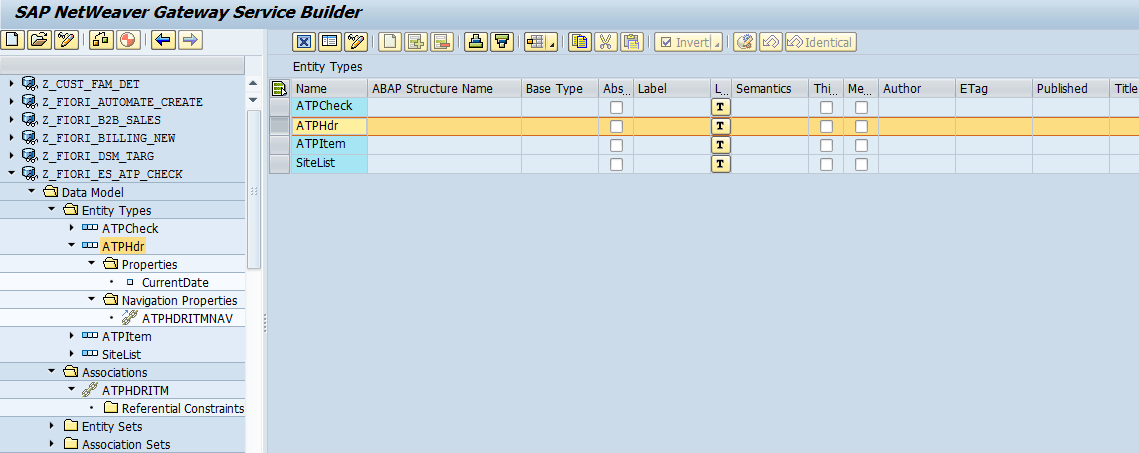
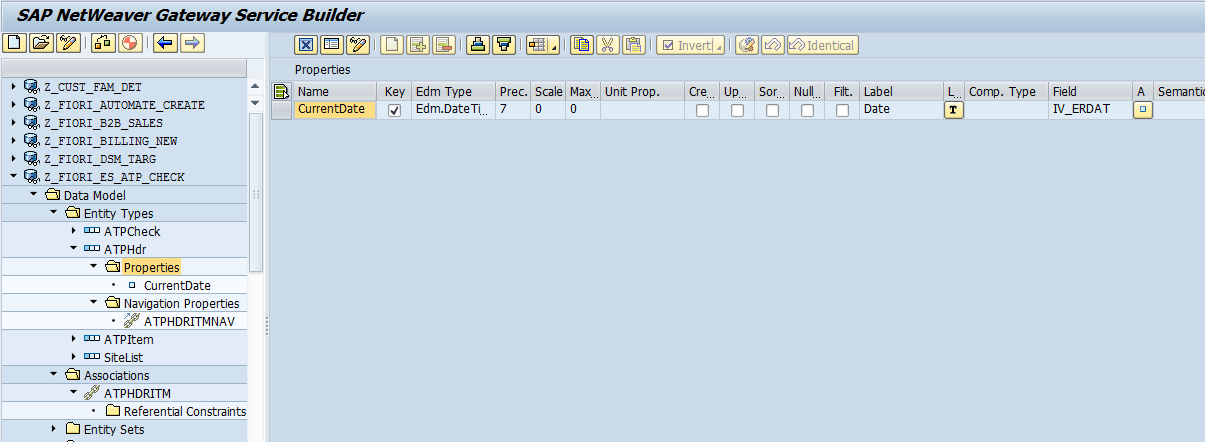
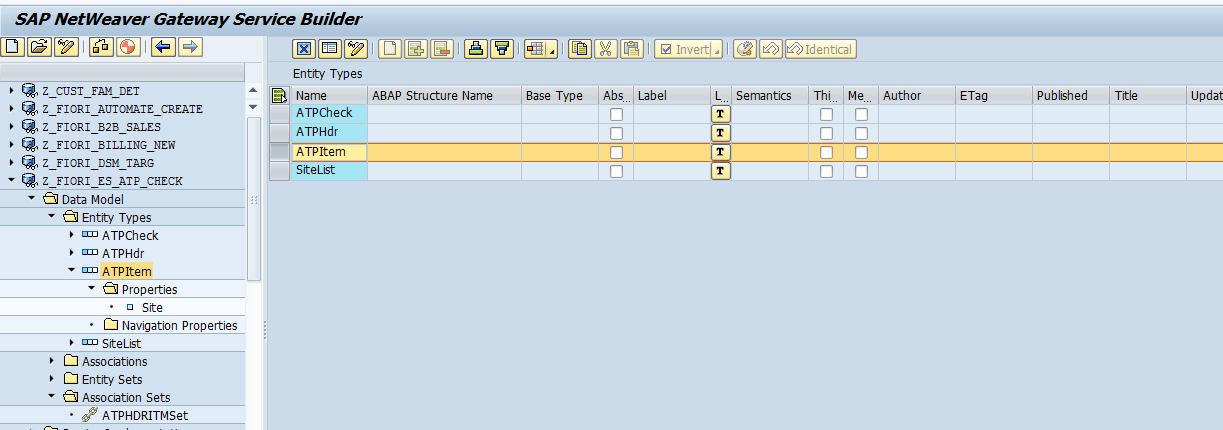
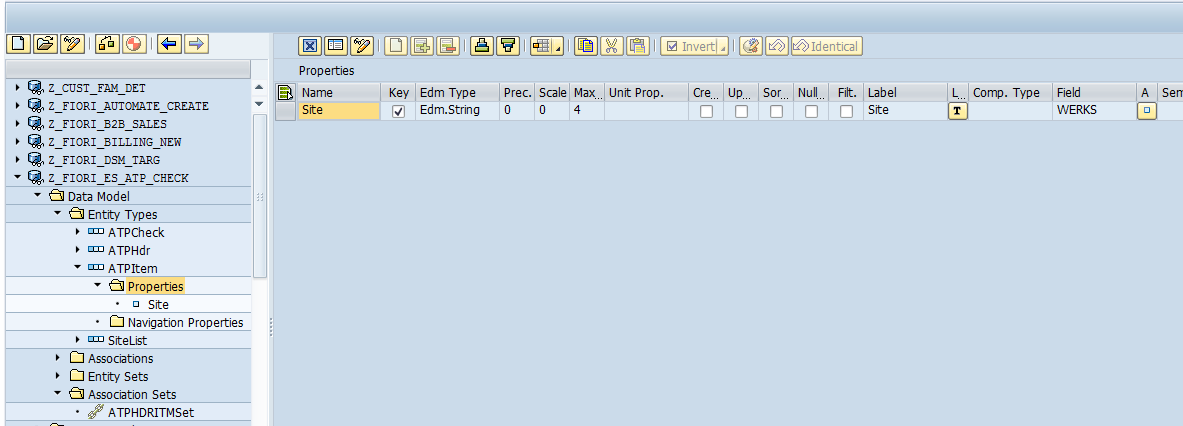
Create Deep Entity:

