ABAP Code Optimization for SAP HANA

Webinar Presenter:

Albrecht Gass

Chief Architect, smartShift Technologies

Moderator

Manasi Kakade

Product Marketing Manager

smartShift Technologies

Agenda

1. Introduction smartShift Technologies
2. SAP HANA Overview
3. New Performance Rules
4. Optimized ABAP Code Examples
5. Q & A
6. Further ABAP Resources

Note: You can also continue discussion via Twitter @smartShiftTech with **#smartSAPHANA**

About smartShift Technologies

About smartShift Tech

smartShift TECHNOLOGIES

RED HERRING EUROPE FINALIST 100

SAP® Certified
Integration with SAP Applications

Gartner 2012 CoolVendor

Select Customers

We Partner with the Best of the Breed!

smartScale for HANA Offering

smartScale for HANA will consists of the appropriate components with newly added HANA Code Optimization Ruleset.

- smartAnalyze
  - SaaS Analysis
- smartUpgrade
  - Unicode Enablement
  - Change Impact Analysis
- smartDevelop
  - On-going ABAP Code Remediation

Presenter

Albrecht Gass

Chief Architect

smartShift Technologies

SAP HANA Overview

SAP HANA Overview

- Column and Row Store
- Dictionary based (compression)
- ACID RDBMS
- Stand-alone or SAP Business Suite/BW
- JBDC, ODBC, ADBC, MDX
- Unstructured Data Support
- Full Text Searchable
- HW Appliance or Cloud Deployment
- Advanced Calculation Engine
- Historic Data

SAP HANA Overview

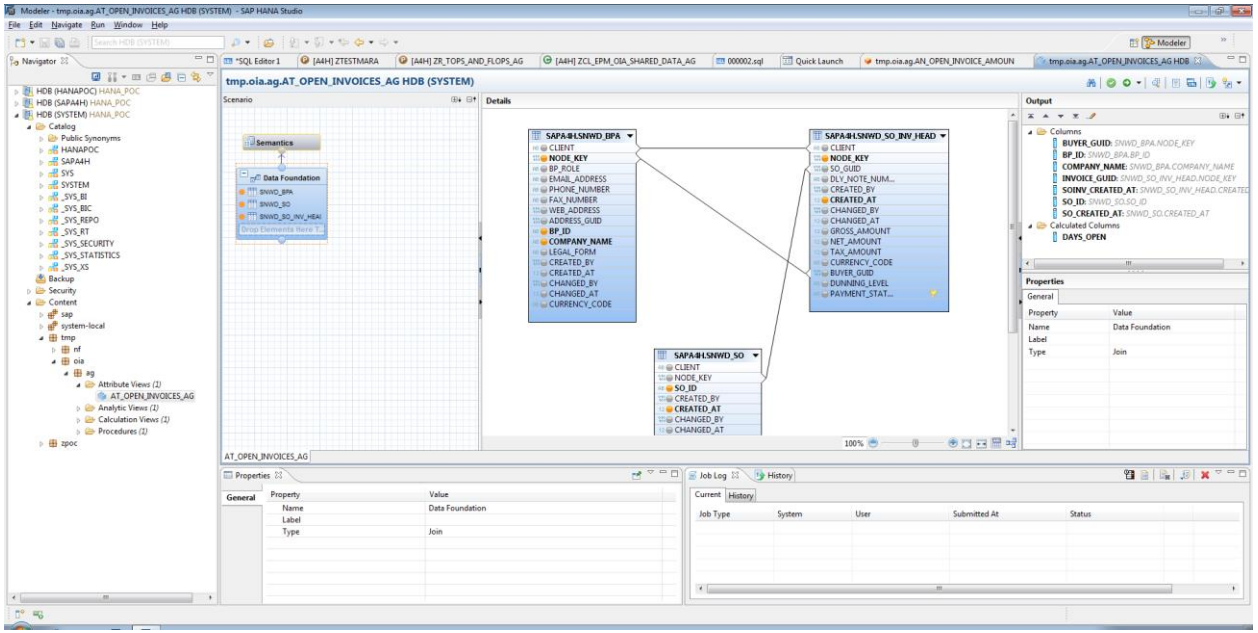
- Unicode Only
- Integration with ‘R’
- Large Tables
- Dynamic Tables
- ABAP Code Must be Optimized
- Eclipse based IDE (ADT)
- Optimization for Star Queries
- Signification Performance Improvements
- ALV paging for large results

HANA “Objectives”

- Does not impact transaction processing
- Enables “reporting without fear” by increasing reporting speeds dramatically
- Eliminates SAP as Access Loader
- Keeps processing within SAP and not Excel
- Avoid using old and/or partial data
- Convert batch processes to real-time operations
- Enable new big data processes

