SAP HANA

Lesson Name: ALV with IDA

Lesson Objectives

After completing this lesson, participants will be able to -

- Proficient about SALV IDA . They can show data, enable hotspot and add buttons in toolbar in SALV IDA.

Contents

Existing or Traditional ALV

- The existing ALV has more functionality on the application layer.
- This makes it very slow as full data is being selected and sent to the ALV framework, which translates that into display on the GUI container.
- Since the entire data is being selected beforehand, the framework has to parse the data as required.
- Assume you have set a filter which only displays a single record in ALV output but the huge dataset was selected.
- Furthermore, you have a many records which you are sorting – again this is happening on the application layer.

SALV IDA introduction

- SALV IDA stands for SAP ABAP LIST VIEWER – INTEGRATED DATA ACCESS

- With HANA database aka HDB aka in-memory DB, many of the operation which can be executed on the front-end can be send to the database – the code pushdown.

- The new SALV IDA (Integrated Data Access) works more on **code push-down concept.**

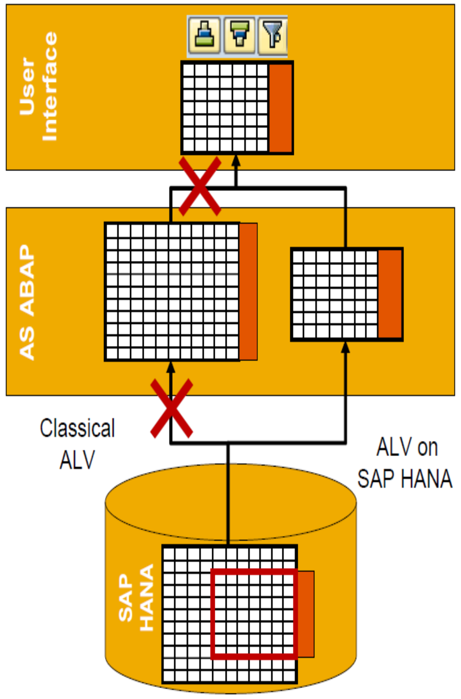
- Means, you don’t select the data and send that to the ALV, instead you generate the ALV for the DB table, DB view or a CDS views.

- The IDA framework then analyze the required columns, analyze the filters to get the required where condition and then executes the select query

SALV IDA Introduction

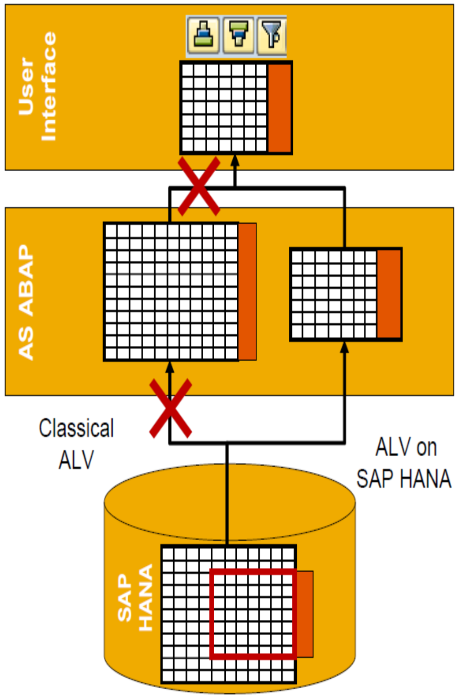
- SALV IDA for SAPGUI is an advanced list control to display data in tabular format from variety of data sources represented in ABAP Data Dictionary, for example SAP HANA artifacts (Ex: calculation view) represented as external views

- In-memory databases like SAP HANA can lead to significant performance improvements when processing large quantities of data.



SALV IDA Introduction

- Users should benefit from the required data being displayed more quickly, and the ability to manipulate their view of that data in different ways.
- In order to make this possible in the ALV environment, SAP has designed a special version of the List Viewer, the SAP List Viewer with Integrated Data Access.
- This variation of the ALV is optimized for use with SAP HANA.
- Only VISIBLE/REQUIRED rows are copied from Database



SALV IDA Features

Most important changes / improvements

- Real-time, regardless size of output table
- Only VISIBLE rows are copied from Database
- Data volume transfered from DB to application server is drastically reduced
- Code-to-data paradigm (aka Code Pushdown)
  Data intensive operations moved to database.
- Tools: CDS View, HANA View (using SQL Script)

ALV with IDA: Programming interface

A factory class CL_SALV_GUI_TABLE_IDA is used for ALV with integrated data access (ALV with IDA).

The create() method of this class can be used to create an instance of interface IF_SALV_GUI_TABLE_IDA using a database view/table.

Eg. Using a database table BSEG

```
  DATA(lo_alv) = cl_salv_gui_table_ida=>create( iv_table_name ='BSEG'
 ).
  lo_alv->fullscreen( )->display( ).
```