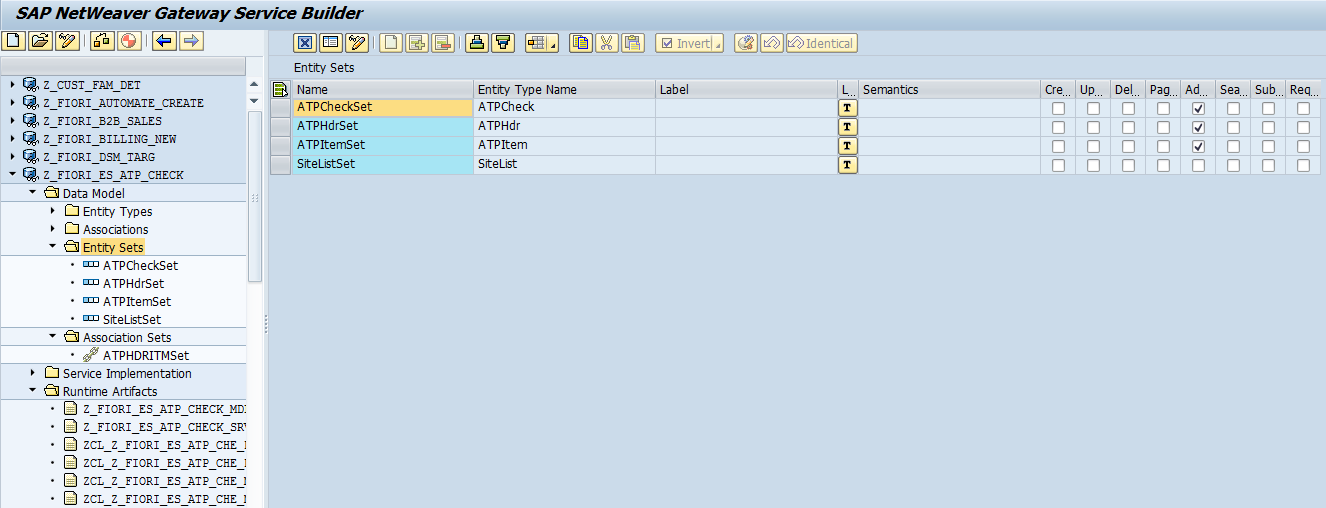




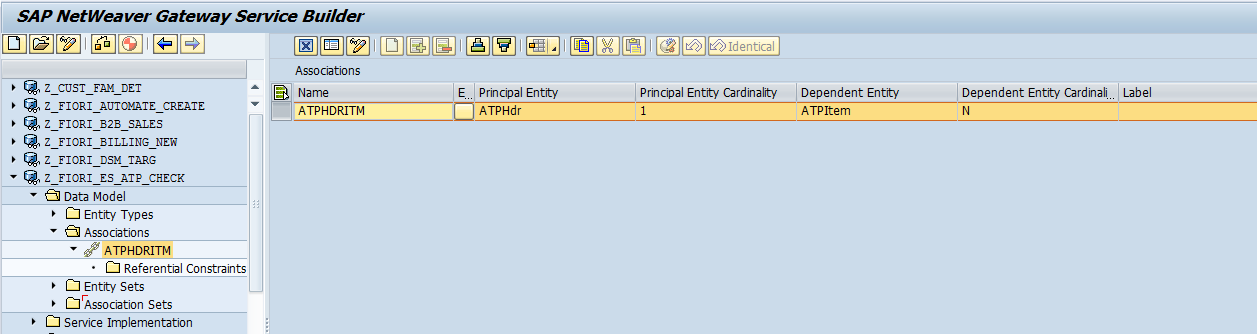
ATPItemSet:

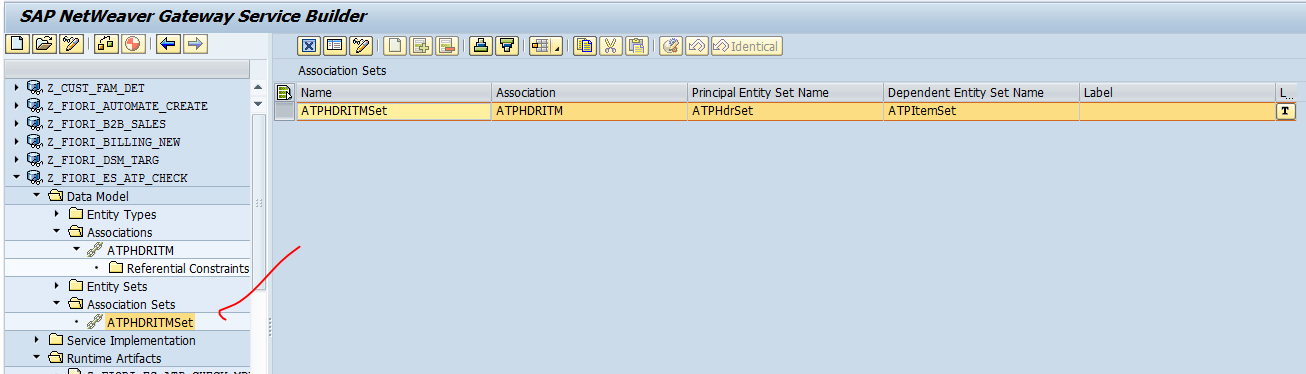




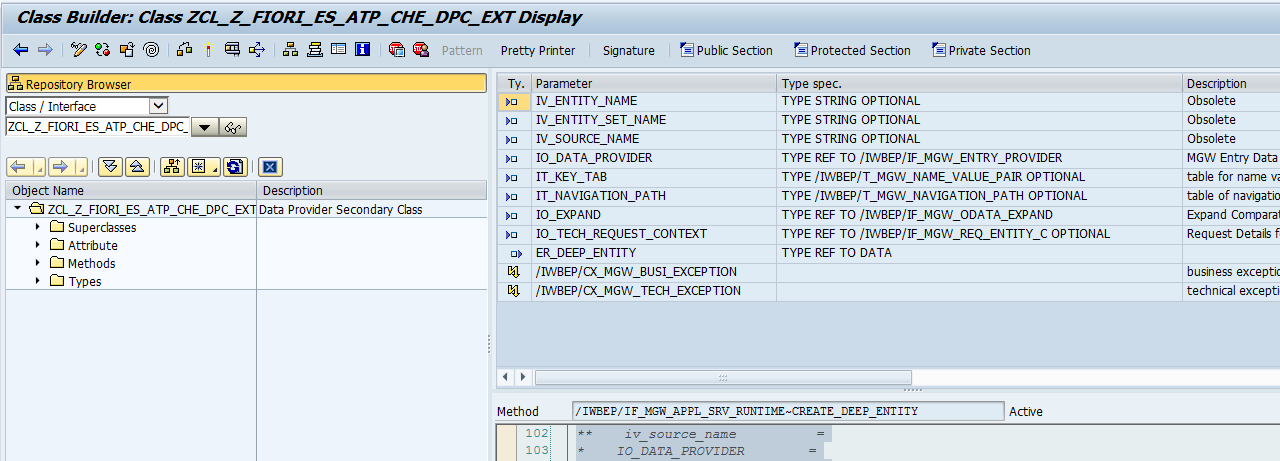


Associations:





Signatute:



Code:

Method /IWBEP/IF\_MGW\_APPL\_SRV\_RUNTIME~CREATE\_DEEP\_ENTITY.

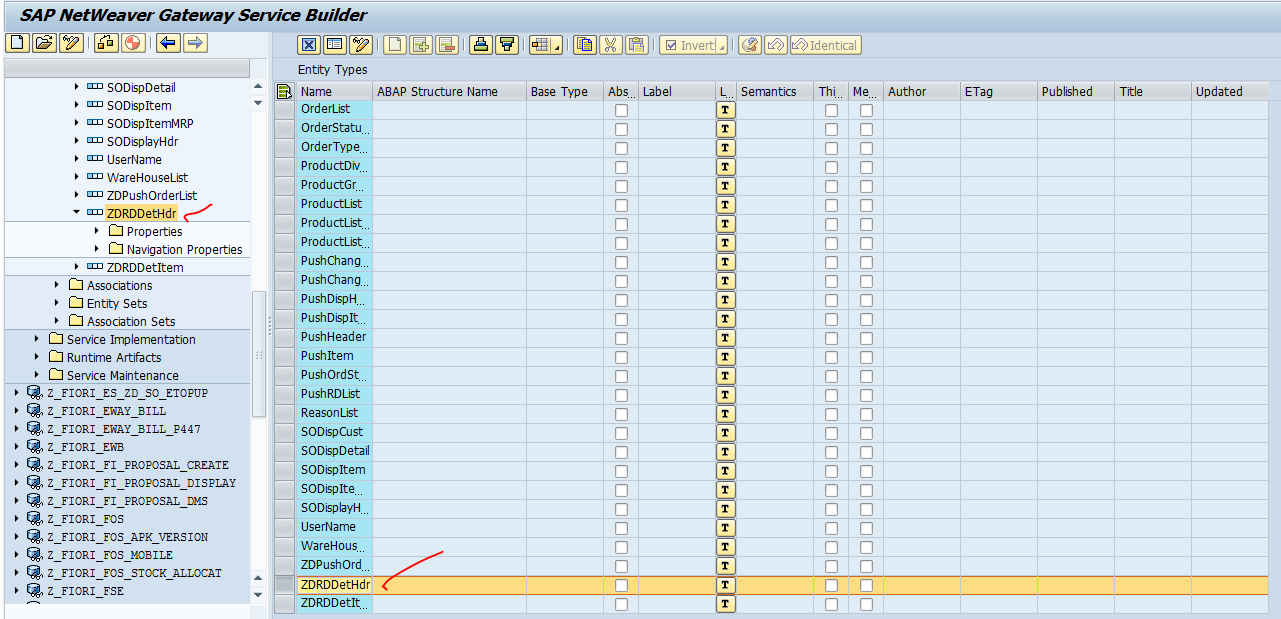
  TYPES : BEGIN OF ty\_headitem.  
  INCLUDE        TYPE zcl\_z\_fiori\_es\_atp\_che\_mpc=>ts\_atphdr.  
  TYPES : atphdritmnav TYPE STANDARD TABLE OF zcl\_z\_fiori\_es\_atp\_che\_mpc=>ts\_atpitem WITH DEFAULT KEY,  
          END OF ty\_headitem.  
  
