**LIT TRAINING**

**Batch Name : SAP ABAP ON HANA** **DAY 14 ASSIGNMENT**

**EMPLOYEE ID – 46255260**

**CDS-Assignment 1 – Create a Simple CDS view with Date Functions in ABAP CDS Views**

**CODE:**

@AbapCatalog.sqlViewName: 'ZLITVIEW'

@AbapCatalog.compiler.compareFilter: true

@AbapCatalog.preserveKey: true

@AccessControl.authorizationCheck: #NOT\_REQUIRED

@EndUserText.label: 'CDS VIEW OF LIT'

**define** **view** Z46255244LIT\_CDS **as** **select** **from** vbak **{**

  vbeln **as** Sales\_Document**,**

  auart **as** Sales\_Document\_Type**,**

  audat **as** Document\_Date**,**

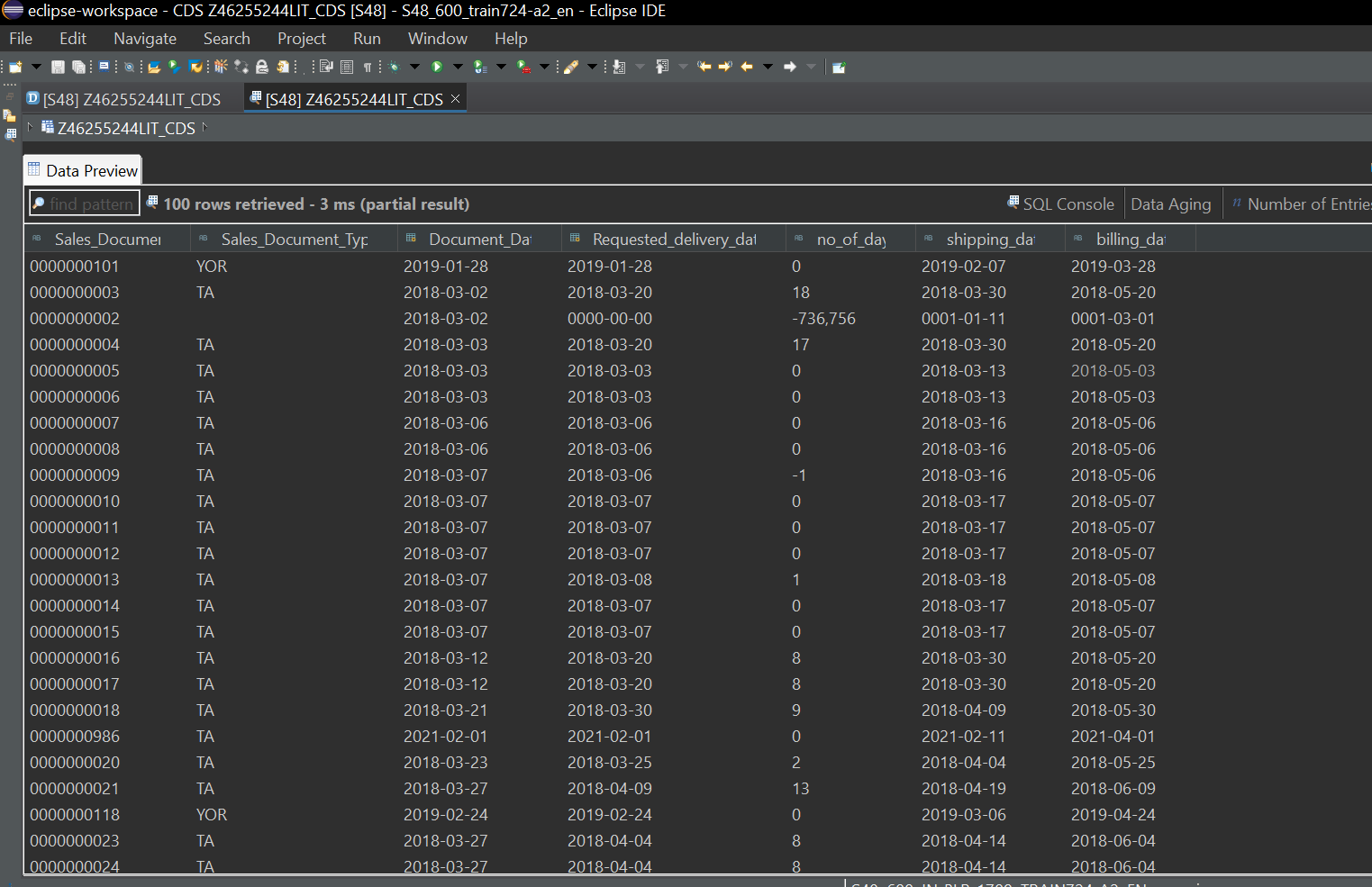
  vdatu **as** Requested\_delivery\_date**,**

