NEW ABAP SYNTAX TIPS & EXPRESSIONS

INSTEAD OF CATCH CX_SY_ITAB_LINE_NOT_FOUND:

```
data(gs_booking) = value #( gt_booking
                                   [ 1 ] optional ).
```

QUICK DATA DISPLAY:

```
cl_demo_output=>write( gt_booking ).
cl_demo_output=>write( gs_booking ).
cl_demo_output=>display( ).
```

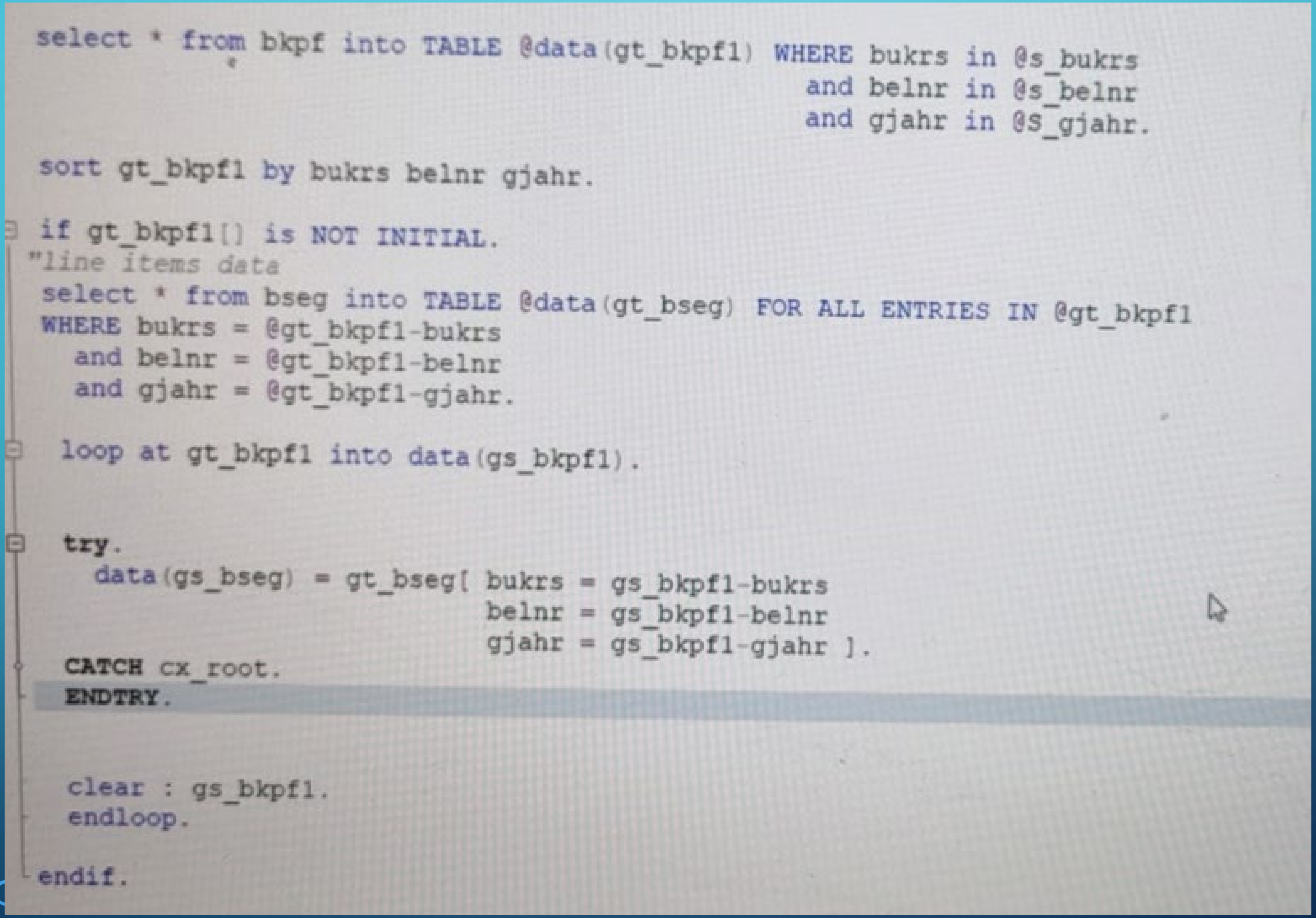
INSTEAD OF DESCRIBE TABLE…LINES:

```
data(gv_lines) = lines( gt_booking ). •
```

Read last row:

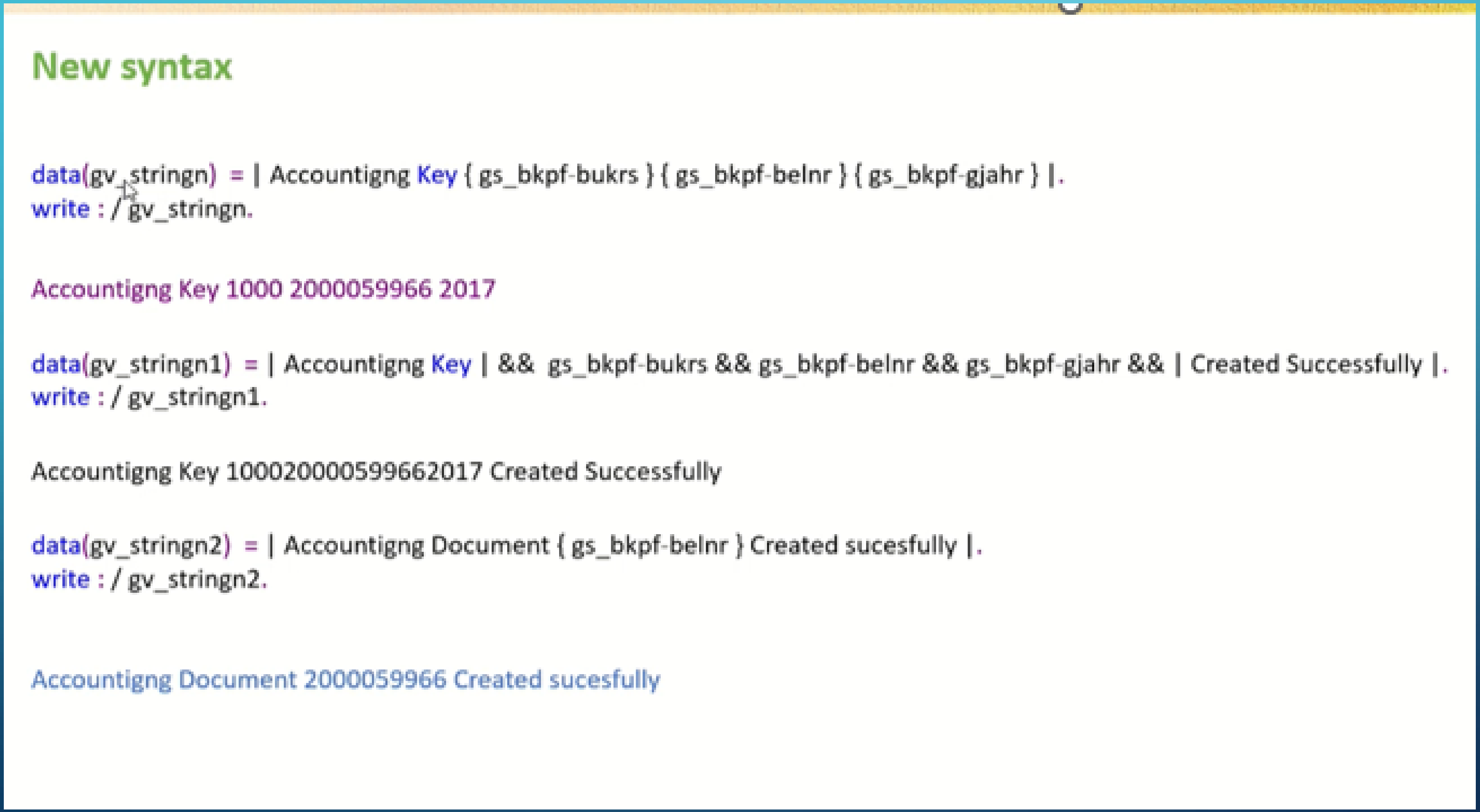
```
data(gs_booking1) = value #( gt_booking[ gv_lines ] optional ).
```