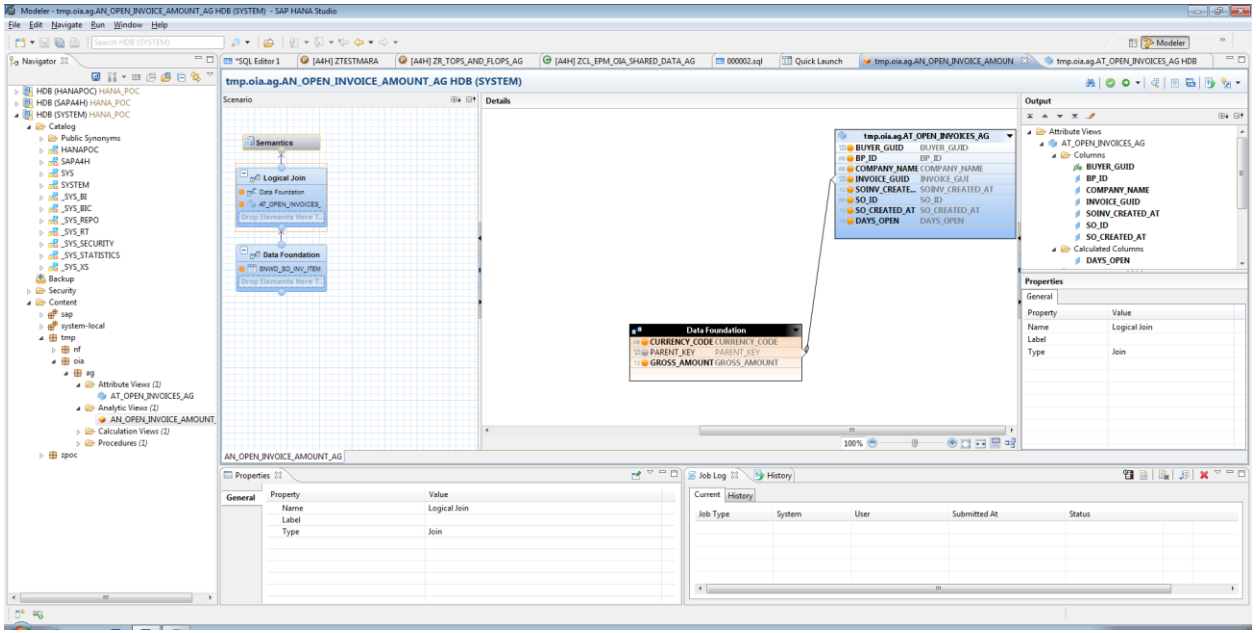
Attribute View

- Can join multiple tables
- Perform simple calculations
- Only SQL Functions can be used



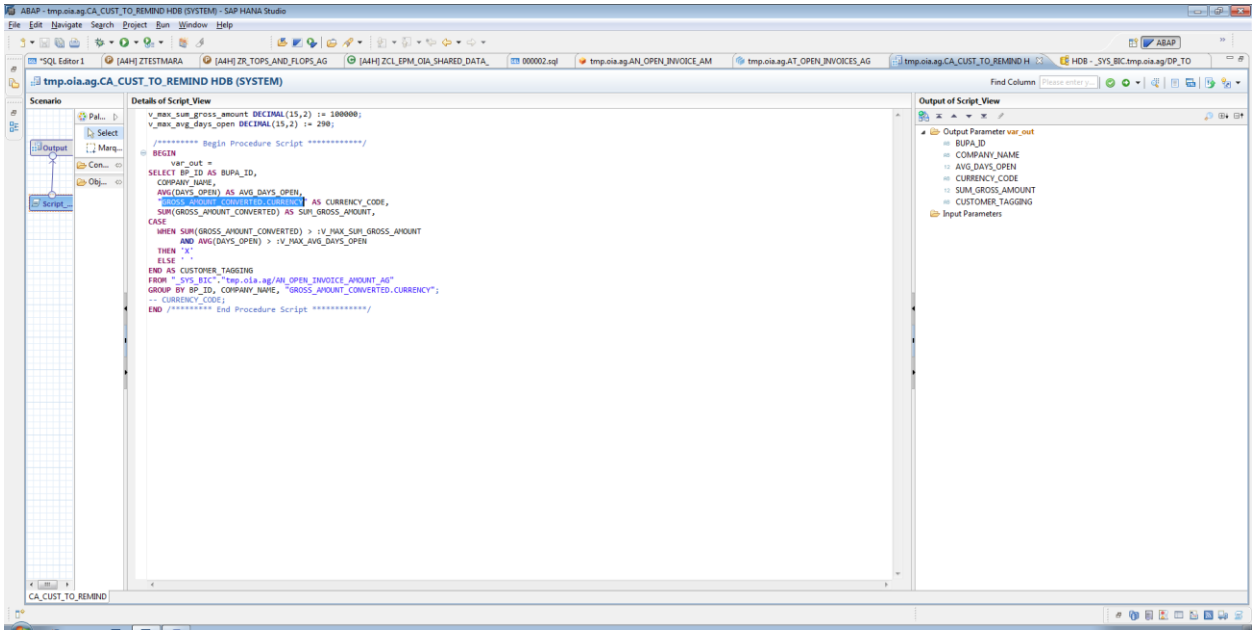
Analytical View

- Replaces cubes in traditional BW
- Joins attribute views and fact tables
- Perform calculations and aggregations
- Only SQL Functions can be used



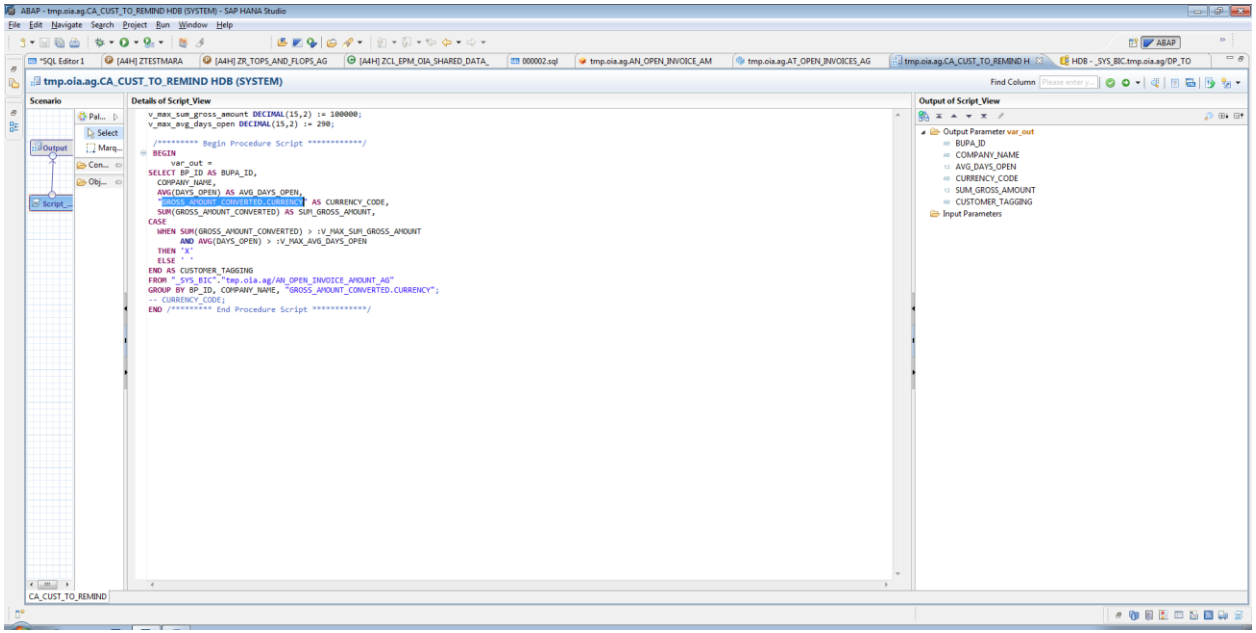
Calculated View

- Full SQLScript
- Must define output record type
- Supports measures and hierarchies
- Callable via standard OPENSQL

