

Tutorial

SAP ABAP Basic

Rev 8

SAP Interface Community
Bogor, Indonesia

Contact Us : 0888-190-4728 (WA)

Training Silabus

Day - 1

1. ABAP Editor
2. SAP GUI Launcher
3. ABAP Syntax
4. Data Type
5. ABAP Debugger

Day - 2

1. ABAP Dictionary
2. Selection Screen
3. Open SQL in ABAP

Day - 3

1. Subroutine
2. Function Module Basic 1
3. Output List Display

Day - 4

1. ALV - ABAP List Viewer
2. ARAD - ABAP Rapid Application Development
3. SAP Interface/Integration with PHP

Day - 5

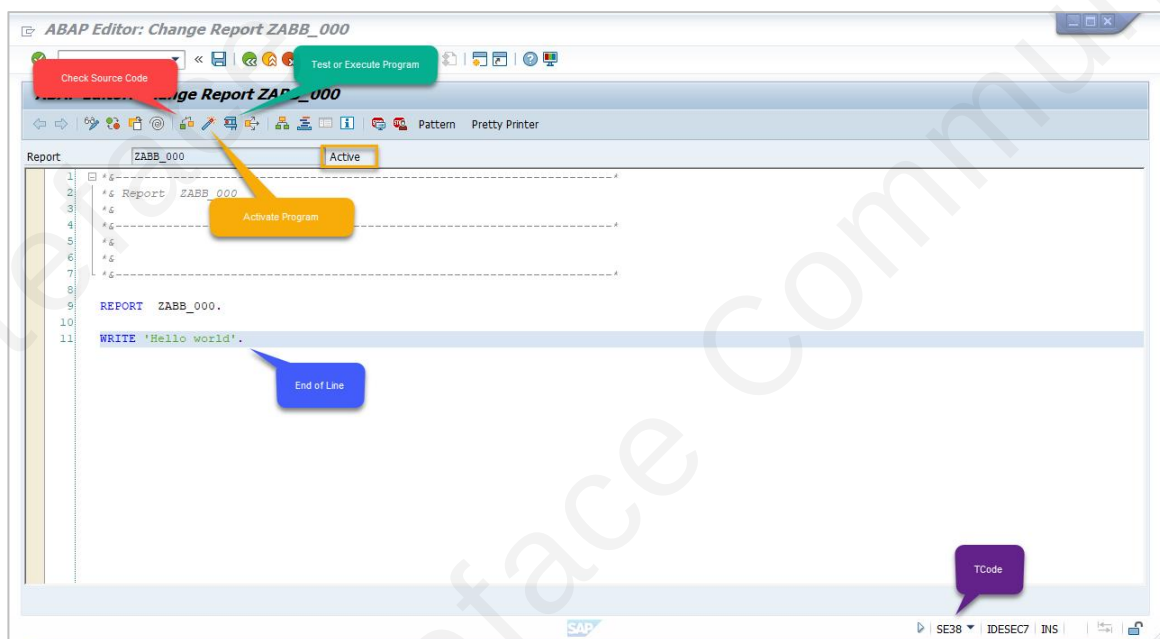
1. Function Module Basic 2
2. ABAP Dialog Programming

Day - 1

TCode	SE38, SE39, SE80
Target	Understanding basic of ABAP Programming

1. ABAP Editor

The ABAP Editor is a source code editing tool designed to handle the specifics of ABAP coding. It is one of the main components of the ABAP Workbench. You use the ABAP Editor to write and edit ABAP programs, class methods, function modules, screen flow logic, type groups, and logical databases.



ABAP Keyword

Line	ABAP Keyword	Description
9	REPORT	First Declaration of ABAP Program
11	WRITE	Display Output
11	“.” dot	End of Line Code

2. SAP GUI Launcher

The SGL (SAP GUI Launcher) is part of ARAD Feature. SGL used to launch "SAP GUI for Windows" and "SAP Web GUI" without entered SAP User ID and Password. SAP parameters are input and saved in the ARAD application.

A. Setup SAP Connection

The screenshot shows the 'SAP-Connection : REIN_IDES_SAP_INTERFACE' window in the ABAP Rapid Application Development vBeta 2 application. The window has tabs for 'Home', 'Applications-SAP GUI Launcher', and 'SAP-Connection : REIN_IDES_SAP_INTERFACE'. The 'SAP-Connection : REIN_IDES_SAP_INTERFACE' tab is active, showing a 'Name' field with 'REIN_IDES_SAP_INTERFACE' and a 'private' folder icon. Below this are buttons for 'Test Connection', 'Open SAP GUI', and 'Open WEB GUI'. The 'Connection Attributes' section is expanded, showing fields for 'Application Server', 'System Number' (00), 'System ID' (EC7), 'SAP Router', 'Gateway URL' (http://), 'Browser' (Internet Explorer selected), 'Logon Info' (Client 800, Username, Password), and 'tCode Parameter'. A status bar at the bottom shows a log of events: '20260208 00:30:45 : [ABAP Rapid Application Development vBeta 2] : Finished', '20260208 00:30:48 : [ABAP Rapid Application Development vBeta 2] : Get Verification Code', '20260208 00:30:52 : [ABAP Rapid Application Development vBeta 2] : Login succeed', '20260208 00:30:52 : [ABAP Rapid Application Development vBeta 2] : Get Subscriptions Data', and '20260208 00:32:11 : [FILE - Selected] : private/REIN_IDES_SAP_INTERFACE'.

B. SAP GUI Launcher

The screenshot shows the 'SAP GUI Launcher' window in the ABAP Rapid Application Development vBeta 2 application. The window has tabs for 'Home', 'Applications-SAP GUI Launcher', and 'SAP GUI'. The 'SAP GUI' tab is active, showing a table with columns 'Pos', 'Connection', and 'Description'. The table contains one entry: '1', 'REIN_IDES_SAP_INTERFACE', and an empty description. Below the table is a status bar showing a log of events: '20260208 00:30:44 : [ABAP Rapid Application Development vBeta 2] : Finished', '20260208 00:30:45 : [ABAP Rapid Application Development vBeta 2] : Get Verification Code', '20260208 00:30:48 : [ABAP Rapid Application Development vBeta 2] : Login succeed', '20260208 00:30:52 : [ABAP Rapid Application Development vBeta 2] : Get Subscriptions Data', and '20260208 00:30:52 : [ABAP Rapid Application Development vBeta 2] : Get Subscriptions Data'.

3. ABAP Syntax

A. Statement

Statement is lines of codes which instruct an order. In ABAP, every statement always close by dot(.) at the end.

These are few examples of statements:

Example Report 000. First ABAP Program

```
WRITE 'Hello World'.
```

Statements are always begined by ABAP Keywords (Example: TYPES , DATA , write dst). Keyword determine statement Catagory.

These are reference of statement at ABAP Program:

1. Data Types

2. Declaration of Data Type

This statement use to define a data type or object as variable of program.

Example :

```
TYPES <variable_name> TYPE <type>.
```

```
DATA <variable_name> TYPE <type>.
```

3. Compound Statement

Compound Statement is group of statements of a block coding. It's also called as "Block Statement".

Example :

```
FORM <form_name>.
```

```
Statement 1.
```

```
Statement 2.
```

```
...
```

```
Statement n.
```

```
ENDFORM.
```

```
FUNCTION <function_name>.
```

```
Statement 1.
```

```
Statement 2.
```

```
...
```

```
Statement n.
```

```
ENDFUNCTION.
```

```
MODULE <module_name>.
```

```
Statement 1.
```

```
Statement 2.
```

```
...
```

Statement n.

ENDMODULE.

4. Control Statement

Control Statement is used to control statement flow according to conditions that suitable to algorithm of program we want.

Example :

```
IF <condition1>.  
    <statement block>  
ELSEIF <condition2>  
    <statement block>.  
ELSEIF <condition3>.  
    <statement block>  
.....  
ELSE.  
    <statement block>  
ENDIF.
```

5. Operator Statement

Operator Statement is keyword for doing some process of type/object that declared in a program.

Example :

```
WRITE <statement>.  
ADD <variable 1> TO <variable 2>.
```

6. Database Statement

Database Statement is statement that used for accessing database. In ABAP, there are 2 statement for accessing database, these are Open SQL and Native SQL.

B. Literal

Literal is the way to declare directly a value in language program.

There are 2 type of literal at ABAP, these are:

1. Numeric Literal

Example :

```
A = 10.  
B = 100.
```

2. Text Literal

Text Literal initiate dan close by sitation (').

Example :

```
WRITE 'Hello World'.
```

C. Chained Statement

ABAP has classical language program characteristic, that is, in every statement in main program or in block statement will be executed in sequence from the top to bottom.

Example :

NO	Statement
1	DATA : A TYPE I, B TYPE I, C TYPE P DECIMALS 2.
2	START-OF-SELECTION.
3	A = 10.
4	B = 100.
5	C = A / B.
6	WRITE : / 'Result : ', A, '/', B, '=', C.

D. Comment

Inside the source code, comment is explanation or infomation for statement or block statement. It's important to give explanation about steps or algoritmn of ABAP program. If someone else want edit that program, it's not difficult to study the logic algorithm in code of program.

There are 2 kind of comment in ABAP, these are:

1. Use key star (*), and locate it in the first character at line program.

Example :

```

FORM fm_process_data.
*Get other requirement information
LOOP AT gi_header.
*   Get Material Description
  READ TABLE gi_makt WITH KEY matnr = gi_header-matnr.
  IF sy-subrc = 0.
    gi_header-maktx = gi_makt-maktx.
  ENDIF.

*   Get Measurement Text
  READ TABLE gi_t006a WITH KEY msehi = gi_header-meins.
  IF sy-subrc = 0.
    gi_header-mseht = gi_t006a-mseht.
  ENDIF.

  MODIFY gi_header.
ENDLOOP.

ENDFORM.

```

2. Use double citation ("), it's free to locate it.

Example :

```

FORM fm_process_data.
"Get other requirement information
LOOP AT gi_header.
  "Get Material Description
  READ TABLE gi_makt WITH KEY matnr = gi_header-matnr.
  IF sy-subrc = 0.
    gi_header-maktx = gi_makt-maktx.
  ENDIF.

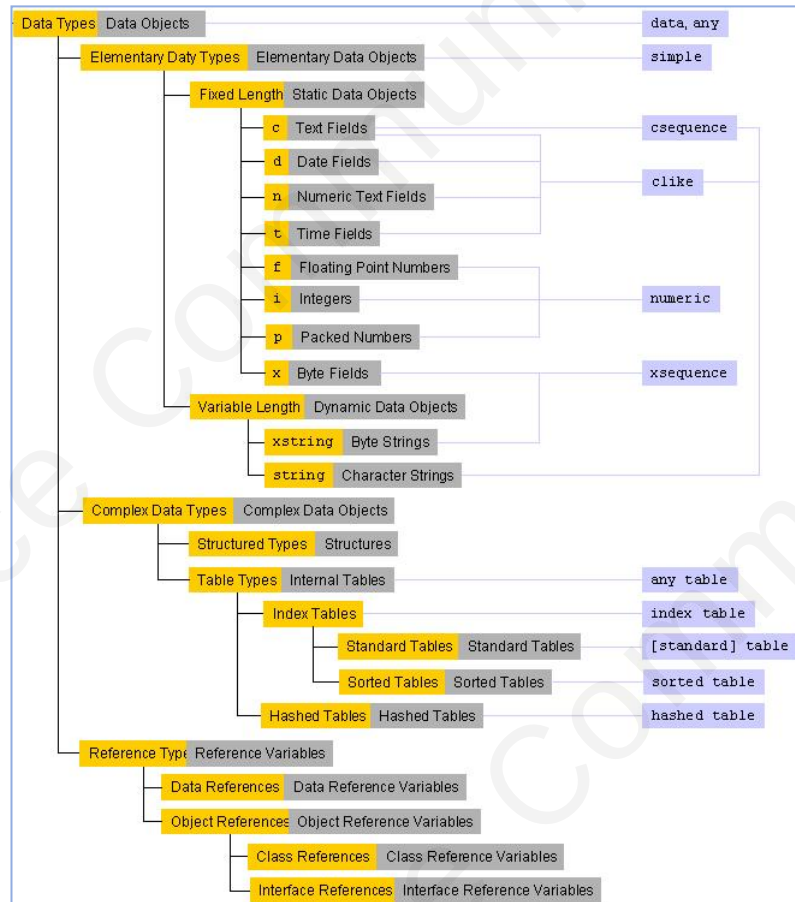
  "Get Measurement Text
  READ TABLE gi_t006a WITH KEY msehi = gi_header-meins.
  IF sy-subrc = 0.
    gi_header-mseht = gi_t006a-mseht.
  ENDIF.

  MODIFY gi_header.
ENDLOOP.

ENDFORM.

```

4. Data Type



Data Types are further subdivided into elementary, complex and reference types. There are predefined (built-in) data types, or they can be defined locally in the program and system-wide in the ABAP Dictionary.

A. Elementary Data Type

they are not composed of other types. They are further classified into elementary types of fixed length and of variable length.

■ Predefined Build-in :

Data Type	Length	Initial Field Length (Bytes)	Valid Field Length (Bytes)
C	Not Fixed	1	1 to 65535
N	not fixed	1	1 to 65535
D	fixed	8 characters	8 character
I	fixed	4	4
F	fixed	8	8
P	not	8	8

	fixed		
X	not fixed	1	1-65,535
T	fixed	6 characters	6 characters
Decfloat16	fixed	8 (16 digits)	8 (16 digits)
Decfloat32	fixed	16 (34 digits)	16 (34 digits)
STRING	dynamic	Allocated at Run time	Allocated at Run time
XSTRING	dynamic	Allocated at Run time	Allocated at Run time

Example Report 001. Variable Declaration

```
DATA : A TYPE I,
      B TYPE I,
      C TYPE P DECIMALS 2.

START-OF-SELECTION.
  A = 10.
  B = 100.
  C = A / B.
  WRITE : / 'Result : ', A, '/', B, '=', C.
```

"A, B, C" is new variable

Example Report 002. Refer to Existing Variable/Types

```
DATA : A TYPE I,
      B TYPE I,
      C TYPE P DECIMALS 2.

DATA D LIKE C. " C variable as Data Types

START-OF-SELECTION.
  A = 10.
  B = 100.
  C = A / B.
  D = C.
  WRITE : / 'Result : ', A, '/', B, '=', D.
```

"C" variable used "C" variable as Data Types.

■ Predifined User Type (Local Definition) :

Example Report 003. New Types Declaration

```
TYPES TY_NEW TYPE P DECIMALS 2. " Predefined Types

DATA : A TYPE I,
      B TYPE I,
      C TYPE TY_NEW.

START-OF-SELECTION.
  A = 10.
  B = 100.
  C = A / B.

  WRITE : / 'Result : ', A, '/', B, '=', C.
```

"TY_NEW" as new types and used by "C" variable

B. Complex Types

Complex data types are composed of other types. A complex type must either be defined in the ABAP program or in the ABAP Dictionary. There are two types of complex data type:

■ Structure Types

Alias : "one dimensional array", "structure", "workarea", "wa"

Example Report 004. Structure Type used Predefined User Types

```
TYPES : BEGIN OF TY_DATA,  
        A TYPE I,  
        B TYPE I,  
        C TYPE P DECIMALS 2,  
      END OF TY_DATA.  
  
DATA : GW_DATA TYPE TY_DATA.  
  
DATA D LIKE GW_DATA-C.  
  
START-OF-SELECTION.  
  GW_DATA-A = 10.  
  GW_DATA-B = 100.  
  GW_DATA-C = GW_DATA-A / GW_DATA-B.  
  D = GW_DATA-C.  
  WRITE : / 'Result : ', GW_DATA-A, '/', GW_DATA-B, '=', D.
```

"A, B, C" inside reference types of "TY_DATA".

"GW_DATA" is new variable used "TY_DATA" as Data Types

Example Report 005. Structure Type without Predefined User Types

```
DATA : BEGIN OF GW_DATA,  
        A TYPE I,  
        B TYPE I,  
        C TYPE P DECIMALS 2,  
      END OF GW_DATA.  
  
DATA D LIKE GW_DATA-C.  
  
START-OF-SELECTION.  
  GW_DATA-A = 10.  
  GW_DATA-B = 100.  
  GW_DATA-C = GW_DATA-A / GW_DATA-B.  
  D = GW_DATA-C.  
  WRITE : / 'Result : ', GW_DATA-A, '/', GW_DATA-B, '=', D.
```

"A, B, C" inside variable of "GW_DATA".

■ Table Types

Alias : "multi dimensional array", "table type", "internal table", "itab"

Example Report 006. Complex Type used Predefined User Types

```
TYPES : BEGIN OF TY_DATA,
        A TYPE I,
        B TYPE I,
        C TYPE P DECIMALS 2,
      END OF TY_DATA.

DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.

START-OF-SELECTION.
  DATA : LV_COUNTER TYPE I,
          LV_PULUHAN TYPE I.
  "Create Dummy Data
  CLEAR LV_COUNTER.
  DO 10 TIMES.
    LV_COUNTER = LV_COUNTER + 1.

    GI_DATA-A = LV_COUNTER.
    LV_PULUHAN = GI_DATA-A MOD 10.
    IF LV_PULUHAN <> 0.
      GI_DATA-B = LV_COUNTER + 4.
      GI_DATA-C = GI_DATA-A + GI_DATA-B.
      APPEND GI_DATA. "Insert data to itab
    ENDIF.
  ENDDO.

END-OF-SELECTION.
  "Display Data
  LOOP AT GI_DATA.
    WRITE : / GI_DATA-A, GI_DATA-B, GI_DATA-C.
  ENDLOOP.
```

"A, B, C" inside reference types of "TY_DATA".

"GI_DATA" is new variable used "TY_DATA" as Data Types

Example Report 007. Complex Type refer all field form SAP table

```
TYPES TY_DATA TYPE STANDARD TABLE OF MAKT.

DATA : GI_DATA TYPE TY_DATA WITH HEADER LINE.

START-OF-SELECTION.
  "Collect Data
  SELECT * INTO TABLE GI_DATA
    UP TO 50 ROWS
  FROM MAKT
  WHERE SPRAS = SY-LANGU.

END-OF-SELECTION.
  "Display Data
  LOOP AT GI_DATA.
    WRITE : / GI_DATA-MATNR, GI_DATA-MAKTX.
  ENDLOOP.
```

"TY_DATA" have all variables form fields in "MAKT" table

"SELECT * INTO TABLE . . . FROM . . ." OPEN SQL statement for reading data from database tables

"LOOP AT . . ." read lines form internal table

Example Report 008. Complex Type refer specific field form SAP table

```
TYPES : BEGIN OF TY_DATA,
        MATNR TYPE MAKT-MATNR,
        MAKTX TYPE MAKT-MAKTX,
        END OF TY_DATA.

DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.

START-OF-SELECTION.
    "Collect Data
    SELECT MATNR MAKTX INTO TABLE GI_DATA
        UP TO 50 ROWS
    FROM MAKT
    WHERE SPRAS = SY-LANGU.

END-OF-SELECTION.
    "Display Data
    LOOP AT GI_DATA.
        WRITE : / GI_DATA-MATNR, GI_DATA-MAKTX.
    ENDLOOP.
```

"TY_DATA" have specific variables form fields in "MAKT" table

C. Reference Types

Reference types describe data objects that contain references to other objects (data objects and instances of classes), which are known as reference variables. There are no predefined reference types. A reference types must either be defined in the ABAP program, the ABAP Dictionary or in The ABAP Object.

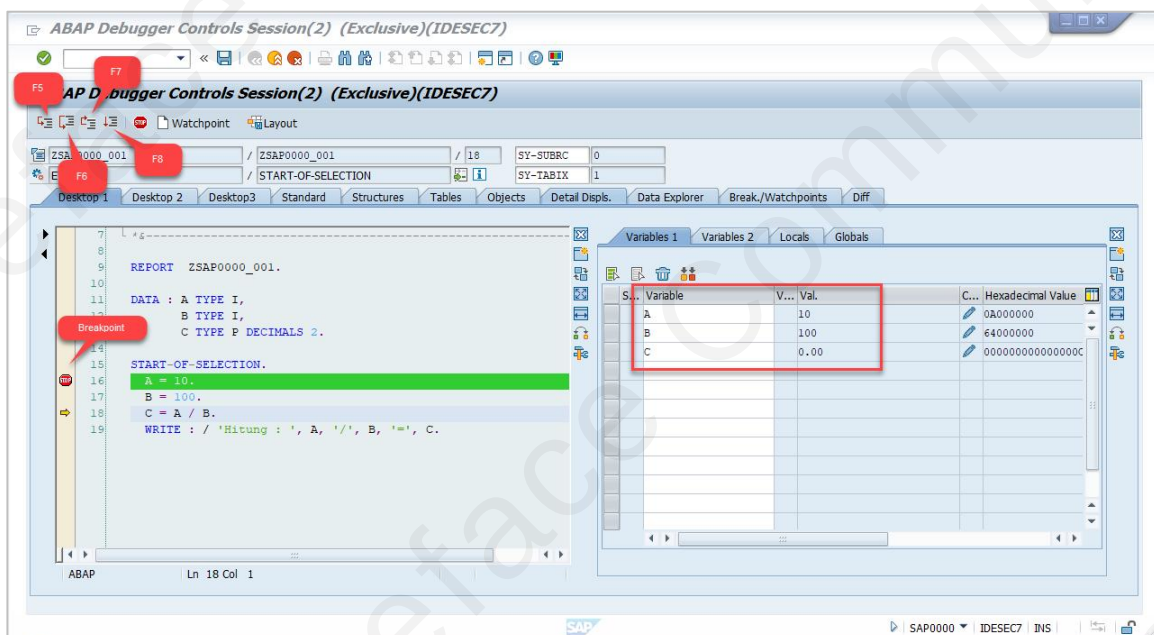
**Off topic and studied in ABAP Object*

5. ABAP Debugger

ABAP Debugger is an integrated testing tool within the SAP ABAP Workbench used by developers to analyze and resolve errors in ABAP code at runtime. It allows users to stop program execution, step through code line-by-line, and inspect the values of data objects and variables to identify the root cause of issues.

Main menus in ABAP Debugger which most used are:

1. Single Step (F5), it's used to enter inside subroutine or module.
2. Execute (F8), it's used to execute line coding one by one.
3. Return (F7), it's used to exit from subroutine.
4. Run (to Cursor) (F8), it's used to execute all line code program and only stop if there are icon"

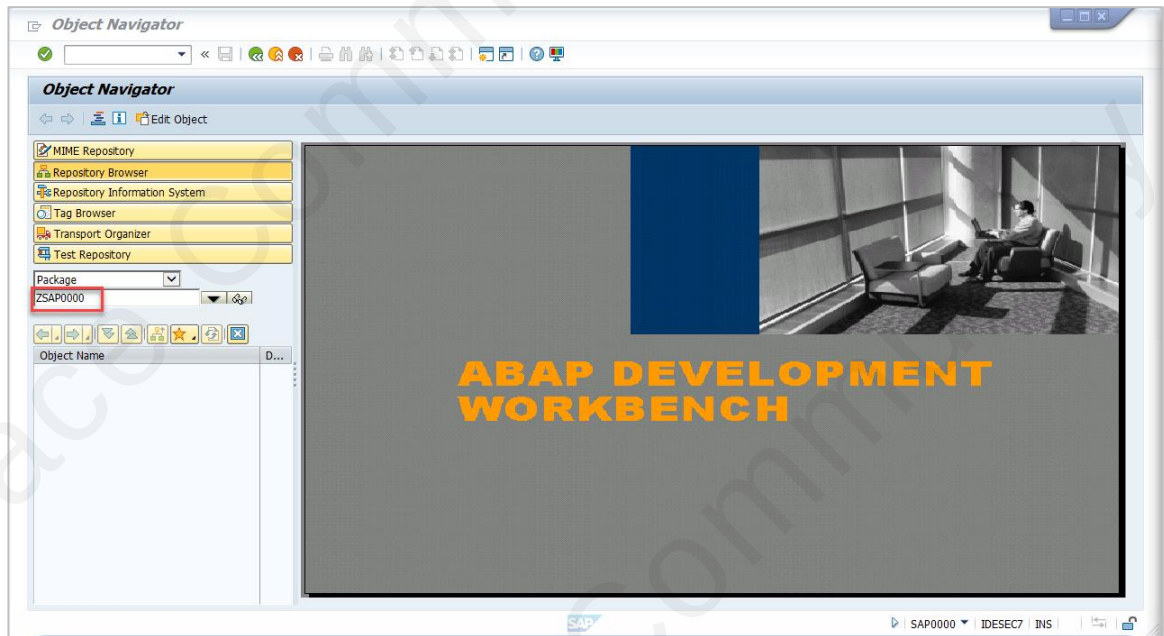


Practice Day - 1

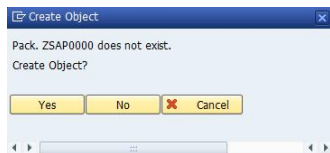
A. Create a Package

Step to create a package:

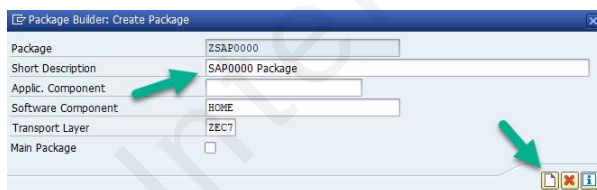
1. Open TCode "SE80"
2. Select "Package"
3. Enter Package Name "ZSAP0000"



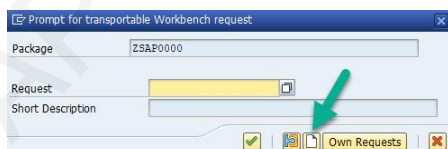
4. Press Enter and Click "Yes"



5. Enter "" and Click "Save"



6. Press "F8" or Click "Create Request"



7. Enter Description "SAP0000 Request Number" and Click "Save"

8. Click "Continue"

9. Done

B. Create a Report 000

Report Name : ZSAP0000_000

Source : Example Report 000

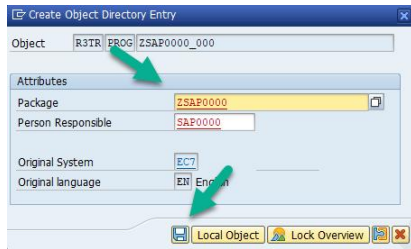
Step to create a report :

1. Open TCode SE38
2. Enter Program Name "ZSAP0000_000"

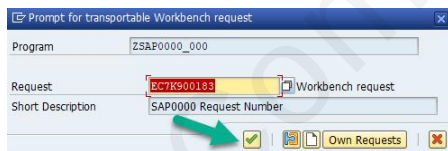
3. Click "Create" button

4. Enter Title "First ABAP Program", Select Attribute Type "Executable program" and Click "Save" button

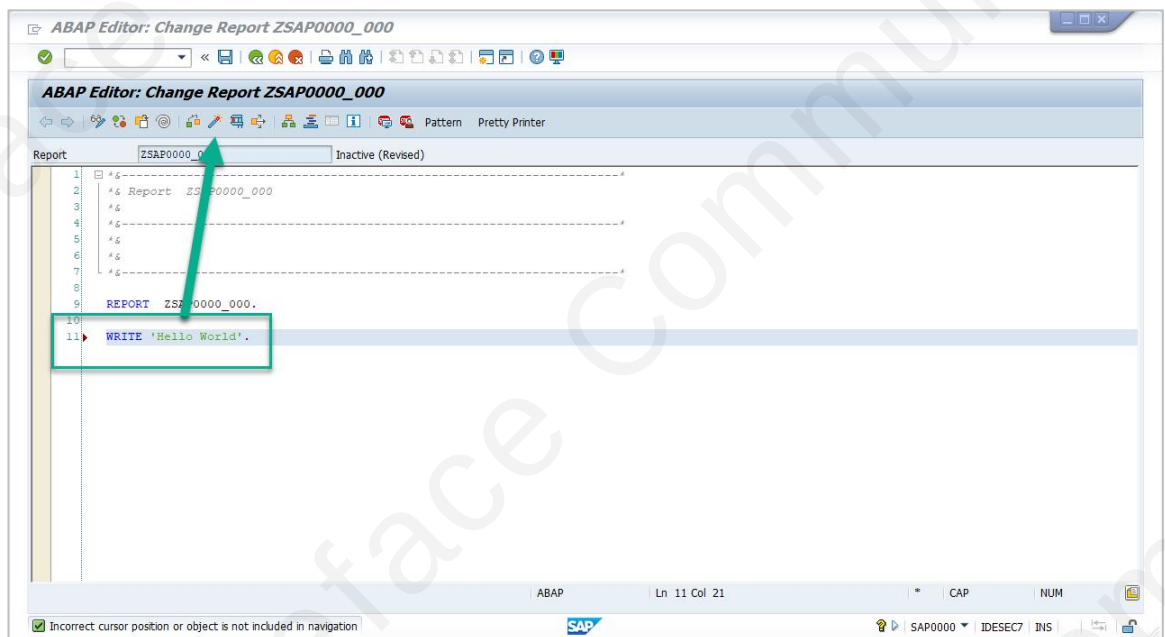
5. Enter Package "ZSAP0000" and Click "Save" button