  DATS\_DAYS\_BETWEEN**(**audat**,** vdatu**)** **as** no\_of\_days**,**

  DATS\_ADD\_DAYS**(**vdatu**,** 10**,** 'NULL'**)** **as** shipping\_date**,**

  DATS\_ADD\_MONTHS**(**vdatu**,** 2**,** 'NULL'**)** **as** billing\_date

**}**



CDS-Assignment 2 – Create a Simple CDS view with String Functions in ABAP CDS Views

CODE:

@AbapCatalog.sqlViewName: 'Z24\_STRING'

@AbapCatalog.compiler.compareFilter: true

@AbapCatalog.preserveKey: true

@AccessControl.authorizationCheck: #NOT\_REQUIRED

@EndUserText.label: 'CDS VIEW WITH STRING FUNCTIONS'

**define** **view** Z46255244LIT\_CDS1 **as** **select** **from** kna1 **{**

kunnr **as** Customer\_Number**,**

land1 **as** Country\_Key**,**

CONCAT**(**name1**,** name2**)** **as** Full\_Name**,**

CONCAT\_WITH\_SPACE**(**stras**,** ort01**,** 4 **)** **as** COMBO**,**

left**(**name1**,** 2**)** **as** F2C\_NAME1**,**

RIGHT**(**name1**,** 3**)** **as** L3C\_NAME2**,**

LENGTH**(**telf1**)** **as** NAME\_LENGTH**,**

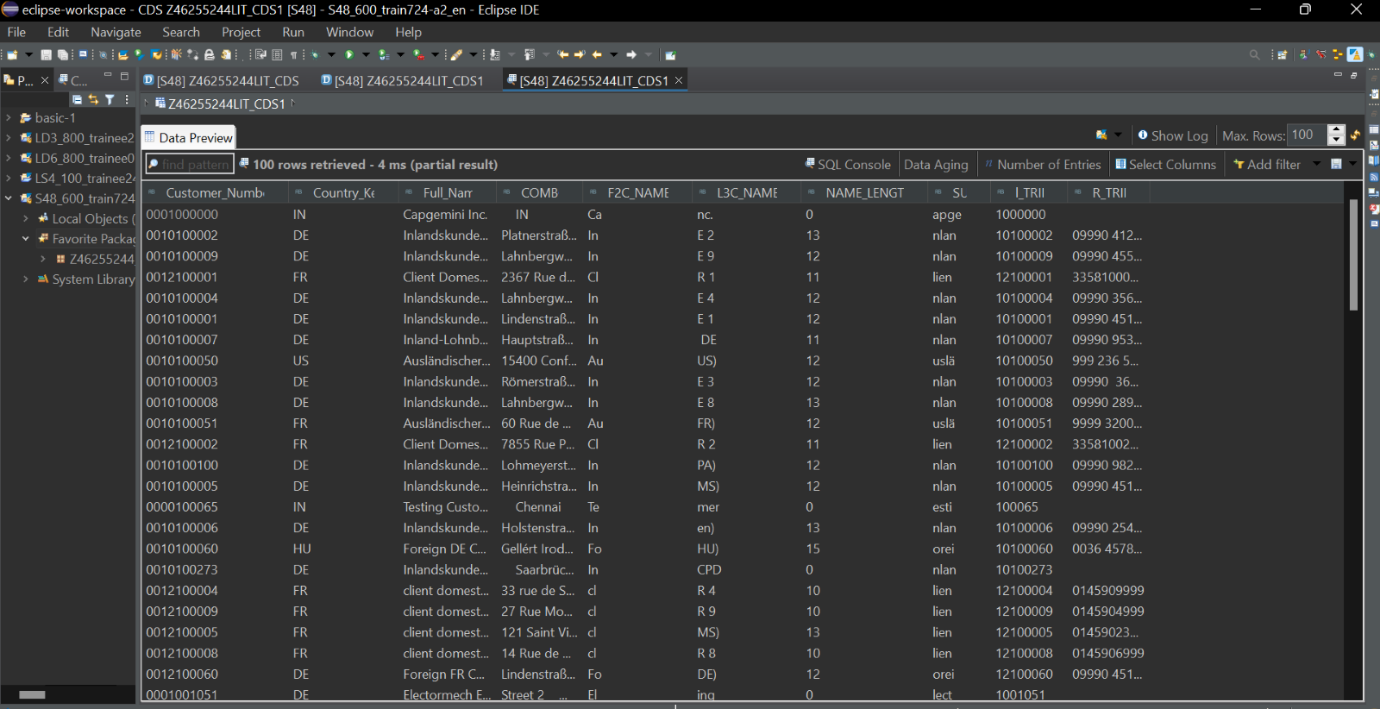
SUBSTRING**(**name1**,** 2**,** 4**)** **as** SUB**,**