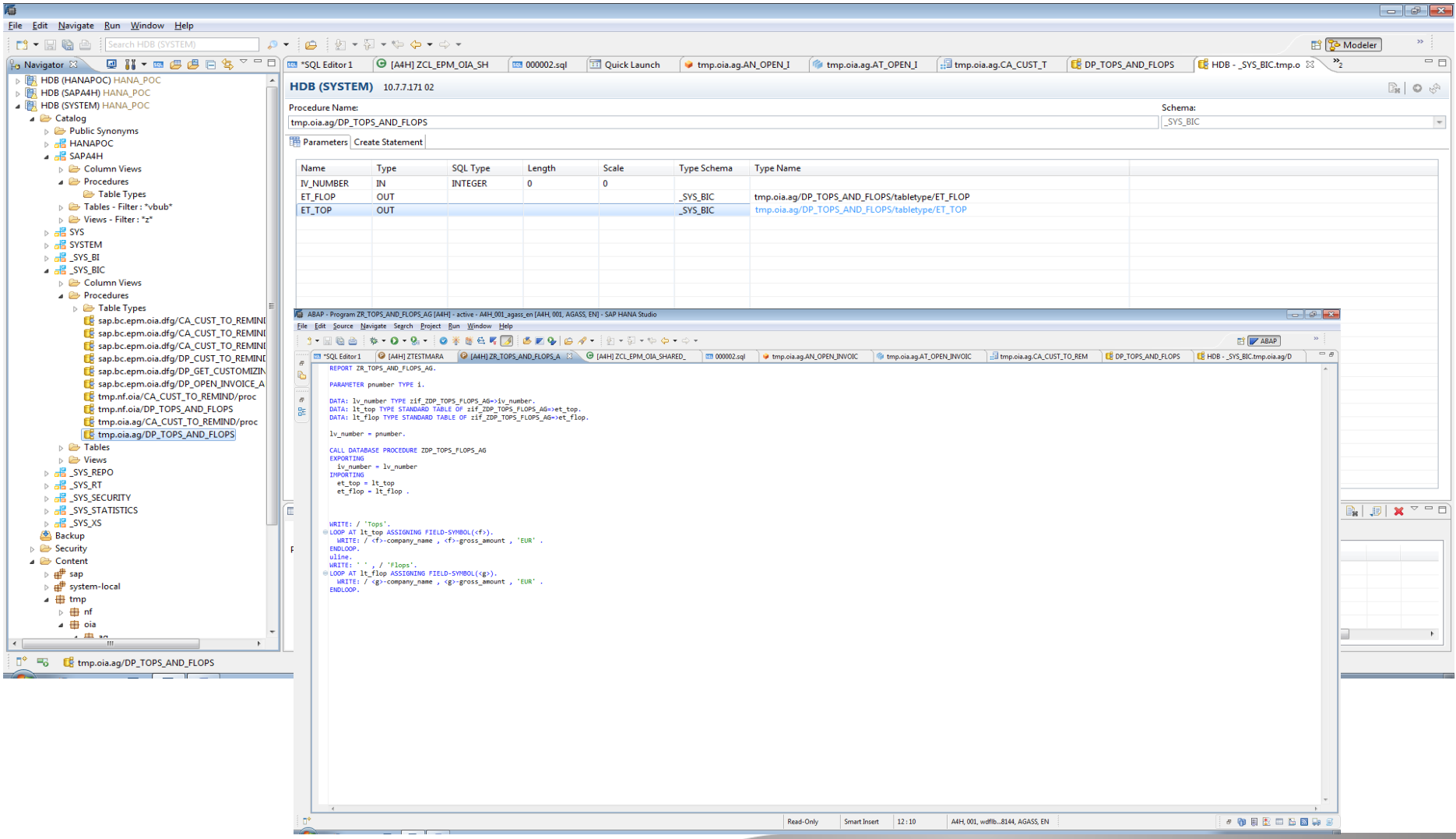


DB Procedure

- Full SQLScript
- Can define a output record type
- Callable via CALL DATABASE PROCEDURE
- Creates table types for input and output arguments



DB Procedure (cont.)



Externalizing a HANA view to be used in ABAP

- View defined in HANA
- Externalized so that it can be referenced by ABAP code
- Can be used in OpenSQL statements

