

CRUD Application with BOPF and DRAFT functionality

- 1) Create the tables
 - a. Sales Order Header

| Field Name | Key | Data Element |
|-----------------|-----|---------------------------|
| MANDT | X | MANDT |
| SALESORDER | X | SNWD_SO_ID |
| BUSINESSPARTNER | | SNWD_PARTNER_ID |
| OVERALLSTATUS | | SNWD_SO_OA_STATUS_CODE |
| CHANGEDAT | | FARP_PRQ_CHG_DATE |
| CREATEDAT | | FARP_PRQ_CR_DATE |
| CHANGEDBY | | SATC_D_AC_LAST_CHANGED_BY |
| CREATEDBY | | SATC_D_CREATED_BY |

Dictionary: Change Table

Transparent Table: ZTSO_H Active

Short Description: Sales Order Header

Attributes Delivery and Maintenance Fields Input Help/Check Currency/Quantity Fields Indexes

Search Built-In Type 8

| Field | Key | Ini... | Data element | Data Type | Length | Deci... | Short Description |
|-----------------|-------------------------------------|-------------------------------------|---------------------------|-----------|--------|---------|--|
| MANDT | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | MANDT | CLNT | 3 | 0 | Client |
| SALESORDER | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | SNWD_SO_ID | CHAR | 10 | 0 | EPM: Sales Order Number |
| BUSINESSPARTNER | <input type="checkbox"/> | <input type="checkbox"/> | SNWD_PARTNER_ID | CHAR | 10 | 0 | EPM: Business Partner ID |
| OVERALLSTATUS | <input type="checkbox"/> | <input type="checkbox"/> | SNWD_SO_OA_STATUS_CODE | CHAR | 1 | 0 | EPM: Sales Order Overall Status |
| CHANGEDAT | <input type="checkbox"/> | <input type="checkbox"/> | FARP_PRQ_CHG_DATE | DEC | 21 | 7 | Change Date and Time |
| CREATEDAT | <input type="checkbox"/> | <input type="checkbox"/> | FARP_PRQ_CR_DATE | DEC | 21 | 7 | Creation Date and Time |
| CHANGEDBY | <input type="checkbox"/> | <input type="checkbox"/> | SATC_D_AC_LAST_CHANGED_BY | CHAR | 12 | 0 | Last editor of the object or subobject |
| CREATEDBY | <input type="checkbox"/> | <input type="checkbox"/> | SATC_D_CREATED_BY | CHAR | 12 | 0 | Created by |

- b. Sales Order Item

| Field Name | Key | Data Element |
|----------------|-----|-----------------------|
| MANDT | X | MANDT |
| SALESORDER | X | SNWD_SO_ID |
| SALESORDERITEM | X | SNWD_SO_ITEM_POS |
| PRODUCT | | SNWD_PRODUCT_ID |
| GROSSAMOUNT | | SNWD_TTL_GROSS_AMOUNT |
| CURRENCYCODE | | SNWD_CURR_CODE |
| QUANTITY | | EDESK_QUANT |

Dictionary: Change Table

Transparent Table: Active

Short Description:

Attributes | Delivery and Maintenance | **Fields** | Input Help/Check | Currency/Quantity Fields | Indexes

Search Built-In Type 7

| Field | Key | Ini... | Data element | Data Type | Length | Deci... | Short Description |
|-----------------------|-------------------------------------|-------------------------------------|------------------------------|-----------|--------|---------|--------------------------------|
| <u>MANDT</u> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <u>MANDT</u> | CLNT | 3 | 0 | Client |
| <u>SALESORDER</u> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <u>SNWD SO ID</u> | CHAR | 10 | 0 | EPM: Sales Order Number |
| <u>SALESORDERITEM</u> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <u>SNWD SO ITEM_POS</u> | CHAR | 10 | 0 | EPM: Sales Order Item Position |
| <u>PRODUCT</u> | <input type="checkbox"/> | <input type="checkbox"/> | <u>SNWD PRODUCT_ID</u> | CHAR | 10 | 0 | EPM: Product ID |
| <u>GROSSAMOUNT</u> | <input type="checkbox"/> | <input type="checkbox"/> | <u>SNWD TTL GROSS AMOUNT</u> | CURR | 15 | 2 | EPM: Total Gross Amount |
| <u>CURRENCYCODE</u> | <input type="checkbox"/> | <input type="checkbox"/> | <u>SNWD CURR_CODE</u> | CUKY | 5 | 0 | EPM: Currency Code |
| <u>QUANTITY</u> | <input type="checkbox"/> | <input type="checkbox"/> | <u>EDESK_QUANT</u> | INT4 | 10 | 0 | Quantity |

c. Create relationship with foreign key:

DS4(2)/100 Create Foreign Key ZTSO_I-SALESORDER

Short text:

Check table:

| Check table | Tabfield | For.key ta... | Foreign Key Field | Generic | Constant |
|-------------|------------|---------------|-------------------|--------------------------|----------|
| ZTSO_H | MANDT | ZTSO_I | MANDT | <input type="checkbox"/> | |
| ZTSO_H | SALESORDER | ZTSO_I | SALESORDER | <input type="checkbox"/> | |

Screen check

☒ Check required Error message MsgNo AArea

Semantic attributes

Foreign key field type

☐ Not Specified
☐ Non-key fields/candidates
☒ Key fields/candidates
☐ Key fields of a text table