LTRIM**(**kunnr**,** '0'**)** **as** l\_TRIM**,**

RTRIM**(**telf1**,** '4'**)**  **as**  R\_TRIM

**}**

**OUTPUT:**



CDS-Assignment 3 – Create a Simple CDS view with Parameters

CODE:

@AbapCatalog.sqlViewName: 'Z24\_PARMS'

@AbapCatalog.compiler.compareFilter: true

@AbapCatalog.preserveKey: true

@AccessControl.authorizationCheck: #NOT\_REQUIRED

@EndUserText.label: 'WITH PARAMETERS'

**define** **view** Z46255244LIT\_CDS2

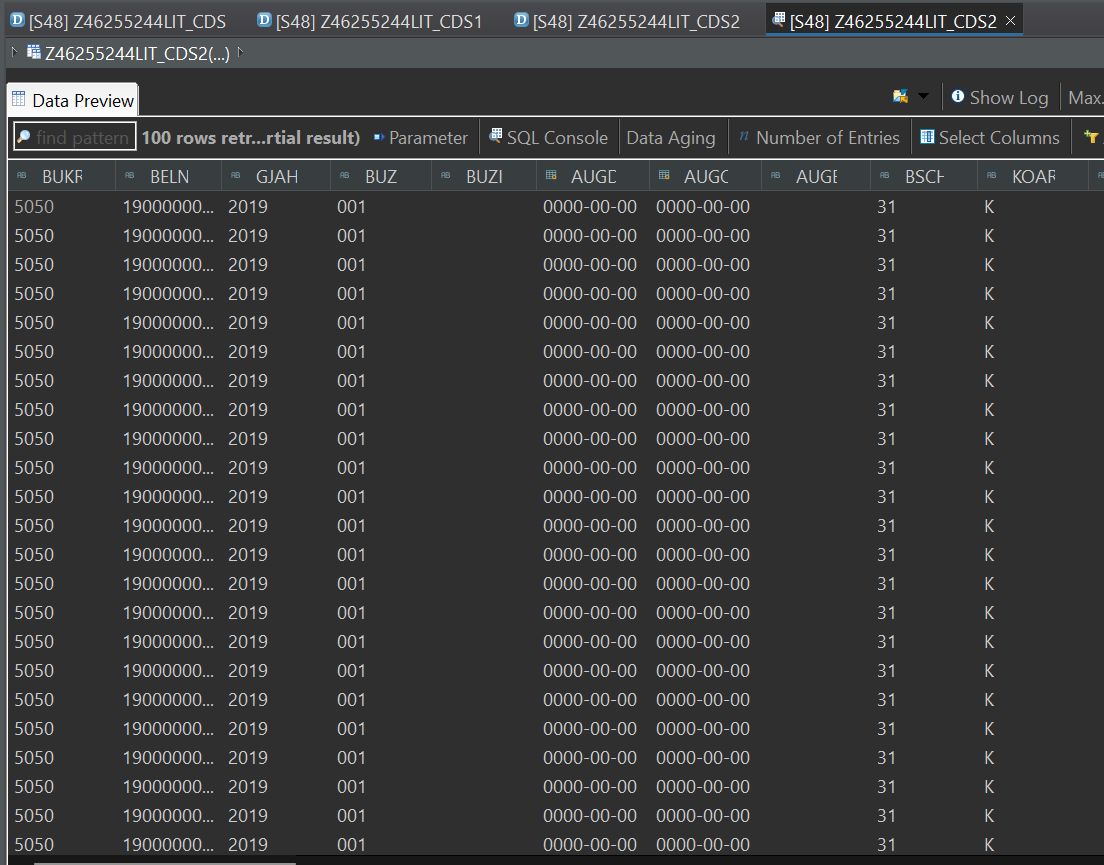
**with** **parameters** S\_PSWSL **:** *abap***.***cuky***(**5**)**

**as** **select** **from** bseg **{**

**\***

**}where** pswsl **=** **$parameters.**S\_PSWSL**;**

OUTPUT:



AMDP-Assignment 1 – Procedures - create a basic procedure to fetch data with scalar variable

**CODE:** **CLASS**

CLASS z46255244lit\_class DEFINITION public.