[A4H] ZR_POC_ALV | [A4H] ZR_POC_2 | [A4H] ZV_DELIVERY_DATA

Dictionary View: ZV_DELIVERY_DATA

General Attributes

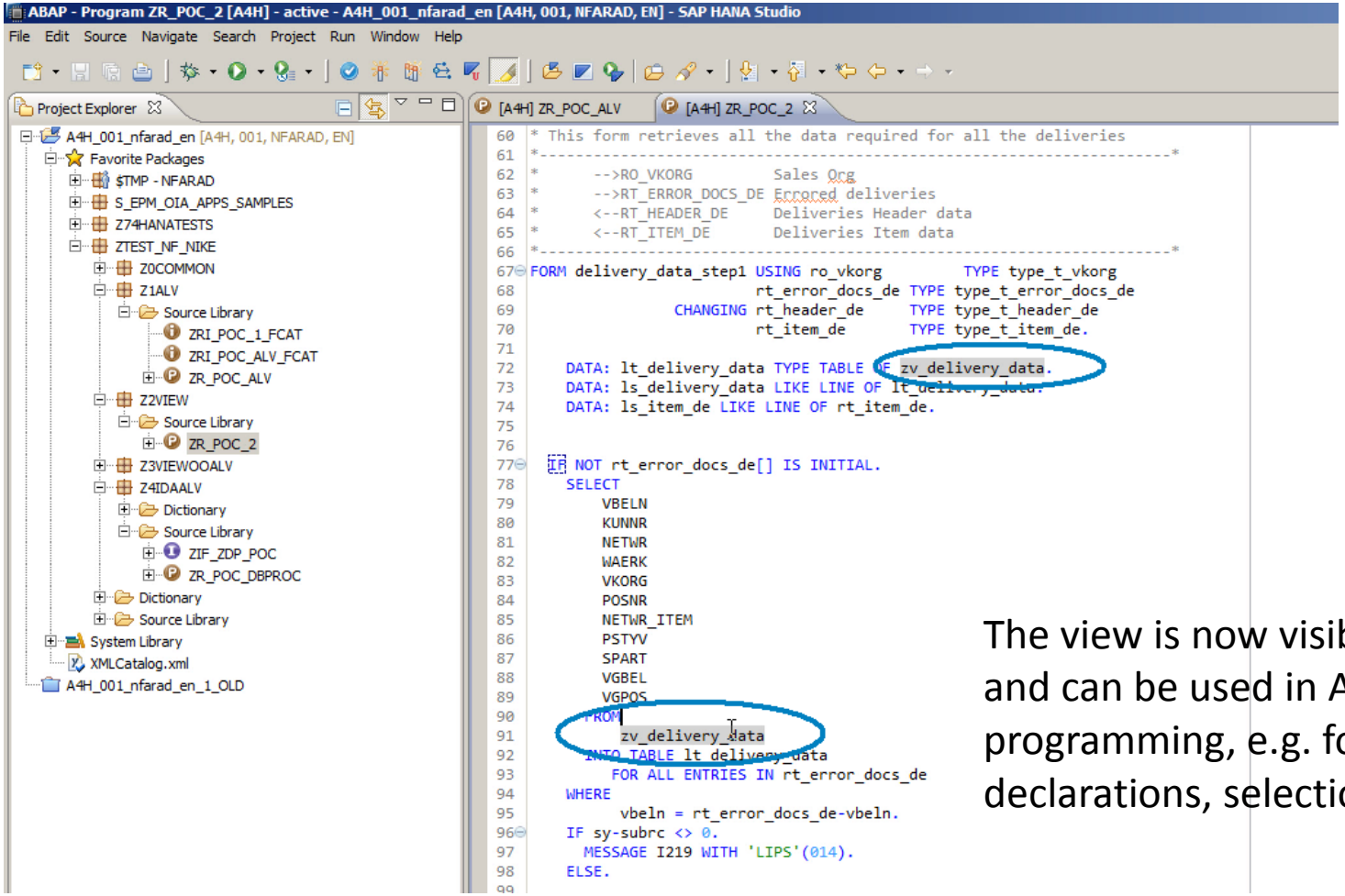
View Type: External View

HANA View: zpoc.ZVC_DELIVERY_DATA

Fields

DDIC Name	DDIC Type	Length	Decimals	HANA Name	HANA Type
VBELN	CHAR	10	0	VBELN	NVARCHAR(10)
KUNNR	CHAR	10	0	KUNNR	NVARCHAR(10)
NETWR	DEC	15	2	NETWR	DECIMAL(15,2)
WAERK	CHAR	5	0	WAERK	NVARCHAR(5)
VKORG	CHAR	4	0	VKORG	NVARCHAR(4)
POSNR	CHAR	6	0	POSNR	NVARCHAR(6)
NETWR_ITEM	DEC	15	2	NETWR_ITEM	DECIMAL(15,2)
PSTYV	CHAR	4	0	PSTYV	NVARCHAR(4)
SPART	CHAR	2	0	SPART	NVARCHAR(2)
VGBEL	CHAR	10	0	VGBEL	NVARCHAR(10)
VGPOS	CHAR	6	0	VGPOS	NVARCHAR(6)

Externalizing a view on MARA to be used by ABAP



The view is now visible in DDIC and can be used in ABAP programming, e.g. for declarations, selections, etc.

