Week 4 Unit 1: **Native SAP HANA Usage in ABAP**

SAP

openSAP

Native SAP HANA Usage in ABAP
Outline

Content

- Native SQL via ABAP Database Connectivity (ADBC)
- Consumption of SAP HANA Procedures
- Sneak Preview: ABAP Managed Database Procedure Consumption

Native SAP HANA Usage in ABAP

Migration to SAP HANA

1. Detect

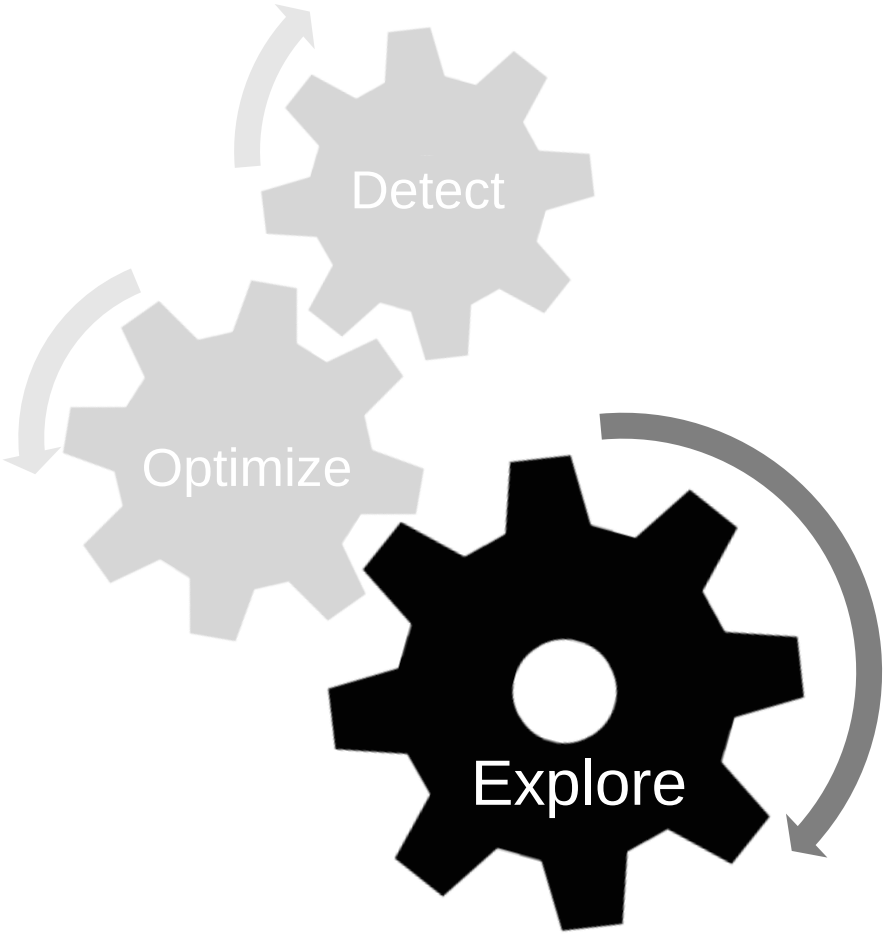
- Functional correctness
- Performance optimization potential