  DATA  : lw\_headitem      TYPE ty\_headitem,  
          lv\_date TYPE dats,  
          lw\_item TYPE zcl\_z\_fiori\_es\_atp\_che\_mpc=>ts\_atpitem,  
          ls\_item TYPE werks,  
          lt\_item TYPE wrf\_pbas\_werks\_tty.  
  
  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 = lw\_headitem ).  
  
  IF lw\_headitem IS NOT INITIAL.  
    lv\_date = lw\_headitem-iv\_erdat = sy-datum.  
  
    LOOP AT lw\_headitem-atphdritmnav INTO lw\_item.  
      ls\_item-werks = lw\_item-werks.  
      APPEND ls\_item TO lt\_item.  
      CLEAR ls\_item.  
    ENDLOOP.  
  ENDIF.  
  
  CALL FUNCTION 'ZSTO\_AVCKCHECK'  
   EXPORTING  
\*     IT\_EBELN        =  
     IT\_WERKS        = lt\_item  
     IV\_ERDAT        = lv\_date  
   IMPORTING  
     ET\_RETURN       = lt\_return.  
  
  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 ).  
    ENDIF.  
  ENDIF.  
  
  CALL METHOD me->copy\_data\_to\_ref  
    EXPORTING  
      is\_data = lw\_headitem  
    CHANGING  
      cr\_data = er\_deep\_entity.  
  
\*\*TRY.  
\*CALL METHOD SUPER->/IWBEP/IF\_MGW\_APPL\_SRV\_RUNTIME~CREATE\_DEEP\_ENTITY  
\*  EXPORTING  
\*\*    iv\_entity\_name          =  
\*\*    iv\_entity\_set\_name      =  
\*\*    iv\_source\_name          =  
\*    IO\_DATA\_PROVIDER        =  
\*\*    it\_key\_tab              =  
\*\*    it\_navigation\_path      =  
\*    IO\_EXPAND               =  
\*\*    io\_tech\_request\_context =  
\*\*  IMPORTING  
\*\*    er\_deep\_entity          =  
\*    .  
\*\* CATCH /iwbep/cx\_mgw\_busi\_exception .  
\*\* CATCH /iwbep/cx\_mgw\_tech\_exception .  
\*\*ENDTRY.

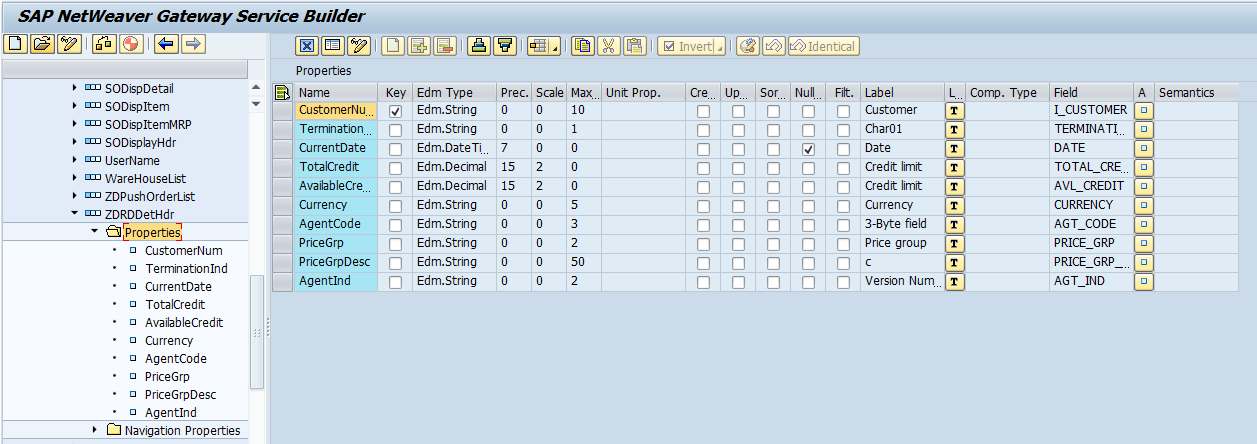
Endmethod.