Performance Guidelines and Rules

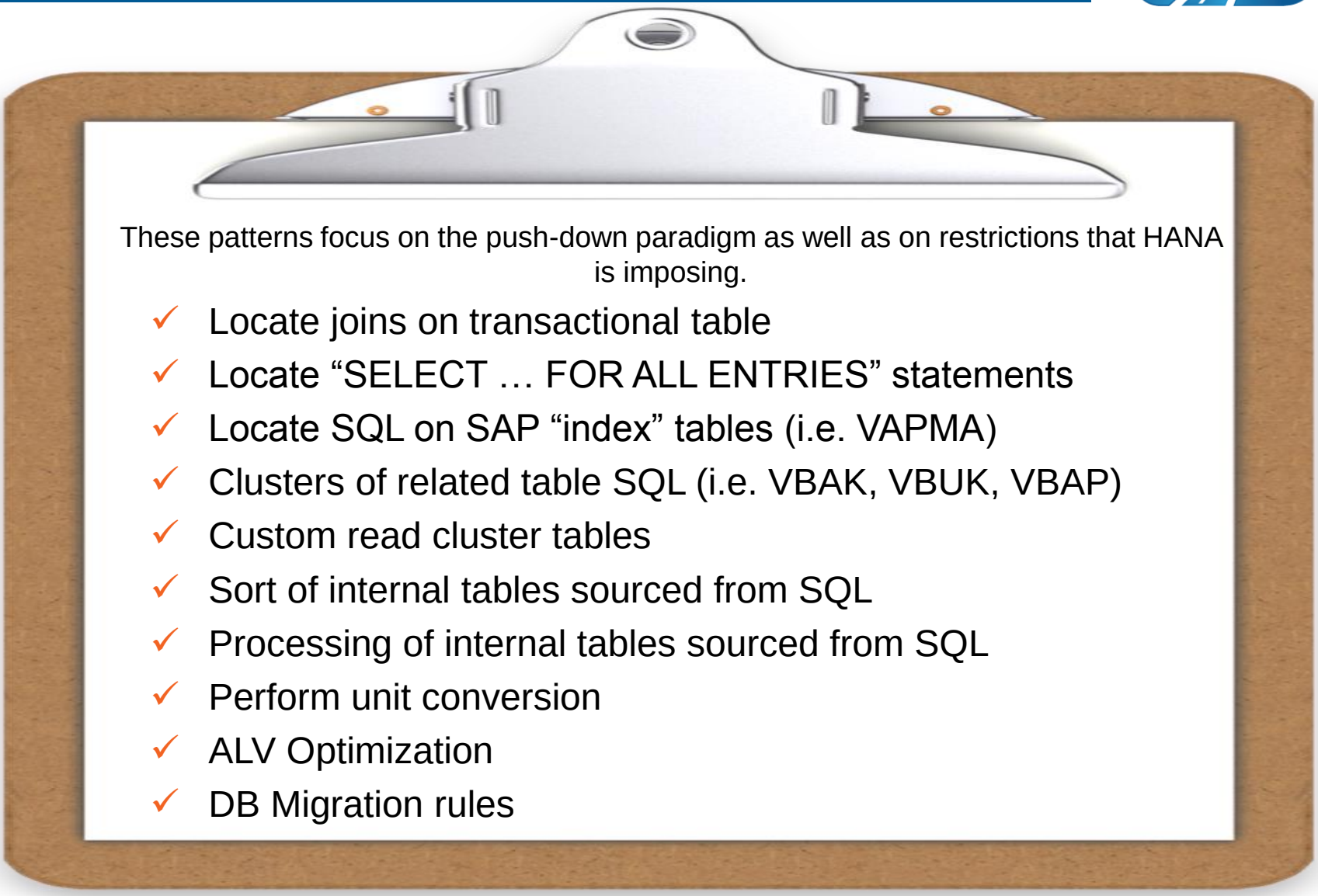
HANA Optimization Guidelines

- Reduce result set
- Reduce amount of data transfer
- Reduce number of DB round trips
- Index/Query optimization
- Text search for F4 help and type-ahead search
- Avoid native SQL
- Consider changes in the order in the result
- Use buffering on application server
- Existing best practices still apply
- Don’t overload HANA server

HANA Implementation Considerations

- Side-car vs. Primary DB
- Instance Size
  - Memory
  - CPU
- Table Partitioning
- Consider Changes in Query Result Ordering
- Incompatible Native SQL Code
- Unicode Requirement

New HANA Performance Rules



These patterns focus on the push-down paradigm as well as on restrictions that HANA is imposing.

- ✓ Locate joins on transactional table
- ✓ Locate “SELECT … FOR ALL ENTRIES” statements
- ✓ Locate SQL on SAP “index” tables (i.e. VAPMA)
- ✓ Clusters of related table SQL (i.e. VBAK, VBUK, VBAP)
- ✓ Custom read cluster tables
- ✓ Sort of internal tables sourced from SQL
- ✓ Processing of internal tables sourced from SQL
- ✓ Perform unit conversion
- ✓ ALV Optimization
- ✓ DB Migration rules