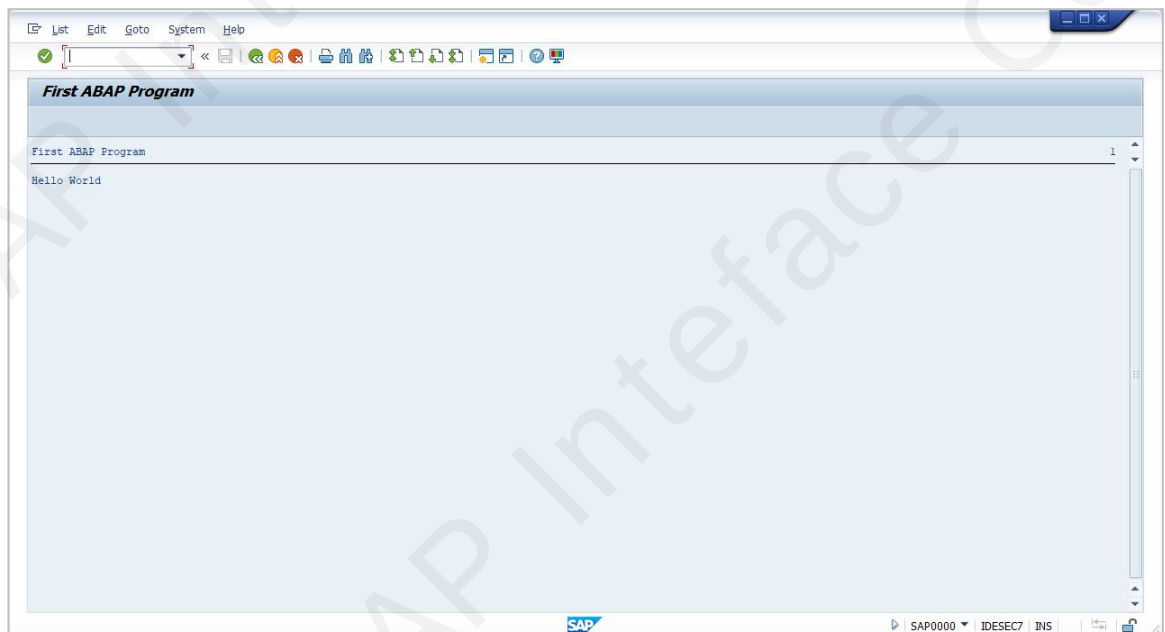
6. Click "Continue"



7. Write Code like this bellow and then click "Activate"



8. Click "Direct Processing" and see the result



C. Create a Report 001

Report Name : ZSAP0000_001
Source : Example Report 001

D. Create a Report 002

Report Name : ZSAP0000_002
Source : Example Report 002

E. Create a Report 003

Report Name : ZSAP0000_003
Source : Example Report 003

F. Create a Report 004

Report Name : ZSAP0000_004
Source : Example Report 004

G. Create a Report 005

Report Name : ZSAP0000_005
Source : Example Report 005

H. Create a Report 006

Report Name : ZSAP0000_006
Source : Example Report 006

I. Create a Report 007

Report Name : ZSAP0000_007
Source : Example Report 007

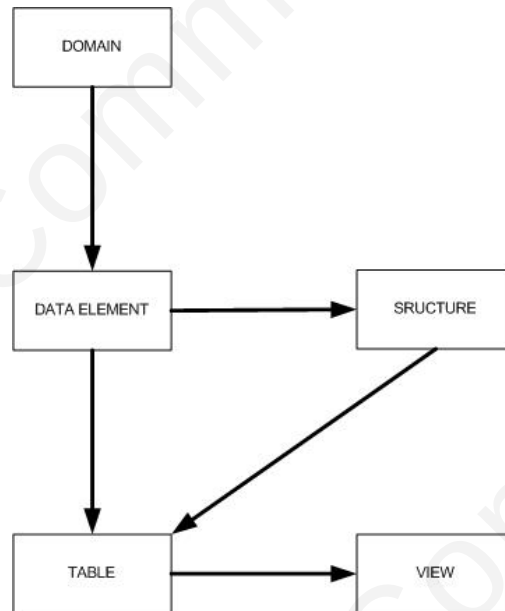
J. Create a Report 008

Report Name : ZSAP0000_008
Source : Example Report 008

Day - 2

TCode	SE11, SE16, SE38
Target	Understanding table design

1. ABAP Dictionary

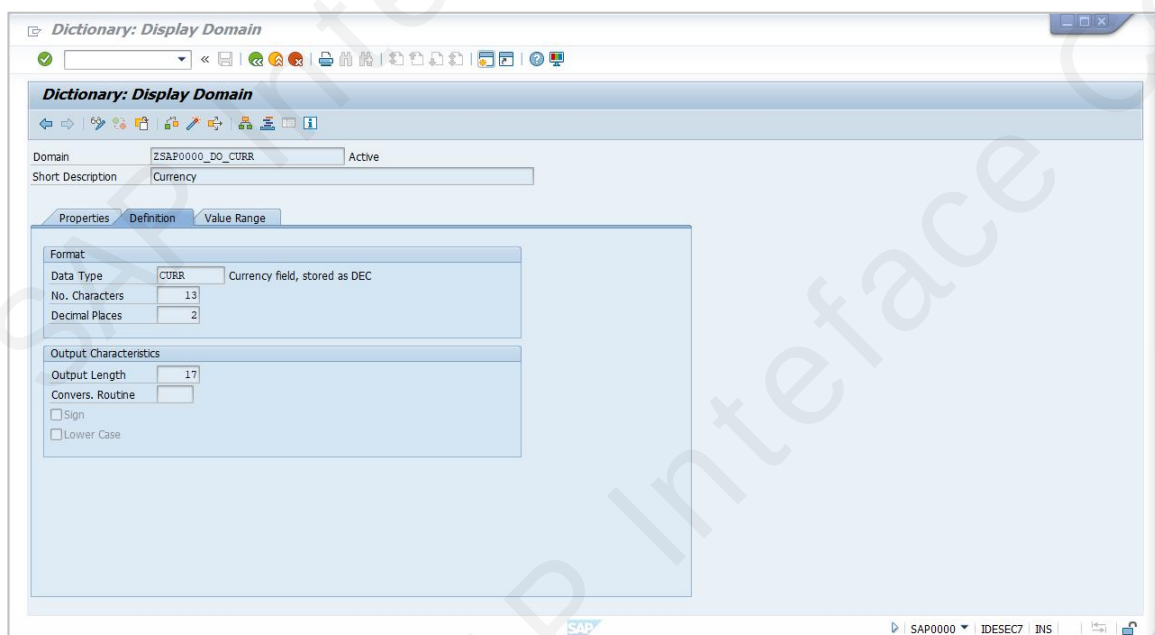


The most used object types in the ABAP Dictionary are domain, data element, structure, table and view.

A. Domain

Domain is a part of Elementary Types with global type, which use at all Data Element.

Example Domain 000.



B. Data Element

Data Element is a part of Elementary Types with global type, which use at all Field (in Table) and to be reference data type in ABAP program.

Example Data Element 000.

The screenshot shows the SAP Dictionary: Display Data Element window for the data element ZSAP0000_DE_CURR. The window is titled "Dictionary: Display Data Element" and has a toolbar with various icons. The main area displays the following information:

- Data element: ZSAP0000_DE_CURR (Active)
- Short Description: Currency
- Attributes tab: Elementary Type (selected), Domain (ZSAP0000_DO_CURR), Data Type (CURR), Length (13), Decimal Places (2), Currency field, stored as DEC.
- Predefined Type: Data Type (), Length (0), Decimal Places (0).
- Reference Type: Name of Ref. Type (), Reference to Predefined Type (Data Type (), Length (0), Decimal Places (0)).

C. Structure

Structure is a part of Complex Types with global type structures, which able to be reference data type in all ABAP Program.

Example Structure 000.

The screenshot shows the SAP Dictionary: Display Structure window for the structure ZSAP0000_ST_CAR. The window is titled "Dictionary: Display Structure" and has a toolbar with various icons. The main area displays the following information:

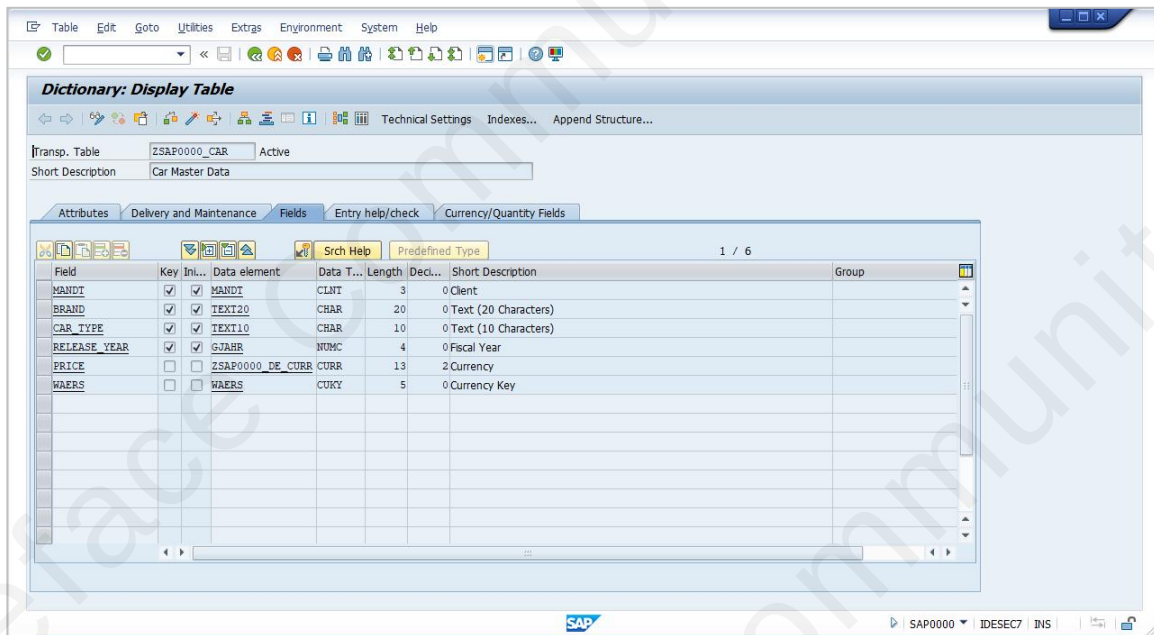
- Structure: ZSAP0000_ST_CAR (Active)
- Short Description: Structure of Car
- Attributes tab: Components (selected), Entry help/check, Currency/quantity fields.
- Table of Components:

Component	RT...	Component type	Data Type	Length	Deci...	Short Description	Group
BRAND	☐	TEXT120	CHAR	20	0	Text (20 Characters)	
CAR_TYPE	☐	TEXT10	CHAR	10	0	Text (10 Characters)	
RELEASE_YEAR	☐	GJAHR	NUMC	4	0	Fiscal Year	
PRICE	☐	ZSAP0000_DE_CURR	CURR	13	2	Currency	
WAERS	☐	WAERS	CURY	5	0	Currency Key	

D. Table

Table is a part of Complex Types with global type tables but, Table is able to filled in and saved at server, which to be reference data type in ABAP program.

Example Table 000.



Dictionary: Display Table

Transp. Table: ZSAP0000_CAR Active

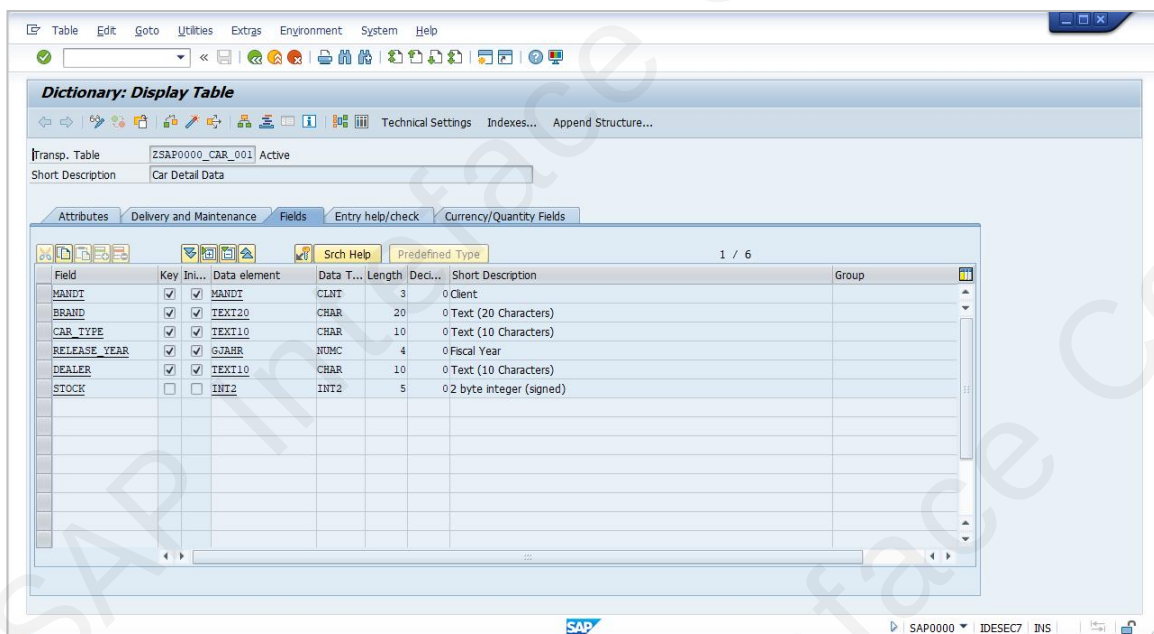
Short Description: Car Master Data

Attributes | Delivery and Maintenance | Fields | Entry help/check | Currency/Quantity Fields

1 / 6

Field	Key	In...	Data element	Data T...	Length	Deci...	Short Description	Group
MANDT	☒	☒	MANDT	CLNT	3		0 Client	
BRAND	☒	☒	TEXT20	CHAR	20		0 Text (20 Characters)	
CAR_TYPE	☒	☒	TEXT10	CHAR	10		0 Text (10 Characters)	
RELEASE_YEAR	☒	☒	GJAHR	NUMC	4		0 Fiscal Year	
PRICE	☐	☐	ZSAP0000_DE_CURR	CURR	13	2	2 Currency	
WAERS	☐	☐	WAERS	CUKY	5		0 Currency Key	

Example Table 001.



Dictionary: Display Table

Transp. Table: ZSAP0000_CAR_001 Active

Short Description: Car Detail Data

Attributes | Delivery and Maintenance | Fields | Entry help/check | Currency/Quantity Fields

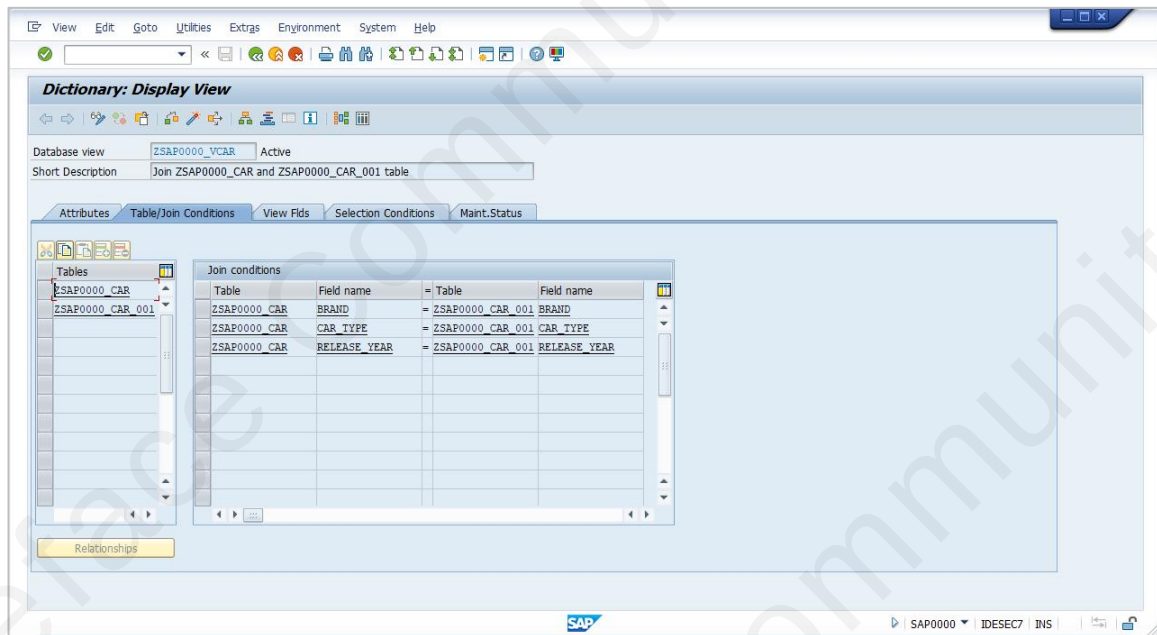
1 / 6

Field	Key	In...	Data element	Data T...	Length	Deci...	Short Description	Group
MANDT	☒	☒	MANDT	CLNT	3		0 Client	
BRAND	☒	☒	TEXT20	CHAR	20		0 Text (20 Characters)	
CAR_TYPE	☒	☒	TEXT10	CHAR	10		0 Text (10 Characters)	
RELEASE_YEAR	☒	☒	GJAHR	NUMC	4		0 Fiscal Year	
DEALER	☒	☒	TEXT10	CHAR	10		0 Text (10 Characters)	
STOCK	☐	☐	INT2	INT2	5		0 2 byte integer (signed)	

E. View

View is a part of Complex Types with global type tables with content of collaborate of tables. Beside that, it can be reference data type in all ABAP Program.

Example View 000.



2. Selection Screen

A. PARAMETERS

Parameters are components of a selection screen that are assigned a global elementary data object in the ABAP program and an input field on the selection screen.

Example Report 009. Parameters Declaration

```
TYPES : BEGIN OF TY_DATA,  
        MATNR TYPE MAKT-MATNR,  
        MAKTX TYPE MAKT-MAKTX,  
        END OF TY_DATA.  
  
DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.  
  
PARAMETERS P_LANGU LIKE SY-LANGU DEFAULT 'EN'.  
  
START-OF-SELECTION.  
    "Collect Data  
    SELECT MATNR MAKTX INTO TABLE GI_DATA  
        UP TO 50 ROWS  
    FROM MAKT  
    WHERE SPRAS = P_LANGU.  
  
END-OF-SELECTION.  
    "Display Data  
    LOOP AT GI_DATA.  
        WRITE : / GI_DATA-MATNR, GI_DATA-MAKTX.  
    ENDLOOP.
```

Example Report 010. Parameters Declaration with "LIKE" Condition

```
TYPES : BEGIN OF TY_DATA,  
        MATNR TYPE MAKT-MATNR,  
        MAKTX TYPE MAKT-MAKTX,  
        END OF TY_DATA.  
  
DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.  
  
PARAMETERS P_MAKTS LIKE MAKT-MAKTX DEFAULT '100'.  
  
START-OF-SELECTION.  
    CONCATENATE '%' P_MAKTS '%' INTO P_MAKTS.  
    "Collect Data  
    SELECT MATNR MAKTX INTO TABLE GI_DATA  
        UP TO 50 ROWS  
    FROM MAKT  
    WHERE SPRAS = SY-LANGU AND  
        MAKTX LIKE P_MAKTS.  
  
END-OF-SELECTION.  
    "Display Data  
    LOOP AT GI_DATA.  
        WRITE : / GI_DATA-MATNR, GI_DATA-MAKTX.  
    ENDLOOP.
```

B. SELECT-OPTIONS

This declares a selection criterion selcrit for a data object (dobj) or a type specified in name. Selection criteria are components of a selection screen, to which are assigned a selection table in the ABAP program, as well as two input fields and a pushbutton for multiple selection.

Example Report 011. Select-Options Declaration

```
TYPES : BEGIN OF TY_DATA,  
        MATNR TYPE MAKT-MATNR,  
        MAKTX TYPE MAKT-MAKTX,  
        END OF TY_DATA.  
  
DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.  
  
SELECT-OPTIONS S_LANGU FOR SY-LANGU DEFAULT 'EN'.  
  
START-OF-SELECTION.  
    "Collect Data  
    SELECT MATNR MAKTX INTO TABLE GI_DATA  
        UP TO 50 ROWS  
    FROM MAKT  
    WHERE SPRAS IN S_LANGU.  
  
END-OF-SELECTION.  
    "Display Data  
    LOOP AT GI_DATA.  
        WRITE : / GI_DATA-MATNR, GI_DATA-MAKTX.  
    ENDLOOP.
```

Example Report 012. Select-Options Declaration with sap table as reference

```
TABLES : MAKT.  
  
TYPES : BEGIN OF TY_DATA,  
        MATNR TYPE MAKT-MATNR,  
        MAKTX TYPE MAKT-MAKTX,  
        END OF TY_DATA.  
  
DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.  
  
SELECT-OPTIONS S_MAKTX FOR MAKT-MAKTX.  
  
START-OF-SELECTION.  
    "Collect Data  
    SELECT MATNR MAKTX INTO TABLE GI_DATA  
        UP TO 50 ROWS  
    FROM MAKT  
    WHERE SPRAS = SY-LANGU AND  
        MAKTX IN S_MAKTX.  
  
END-OF-SELECTION.  
    "Display Data  
    LOOP AT GI_DATA.  
        WRITE : / GI_DATA-MATNR, GI_DATA-MAKTX.  
    ENDLOOP.
```

"TABLES : MAKT." is required for SELECT-OPTIONS with sap table as reference

3. Open SQL in ABAP

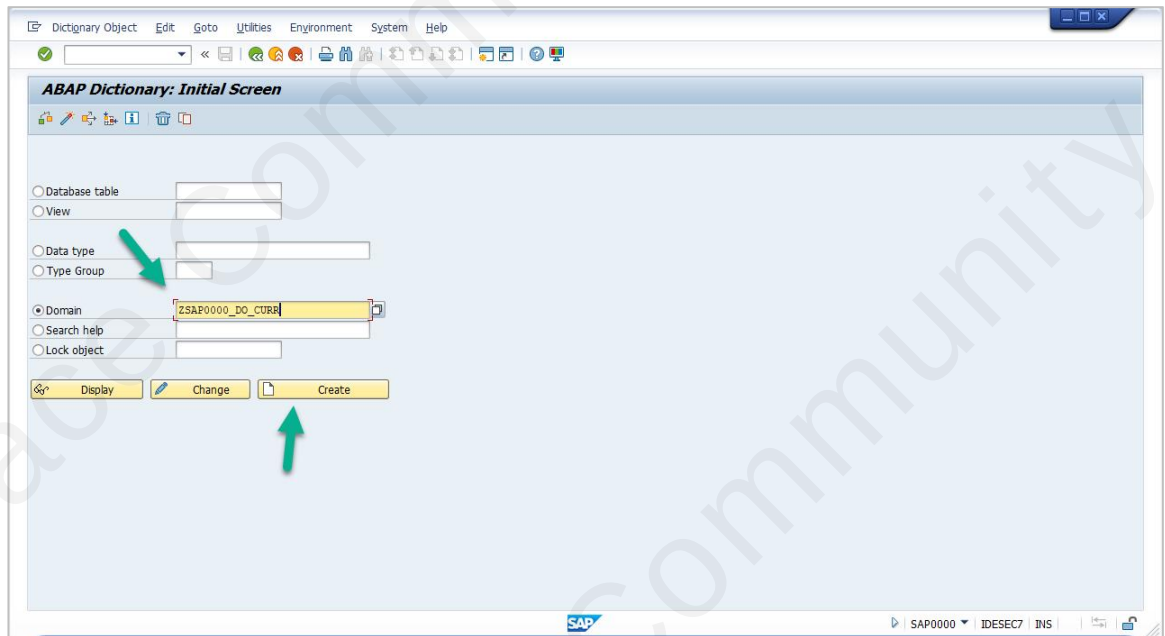
Open SQL refers to the subset (SELECT, INSERT, UPDATE, etc) of ABAP statements that enable direct access to data of the central database of the current ABAP system. The Open SQL statements form the DMLSQL in ABAP, which is supported by all database systems.

Practice Day - 2

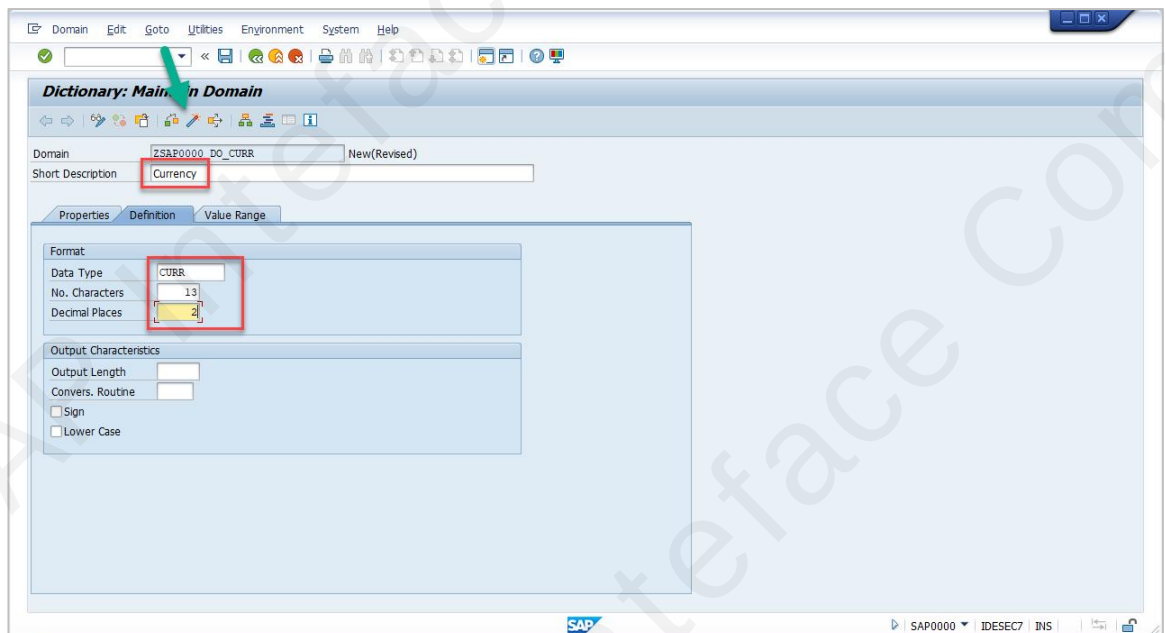
A. Create a Domain 000

Step to create a package:

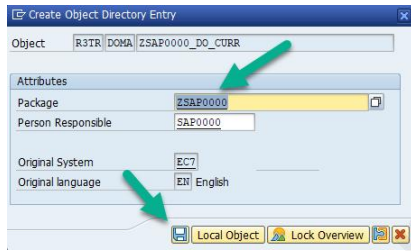
1. Open TCode "SE11"
2. Select "Domain" radiobutton
3. Enter Domain name "ZSAP0000_DO_CURR" and click "Create" button



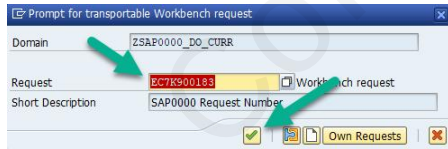
4. Enter value "Currency", "CURR", "13", "2" like this bellow and then click "Activate"



5. Enter package "ZSAP0000" and click "Save"



6. Enter Request Number and click "Continue"

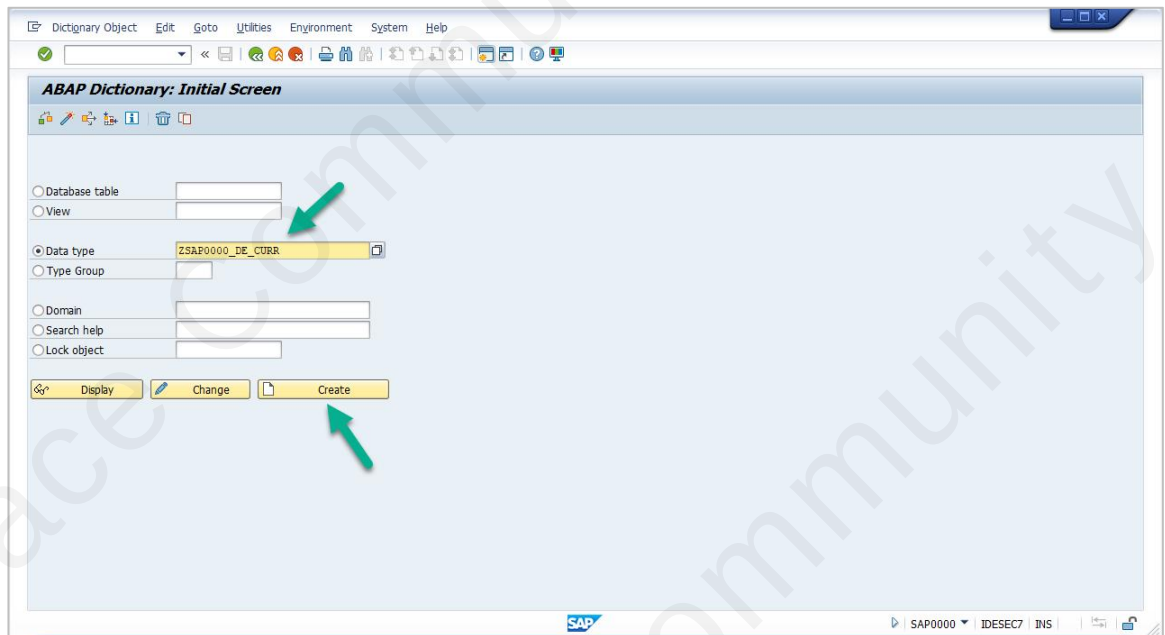


7. Done

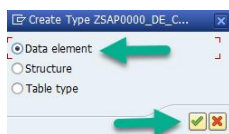
B. Create a Data Element 000

Step to create a package:

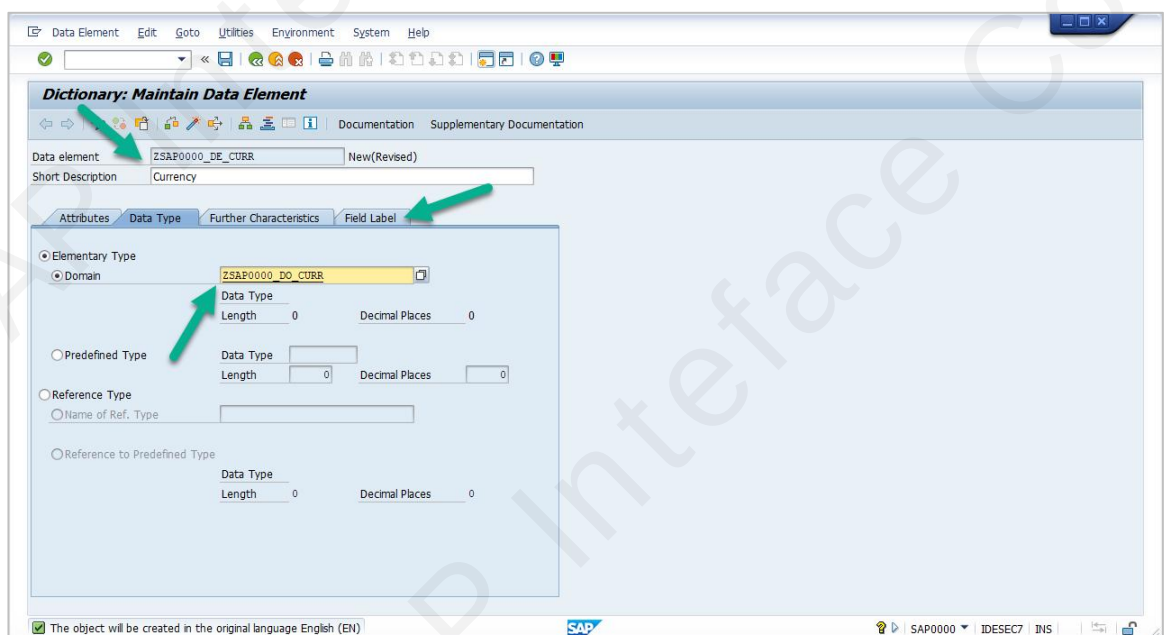
1. Open TCode "SE11"
2. Select "Data type" radiobutton
3. Enter Data type name "ZSAP0000_DE_CURR" and click "Create" button



4. Select "Data Element" radiobutton and click "Continue" button



5. Enter Short Description "Currency", domain "" and then click "Field Label" tab



6. Enter value like this bellow and press Enter

Dictionary: Maintain Data Element

Data element: ZSAP0000_DE_CURR New(Revised)

Short Description: Currency

Attributes Data Type Further Characteristics Field Label

	Length	Field Label
Short		Currency
Medium		Currency
Long		Currency
Heading		Currency

7. Click "Activate" button

Dictionary: Maintain Data Element

Data element: ZSAP0000_DE_CURR New(Revised)

Short Description: Currency

Attributes Data Type Further Characteristics Field Label

	Length	Field Label
Short	10	Currency
Medium	15	Currency
Long	20	Currency
Heading	17	Currency

8. Enter package "ZSAP0000" and click "Save"

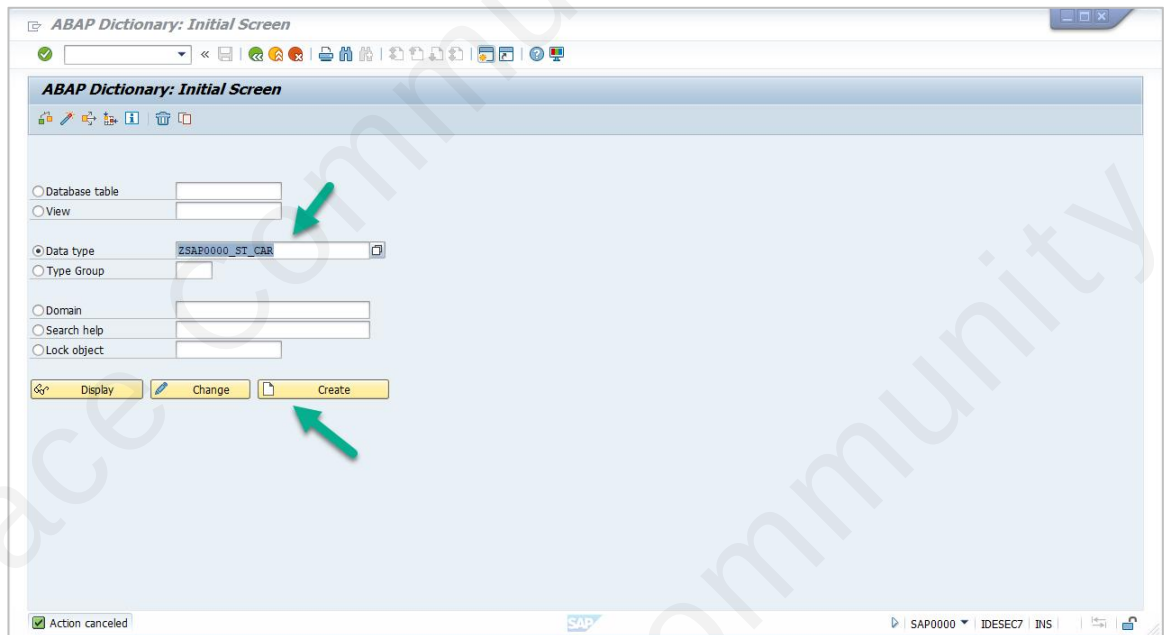
9. Enter Request Number and click "Continue"

10. Done

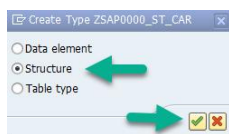
c. Create a Structure 000

Step to create a package:

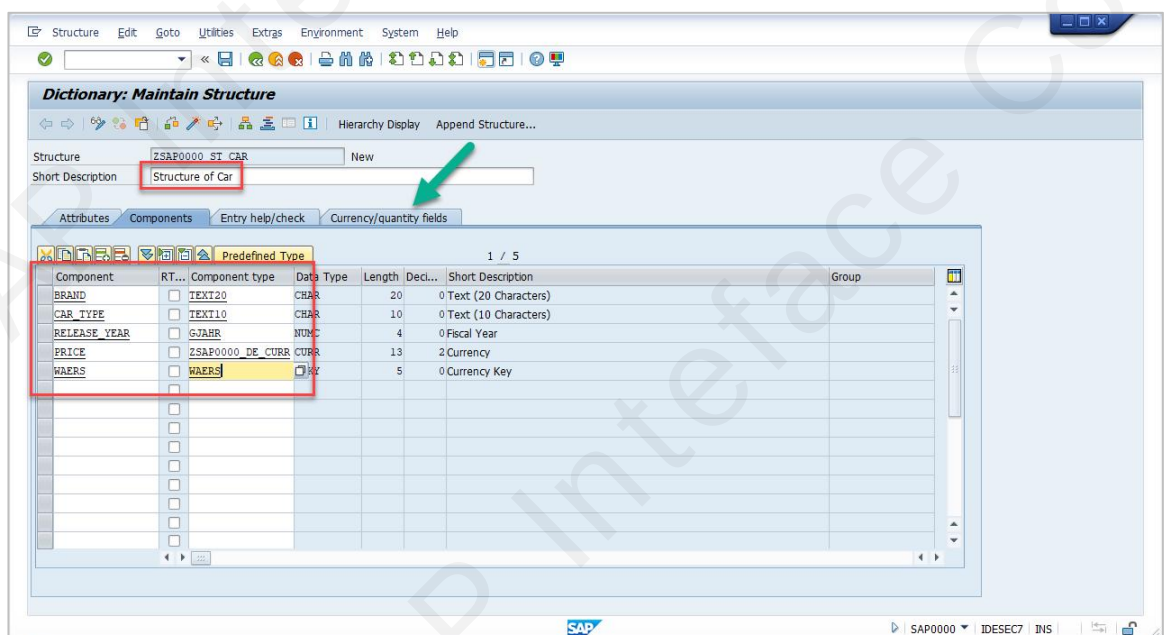
1. Open TCode "SE11"
2. Select "Data type" radiobutton
3. Enter Data type name "ZSAP0000_ST_CAR" and click "Create" button



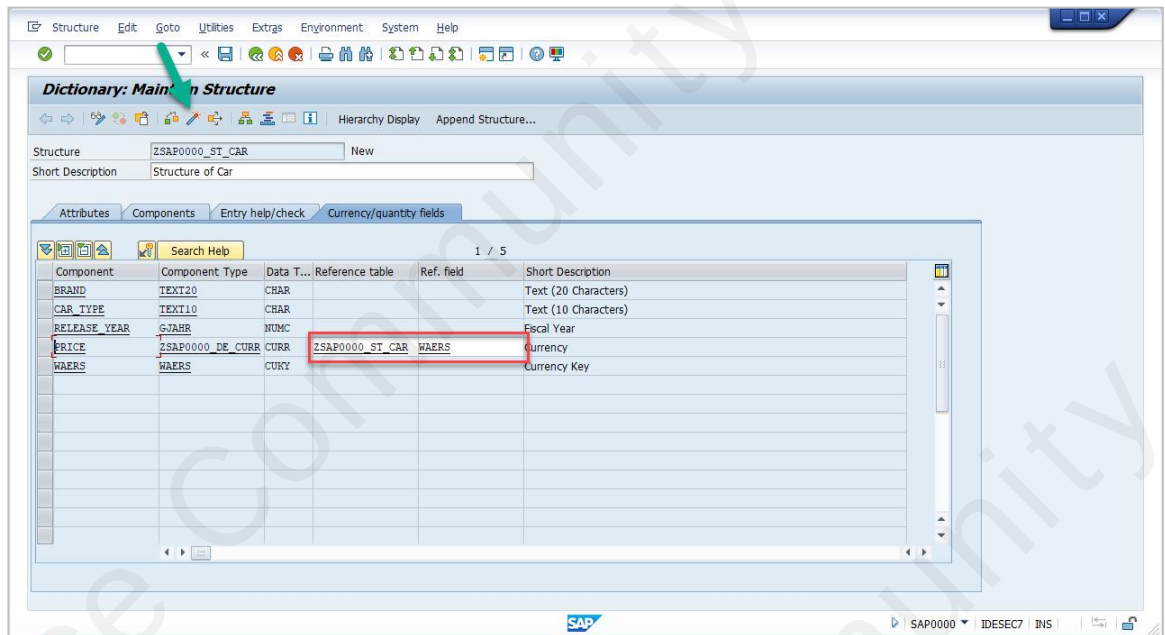
8. Select "Structure" radiobutton and click "Continue" button



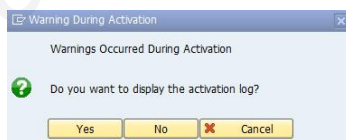
9. Enter Short Description "" and others value like this bellow and then click "Currency/quantity fields" tab



10. Enter reference value and click "Activate" button



11. Click "No"



12. Enter package "ZSAP0000" and click "Save"

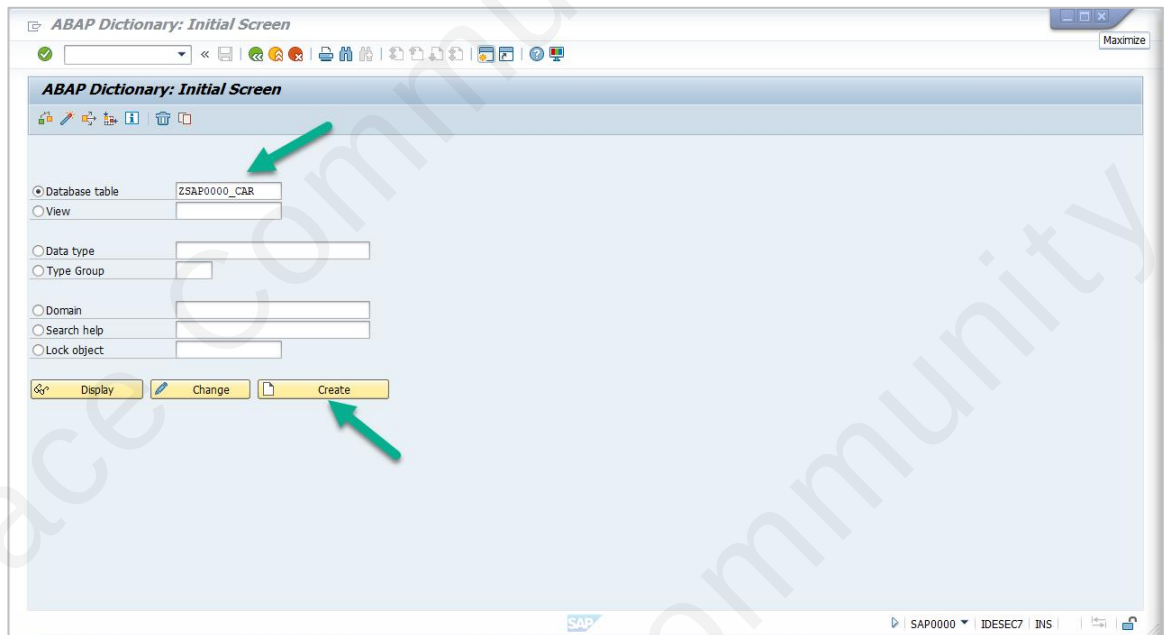
13. Enter Request Number and click "Continue"

14. Done

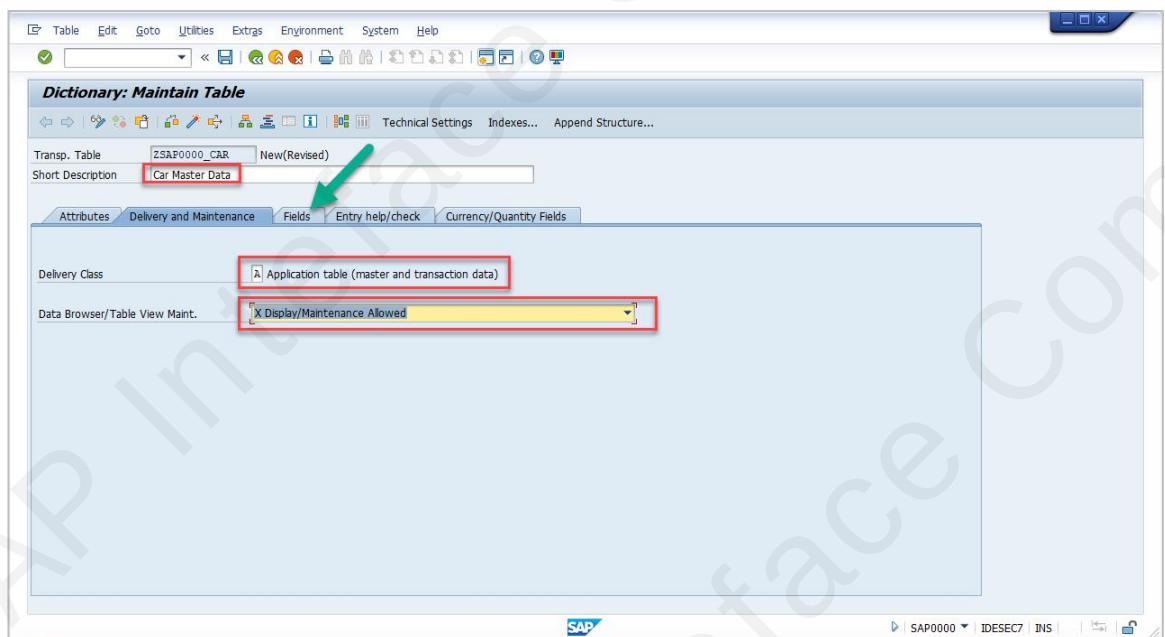
D. Create a Table 000

Step to create a package:

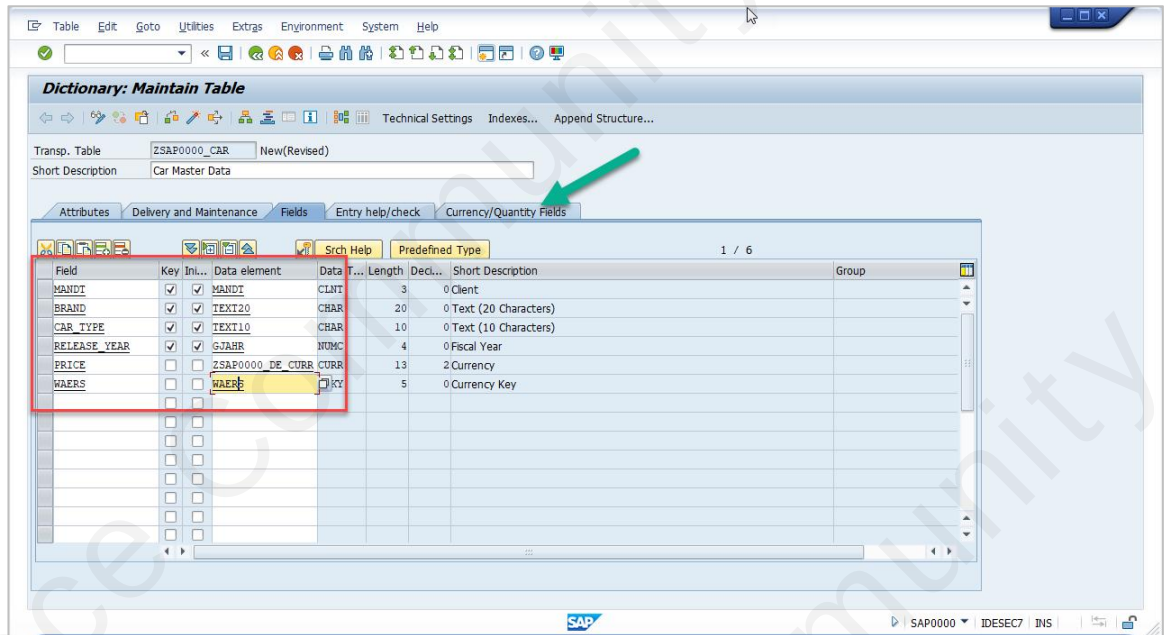
1. Open TCode "SE11"
2. Select "Database table" radiobutton
3. Enter Database table "ZSAP0000_CAR" and click "Create" button



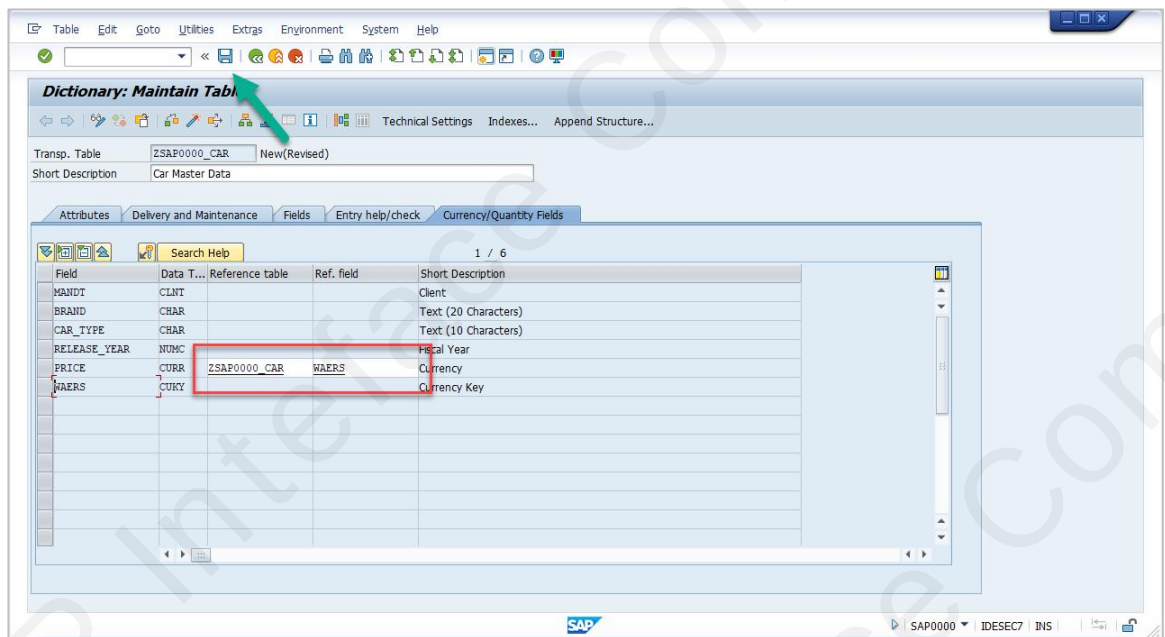
4. Enter value like this bellow and then click "Fields" tab



5. Enter value like this bellow and then click "Currency/quantity fields" tab

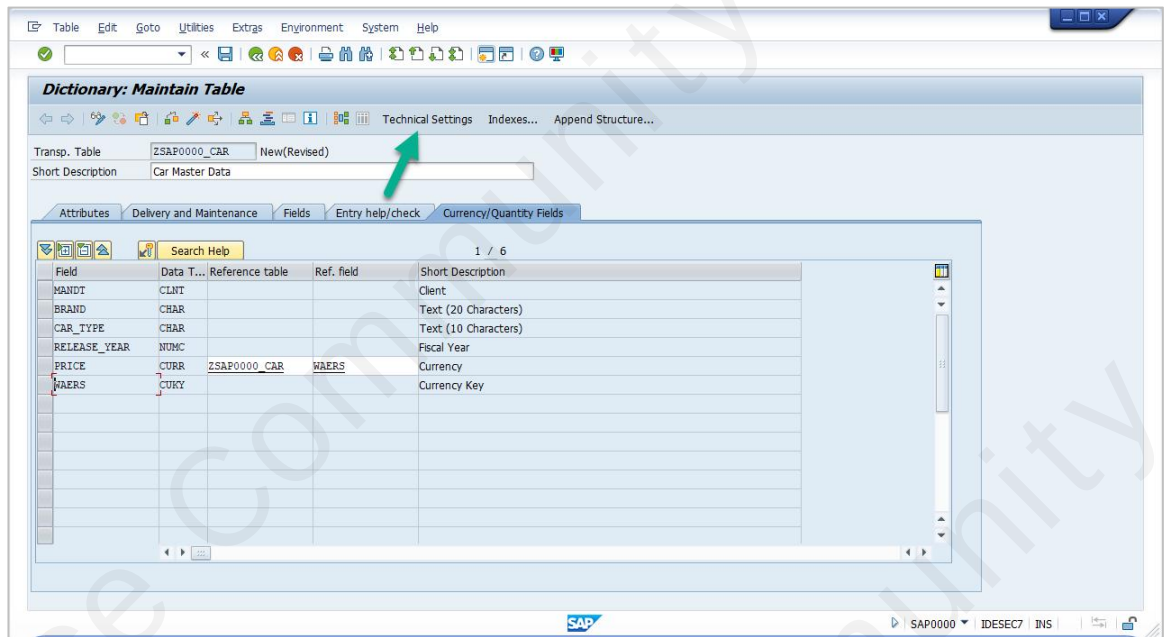


6. Enter reference value and then click "Save" button

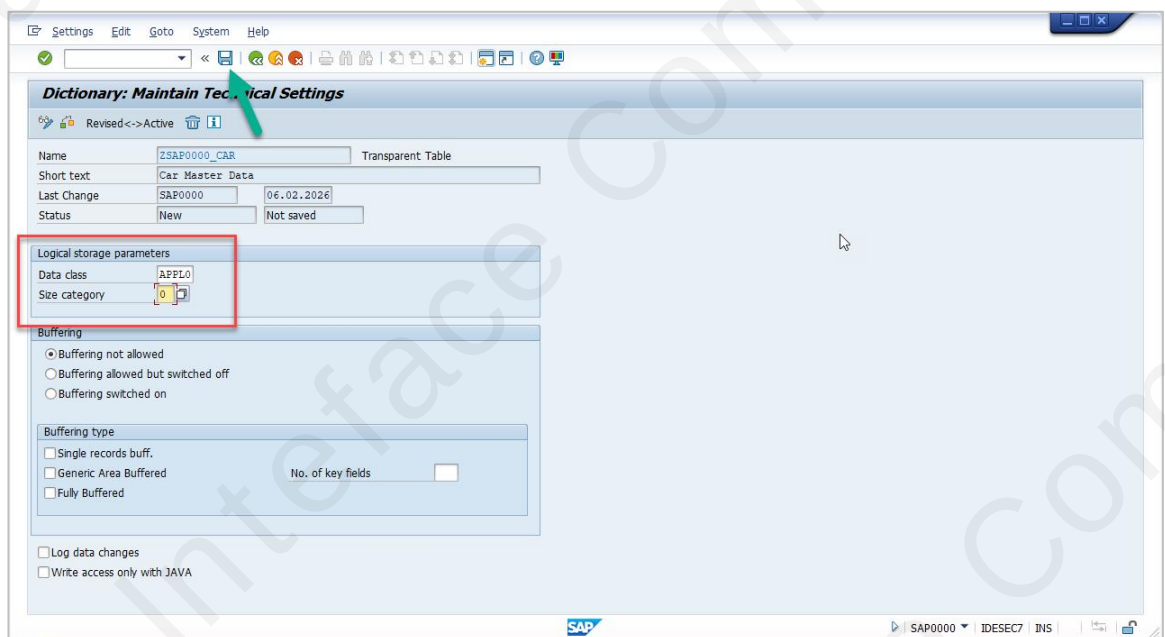


15. Enter package "ZSAP0000" and click "Save"
16. Enter Request Number and click "Continue"

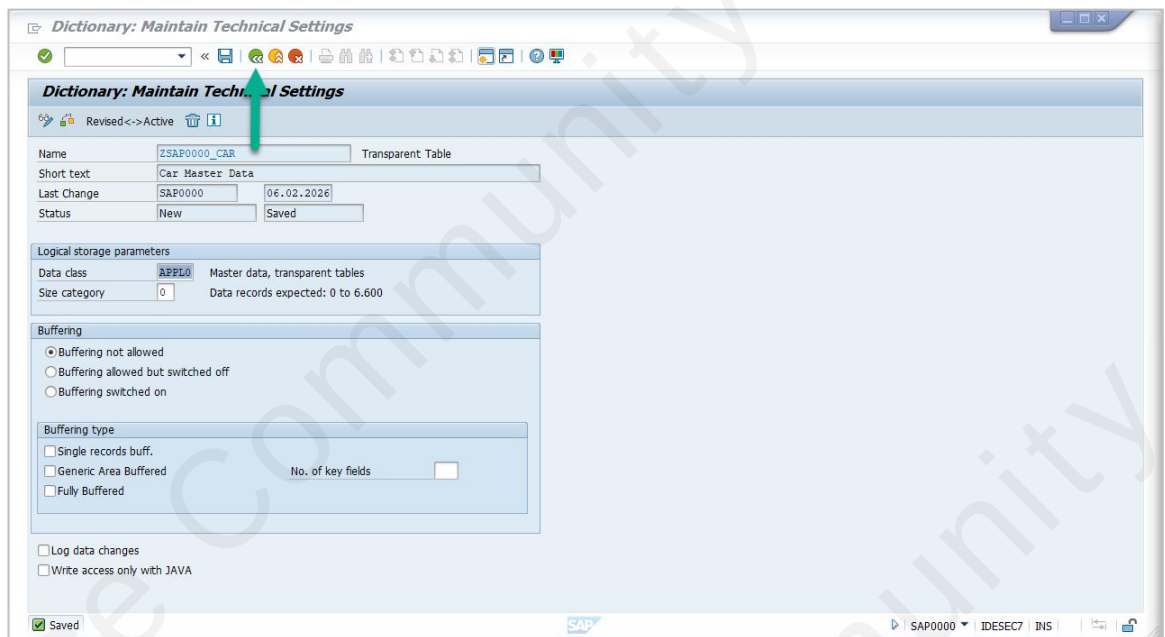
17. Click "Technical Settings" button



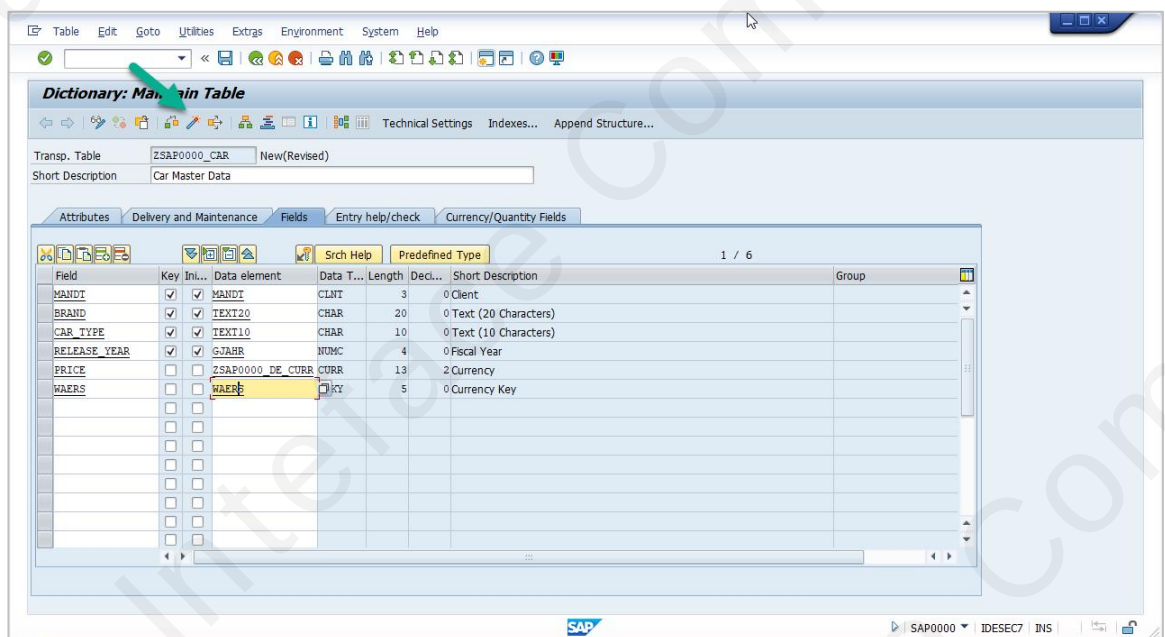
18. Enter Logical storage parameters and click "save" button



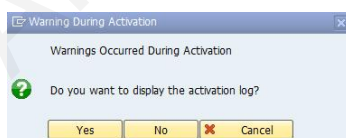
19. Click "Back" or press "ESC"