Cardinality 1 :

- 2) Generate Number Object Range
 - a. Define the domain name and warning, e.g:

Number Range Object ZZ_IMU_SO: Create

ShortTxt: Sales Order
Long Txt: Sales Order

Intervals Customizing Group

Number Length Domain: D_EPM_ID
% Warning: 90,0
Subobject Data Element:

☐ To-business year flag
☐ No rolling

- b. Define the interval:

✓ [Dropdown] << [Icons]

Edit Intervals: Sales Order, Object ZZ_IM

[Icons]

| No | From No. | To Number | NR S |
|----|------------|------------|------|
| 01 | 0000000001 | 9999999999 | 0 |

- 3) Create BASIC CDS Views in Eclipse
 - a. CDS Basic View for Sales Order Header

```
@AbapCatalog.sqlViewName: 'ZBSOH'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Basic CDS View for ZTSO_H table'

@VDM.viewType: #BASIC
define view ZBSO_H
  as select from ztso_h
{
  key salesorder,
    businesspartner,
    overallstatus,
    changedat,
    createdat,
    changedby,
```

```
        createdby
    }
}
```

b. CDS Basic View for Sales Order Item

```
@AbapCatalog.sqlViewName: 'ZBSOI'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Basic CDS View for ZTSO_I table'

@VDM.viewType: #BASIC
define view ZBSO_I
    as select from ztso_i
    {
        key salesorder,
        key salesorderitem,
        product,
        grossamount,
        currencycode,
        quantity
    }
}
```

4) Create Transactional CDS Views in Eclipse

a. CDS Transactional View for Sales Order Header

```
@AbapCatalog.sqlViewName: 'ZISOHTP'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Transactional CDS View for ZBSO_H'

@VDM.viewType: #TRANSACTIONAL

//****---->>> Add the below objectModel code after activation of
// both header and items
@ObjectModel: {
    transactionalProcessingEnabled: true,
    compositionRoot: true,
    createEnabled: true,
    updateEnabled: true,
    deleteEnabled: true,
    draftEnabled: true,
    writeDraftPersistence: 'ZTSO_H_DRAFT',
    semanticKey: [ 'salesorder' ]
}
define view ZISO_H_TP
    as select from ZBSO_H as Header
    // Association
    association [0..*] to ZISO_I_TP as _Items
        on $projection.salesorder = _Items.salesorder
    // Value Help
```

```

association [0..*] to SEPM_I_BusinessPartner as _BP
on $projection.businesspartner = _BP.BusinessPartner
association [0..1] to Sepm_I_SalesOrdOverallStatus as _Status
on $projection.overallstatus = _Status.SalesOrderOverallStatus
{
    @ObjectModel.readOnly: true
    key Header.salesorder,
        // @ObjectModel.foreignKey.association: '_BP'
        Header.businesspartner,
        @ObjectModel.foreignKey.association: '_Status'
        Header.overallstatus,
        @ObjectModel.readOnly: true
        Header.changedat,
        @ObjectModel.readOnly: true
        Header.createdat,
        @ObjectModel.readOnly: true
        Header.changedby,
        @ObjectModel.readOnly: true
        Header.createdby,
        @ObjectModel.association: {
            type: [ #TO_COMPOSITION_CHILD ]
        }
        _Items,
        // Value help associations
        _Status,
        _BP
}

