\_return-message\_v3  
              iv\_msg\_v4     = lw\_return-message\_v4.  
        ENDLOOP.  
  
        RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
          EXPORTING  
            message\_container = lo\_container.  
      ELSE.  
        CREATE OBJECT lo\_message.  
        CALL METHOD lo\_message->message\_format  
          EXPORTING  
            im\_returnmsg = lt\_return  
          IMPORTING  
            ex\_json      = json.  
  
        lw\_lheader-name = 'sap-message'.  
        lw\_lheader-value = json.  
        set\_header( is\_header = lw\_lheader ).  
  
        copy\_data\_to\_ref( EXPORTING is\_data = wa\_mashed  
                      CHANGING cr\_data = er\_deep\_entity ).  
      ENDIF.  
    ENDIF.  
  
  ENDMETHOD.

Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

-> Select our Service

-> Click on Load Metadata

-> Click on Gateway Client

-> Click on EntitySets and Select Entityset Name

-> Select the HTTP Method ‘Post’

-> Write the Payload in HTTP Request

**URL:** /sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/MasterSet

**Payload:**

<?xml version="1.0" encoding="UTF-8"?>

<atom:entry xmlns:atom ="http://www.w3.org/2005/Atom"

xml:base="http://nvmbd01vsr77.bss.dev.jio.com:8000/sap/opu/odata/sap/ap/ZTEST\_PAVAN\_SRV/"

xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"

xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices">

<atom:content type="application/xml">

<m:properties>

<d:VendorNum>0000001005</d:VendorNum>

<d:VendorName>MG Group</d:VendorName>

<d:City>Chennai</d:City>

<d:State>TN</d:State>

<d:Country>IN</d:Country>

</m:properties>

</atom:content>

<atom:link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/MASTHEADNAV"

type="application/atom+xml;type=feed"

title="ZTEST\_PAVAN\_SRV.HeaderSet">

<m:inline>

<atom:feed>

<atom:entry>

<atom:content type="application/xml">

<m:properties>

<d:OrderNum>0000000001</d:OrderNum>

<d:VendorNum>0000001005</d:VendorNum>

<d:CompanyCode>RJIL</d:CompanyCode>

<d:CompanyDescr>Reliance JIO</d:CompanyDescr>

<d:CreationDate>2019-04-10T00:00:00</d:CreationDate>

<d:CreatedBy>Manesh</d:CreatedBy>

</m:properties>

</atom:content>

<atom:link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/HEADITEMNAV"

type="application/atom+xml;type=feed"

title="ZTEST\_PAVAN\_SRV.ItemSet">

<m:inline>

<atom:feed>

<atom:entry>

<atom:content type="application/xml">

<m:properties>

<d:OrderNum>0000000001</d:OrderNum>

<d:ItemNum>00001</d:ItemNum>

<d:ArticleNum>0000123456</d:ArticleNum>

<d:ArticleDesc>SIM</d:ArticleDesc>

<d:CompanyCode>RJIL</d:CompanyCode>

<d:Plant></d:Plant>

</m:properties>

</atom:content>

</atom:entry>

</atom:feed>

</m:inline>

</atom:link>

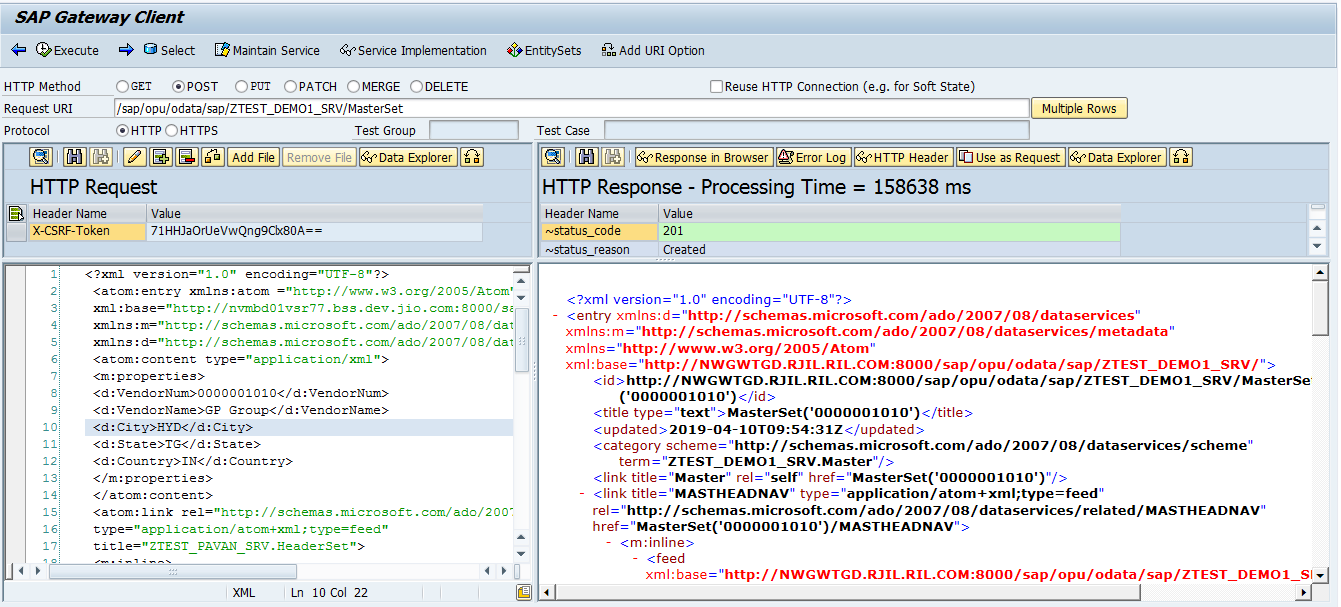
</atom:entry>

</atom:feed>

</m:inline>

</atom:link>

</atom:entry>



**GET\_EXPANDED\_ENTITY Method:**

This method can be used for a read operation for an expanded entity / entry. The read operation is similar to GET\_ENTITY which has an additional expand reference to retrieve data in an in lined/deep format.

The expand query option is passed via the object reference IO\_EXPAND. It can be used to validate whether the current expand expression matches a given expand expression. If the expand expression matches the data reference, then ER\_ENTITY can directly be filled by the implementation with data in a deep format. All navigation properties of an entity type including. Its sub paths are valid expand expressions to check and to handle by the application. All expand paths (separated by comma) which are handled by the implementation have to be passed back to the framework via the parameter ET\_EXPANDED\_TECH\_CLAUSES.

**Parameters Used in GET\_EXPANDED\_ENTITY:**

**IV\_ENTITY\_NAME:** Name of the entity type which is requested. If there is a navigation it means that it represents the end/target of the navigation path. Type string

**IT\_KEY\_TAB:** This is an Input Parameter Which Contains Two Fields Name and Value. it is Used to Represents the keys of the First Segment.

Type /IWBEP/T\_MGW\_NAME\_VALUE\_PAIR

**ET\_EXPANDED\_TECH\_CLAUSES:** This parameter is filled by the application and contains all expanded paths which have been handled by the application itself. This list of expand clauses is based on the technical (internal) navigation property names, not the external names

**GET\_EXPANDED\_ENTITY Code:**

  METHOD /iwbep/if\_mgw\_appl\_srv\_runtime~get\_expanded\_entity.  
    TYPES: BEGIN OF ty\_heditm.  
            INCLUDE TYPE zcl\_ztest\_pavan\_mpc=>ts\_header.  
    TYPES: headitemnav TYPE STANDARD TABLE OF zcl\_ztest\_pavan\_mpc=>ts\_item  
                                                WITH DEFAULT KEY,  
           END OF ty\_heditm.  
  
    TYPES: BEGIN OF ty\_mashed.  
            INCLUDE TYPE zcl\_ztest\_pavan\_mpc=>ts\_master.  
    TYPES: mastheadnav TYPE STANDARD TABLE OF ty\_heditm WITH DEFAULT KEY,  
           END OF ty\_mashed.  
  
    CONSTANTS: lc\_expanded\_tech\_clauses TYPE string  
                                       VALUE 'MASTHEADNAV/HEADITEMNAV'.  
  
    DATA: wa\_mashed    TYPE ty\_mashed,  
          wa\_heditm    TYPE ty\_heditm,  
          wa\_itm       TYPE zcl\_ztest\_pavan\_mpc=>ts\_item,  
          wa\_lfa1      TYPE zlfa1\_master,  
          wa\_ekko      TYPE zekko\_header,  
          it\_ekko      TYPE STANDARD TABLE OF zekko\_header,  
          wa\_ekpo      TYPE zekpo\_item,  
          it\_ekpo      TYPE STANDARD TABLE OF zekpo\_item,  
          wa\_key\_tab   LIKE LINE OF it\_key\_tab,  
          lv\_lifnr     TYPE zlfa1\_master-lifnr,  
          lo\_container TYPE REF TO /iwbep/if\_message\_container,  
          ls\_return    TYPE bapiret2.  
  
    READ TABLE it\_key\_tab INTO wa\_key\_tab WITH KEY  
                                          name = 'VendorNum'.  
    IF sy-subrc = 0.  
      lv\_lifnr = wa\_key\_tab-value.  
      UNPACK lv\_lifnr TO lv\_lifnr.  
    ENDIF.  
  
    SELECT SINGLE \* FROM zlfa1\_master INTO wa\_lfa1 WHERE lifnr = lv\_lifnr.  
    IF sy-subrc = 0 .  
      SELECT \* FROM zekko\_header INTO TABLE it\_ekko WHERE lifnr = lv\_lifnr.  
      IF it\_ekko[] IS NOT INITIAL.  
        SELECT \* FROM zekpo\_item INTO TABLE it\_ekpo FOR ALL ENTRIES  
           IN it\_ekko WHERE ebeln = it\_ekko-ebeln AND lifnr = it\_ekko-lifnr.  
      ELSE.  
        ls\_return-type = 'E'.  
        ls\_return-message = 'No Data Found'.  
  
        lo\_container = me->mo\_context->get\_message\_container( ).  
  
        CALL METHOD lo\_container->add\_message  
          EXPORTING  
            iv\_msg\_type   = ls\_return-type  
            iv\_msg\_id     = ls\_return-id  
            iv\_msg\_number = ls\_return-number  
            iv\_msg\_text   = ls\_return-message.  
  
        RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
          EXPORTING  
            message\_container = lo\_container.  
        RETURN.  
      ENDIF.  
  
    ELSE.  
      ls\_return-type = 'E'.  
      ls\_return-message = 'No Data Found'.  
  
      lo\_container = me->mo\_context->get\_message\_container( ).  
  
      CALL METHOD lo\_container->add\_message  
        EXPORTING  
          iv\_msg\_type   = ls\_return-type  
          iv\_msg\_id     = ls\_return-id  
          iv\_msg\_number = ls\_return-number  
          iv\_msg\_text   = ls\_return-message.  
  
      RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
        EXPORTING  
          message\_container = lo\_container.  
      RETURN.  
    ENDIF.  
  
    wa\_mashed-lifnr = wa\_lfa1-lifnr.  
    wa\_mashed-name1 = wa\_lfa1-name1.  
    wa\_mashed-ort01 = wa\_lfa1-ort01.  
    wa\_mashed-regio = wa\_lfa1-regio.  
    wa\_mashed-land1 = wa\_lfa1-land1.  
  
    LOOP AT it\_ekko INTO wa\_ekko WHERE lifnr = wa\_lfa1-lifnr.  
      wa\_heditm-ebeln = wa\_ekko-ebeln.  
      wa\_heditm-lifnr = wa\_ekko-lifnr.  
      wa\_heditm-bukrs = wa\_ekko-bukrs.  
      wa\_heditm-butxt = wa\_ekko-butxt.  
      wa\_heditm-aedat = wa\_ekko-aedat.  
      wa\_heditm-ernam = wa\_ekko-ernam.  
  
      LOOP AT it\_ekpo INTO wa\_ekpo WHERE ebeln = wa\_ekko-ebeln.  
        wa\_itm-ebeln = wa\_ekpo-ebeln.  
        wa\_itm-ebelp = wa\_ekpo-ebelp.  
        wa\_itm-matnr = wa\_ekpo-matnr.  
        wa\_itm-maktx = wa\_ekpo-maktx.  
        wa\_itm-bukrs = wa\_ekpo-bukrs.  
        wa\_itm-werks = wa\_ekpo-werks.  
        APPEND wa\_itm TO wa\_heditm-headitemnav.  
        CLEAR: wa\_itm, wa\_ekpo.  
      ENDLOOP.  
  
      APPEND wa\_heditm TO wa\_mashed-mastheadnav.  
      CLEAR: wa\_heditm, wa\_ekko.  
    ENDLOOP.  
  
    copy\_data\_to\_ref( EXPORTING is\_data = wa\_mashed  
                      CHANGING cr\_data = er\_entity ).  
  
    INSERT lc\_expanded\_tech\_clauses INTO TABLE et\_expanded\_tech\_clauses.  
  
  ENDMETHOD.

Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

-> Select our Service

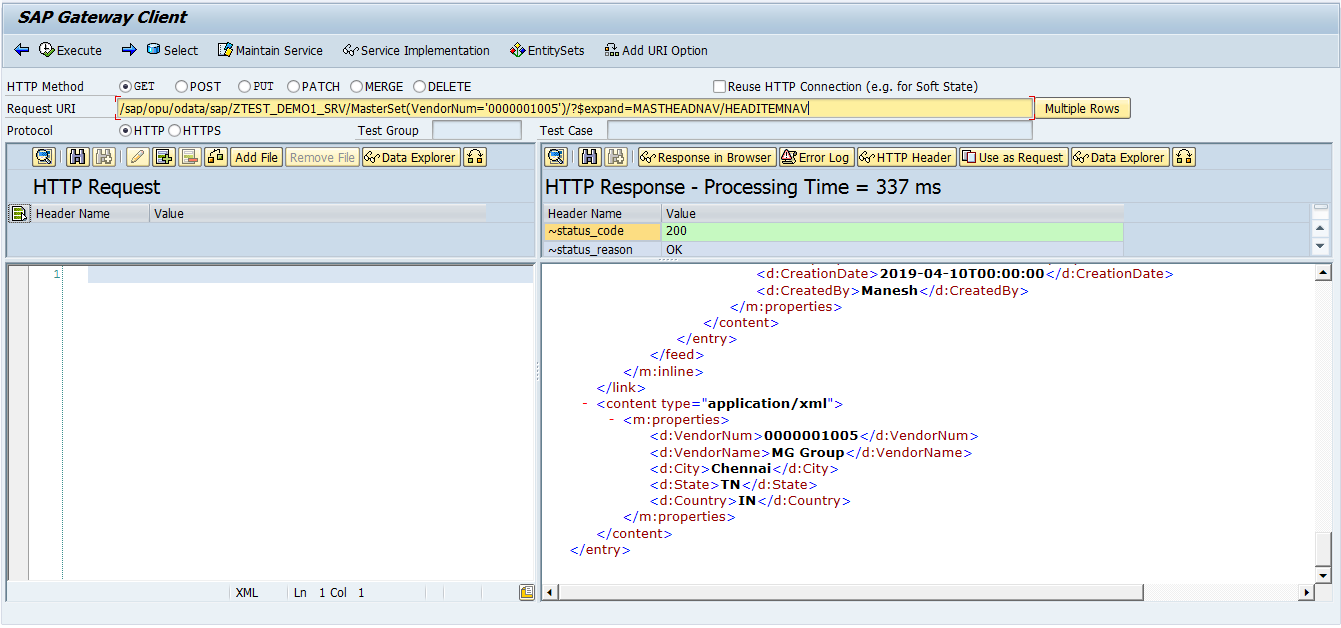
-> Click on Load Metadata

-> Click on Gateway Client

-> Click on EntitySets and Select Entityset Name

-> Select the HTTP Method ‘Get’

**URL:**/sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/MasterSet('0000001005')/?$expand= MASTHEADNAV/HEADITEMNAV



**GET\_EXPANDED\_ENTITYSET Method :**

The (query) operation similar to GET\_ENTITYSET which has an additional expand reference to retrieve data in an inlined/deep format. Note that the SAP NetWeaver Gateway framework can completely handle expand requests in a generic way. In case of performance-critical scenarios or if you experience a bad response time it is highly recommended to handle complex full/subsets of the expand request by specific implementations. Therefore it is required to redefine GET\_EXPANDED\_ENTITYSET. If you redefine GET\_EXPANDED\_ENTITYSET and the given expand expression cannot be handled by the application logic it is required to delegate to the super method or to the implementation of the super class.

The expand query option is passed via the object reference IO\_EXPAND. It can be used to validate whether the current expand expression matches a given expand expression. If the expand expression matches the data reference, then ER\_ENTITY can directly be filled by the implementation with data in a deep format. All navigation properties of an entity type including. its sub paths are valid expand expressions to check and to handle by the application. All expand paths (separated by comma) which are handled by the implementation have to be passed back to the framework via the parameter ET\_EXPANDED\_TECH\_CLAUSES.

**Parameters Used in GET\_EXPANDED\_ENTITYSET:**

**ER\_ENTITYSET:** The returned reference to the application specific table which contains the data. In difference to the GET\_ENTITYSET method the ER\_ENTITYSET reference is already pre-filled with a data reference of a deep structure component of the final expanded result feed. The ER\_ENTITYSET can then be filled by the application specific implementation.

**IO\_TECH\_REQUEST\_CONTEXT:** This parameter of type /IWBEP/IF\_MGW\_REQ\_ENTITY contains all request information and represents the requested entities in a technical format meaning with their technical naming.

**IO\_EXPAND:** Represents the expand expression for the navigation properties of the nested entry/feed which need to be inlined.

**ET\_EXPANDED\_CLAUSES:** This parameter is obsolete, use ET\_EXPANDED\_TECH\_CLAUSES instead.

**ET\_EXPANDED\_TECH\_CLAUSES:** This parameter is filled by the application and contains all expanded paths which have been handled by the application itself. This list of expand clauses is based on the technical (internal) navigation property names, not the external names

**GET\_EXPANDED\_ENTITYSET Code:**

  method /IWBEP/IF\_MGW\_APPL\_SRV\_RUNTIME~GET\_EXPANDED\_ENTITYSET.  
TYPES: BEGIN OF ty\_heditm.  
            INCLUDE TYPE zcl\_ztest\_pavan\_mpc=>ts\_header.  
    TYPES: headitemnav TYPE STANDARD TABLE OF zcl\_ztest\_pavan\_mpc=>ts\_item                                               WITH DEFAULT KEY,  
           END OF ty\_heditm.  
  
  
    TYPES: BEGIN OF ty\_mashed.  
            INCLUDE TYPE zcl\_ztest\_pavan\_mpc=>ts\_master.  
    TYPES: mastheadnav TYPE STANDARD TABLE OF ty\_heditm WITH DEFAULT KEY,  
           END OF ty\_mashed.  
  
    CONSTANTS: lc\_expanded\_tech\_clauses TYPE string VALUE 'MASTHEADNAV/HEADITEMNAV'.  
  
    DATA: wa\_mashed       TYPE ty\_mashed,  
          it\_mashed       TYPE TABLE OF ty\_mashed,  
          wa\_heditm       TYPE ty\_heditm,  
          wa\_itm          TYPE zcl\_ztest\_pavan\_mpc=>ts\_item,  
          wa\_lfa1         TYPE zlfa1\_master,  
          it\_lfa1         TYPE STANDARD TABLE OF zlfa1\_master,  
          wa\_ekko         TYPE zekko\_header,  
          it\_ekko         TYPE STANDARD TABLE OF zekko\_header,  
          wa\_ekpo         TYPE zekpo\_item,  
          it\_ekpo         TYPE STANDARD TABLE OF zekpo\_item,  
          ls\_filter       TYPE /iwbep/s\_mgw\_select\_option,  
          ls\_filter\_range TYPE /iwbep/s\_cod\_select\_option,  
          lr\_lifnr        TYPE RANGE OF lifnr,  
          ls\_lifnr        LIKE LINE OF lr\_lifnr,  
          lo\_container    TYPE REF TO /iwbep/if\_message\_container,  
          ls\_return       TYPE bapiret2..  
  
    LOOP AT it\_filter\_select\_options INTO ls\_filter.  
  
      CASE ls\_filter-property.  
        WHEN 'VendorNum'.  
          LOOP AT ls\_filter-select\_options INTO ls\_filter\_range.  
            ls\_lifnr-low = ls\_filter\_range-low.  
            ls\_lifnr-option = 'EQ'.  
            ls\_lifnr-sign = 'I'.  
            APPEND ls\_lifnr TO lr\_lifnr.  
            CLEAR: ls\_lifnr.  
          ENDLOOP.  
      ENDCASE.  
    ENDLOOP.  
  
  
    SELECT \* FROM zlfa1\_master INTO TABLE it\_lfa1 WHERE lifnr IN lr\_lifnr.  
    IF sy-subrc = 0 .  
      SELECT \* FROM zekko\_header INTO TABLE it\_ekko FOR ALL ENTRIES IN it\_lfa1 WHERE lifnr = it\_lfa1-lifnr.  
      IF it\_ekko[] IS NOT INITIAL.  
        SELECT \* FROM zekpo\_item INTO TABLE it\_ekpo FOR ALL ENTRIES IN it\_ekko WHERE ebeln = it\_ekko-ebeln.  
      ELSE.  
        ls\_return-type = 'E'.  
        ls\_return-message = 'No Data Found'.  
  
        lo\_container = me->mo\_context->get\_message\_container( ).  
  