Optimized ABAP Code Examples

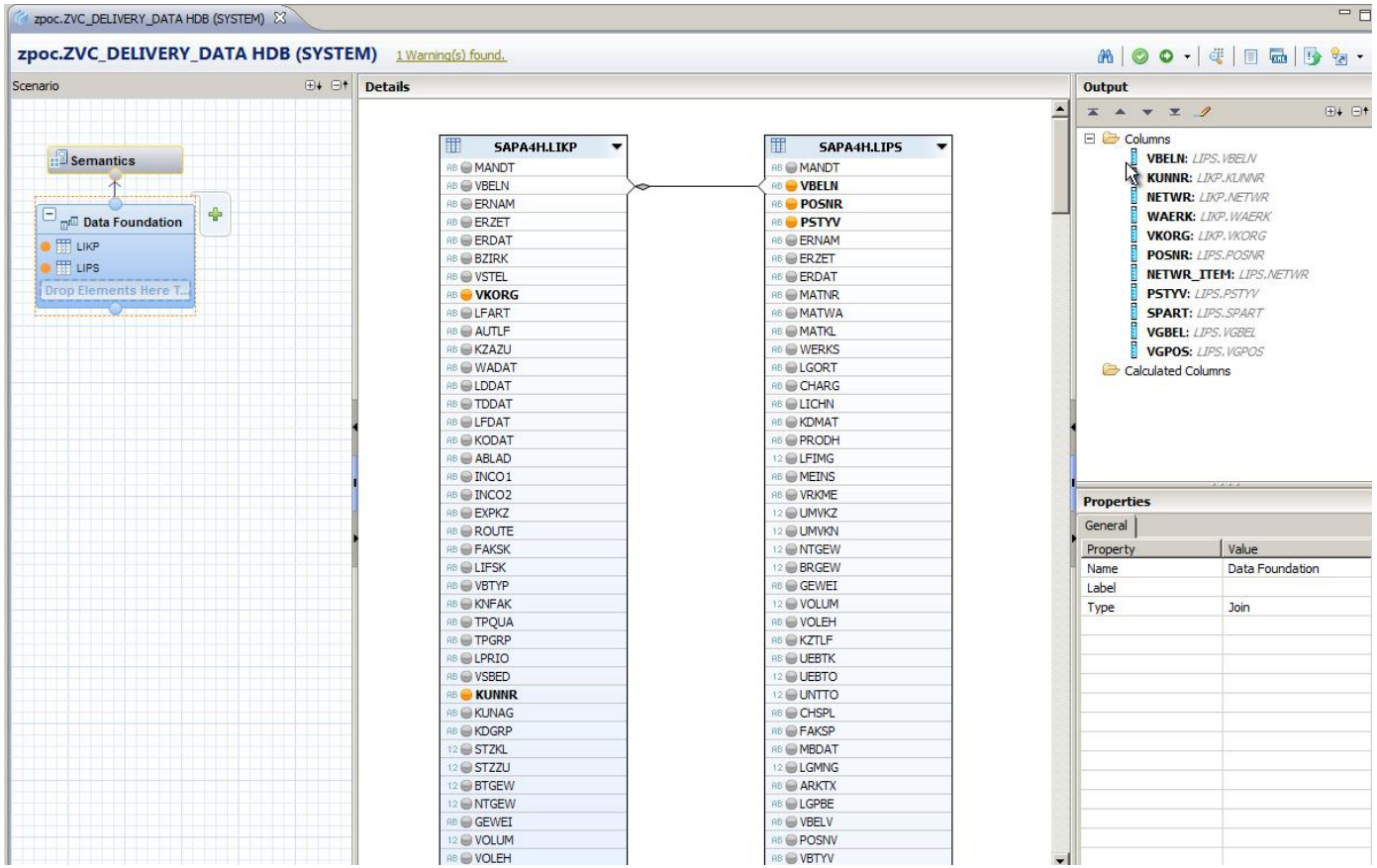
Example 1 – Using an external view

- Originally using nested SELECT FOR ALL ENTRIES

```
FORM delivery_data USING ro_vkorg         TYPE type_t_vkorg
                         rt_error_docs_de TYPE type_t_error_docs_de
                CHANGING rt_header_de     TYPE type_t_header_de
                         rt_item_de       TYPE type_t_item_de.

  IF NOT rt_error_docs_de[] IS INITIAL.
    SELECT kunnr
           vbeln
           netwr
           waerk
           vkorg
      FROM likp
      INTO TABLE rt_header_de
       FOR ALL ENTRIES IN rt_error_docs_de
     WHERE vbeln = rt_error_docs_de-vbeln
"      AND vkorg IN ro_vkorg.
       AND vkorg IN ro_vkorg.
    IF sy-subrc = 0.

      SELECT vbeln
             posnr
             netwr
             pstyv
             spart
             vgbel
             vgpos
        FROM lips
        INTO TABLE rt_item_de
         FOR ALL ENTRIES IN rt_header_de
       WHERE vbeln = rt_header_de-vbeln.
        IF sy-subrc <> 0.
          MESSAGE I219 WITH 'LIPS'(014).
        ENDIF.
      ELSE.
        MESSAGE I034 WITH 'Delivery'(015).
      ENDIF.
    ENDIF.
ENDFORM.                    " delivery_data
```

Example 1 – Using an external view (2)

- Model a view based on a join of LIKP and LIPS



Example 1 – Using an external view (2)

- Make the new view visible to app layer: define a dictionary view

Dictionary View: ZV_DELIVERY_DATA

General Attributes

View Type: External View

HANA View: zpoc.ZVC_DELIVERY_DATA

Fields

DDIC Name	DDIC Type	Length	Decimals	HANA Name	HANA Type
VBELN	CHAR	10	0	VBELN	NVARCHAR(10)
KUNNR	CHAR	10	0	KUNNR	NVARCHAR(10)
NETWR	DEC	15	2	NETWR	DECIMAL(15,2)
WAERK	CHAR	5	0	WAERK	NVARCHAR(5)
VKORG	CHAR	4	0	VKORG	NVARCHAR(4)
POSNR	CHAR	6	0	POSNR	NVARCHAR(6)
NETWR_ITEM	DEC	15	2	NETWR_ITEM	DECIMAL(15,2)
PSTYV	CHAR	4	0	PSTYV	NVARCHAR(4)
SPART	CHAR	2	0	SPART	NVARCHAR(2)
VGBEL	CHAR	10	0	VGBEL	NVARCHAR(10)
VGPOS	CHAR	6	0	VGPOS	NVARCHAR(6)

Example 1 – Using an external view (3)

Replacing SELECT ... FOR ALL ENTRIES:

- Required data is provides by View ZV_ERROR_DOCS, based on VBUK, VBFS, VBSK,
- Join of ZV_ERROR_DOCS with ZV_DELIVERY_DATA,
- Integration of pre-requisite conditions, selection criteria and type

```
SELECT
  zd~vbeln zd~kunnr zd~netwr zd~waerk zd~vkorg zd~posnr
  zd~netwr_item zd~pstyv zd~spart zd~vgbel zd~vgpos
INTO TABLE  lt_delivery_data
FROM
   ( zv_error_docs AS ze INNER JOIN  zv_delivery_data AS zd ON ze~vbeln = zd~vbeln )
WHERE
  ze~msgno IN ro_msgno
  AND ze~posnr IN ro_posnr
  AND ze~vbeln IN ro_vbeln
  AND ze~sammg IN ro_sammg
  AND ze~smart = rp_ltype
  AND ze~erdat IN ro_cdate
  AND ze~ernam IN ro_ernam
  AND ze~ernum <> 0
  AND ze~vbtyp = 'J'
  AND zd~vkorg IN ro_vkorg.
```

Example 2 – Pushing code to DB layer

- Original code as in example 1

```
FORM delivery_data USING ro_vkorg         TYPE type_t_vkorg
                         rt_error_docs_de TYPE type_t_error_docs_de
                CHANGING rt_header_de     TYPE type_t_header_de
                         rt_item_de       TYPE type_t_item_de.

 IF NOT rt_error_docs_de[] IS INITIAL.
   SELECT kunnr
          vbeln
          netwr
          waerk
          vkorg
     FROM likp
     INTO TABLE rt_header_de
      FOR ALL ENTRIES IN rt_error_docs_de
    WHERE vbeln = rt_error_docs_de-vbeln
"       AND vkorg IN ro_vkorg.
      AND vkorg IN ro_vkorg.
   IF sy-subrc = 0.

     SELECT vbeln
            posnr
            netwr
            pstyv
            spart
            vgbel
            vgpos
       FROM lips
       INTO TABLE rt_item_de
        FOR ALL ENTRIES IN rt_header_de
      WHERE vbeln = rt_header_de-vbeln.
       IF sy-subrc <> 0.
         MESSAGE I219 WITH 'LIPS'(014).
       ENDIF.
     ELSE.
       MESSAGE I034 WITH 'Delivery'(015).
     ENDIF.
  ENDIF.
ENDFORM.                    " delivery_data
```

Example 2 – Pushing code to DB layer (2)

- Define procedure parameters

Database Procedure Proxy: ZDP_POC

General Attributes

HANA Procedure: HDB.zpoc.ZDP_POC

Database Procedure Interface: ZIF_ZDP_POC

Parameters

HANA Name	HANA Type	ABAP Name	ABAP Type
RT_ERROR_DOCS_DE	HDB.zpoc/ZDP_POC/tabletype/rt_error_docs_de	RT_ERROR_DOCS_DE	
VBELN	CHAR(10)	VBELN	C length 10
POSNR	DECIMAL(6)	POSNR	P length 4 decimals 0
SAMMG	CHAR(1)	SAMMG	C length 1
MSGID	CHAR(20)	MSGID	C length 20
MSGNO	CHAR(3)	MSGNO	C length 3
MSGV1	CHAR(50)	MSGV1	C length 50
MSGV2	CHAR(50)	MSGV2	C length 50
MSGV3	CHAR(50)	MSGV3	C length 50
MSGV4	CHAR(50)	MSGV4	C length 50
VBTYP	CHAR(1)	VBTYP	C length 1
IV_VKORG	CHAR(4)	IV_VKORG	C length 4
ET_HEADER_DE	HDB.zpoc/ZDP_POC/tabletype/et_header_de	ET_HEADER_DE	
KUNNR	CHAR(10)	KUNNR	C length 10
VBELN	CHAR(1)	VBELN	C length 1
NETWRH	DECIMAL(15,2)	NETWRH	P length 8 decimals 2
WAERK	CHAR(5)	WAERK	C length 5
VKORG	CHAR(4)	VKORG	C length 4
ET_ITEM_DE	HDB.zpoc/ZDP_POC/tabletype/et_item_de	ET_ITEM_DE	
VBELN	CHAR(10)	VBELN	C length 10
POSNR	DECIMAL(6)	POSNR	P length 4 decimals 0
NETWRI	DECIMAL(15,2)	NETWRI	P length 8 decimals 2
PSTYV	CHAR(4)	PSTYV	C length 4
SPART	CHAR(2)	SPART	C length 2
VGBEL	CHAR(10)	VGBEL	C length 10
VGPOS	CHAR(6)	VGPOS	C length 6