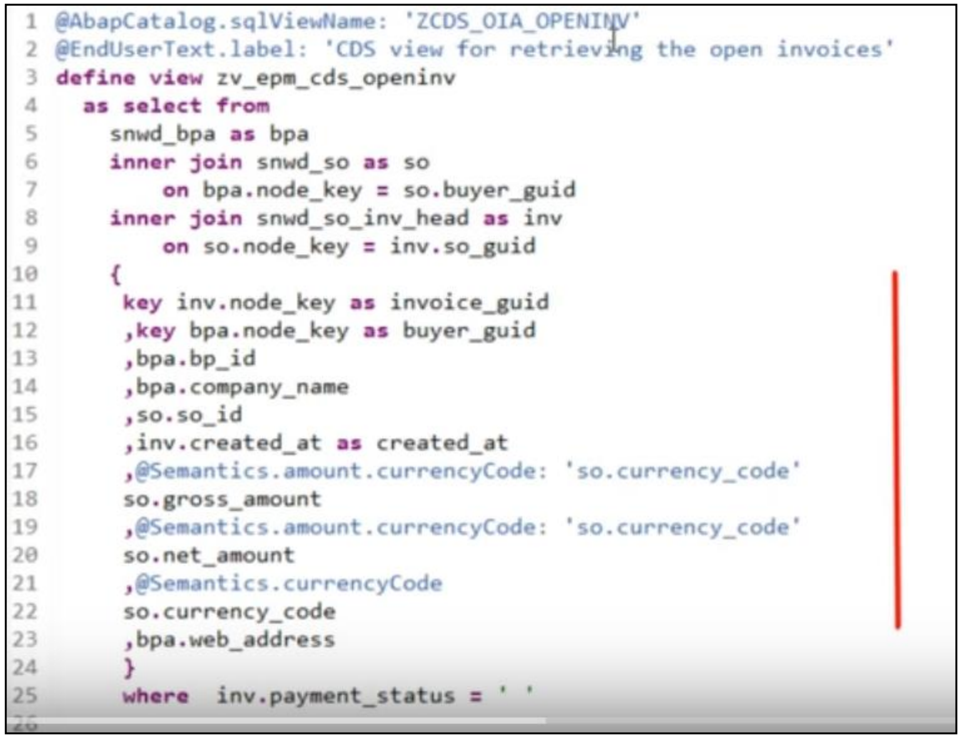
Eg. Using a view ZDCS_VIEW

```
DATA(lo_alv) = cl_salv_gui_table_ida=>create( ’ZCDS_VIEW' ).
  lo_alv->fullscreen( )->display( ).
```

Steps to consume CDS View in SALV IDA

Steps to consume CDS view in SALV IDA

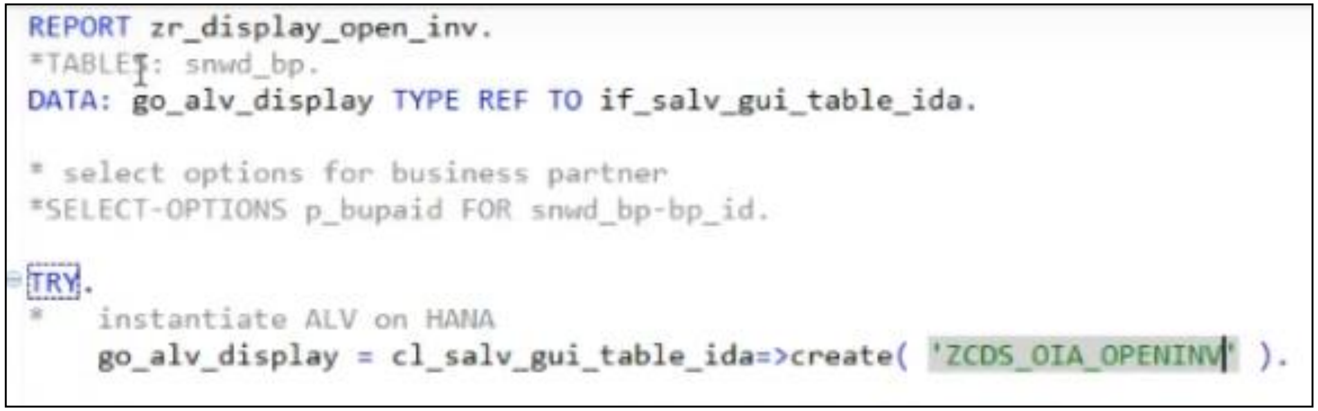
1) Create a DCS view
Here the view is ZCDS_OIA_OPENINV



```
@AbapCatalog.sqlViewName: 'ZCDS_OIA_OPENINV'
@EndUserText.label: 'CDS view for retrieving the open invoices'
define view zv_epm_cds_openinv
  as select from
    snwd_bpa as bpa
    inner join snwd_so as so
        on bpa.node_key = so.buyer_guid
    inner join snwd_so_inv_head as inv
        on so.node_key = inv.so_guid
    {
     key inv.node_key as invoice_guid
     ,key bpa.node_key as buyer_guid
     ,bpa.bp_id
     ,bpa.company_name
     ,so.so_id
     ,inv.created_at as created_at
     ,@Semantics.amount.currencyCode: 'so.currency_code'
     so.gross_amount
     ,@Semantics.amount.currencyCode: 'so.currency_code'
     so.net_amount
     ,@Semantics.currencyCode
     so.currency_code
     ,bpa.web_address
     }
     where  inv.payment_status = ' '
```

Steps to consume CDS View

2) Pass the CDS view name to the method create();