        CALL METHOD lo\_container->add\_message  
          EXPORTING  
            iv\_msg\_type   = ls\_return-type  
            iv\_msg\_id     = ls\_return-id  
            iv\_msg\_number = ls\_return-number  
            iv\_msg\_text   = ls\_return-message.  
  
        RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
          EXPORTING  
            message\_container = lo\_container.  
        RETURN.  
      ENDIF.  
    ELSE.  
      ls\_return-type = 'E'.  
      ls\_return-message = 'No Data Found'.  
  
      lo\_container = me->mo\_context->get\_message\_container( ).  
  
      CALL METHOD lo\_container->add\_message  
        EXPORTING  
          iv\_msg\_type   = ls\_return-type  
          iv\_msg\_id     = ls\_return-id  
          iv\_msg\_number = ls\_return-number  
          iv\_msg\_text   = ls\_return-message.  
  
      RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
        EXPORTING  
          message\_container = lo\_container.  
      RETURN.  
    ENDIF.  
    LOOP AT it\_lfa1 INTO wa\_lfa1.  
      wa\_mashed-lifnr = wa\_lfa1-lifnr.  
      wa\_mashed-name1 = wa\_lfa1-name1.  
      wa\_mashed-ort01 = wa\_lfa1-ort01.  
      wa\_mashed-regio = wa\_lfa1-regio.  
      wa\_mashed-land1 = wa\_lfa1-land1.  
  
      LOOP AT it\_ekko INTO wa\_ekko WHERE lifnr = wa\_lfa1-lifnr.  
        wa\_heditm-ebeln = wa\_ekko-ebeln.  
        wa\_heditm-lifnr = wa\_ekko-lifnr.  
        wa\_heditm-bukrs = wa\_ekko-bukrs.  
        wa\_heditm-butxt = wa\_ekko-butxt.  
        wa\_heditm-aedat = wa\_ekko-aedat.  
        wa\_heditm-ernam = wa\_ekko-ernam.  
  
        LOOP AT it\_ekpo INTO wa\_ekpo WHERE ebeln = wa\_ekko-ebeln.  
          wa\_itm-ebeln = wa\_ekpo-ebeln.  
          wa\_itm-ebelp = wa\_ekpo-ebelp.  
          wa\_itm-matnr = wa\_ekpo-matnr.  
          wa\_itm-maktx = wa\_ekpo-maktx.  
          wa\_itm-bukrs = wa\_ekpo-bukrs.  
          wa\_itm-werks = wa\_ekpo-werks.  
          APPEND wa\_itm TO wa\_heditm-headitemnav.  
          CLEAR: wa\_itm, wa\_ekpo.  
        ENDLOOP.  
        APPEND wa\_heditm TO wa\_mashed-mastheadnav.  
        CLEAR: wa\_heditm, wa\_ekko.  
      ENDLOOP.  
      APPEND wa\_mashed TO it\_mashed.  
      CLEAR: wa\_mashed, wa\_lfa1.  
    ENDLOOP.  
  
    copy\_data\_to\_ref( EXPORTING is\_data = it\_mashed  
                      CHANGING cr\_data = er\_entityset ).  
  
    INSERT lc\_expanded\_tech\_clauses INTO TABLE et\_expanded\_tech\_clauses.  
  
  endmethod. Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

-> Select our Service

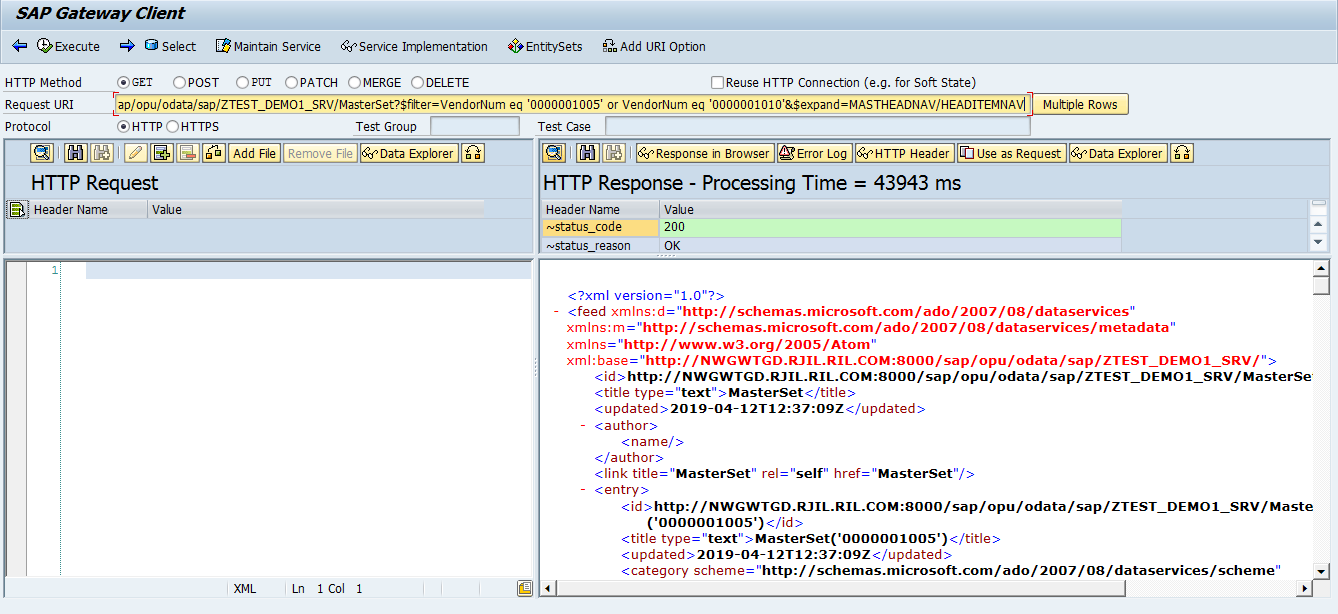
-> Click on Load Metadata

-> Click on Gateway Client

-> Click on EntitySets and Select Entityset Name

-> Select the HTTP Method ‘Get’

**URL:**/sap/opu/odata/sap/ZTEST\_DEMO1\_SRV/MasterSet?$filter=VendorNum eq '0000001005' or VendorNum eq '0000001010'&$expand=MASTHEADNAV/HEADITEMNAV



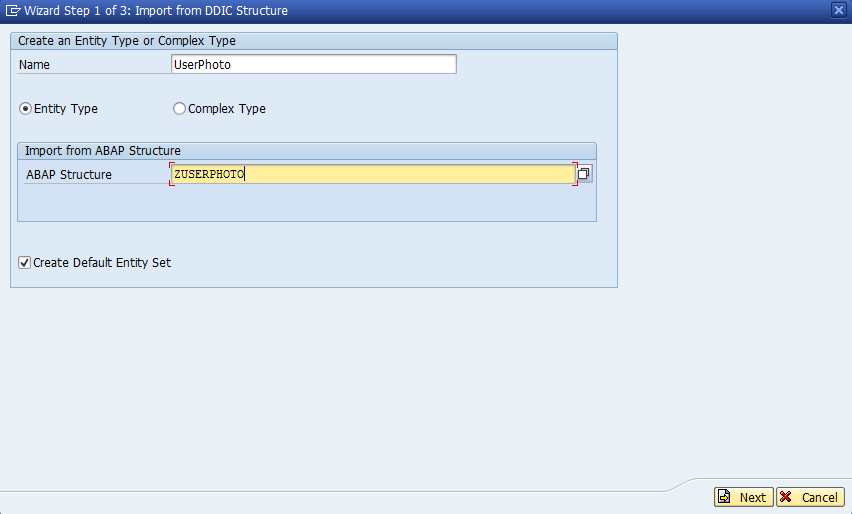
**Create Stream(File Upload) and Get Stream(File Download)**

**UserPhoto Entity Type:-**

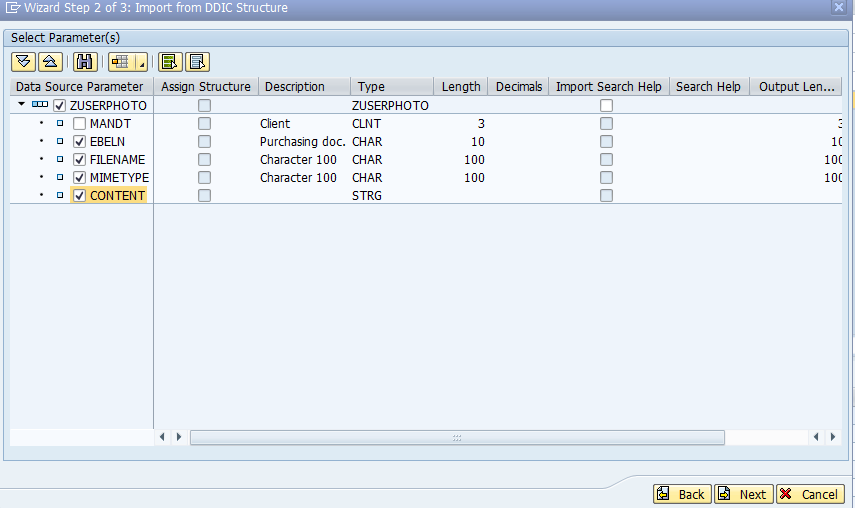
Expand The Project ‘ZTEST\_PAVAN’.

Select The Data Model and Right Click on it -> IMPORT -> DDIC Structure.

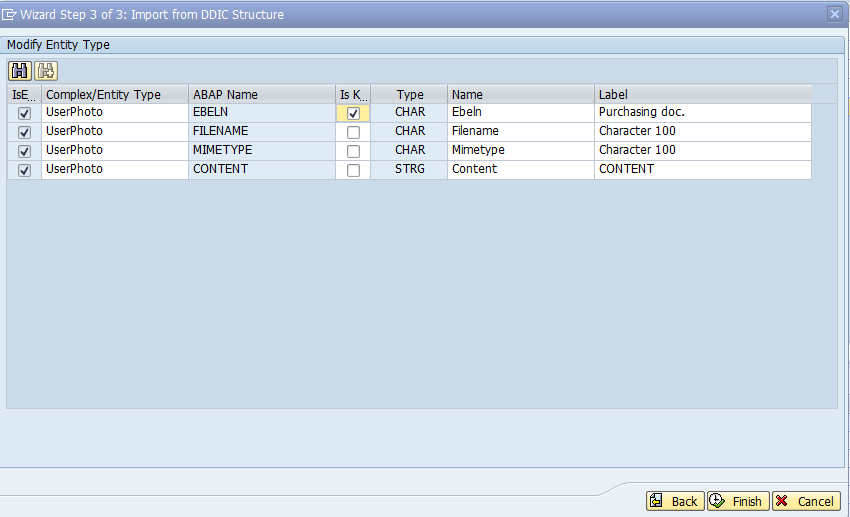
Provide Entity Name: ‘UserPhoto’, ABAP Structure Name: ‘ZUSERPHOTO’ Click on Next.