USING READ TABLES:



```
select * from bkpf into TABLE @data(gt_bkpf1) WHERE bukrs in @s_bukrs
                                                and belnr in @s_belnr
                                                and gjahr in @s_gjahr.

sort gt_bkpf1 by bukrs belnr gjahr.

if gt_bkpf1[] is NOT INITIAL.
"line items data
select * from bseg into TABLE @data(gt_bseg) FOR ALL ENTRIES IN @gt_bkpf1
WHERE bukrs = @gt_bkpf1-bukrs
  and belnr = @gt_bkpf1-belnr
  and gjahr = @gt_bkpf1-gjahr.

 loop at gt_bkpf1 into data(gs_bkpf1).


  try.
    data(gs_bseg) = gt_bseg[ bukrs = gs_bkpf1-bukrs
                             belnr = gs_bkpf1-belnr
                             gjahr = gs_bkpf1-gjahr ].
  CATCH cx_root.
  ENDTRY.


  clear : gs_bkpf1.
  endloop.

endif.
```

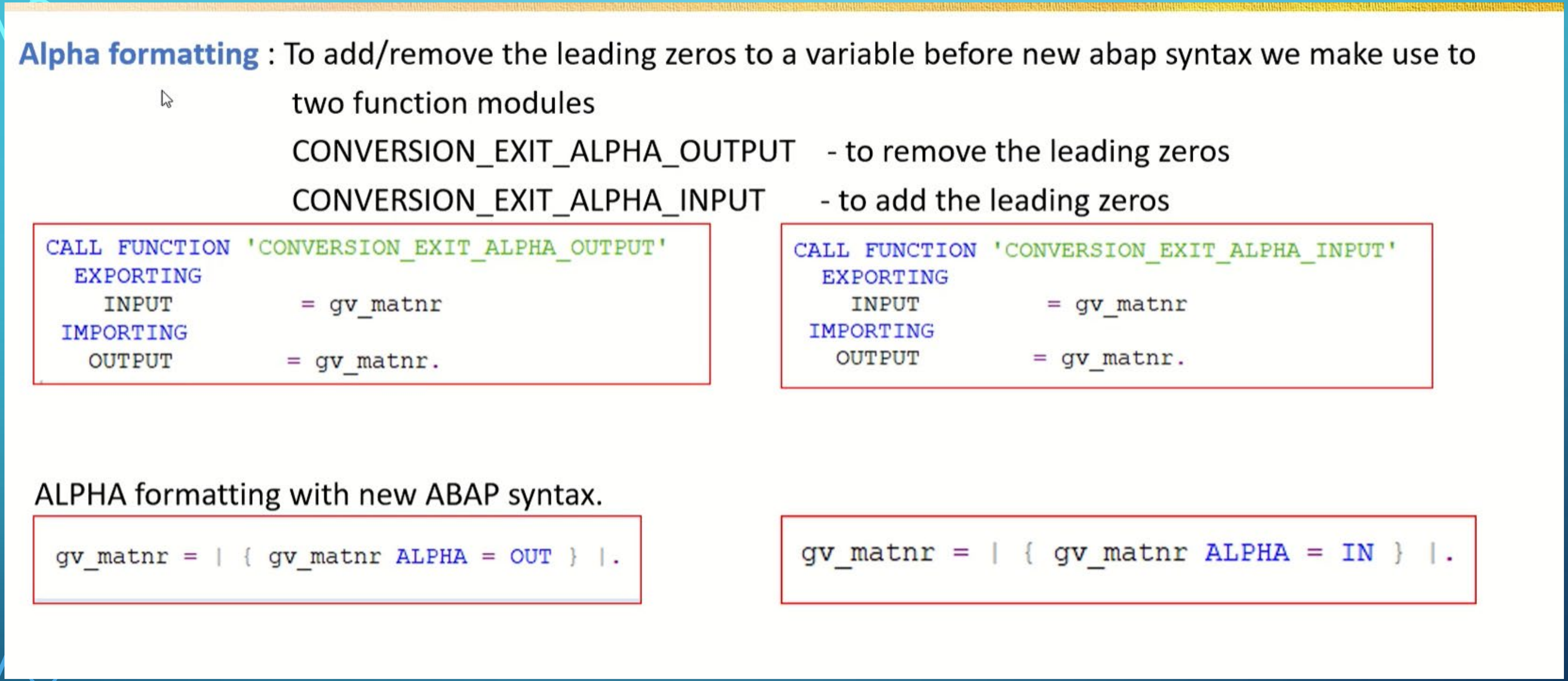
NEW CONCATENATE SYNTAX:



New syntax

```
data(gv_stringn)  = | Accountigng Key { gs_bkpf-bukrs } { gs_bkpf-belnr } { gs_bkpf-gjahr } |.
write : / gv_stringn.

Accountigng Key 1000 2000059966 2017

data(gv_stringn1)  = | Accountigng Key | &&  gs_bkpf-bukrs && gs_bkpf-belnr && gs_bkpf-gjahr && | Created Successfully |.
write : / gv_stringn1.

Accountigng Key 100020000599662017 Created Successfully

data(gv_stringn2)  = | Accountigng Document { gs_bkpf-belnr } Created sucesfully |.
write : / gv_stringn2.

Accountigng Document 2000059966 Created sucesfully
```

FORMATTING



Alpha formatting : To add/remove the leading zeros to a variable before new abap syntax we make use to two function modules