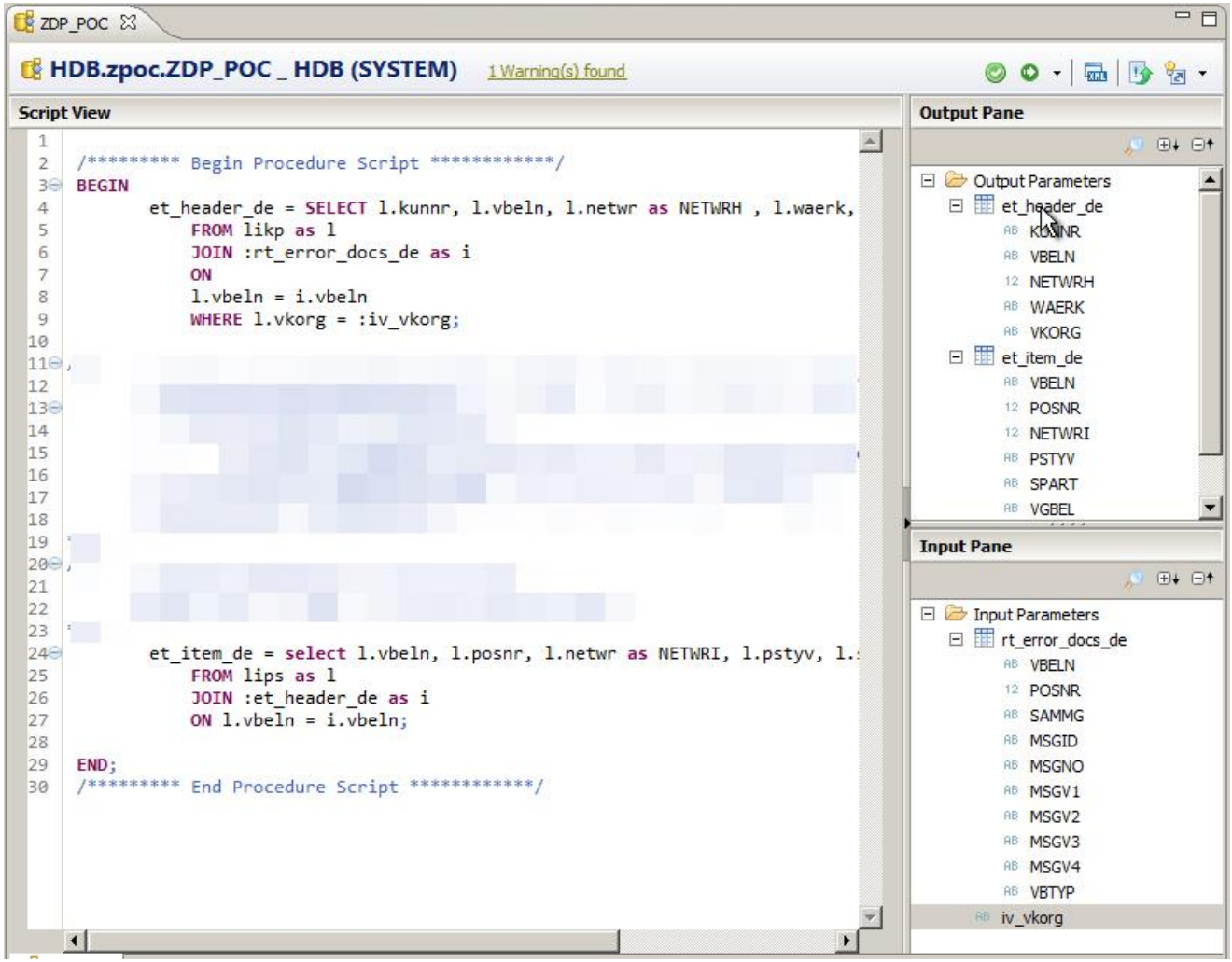
Example 2 – Pushing code to DB layer (3)

- Generated interface object, to be used in application logic

```
interface ZIF_ZDP_POC public.

" This interface pool has been generated.
" It contains the type definitions for
" database procedure proxy ZDP_POC
" representing db procedure _SYS_BIC.HDB.zpoc/ZDP_POC

types: begin of rt_error_docs_de,
         vbeln                          type c length 10,
         posnr                          type p length 4 decimals 0,
         sammg                          type c length 1,
         msgid                          type c length 20,
         msgno                          type c length 3,
         msgv1                          type c length 50,
         msgv2                          type c length 50,
         msgv3                          type c length 50,
         msgv4                          type c length 50,
         vbtyp                          type c length 1,
       end of rt_error_docs_de.
types: iv_vkorg                        type c length 4.
types: begin of et_header_de,
         kunnr                          type c length 10,
         vbeln                          type c length 1,
         netwrh                         type p length 8 decimals 2,
         waerk                          type c length 5,
         vkorg                          type c length 4,
       end of et_header_de.
types: begin of et_item_de,
         vbeln                          type c length 10,
         posnr                          type p length 4 decimals 0,
         netwri                         type p length 8 decimals 2,
         pstyv                          type c length 4,
         spart                          type c length 2,
         vgbel                          type c length 10,
         vgpos                          type c length 6,
       end of et_item_de.

endinterface .
```

Example 2 – Pushing code to DB layer (4)

- SQL script that implements reading from relevant tables



Example 2 – Pushing code to DB layer (5)