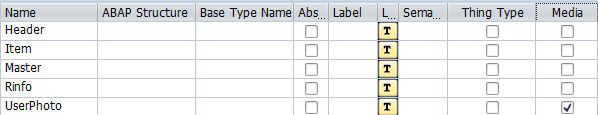
Select our Required Fields and Click on Next.



Select the check Box Is Key and Click on Finish.



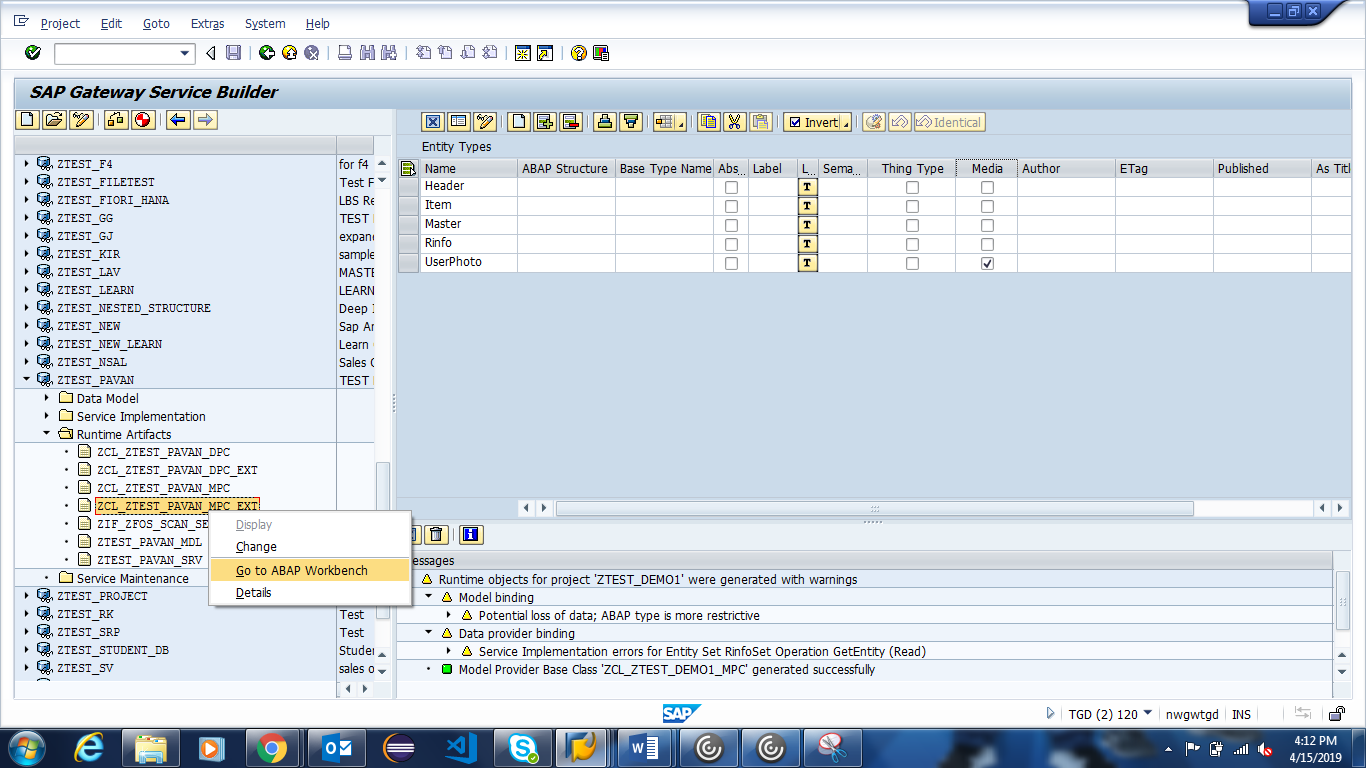
* Expand the Data Model->Expand the Entity Types->double click on Entity Type select the check box ‘Media’



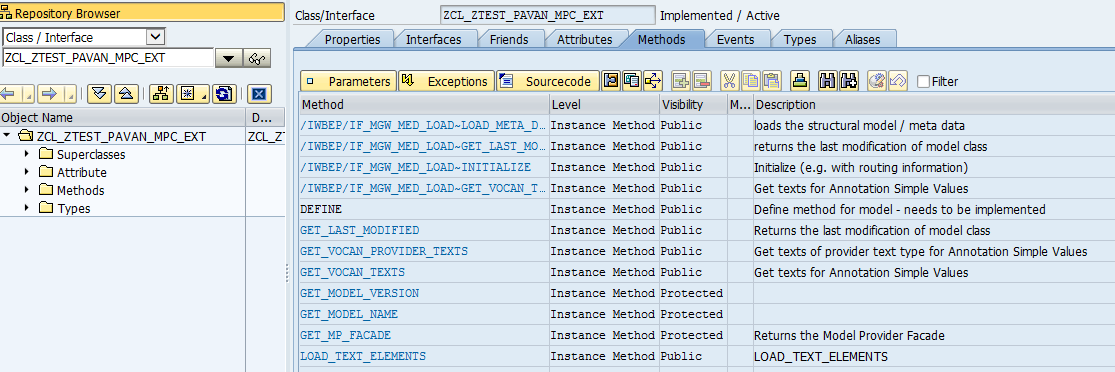
Save Project, Click on Check Project Consistency and Click on Generate Runtime Objects.

To Implement Methods Expand the Runtime Artifacts.

Select the ‘ZCL\_ZTEST\_PAVAN\_MPC\_EXT’ -> Right Click on it -> Go to Abap Work Bench



Select the ‘Define’ Method -> Click on Edit Mode -> Click on Redefine



  method DEFINE.  
    super->DEFINE( ).  
  
   data: lo\_entity TYPE ref to /IWBEP/IF\_MGW\_ODATA\_ENTITY\_TYP,  
         lo\_property TYPE REF TO /IWBEP/IF\_MGW\_ODATA\_PROPERTY.  
  
    lo\_entity = model->GET\_ENTITY\_TYPE( IV\_ENTITY\_NAME = 'UserPhoto').  
    if LO\_ENTITY is BOUND.  
    lo\_property = LO\_ENTITY->GET\_PROPERTY( IV\_PROPERTY\_NAME = 'Mimetype').  
    LO\_PROPERTY->SET\_AS\_CONTENT\_TYPE( ).  
    ENDIF.  
  endmethod.

Save, Check and Active the Method.

**CREATE\_STREAM Method:**

**Parameters Used in CREATE\_STREAM:**

**IS\_MEDIA\_RESOURCE:** Content type and content value. For example, Content type = ‘image/jpeg’ and content value ’sdfghjk’ Type TY\_S\_MEDIA\_RESOURCE

**IV\_SLUG:** Slug is a special HTTP header parameter to pass additional value to Create Stream. For example, slug

**CREATE\_STREAM Code:**

  METHOD /iwbep/if\_mgw\_appl\_srv\_runtime~create\_stream.  
    DATA : wa\_key\_tab   TYPE /iwbep/s\_mgw\_name\_value\_pair,  
           wa\_image     TYPE zuserphoto,  
           lv\_ebeln     TYPE zuserphoto-ebeln,  
           wa\_media     TYPE ty\_s\_media\_resource,  
           json         TYPE string,  
           ls\_return    TYPE bapiret2,  
           lt\_return    TYPE bapiret2\_t,  
           lv\_slug      TYPE string,  
           lw\_lheader   TYPE ihttpnvp,  
           lo\_message   TYPE REF TO zcl\_fiori\_format\_message\_json,  
           lo\_container TYPE REF TO /iwbep/if\_message\_container.  
  
    lv\_ebeln = iv\_slug.  
  
    CALL FUNCTION 'CONVERSION\_EXIT\_ALPHA\_INPUT'  
      EXPORTING  
        input  = lv\_ebeln  
      IMPORTING  
        output = lv\_ebeln.  
  
    wa\_image-ebeln = lv\_ebeln.  
    wa\_image-content = is\_media\_resource-value.  
    wa\_image-mimetype = is\_media\_resource-mime\_type.  
    wa\_image-filename = iv\_slug.  
  
    wa\_media = is\_media\_resource.  
  
  
    INSERT zuserphoto FROM wa\_image.  
    IF sy-subrc EQ 0.  
      COMMIT WORK.  
  
      ls\_return-type = 'S'.  
      ls\_return-message = 'Data Updated Successfully'.  
      APPEND ls\_return TO lt\_return.  
  
      CREATE OBJECT lo\_message.  
      CALL METHOD lo\_message->message\_format  
        EXPORTING  
          im\_returnmsg = lt\_return  
        IMPORTING  
          ex\_json      = json.  
  
  
      lw\_lheader-name = 'sap-message'.  
      lw\_lheader-value = json.  
      set\_header( is\_header = lw\_lheader ).  
  
    ELSE.  
      ls\_return-type = 'E'.  
      ls\_return-message = 'Error in updating data '.  
  
      lo\_container = me->mo\_context->get\_message\_container( ).  
  
      CALL METHOD lo\_container->add\_message  
        EXPORTING  
          iv\_msg\_type   = ls\_return-type  
          iv\_msg\_id     = ls\_return-id  
          iv\_msg\_number = ls\_return-number  
          iv\_msg\_text   = ls\_return-message.  
  
      RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
        EXPORTING  
          message\_container = lo\_container.  
    ENDIF.  
  
    copy\_data\_to\_ref( EXPORTING is\_data = wa\_image  
              CHANGING cr\_data = er\_entity ).  
  
  ENDMETHOD.

Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

-> Select our Service

-> Click on Load Metadata

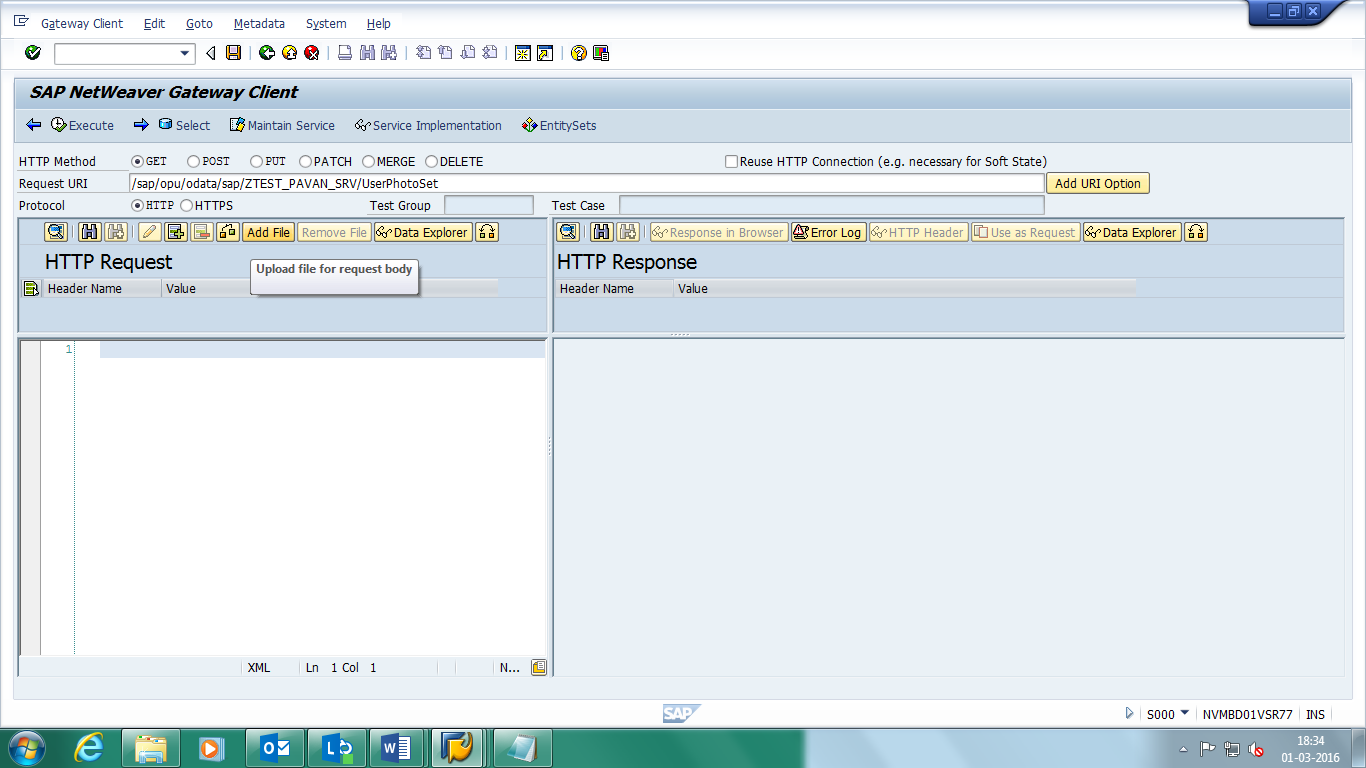
-> Click on Gateway Client

-> Click on EntitySets and Select Entityset Name

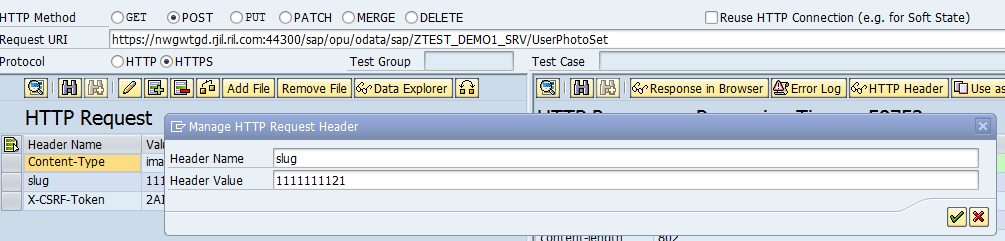
-> Select the HTTP Method ‘Post’

**URL:** /sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/UserPhotoSet

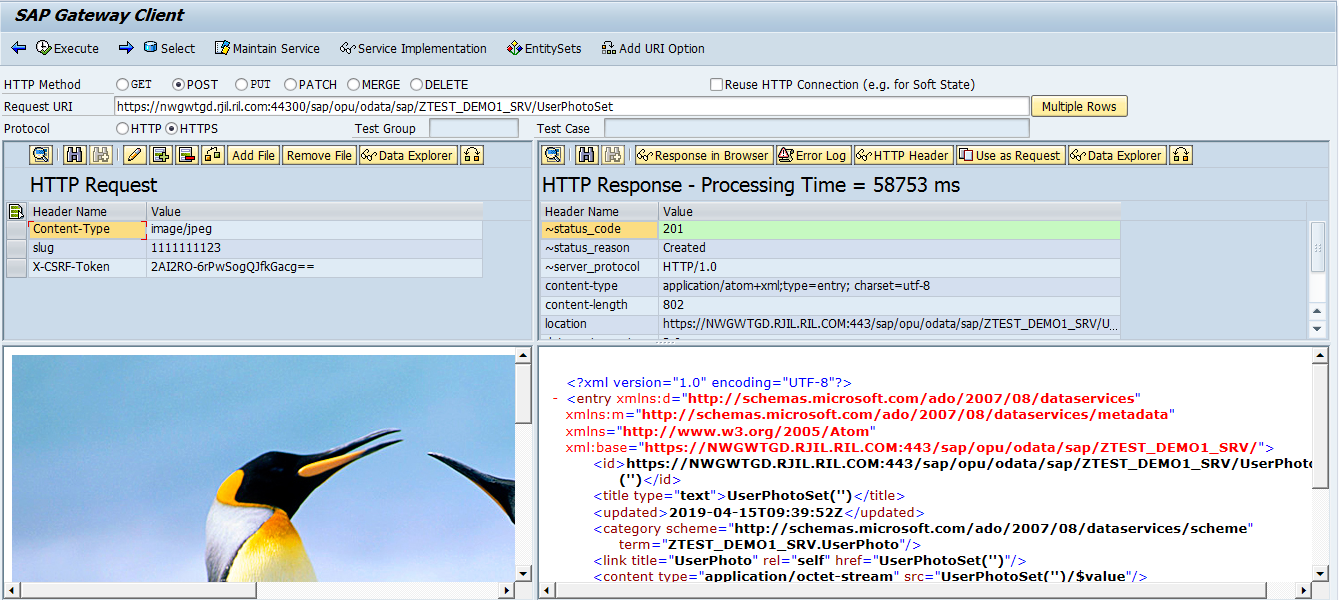
Click on ‘Add File’



Click on ‘Add Header’->Provide Header Name = ‘slug’ and Header Value = ‘1111111121’(Document Number)



Click on Execute.



**GET\_STREAM Method:**

**Parameters Used in GET\_STREAM:**

**ER\_STREAM:** The returned reference to the media resource structure which contains the binary data and the MIME type of a media resource.

**IT\_KEY\_TAB:** Represents the keys or NavPropCollection of the first segment. Type /IWBEP/T\_MGW\_NAME\_VALUE\_PAIR

**IV\_ENTITY\_NAME:** Name of the entity type which is requested. If there is a navigation it means that it represents the end/target of the navigation path. Type string

**GET\_STREAM Code:**

 method /IWBEP/IF\_MGW\_APPL\_SRV\_RUNTIME~GET\_STREAM.  
  
    DATA : wa\_key\_tab  TYPE /iwbep/s\_mgw\_name\_value\_pair,  
           wa\_image    TYPE zuserphoto,  
           lv\_ebeln    TYPE ebeln,  
           lw\_header   TYPE ihttpnvp,  
           lw\_filename TYPE string,  
           wa\_media    TYPE ty\_s\_media\_resource,  
          ls\_return    TYPE bapiret2,  
          lo\_container TYPE REF TO /iwbep/if\_message\_container.  
  
  
    CASE iv\_entity\_name.  
      WHEN 'UserPhoto'.  
  
        READ TABLE it\_key\_tab INTO wa\_key\_tab WITH KEY name = 'Ebeln'.  
        IF sy-subrc EQ 0.  
          lv\_ebeln = wa\_key\_tab-value.  
        ENDIF.  
  
        CALL FUNCTION 'CONVERSION\_EXIT\_ALPHA\_INPUT'  
          EXPORTING  
            input  = lv\_ebeln  
          IMPORTING  
            output = lv\_ebeln.  
  
        SELECT SINGLE \* FROM zuserphoto INTO wa\_image WHERE ebeln = lv\_ebeln.  
        IF wa\_image IS NOT INITIAL.  
  
          wa\_media-value = wa\_image-content.  
          wa\_media-mime\_type = wa\_image-mimetype.  
  
          lw\_filename = lv\_ebeln.  
          CONCATENATE lw\_filename '.PDF' into lw\_filename.  
  
          lw\_filename = escape( val    = lw\_filename  
                                format = cl\_abap\_format=>e\_url ).  
          lw\_header-name  = 'Content-Disposition'(003).  
          lw\_header-value = |inline; filename=*"{ lw\_filename }"|.*  
          set\_header( is\_header = lw\_header ).  
  
          copy\_data\_to\_ref( EXPORTING is\_data = wa\_media  
                                      CHANGING cr\_data = er\_stream ).  
          else.  
  
          ls\_return-type = 'E'.  
          ls\_return-message = 'No Data Found'.  
  
          lo\_container = me->mo\_context->get\_message\_container( ).  
  
          CALL METHOD lo\_container->add\_message  
            EXPORTING  
              iv\_msg\_type   = ls\_return-type  
              iv\_msg\_id     = ls\_return-id  
              iv\_msg\_number = ls\_return-number  
              iv\_msg\_text   = ls\_return-message.  
  
          RAISE EXCEPTION TYPE /iwbep/cx\_mgw\_busi\_exception  
            EXPORTING  
              message\_container = lo\_container.  
  
  
        ENDIF.  
    ENDCASE.

endmethod.

Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

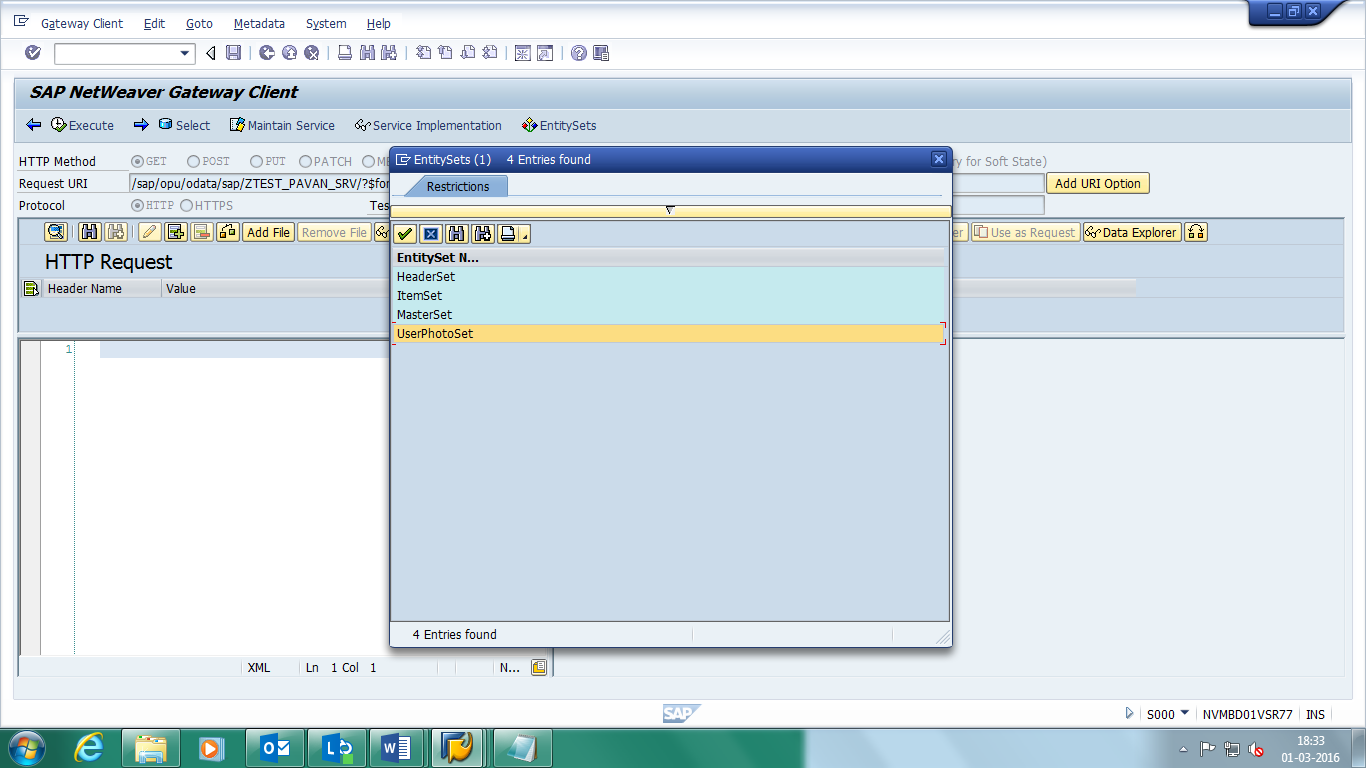
-> Select our Service

-> Click on Load Metadata

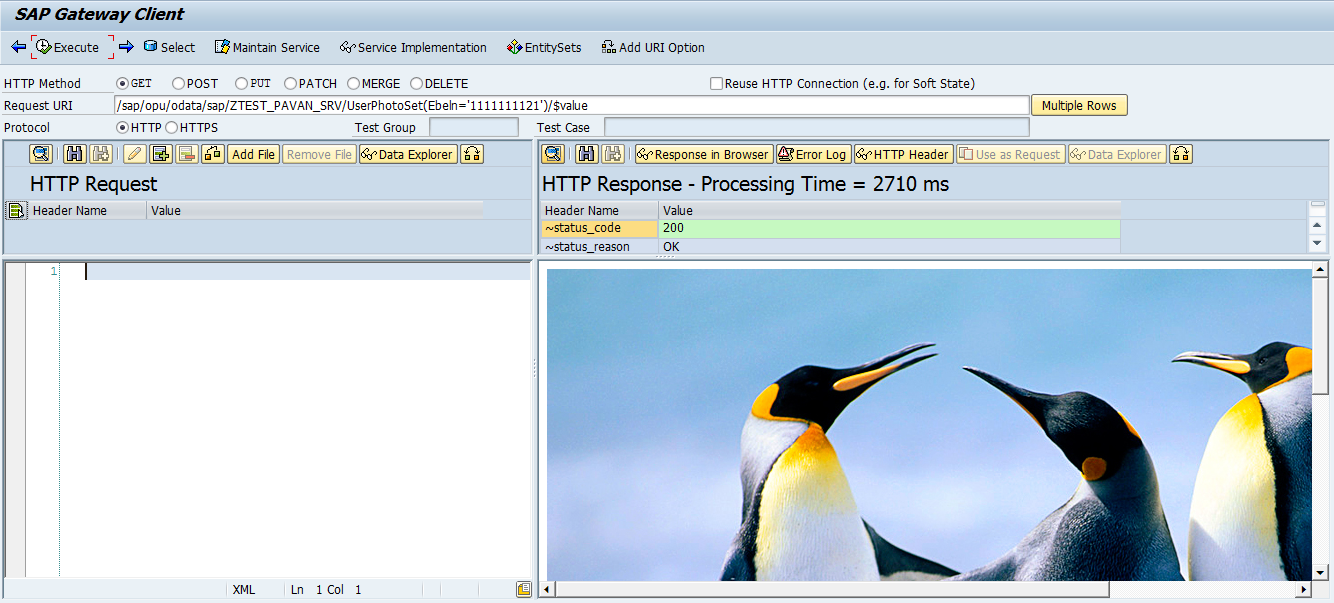
-> Click on Gateway Client

-> Click on EntitySets and Select Entityset Name

-> Select the HTTP Method ‘Get’



**URL:** /sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/UserPhotoSet(Eblen='1111111121')/$value



**Multi Origin Composition:**

Multiple Origin Composition (MOC) is a new and an emerging concept in SAP that describes the ability to collect data from different backend systems, integrate them into a single service and update diverse backend systems, while using the same user. we need to understand few constraints (limitations) for implementing MOC, which are as follows:

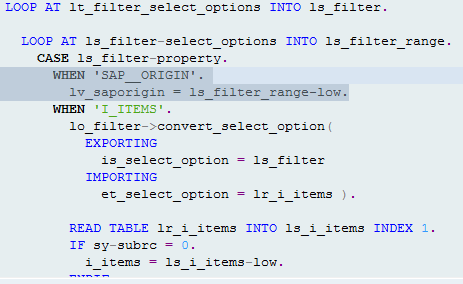
1. OData Project name and service name must be same.
2. OData EntitySet and Properties name must be same.
3. Login user must available in backend systems.

Before register our Odata service we need to redefine and write below code in each and every GetEntitySet method which are implemented in our Odata project.

  DATA lv\_saporigin TYPE c LENGTH 11.

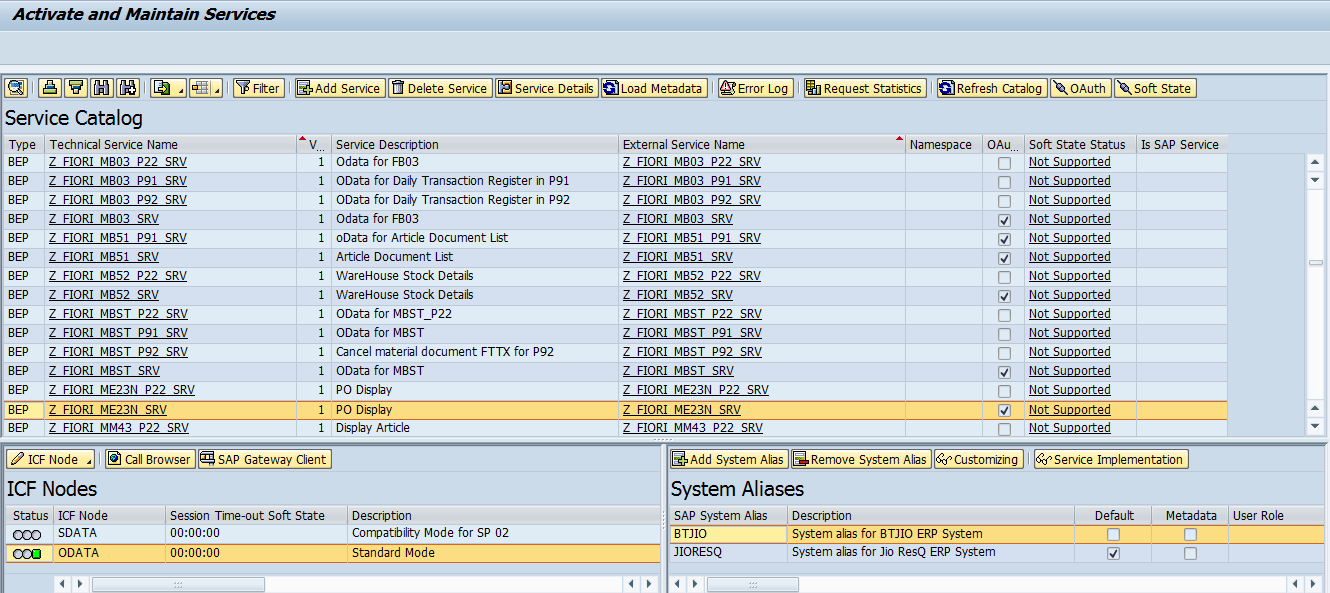
        WHEN 'SAP\_\_ORIGIN'.  
          lv\_saporigin = ls\_filter\_range-low.



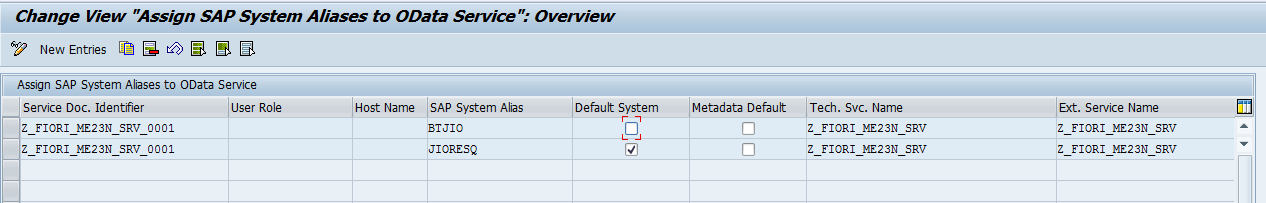


After register the service name we need add System Aliases in ‘/iwfnd/maint\_service’ Tcode.