CONVERSION_EXIT_ALPHA_OUTPUT - to remove the leading zeros

CONVERSION_EXIT_ALPHA_INPUT - to add the leading zeros

```
CALL FUNCTION 'CONVERSION_EXIT_ALPHA_OUTPUT'
  EXPORTING
    INPUT          = gv_matnr
 IMPORTING
   OUTPUT         = gv_matnr.
```

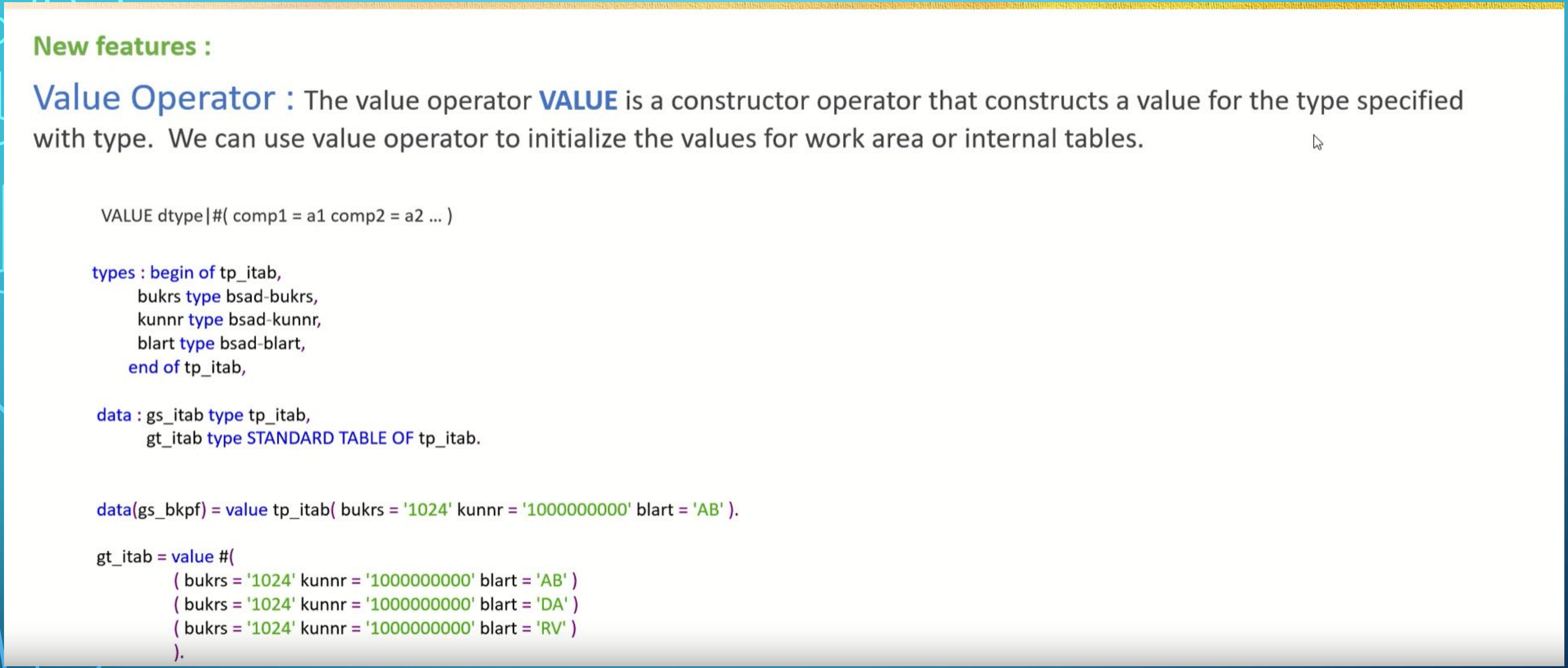
```
CALL FUNCTION 'CONVERSION_EXIT_ALPHA_INPUT'
  EXPORTING
    INPUT          = gv_matnr
 IMPORTING
   OUTPUT         = gv_matnr.
```

ALPHA formatting with new ABAP syntax.

```
gv_matnr = | { gv_matnr ALPHA = OUT } |.
```

```
gv_matnr = | { gv_matnr ALPHA = IN } |.
```