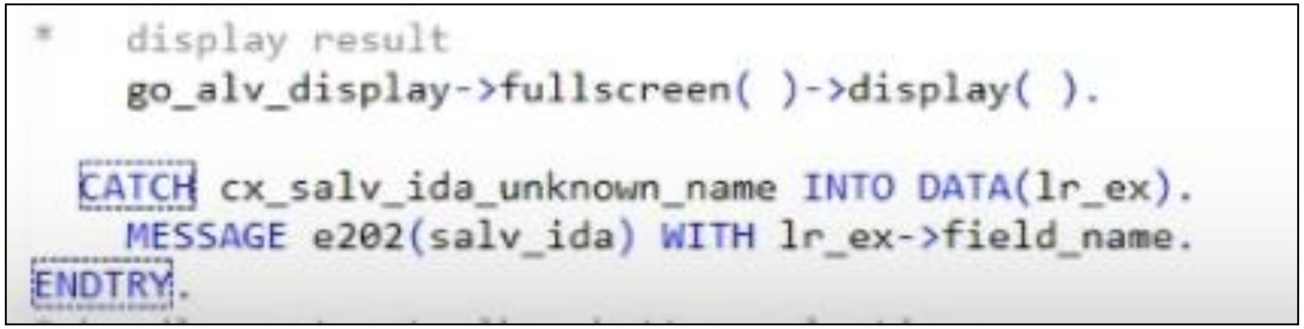


```
REPORT zr_display_open_inv.
*TABLES: snwd_bp.
DATA: go_alv_display TYPE REF TO if_salv_gui_table_ida.

* select options for business partner
*SELECT-OPTIONS p_bupaid FOR snwd_bp-bp_id.

TRY.
*    instantiate ALV on HANA
    go_alv_display = cl_salv_gui_table_ida=>create( 'ZCDS_OIA_OPENINV' ).
```

3) Call the display() method to print complete CDS view data with SALV.

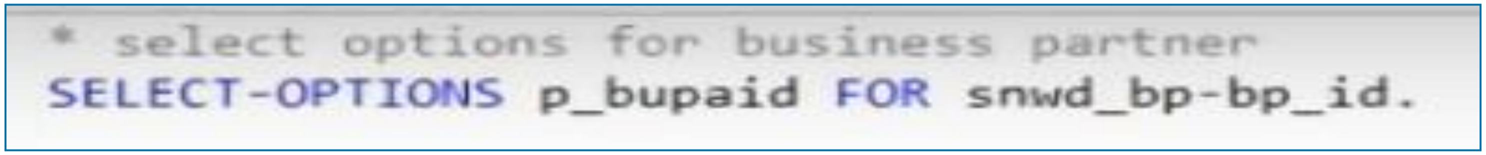


```
*    display result
     go_alv_display->fullscreen( )->display( ).

  CATCH cx_salv_ida_unknown_name INTO DATA(lr_ex).
     MESSAGE e202(salv_ida) WITH lr_ex->field_name.
ENDTRY.
```

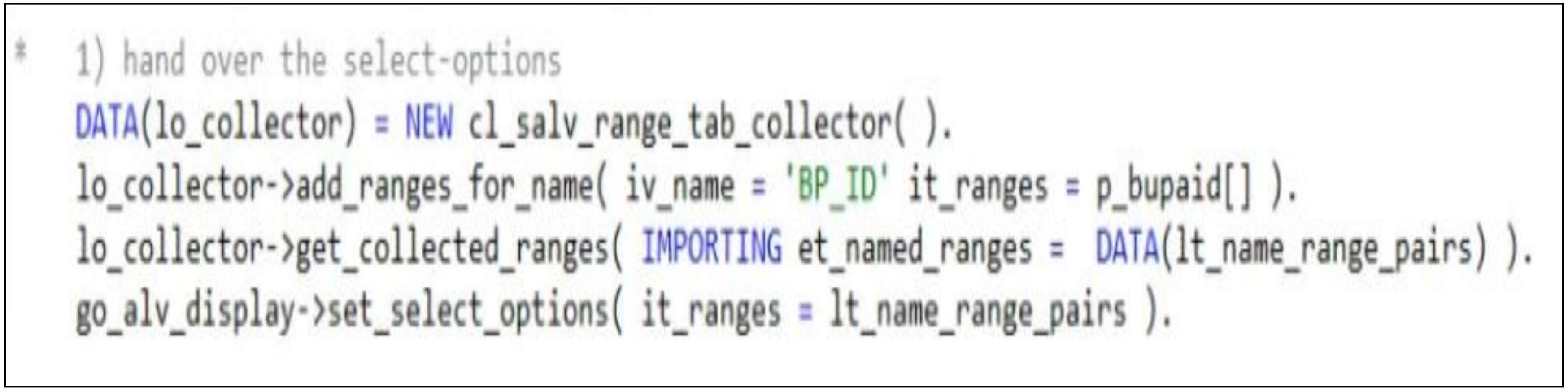
Handling Selections in SALV IDA

Declare select option as shown below



```
* select options for business partner
SELECT-OPTIONS p_bupaid FOR snwd_bp-bp_id.
```

Then we need to pass the entered selection criteria to the ALV dynamically by calling the set_select_options()



```
*   1) hand over the select-options
    DATA(lo_collector) = NEW cl_salv_range_tab_collector( ).
    lo_collector->add_ranges_for_name( iv_name = 'BP_ID' it_ranges = p_bupaid[] ).
    lo_collector->get_collected_ranges( IMPORTING et_named_ranges =  DATA(lt_name_range_pairs) ).
    go_alv_display->set_select_options( it_ranges = lt_name_range_pairs ).
```

SALV IDA Methods

Few frequently used methods are :

- Field_catalog()
    - ❑ set_field_header_texts()
    - ❑ get_available_fields ()
    - ❑ set_available_fields ()
    - ❑ disable_aggregation()
    - ❑ disable_sort()
    - ❑ disable_filter()