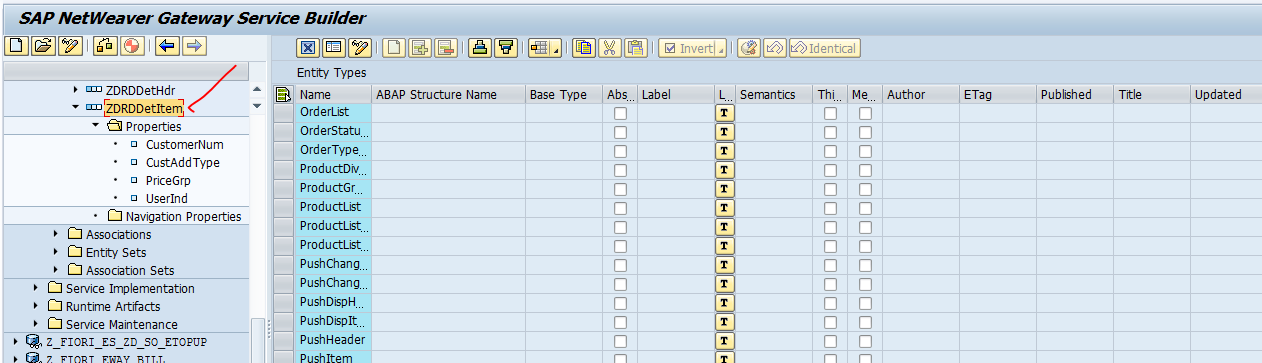
Get expand Enity set

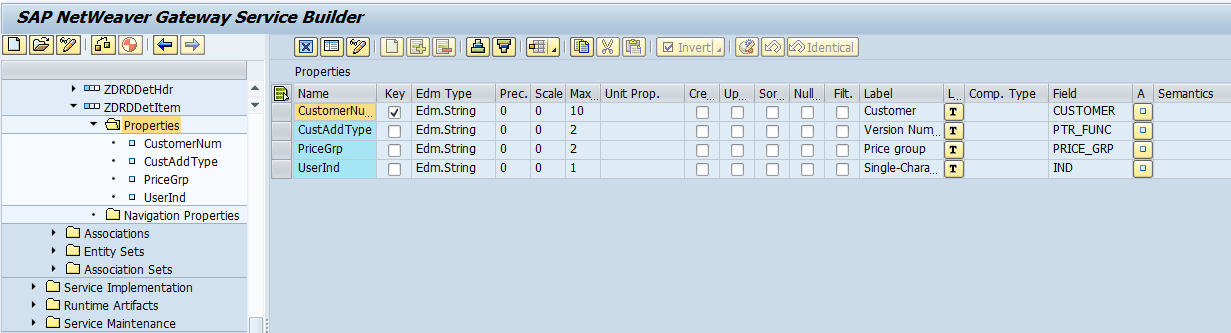
**Get Expand Entity**

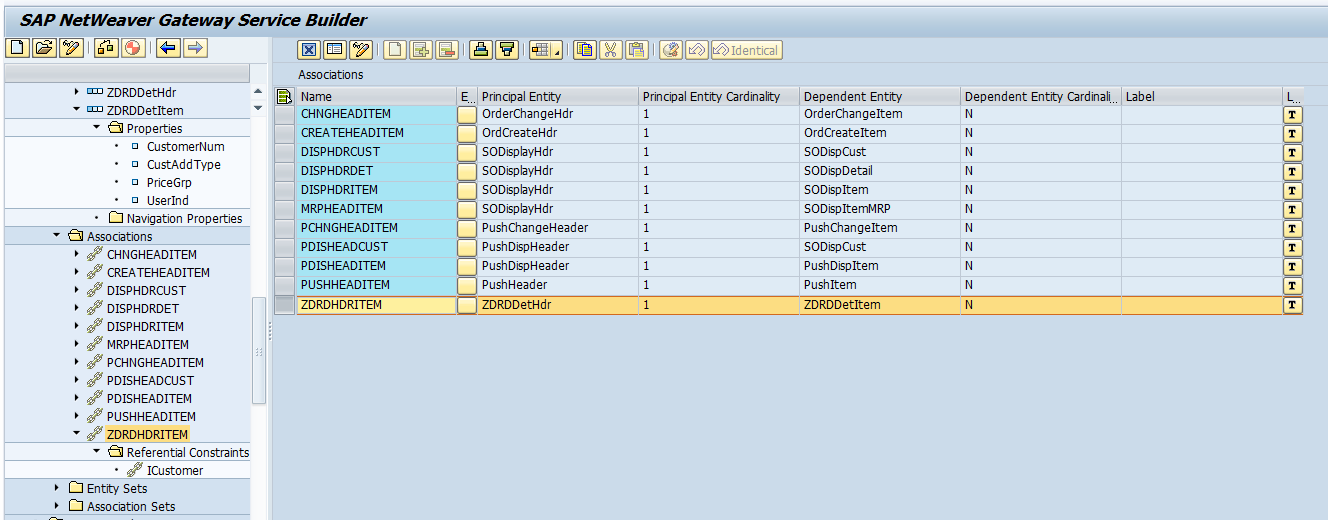
**HDR 🡪ITEM**

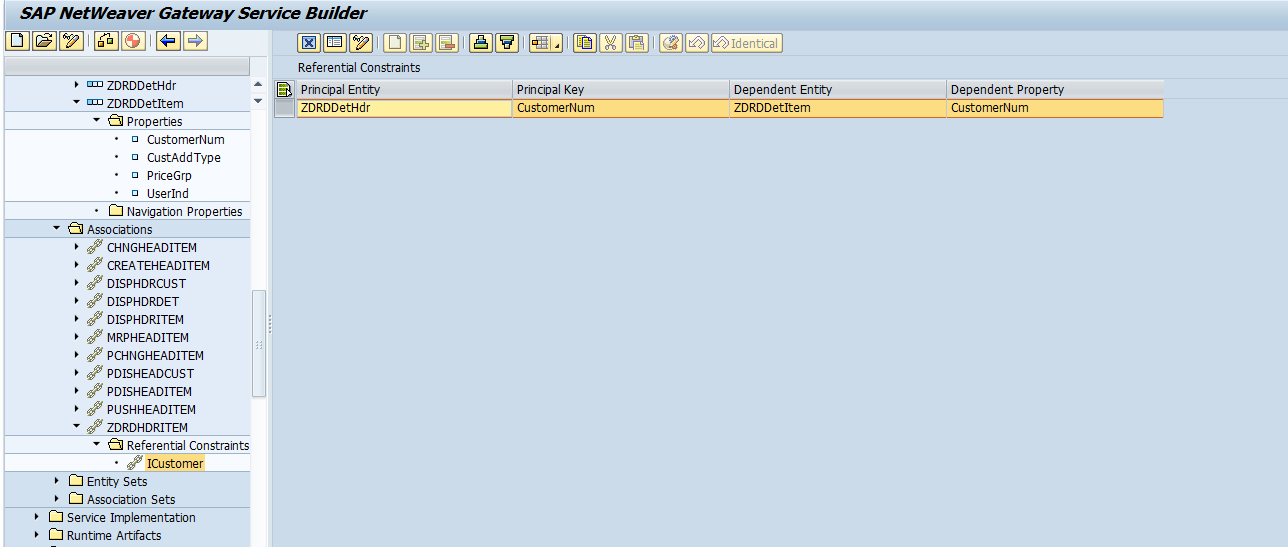