- Modify code to use interface / call procedure

```
58    DATA: lt_delivery_data TYPE TABLE OF zv_delivery_data.
59    DATA: ls_delivery_data LIKE LINE OF lt_delivery_data.
60    DATA: ls_item_de LIKE LINE OF rt_item_de.
61
62    DATA: lv_vkorg TYPE zif_zdp_poc=>iv_vkorg VALUE '1000'.
63
64    DATA: ls_error_docs_de LIKE LINE OF rt_error_docs_de.
65    DATA: ls_header_de LIKE LINE OF rt_header_de.
66
67    DATA: lt_error_docs_de_db TYPE STANDARD TABLE OF zif_zdp_poc=>rt_error_docs_de.
68    DATA: ls_error_docs_de_db TYPE zif_zdp_poc=>rt_error_docs_de.
69
70    DATA: ls_header_de_db TYPE zif_zdp_poc=>et_header_de.
71    DATA: lt_header_de_db TYPE STANDARD TABLE OF zif_zdp_poc=>et_header_de.
72
73    DATA: ls_item_de_db TYPE zif_zdp_poc=>et_item_de.
74    DATA: lt_item_de_db TYPE STANDARD TABLE OF zif_zdp_poc=>et_item_de.
75
76  IF NOT rt_error_docs_de[] IS INITIAL.
77
78
79
80
81
82
83
84
85
86
87
88    CALL DATABASE PROCEDURE zdp_poc
89          EXPORTING
90              iv_vkorg = lv_vkorg
91              rt_error_docs_de = lt_error_docs_de_db
92          IMPORTING
93              et_header_de = lt_header_de_db
94              et_item_de = lt_item_de_db
95              .
96    IF sy-subrc <> 0.
97        MESSAGE I034 WITH 'Failure'(015).
98    ELSE.
```

New ABAP language feature!

NetWeaver 7.4 ABAP Enhancements

- Example 2 demonstrated a new ABAP language feature:
  - CALL PROCEDURE
- ABAP for HANA introduces a new usage pattern for a well known and commonly used programming feature:
  - ALV Grids
- True to the motto “push code down to db layer”, the new ALV grid model, also called IDA-ALV (inegrated data access list viewer), implements much of the processing in the DB layer
- Example from NW 7.4 SP0 follows. Note that SP2 adds some features not available in SP0:
  - Setting field header texts
  - Fuzzy text search

NetWeaver 7.4 ABAP Enhancements (cont.)

- Internal Tables with Empty Keys
- New Internal Table Functions
- ABAP Objects (Exporting, Importing and Changing, Partially Implemented Interfaces for Testing)
- Table Expressions
- Conditional Operators
- Lossless Operator EXACT
- Conversion Operators
- Reference Operator
- Value Operator
- Constructor Operator
- Inline Declarations
- Expressions

Use of Integrated Data Access List Grid (IDA-ALV)

```
MODULE pbo OUTPUT.
    IF g_custom_container IS INITIAL.
        CREATE OBJECT g_custom_container
            EXPORTING
                container_name = g_container.

*       Instantiate IDA_ALV (ALV with IDA=Integrated Data Access)
        data(lo_alv_display) = cl_salv_gui_table_ida=>create( iv_table_name = 'ZV_DELIVERY_DATA'
                                                              io_gui_container = g_custom_container ).

        data(lo_collector) = new cl_salv_range_tab_collector( ).  "Helper Class
        lo_collector->add_ranges_for_name( iv_name = 'VKORG' it_ranges = o_vkorg[] ).
        lo_collector->get_collected_ranges( IMPORTING et_named_ranges = data(lt_name_range_pairs) ).
        lo_alv_display->set_select_options( it_ranges = lt_name_range_pairs ).


*       Initial Grouping
        lo_alv_display->default_layout( )->set_sort_order( value #( ( field_name = 'VKORG' is_grouped = abap_true )
                                                                    ( field_name = 'SPART' is_grouped = abap_true )
                                                                    ( field_name = 'VBELN' is_grouped = abap_true ) ) ).

    ENDIF.
ENDMODULE.
```

Access a view

Add applicable ranges

Define sort order

smartScale for HANA Offering

smartScale for HANA will consists of the appropriate components with newly added HANA Code Optimization Ruleset.

- smartAnalyze
  - SaaS Analysis
- smartUpgrade
  - Unicode Enablement
  - Change Impact Analysis
- smartDevelop
  - On-going ABAP Code Remediation

Q & A

Contact

Albrecht Gass

Chief Architect, smartShift Technologies

www.smartShiftTech.com

SCN

LinkedIn

@AlbrechtGass

Where to start?

Get Started with Your HANA Proof of Concept

Further Resources

Online Resources

High-level Info

- Main Landing Page
  http://www.saphana.com/welcome
- HANA Academy (Instructional Videos)
  http://www.saphana.com/community/implement/hana-academy
- SAP HANA One Cloud
  http://www.saphana.com/community/learn/cloud
- SAP HANA Marketplace
  http://www.saphana.com/community/marketplace

Online Resources

Developer Level

- **SAP community center**
  It is highly recommended to join SCN (SAP Community Network, www.scn.com) if you have not done so. Besides a dedicated HANA space (http://scn.sap.com/community/hana-in-memory) you can also find interesting information on related spaces (e.g. NW 7.4, mobility, cloud computing).
- **SAP HANA Developer Center**
  If you are a developer and you would like to get some first hands-on experience you should check the SAP HANA Developer Center (http://scn.sap.com/community/developer-center/hana). You will get more information on how to sign up for a trial cloud instance and links to HANA client and studio software for modeling (https://hanadeveditionsapicl.hana.ondemand.com/hanadevedition). Again, a resource highly recommended for developers. Please do check from time to time for the latest revision (yes, it makes a difference).
- **openSAP**
  Massive Open Online Course for HANA at https://open.sap.com/

Online References

NW 7.4 ABAP Enhancements

- Blog Series by Host Keller
  http://scn.sap.com/people/horst.keller/blog
- Netweaver Blog
  http://scn.sap.com/community/netweaver/blog
- NW 7.4 Landing Page
  http://scn.sap.com/docs/DOC-35002

Online References

- **Where to get sample data?**
  - **ERP data / SFLIGHT**
    Please refer to http://scn.sap.com/docs/DOC-32854 (needs SCN account)
  - **Open Items Analytics**
    Please refer to http://scn.sap.com/docs/DOC-35518 (needs SCN account)
  - **Other large data**
    Should you need large amounts of data to play with you can also look at http://www.infochimps.com/datasets. They come as CSV files which you can easily import into your HANA instance.