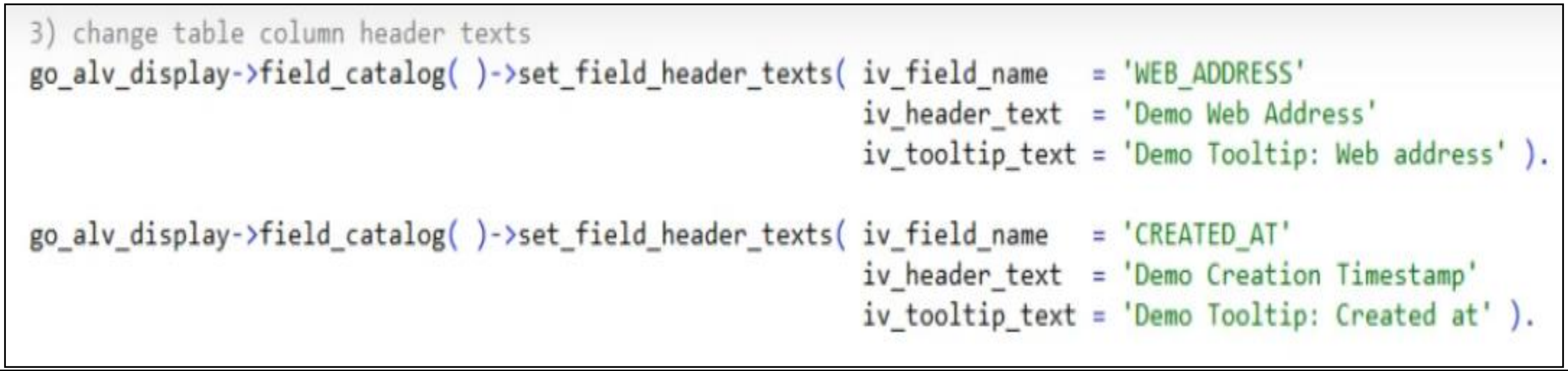
- Display_Options()
    - ❑ enable_alternating_row_pattern ()
    - ❑ set_title ()

- default_Layout()
- Standard_functions()
- Toolbar()

Field_Catalog()

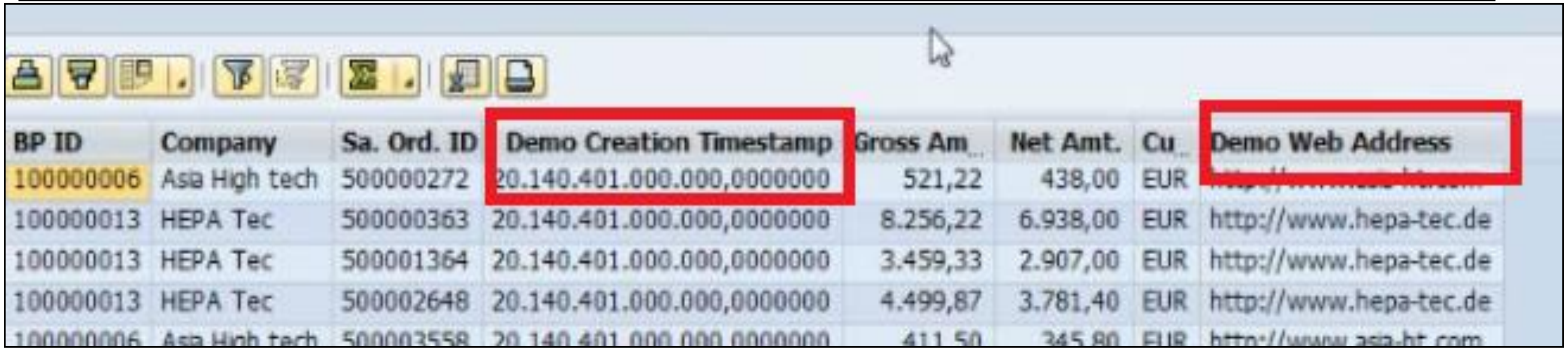
Set_field_header_texts()

This method can be used to set field catalog texts and tooltip texts for each fields present in the alv.



```
3) change table column header texts
go_alv_display->field_catalog( )->set_field_header_texts( iv_field_name   = 'WEB_ADDRESS'
                                                           iv_header_text  = 'Demo Web Address'
                                                           iv_tooltip_text = 'Demo Tooltip: Web address' ).

go_alv_display->field_catalog( )->set_field_header_texts( iv_field_name   = 'CREATED_AT'
                                                           iv_header_text  = 'Demo Creation Timestamp'
                                                           iv_tooltip_text = 'Demo Tooltip: Created at' ).
```



BP ID	Company	Sa. Ord. ID	Demo Creation Timestamp	Gross Am...	Net Amt.	Cu...	Demo Web Address
100000006	Asia High tech	500000272	20.140.401.000.000,0000000	521,22	438,00	EUR	[illegible]
100000013	HEPA Tec	500000363	20.140.401.000.000,0000000	8.256,22	6.938,00	EUR	http://www.hepa-tec.de
100000013	HEPA Tec	500001364	20.140.401.000.000,0000000	3.459,33	2.907,00	EUR	http://www.hepa-tec.de
100000013	HEPA Tec	500002648	20.140.401.000.000,0000000	4.499,87	3.781,40	EUR	http://www.hepa-tec.de
100000006	Asia High tech	500003558	20.140.401.000.000,0000000	411,50	345,80	EUR	http://www.asia-ht.com