2. Optimize

- Database-oriented programming

3. Explore

- Use SAP HANA-specific features
- Rethink & innovate

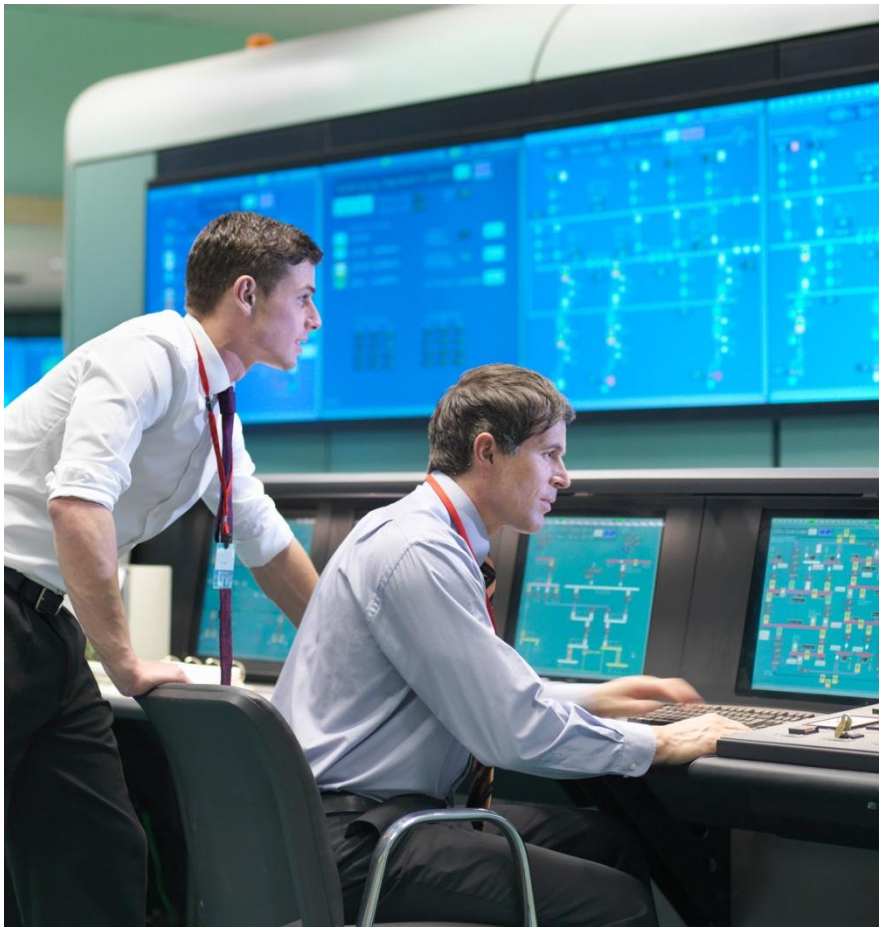


Native SAP HANA Usage in ABAP

ABAP Database Connectivity (ADBC)

Object-based ABAP API for programming relational database access

- Access to entire SQL functionality
- Dynamic creation and execution of native SQL statements
- Clean concept for multiple DB connections
- Exception handling

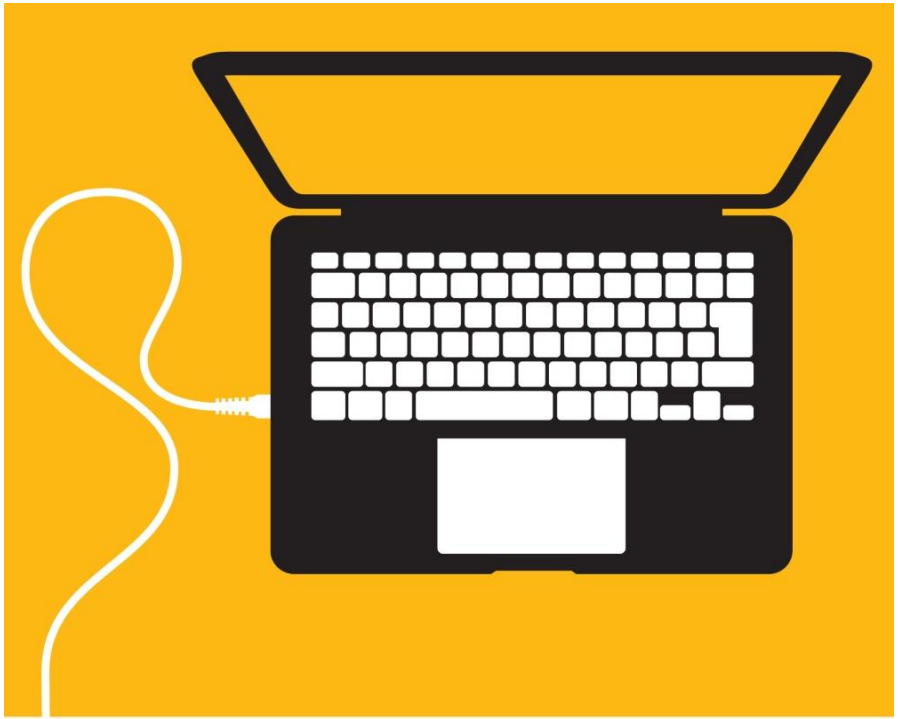


Native SAP HANA Usage in ABAP

ABAP Database Connectivity: Demo

ABAP Database Connectivity (ADBC)