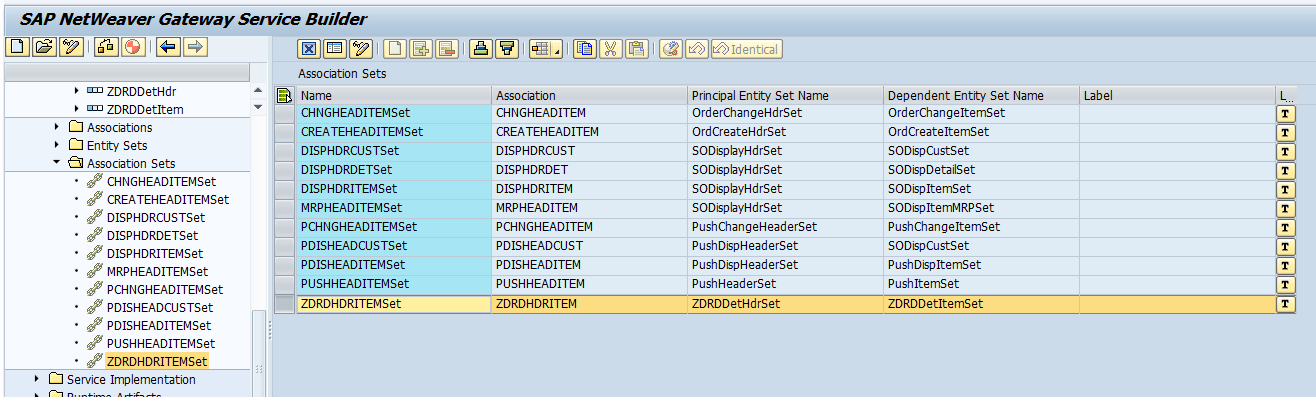


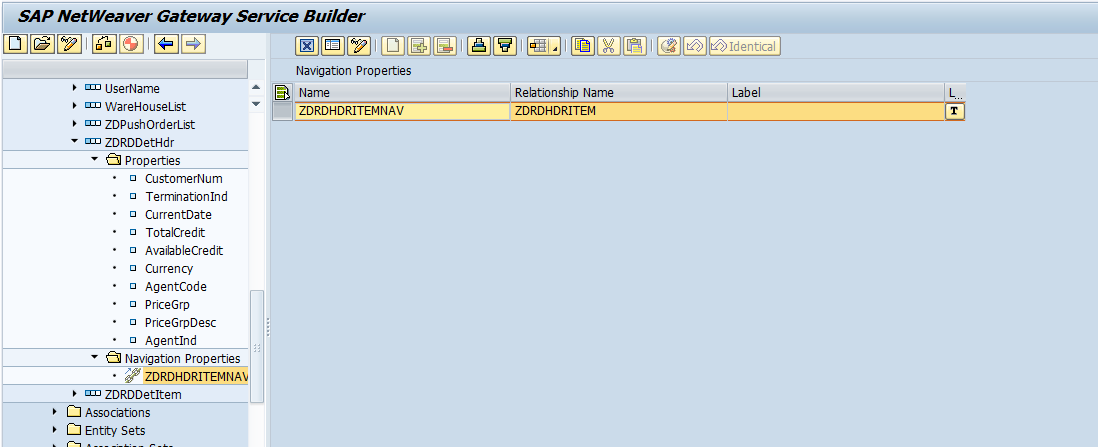










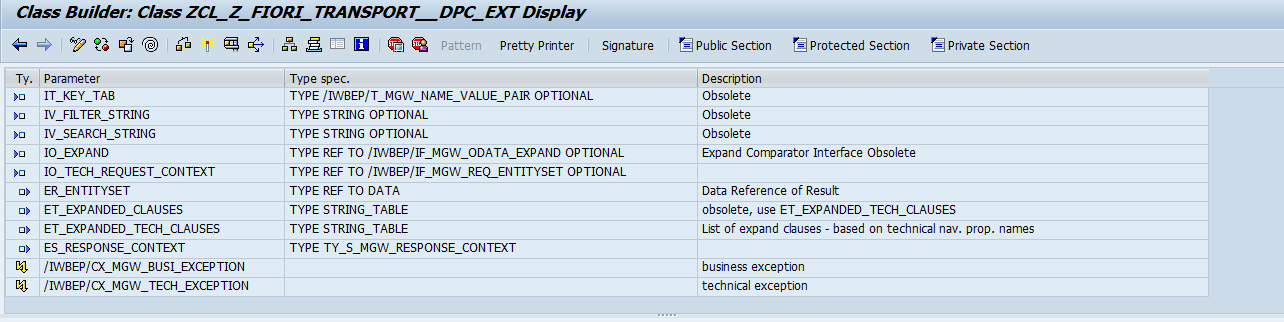
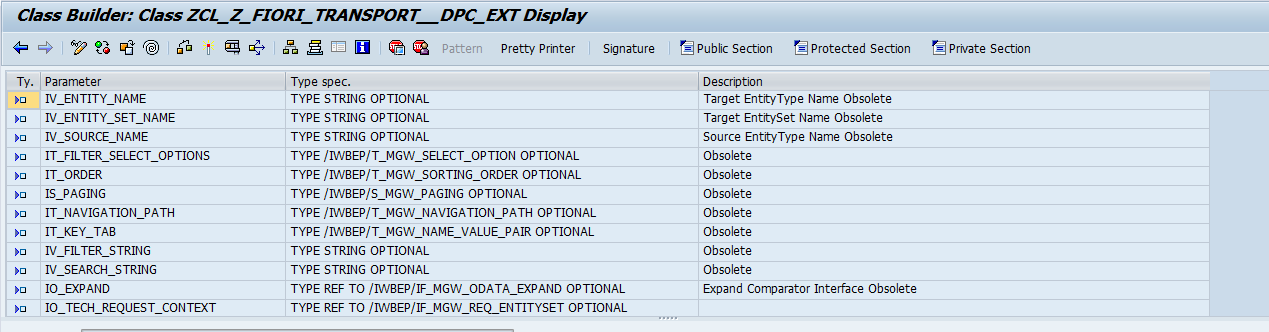


METHOD /iwbep/if\_mgw\_appl\_srv\_runtime~get\_expanded\_entity.  
  