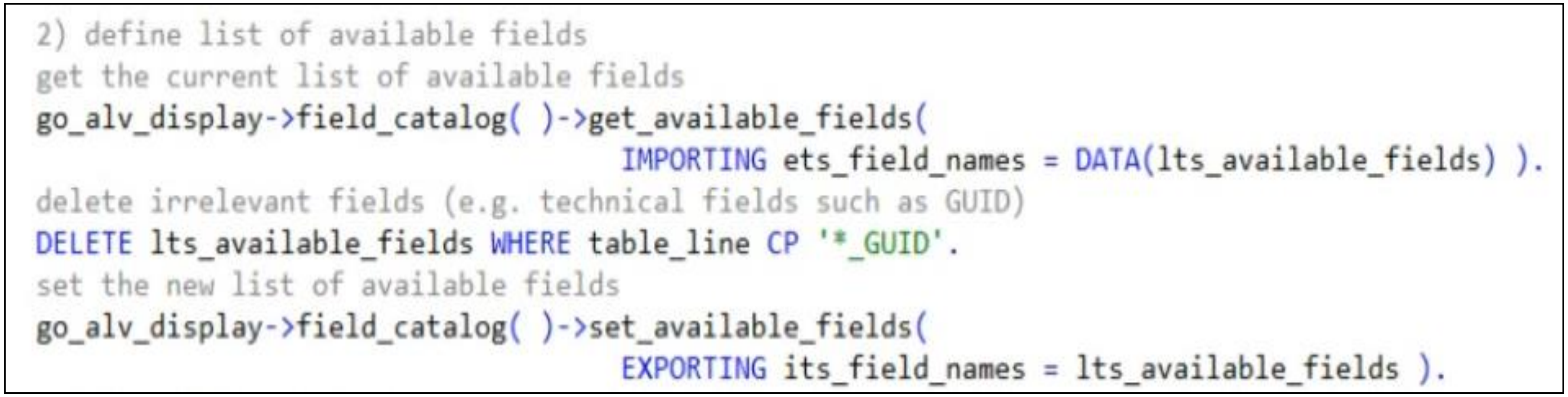
Field_Catalog(): GET/SET

get_available_fields ()

This method can be used to extract all the available fields.

set_available_fields ()

This method can be used to set fewer fields as per the requirement.



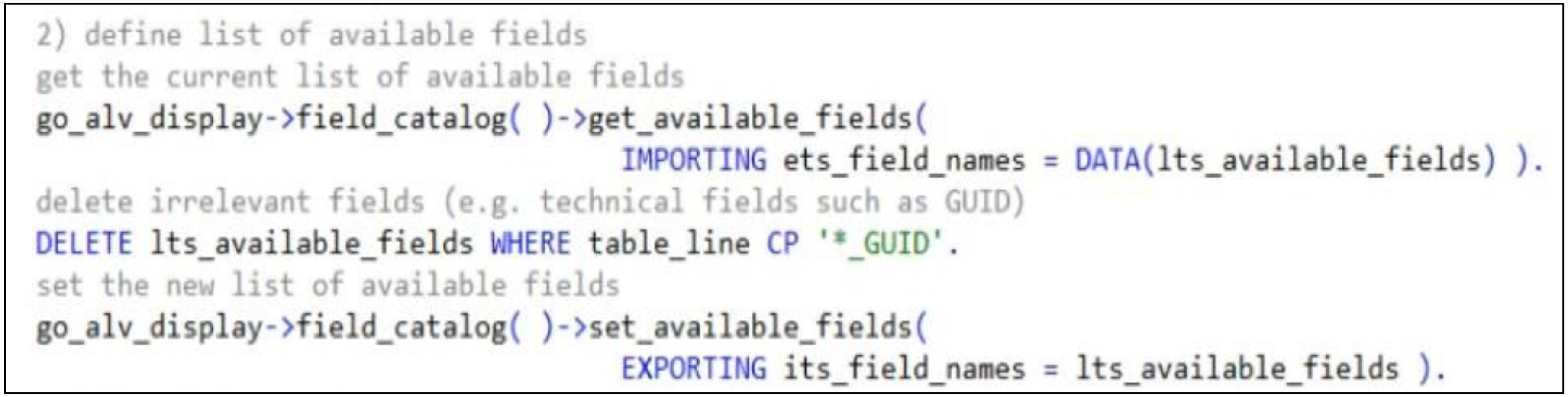
```
2) define list of available fields
get the current list of available fields
go_alv_display->field_catalog( )->get_available_fields(
                                    IMPORTING ets_field_names = DATA(lts_available_fields) ).
delete irrelevant fields (e.g. technical fields such as GUID)
DELETE lts_available_fields WHERE table_line CP '*_GUID'.
set the new list of available fields
go_alv_display->field_catalog( )->set_available_fields(
                                    EXPORTING its_field_names = lts_available_fields ).
```

Field_Catalog(): GET/SET

In the example given below ,the method **get_available_fields()** gives the list of all the fields.

Irrelevant fields (here containing _GUID) is removed from the list by using the DELETE statement.

The necessary new list of fields is set back by using the method **set_available_fields()**



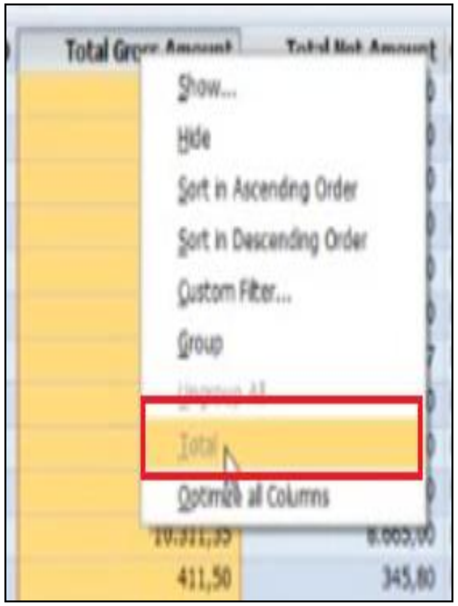
```
2) define list of available fields
get the current list of available fields
go_alv_display->field_catalog( )->get_available_fields(
                                    IMPORTING ets_field_names = DATA(lts_available_fields) ).
delete irrelevant fields (e.g. technical fields such as GUID)
DELETE lts_available_fields WHERE table_line CP '*_GUID'.
set the new list of available fields
go_alv_display->field_catalog( )->set_available_fields(
                                    EXPORTING its_field_names = lts_available_fields ).
```