- Typical code / call sequence
- Call of an SAP HANA database procedure



More information

- See demo reports ADBC_DEMO and ADBC_DEMO_PROC_CALLS_HDB

Native SAP HANA Usage in ABAP

ABAP Database Connectivity: Typical Code

Additional Material

Typical call sequence

- Target type definition / data declarations
- Concatenation / Definition of the DBSYS-dependent SQL Query
- Create statement object (and DB connection if necessary)
- Execute query, passing SQL query as string
- Assign internal table for query result
- Retrieve result
- Close result set / free allocated resources

```
REPORT zr_adbc_simple.

TYPES:
  BEGIN OF ty_res,
    bp_id              TYPE snwd_bpa-bp_id,
    company_name       TYPE snwd_bpa-company_name,
    currency_code      TYPE snwd_so-currency_code,
    total_gross_amount TYPE snwd_so-gross_amount,
  END OF ty_res.

DATA lv_stmt   TYPE string.
DATA lo_stmt   TYPE REF TO cl_sql_statement.
DATA lo_res    TYPE REF TO cl_sql_result_set.
DATA lt_result TYPE STANDARD TABLE OF ty_res WITH EMPTY KEY.

lv_stmt = |SELECT BP_ID, COMPANY_NAME, SO.CURRENCY_CODE,        | &&
          |       SUM( SO.GROSS_AMOUNT ) as TOTAL_GROSS_AMOUNT | &&
          | FROM SNWD_BPA AS BPA                               | &&
          | INNER JOIN SNWD_SO AS SO                           | &&
          |   ON SO.BUYER_GUID = BPA.NODE_KEY                  | &&
          | GROUP BY BP_ID, COMPANY_NAME, SO.CURRENCY_CODE     |.

TRY.
    lo_stmt = NEW cl_sql_statement( ).
    lo_res  = lo_stmt->execute_query( lv_stmt ).
    lo_res->set_param_table( ref #( lt_result ) ).
    lo_res->next_package( ).
    lo_res->close( ).

    CATCH cx_sql_exception INTO DATA(lx).
    "do some meaningful error handling
    WRITE: lx->sql_message.
ENDTRY.
```

Native SAP HANA Usage in ABAP

Native SQL Pitfalls / Disadvantages

No syntax check at compile time

- SQL query is passed as string

Developer responsible for

- Client handling, accessing correct database schema
- Type mapping, for example, of internal table for result retrieval
- Releasing DB resources
- Proper locking and commit handling

ABAP table buffers bypassed

Coding is database-dependent

- Remember the ABAP Test Cockpit check on Native SQL usage

Native SAP HANA Usage in ABAP

Consumption of an SAP HANA Database Procedure with ADBC

Additional Material

Consumption

- Consumption of SAP HANA procedures via „CALL <PROCEDURE>“
- Cumbersome, especially for procedures with (several) input / output parameters

More Information

- Demo report ADBC_DEMO_PROC_CALLS_HDB

```
REPORT zr_adbc_proc_call.
"lots of type definitions & data declarations
lv_stmt =
  | CALL "SAPHANAABAP"."ZCL_DEMO_PAID_ON_DATE_AMDP=>PAID_ON_DATE" | &&
  |                ( '20140401', NULL, NULL, NULL ) WITH OVERVIEW |.

TRY.
    lo_stmt = NEW cl_sql_statement( ).
    lo_res  = lo_stmt->execute_query( lv_stmt ).
    "get a reference of the overview table and prepare the result set
    lo_res->set_param_table( REF #( lt_overview ) ).
    "retrieve the overview table
    lo_res->next_package( ).
    LOOP AT lt_overview INTO DATA(ls_overview).
      "select from the corresponding DB table listed in the overview table
      DATA(lo_res_tab) =
        lo_stmt->execute_query( | select * from { ls_overview-table }| ).
      IF ls_overview-value CS 'ET_INVOICE_ITEM'.
        "prepare the result set
        lo_res_tab->set_param_table( REF #( lt_inv_item ) ).
      ELSEIF ls_overview-value CS 'ET_INVOICE_HEAD'.
        "prepare the result set
        lo_res_tab->set_param_table( REF #( lt_inv_head ) ).
      ELSEIF ls_overview-value CS 'ET_CUSTOMER_INFO'.
        "prepare the result set
        lo_res_tab->set_param_table( REF #( lt_cust_info ) ).
      ENDIF.
      lo_res_tab->next_package( ).
    ENDLOOP.
    lo_res->close( ).
  CATCH cx_sql_exception INTO DATA(lx).
    "do some meaningful error handling
ENDTRY.
```

Native SAP HANA Usage in ABAP

Consumption of an ABAP Managed Database Procedure

Consumption

- Consumption of an ABAP Managed Database Procedure

More Information

- Upcoming unit – stay tuned ☺

```
REPORT zr_amdp_call.

DATA(lo_info_list) = NEW zcl_demo_paid_on_date_amdp( ).

lo_info_list->paid_on_date(
  EXPORTING
    iv_payment_date = '20140401'
  IMPORTING
    et_customer_info  = DATA(lt_cust_info_amdp)
    et_invoice_header = DATA(lt_inv_head_amdp)
    et_invoice_item   = DATA(lt_inv_item_amdp) ).
```

Native SAP HANA Usage in ABAP

What's Next?

Thank you

Contact information:

open@sap.com

openSAP

© 2014 SAP AG or an SAP affiliate company. All rights reserved.