```

b. CDS Transactional View for Sales Order Item

```

@AbapCatalog.sqlViewName: 'ZISOITP'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Transactional CDS View for ZBSO_I'

@VDM.viewType: #TRANSACTIONAL
@ObjectModel:{
    createEnabled: true,
    updateEnabled: true,
    deleteEnabled: true,
    writeDraftPersistence: 'ZTSO_I_DRAFT',
    semanticKey:['salesorder', 'salesorderitem']
}
define view ZISO_I_TP
as select from ZBSO_I as Items
association [1..1] to ZISO_H_TP as _Header
on $projection.salesorder = _Header.salesorder
// Value Help
association [0..1] to SEPM_I_Product_E as _Product
on $projection.product = _Product.Product
association [0..1] to SEPM_I_Currency as _Currency
on $projection.currencycode = _Currency.Currency
{
    @ObjectModel.readOnly: true
    key Items.salesorder,
        @ObjectModel.readOnly: true

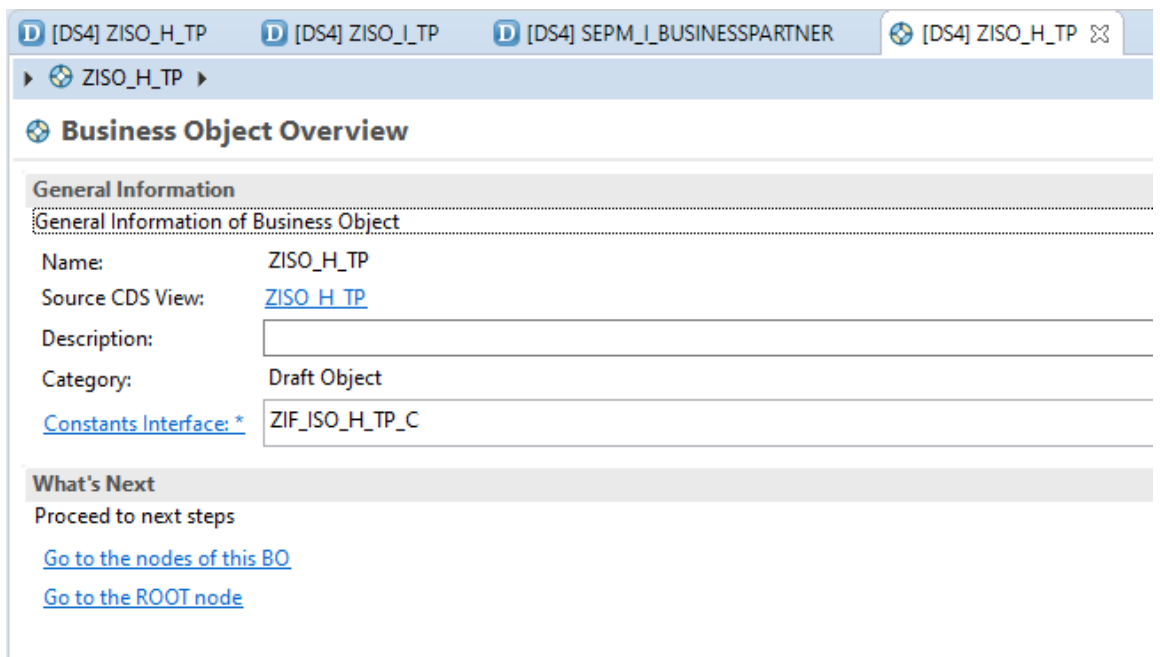
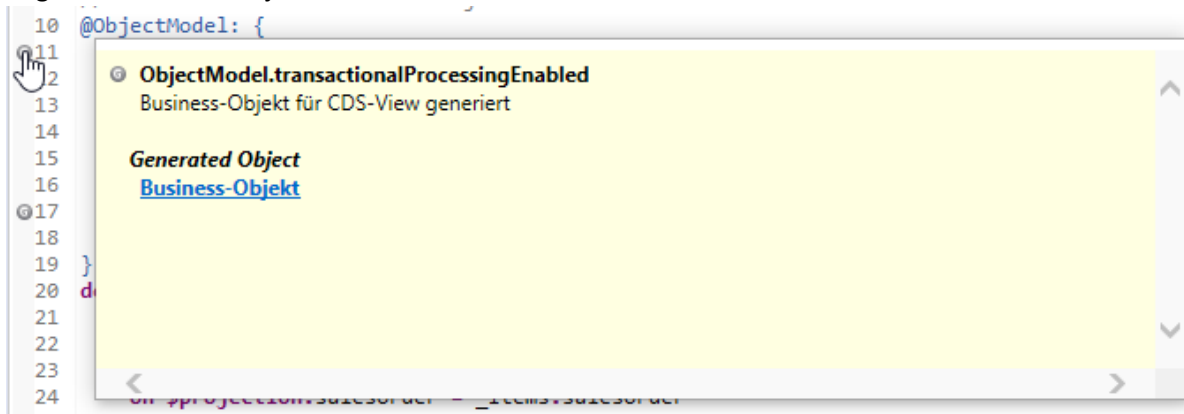
```

```

key Items.salesorderitem,
  @ObjectModel.mandatory: true
  @ObjectModel.foreignKey.association: '_Product'
  Items.product,
  @Semantics.amount.currencyCode: 'currencycode'
  Items.grossamount,
  @ObjectModel.foreignKey.association: '_currency'
  @Semantics.currencyCode: true
  Items.currencycode,
  Items.quantity,
  @ObjectModel.association: {
    type: [ #TO_COMPOSITION_PARENT, #TO_COMPOSITION_ROOT ]
  }
  _Header,
  _Product,
  _Currency
}

```

5) Check the generated BOPF objects and draft tables:



- ABAP Dictionary (22)
 - Datenbanktabellen (4)
 - ZTSO_H Sales Order Header
 - ZTSO_H_DRAFT ZISO_H_TP ZISO_H_TP
 - ZTSO_I Sales Order Item
 - ZTSO_I_DRAFT ZISO_H_TP ZISO_I_TP

- a. Go into the DRAFT class and implement the code to save the DRAFT data to the database tables:

Class Builder Class ZCL_DR_ISO_H_TP Change

Navigation icons: Back, Forward, Find, Replace, Copy, Paste, Undo, Redo, Print, etc.

| Ty. | Parameter | Typing | Description |
|-----|---------------------|-------------------------------------|--|
| | IS_CTX | TYPE /BOBF/S_FRW_CTX_DRAFT | BOPF draft context |
| | IT_DRAFT_KEY | TYPE /BOBF/T_FRW_KEY | Keys of the drafts to be copied |
| | IO_READ | TYPE REF TO /BOBF/IF_FRW_READ | BOPF access object for read access |
| | IO_MODIFY | TYPE REF TO /BOBF/IF_FRW_MODIFY | BOPF access object for modifications |
| | EO_MESSAGE | TYPE REF TO /BOBF/IF_FRW_MESSAGE | BOPF message container |
| | ET_KEY_LINK | TYPE /BOBF/T_FRW_KEY_LINK_ACT_DRAFT | (Active/Draft) key pairs from successful drafts |
| | ET_FAILED_DRAFT_KEY | TYPE /BOBF/T_FRW_KEY | Keys of the drafts for which the activation failed |