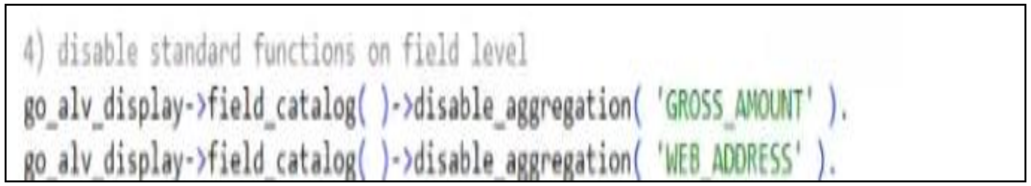
Field_Catalog() : Aggregation

disable_aggregation()

This method can be used to disable different functionalities for designated fields .

Eg. The mathematical operations on field GROSS_AMOUNT and WEB_ADDRESS are disabled.





```
4) disable standard functions on field level
go_alv_display->field_catalog( )->disable_aggregation( 'GROSS_AMOUNT' ).
go_alv_display->field_catalog( )->disable_aggregation( 'WEB_ADDRESS' ).
```

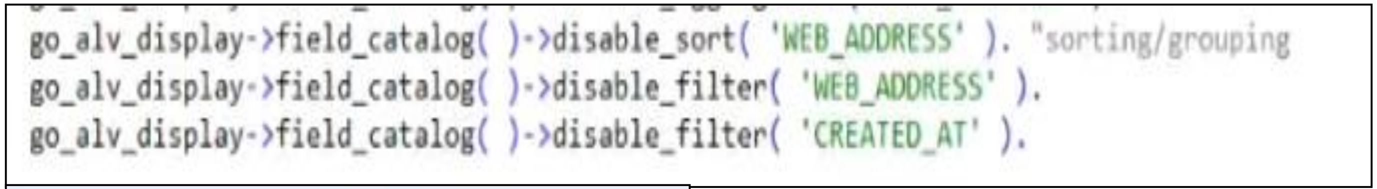
Field_Catalog() : SORT/FILTER

disable_sort()

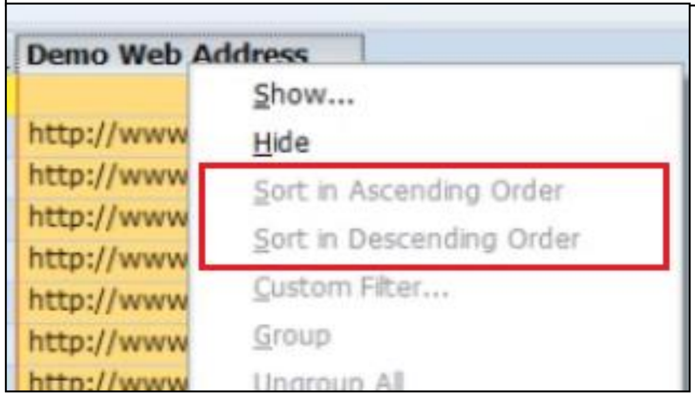
This method can be used to disable sort functionalities for designated fields .

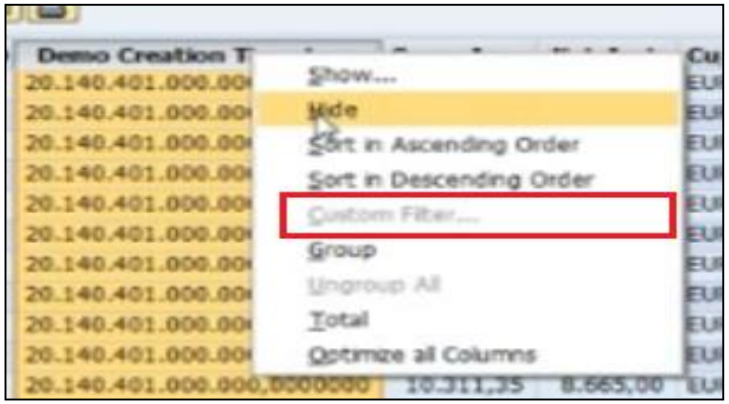
disable_filter()

This method can be used to disable sort functionalities for designated fields .



```
go_alv_display->field_catalog( )->disable_sort( 'WEB_ADDRESS' ). "sorting/grouping
go_alv_display->field_catalog( )->disable_filter( 'WEB_ADDRESS' ).
go_alv_display->field_catalog( )->disable_filter( 'CREATED_AT' ).
```





Display_Options()

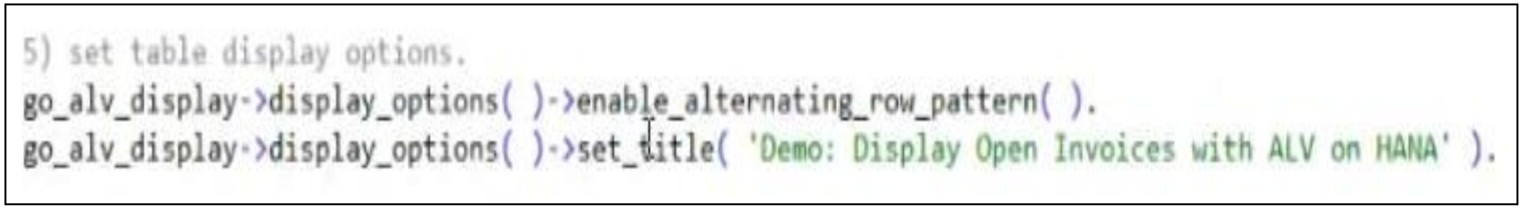
Below 2 methods of display_option is frequently used

enable_alternating_row_pattern ()