  DATA:     lw\_return     TYPE bapiret2,  
            lt\_return     TYPE STANDARD TABLE OF bapiret2,  
            ls\_key\_tab    TYPE /iwbep/s\_mgw\_name\_value\_pair,  
            lo\_container  TYPE REF TO /iwbep/if\_message\_container.  
  
  CONSTANTS: lc\_e TYPE char01 VALUE 'E'.  
  
  IF iv\_entity\_set\_name = 'ZDRDDetHdrSet'.  
  
    TYPES    : BEGIN OF ty\_headitem.  
    INCLUDE        TYPE   zcl\_z\_fiori\_es\_so\_crea\_mpc=>ts\_zdrddethdr.  
    TYPES    : zdrdhdritemnav TYPE   zso\_customer\_tt,  
               END OF ty\_headitem.  
  
  
    DATA     : lv\_customer   TYPE kunnr,  
               ls\_header     TYPE zso\_credit\_dtl\_st,  
               ls\_headitem   TYPE ty\_headitem,  
               ter\_ind       TYPE char01.  
  
    CONSTANTS : lc\_expand\_tech\_clause TYPE string VALUE 'ZDRDHDRITEMNAV'.  
  
    READ TABLE it\_key\_tab INTO ls\_key\_tab WITH KEY name = 'CustomerNum'.  
    lv\_customer = ls\_key\_tab-value.  
  
\*    CALL FUNCTION 'ZSO\_FETCH\_ZDRD'  
\*      EXPORTING  
\*        i\_customer = lv\_customer  
\*      IMPORTING  
\*        et\_return  = lt\_return  
\*        et\_output  = ls\_headitem-zdrdhdritemnav  
\*        es\_header  = ls\_header  
\*        TERMINATION\_IND = ter\_ind.  
    CALL FUNCTION 'ZSO\_FETCH\_ZDRD'  
      EXPORTING  
        i\_customer      = lv\_customer  
      IMPORTING  
        et\_return       = lt\_return  
        et\_output       = ls\_headitem-zdrdhdritemnav  
        es\_header       = ls\_header  
        termination\_ind = ter\_ind.  
  
  
    ls\_headitem-agt\_code       = ls\_header-agt\_code.  
    ls\_headitem-avl\_credit     = ls\_header-avl\_credit.  
    ls\_headitem-currency       = ls\_header-currency.  
    ls\_headitem-date           = ls\_header-date.  
    ls\_headitem-i\_customer     = ls\_header-customer.  
    ls\_headitem-price\_grp      = ls\_header-price\_grp.  
    ls\_headitem-price\_grp\_desc = ls\_header-price\_grp\_desc.  
    ls\_headitem-total\_credit   = ls\_header-total\_credit.  
    ls\_headitem-agt\_ind        = ls\_header-agt\_ind.  
    ls\_headitem-termination\_ind        = ter\_ind.  " ankit on 26-06-18  
  
    IF lt\_return IS NOT INITIAL.  
      lo\_container = me->mo\_context->get\_message\_container( ).  
      CLEAR lw\_return.  
      LOOP AT lt\_return INTO lw\_return WHERE type = lc\_e.  
        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.  
    ENDIF.  
  
    copy\_data\_to\_ref(    EXPORTING is\_data = ls\_headitem  
                         CHANGING cr\_data  = er\_entity ).  
  
    INSERT lc\_expand\_tech\_clause INTO TABLE et\_expanded\_tech\_clauses.

Endif.

Endmethod.

GET\_EXPANDED\_ENTITYSET:



Code:

METHOD /iwbep/if\_mgw\_appl\_srv\_runtime~get\_expanded\_entityset.  
  
  TYPES BEGIN OF ty\_schdule.  
          INCLUDE TYPE zcl\_z\_fiori\_transport\_\_mpc=>ts\_schedulelist.  
  TYPES: schdeliverynav TYPE STANDARD TABLE OF zcl\_z\_fiori\_transport\_\_mpc=>ts\_deliverylist WITH NON-UNIQUE DEFAULT KEY,  
  END OF ty\_schdule.  
  
  DATA: ls\_schdule TYPE ty\_schdule,  
        lt\_schdule TYPE TABLE OF ty\_schdule,  
        lt\_final TYPE ztt\_tr\_courier\_track1,  
        lw\_schdule TYPE zst\_tr\_courier\_track1,  
        lt\_delivery TYPE ztt\_del\_det,  
        lw\_delivery TYPE zst\_del\_det,  
        ls\_delivery TYPE zcl\_z\_fiori\_transport\_\_mpc=>ts\_deliverylist.  
  
  DATA: ls\_filter TYPE /iwbep/s\_mgw\_select\_option,  
        ls\_filter\_range TYPE /iwbep/s\_cod\_select\_option,  
        lt\_date TYPE efg\_tab\_ranges,  
        ls\_date LIKE LINE OF lt\_date,  
        lv\_fromdate TYPE tvarv\_val,  
        lv\_todate TYPE tvarv\_val,  
        lv\_mode TYPE char3.  
  
  DATA:      lw\_return     TYPE bapiret2,  
             lt\_return     TYPE STANDARD TABLE OF bapiret2,  
             ls\_key\_tab    TYPE /iwbep/s\_mgw\_name\_value\_pair,  
             lo\_container  TYPE REF TO /iwbep/if\_message\_container.  
  
  CONSTANTS: lc\_expanded\_tech\_clauses TYPE string VALUE 'SCHDELIVERYNAV',  
             lc\_e TYPE char01 VALUE 'E'.  
  