Method: active

```

1 method /BOBF/IF_FRW_DRAFT~COPY_DRAFT_TO_ACTIVE_ENTITY.
2   endmethod.
    
```

```

METHOD /bobf/if_frwr_draft~copy_draft_to_active_entity.

DATA: ls_msg TYPE symsg.

"Get the message container, to push any messages to the UI if required
IF eo_message IS NOT BOUND.
    eo_message = /bobf/cl_frwr_message_factory=>create_container( ).
ENDIF.

"Header and items data declaration( Generated after activation of the Transactional View by the BOPF)
DATA(lt_header) = VALUE ztiso_h_tp( ).
DATA(lt_items) = VALUE ztiso_i_tp( ).

"Read Header Data
"Pass the node key, which tells if we are trying to read header or item, in this case header
"Pass the draft keys, if we create an entry in the Fiori, they will be saved as draft initially, those keys we will pass to get the data
"Use the Constant Interface generated by your BOPF
io_read->retrieve(
    EXPORTING
        iv_node          = zif_iso_h_tp_c=>sc_node-ziso_h_tp
        it_key            = it_draft_key
        iv_fill_data     = abap_true
    IMPORTING
        et_data          = lt_header ).

" Read the Items data
" We need to read via the association, so passing the association key here

```

```

io_read->retrieve_by_association(
    EXPORTING
        iv_node                = zif_iso_h_tp_c=>sc_node-
ziso_h_tp " Node Name
        it_key                  = it_draft_key
Key Table
        iv_association         = zif_iso_h_tp_c=>sc_association-
ziso_h_tp-items
        iv_fill_data = abap_true
    IMPORTING
        et_data                = lt_items ).

" Always 1 header only expected as we can only create one sales o
rder at a time
DATA(ls_so_header) = CORRESPONDING ztso_h( lt_header[ 1 ] ). "Use
the structure created by the generation of the BOPF
" Updating the header
GET TIME STAMP FIELD ls_so_header-changedat.
ls_so_header-changedby = sy-uname.
" I am checking if the sales order number is initial, it means it
is the new entry
" Else it is an existing entry
" Alternatively we can also use lt_header[ 1 ]-
HASACTIVEENTRY, which will tell if the active entry is available
" If the active entry is available means it is update sceanrio el
se it is create scenario
IF ls_so_header-salesorder IS NOT INITIAL.
    MODIFY ztso_h FROM ls_so_header.
ELSE.
    GET TIME STAMP FIELD ls_so_header-createdat.
    ls_so_header-createdby = sy-uname.

    CALL FUNCTION 'NUMBER_GET_NEXT'
        EXPORTING
            nr_range_nr = '01'
            object       = 'ZZ_IMU_SO'
        IMPORTING
            number       = ls_so_header-salesorder.

    INSERT ztso_h FROM ls_so_header.
ENDIF.

" Now update/create/delete the items
DATA: lt_items_save TYPE TABLE OF ztso_i,
      lt_ztso_i_aux TYPE TABLE OF ztso_i,
      lv_count      TYPE i.

" Deleting the items
SELECT *
    FROM ztso_i
    INTO TABLE @DATA(lt_ztso_i)
    WHERE salesorder = @ls_so_header-
salesorder ORDER BY salesorderitem DESCENDING.

LOOP AT lt_ztso_i ASSIGNING FIELD-SYMBOL(<fs_ztso_i>).
    DATA(lv_tabix) = sy-tabix.
    READ TABLE lt_items WITH KEY salesorder      = <fs_ztso_i>-
salesorder
                                salesorderitem = <fs_ztso_i>-
salesorderitem
                                hasactiveentity = abap_true TRANSP
ORTING NO FIELDS.

```



```

        IF sy-subrc <> 0.
            INSERT <fs_ztso_i> INTO TABLE lt_ztso_i_aux.
            DELETE lt_ztso_i INDEX lv_tabix.
        ENDIF.
    ENDLOOP.

    IF lt_ztso_i_aux IS NOT INITIAL.
        DELETE ztso_i FROM TABLE lt_ztso_i_aux.
    ENDIF.

    " Updating the items
    READ TABLE lt_ztso_i ASSIGNING <fs_ztso_i> INDEX 1. "Get last item
    IF sy-subrc = 0.
        lv_count = <fs_ztso_i>-salesorderitem.
    ENDIF.
    " Setting the item number here.
    LOOP AT lt_items REFERENCE INTO DATA(lo_item) WHERE hasactiveentity EQ abap_false.
        lv_count = lv_count + 1.
        lo_item->salesorder = ls_so_header-salesorder.
        lo_item->salesorderitem = lv_count.
        lo_item->salesorderitem = |{ lo_item->salesorderitem ALPHA = IN }|.
    ENDLOOP.

    lt_items_save = CORRESPONDING #( lt_items ).
    IF lt_items_save IS NOT INITIAL.
        MODIFY ztso_i FROM TABLE lt_items_save.
    ENDIF.

    " Now we need to tell the BOPF framework to delete the draft entries as we have successfully created the data in the DB
    " But BOPF only understands the data in GUIDs, so we need to convert the sales order number to BOPF key, we just need to parent key
    DATA(lr_key_util) = /bobf/cl_lib_legacy_key=>get_instance( zif_iso_h_tp_c=>sc_bo_key ).

    DATA(lv_bobf_key) = lr_key_util->convert_legacy_to_bopf_key( iv_node_key = zif_iso_h_tp_c=>sc_node-ziso_h_tp
                                                                                                     is_legacy_key = ls_so_header-salesorder ).

    APPEND VALUE #( draft = it_draft_key[ 1 ]-key active = lv_bobf_key ) TO et_key_link.
ENDMETHOD.