VALUE OPERATOR

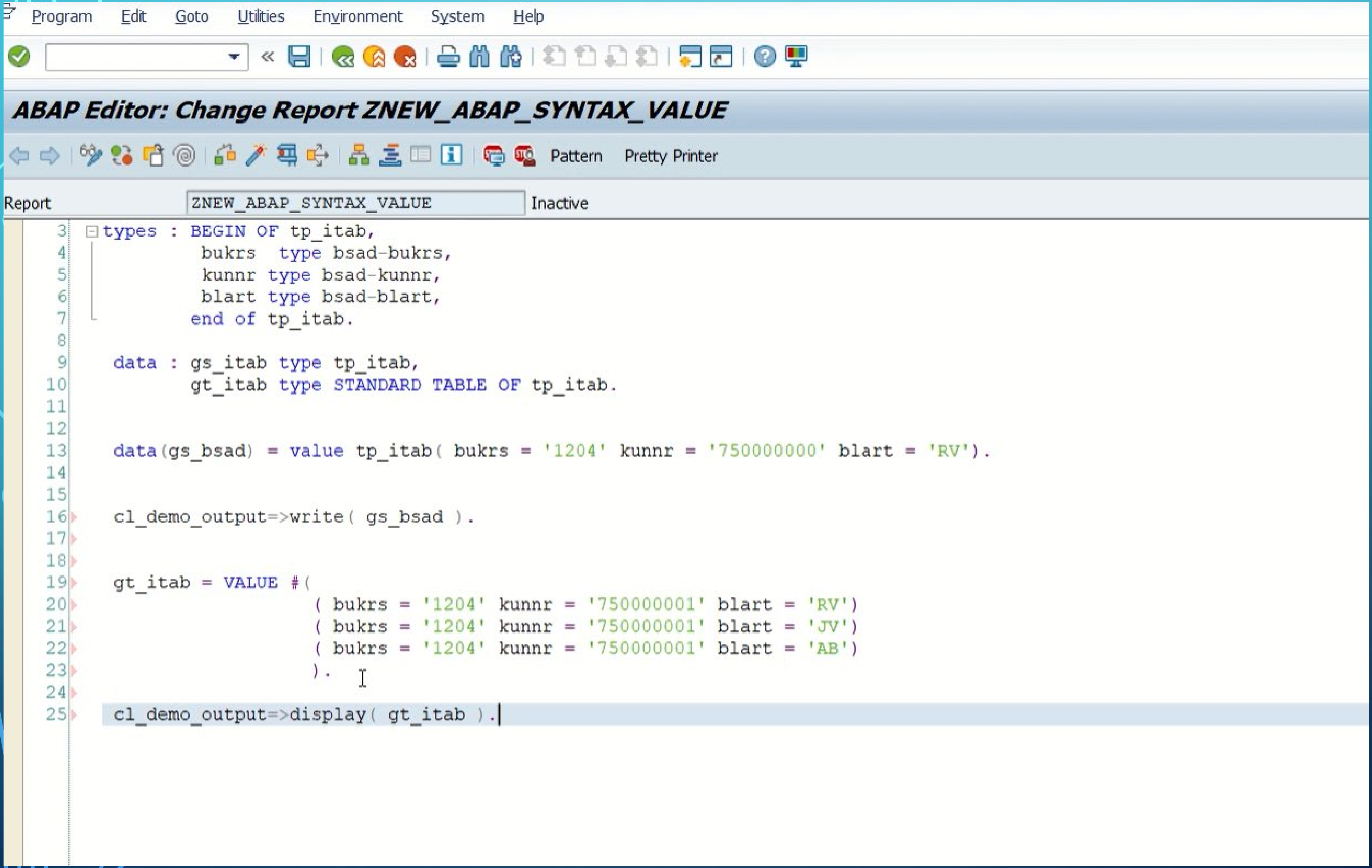


New features :

Value Operator : The value operator **VALUE** is a constructor operator that constructs a value for the type specified with type. We can use value operator to initialize the values for work area or internal tables.

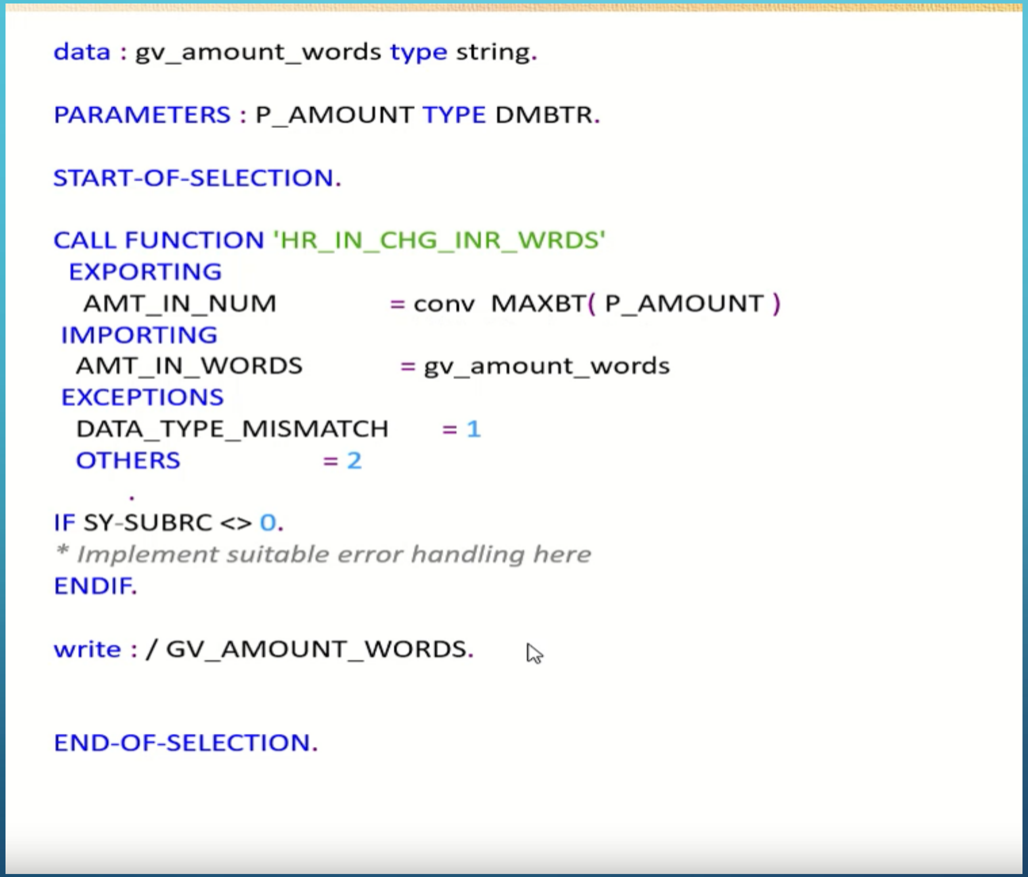
```
VALUE dtype|#( comp1 = a1 comp2 = a2 ... )

types : begin of tp_itab,
        bukrs type bsad-bukrs,
        kunnr type bsad-kunnr,
        blart type bsad-blart,
      end of tp_itab,

data : gs_itab type tp_itab,
       gt_itab type STANDARD TABLE OF tp_itab.


data(gs_bkpf) = value tp_itab( bukrs = '1024' kunnr = '1000000000' blart = 'AB' ).

gt_itab = value #(
            ( bukrs = '1024' kunnr = '1000000000' blart = 'AB' )
            ( bukrs = '1024' kunnr = '1000000000' blart = 'DA' )
            ( bukrs = '1024' kunnr = '1000000000' blart = 'RV' )
            ).
```

VALUE OPERATOR


Program Edit Goto Utilities Environment System Help
ABAP Editor: Change Report ZNEW_ABAP_SYNTAX_VALUE
Pattern Pretty Printer
Report ZNEW_ABAP_SYNTAX_VALUE Inactive