20. Click "Activate" button



21. Click "No"



22. Done

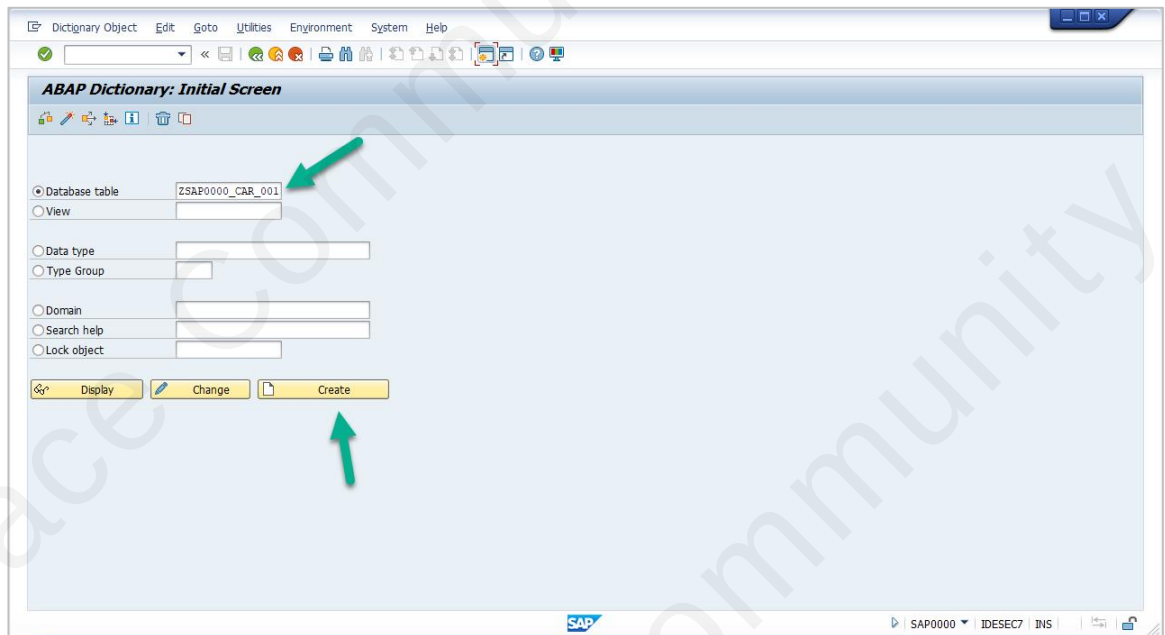
E. Create a Table 001

Step to create a package:

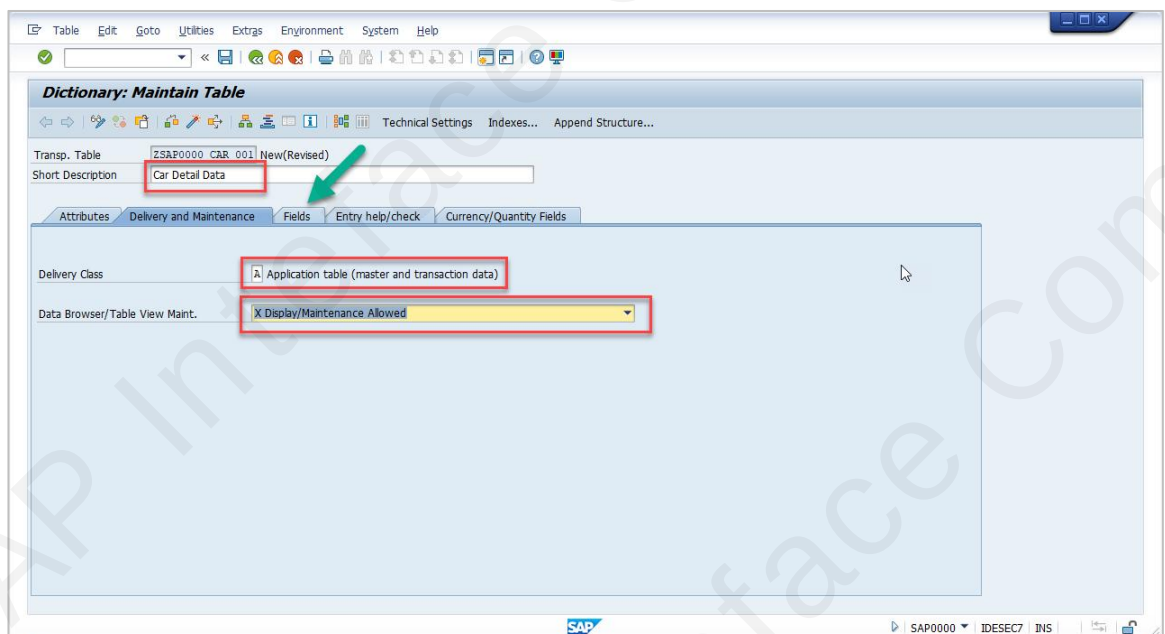
7. Open TCode "SE11"

8. Select "Database table" radiobutton

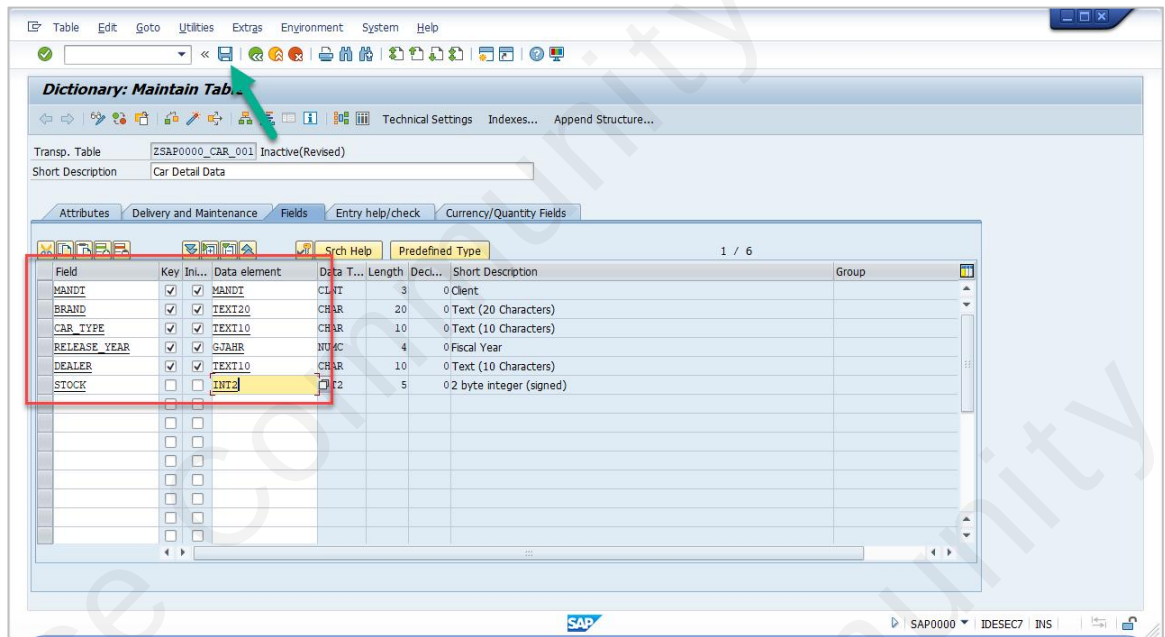
9. Enter Database table "ZSAP0000_CAR_001" and click "Create" button



10. Enter value like this bellow and then click "Fields" tab



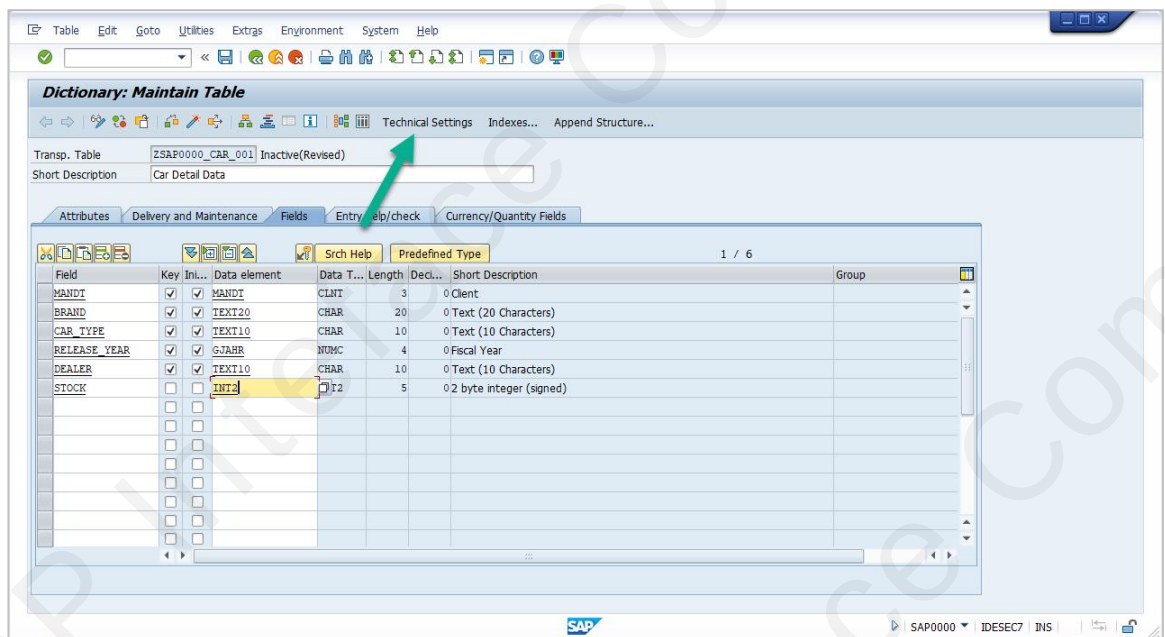
11. Enter value like this bellow and then click "Save" button



23. Enter package "ZSAP0000" and click "Save"

24. Enter Request Number and click "Continue"

25. Click "Technical Settings" button



26. Enter Logical storage parameters and click "save" button

Dictionary: Maintain Technical Settings

Revised <-> Active

Name: ZSAP0000_CAR_001 Transparent Table

Short text: Car Detail Data

Last Change: SAP0000 06.02.2026

Status: New Not saved

Logical storage parameters

Data class: APFL0

Size category: 0

Buffering

☒ Buffering not allowed

☐ Buffering allowed but switched off

☐ Buffering switched on

Buffering type

☐ Single records buff.

☐ Generic Area Buffered

☐ Fully Buffered

No. of key fields:

☐ Log data changes

☐ Write access only with JAVA

SAP0000 IDESEC7 INS

27. Click "Back" or press "ESC"

Dictionary: Maintain Technical Settings

Revised <-> Active

Name: ZSAP0000_CAR_001 Transparent Table

Short text: Car Detail Data

Last Change: SAP0000 06.02.2026

Status: New Not saved

Logical storage parameters

Data class: APFL0

Size category: 0

Buffering

☒ Buffering not allowed

☐ Buffering allowed but switched off

☐ Buffering switched on

Buffering type

☐ Single records buff.

☐ Generic Area Buffered

☐ Fully Buffered

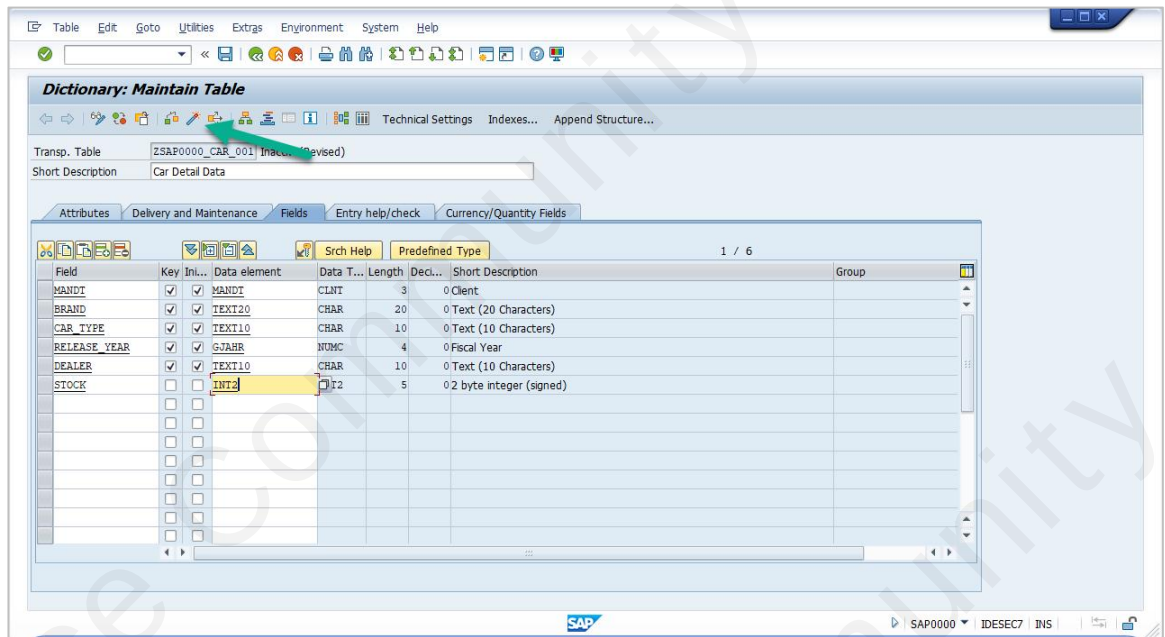
No. of key fields:

☐ Log data changes

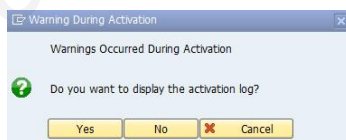
☐ Write access only with JAVA

SAP0000 IDESEC7 INS

28. Click "Activate" button



29. Click "No"

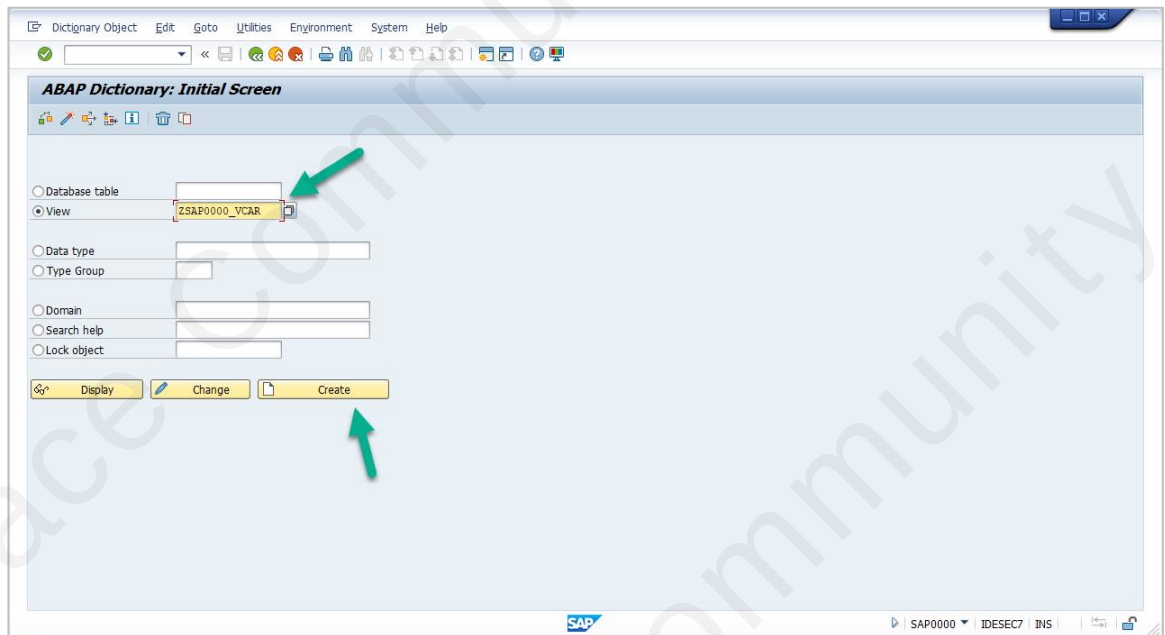


30. Done

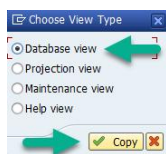
F. Craete a View 000

Step to create a package:

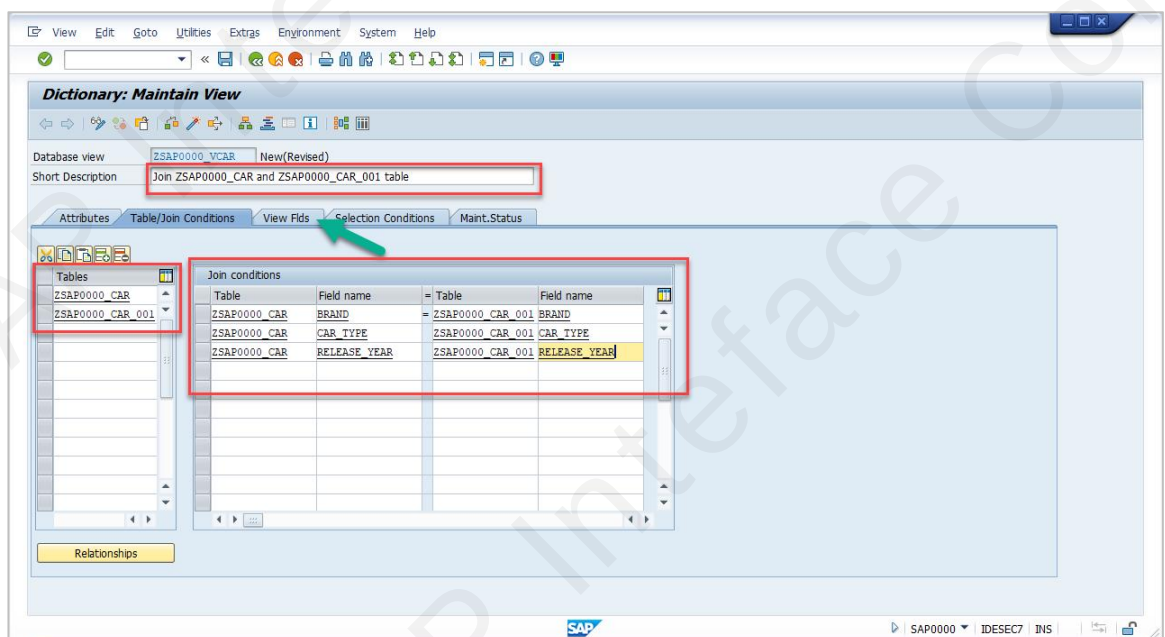
1. Open TCode "SE11"
2. Select "View" radiobutton
3. Enter Domain name "ZSAP0000_VCAR" and click "Create" button



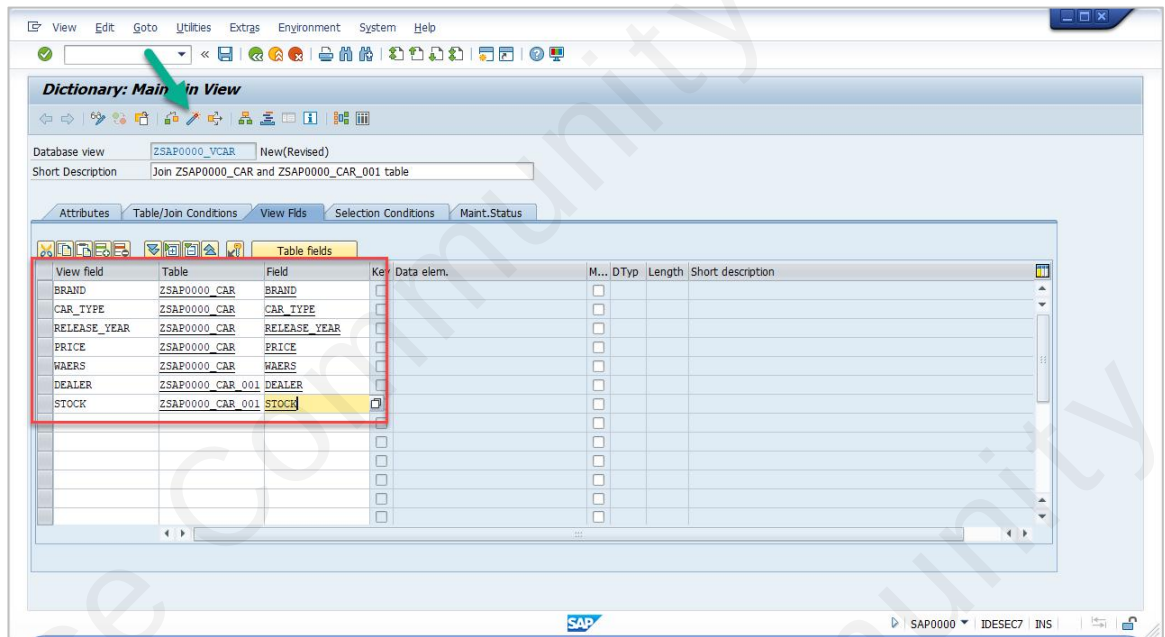
4. Select "Database view" radiobutton and click "Copy" button



5. Enter all values like this bellow and click "View Flds" tab



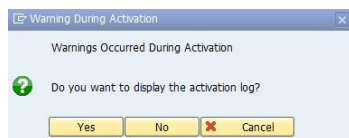
6. Enter all View Fields value and then click "Activate" button



7. Enter package "ZSAP0000" and click "Save"

8. Enter Request Number and click "Continue"

9. Click "No"



10.

11. Done

G. Create a Report 009

Report Name : ZSAP0000_009

Source : Example Report 009

H. Create a Report 010

Report Name : ZSAP0000_010

Source : Example Report 010

I. Create a Report 011

Report Name : ZSAP0000_011

Source : Example Report 011

J. Create a Report 012

Report Name : ZSAP0000_012

Source : Example Report 012

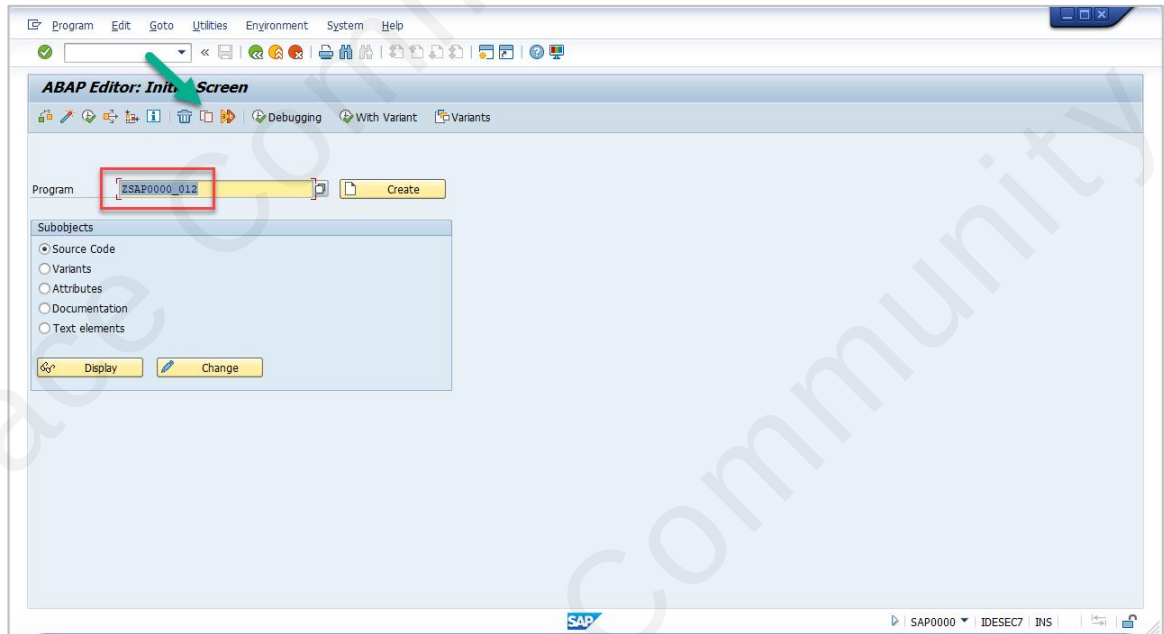
K. Create a Report 012A

Report Name : ZSAP0000_012A

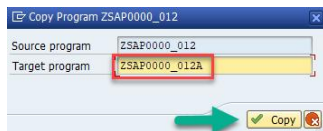
Source : Modified using the Different Table

Step to create a report :

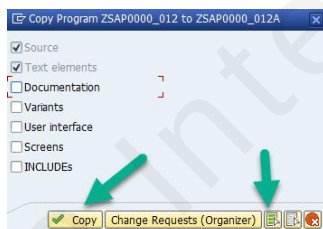
1. Open TCode SE38
2. Enter Program Name "ZSAP0000_012" and click "Copy" button



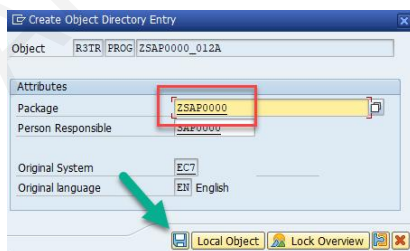
3. Enter target program "ZSAP0000_012A" and click "Copy" button



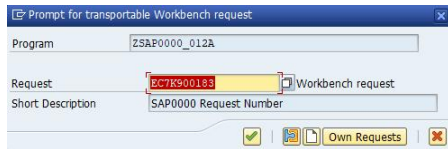
4. Click "Select All" and then click "Copy" button



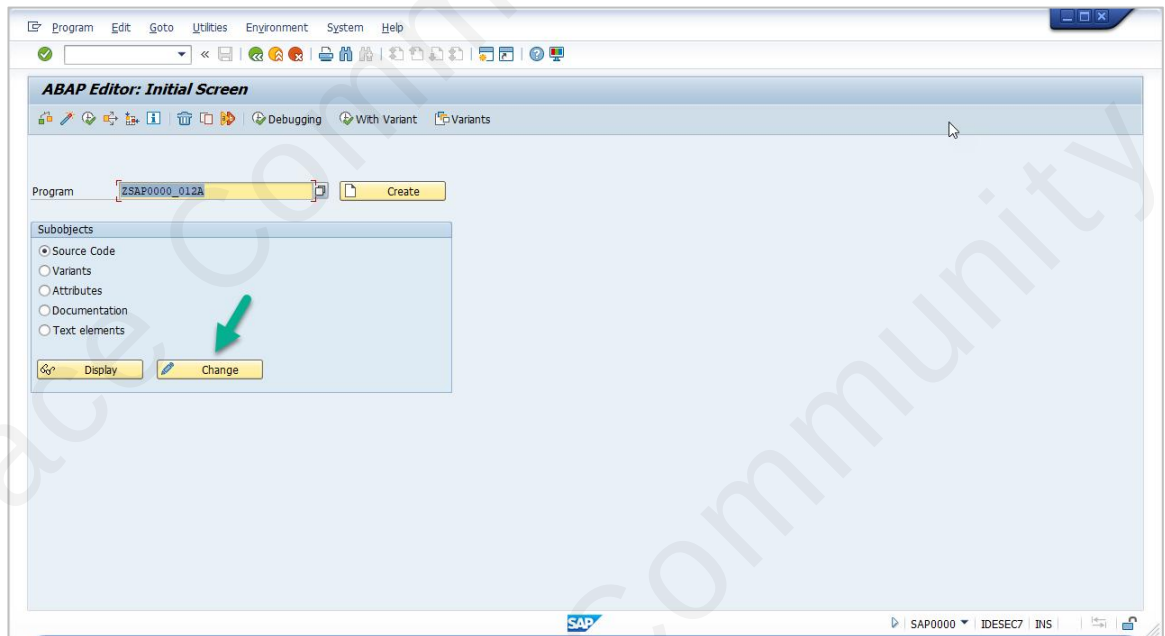
5. Enter Package "ZSAP0000" and Click "Save" button



6. Click "Continue"



7. Click "Change" button



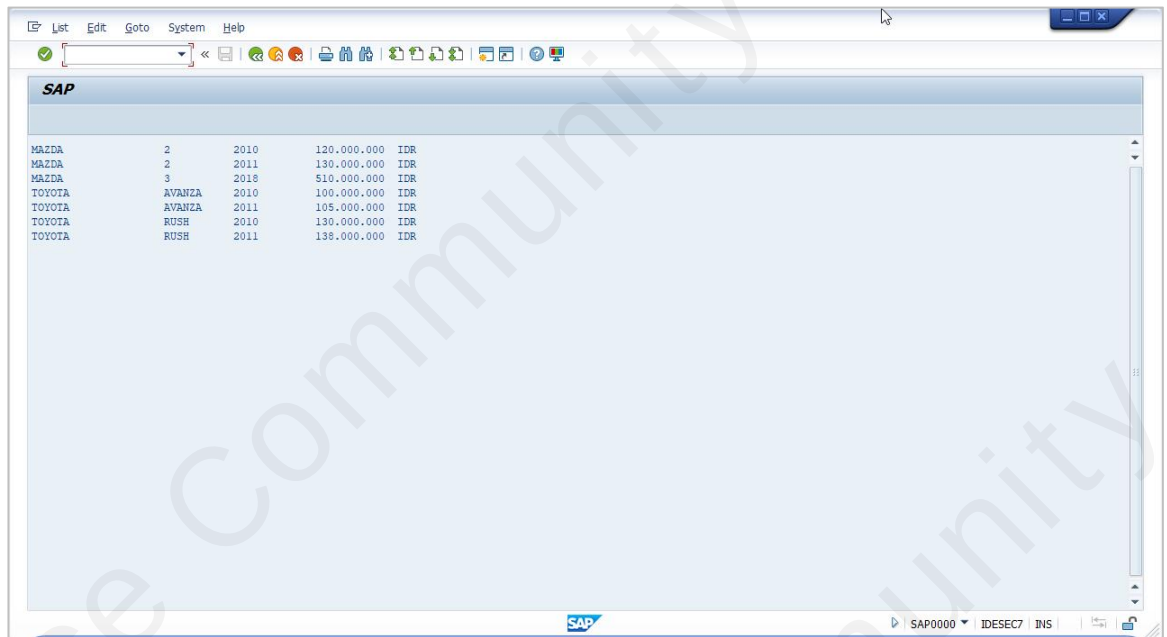
8. Modify Code like this bellow and then click "Activate"

```

1  REPORT ZSAP0000_012A.
2
3  TABLES : ZSAP0000_CAR.
4
5  TYPES : BEGIN OF TY_DATA,
6           BRAND LIKE ZSAP0000_CAR-BRAND,
7           CAR_TYPE LIKE ZSAP0000_CAR-CAR_TYPE,
8           RELEASE_YEAR LIKE ZSAP0000_CAR-RELEASE_YEAR,
9           PRICE LIKE ZSAP0000_CAR-PRICE,
10          WAERS LIKE ZSAP0000_CAR-WAERS,
11          END OF TY_DATA.
12
13  DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.
14
15  SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.
16
17  START-OF-SELECTION.
18      "Collect Data
19      SELECT BRAND CAR_TYPE RELEASE_YEAR PRICE WAERS
20             INTO TABLE GI_DATA
21             UP TO 50 ROWS
22             FROM ZSAP0000_CAR
23             WHERE BRAND IN S_BRAND.
24
25  END-OF-SELECTION.
26  DATA LV_PRICE TYPE C LENGTH 20. " 1 - Buat Variable
27
28  "Display Data
29  LOOP AT GI_DATA.
30      WRITE GI_DATA-PRICE TO LV_PRICE CURRENCY GI_DATA-WAERS. " 2 - Buat Conversi
31
32      WRITE : / GI_DATA-BRAND,
33              GI_DATA-CAR_TYPE,
34              GI_DATA-RELEASE_YEAR,
35              LV_PRICE, "3 - Tampilkan variable baru
36              GI_DATA-WAERS.
37  ENLOOP.