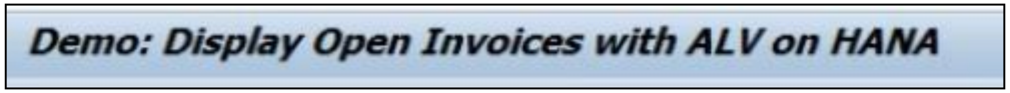
We have the freedom to set the fields interchangeable in the ALV . To do that we refer to enable_alternating_row_pattern for the display option .

set_title ()

Normally for reports the program title is set as we enter in the program description. But for IDA-ALV we can overwrite that at runtime by calling method set_title in display options.



```
5) set table display options.
go_alv_display->display_options( )->enable_alternating_row_pattern( ).
go_alv_display->display_options( )->set_title( 'Demo: Display Open Invoices with ALV on HANA' ).
```



Demo: Display Open Invoices with ALV on HANA

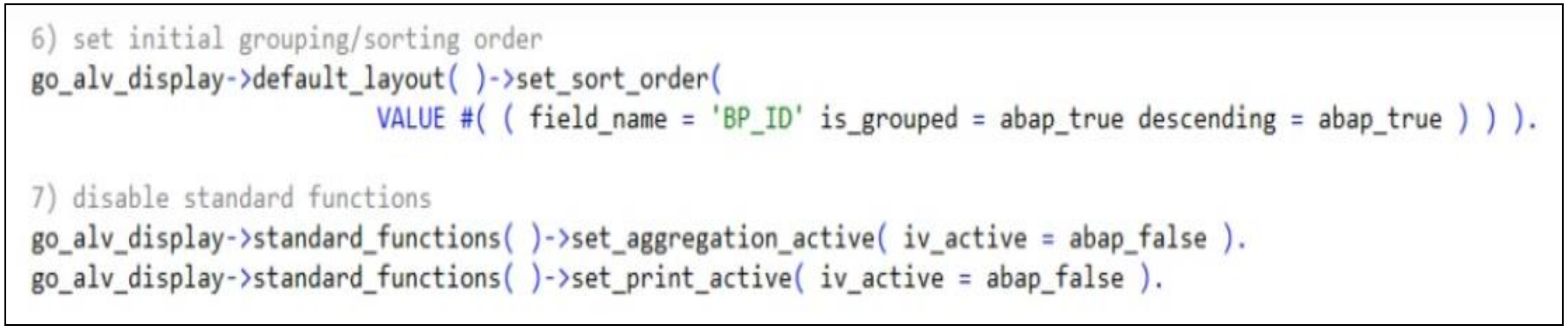
More methods

default_Layout()

We can also set the initial grouping if applicable with a particular field by specifying that in default layout properties.

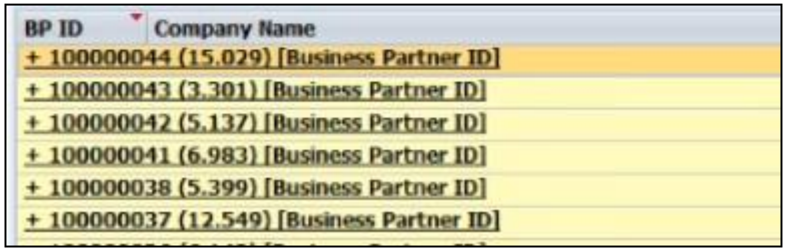
standard_functions().

We can also disable some standard functionalities if needed by calling the standard_functions.



```
6) set initial grouping/sorting order
go_alv_display->default_layout( )->set_sort_order(
                        VALUE #( ( field_name = 'BP_ID' is_grouped = abap_true descending = abap_true ) ) ).

7) disable standard functions
go_alv_display->standard_functions( )->set_aggregation_active( iv_active = abap_false ).
go_alv_display->standard_functions( )->set_print_active( iv_active = abap_false ).
```



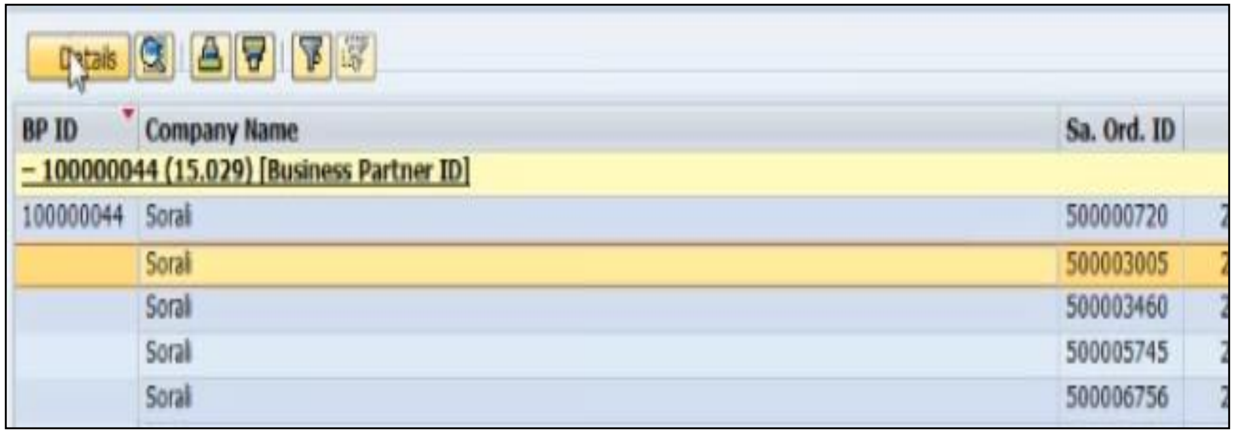
BP ID	Company Name
+ 100000044 (15.029) [Business Partner ID]	
+ 100000043 (3.301) [Business Partner ID]	
+ 100000042 (5.137) [Business Partner ID]	
+ 100000041 (6.983) [Business Partner ID]	
+ 100000038 (5.399) [Business Partner ID]	
+ 100000037 (12.549) [Business Partner ID]	