  LOOP AT it\_filter\_select\_options INTO ls\_filter.  
    CASE ls\_filter-property.  
  
      WHEN 'FromDate'.  
\*        loop at ls\_filter-select\_options into  
        READ TABLE ls\_filter-select\_options INTO ls\_filter\_range INDEX 1.  
        IF sy-subrc = 0.  
          lv\_fromdate = ls\_filter\_range-low.  
        ENDIF.  
  
      WHEN 'ToDate'.  
\*        loop at ls\_filter-select\_options into  
        READ TABLE ls\_filter-select\_options INTO ls\_filter\_range INDEX 1.  
        IF sy-subrc = 0.  
          lv\_todate = ls\_filter\_range-low.  
        ENDIF.  
  
      WHEN 'Mode'.  
\*        loop at ls\_filter-select\_options into  
        READ TABLE ls\_filter-select\_options INTO ls\_filter\_range INDEX 1.  
        IF sy-subrc = 0.  
          lv\_mode = ls\_filter\_range-low.  
        ENDIF.  
    ENDCASE.  
  ENDLOOP.  
  
  IF lv\_fromdate IS NOT INITIAL.  
    ls\_date-sign = 'I'.  
    ls\_date-option = 'BT'.  
    ls\_date-low = lv\_fromdate.  
    ls\_date-high = lv\_todate.  
    APPEND ls\_date TO lt\_date.  
    CLEAR ls\_date.  
  ENDIF.  
  
  
  
  CALL FUNCTION 'ZTR\_SCH\_LIST'  
    EXPORTING  
      s\_date      = lt\_date  
\*     S\_STO       =  
\*     S\_VBELN     =  
\*     S\_STN       =  
      iv\_mode     = lv\_mode  
    IMPORTING  
      lt\_final    = lt\_final  
\*     LT\_F4       =  
      lt\_delivery = lt\_delivery  
      et\_return   = lt\_return.  
  
  
\*  SORT lt\_final BY vbelnd.  
\*  DELETE ADJACENT DUPLICATES FROM lt\_final COMPARING vbelnd.  
  
  LOOP AT  lt\_final INTO lw\_schdule.  
    ls\_schdule-vbeln = lw\_schdule-vbeln.  
    ls\_schdule-posnr = lw\_schdule-posnr.  
    ls\_schdule-vbelnd = lw\_schdule-vbelnd.  
    ls\_schdule-zshtyp = lw\_schdule-zshtyp.  
    ls\_schdule-zshtyp\_descr = lw\_schdule-zshtyp\_descr.  
    ls\_schdule-confdat = lw\_schdule-confdat.  
    ls\_schdule-sysid = lw\_schdule-sysid.  
    ls\_schdule-grpid = lw\_schdule-grpid.  
    ls\_schdule-route = lw\_schdule-route.  
    ls\_schdule-stageseq = lw\_schdule-stageseq.  
    ls\_schdule-kunnr = lw\_schdule-kunnr.  
    ls\_schdule-name1 = lw\_schdule-name1.  
    ls\_schdule-zzdropseq = lw\_schdule-zzdropseq.  
    ls\_schdule-stn\_no = lw\_schdule-stn\_no.  
    ls\_schdule-city = lw\_schdule-city.  
    ls\_schdule-pgi\_status = lw\_schdule-pgi\_status.  
    ls\_schdule-pstyv = lw\_schdule-pstyv.  
    ls\_schdule-werks = lw\_schdule-werks.  
    ls\_schdule-vstel = lw\_schdule-vstel.  
    ls\_schdule-werks1 = lw\_schdule-werks1.  
    ls\_schdule-inco1 = lw\_schdule-inco1.  
    ls\_schdule-type = lw\_schdule-type.  
    ls\_schdule-btgew = lw\_schdule-btgew.  
    ls\_schdule-ntgew = lw\_schdule-ntgew.  
    ls\_schdule-gewei = lw\_schdule-gewei.  
    ls\_schdule-kwmeng = lw\_schdule-kwmeng.  
    ls\_schdule-meins = lw\_schdule-meins.  
    ls\_schdule-xblnr = lw\_schdule-xblnr.        "++Pavan on 12-02-2019  
  
  
    LOOP AT lt\_delivery INTO lw\_delivery WHERE vbelnd = lw\_schdule-vbelnd.  
\*READ TABLE lt\_delivery into lw\_delivery with key vbelnd = lw\_schdule-vbelnd.  
  
      ls\_delivery-vbelnd = lw\_delivery-vbelnd.  
      ls\_delivery-posnrd = lw\_delivery-posnrd.  
      ls\_delivery-kwmeng = lw\_delivery-kwmeng.  
      ls\_delivery-meins = lw\_delivery-meins.  
      ls\_delivery-maktx = lw\_delivery-maktx.  
      APPEND ls\_delivery TO ls\_schdule-schdeliverynav.  
\*      DELETE lt\_delivery where vbelnd = ls\_delivery-vbelnd and posnrd = ls\_delivery-posnrd.  
      CLEAR: ls\_delivery, lw\_delivery.  
    ENDLOOP.  
  
  
    APPEND ls\_schdule TO lt\_schdule.  
    CLEAR: ls\_schdule, lw\_schdule.  
  ENDLOOP.  
  
\*  SORT lt\_schdule BY grpid vbelnd.  
  
  IF lt\_return IS NOT INITIAL.  
    lo\_container = me->mo\_context->get\_message\_container( ).  
    CLEAR lw\_return.  
    LOOP AT lt\_return INTO lw\_return WHERE type = lc\_e.  
      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.  
  ENDIF.  
  
  copy\_data\_to\_ref( EXPORTING is\_data = lt\_schdule  
                  CHANGING cr\_data = er\_entityset ).  
  
  INSERT lc\_expanded\_tech\_clauses INTO TABLE et\_expanded\_tech\_clauses.  
  
ENDMETHOD.