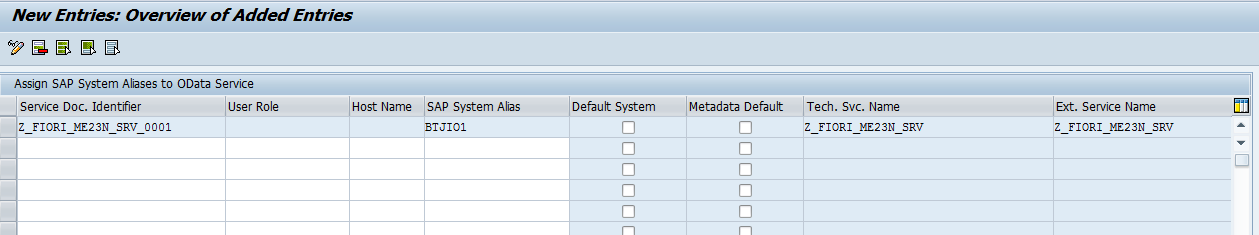
-> Select our Service-> Click on Load Metadata



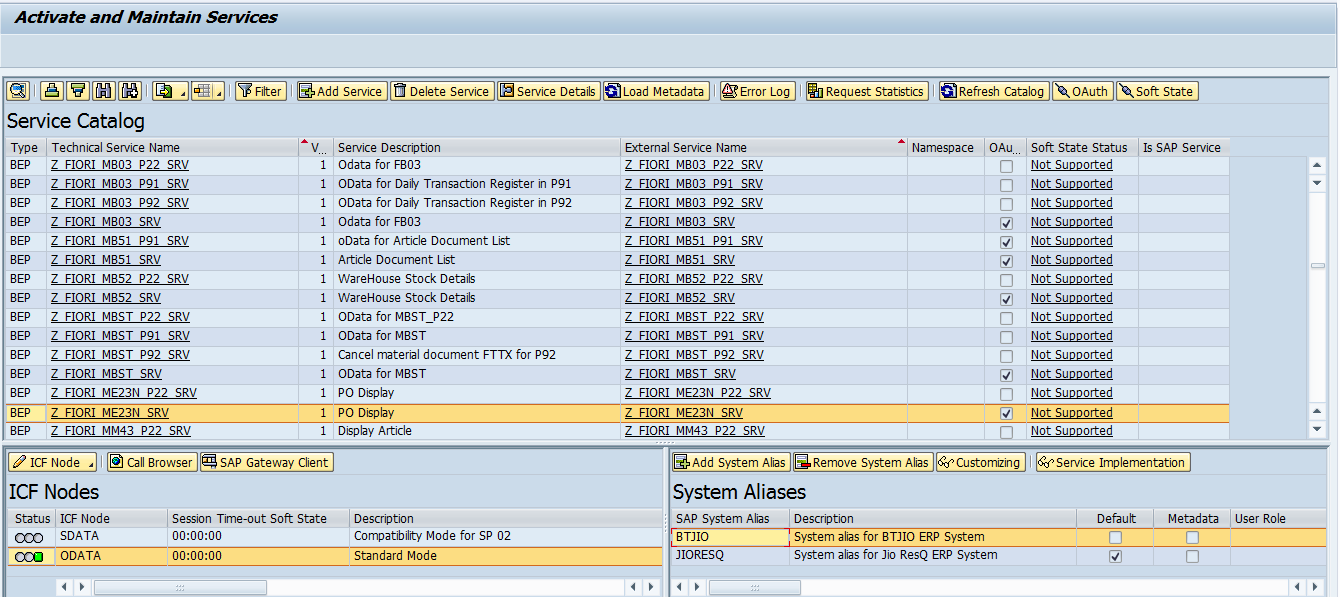
Click on Add system Aliases -> Click on Enter



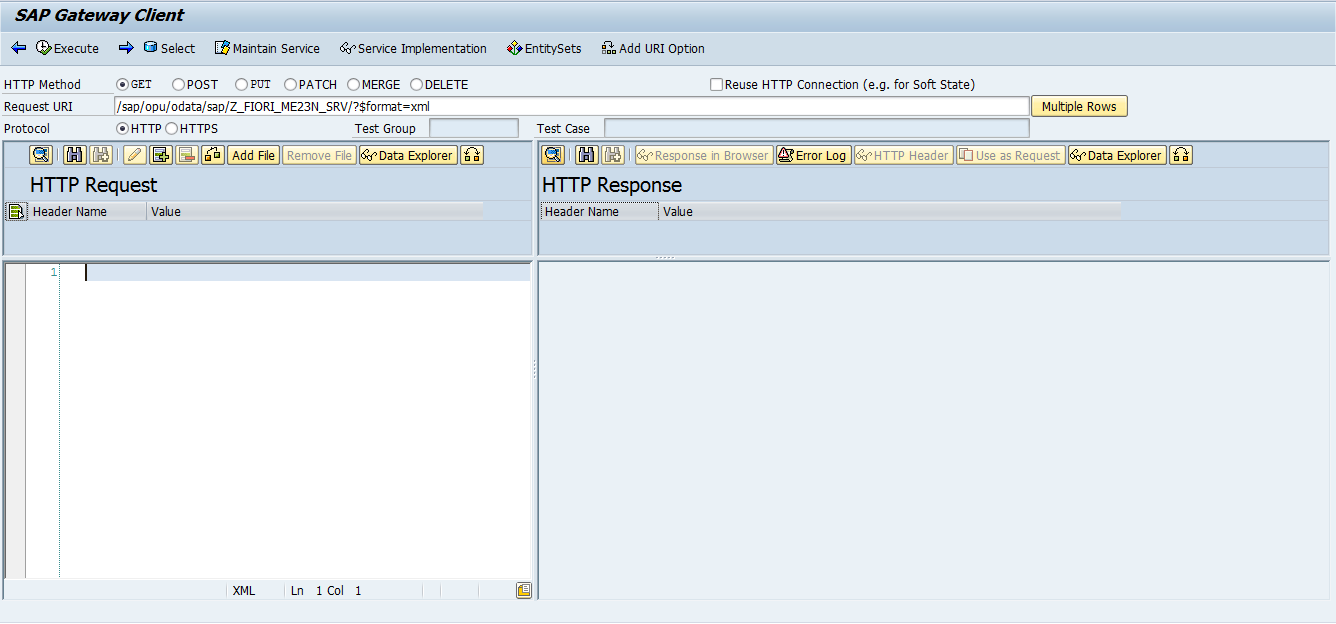
Click on New Entries -> Provide require details.



Click on Save -> click on Back



Click on Sap Gateway Client.



URL’S:

GetEntitySet URL’s:

URL for fetching data from two different Servers:

MaterialListSet:

/sap/opu/odata/sap/Z\_FIORI\_ME23N\_SRV;mo/MaterialListSet?$filter=(SAP\_\_Origin eq 'BTJIO' or SAP\_\_Origin eq 'JIORESQ') and Article eq '400004351' and ArticleIndicator eq 'X' and ArtDescrIndicator eq '' and Site eq '8737' and ArticleDescr eq ''

URL for fetching data from single Server:

MaterialListSet:

/sap/opu/odata/sap/Z\_FIORI\_ME23N\_SRV;mo/MaterialListSet?$filter= SAP\_\_Origin eq 'JIORESQ' and Article eq '400004351' and ArticleIndicator eq 'X' and ArtDescrIndicator eq '' and Site eq '8737' and ArticleDescr eq ''

Note : in Multi Origin concept we can fetch the data form one backend system by using GetEntity method.

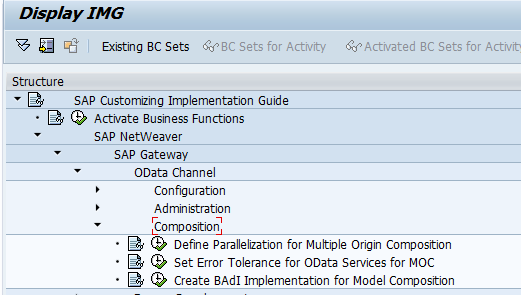
GetEntity URL’s:

POHeaderSet:

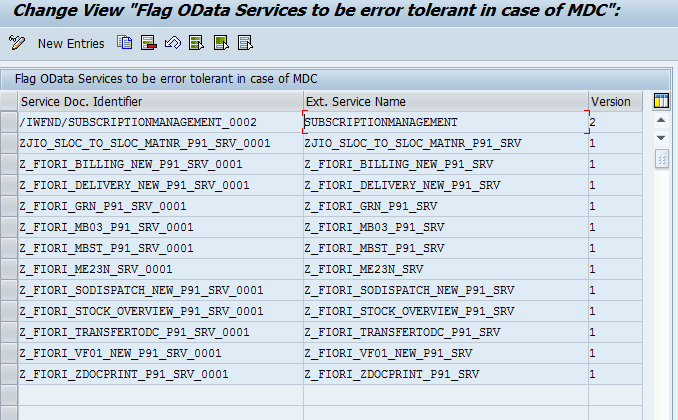
/sap/opu/odata/sap/Z\_FIORI\_ME23N\_SRV;mo/POHeaderSet(SAP\_\_Origin='JIORESQ',PurchaseOrderNum='6000009506')

Note: after implementing multi origin concept we have configure our service in TGD server by using SPRO tcode.

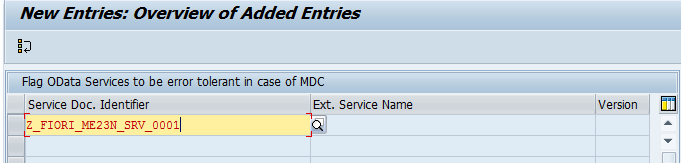
Execute SPRO tcode -> Click on SAP Reference IMG.



Execute Set Error Tolerance for OData Service for MOC



Click on New Entries.



Click on Save.

**Batch Operations:**

OData Batch requests allow the grouping of multiple operations into a single HTTP request payload. Batch Requests are submitted as a single HTTP POST request to the batch endpoint of a service.

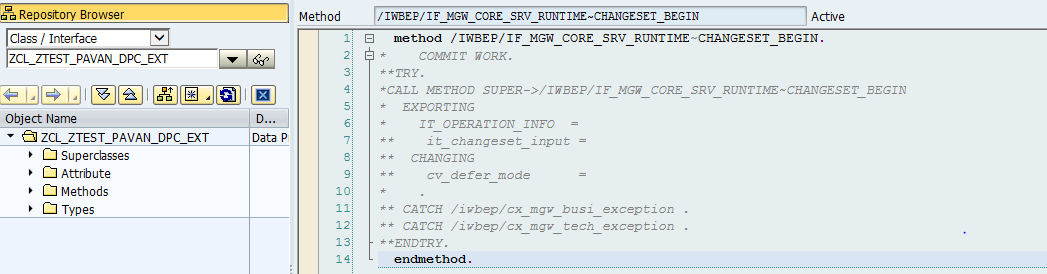
Execute ‘SEGW’ Tcode

.-> Expand the Project

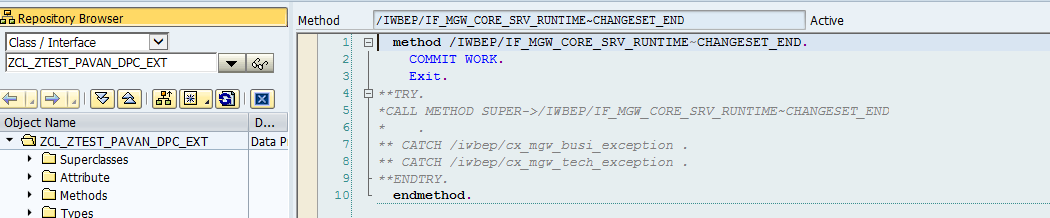
-> Expand the Runtime Artifacts

-> Redefine below methods.