Toolbar : add_button()

In the toolbar, we can add a button of a specific functionality by using the method add_button()

Eg.A button as “Details” in added in the toolbar area using the below code snippet.

```
8) add a custom toolbar button.
add custom toolbar button/function
go_alv_display->toolbar( )->add_button( EXPORTING iv_fcode        = 'DETAIL_SCREEN'
                                                  iv_text         = 'Details'
                                                  iv_quickinfo    = 'Demo: Sales Order Details'
                                                  iv_before_standard_functions = abap_true   ).
```



Details

BP ID	Company Name	Sa. Ord. ID
- 100000044 (15.029) [Business Partner ID]		
100000044	Sorali	500000720
	Sorali	500003005
	Sorali	500003460
	Sorali	500005745
	Sorali	500006756

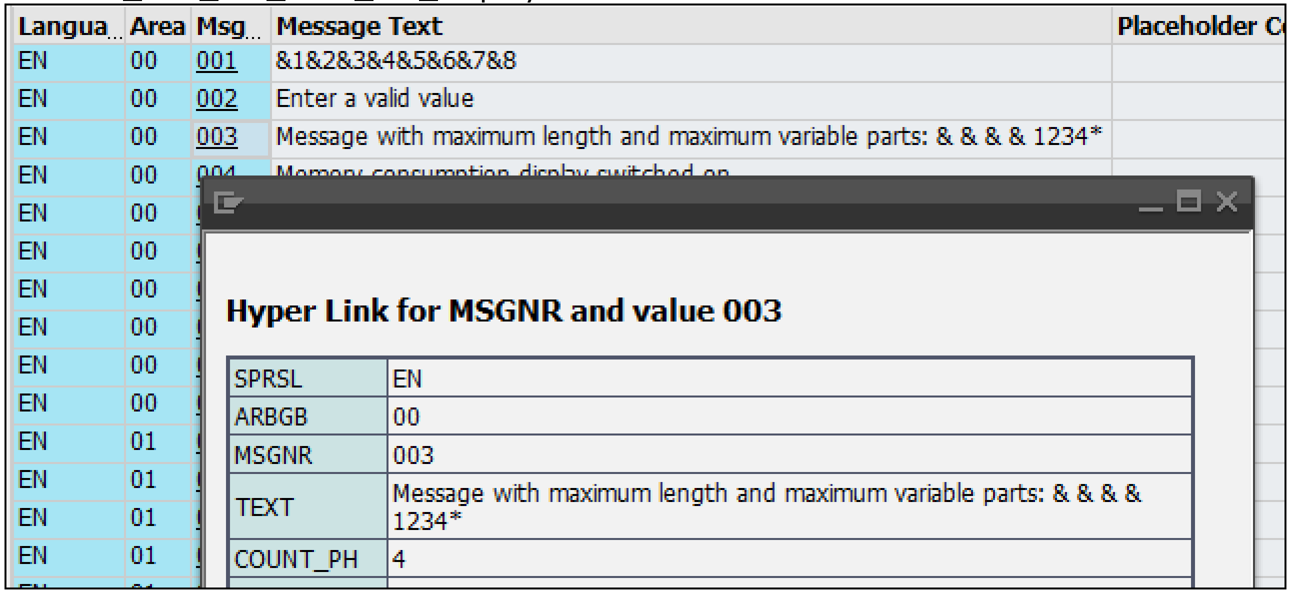
Hotspot

To enable hotpsot on a field, use **handle_hot_spot ().**

Using the below code ,the field Message Number is made as hotspot.

When user clicks on that message number a pop up appears to show all the relevant data regarding that message number.

```
TRY.
o_salv_ida->field_catalog( )->display_options( )->display_as_link_to_action( 'MSGNR' ).
SET HANDLER me->handle_hot_spot FOR o_salv_ida->field_catalog( )->display_options( ).
 CATCH cx_salv_ida_unknown_name cx_salv_call_after_1st_display.
ENDTRY.
```



Langua…	Area	Msg…	Message Text	Placeholder C
EN	00	001	&1&2&3&4&5&6&7&8	
EN	00	002	Enter a valid value	
EN	00	003	Message with maximum length and maximum variable parts: & & & & 1234*	
EN	00	004	Memory consumption display switched on	
EN	00			
EN	00			
EN	00			
EN	00			
EN	00			
EN	00			
EN	01			
EN	01			
EN	01			
EN	01			

Hyper Link for MSGNR and value 003

SPRSL	EN
ARBGB	00
MSGNR	003
TEXT	Message with maximum length and maximum variable parts: & & & & 1234*
COUNT_PH	4

Summary

In this lesson, you have learnt:

SALV Features
ALV with IDA – Programming Interface
Consuming CDS View Output in SALV IDA
Handling Selections in SALV IDA
Methods of SALV IDA
For Hotspot