```

9. Click "Direct Processing" and see the result



The screenshot shows the SAP software interface. At the top, there is a menu bar with 'List', 'Edit', 'Goto', 'System', and 'Help'. Below the menu bar is a toolbar with various icons. The main area displays a table with the following data:

SAP				
MAZDA	2	2010	120.000.000	IDR
MAZDA	2	2011	130.000.000	IDR
MAZDA	3	2018	510.000.000	IDR
TOYOTA	AVANZA	2010	100.000.000	IDR
TOYOTA	AVANZA	2011	105.000.000	IDR
TOYOTA	RUSH	2010	130.000.000	IDR
TOYOTA	RUSH	2011	138.000.000	IDR

At the bottom of the window, there is a status bar with the SAP logo and the text 'SAP0000 IDESEC7 INS'.

10. Done

Day - 3

TCode	SE11, SE16, SE38
Target	Understanding efficiency of compound statements and Standard SAP Report

1. Subroutine

SAP ABAP subroutines are modularization units defined by FORM and ENDFORM statements, used to encapsulate reusable code blocks and reduce redundancy within a program. They are invoked using the PERFORM statement. Subroutines can be internal (same program) or external (different program) and support parameter passing via USING and CHANGING.

A. Without Parameter

Example Report 013. Subroutine - Without Parameter

```

TABLES : ZSAP0000_CAR.

TYPES : BEGIN OF TY_DATA,
        BRAND LIKE ZSAP0000_CAR-BRAND,
        CAR_TYPE LIKE ZSAP0000_CAR-CAR_TYPE,
        RELEASE_YEAR LIKE ZSAP0000_CAR-RELEASE_YEAR,
        PRICE LIKE ZSAP0000_CAR-PRICE,
        WAERS LIKE ZSAP0000_CAR-WAERS,
        END OF TY_DATA.

DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.

SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.

START-OF-SELECTION.
    PERFORM FM_COLLECT_DATA.

END-OF-SELECTION.
    PERFORM FM_DISPLAY_DATA.

*&-----*
FORM FM_COLLECT_DATA.
    "Collect Data
    SELECT BRAND CAR_TYPE RELEASE_YEAR PRICE WAERS
        INTO TABLE GI_DATA
        UP TO 50 ROWS
        FROM ZSAP0000_CAR
        WHERE BRAND IN S_BRAND.
    ENDFORM.                                "fm_collect_data

*&-----*
FORM FM_DISPLAY_DATA.
    DATA LV_PRICE TYPE C LENGTH 20. " 1 - Buat Variable

    "Display Data
    LOOP AT GI_DATA.
        WRITE GI_DATA-PRICE TO LV_PRICE CURRENCY GI_DATA-WAERS. " 2 - Buat Conversi

        WRITE : / GI_DATA-BRAND,
            GI_DATA-CAR_TYPE,
            GI_DATA-RELEASE_YEAR,
            LV_PRICE, "3 - Tampilkan variable baru
            GI_DATA-WAERS.

    ENDLOOP.

    ENDFORM.                                "fm_display_data

```

B. With Parameter

Example Report 014. Subroutine - With Parameter Part 1

```
TABLES : ZSAP0000_CAR.

TYPES : BEGIN OF TY_DATA,
        BRAND LIKE ZSAP0000_CAR-BRAND,
        CAR_TYPE LIKE ZSAP0000_CAR-CAR_TYPE,
        RELEASE_YEAR LIKE ZSAP0000_CAR-RELEASE_YEAR,
        PRICE LIKE ZSAP0000_CAR-PRICE,
        WAERS LIKE ZSAP0000_CAR-WAERS,
      END OF TY_DATA.

DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.

SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.

START-OF-SELECTION.
  PERFORM FM_COLLECT_DATA.

END-OF-SELECTION.
  PERFORM FM_DISPLAY_DATA TABLES GI_DATA.

*&-----*
FORM FM_COLLECT_DATA.
  "Collect Data
  SELECT BRAND CAR_TYPE RELEASE_YEAR PRICE WAERS
    INTO TABLE GI_DATA
    UP TO 50 ROWS
  FROM ZSAP0000_CAR
  WHERE BRAND IN S_BRAND.
ENDFORM.                    "fm_collect_data

*&-----*
FORM FM_DISPLAY_DATA TABLES FT_DATA STRUCTURE GI_DATA.
  DATA LV_PRICE TYPE C LENGTH 20. " 1 - Buat Variable

  "Display Data
  LOOP AT FT_DATA.
    WRITE FT_DATA-PRICE TO LV_PRICE CURRENCY FT_DATA-WAERS. " 2 - Buat Conversi

    WRITE : / FT_DATA-BRAND,
            FT_DATA-CAR_TYPE,
            FT_DATA-RELEASE_YEAR,
            LV_PRICE, "3 - Tampilkan variable baru
            FT_DATA-WAERS.

  ENDLOOP.

ENDFORM.                    "fm_display_data
```

Example Report 015. Subroutine - With Parameter Part 2

```
TABLES : ZSAP0000_CAR.

TYPES : BEGIN OF TY_DATA,
        BRAND LIKE ZSAP0000_CAR-BRAND,
        CAR_TYPE LIKE ZSAP0000_CAR-CAR_TYPE,
        RELEASE_YEAR LIKE ZSAP0000_CAR-RELEASE_YEAR,
        PRICE LIKE ZSAP0000_CAR-PRICE,
        WAERS LIKE ZSAP0000_CAR-WAERS,
      END OF TY_DATA.

DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.

SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.
PARAMETERS P_MAXROW TYPE I. "Maximum Rows

START-OF-SELECTION.
  PERFORM FM_COLLECT_DATA TABLES S_BRAND GI_DATA USING P_MAXROW .

END-OF-SELECTION.
  PERFORM FM_DISPLAY_DATA TABLES GI_DATA.

*&-----*
FORM FM_COLLECT_DATA TABLES FT_BRAND STRUCTURE S_BRAND
                        FT_DATA STRUCTURE GI_DATA
                        USING FU_MAXROW .

  "Collect Data
  SELECT BRAND CAR_TYPE RELEASE_YEAR PRICE WAERS
    INTO TABLE FT_DATA
    UP TO FU_MAXROW ROWS
  FROM ZSAP0000_CAR
  WHERE BRAND IN FT_BRAND.
ENDFORM.          "fm_collect_data

*&-----*
FORM FM_DISPLAY_DATA TABLES FT_DATA STRUCTURE GI_DATA.
  DATA LV_PRICE TYPE C LENGTH 20. " 1 - Buat Variable

  "Display Data
  LOOP AT FT_DATA.
    WRITE FT_DATA-PRICE TO LV_PRICE CURRENCY FT_DATA-WAERS. " 2 - Buat Conversi

    WRITE : / FT_DATA-BRAND,
            FT_DATA-CAR_TYPE,
            FT_DATA-RELEASE_YEAR,
            LV_PRICE, "3 - Tampilkan variable baru
            FT_DATA-WAERS.
  ENDLOOP.

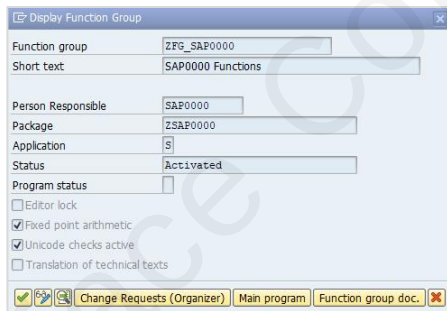
ENDFORM.          "fm_display_data
```

2. Function Module Basic 1

The Function Builder is a tool in the ABAP Workbench that allows you to create, test, and administer function modules in an integrated environment.

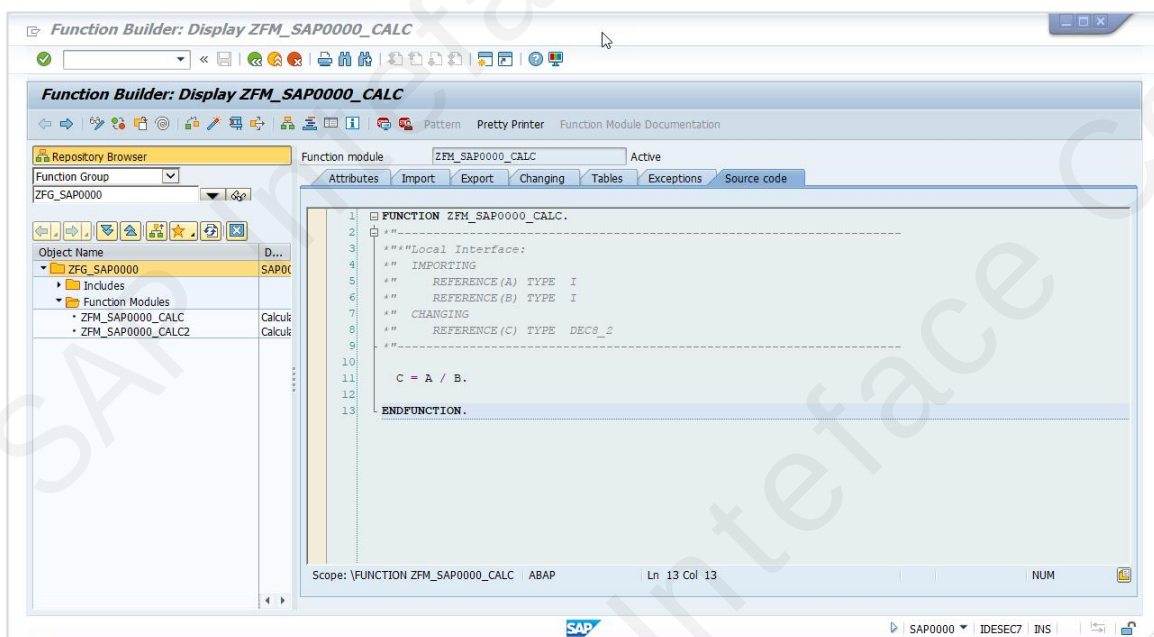
A. Function Group

Function groups (also called function pools) are ABAP programs of a special type. They are the only program type that can contain function modules.



B. Function Module

Function modules are ABAP routines that encapsulate program code and provide an interface for data exchange. Function modules are stored in a central function library. They are not application-specific and are available system-wide. The ABAP Workbench comes with a large number of standard function modules.



Example Function Module Part 1

```
FUNCTION ZFM_SAP0000_CALC.  
*-----  
*""Local Interface:  
*  IMPORTING  
*    REFERENCE(A) TYPE I  
*    REFERENCE(B) TYPE I  
*  CHANGING  
*    REFERENCE(C) TYPE DEC8_2  
*-----  
  
  C = A / B.  
  
ENDFUNCTION.
```

Example Function Module Part 2

```
FUNCTION ZFM_SAP0000_CALC2.  
*-----  
*""Local Interface:  
*  IMPORTING  
*    REFERENCE(A) TYPE I  
*    REFERENCE(B) TYPE I  
*    REFERENCE(OPTION) TYPE C DEFAULT '+'  
*  CHANGING  
*    REFERENCE(C) TYPE DEC8_2  
*-----  
  
  CASE OPTION.  
    WHEN '+'.  
      C = A + B.  
    WHEN '-'.  
      C = A - B.  
    WHEN '*'.  
      C = A * B.  
    WHEN '/'.  
      C = A / B.  
    WHEN OTHERS.  
      C = A + B.  
  ENDCASE.  
  
ENDFUNCTION.
```

3. Output List Display

A. Basic List

Basic lists enable you to output detailed overviews.

The screenshot shows the SAP selection screen for transaction ZSAP0000_101. The interface includes a menu bar (Program, Edit, Goto, System, Help) and a toolbar. The main area is divided into two sections: 'Selection Parameter' and 'Page Setup'. The 'Selection Parameter' section has a 'Brand' field with a dropdown arrow and a search icon. The 'Page Setup' section contains fields for 'Maximum Rows' (set to 1,000), 'Max. Width' (set to 75), 'Max. Line per Page' (set to 10), and 'Delimiter' (set to '|'). The SAP logo and status bar (SAP0000, IDESEC7, INS) are visible at the bottom.

The screenshot shows the SAP output list display for transaction ZSAP0000_101. The table displays car data with columns: NO, Brand, Car Type, Year, Price, and CuKey. The data is as follows:

NO	Brand	Car Type	Year	Price	CuKey
0000000001	HAZDA	2	2010	120.000.000	IDR
0000000002	HAZDA	2	2011	130.000.000	IDR
0000000003	HAZDA	3	2010	510.000.000	IDR
0000000004	TOYOTA	AVANZA	2010	100.000.000	IDR
0000000005	TOYOTA	AVANZA	2011	105.000.000	IDR
0000000006	TOYOTA	RUSH	2010	130.000.000	IDR
0000000007	TOYOTA	RUSH	2011	138.000.000	IDR

The SAP logo and status bar (SAP0000, IDESEC7, INS) are visible at the bottom.

Example Report 101. Output List - Basic

```

. . .

END-OF-SELECTION.
  PERFORM FM_DISPLAY_DATA TABLES GI_DATA USING '' 'X'.

TOP-OF-PAGE.
  PERFORM FM_DISPLAY_DATA TABLES GI_DATA USING 'X' ''.

. . .

FORM FM_DISPLAY_DATA TABLES FT_DATA STRUCTURE GI_DATA USING FU_HEADER FU_DATA.
  DATA : LV_TOTAL TYPE N LENGTH 10,
          LV_COUNT TYPE N LENGTH 10,
          LV_COUNT_PAGE TYPE N LENGTH 10,
          LV_INDEX TYPE N LENGTH 10.
  LV_TOTAL = LINES( FT_DATA ).

  DATA LV_H0(10) VALUE 'NO'.
  DATA LV_H1(20) VALUE 'Brand'.
  DATA LV_H2(10) VALUE 'Car Type'.
  DATA LV_H3(4) VALUE 'Year'.
  DATA LV_H4(13) VALUE 'Price'.
  DATA LV_H5(5) VALUE 'CuKey'.

  "Header
  IF FU_HEADER = 'X'.
    ULINE AT (P_MWIDTH).
    FORMAT COLOR COL_HEADING INTENSIFIED.
    WRITE : / P_DELM NO-GAP,
              LV_H0 CENTERED, P_DELM NO-GAP,
              LV_H1 CENTERED, P_DELM NO-GAP,
              LV_H2 CENTERED, P_DELM NO-GAP,
              LV_H3 CENTERED, P_DELM NO-GAP,
              LV_H4 CENTERED, P_DELM NO-GAP,
              LV_H5 CENTERED, P_DELM NO-GAP.
    FORMAT COLOR OFF.
    ULINE AT /(P_MWIDTH).
  ENDIF.

  "Content
  IF FU_DATA = 'X'.
    LV_COUNT_PAGE = P_MLINE - 3 - 1. " 3 = Total Line Header; 1 = Total Line Footer

    IF LV_COUNT_PAGE < 2.
      MESSAGE 'Minimal 2 Line of Content !!!' TYPE 'I'.
      EXIT.
    ENDIF.

    LOOP AT FT_DATA.
      ADD 1 TO LV_INDEX.
      LV_H0 = LV_INDEX.
      WRITE FT_DATA-BRAND TO LV_H1.
      WRITE FT_DATA-CAR TYPE TO LV_H2.
      WRITE FT_DATA-RELEASE_YEAR TO LV_H3.
      WRITE FT_DATA-PRICE TO LV_H4 CURRENCY FT_DATA-WAERS. " 2 - Buat Conversi
      WRITE FT_DATA-WAERS TO LV_H5.
      WRITE : / P_DELM NO-GAP,
                LV_H0, P_DELM NO-GAP,
                LV_H1, P_DELM NO-GAP,
                LV_H2, P_DELM NO-GAP,
                LV_H3, P_DELM NO-GAP,
                LV_H4, P_DELM NO-GAP,
                LV_H5, P_DELM NO-GAP.

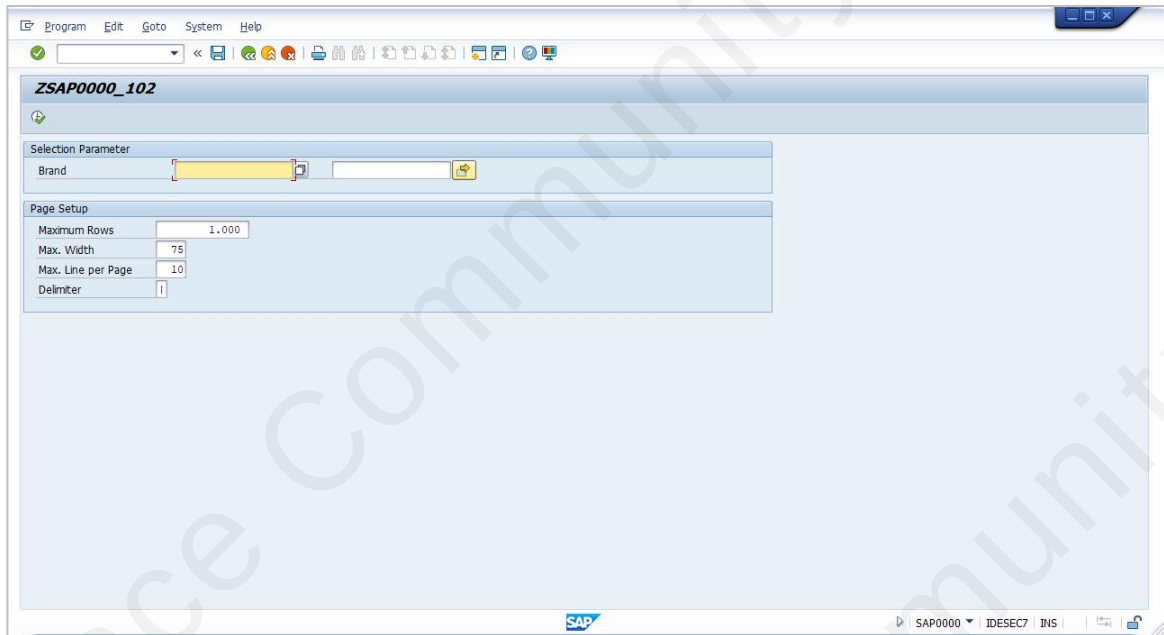
      LV_COUNT = SY-TABIX MOD LV_COUNT_PAGE.
      IF LV_COUNT = 0.
        ULINE AT /(P_MWIDTH).
        NEW-PAGE.
      ENDIF.
    ENDLOOP.

    IF LV_TOTAL = LV_INDEX.
      ULINE AT /(P_MWIDTH).
    ENDIF.
  ENDIF.

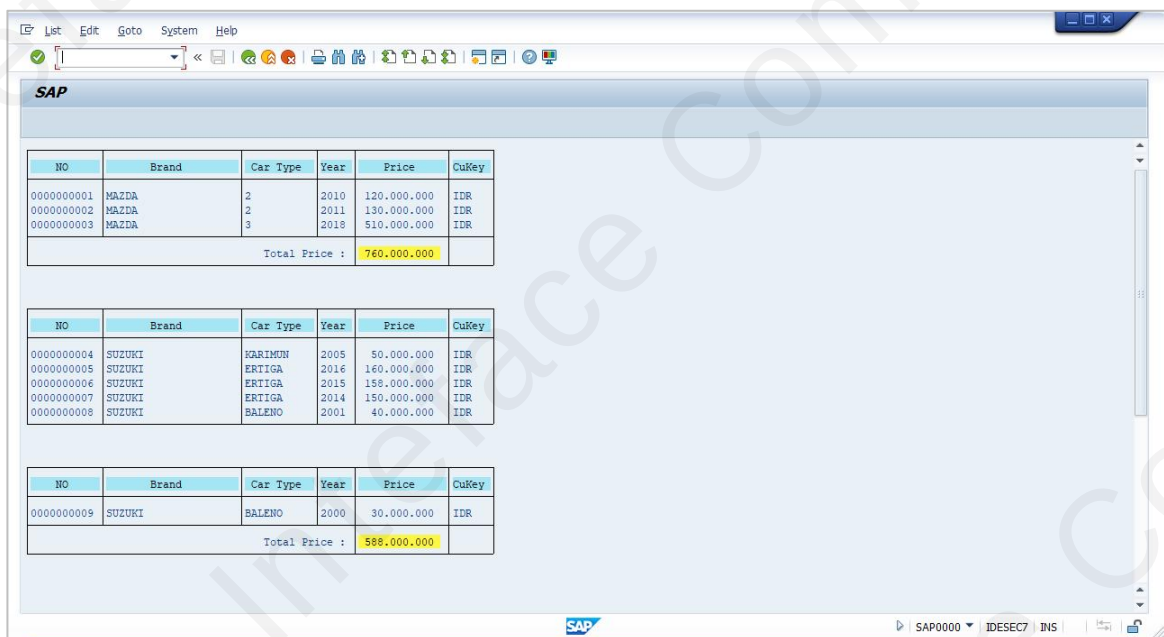
```

B. Statistics

Statistics and ranked lists enable you to output aggregated data.



The screenshot shows the SAP selection screen for transaction ZSAP0000_102. The interface includes a menu bar (Program, Edit, Goto, System, Help) and a toolbar. The main area is divided into two sections: 'Selection Parameter' and 'Page Setup'. The 'Selection Parameter' section has a 'Brand' field with a dropdown arrow and a search icon. The 'Page Setup' section contains fields for 'Maximum Rows' (set to 1,000), 'Max. Width' (set to 75), 'Max. Line per Page' (set to 10), and 'Delimiter' (set to '|'). The SAP logo and status bar (SAP0000, IDESEC7, INS) are visible at the bottom.



The screenshot shows the results of the selection screen ZSAP0000_102. The interface includes a menu bar (List, Edit, Goto, System, Help) and a toolbar. The main area displays three tables of car data. Each table has columns for NO, Brand, Car Type, Year, Price, and CuKey. The first table shows data for MAZDA cars, the second for SUZUKI cars, and the third for SUZUKI cars. Each table is followed by a 'Total Price' row. The SAP logo and status bar (SAP0000, IDESEC7, INS) are visible at the bottom.

NO	Brand	Car Type	Year	Price	CuKey
0000000001	MAZDA	2	2010	120.000.000	IDR
0000000002	MAZDA	2	2011	130.000.000	IDR
0000000003	MAZDA	3	2018	510.000.000	IDR
Total Price :				760.000.000	

NO	Brand	Car Type	Year	Price	CuKey
0000000004	SUZUKI	KARIMUN	2005	50.000.000	IDR
0000000005	SUZUKI	ERTIGA	2016	160.000.000	IDR
0000000006	SUZUKI	ERTIGA	2015	158.000.000	IDR
0000000007	SUZUKI	ERTIGA	2014	150.000.000	IDR
0000000008	SUZUKI	BALENO	2001	40.000.000	IDR

NO	Brand	Car Type	Year	Price	CuKey
0000000009	SUZUKI	BALENO	2000	30.000.000	IDR
Total Price :				588.000.000	

Example Report 102. Output List - Statistics

```

FORM FM_DISPLAY_DATA TABLES FT_DATA STRUCTURE GI_DATA USING FU_HEADER FU_DATA.
DATA : LV_TOTAL TYPE N LENGTH 10,
      LV_COUNT TYPE N LENGTH 10,
      LV_COUNT_PAGE TYPE N LENGTH 10,
      LV_INDEX TYPE N LENGTH 10,
      LV_TOTAL_PRICE LIKE FT_DATA-PRICE,
      LV_HTOTAL_PRICE(13),
      LV_LAST_BRAND LIKE FT_DATA-BRAND.

LV_TOTAL = LINES( FT_DATA ).

DATA LV_H0(10) VALUE 'NO'.
DATA LV_H1(20) VALUE 'Brand'.
DATA LV_H2(10) VALUE 'Car Type'.
DATA LV_H3(4) VALUE 'Year'.
DATA LV_H4(13) VALUE 'Price'.
DATA LV_H5(5) VALUE 'CuKey'.

"Header
IF FU_HEADER = 'X'.
  ULINE AT (P_MWIDTH).
  FORMAT COLOR COL_HEADING INTENSIFIED.
  WRITE : / P_DELM NO-GAP,
          LV_H0 CENTERED, P_DELM NO-GAP,
          LV_H1 CENTERED, P_DELM NO-GAP,
          LV_H2 CENTERED, P_DELM NO-GAP,
          LV_H3 CENTERED, P_DELM NO-GAP,
          LV_H4 CENTERED, P_DELM NO-GAP,
          LV_H5 CENTERED, P_DELM NO-GAP.
  FORMAT COLOR OFF.
  ULINE AT / (P_MWIDTH).
ENDIF.

"Content
IF FU_DATA = 'X'.
  SORT FT_DATA BY BRAND.

  READ TABLE FT_DATA INDEX 1.
  IF SY-SUBRC = 0.
    LV_LAST_BRAND = FT_DATA-BRAND.
  ENDIF.

  LV_COUNT_PAGE = P_MLINE - 3 - 3. " 3 = Total Line Header ; 2 = Total Line Footer
  IF LV_COUNT_PAGE < 2.
    MESSAGE 'Minimal 2 Line of Content !!' TYPE 'I'.
    EXIT.
  ENDIF.

  LOOP AT FT_DATA.
    IF LV_LAST_BRAND NE FT_DATA-BRAND.
      ULINE AT / (P_MWIDTH).
      WRITE LV_TOTAL_PRICE TO LV_HTOTAL_PRICE CURRENCY FT_DATA-WAERS.
      WRITE : / P_DELM NO-GAP,
              2(50) 'Total Price :' RIGHT-JUSTIFIED, P_DELM NO-GAP,
              LV_HTOTAL_PRICE COLOR COL_TOTAL, P_DELM NO-GAP, ' ', P_DELM NO-GAP.
      ULINE AT / (P_MWIDTH).
      NEW-PAGE.
      LV_TOTAL_PRICE = 0.
    ENDIF.

    ADD 1 TO LV_INDEX.
    LV_H0 = LV_INDEX.
    WRITE FT_DATA-BRAND TO LV_H1.
    WRITE FT_DATA-CAR_TYPE TO LV_H2.
    WRITE FT_DATA-RELEASE_YEAR TO LV_H3.
    WRITE FT_DATA-PRICE TO LV_H4 CURRENCY FT_DATA-WAERS. " 2 - Buat Conversi
    WRITE FT_DATA-WAERS TO LV_H5.
    WRITE : / P_DELM NO-GAP,
            LV_H0, P_DELM NO-GAP,
            LV_H1, P_DELM NO-GAP,
            LV_H2, P_DELM NO-GAP,
            LV_H3, P_DELM NO-GAP,
            LV_H4, P_DELM NO-GAP,
            LV_H5, P_DELM NO-GAP.

    ADD FT_DATA-PRICE TO LV_TOTAL_PRICE.

    LV_COUNT = SY-TABIX MOD LV_COUNT_PAGE.
    IF LV_COUNT = 0 AND LV_LAST_BRAND = FT_DATA-BRAND.
      ULINE AT / (P_MWIDTH).
      NEW-PAGE.
    ENDIF.
    LV_LAST_BRAND = FT_DATA-BRAND.
  ENDLOOP.

  IF LV_TOTAL = LV_INDEX.
    ULINE AT / (P_MWIDTH).
    WRITE LV_TOTAL_PRICE TO LV_HTOTAL_PRICE CURRENCY FT_DATA-WAERS.
    WRITE : / P_DELM NO-GAP,
            2(50) 'Total Price :' RIGHT-JUSTIFIED, P_DELM NO-GAP,
            LV_HTOTAL_PRICE COLOR COL_TOTAL, P_DELM NO-GAP, ' ', P_DELM NO-GAP.
    ULINE AT / (P_MWIDTH).
  ENDIF.
ENDIF.
ENDFORM.