```

- b. By creating the new item in the UI, we need to link the item with the header, this needs to be done manually through BOPF Determination, so open the Item node and create the Determination:

Determinations

New...

Delete

type filter text

| Name | Implementation Class | Category | Triggers |
|--------------------------|--------------------------------|----------------------|-----------------------|
| ACTION_AND_FIELD_CONTROL | ZCL_ISO_TP_ACTION_AND_FIEL | Calculate properties | Triggers not required |
| DRAFT_SYS_ADMIN_DATA | /BOBF/CL_LIB_D_SET_DRAFTSYSADM | React during save | Triggers configured |
| DRAFT_ACTION_CONTROL | /BOBF/CL_LIB_D_SET_DRAFTSYSADM | | |

New Determination

General Attributes

Enter the general attributes for this determination

Name: * SET_SEMATIC_KEY
 Description: Determine Sales Order Number
 Implementation Class: * ZCL_ISO_D_SET_SEMATIC_KEY Browse...
 Category: React after modification



Finish

Cancel

Determination Overview

General Information

General Information of Determination

Name: * SET_SEMATIC_KEY
 Description: Determine Sales Order Number
 Implementation Class: * ZCL_ISO_D_SET_SEMATIC_KEY
 Category: React after modification

Triggers

Information on Triggers

New

Delete

type filter text

| Node | Association | Create | Update | Delete | Load | Determine |
|-----------|-------------|-------------------------------------|-------------------------------------|--------------------------|--------------------------|--------------------------|
| ZISO_I_TP | | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |

Open the generated implementation class ZCL_ISO_D_SET_SEMATIC_KEY, and do the code:

```
METHOD /bobf/if_fr_w_determination~execute.
```

```
" This is used in the update scenario of the sales order (which has already been created) and when we are creating sales order items
```

```
DATA: lt_header TYPE ztiso_h_tp,  
      lt_items  TYPE ztiso_i_tp.  
" read the sales order items  
io_read->retrieve(  
  EXPORTING  
    iv_node           = zif_iso_h_tp_c=>sc_node-  
ziso_i_tp  
    it_key            = it_key  
  IMPORTING  
    et_data           = lt_items ).
```

```
" Obviously one item is only expected, check if the sales o
```

```

order number is initial for the draft entry
  READ TABLE lt_items REFERENCE INTO DATA(lo_item) INDEX 1.
  IF lo_item->salesorder IS INITIAL.
    " If initial, then get the header draft entry, which will
    have the sales order number via the association from child to
    header
    io_read->retrieve_by_association(
      EXPORTING
        iv_node           = zif_iso_h_tp_c=>sc_node-
ziso_i_tp
        it_key            = it_key              " Key
y Table
        iv_association    = zif_iso_h_tp_c=>sc_associat
ion-ziso_i_tp-to_root
        iv_fill_data      = abap_true
      IMPORTING
        et_data           = lt_header ).
    " Update the salesorder item with the so number
    lo_item->salesorder = lt_header[ key = lo_item-
>parent_key ]-salesorder.
    io_modify->update(
      EXPORTING
        iv_node           = zif_iso_h_tp_c=>sc_node-ziso_i_tp
        iv_key            = lo_item->key
        iv_root_key       = lo_item->root_key
        is_data           = lo_item ).
  ENDIF.
ENDMETHOD.

```

6) Now lets create the code to do the deletion. We will do it through an BOPF Action:

a. Code the Action in the root node:

type filter text

| Name | Implementation Class | Instance Multiplicity | Parameter Structure | Exporting Type |
|------------------|----------------------------|-------------------------|--------------------------------|----------------|
| LOCK_ZISO_H_TP | ZCL_A_LOCK_ISO_H_TP | Multiple Node Instances | /BOBF/S_FRW_LOCK_PARAMETERS | None |
| DELETE_ZISO_H_TP | ZCL_A_DELETE_ISO_H_TP | Multiple Node Instances | | None |
| EDIT | /BOBF/CL_LIB_A_EDIT | Single Node Instance | /BOBF/S_LIB_A_DRAFT_EDIT_IMP | Node |
| ACTIVATION | /BOBF/CL_LIB_A_ACTIVATION | Single Node Instance | | Node |
| VALIDATION | /BOBF/CL_LIB_A_VALIDATION | Single Node Instance | /BOBF/S_LIB_A_IN_DRAFT_VALIDAT | Type |
| PREPARATION | /BOBF/CL_LIB_A_PREPARATION | Single Node Instance | /BOBF/S_LIB_A_IN_DRAFT_PREPARE | Node |

b. Code for the method delete_active_entity:

```

METHOD /bobf/if_lib_delete_active~delete_active_entity.

  " To get only the active data as this code wil trigger for
  draft deletion as well
  " For draft deletion the framework will take care, for acti
  ve, we need to code
  /bobf/cl_lib_draft=>/bobf/if_lib_union_utilities~separate_k
  eys(
    EXPORTING iv_bo_key   = is_ctx-bo_key
              iv_node_key = is_ctx-node_key
              it_key      = it_key

    IMPORTING
      et_active_key = DATA(lt_active_bopf_keys) ).
  " We will get the data in BOPF keys format, we need to get
  the sales order number from that
  CHECK lt_active_bopf_keys IS NOT INITIAL.