  PUBLIC SECTION.

  types: begin of s\_str,

        vbeln type vbap-vbeln,

        matnr type vbap-matnr,

        zmeng type vbap-zmeng,

        end of s\_str.

  types : t\_str type STANDARD table of S\_STR.

  interfaces : if\_amdp\_marker\_hdb.

  methods : get\_method exporting value(i\_out) type t\_str .

  endclass.

CLASS z46255244lit\_class IMPLEMENTATION.

method get\_method by database procedure for HDB

language sqlscript options read-only USING VBAP .

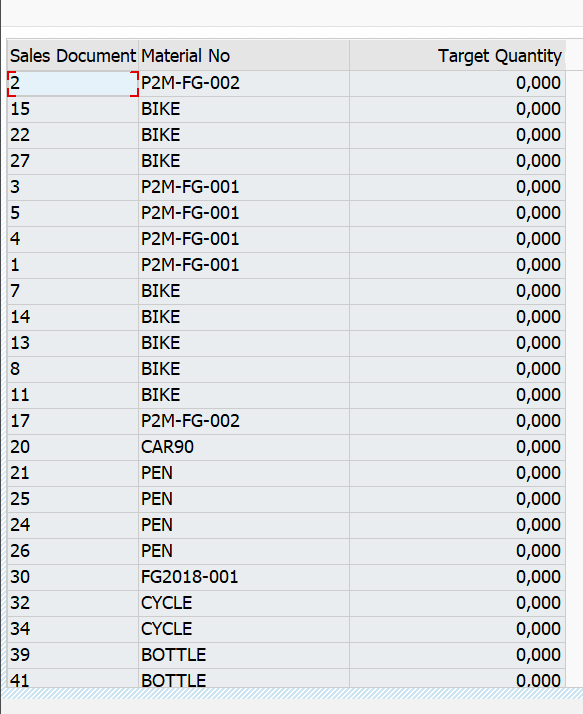
i\_out =

    select VBELN, MATNR, ZMENG FROM vbap;

endmethod.

ENDCLASS.

**PROGRAM**



AMDP-Assignment 2 – Procedures - create a basic procedure with Table

**CLASS:**

CLASS z46255244lit\_class1 DEFINITION PUBLIC.

  PUBLIC SECTION.

  types: begin of ty\_tab,

        vbeln type vbak-vbeln,

        vkorg type vbak-vkorg,

        matnr type vbap-matnr,

        zmeng type vbap-zmeng,

        end of ty\_tab.

  types : tt\_out type table of ty\_tab.

  interfaces : if\_amdp\_marker\_hdb.

methods : get\_method1

importing value(im\_matnr) type vbap-matnr

exporting value(I\_OUT) type tt\_out.

  endclass.

CLASS z46255244lit\_class1 IMPLEMENTATION.

method get\_method1 by database procedure for HDB

language sqlscript options read-only USING VBAP vbak .

i\_out =

    select a.VBELN, a.vkorg, b.matnr, b.zmeng FROM vbak as a inner join

    vbap as b on a.vbeln =  b.vbeln where b.matnr = im\_matnr;

endmethod.

ENDCLASS.

**SE38-CODE**

REPORT z46255244lit\_amdp1.

parameters : s\_matnr type vbap-matnr.

DATA : c\_obj TYPE REF TO Z46255244LIT\_CLASS1,

       t\_out TYPE REF TO cl\_salv\_table.

CREATE OBJECT c\_obj.

c\_obj->get\_method1(

  EXPORTING

    im\_matnr = s\_matnr

  IMPORTING

    i\_out    = data(ref)

).

TRY.

CALL METHOD cl\_salv\_table=>factory

IMPORTING

r\_salv\_table = t\_out

CHANGING

t\_table = ref

.

CATCH cx\_salv\_msg .

ENDTRY.

t\_out->display( ).

**OUTPUT:**