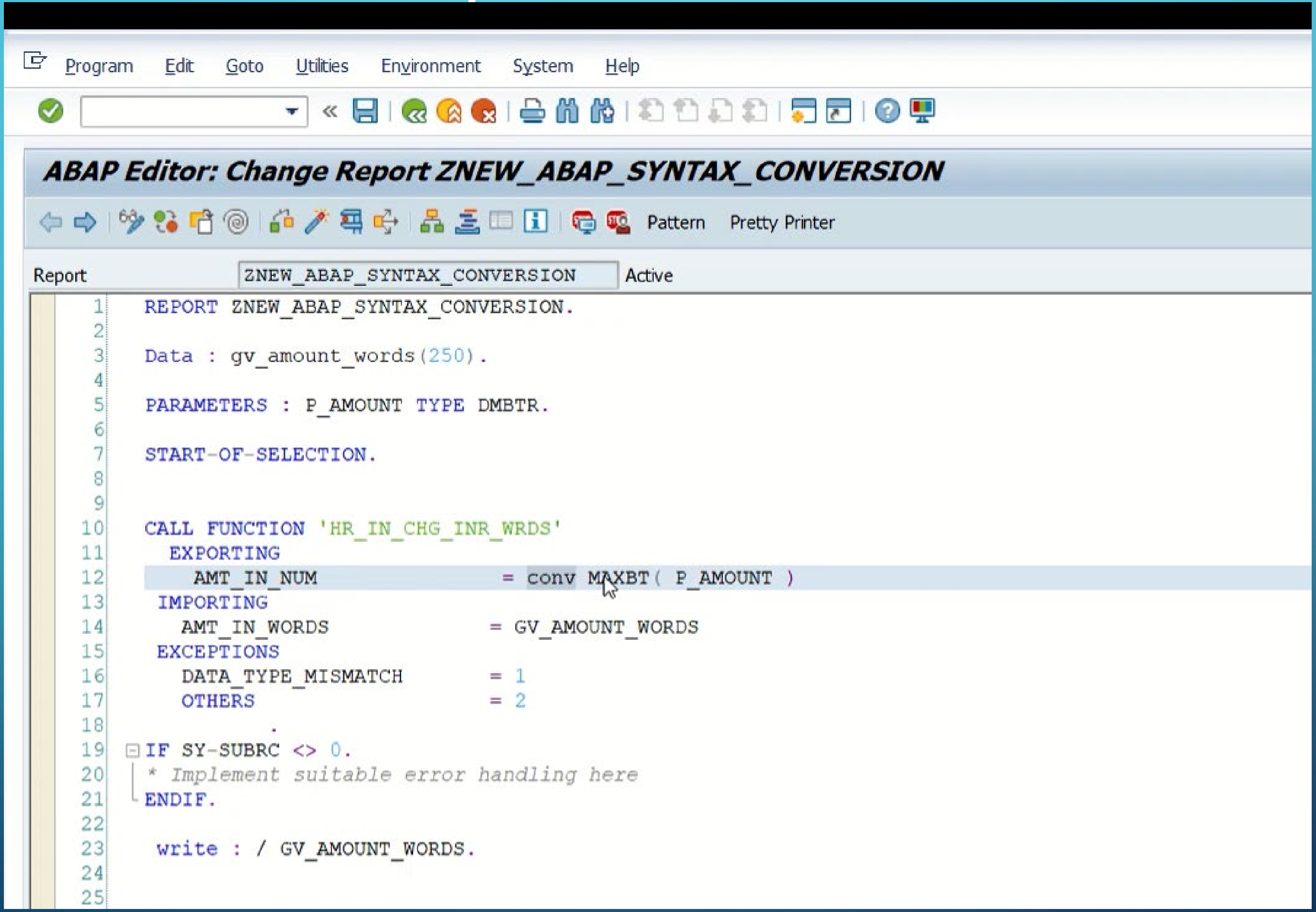
```
types : BEGIN OF tp_itab,
         bukrs  type bsad-bukrs,
         kunnr type bsad-kunnr,
         blart type bsad-blart,
        end of tp_itab.

data : gs_itab type tp_itab,
       gt_itab type STANDARD TABLE OF tp_itab.


data(gs_bsad) = value tp_itab( bukrs = '1204' kunnr = '750000000' blart = 'RV').


cl_demo_output=>write( gs_bsad ).


gt_itab = VALUE #(
                   ( bukrs = '1204' kunnr = '750000001' blart = 'RV')
                   ( bukrs = '1204' kunnr = '750000001' blart = 'JV')
                   ( bukrs = '1204' kunnr = '750000001' blart = 'AB')
                   ).

cl_demo_output=>display( gt_itab ).
```

CONV OPERATOR



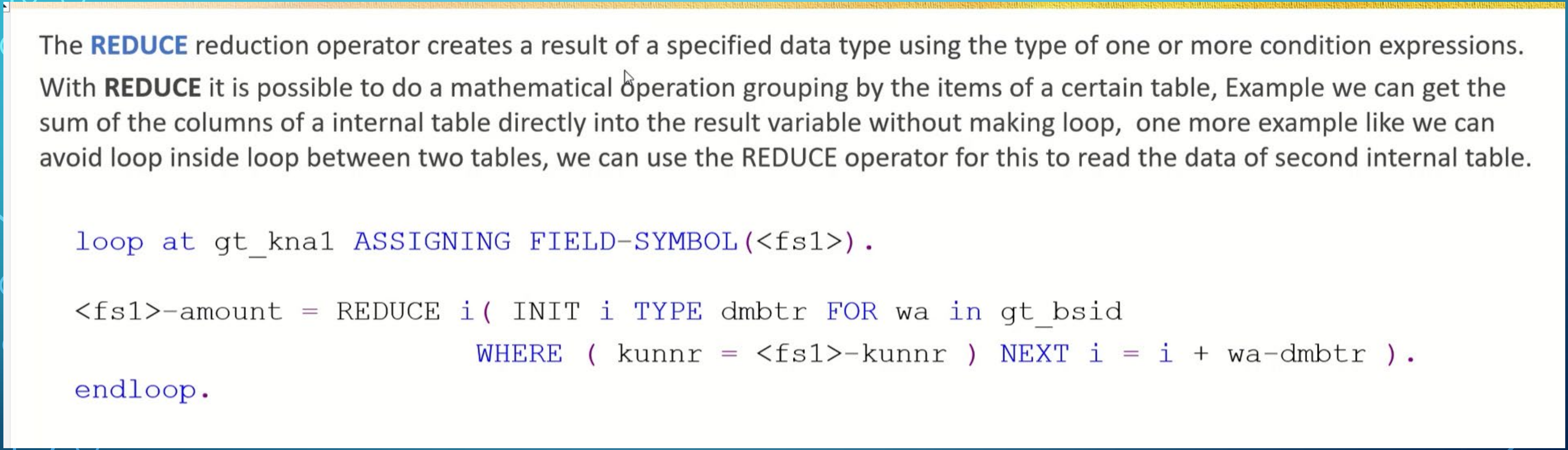
```
data : gv_amount_words type string.

PARAMETERS : P_AMOUNT TYPE DMBTR.

START-OF-SELECTION.

CALL FUNCTION 'HR_IN_CHG_INR_WRDS'
  EXPORTING
    AMT_IN_NUM              = conv  MAXBT( P_AMOUNT )
 IMPORTING
   AMT_IN_WORDS             = gv_amount_words
 EXCEPTIONS
   DATA_TYPE_MISMATCH       = 1
   OTHERS                   = 2
          .
IF SY-SUBRC <> 0.
* Implement suitable error handling here
ENDIF.

write : / GV_AMOUNT_WORDS.


END-OF-SELECTION.
```

CONV OPERATOR)


Program Edit Goto Utilities Environment System Help
ABAP Editor: Change Report ZNEW_ABAP_SYNTAX_CONVERSION
Pattern Pretty Printer
Report ZNEW_ABAP_SYNTAX_CONVERSION Active

```
REPORT ZNEW_ABAP_SYNTAX_CONVERSION.

Data : gv_amount_words(250).

PARAMETERS : P_AMOUNT TYPE DMBTR.

START-OF-SELECTION.


CALL FUNCTION 'HR_IN_CHG_INR_WRDS'
  EXPORTING
    AMT_IN_NUM                     = conv MAXBT( P_AMOUNT )
 IMPORTING
   AMT_IN_WORDS                  = GV_AMOUNT_WORDS
 EXCEPTIONS
   DATA_TYPE_MISMATCH            = 1
   OTHERS                        = 2
          .
IF SY-SUBRC <> 0.
* Implement suitable error handling here
ENDIF.

 write : / GV_AMOUNT_WORDS.
```