```

```

DATA(ls_header) = VALUE zsk_iso_h_tp_active_entity_key( ).
" Convert the bopf key to active document key
/bobf/cl_lib_draft=>/bobf/if_lib_union_utilities~get_active
_document_key(
    EXPORTING iv_bo_key    = is_ctx-bo_key
              iv_node_key  = is_ctx-node_key
              iv_bopf_key  = lt_active_bopf_keys[ 1 ]-key
    IMPORTING es_active_document_key = ls_header ).
" Delete the data
DELETE FROM ztso_h WHERE salesorder = ls_header-
salesorder.
DELETE FROM ztso_i WHERE salesorder = ls_header-
salesorder.

ENDMETHOD.

```

7) Lets create now the CDS Consumption Views:

a. Header Consumption View

```

@AbapCatalog.sqlViewName: 'ZCSOH'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Consumption CDS View for ZISO_H_TP'

@VDM.viewType: #CONSUMPTION

@Metadata.allowExtensions: true

@ObjectModel.transactionalProcessingDelegated: true
@ObjectModel: {
    compositionRoot: true,
    createEnabled: true,
    updateEnabled: true,
    deleteEnabled: true,
    draftEnabled: true
}

@ObjectModel.semanticKey: [ 'salesorder' ]

@OData.publish: true
//@UI.headerInfo: {
//    typeName: 'Sales Order',
//    typeNamePlural: 'Sales Order List'
//}

@UI: {
    headerInfo: {
        typeName: 'Sales Order',
        typeNamePlural: 'Sales Order List'
    }
}

define view ZCSO_H
as select from ZISO_H_TP as Header
association [0..*] to ZCSO_I as _Items
on $projection.salesorder = _Items.salesorder
{
    //Header
    @UI.selectionField: [{ position: 10 }]
}

```

```

        @UI.lineItem.position: 10
        @UI.identification: [{ position: 10 }]
    key Header.salesorder,
        @UI.lineItem.position: 20
        @UI.identification: [{ position: 20 }]
        @Consumption.valueHelp: '_BP'
        Header.businesspartner,
        @UI.lineItem.position: 30
        @UI.identification: [{ position: 30 }]
        Header.overallstatus,
        @UI.lineItem.position: 40
        @UI.identification: [{ position: 40 }]
        Header.changedat,
        @UI.lineItem.position: 50
        @UI.identification: [{ position: 50 }]
        Header.createdat,
        @UI.lineItem.position: 60
        @UI.identification: [{ position: 60 }]
        Header.changedby,
        @UI.lineItem.position: 70
        @UI.identification: [{ position: 70 }]
        Header.createdby,
        /* Associations */
        //Header
        //    _BP,
        @ObjectModel.association.type: [ #TO_COMPOSITION_CHILD ]
        _Items,
        Header._BP,
        Header._Status
        //    _Status
}

```

b. Item Consumption View

```

@AbapCatalog.sqlViewName: 'ZCSOI'
@AbapCatalog.compiler.compareFilter: true
@AbapCatalog.preserveKey: true
@AccessControl.authorizationCheck: #CHECK
@EndUserText.label: 'Consumption CDS View for ZISO_I_TP'

@VDM.viewType: #CONSUMPTION
@Metadata.allowExtensions: true

@ObjectModel: {
    semanticKey:['salesorder', 'salesorderitem'],

    createEnabled: true,
    deleteEnabled: true,
    updateEnabled: true
}
define view ZCSO_I
as select from ZISO_I_TP as Items
association [1..1] to ZCSO_H as _Header
on $projection.salesorder = _Header.salesorder
{
    //Items
    @UI.hidden: true
    key Items.salesorder,

```

```

    @UI.identification.position: 10
    @UI.lineItem.position: 10
    key Items.salesorderitem,
    @UI.identification.position: 20
    @UI.lineItem.position: 20
    Items.product,
    @UI.identification.position: 30
    @UI.lineItem.position: 30
    Items.grossamount,
    @UI.hidden: true
    Items.currencycode,
    @UI.identification.position: 40
    @UI.lineItem.position: 40
    Items.quantity,
    /* Associations */
    @ObjectModel.association.type: [#TO_COMPOSITION_PARENT,
    #TO_COMPOSITION_ROOT]
    _Header,
    Items._Product,
    Items._Currency
}

```

- 8) Activate the service at transaction /IWFND/MAINT_SERVICE.