```

"fm display data

Practice Day - 3

A. Create a Report 013

Report Name : ZSAP0000_013
Source : Example Report 013

Step to create a package:

1. Open TCode SE38
2. Enter Program Name "ZSAP0000_012A" and Copy to "ZSAP0000_013"
3. Modified code like this bellow and then click "Activate"

```
1 |REPORT ZSAP0000_013.
2 |
3 |TABLES : ZSAP0000_CAR.
4 |
5 |TYPES : BEGIN OF TY_DATA,
6 |         BRAND LIKE ZSAP0000_CAR-BRAND,
7 |         CAR_TYPE LIKE ZSAP0000_CAR-CAR_TYPE,
8 |         RELEASE_YEAR LIKE ZSAP0000_CAR-RELEASE_YEAR,
9 |         PRICE LIKE ZSAP0000_CAR-PRICE,
10 |        WAERS LIKE ZSAP0000_CAR-WAERS,
11 |        END OF TY_DATA.
12 |
13 |DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.
14 |
15 |SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.
16 |
17 |START-OF-SELECTION.
18 |   PERFORM FM_COLLECT_DATA.
19 |
20 |END-OF-SELECTION.
21 |   PERFORM FM_DISPLAY_DATA.
22 |
23 |
24 |*-----
25 |FORM FM_COLLECT_DATA.
26 |  "Collect Data
27 |  SELECT BRAND CAR_TYPE RELEASE_YEAR PRICE WAERS
28 |    INTO TABLE GI_DATA
29 |    UP TO 50 ROWS
30 |    FROM ZSAP0000_CAR
31 |    WHERE BRAND IN S_BRAND.
32 |  ENDFORM.                  "fm_collect_data
33 |
34 |*-----
35 |FORM FM_DISPLAY_DATA.
36 |  DATA LV_PRICE TYPE C LENGTH 20. " 1 - Bust Variable
37 |
38 |  "Display Data
39 |  LOOP AT GI_DATA.
40 |    WRITE GI_DATA-PRICE TO LV_PRICE CURRENCY GI_DATA-WAERS. " 2 - Bust Conversi
41 |
42 |    WRITE : / GI_DATA-BRAND,
43 |            GI_DATA-CAR_TYPE,
44 |            GI_DATA-RELEASE_YEAR,
45 |            LV_PRICE, "3 - Tampilkan variable baru
46 |            GI_DATA-WAERS.
47 |
48 |  ENDLOOP.
49 |  ENDFORM.                  "fm_display_data
50 |
```

You can place "FORM ... ENDFORM." anywhere

4. Done

B. Create a Report 014

Report Name : ZSAP0000_014
Source : Example Report 014

Step to create a package:

1. Open TCode SE38
2. Enter Program Name "ZSAP0000_013" and Copy to "ZSAP0000_014"
3. Modified code like this bellow and then click "Activate"

```
1  REPORT ZSAP0000_014.
2
3  TABLES : ZSAP0000_CAR.
4
5  TYPES : BEGIN OF TY_DATA,
6           BRAND LIKE ZSAP0000_CAR-BRAND,
7           CAR_TYPE LIKE ZSAP0000_CAR-CAR_TYPE,
8           RELEASE_YEAR LIKE ZSAP0000_CAR-RELEASE_YEAR,
9           PRICE LIKE ZSAP0000_CAR-PRICE,
10          WAERS LIKE ZSAP0000_CAR-WAERS,
11          END OF TY_DATA.
12
13 DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.
14
15 SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.
16
17 START-OF-SELECTION.
18   PERFORM FM_COLLECT_DATA.
19
20 END-OF-SELECTION.
21   PERFORM FM_DISPLAY_DATA TABLES GI_DATA.
22
23 -----
24 FORM FM_COLLECT_DATA.
25   "Collect Data
26   SELECT BRAND CAR_TYPE RELEASE_YEAR PRICE WAERS
27     INTO TABLE GI_DATA
28     UP TO 50 ROWS
29     FROM ZSAP0000_CAR
30     WHERE BRAND IN S_BRAND.
31   ENDFORM.
32   "fm_collect_data
33
34 -----
35 FORM FM_DISPLAY_DATA TABLES FT_DATA STRUCTURE GI_DATA.
36   DATA LV_PRICE TYPE C LENGTH 20. " 1 - Subst Variable
37
38   "Display Data
39   LOOP AT FT_DATA.
40     WRITE FT_DATA-PRICE TO LV_PRICE CURRENCY FT_DATA-WAERS. " 2 - Subst Conversi
41
42     WRITE : / FT_DATA-BRAND,
43             FT_DATA-CAR_TYPE,
44             FT_DATA-RELEASE_YEAR,
45             LV_PRICE, "3 - Tampilkan variable baru
46             FT_DATA-WAERS.
47   ENDLOOP.
48   "fm_display_data
49
50 Scope: VFORM FM_DISPLAY_DATA ABAP Ln 48 Col 4
```

"TABLES" parameter used for internal table

4. Done

C. Create a Report 015

Report Name : ZSAP0000_015

Source : Example Report 015

Step to create a package:

1. Open TCode SE38
2. Enter Program Name "ZSAP0000_014" and Copy to "ZSAP0000_015"
3. Modify code like this bellow and then click "Activate"

```
1 REPORT ZSAP0000_015.
2
3 TABLES : ZSAP0000_CAR.
4
5 TYPES : BEGIN OF TY_DATA,
6         BRAND LIKE ZSAP0000_CAR-BRAND,
7         CAR_TYPE LIKE ZSAP0000_CAR-CAR_TYPE,
8         RELEASE_YEAR LIKE ZSAP0000_CAR-RELEASE_YEAR,
9         PRICE LIKE ZSAP0000_CAR-PRICE,
10        WAERS LIKE ZSAP0000_CAR-WAERS,
11      END OF TY_DATA.
12
13 DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.
14
15 SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.
16 PARAMETERS P_MAXROW TYPE I. "Maximum Rows
17
18 START-OF-SELECTION.
19   PERFORM FM_COLLECT_DATA TABLES S_BRAND GI_DATA USING P_MAXROW .
20
21 END-OF-SELECTION.
22   PERFORM FM_DISPLAY_DATA TABLES GI_DATA.
23
24 *-----*
25 FORM FM_COLLECT_DATA TABLES FT_BRAND STRUCTURE S_BRAND
26   FT_DATA STRUCTURE GI_DATA
27   USING FU_MAXROW .
28
29 "Collect Data
30 SELECT BRAND CAR_TYPE RELEASE_YEAR PRICE WAERS
31   INTO TABLE FT_DATA
32   UP TO FU_MAXROW ROWS
33 FROM ZSAP0000_CAR
34 WHERE BRAND IN FT_BRAND.
35 ENDFORM. "fm_collect_data
36
37 *-----*
38 FORM FM_DISPLAY_DATA TABLES FT_DATA STRUCTURE GI_DATA.
39 DATA LV_PRICE TYPE C LENGTH 20. " 1 - Buat Variable
40
41 "Display Data
42 LOOP AT FT_DATA.
43   WRITE FT_DATA-PRICE TO LV_PRICE CURRENCY FT_DATA-WAERS. " 2 - Buat Conversi
44
45   WRITE : / FT_DATA-BRAND,
46           FT_DATA-CAR_TYPE,
47           FT_DATA-RELEASE_YEAR,
48           LV_PRICE, "3 - Tampilkan variable baru
49           FT_DATA-WAERS.
50
51 ENDLOOP.
52 ENDFORM. "fm_display_data
```

"Using " used for single variable or sturcture

4. Done

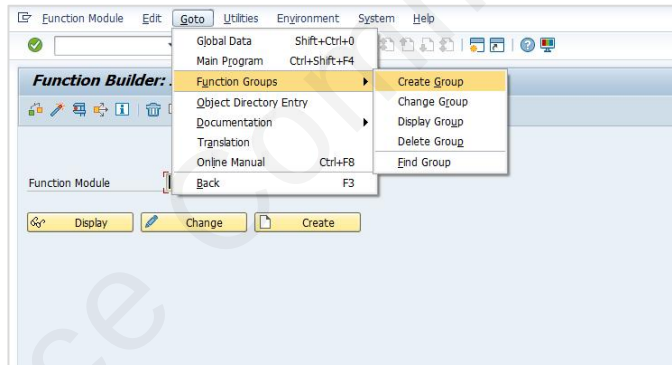
D. Create a Function Group

Function Group : ZFG_SAP0000

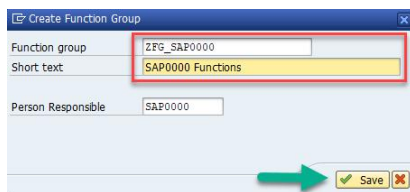
Source : -

Step to create a package:

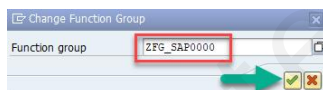
1. Open TCode SE37
2. Open menu "Goto->Function Groups->Create Group"



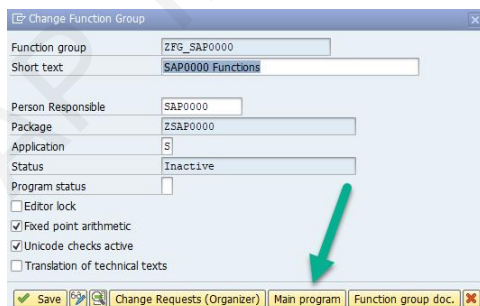
3. Enter values like this bellow and click "Save"



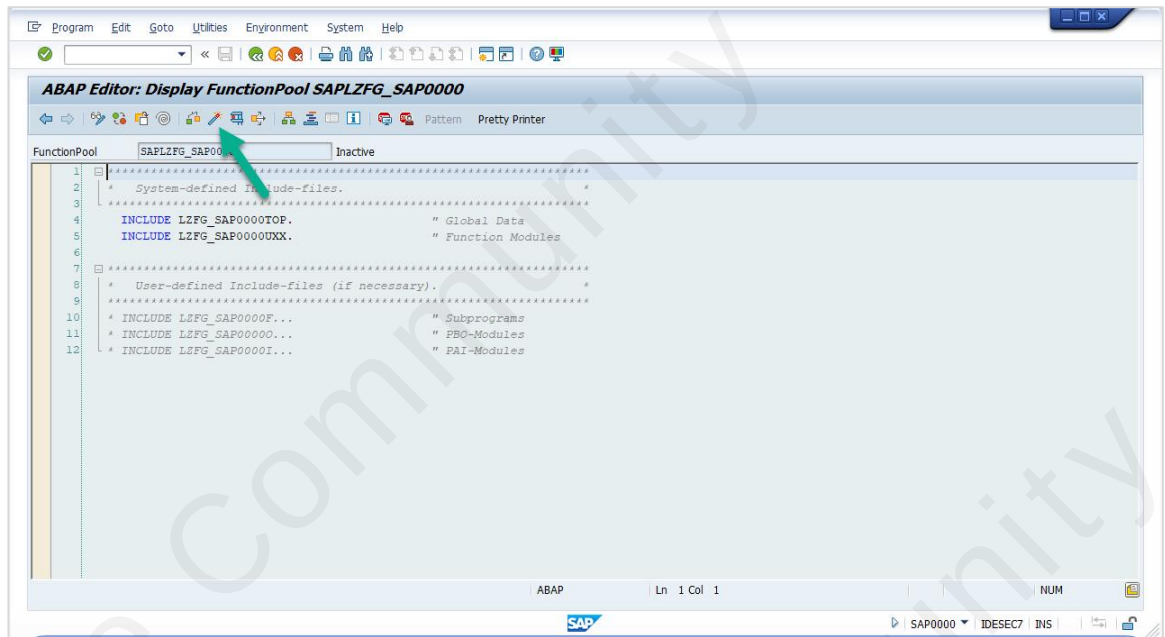
4. Save to "ZSAP0000" package and your request number
5. Open menu "Goto->Function Groups->Change Group"
6. Enter "ZFG_SAP0000" and click "Continue" button



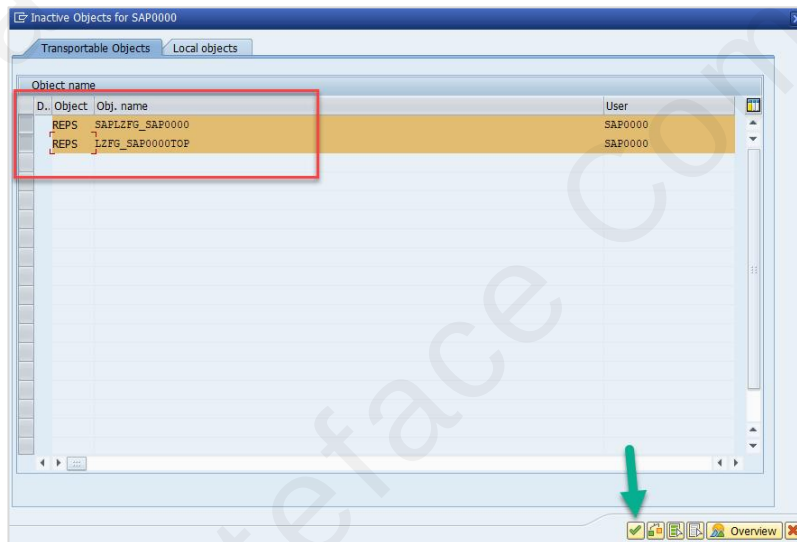
7. Click "Main Program" button



8. Click "Activate" button



9. Select 2 object and click "Continue"



10. Done

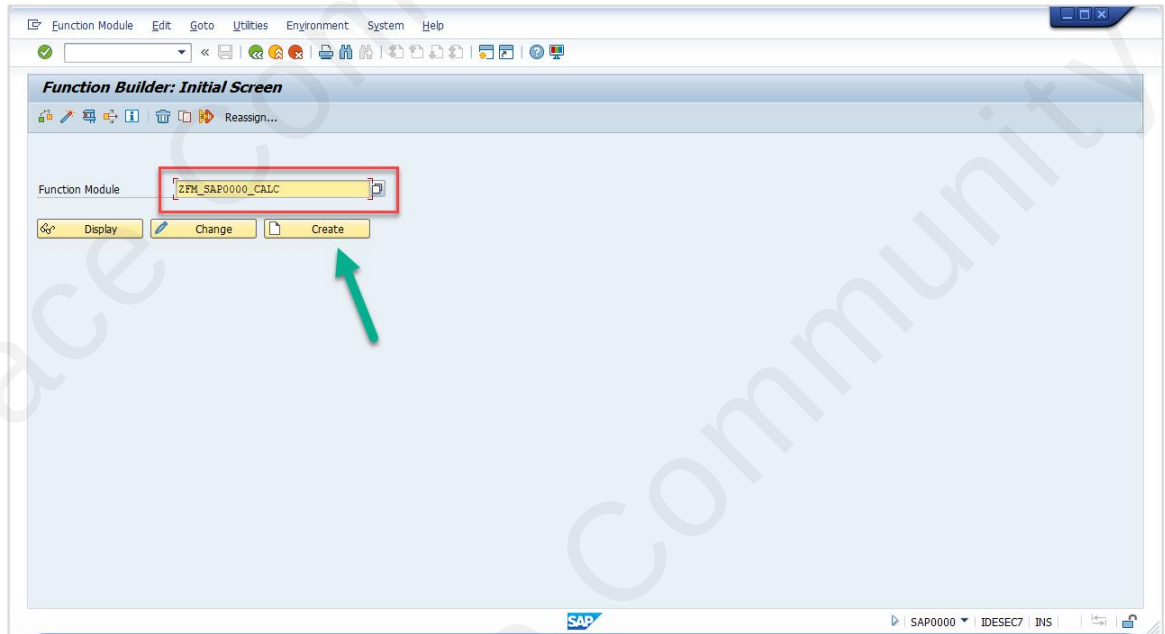
E. Create a Function Module Part 1

Function Module : ZFM_SAP0000_CALC

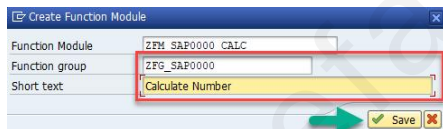
Source : Example Function Part 1

Step to create a package:

1. Open TCode SE37
2. Enter "ZFM_SAP0000_CALC" value as Function Module Name and click "Create" button



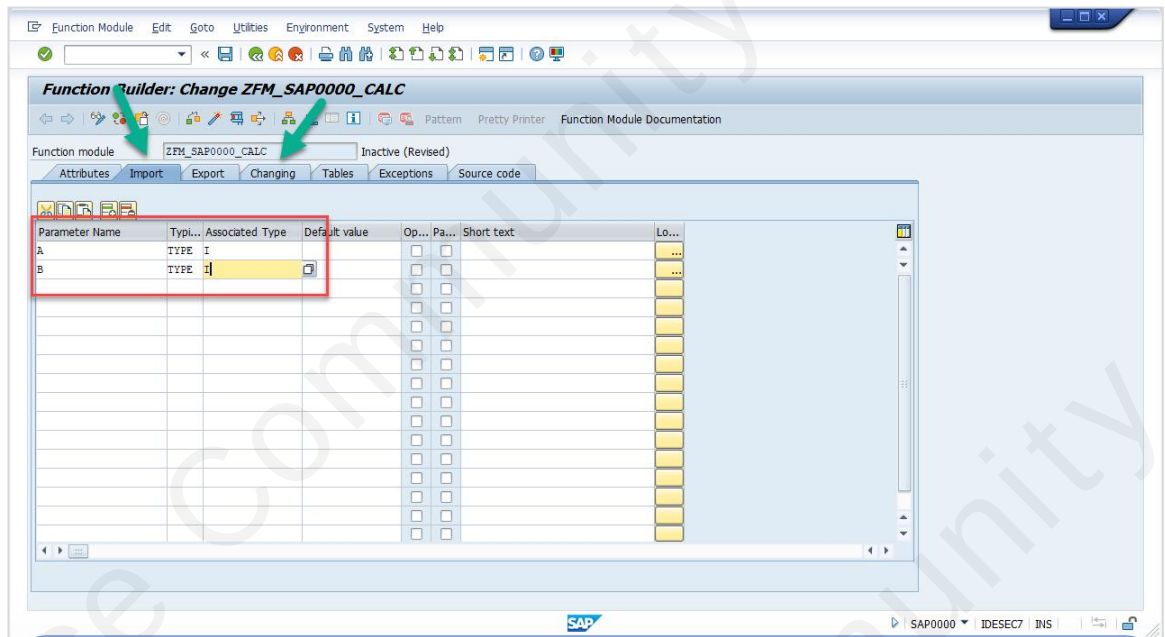
3. Enter value like this bellow and then click "Save"



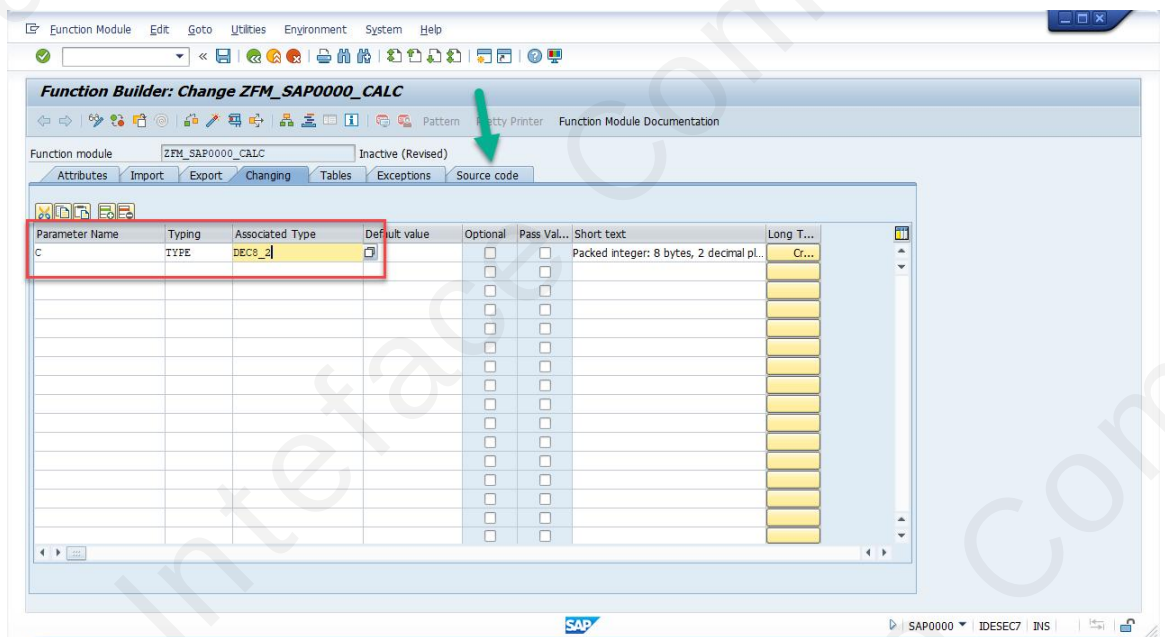
4. Click "Continue"



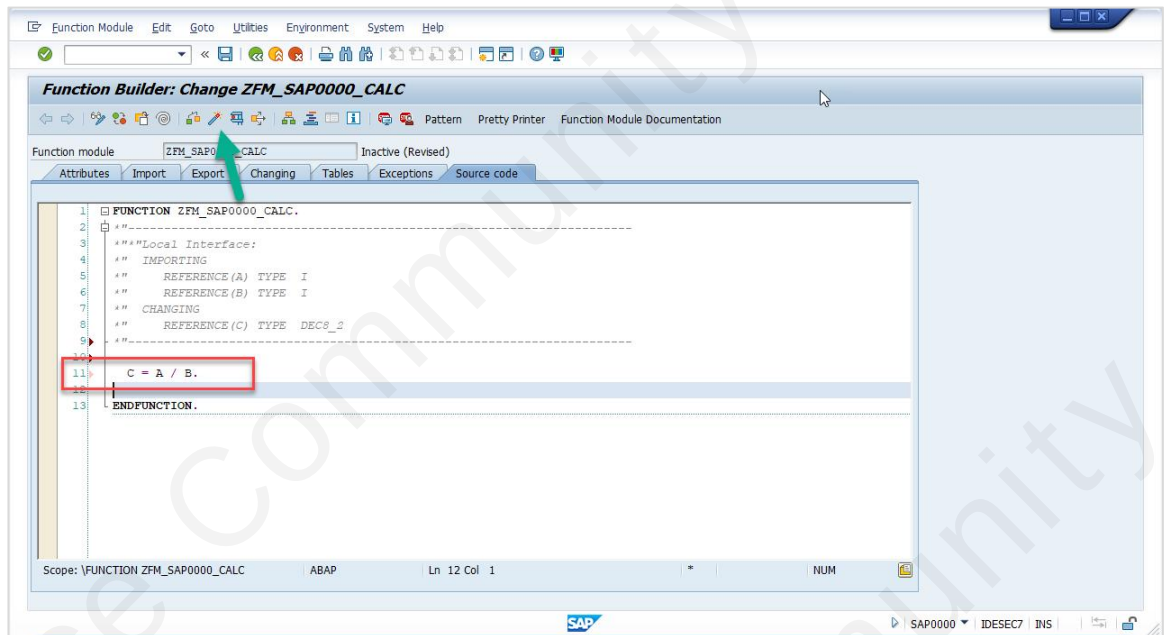
5. Enter parameter like this bellow and then click "Changing" tab



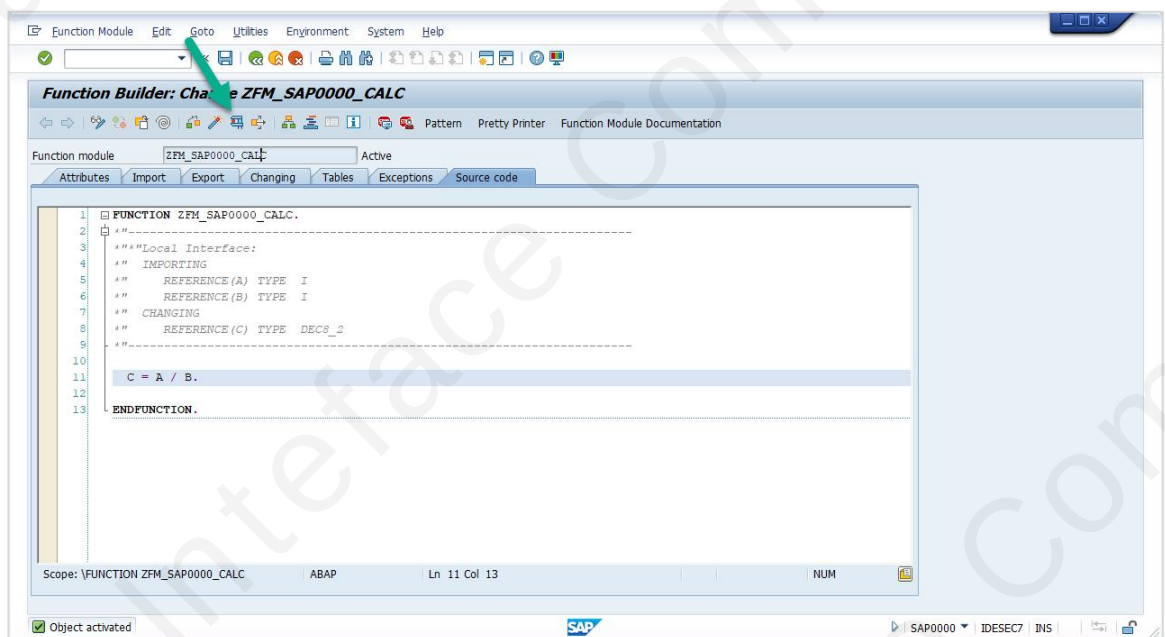
6. Enter parameter like this bellow and then click "Source Code" tab



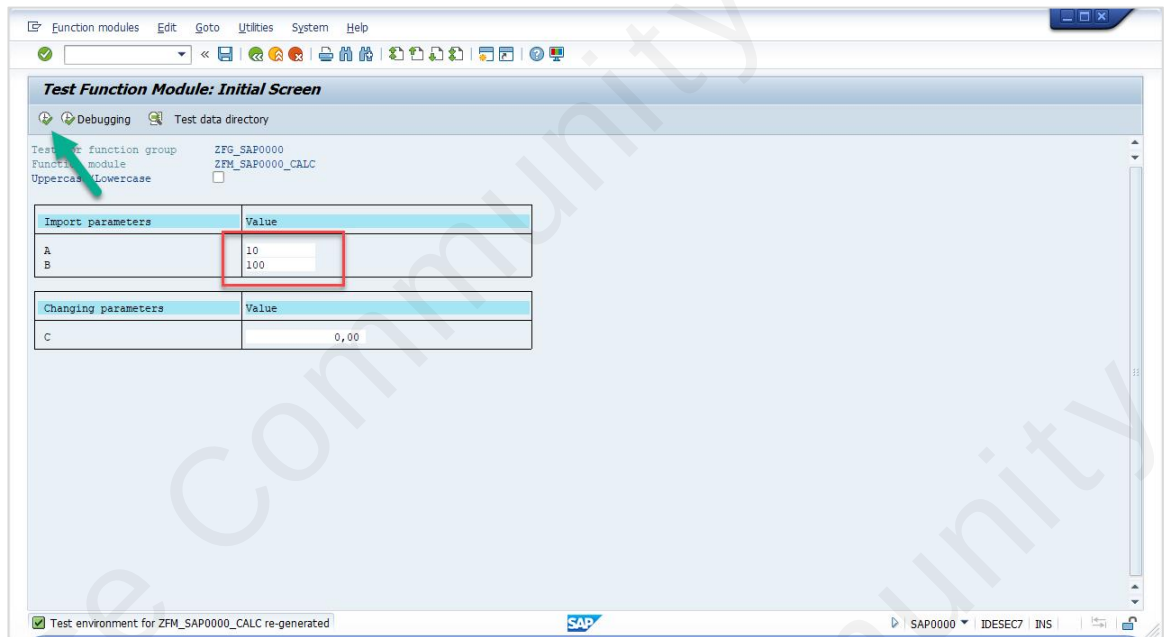
7. Create new source code and then click "Activate" button



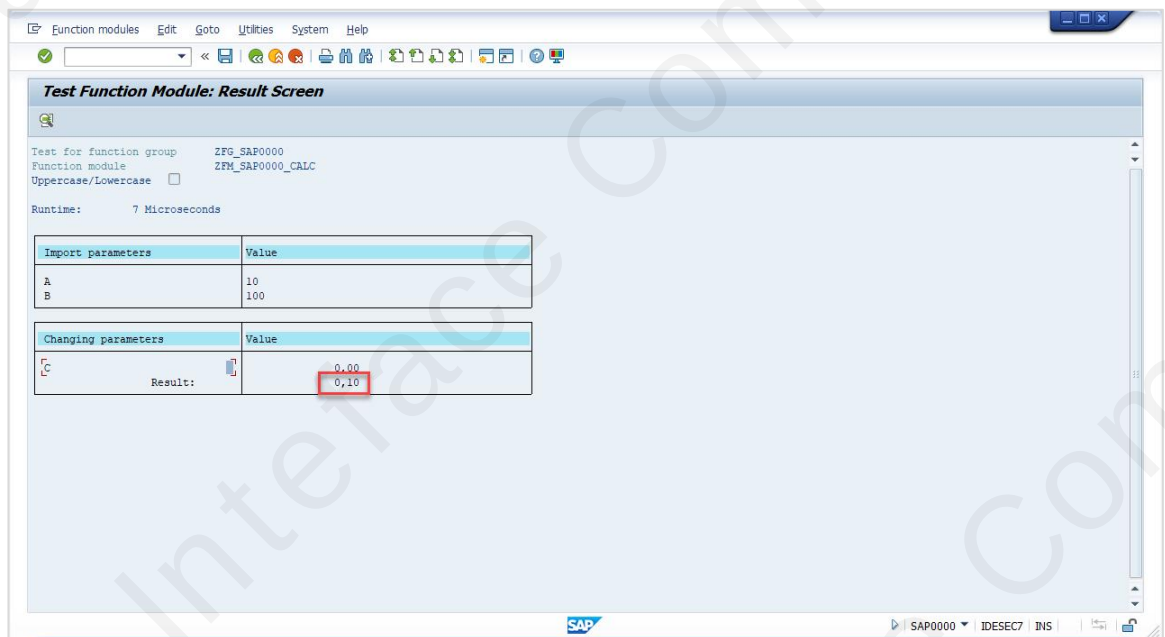
8. Click "Test/Execute" button



9. Enter value A = 10 and B = 100 and click "Execute" button



10. Result



11. Done

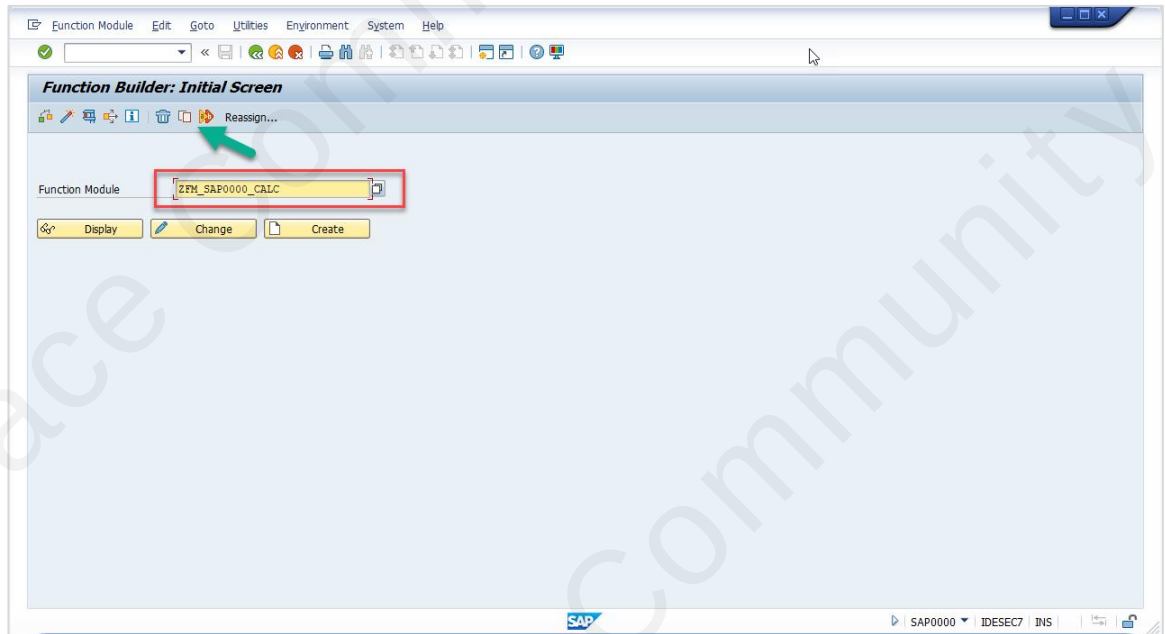
F. Create a Function Module Part 2

Function Module : ZFM_SAP0000_CALC2

Source : Example Function Part 2

Step to create a package:

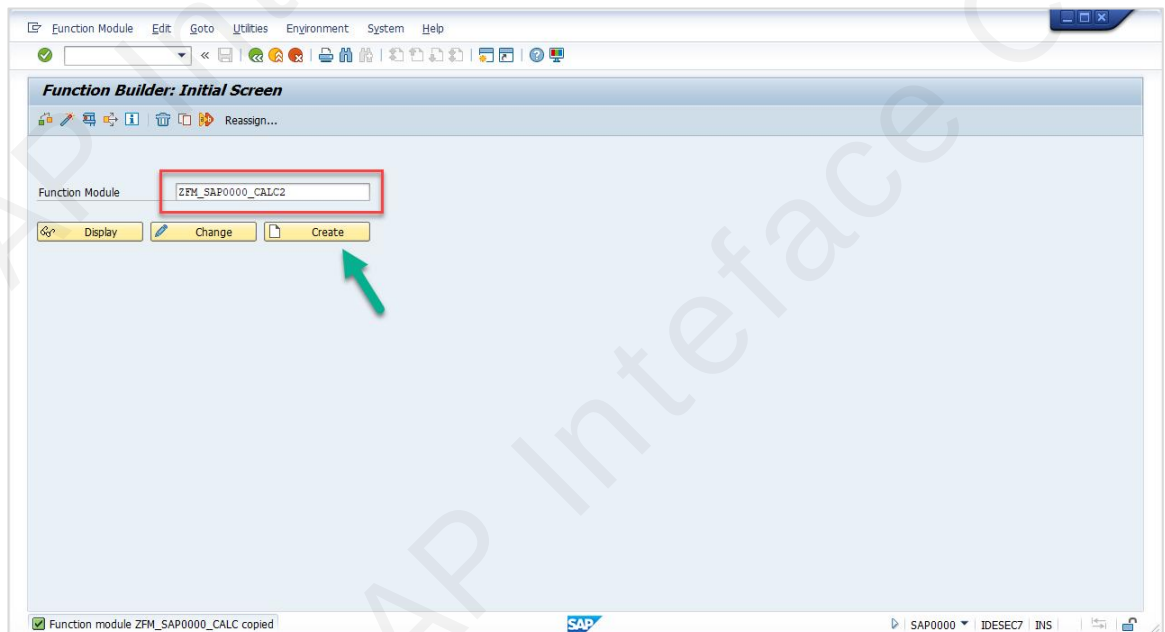
1. Open TCode SE37
2. Enter "ZFM_SAP0000_CALC" value and click "Copy" button



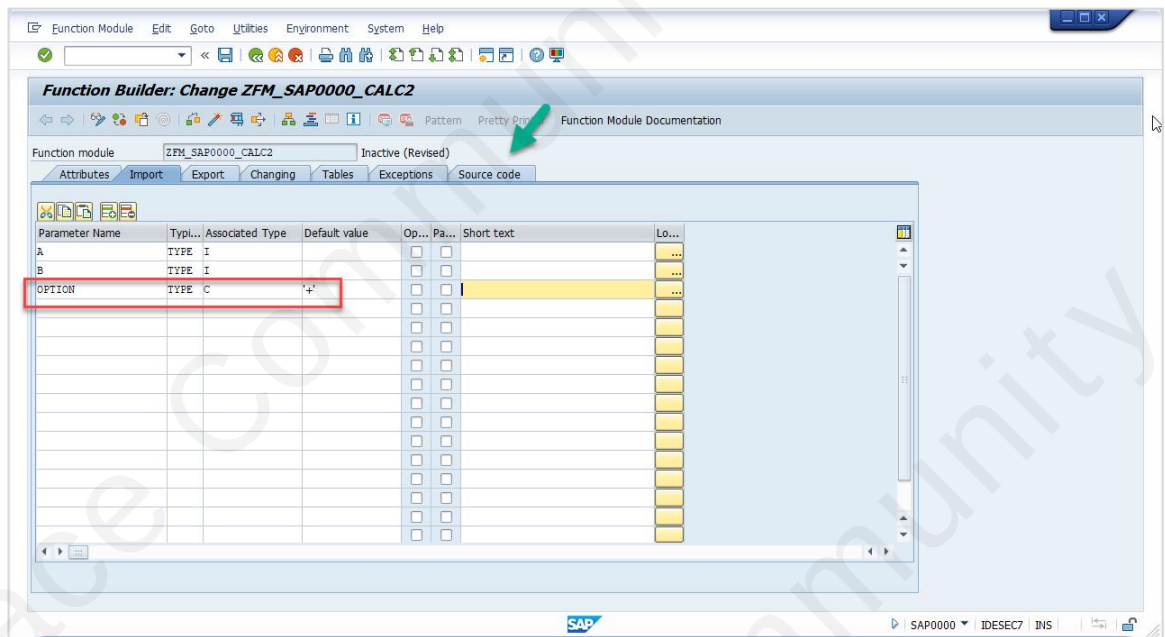
3. Enter "ZFM_SAP0000_CALC2" value as new Function Module Name and click "Save" button
4. Click "Continue"



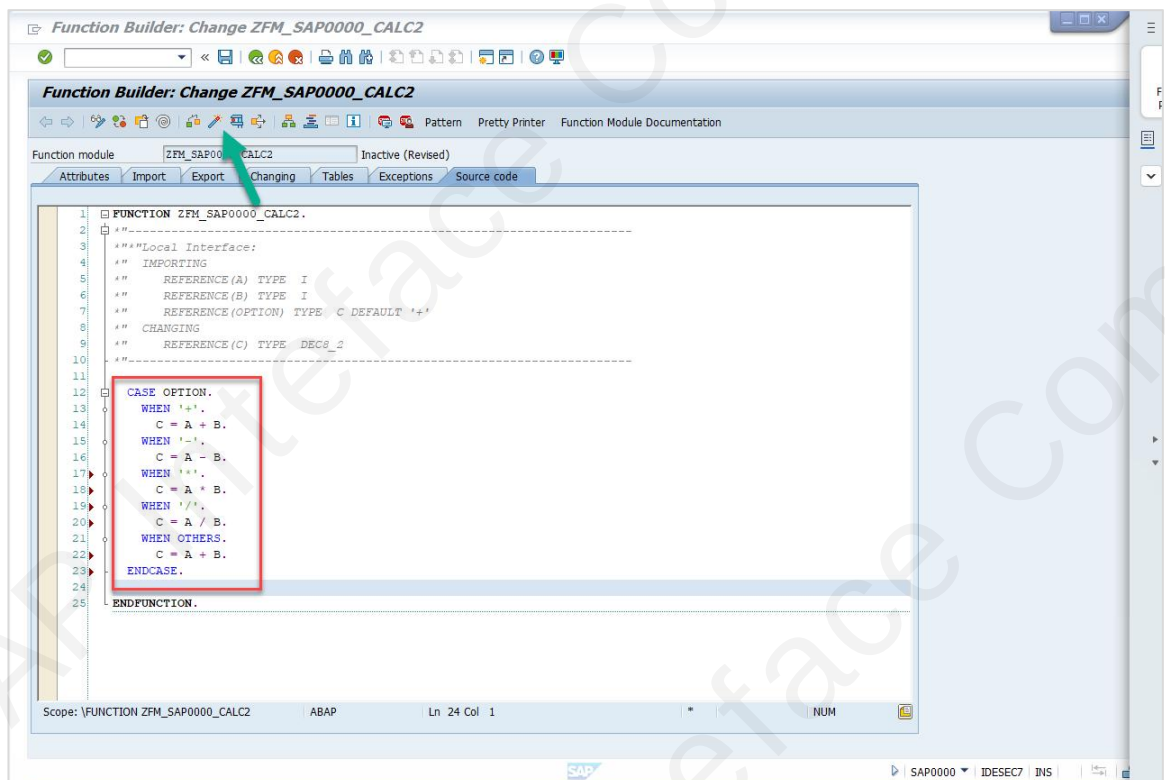
5. Change "ZFM_SAP0000_CALC" to "ZFM_SAP0000_CALC2" and click "Change"



6. Create "OPTION" as new Import Parameter and then click "Source Code" tab



7. Change source code like this bellow and click "Activate"



8. Done

G. Create a Report 101 - Output List "Basic"

Report Name : ZSAP0000_101
Source : Example Report 101

Step to create a package:

1. Open TCode SE38
2. Enter Program Name "ZSAP0000_015" and Copy to "ZSAP0000_101"
3. Modified Report Declaration

Before

```
REPORT ZSAP0000 101.
```

After

```
REPORT ZSAP0000_101 NO STANDARD PAGE HEADING.
```

4. Modified "Selection Screen"

Before

```
SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.  
PARAMETERS P_MAXROW TYPE I. "Maximum Rows
```

After

```
SELECTION-SCREEN BEGIN OF BLOCK GROUPBOX1 WITH FRAME TITLE TEXT_100. "Declare Group Box 1  
SELECTION-SCREEN BEGIN OF LINE.  
SELECTION-SCREEN COMMENT 1(18) TEXT_101 FOR FIELD S_BRAND.  
SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.  
SELECTION-SCREEN END OF LINE.  
SELECTION-SCREEN END OF BLOCK GROUPBOX1. "End Group Box 1  
  
SELECTION-SCREEN BEGIN OF BLOCK GROUPBOX2 WITH FRAME TITLE TEXT_200. "Declare Group Box 2  
SELECTION-SCREEN BEGIN OF LINE.  
SELECTION-SCREEN COMMENT 1(18) TEXT_201 FOR FIELD P_MAXROW.  
PARAMETERS P_MAXROW TYPE I DEFAULT 1000. "Maximum Rows  
SELECTION-SCREEN END OF LINE.  
  
SELECTION-SCREEN BEGIN OF LINE.  
SELECTION-SCREEN COMMENT 1(18) TEXT_202 FOR FIELD P_MWIDTH.  
PARAMETERS P_MWIDTH TYPE N DEFAULT 75 LENGTH 4.  
SELECTION-SCREEN END OF LINE.  
  
SELECTION-SCREEN BEGIN OF LINE.  
SELECTION-SCREEN COMMENT 1(18) TEXT_203 FOR FIELD P_MLINE .  
PARAMETERS P_MLINE TYPE N DEFAULT 10 LENGTH 4.  
SELECTION-SCREEN END OF LINE.  
  
SELECTION-SCREEN BEGIN OF LINE.  
SELECTION-SCREEN COMMENT 1(18) TEXT_204 FOR FIELD P_DELM .  
PARAMETERS P_DELM TYPE C DEFAULT '|' . "Delimiter  
SELECTION-SCREEN END OF LINE.  
  
SELECTION-SCREEN END OF BLOCK GROUPBOX2. "End Group Box 2
```

5. Create new block code of "INITIALIZATION" under "Groupbox2"

```
INITIALIZATION.  
TEXT_100 = 'Selection Parameter'.  
TEXT_101 = 'Brand'.  
  
TEXT_200 = 'Page Setup'.  
TEXT_201 = 'Maximum Rows'.  
TEXT_202 = 'Max. Width'.  
TEXT_203 = 'Max. Line per Page'.  
TEXT_204 = 'Delimiter'.
```

6. Create new code under "START-OF-SELECTION"

Before

```
START-OF-SELECTION.  
PERFORM FM_COLLECT_DATA TABLES S_BRAND S_BRAND USING P_MAXROW .
```

After

```
START-OF-SELECTION.  
NEW-PAGE LINE-SIZE P_MWIDTH.  
NEW-PAGE LINE-COUNT P_MLINE.  
  
PERFORM FM_COLLECT_DATA TABLES S_BRAND GI_DATA USING P_MAXROW .
```

7. Modified subroutine of "FM_DISPLAY_DATA"

Before

```
FORM FM_DISPLAY_DATA TABLES FT_DATA STRUCTURE GI_DATA.  
DATA LV_PRICE TYPE C LENGTH 20. " 1 - Buat Variable  
  
"Display Data  
LOOP AT FT_DATA.  
WRITE FT_DATA-PRICE TO LV_PRICE CURRENCY FT_DATA-WAERS. " 2 - Buat Conversi  
  
WRITE : / FT_DATA-BRAND,  
FT_DATA-CAR_TYPE,  
FT_DATA-RELEASE_YEAR,  
LV_PRICE, "3 - Tampilkan variable baru  
FT_DATA-WAERS.  
  
ENDLOOP.  
  
ENDFORM. "fm_display_data
```

After

```
FORM FM_DISPLAY_DATA TABLES FT_DATA STRUCTURE GI_DATA USING FU_HEADER FU_DATA.
DATA : LV_TOTAL TYPE N LENGTH 10,
      LV_COUNT TYPE N LENGTH 10,
      LV_COUNT_PAGE TYPE N LENGTH 10,
      LV_INDEX TYPE N LENGTH 10.
LV_TOTAL = LINES( FT_DATA ).

DATA LV_H0(10) VALUE 'NO'.
DATA LV_H1(20) VALUE 'Brand'.
DATA LV_H2(10) VALUE 'Car Type'.
DATA LV_H3(4) VALUE 'Year'.
DATA LV_H4(13) VALUE 'Price'.
DATA LV_H5(5) VALUE 'CuKey'.

"Header
IF FU_HEADER = 'X'.
  ULINE AT (P_MWIDTH).
  FORMAT COLOR COL_HEADING INTENSIFIED.
  WRITE : / P_DELM NO-GAP,
          LV_H0 CENTERED, P_DELM NO-GAP,
          LV_H1 CENTERED, P_DELM NO-GAP,
          LV_H2 CENTERED, P_DELM NO-GAP,
          LV_H3 CENTERED, P_DELM NO-GAP,
          LV_H4 CENTERED, P_DELM NO-GAP,
          LV_H5 CENTERED, P_DELM NO-GAP.
  FORMAT COLOR OFF.
  ULINE AT / (P_MWIDTH).
ENDIF.
```

```
"Content
IF FU_DATA = 'X'.
  LV_COUNT_PAGE = P_MLINE - 3 - 1. " 3 = Total Line Header; 1 = Total Line Footer

  IF LV_COUNT_PAGE < 2.
    MESSAGE 'Minimal 2 Line of Content !!!' TYPE 'I'.
    EXIT.
  ENDIF.

  LOOP AT FT_DATA.
    ADD 1 TO LV_INDEX.
    LV_H0 = LV_INDEX.
    WRITE FT_DATA-BRAND TO LV_H1.
    WRITE FT_DATA-CAR_TYPE TO LV_H2.
    WRITE FT_DATA-RELEASE_YEAR TO LV_H3.
    WRITE FT_DATA-PRICE TO LV_H4 CURRENCY FT_DATA-WAERS. " 2 - Buat Conversi
    WRITE FT_DATA-WAERS TO LV_H5.
    WRITE : / P_DELM NO-GAP,
            LV_H0, P_DELM NO-GAP,
            LV_H1, P_DELM NO-GAP,
            LV_H2, P_DELM NO-GAP,
            LV_H3, P_DELM NO-GAP,
            LV_H4, P_DELM NO-GAP,
            LV_H5, P_DELM NO-GAP.

    LV_COUNT = SY-TABIX MOD LV_COUNT_PAGE.
    IF LV_COUNT = 0.
      ULINE AT / (P_MWIDTH).
      NEW-PAGE.
    ENDIF.
  ENDLOOP.

  IF LV_TOTAL = LV_INDEX.
    ULINE AT / (P_MWIDTH).
  ENDIF.
ENDIF.
ENDFORM. "fm_display_data
```

8. Create new block code of "TOP-OF-PAGE"

```
TOP-OF-PAGE.  
  PERFORM FM_DISPLAY_DATA TABLES GI_DATA USING 'X' ''.
```

9. Modified block code of "END-OF-SELECTION"

Before

```
END-OF-SELECTION.  
  PERFORM FM_DISPLAY_DATA TABLES GI_DATA.
```

After

```
END-OF-SELECTION.  
  PERFORM FM_DISPLAY_DATA TABLES GI_DATA USING '' 'X'.
```

10. Click "Activate"

11. Done

H. Create a Report 102 - Output List "Statistic"

Report Name : ZSAP0000_102
Source : Example Report 102

Step to create a package:

1. Open TCode SE38
2. Enter Program Name "ZSAP0000_101" and Copy to "ZSAP0000_102"
3. Modified subroutine of "FM_DISPLAY_DATA" part 1

Before

```
DATA : LV_TOTAL TYPE N LENGTH 10,  
        LV_COUNT TYPE N LENGTH 10,  
        LV_COUNT_PAGE TYPE N LENGTH 10,  
        LV_INDEX TYPE N LENGTH 10.  
LV_TOTAL = LINES( FT_DATA ).
```

After

```
DATA : LV_TOTAL TYPE N LENGTH 10,  
        LV_COUNT TYPE N LENGTH 10,  
        LV_COUNT_PAGE TYPE N LENGTH 10,  
        LV_INDEX TYPE N LENGTH 10,  
        LV_TOTAL_PRICE LIKE FT_DATA-PRICE,  
        LV_HTOTAL_PRICE(13),  
        LV_LAST_BRAND LIKE FT_DATA-BRAND.  
  
LV_TOTAL = LINES( FT_DATA ).
```

4. Modified subroutine of "FM_DISPLAY_DATA" part 2

Before

```
"Content  
IF FU_DATA = 'X'.  
    LV_COUNT_PAGE = P_MLINE - 3 - 1. " 3 = Total Line Header; 1 = Total Line Footer  
  
    IF LV_COUNT_PAGE < 2.  
        MESSAGE 'Minimal 2 Line of Content !!!' TYPE 'I'.  
        EXIT.  
    ENDIF.
```

After

```
"Content  
IF FU_DATA = 'X'.  
    SORT FT_DATA BY BRAND.  
  
    READ TABLE FT_DATA INDEX 1.  
    IF SY-SUBRC = 0.  
        LV_LAST_BRAND = FT_DATA-BRAND.  
    ENDIF.  
  
    LV_COUNT_PAGE = P_MLINE - 3 - 3. " 3 = Total Line Header ; 2 = Total Line Footer  
  
    IF LV_COUNT_PAGE < 2.  
        MESSAGE 'Minimal 2 Line of Content !!!' TYPE 'I'.  
        EXIT.  
    ENDIF.
```

5. Modified subroutine of "FM_DISPLAY_DATA" part 3

Before

```

LOOP AT FT_DATA.
  ADD 1 TO LV_INDEX.
  LV_H0 = LV_INDEX.
  WRITE FT_DATA-BRAND TO LV_H1.
  WRITE FT_DATA-CAR_TYPE TO LV_H2.
  WRITE FT_DATA-RELEASE_YEAR TO LV_H3.
  WRITE FT_DATA-PRICE TO LV_H4 CURRENCY FT_DATA-WAERS. " 2 - Buat Conversi
  WRITE FT_DATA-WAERS TO LV_H5.
  WRITE : / P_DELM NO-GAP,
          LV_H0, P_DELM NO-GAP,
          LV_H1, P_DELM NO-GAP,
          LV_H2, P_DELM NO-GAP,
          LV_H3, P_DELM NO-GAP,
          LV_H4, P_DELM NO-GAP,
          LV_H5, P_DELM NO-GAP.

  LV_COUNT = SY-TABIX MOD LV_COUNT_PAGE.
  IF LV_COUNT = 0.
    ULINE AT / (P_MWIDTH).
    NEW-PAGE.
  ENDIF.
ENDLOOP.

```

After

```

LOOP AT FT_DATA.
  IF LV_LAST_BRAND NE FT_DATA-BRAND.
    ULINE AT / (P_MWIDTH).
    WRITE LV_TOTAL_PRICE TO LV_HTOTAL_PRICE CURRENCY FT_DATA-WAERS.
    WRITE : / P_DELM NO-GAP,
            2(50) 'Total Price :' RIGHT-JUSTIFIED, P_DELM NO-GAP,
            LV_HTOTAL_PRICE COLOR COL_TOTAL, P_DELM NO-GAP,
            ' ', P_DELM NO-GAP.
    ULINE AT / (P_MWIDTH).
    NEW-PAGE.
    LV_TOTAL_PRICE = 0.
  ENDIF.

  ADD 1 TO LV_INDEX.
  LV_H0 = LV_INDEX.
  WRITE FT_DATA-BRAND TO LV_H1.
  WRITE FT_DATA-CAR_TYPE TO LV_H2.
  WRITE FT_DATA-RELEASE_YEAR TO LV_H3.
  WRITE FT_DATA-PRICE TO LV_H4 CURRENCY FT_DATA-WAERS. " 2 - Buat Conversi
  WRITE FT_DATA-WAERS TO LV_H5.
  WRITE : / P_DELM NO-GAP,
          LV_H0, P_DELM NO-GAP,
          LV_H1, P_DELM NO-GAP,
          LV_H2, P_DELM NO-GAP,
          LV_H3, P_DELM NO-GAP,
          LV_H4, P_DELM NO-GAP,
          LV_H5, P_DELM NO-GAP.

  ADD FT_DATA-PRICE TO LV_TOTAL_PRICE.

  LV_COUNT = SY-TABIX MOD LV_COUNT_PAGE.
  IF LV_COUNT = 0 AND LV_LAST_BRAND = FT_DATA-BRAND.
    ULINE AT / (P_MWIDTH).
    NEW-PAGE.
  ENDIF.
  LV_LAST_BRAND = FT_DATA-BRAND.
ENDLOOP.

```

6. Modified subroutine of "FM_DISPLAY_DATA" part 4

Before

```
IF LV_TOTAL = LV_INDEX.  
  ULINE AT / (P_MWIDTH).  
ENDIF.
```

After

```
IF LV_TOTAL = LV_INDEX.  
  ULINE AT / (P_MWIDTH).  
  WRITE LV_TOTAL_PRICE TO LV_HTOTAL_PRICE CURRENCY FT_DATA-WAERS.  
  WRITE : / P_DELM NO-GAP,  
          2(50) 'Total Price :' RIGHT-JUSTIFIED, P_DELM NO-GAP,  
          LV_HTOTAL_PRICE COLOR COL_TOTAL, P_DELM NO-GAP,  
          ' ', P_DELM NO-GAP.  
  ULINE AT / (P_MWIDTH).  
ENDIF.
```

7. Click "Activate"

8. Done

I. Create a Report 102A

Report Name : ZSAP0000_102A

Source : Copy and Modified using the "ZSAP0000_CAR_001" Table

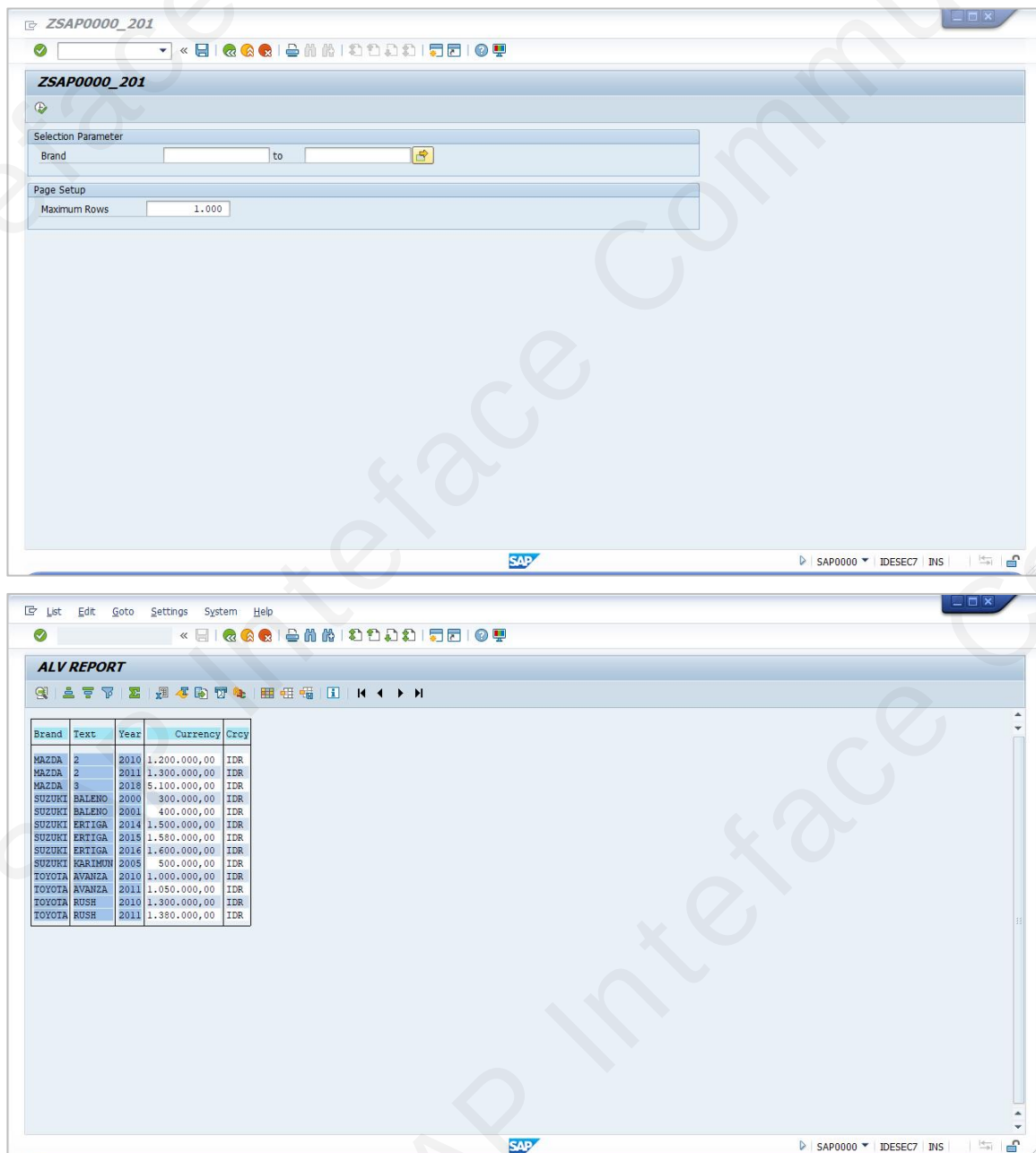
Day - 4

TCode	SE37, SE38, ARAD
Target	Understanding Function in ABAP and SAP Interface/Integration

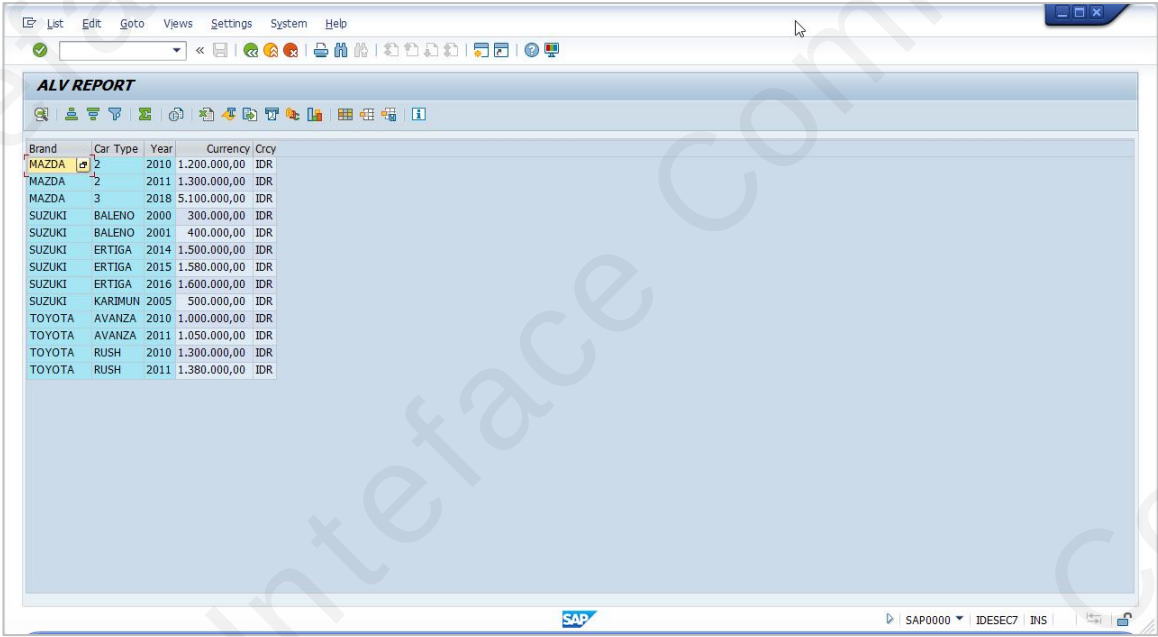
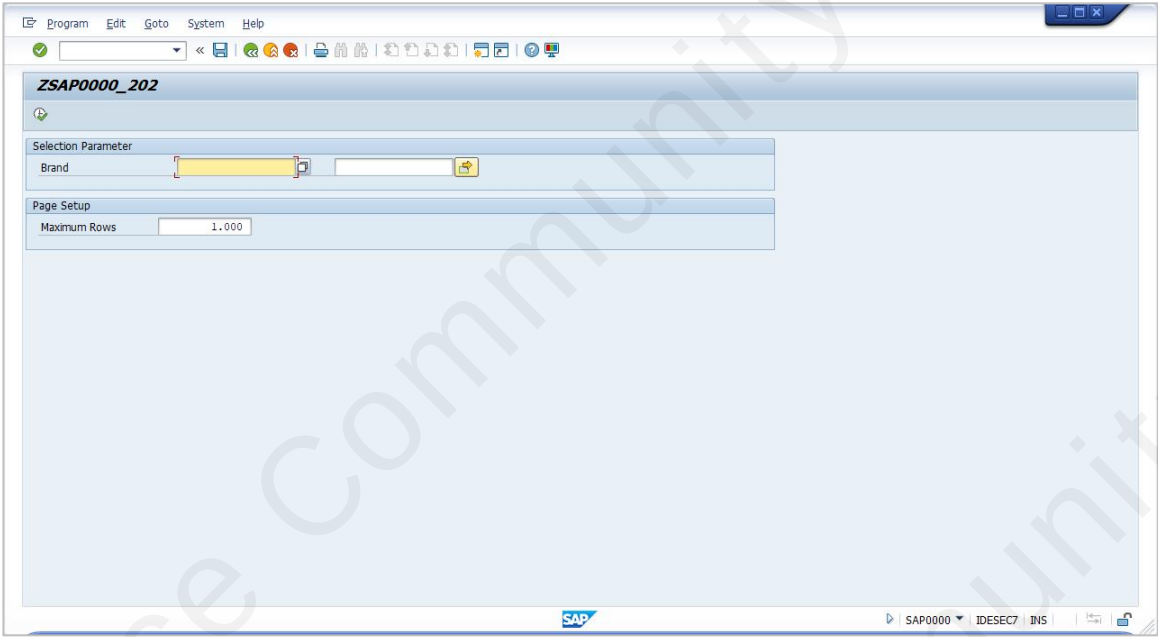
1. ALV - ABAP List Viewer

The ABAP List Viewer (ALV) (or SAP List Viewer) is a powerful set of application programming interfaces (APIs) used in SAP to display, sort, filter, and format data in tabular or hierarchical lists. It significantly reduces development time compared to traditional WRITE statements, offering interactive features like custom layouts, totals, and Excel export, making it essential for building flexible custom reports.