REDUCE OPERATOR

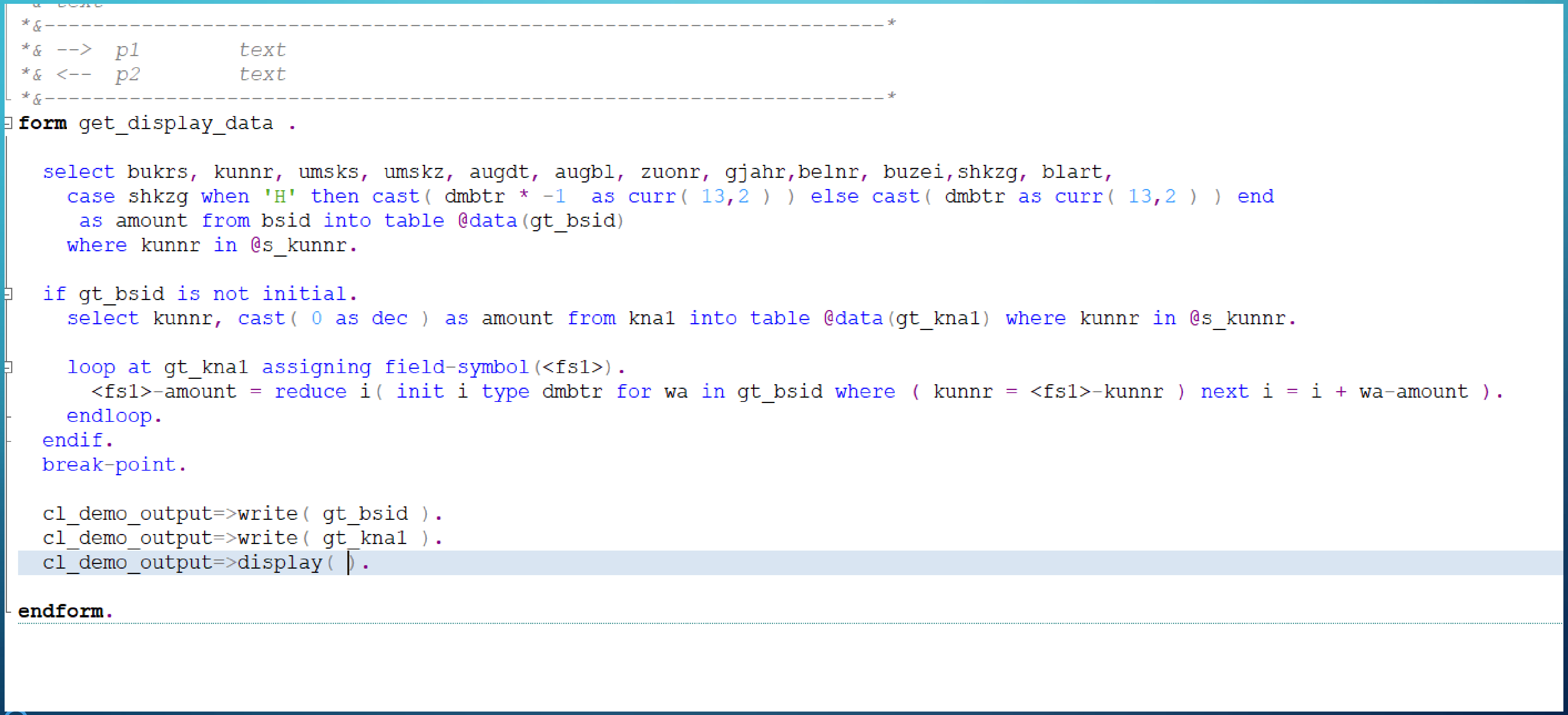


The **REDUCE** reduction operator creates a result of a specified data type using the type of one or more condition expressions.

With **REDUCE** it is possible to do a mathematical operation grouping by the items of a certain table, Example we can get the sum of the columns of a internal table directly into the result variable without making loop, one more example like we can avoid loop inside loop between two tables, we can use the REDUCE operator for this to read the data of second internal table.

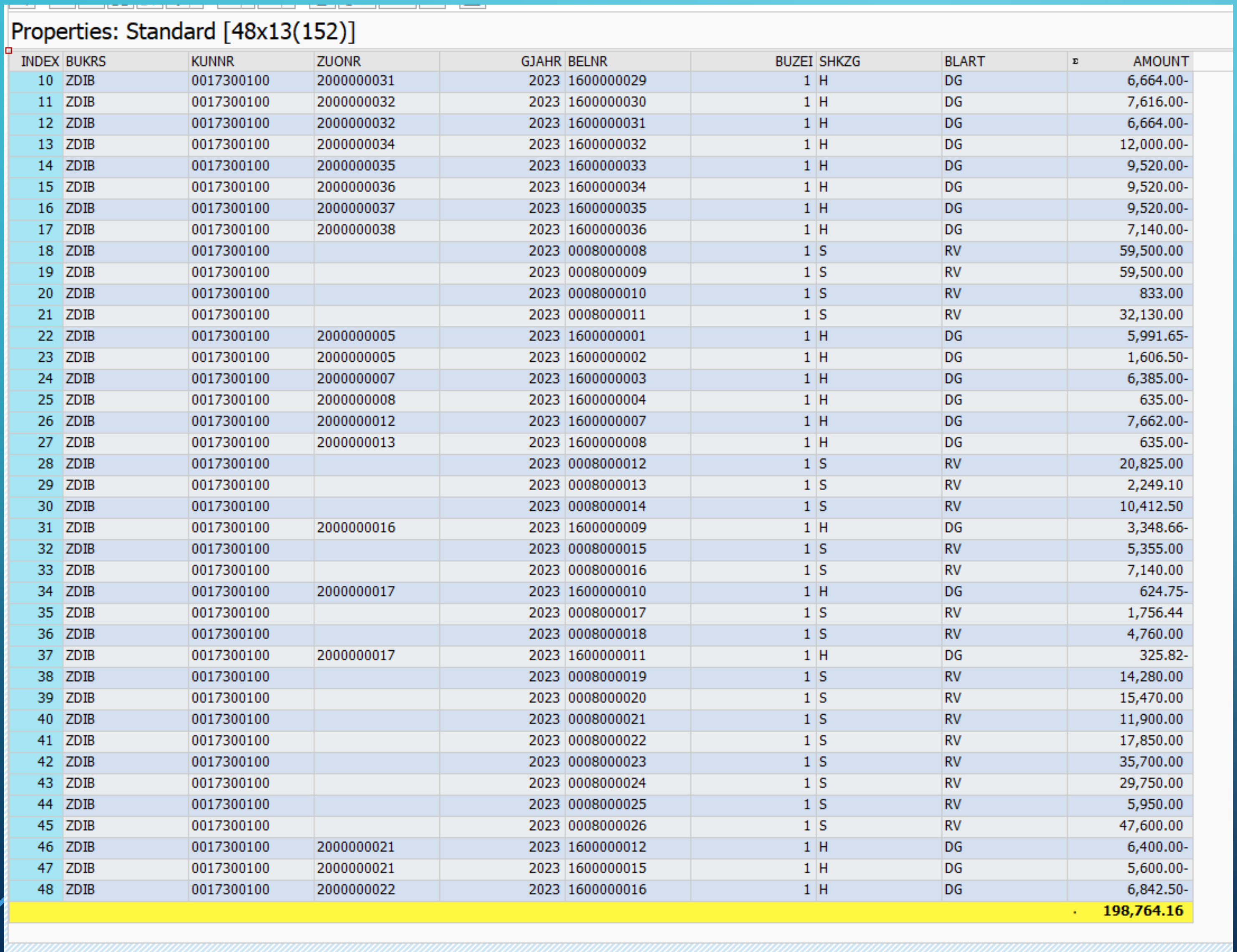
```
loop at gt_kna1 ASSIGNING FIELD-SYMBOL(<fs1>).

<fs1>-amount = REDUCE i( INIT i TYPE dmbtr FOR wa in gt_bsid
                      WHERE ( kunnr = <fs1>-kunnr ) NEXT i = i + wa-dmbtr ).
endloop.
```