Services aktivieren und verwalten

Katalog aktualisier. | OAuth | Soft-State | Verarbeitungsmodus | Zu Transport hinzu

Servicekatalog

| Typ | Techn. Servicename | Ver... | Servicebeschreibung |
|-----|---------------------------|--------|--|
| BEP | ZGW_REPAIRSERVICE_KKE_SRV | 1 | GW Service für den Reparatur-Service KKE |

Ausgewählte Services hinzufügen

Services abrufen


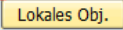
Filter

Systemalias: LOCAL ☐ Int.C
 Technischer Servicename: ZCSO_H* ☐ Version
 Externer Servicename: ☐ Extern

Ausgewählte Backend-Services

| Typ | Techn. Servicename | Ver... | Servicebeschreibung |
|-----|--------------------|--------|----------------------------------|
| BEP | ZCSO_H_CDS | 1 | Consumption CDS View for ZISO_H_ |

DS4(1)/100 Service hinzufügen

| | |
|--|---|
| Service | |
| Technischer Servicename | ZCSO_H_CDS |
| Serviceversion | 1 |
| Beschreibung | Consumption CDS View for ZISO_H_IP |
| Externer Servicename | ZCSO_H_CDS |
| Namensraum | |
| Externe Zuordnungs-ID | |
| Externer Datenquellentyp | C |
| Modell | |
| Technischer Modellname | ZCSO_H_CDS |
| Modellversion | 1 |
| Informationen zum Anlegen | |
| Paketzuordnung | ZIMU_BOPF_DRAFT  |
|  | |
| ICF-Knoten | |
| <input checked="" type="radio"/> Standardmodus <input type="radio"/> Keine | |
| <input checked="" type="checkbox"/> Akt.Mandant als StndMandant in ICF-Knoten einricht | |
| OAuth-Aktivierung | |
| <input type="checkbox"/> OAuth für Service aktiv. | |

9) Create List Report Template through SAP Web IDE.

New List Report Application

Data Connection

Service: ZCSO_H_CDS is selected.

Choose a service from one of the sources listed below.

Sources

Service Catalog

Workspace
File System
Service URL

Choose a system, explore, and select a service. Optionally, you can drill down into the service details.

S4_On_Premise - S4_On_Premise

Services



Show Details

| Service | Description | System |
|--|----------------------------|---------------|
| >  ZCSO_H_CDS | Consumption CDS View for Z | S4_On_Premise |

New List Report Application

Annotation Selection

Select the desired annotation files and rank them in the order in which they will be loaded.
Note: If the annotation files overlap, the one loaded last will overwrite the previous ones.
The selected service contains annotation data.

+ Add Annotation Files

| <input checked="" type="checkbox"/> | Rank | Name | Source | |
|-------------------------------------|------|---------------------------|--------|--|
| <input checked="" type="checkbox"/> | 1 | Selected Service Metadata | Remote | |
| <input checked="" type="checkbox"/> | 2 | ZCSO_H_CDS_VAN | Remote | |
| | | | | |
| | | | | |
| | | | | |

Template Selection
Basic Information
Data Connection
Annotation Selection
Template Customization
Confirmation

New List Report Application

Template Customization

Data Binding

OData Collection*
ZCSO_H

OData Navigation
to_Items

OData Sub Navigation
OData navigation attribute to a collection of items on sul

☒ Smart Variant Management for List Report

☐ Flexible Column Layout

Previous
Next
Finish

50
"icons": {

10) Test the application

a. Adding data:

SAP
Sales Order BOFF Draft

Standard

Bearbeitungsstatus:
Alles

Kundenauftr.-ID:

Filter anpassen (1)
Start

| Kundenauftr.-ID | Geschäftspartner-ID | Gesamtstatus | Änderungsdatum | Angelegt am | Geändert von | Angelegt von | Löschen |
|--|---------------------|--------------|----------------|-------------|--------------|--------------|---------|
| Keine Elemente gefunden. Überprüfen Sie Ihre Such- und Filtereinstellungen. | | | | | | | |

SAP

Sales Order

Sales Order Header Information

Sales Order Items

Sales Order Header Information

Kundenauftr.-ID:

Geschäftspartner-ID:

100000002

Gesamtstatus:

D

Änderungsdatum:

Angelegt am:

Geändert von:

Angelegt von:

Sales Order Items

Löschen

+

| Zeile der Position | Produkt-ID | Bruttobetrag | Menge |
|--|------------|--------------|-------|
| Keine Elemente gefunden. Überprüfen Sie Ihre Such- und Filtereinstellungen. | | | |

Sales Order Items

Löschen

+

| Zeile der Position | Produkt-ID | Bruttobetrag | Menge |
|--|------------|--------------|-------|
| Keine Elemente gefunden. Überprüfen Sie Ihre Such- und Filtereinstellungen. | | | |

Sales Order /

Sales Order Header Information

Sales Order Header Information

Zeile der Position:

*Produkt-ID:

HT-1068

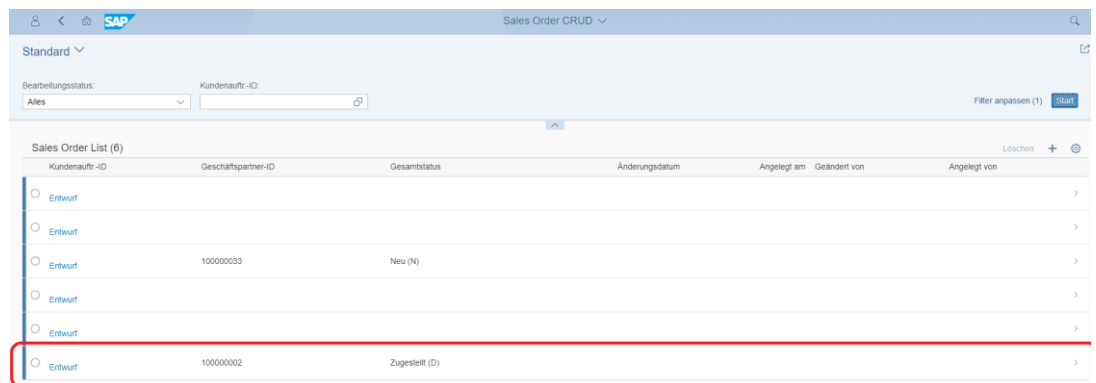
Bruttobetrag:

4,00

Menge:

3

- b. Close the Browser without saving and open again the app:



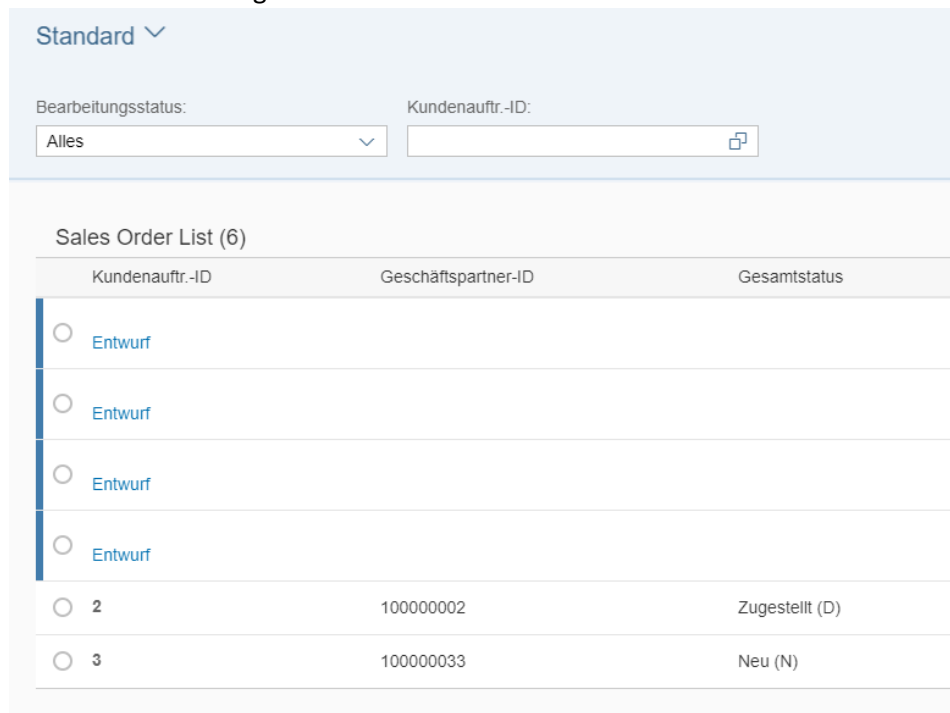
The screenshot shows the SAP Sales Order CRUD app interface. At the top, there's a header with the SAP logo and a search icon. Below the header, there's a filter section with 'Standard' selected, a dropdown for 'Bearbeitungsstatus' set to 'Alles', and a text input for 'Kundenauftr.-ID'. A 'Filter anpassen (1)' button and a 'Start' button are also present. The main area displays a table titled 'Sales Order List (6)'. The table has columns: 'Kundenauftr.-ID', 'Geschäftspartner-ID', 'Gesamtstatus', 'Änderungsdatum', 'Angelegt am', 'Geändert von', and 'Angelegt von'. There are six rows, all with 'Entwurf' status. The last row, with 'Kundenauftr.-ID' 100000002 and 'Gesamtstatus' 'Zugestellt (D)', is highlighted with a red rectangle.

| Kundenauftr.-ID | Geschäftspartner-ID | Gesamtstatus | Änderungsdatum | Angelegt am | Geändert von | Angelegt von |
|-----------------|---------------------|----------------|----------------|-------------|--------------|--------------|
| Entwurf | | | | | | |
| Entwurf | | | | | | |
| Entwurf | 100000003 | Neu (N) | | | | |
| Entwurf | | | | | | |
| Entwurf | | | | | | |
| Entwurf | 100000002 | Zugestellt (D) | | | | |

- c. Open the Draft and then save it:



- d. Check the final data generated:



The screenshot shows the same SAP Sales Order CRUD app interface as before. The 'Sales Order List (6)' table now has 6 rows. The first four rows are 'Entwurf'. The fifth row has 'Kundenauftr.-ID' 2, 'Geschäftspartner-ID' 100000002, and 'Gesamtstatus' 'Zugestellt (D)'. The sixth row has 'Kundenauftr.-ID' 3, 'Geschäftspartner-ID' 100000033, and 'Gesamtstatus' 'Neu (N)'. The last two rows are highlighted with a blue bar on the left.

| Kundenauftr.-ID | Geschäftspartner-ID | Gesamtstatus |
|-----------------|---------------------|----------------|
| Entwurf | | |
| Entwurf | | |
| Entwurf | | |
| Entwurf | | |
| 2 | 100000002 | Zugestellt (D) |
| 3 | 100000033 | Neu (N) |