    1./IWBEP/IF\_MGW\_APPL\_SRV\_RUNTIME~CHANGESET\_BEGIN.



     2./IWBEP/IF\_MGW\_APPL\_SRV\_RUNTIME~CHANGESET\_END.



Save, Check and Active the Method.

Open the ‘ACTIVATE AND MAINTAIN SERVICES’ in New Session

-> Select our Service

-> Click on Load Metadata

-> Click on Gateway Client

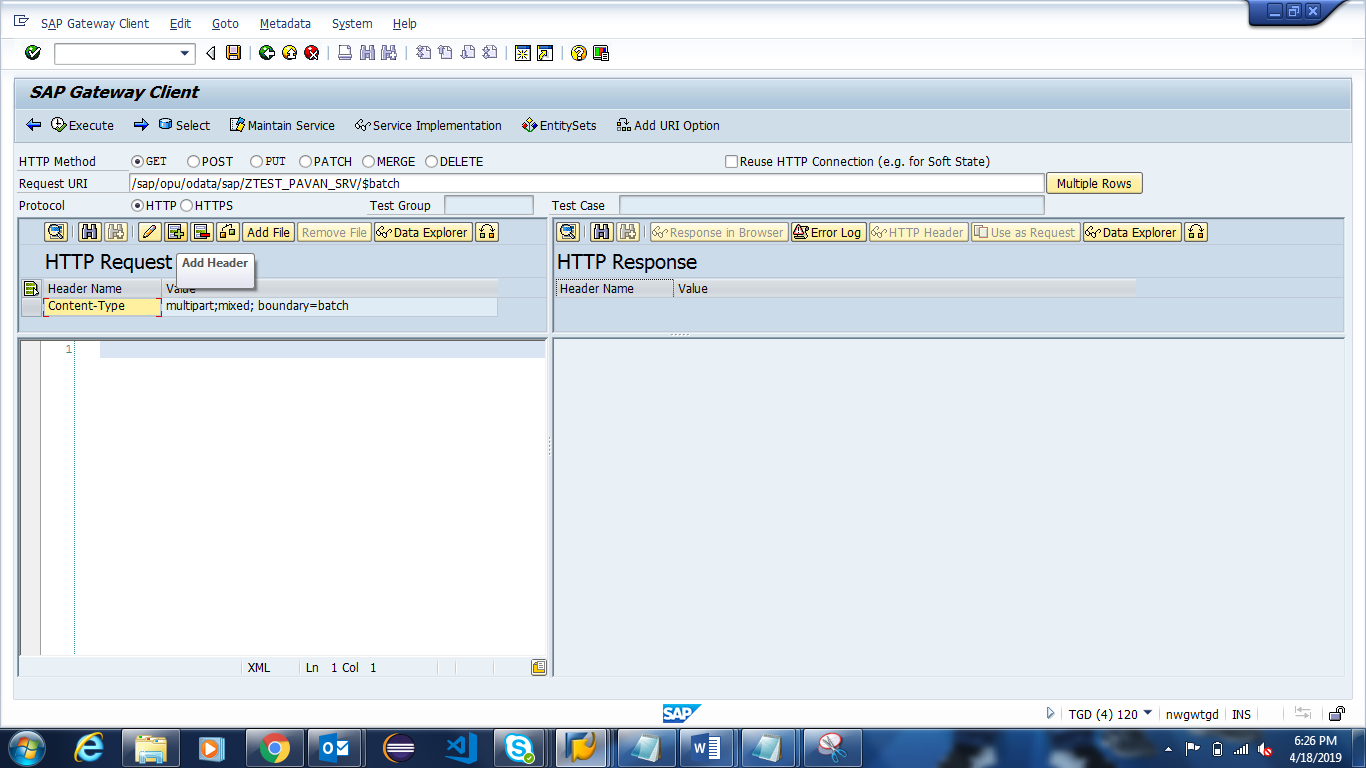
-> Click on EntitySets and Select Entityset Name

-> Select the HTTP Method ‘Post’

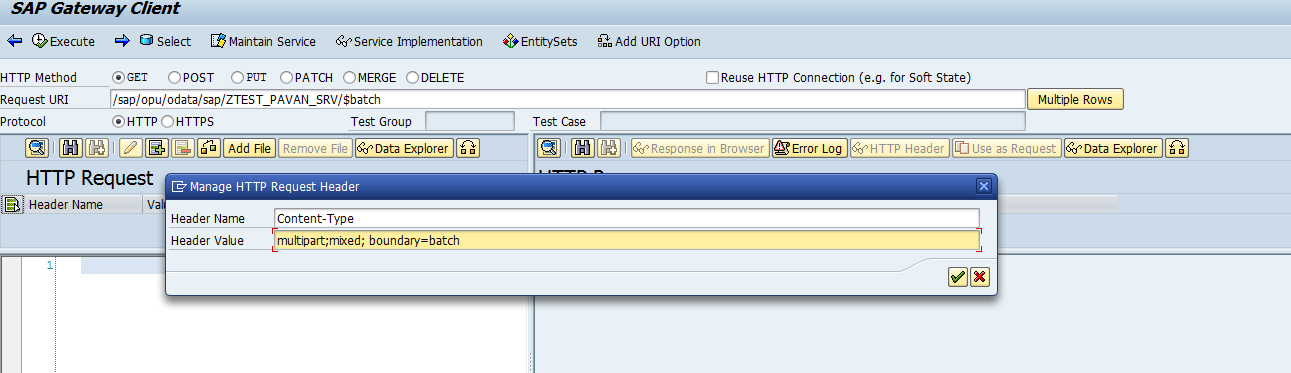
-> Write the Payload in HTTP Request

**URL:** /sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/$batch

Click on Add Header



-> provide Header Name: Content-Type and Header Value: multiport/mixed; boundary=batch



**Payload for Display:**

--batch

Content-Type: application/http

Content-Transfer-Encoding: binary

GET MasterSet?$filter=VendorNum+eq+'0000001006' HTTP/1.1

--batch

Content-Type: application/http

Content-Transfer-Encoding: binary

GET MasterSet?$filter=VendorNum+eq+'0000001006' HTTP/1.1

--batch--

**Payload for Posting:**

--batch

Content-Type: multipart/mixed; boundary=changeset

--changeset

Content-Type: application/http

Content-Transfer-Encoding: binary

POST MasterSet HTTP/1.1

Content-Type: application/atom+xml

Content-Length: 985

<?xml version="1.0" encoding="UTF-8"?>

<atom:entry xmlns:atom ="http://www.w3.org/2005/Atom"

xml:base="http://nvmbd01vsr77.bss.dev.jio.com:8000/sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/"

xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"

xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices">

<atom:content type="application/xml">

<m:properties>

<d:VendorNum>0000001004</d:VendorNum>

<d:VendorName>MANESH</d:VendorName>

<d:City>MUM</d:City>

<d:State>MP</d:State>

<d:Country>IN</d:Country>

</m:properties>

</atom:content>

</atom:entry>

--changeset

Content-Type: application/http

Content-Transfer-Encoding: binary

POST MasterSet HTTP/1.1

Content-Type: application/atom+xml

Content-Length: 985

<?xml version="1.0" encoding="UTF-8"?>

<atom:entry xmlns:atom ="http://www.w3.org/2005/Atom"

xml:base="http://nvmbd01vsr77.bss.dev.jio.com:8000/sap/opu/odata/sap/ZTEST\_PAVAN\_SRV/"

xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"

xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices">

<atom:content type="application/xml">

<m:properties>

<d:VendorNum>0000001004</d:VendorNum>

<d:VendorName>MANESH</d:VendorName>

<d:City>MUM</d:City>

<d:State>MP</d:State>

<d:Country>IN</d:Country>

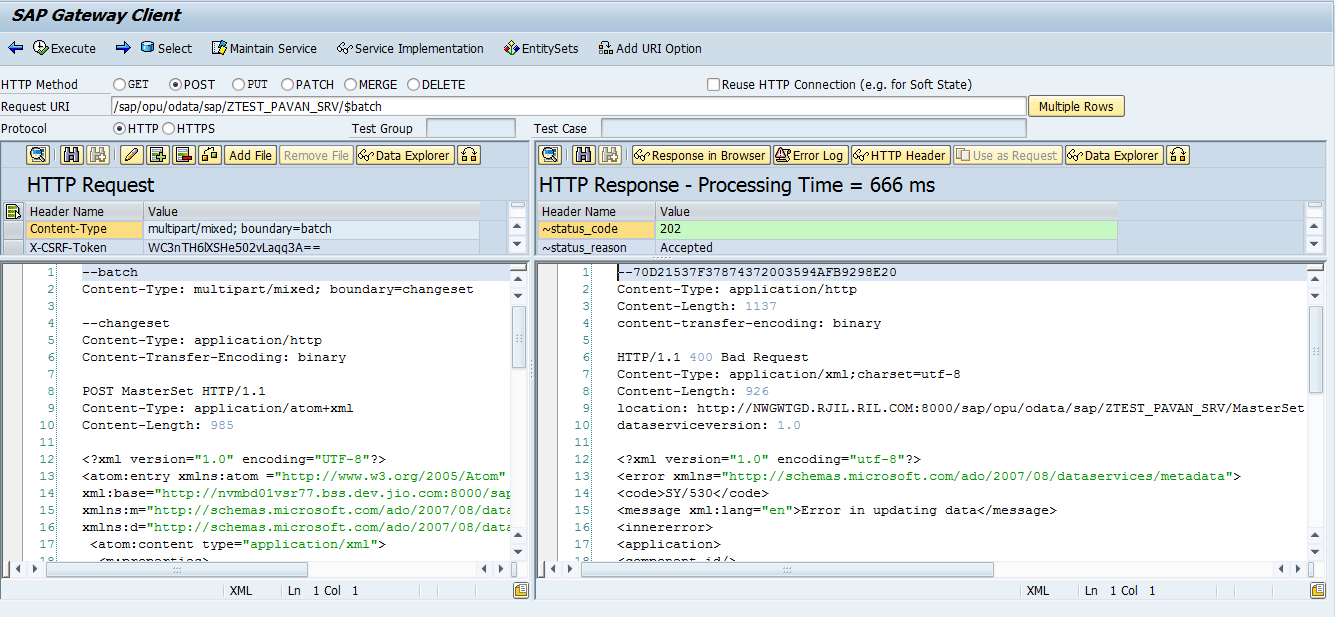
</m:properties>

</atom:content>

</atom:entry>

--changeset--

--batch--



**\*** **THANKING YOU\***