REDUCE OPERATOR (USING ALSO CAST OPERATOR)



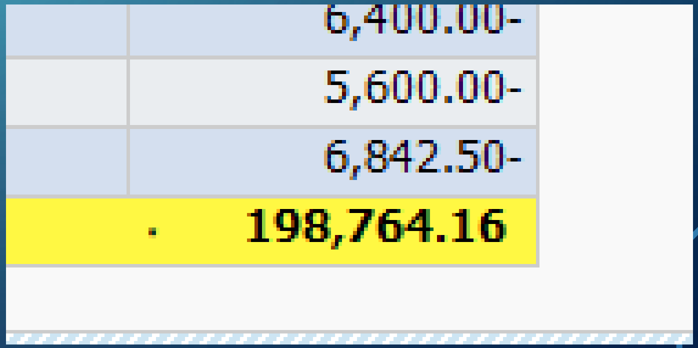
```
*&---------------------------------------------------------------------*
*& -->  p1        text
*& <--  p2        text
*&---------------------------------------------------------------------*
form get_display_data .

  select bukrs, kunnr, umsks, umskz, augdt, augbl, zuonr, gjahr,belnr, buzei,shkzg, blart,
    case shkzg when 'H' then cast( dmbtr * -1  as curr( 13,2 ) ) else cast( dmbtr as curr( 13,2 ) ) end
     as amount from bsid into table @data(gt_bsid)
    where kunnr in @s_kunnr.

  if gt_bsid is not initial.
    select kunnr, cast( 0 as dec ) as amount from kna1 into table @data(gt_kna1) where kunnr in @s_kunnr.

    loop at gt_kna1 assigning field-symbol(<fs1>).
      <fs1>-amount = reduce i( init i type dmbtr for wa in gt_bsid where ( kunnr = <fs1>-kunnr ) next i = i + wa-amount ).
    endloop.
  endif.
  break-point.

  cl_demo_output=>write( gt_bsid ).
  cl_demo_output=>write( gt_kna1 ).
  cl_demo_output=>display( ).

endform.
```

REDUCE OPERATOR (USING ALSO CAST OPERATOR) – FROM DEBUGGER BEFORE REDUCE



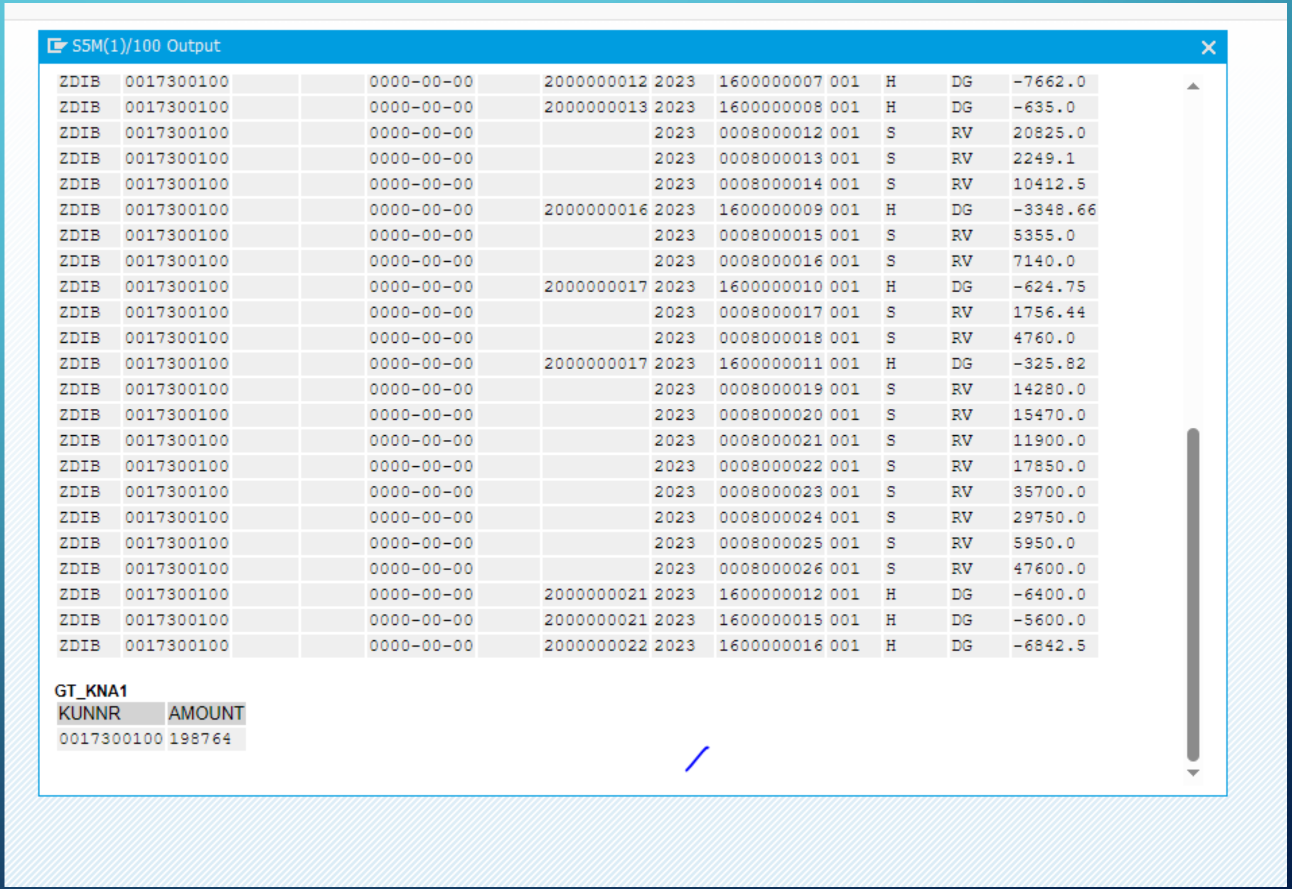
Properties: Standard [48x13(152)]

INDEX	BUKRS	KUNNR	ZUONR	GJAHR	BELNR	BUZEI	SHKZG	BLART	Σ	AMOUNT
10	ZDIB	0017300100	2000000031	2023	1600000029	1	H	DG		6,664.00-
11	ZDIB	0017300100	2000000032	2023	1600000030	1	H	DG		7,616.00-
12	ZDIB	0017300100	2000000032	2023	1600000031	1	H	DG		6,664.00-
13	ZDIB	0017300100	2000000034	2023	1600000032	1	H	DG		12,000.00-
14	ZDIB	0017300100	2000000035	2023	1600000033	1	H	DG		9,520.00-
15	ZDIB	0017300100	2000000036	2023	1600000034	1	H	DG		9,520.00-
16	ZDIB	0017300100	2000000037	2023	1600000035	1	H	DG		9,520.00-
17	ZDIB	0017300100	2000000038	2023	1600000036	1	H	DG		7,140.00-
18	ZDIB	0017300100		2023	0008000008	1	S	RV		59,500.00
19	ZDIB	0017300100		2023	0008000009	1	S	RV		59,500.00
20	ZDIB	0017300100		2023	0008000010	1	S	RV		833.00
21	ZDIB	0017300100		2023	0008000011	1	S	RV		32,130.00
22	ZDIB	0017300100	2000000005	2023	1600000001	1	H	DG		5,991.65-
23	ZDIB	0017300100	2000000005	2023	1600000002	1	H	DG		1,606.50-
24	ZDIB	0017300100	2000000007	2023	1600000003	1	H	DG		6,385.00-
25	ZDIB	0017300100	2000000008	2023	1600000004	1	H	DG		635.00-
26	ZDIB	0017300100	2000000012	2023	1600000007	1	H	DG		7,662.00-
27	ZDIB	0017300100	2000000013	2023	1600000008	1	H	DG		635.00-
28	ZDIB	0017300100		2023	0008000012	1	S	RV		20,825.00
29	ZDIB	0017300100		2023	0008000013	1	S	RV		2,249.10
30	ZDIB	0017300100		2023	0008000014	1	S	RV		10,412.50
31	ZDIB	0017300100	2000000016	2023	1600000009	1	H	DG		3,348.66-
32	ZDIB	0017300100		2023	0008000015	1	S	RV		5,355.00
33	ZDIB	0017300100		2023	0008000016	1	S	RV		7,140.00
34	ZDIB	0017300100	2000000017	2023	1600000010	1	H	DG		624.75-
35	ZDIB	0017300100		2023	0008000017	1	S	RV		1,756.44
36	ZDIB	0017300100		2023	0008000018	1	S	RV		4,760.00
37	ZDIB	0017300100	2000000017	2023	1600000011	1	H	DG		325.82-
38	ZDIB	0017300100		2023	0008000019	1	S	RV		14,280.00
39	ZDIB	0017300100		2023	0008000020	1	S	RV		15,470.00
40	ZDIB	0017300100		2023	0008000021	1	S	RV		11,900.00
41	ZDIB	0017300100		2023	0008000022	1	S	RV		17,850.00
42	ZDIB	0017300100		2023	0008000023	1	S	RV		35,700.00
43	ZDIB	0017300100		2023	0008000024	1	S	RV		29,750.00
44	ZDIB	0017300100		2023	0008000025	1	S	RV		5,950.00
45	ZDIB	0017300100		2023	0008000026	1	S	RV		47,600.00
46	ZDIB	0017300100	2000000021	2023	1600000012	1	H	DG		6,400.00-
47	ZDIB	0017300100	2000000021	2023	1600000015	1	H	DG		5,600.00-
48	ZDIB	0017300100	2000000022	2023	1600000016	1	H	DG		6,842.50-
									·	198,764.16



	6,400.00-
	5,600.00-
	6,842.50-
·	198,764.16

REDUCE OPERATOR (USING ALSO CAST OPERATOR) – RESULT



S5M(1)/100 Output

```
ZDIB 0017300100   0000-00-00 2000000012 2023 1600000007 001 H DG -7662.0
ZDIB 0017300100   0000-00-00 2000000013 2023 1600000008 001 H DG -635.0
ZDIB 0017300100   0000-00-00            2023 0008000012 001 S RV 20825.0
ZDIB 0017300100   0000-00-00            2023 0008000013 001 S RV 2249.1
ZDIB 0017300100   0000-00-00            2023 0008000014 001 S RV 10412.5
ZDIB 0017300100   0000-00-00 2000000016 2023 1600000009 001 H DG -3348.66
ZDIB 0017300100   0000-00-00            2023 0008000015 001 S RV 5355.0
ZDIB 0017300100   0000-00-00            2023 0008000016 001 S RV 7140.0
ZDIB 0017300100   0000-00-00 2000000017 2023 1600000010 001 H DG -624.75
ZDIB 0017300100   0000-00-00            2023 0008000017 001 S RV 1756.44
ZDIB 0017300100   0000-00-00            2023 0008000018 001 S RV 4760.0
ZDIB 0017300100   0000-00-00 2000000017 2023 1600000011 001 H DG -325.82
ZDIB 0017300100   0000-00-00            2023 0008000019 001 S RV 14280.0
ZDIB 0017300100   0000-00-00            2023 0008000020 001 S RV 15470.0
ZDIB 0017300100   0000-00-00            2023 0008000021 001 S RV 11900.0
ZDIB 0017300100   0000-00-00            2023 0008000022 001 S RV 17850.0
ZDIB 0017300100   0000-00-00            2023 0008000023 001 S RV 35700.0
ZDIB 0017300100   0000-00-00            2023 0008000024 001 S RV 29750.0
ZDIB 0017300100   0000-00-00            2023 0008000025 001 S RV 5950.0
ZDIB 0017300100   0000-00-00            2023 0008000026 001 S RV 47600.0
ZDIB 0017300100   0000-00-00 2000000021 2023 1600000012 001 H DG -6400.0
ZDIB 0017300100   0000-00-00 2000000021 2023 1600000015 001 H DG -5600.0
ZDIB 0017300100   0000-00-00 2000000022 2023 1600000016 001 H DG -6842.5
```

GT_KNA1

KUNNR	AMOUNT
0017300100	198764