A. ALV List / Classic



B. ALV Grid

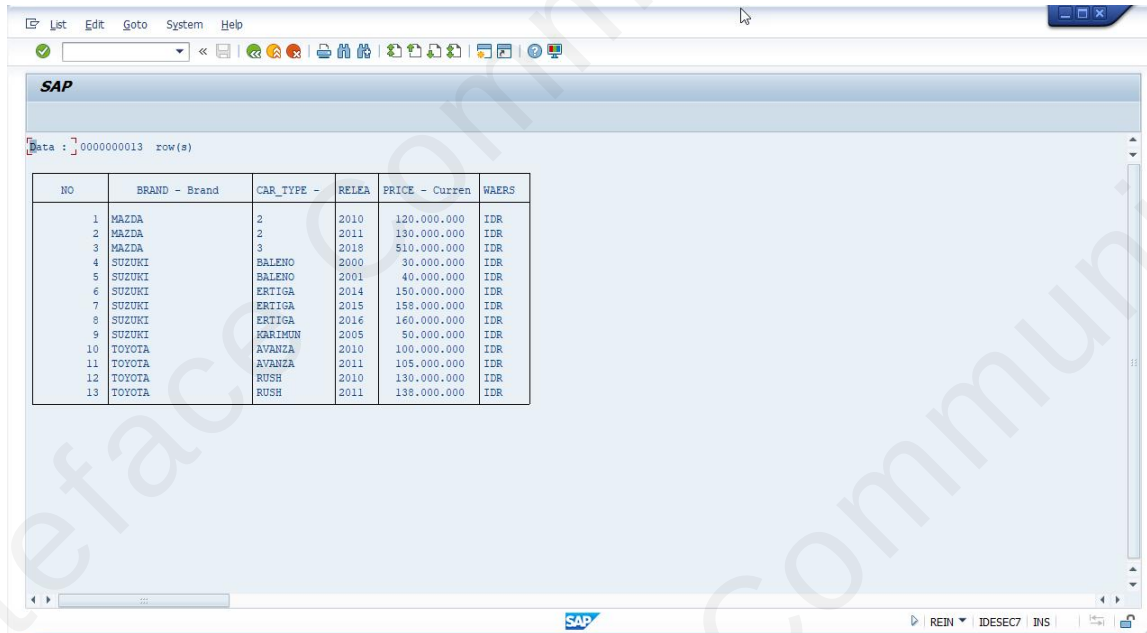


2. ARAD - ABAP Rapid Application Development

The ARAD application can help speed up a report development. With just a few clicks, the report is automatically generated.

A. Output List Display

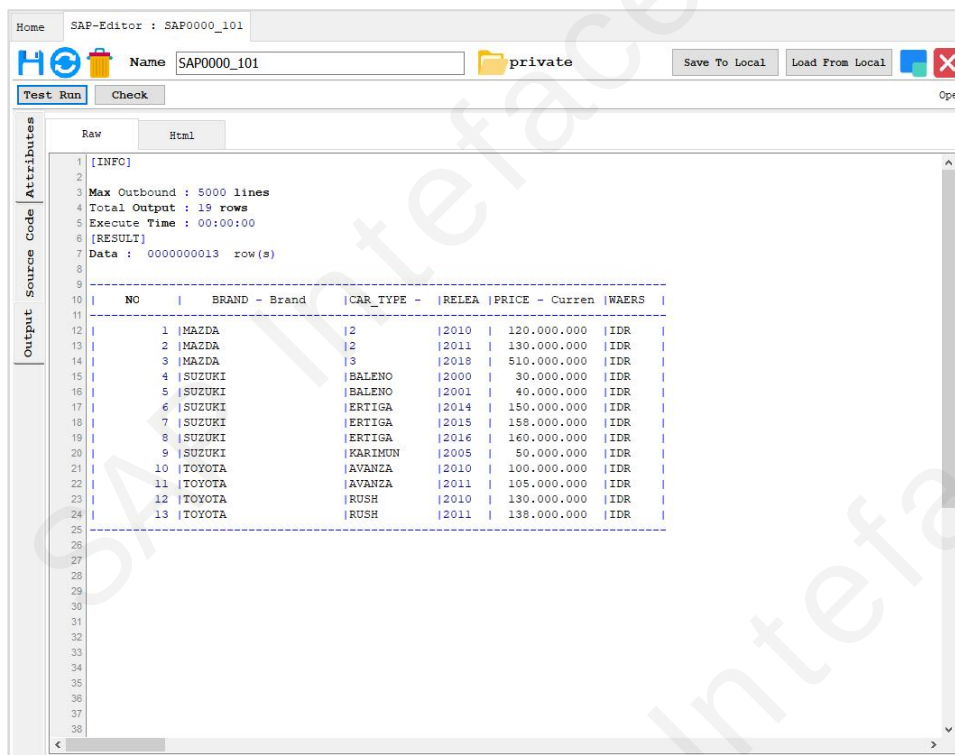
SAP Output



The screenshot shows the SAP Output List Display window. The title bar includes 'List', 'Edit', 'Goto', 'System', and 'Help'. The main area displays a table with the following data:

NO	BRAND - Brand	CAR_TYPE -	RELEA	PRICE - Curren	WAERS
1	MAZDA	2	2010	120.000.000	IDR
2	MAZDA	2	2011	130.000.000	IDR
3	MAZDA	3	2018	510.000.000	IDR
4	SUZUKI	BALENO	2000	30.000.000	IDR
5	SUZUKI	BALENO	2001	40.000.000	IDR
6	SUZUKI	ERTIGA	2014	150.000.000	IDR
7	SUZUKI	ERTIGA	2015	158.000.000	IDR
8	SUZUKI	ERTIGA	2016	160.000.000	IDR
9	SUZUKI	KARIMUN	2005	50.000.000	IDR
10	TOYOTA	AVANZA	2010	100.000.000	IDR
11	TOYOTA	AVANZA	2011	105.000.000	IDR
12	TOYOTA	RUSH	2010	130.000.000	IDR
13	TOYOTA	RUSH	2011	138.000.000	IDR

ARAD Output



The screenshot shows the ARAD Output window. The title bar includes 'Home', 'SAP-Editor : SAP0000_101', and buttons for 'Name', 'private', 'Save To Local', 'Load From Local', and 'Open'. The main area displays a table with the following data:

NO	BRAND - Brand	CAR_TYPE -	RELEA	PRICE - Curren	WAERS
1	MAZDA	2	2010	120.000.000	IDR
2	MAZDA	2	2011	130.000.000	IDR
3	MAZDA	3	2018	510.000.000	IDR
4	SUZUKI	BALENO	2000	30.000.000	IDR
5	SUZUKI	BALENO	2001	40.000.000	IDR
6	SUZUKI	ERTIGA	2014	150.000.000	IDR
7	SUZUKI	ERTIGA	2015	158.000.000	IDR
8	SUZUKI	ERTIGA	2016	160.000.000	IDR
9	SUZUKI	KARIMUN	2005	50.000.000	IDR
10	TOYOTA	AVANZA	2010	100.000.000	IDR
11	TOYOTA	AVANZA	2011	105.000.000	IDR
12	TOYOTA	RUSH	2010	130.000.000	IDR
13	TOYOTA	RUSH	2011	138.000.000	IDR

B. ABAP List Viewer

SAP Output

ALV REPORT

Brand	Text	Year	Currency	Crcy
MAZDA	2	2010	120.000.000	IDR
MAZDA	2	2011	130.000.000	IDR
MAZDA	3	2018	510.000.000	IDR
SUZUKI	BALENO	2000	30.000.000	IDR
SUZUKI	BALENO	2001	40.000.000	IDR
SUZUKI	ERTIGA	2014	150.000.000	IDR
SUZUKI	ERTIGA	2015	158.000.000	IDR
SUZUKI	ERTIGA	2016	160.000.000	IDR
SUZUKI	KARIMUN	2005	50.000.000	IDR
TOYOTA	AVANZA	2010	100.000.000	IDR
TOYOTA	AVANZA	2011	105.000.000	IDR
TOYOTA	RUSH	2010	130.000.000	IDR
TOYOTA	RUSH	2011	138.000.000	IDR

ARAD Output

SAP-Editor : SAP0000_201

Name: SAP0000_201

private

Save To Local Load From Local

Test Run Check

Open

Raw Html

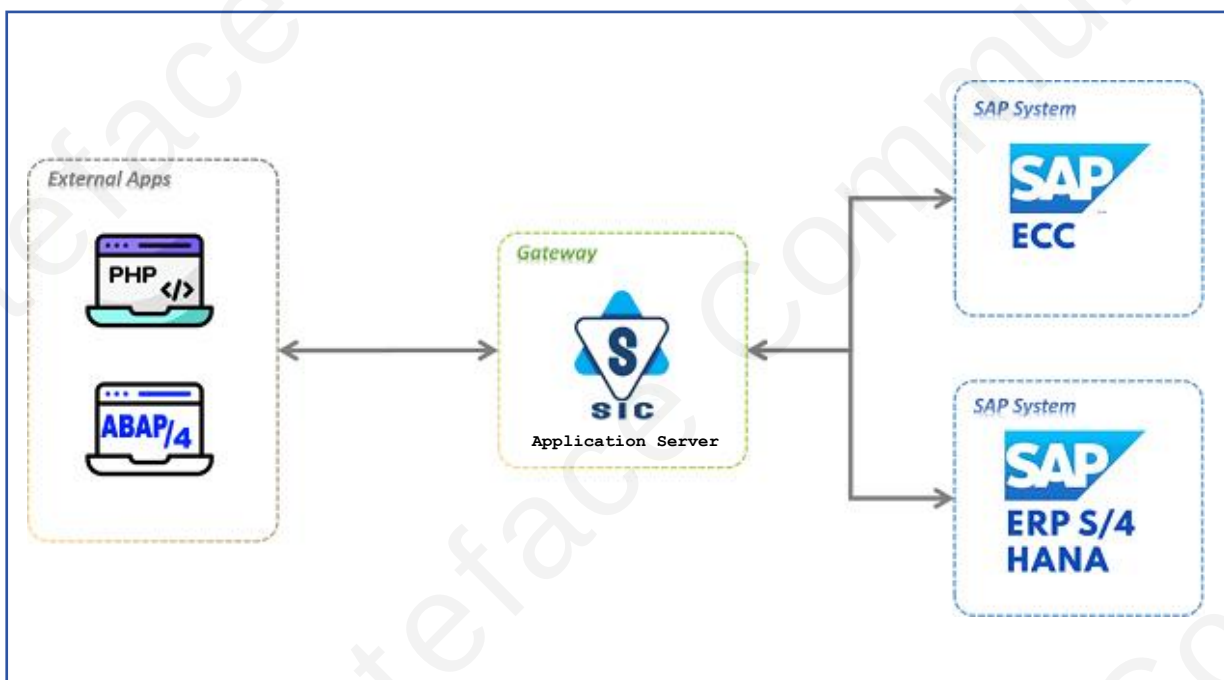
Output Source Code Attributes

1 [INFO]
2
3 Max Outbound : 5000 lines
4 Total Output : 17 rows
5 Execute Time : 00:00:00
6 [RESULT]
7
8 Brand | Text | Year | Currency | Crcy |
9
10 MAZDA | 2 | 2010 | 120.000.000 | IDR |
11 MAZDA | 2 | 2011 | 130.000.000 | IDR |
12 MAZDA | 3 | 2018 | 510.000.000 | IDR |
13 SUZUKI | BALENO | 2000 | 30.000.000 | IDR |
14 SUZUKI | BALENO | 2001 | 40.000.000 | IDR |
15 SUZUKI | ERTIGA | 2014 | 150.000.000 | IDR |
16 SUZUKI | ERTIGA | 2015 | 158.000.000 | IDR |
17 SUZUKI | ERTIGA | 2016 | 160.000.000 | IDR |
18 SUZUKI | KARIMUN | 2005 | 50.000.000 | IDR |
19 TOYOTA | AVANZA | 2010 | 100.000.000 | IDR |
20 TOYOTA | AVANZA | 2011 | 105.000.000 | IDR |
21 TOYOTA | RUSH | 2010 | 130.000.000 | IDR |
22 TOYOTA | RUSH | 2011 | 138.000.000 | IDR |
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38

3. SAP Interface/Integration with PHP

SAP stopped developing PHP-RFC library around 2009-2010. PHP-RFC library is used to call Function modules in SAP. After that, PHP developers used web APIs to integrate with SAP. However, there were numerous technical difficulties and complexities involved in development.

Now, all the complexities in development can be solved by using the ARAD application. ARAD application has SAP core level integration capabilities instead of web API. This technology is also known as "Clean Core". ABAP source code is outside the SAP system and can be executed on any SAP server. One of the features in the arad application that can be used to create integrations is SWF (SAP Web Framework).



"Integration Flow"

A. SWF Basic Template ([download](#))

SAP - PHP Integration
version 0.4

Login to SAP System

Client

Username

Password

Free Gateway Access to SAP System

Login

[Free Source Code](#)
Created by Albertus Reinandang @2026

SAP Interface Community

[Logout](#) Welcome : DEMO

Text Report

- [Template Text](#)

HTML Report

- [Template Html](#)

Graph

- [Template Graph](#)

Server Information : 45.127.134.174 EC7(00)800
Created by Albertus Reinandang @2025

HTML Report - Material Description

[< Back](#) [Execute](#) [View ABAP Code](#)

Maximum row :
Material No :
Description :

Result

TOTAL DATA : 0000000050 row(s)

No	Material Number	Material Description (Short Text)
1	23 view	Test material
2	38 view	Classification test
3	43 view	english Checking
4	58 view	Ventilation, complete
5	59 view	Filter
6	68 view	a portable 1 ton crane
7	78 view	Component Full Repair Service
8	88 view	AS-100 T-shirt

B. SWF Advanced Template ([download](#))

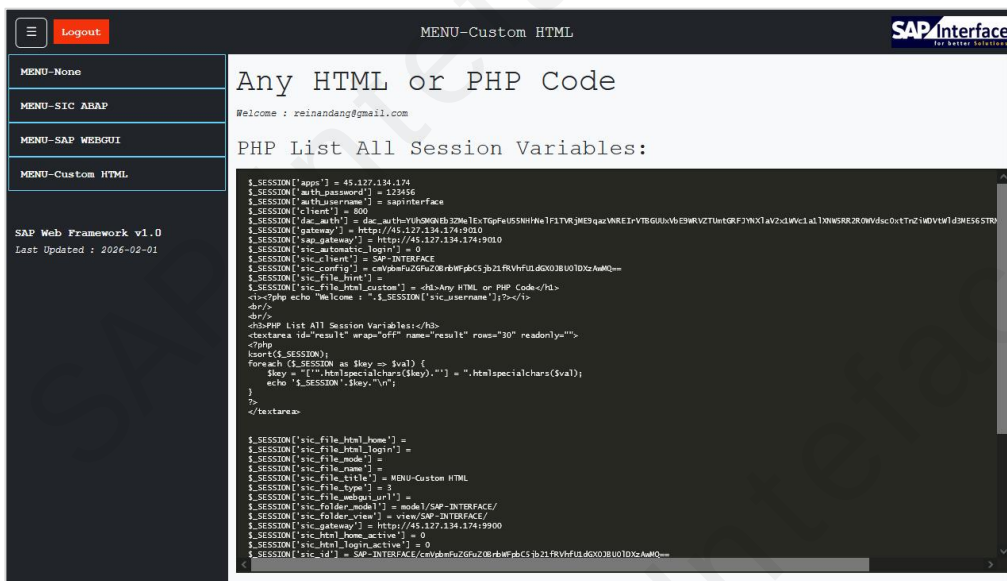
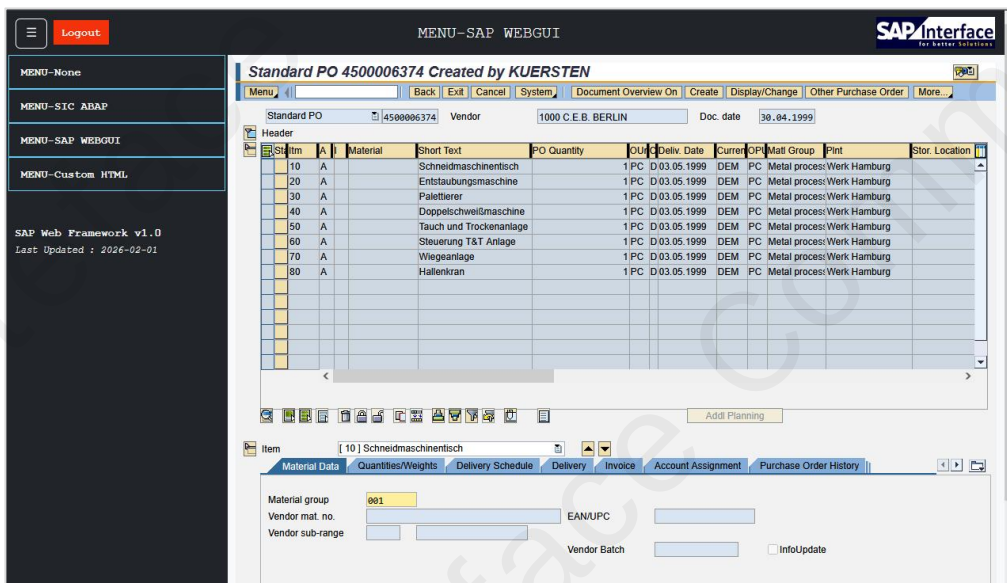
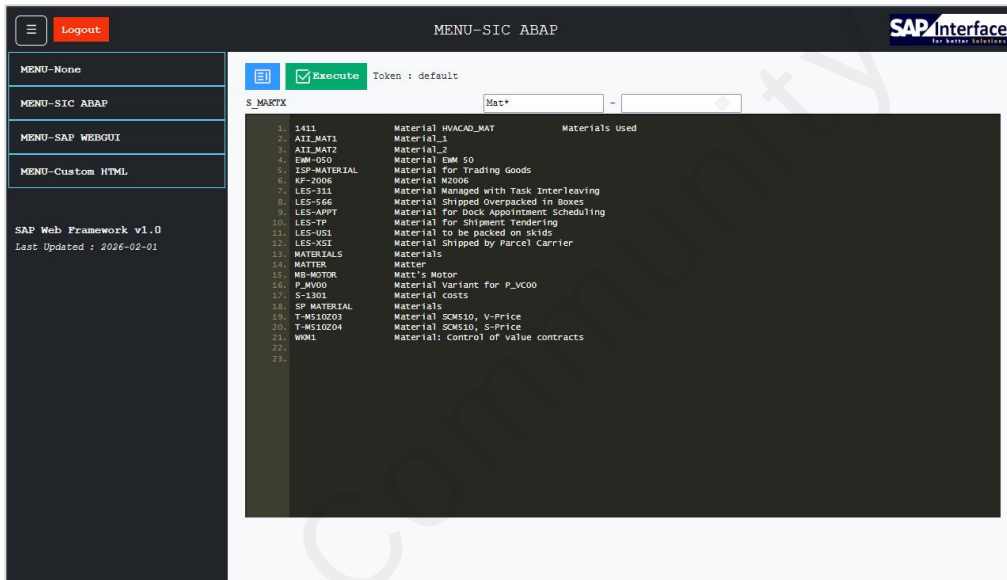
Not Secure http://45.127.134.174:9090/swf/v1.0/

[Logout](#) EX - Menu Introduction **SAP Interface**
for better Solutions

Home Page
Welcome : reinandang@gmail.com

MENU-None
MENU-SIC ABAP
MENU-SAP WEBGUI
MENU-Custom HTML

SAP Web Framework v1.0
Last Updated : 2026-02-01



Real Project (Live) visit : www.sap-interface.com

Practice Day - 4

A. Create a Report 201 - ALV List

Report Name : ZSAP0000_201

Source : -

Step to create a package:

1. Open TCode SE38
2. Enter Program Name "ZSAP0000_015" and Copy to "ZSAP0000_201"
3. Create new "Global Variable" for ALV

Before

```
DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.
```

After

```
DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.
```

```
"ALV Variable
```

```
DATA : GV_I_DEFAULT, GV_I_SAVE.
```

4. Create new include Program under "ALV Variable"

Before

```
DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.
```

```
"ALV Variable
```

```
DATA : GV_I_DEFAULT, GV_I_SAVE.
```

After

```
DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.
```

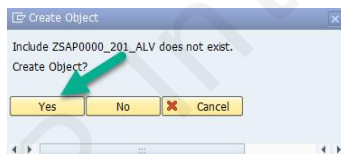
```
"ALV Variable
```

```
DATA : GV_I_DEFAULT, GV_I_SAVE.
```

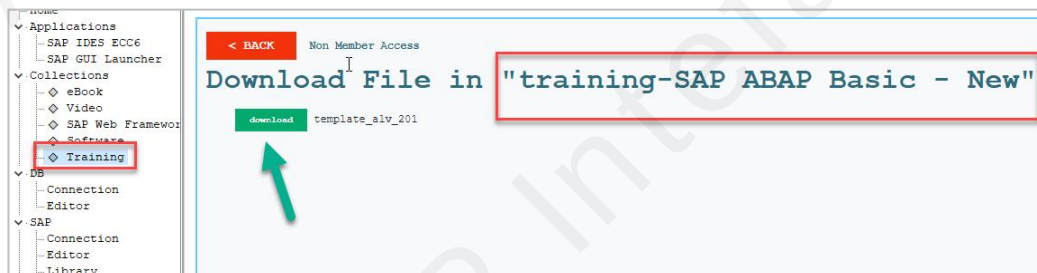
```
INCLUDE ZSAP0000_201_ALV.
```

5. Double Click on "ZSAP0000_201_ALV"

6. Click "Yes" button



7. Run ARAD Application, open "Collection->Training->SAP ABAP Basic - New" and download "template_alv_201"



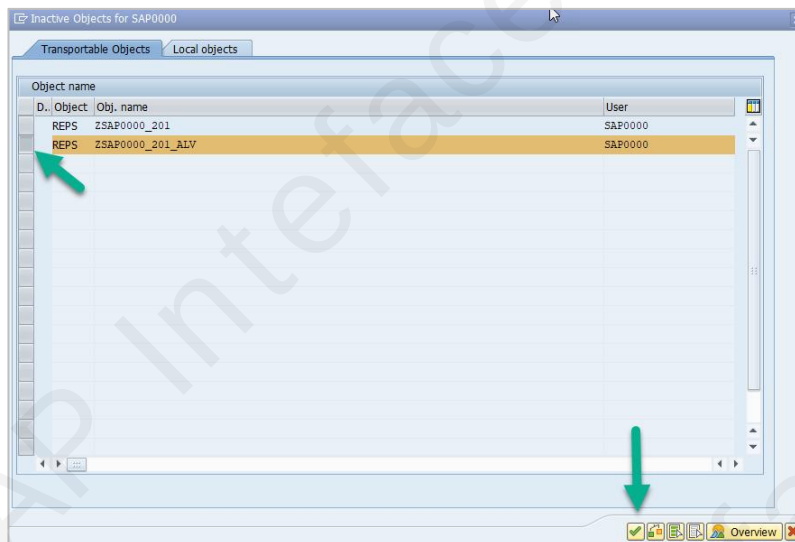
>> [Download ARAD](#) <<

8. Select all and copy "template_alv_01" content and paste it to SE38

[illegible]

9. Click "Activate"

10. Select "ZSAP0000_201_ALV" and click "continue"



11. Click "Back" or press "ESC"

12. Modified "Selection Screen"

Before

```
SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.  
PARAMETERS P_MAXROW TYPE I. "Maximum Rows
```

After

```
SELECTION-SCREEN BEGIN OF BLOCK GROUPBOX1 WITH FRAME TITLE TEXT_100. "Declare Group Box 1  
SELECTION-SCREEN BEGIN OF LINE.  
SELECTION-SCREEN COMMENT 1(18) TEXT_101 FOR FIELD S_BRAND.  
SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND.  
SELECTION-SCREEN END OF LINE.  
SELECTION-SCREEN END OF BLOCK GROUPBOX1. "End Group Box 1  
  
SELECTION-SCREEN BEGIN OF BLOCK GROUPBOX2 WITH FRAME TITLE TEXT_200. "Declare Group Box 2  
SELECTION-SCREEN BEGIN OF LINE.  
SELECTION-SCREEN COMMENT 1(18) TEXT_201 FOR FIELD P_MAXROW.  
PARAMETERS P_MAXROW TYPE I DEFAULT 1000. "Maximum Rows  
SELECTION-SCREEN END OF LINE.  
SELECTION-SCREEN END OF BLOCK GROUPBOX2. "End Group Box 2
```

12. Create new block code of "INITIALIZATION" under "Groupbox2"

```
INITIALIZATION.  
TEXT_100 = 'Selection Parameter'.  
TEXT_101 = 'Brand'.  
  
TEXT_200 = 'Page Setup'.  
TEXT_201 = 'Maximum Rows'.
```

13. Modified subroutine of "FM_DISPLAY_DATA"

Before

```
DATA LV_PRICE TYPE C LENGTH 20. " 1 - Buat Variable  
  
"Display Data  
LOOP AT FT_DATA.  
WRITE FT_DATA-PRICE TO LV_PRICE CURRENCY FT_DATA-WAERS. " 2 - Buat Conversi  
  
WRITE : / FT_DATA-BRAND,  
FT_DATA-CAR_TYPE,  
FT_DATA-RELEASE_YEAR,  
LV_PRICE, "3 - Tampilkan variable baru  
FT_DATA-WAERS.  
ENDLOOP.
```

After

```
PERFORM FM_ALV_RESET_DATA.  
PERFORM FM_ALV_SET_LAYOUT USING 'ALV REPORT'.  
PERFORM FM_ALV_SET_PRINT.  
PERFORM FM_ALV_SET_COLUMN.  
GV_I_DEFAULT = 'X'.  
GV_I_SAVE = 'A'.  
PERFORM FM_ALV_SHOW TABLES FT_DATA.
```

14. Click "Activate"

15. Done

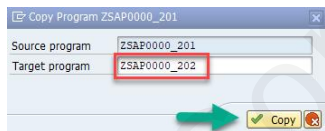
B. Create a Report 202 - ALV Grid

Report Name : ZSAP0000_202

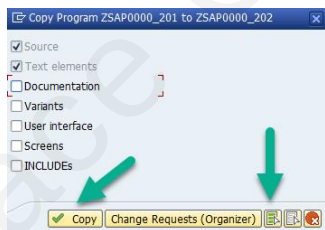
Source : -

Step to create a package:

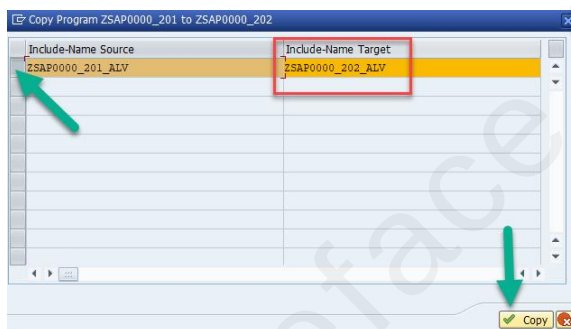
1. Open TCode SE38
2. Enter Program Name "ZSAP0000_201" and click "Copy" button
3. Enter Target program "ZSAP0000_202" and click "Copy" button



4. Click "Select All" and click "Copy" button



5. Enter include-name target "ZSAP0000_202_ALV", select row and then click "Copy" button



6. Save to package "ZSAP0000" and enter "Request Number"
7. Click "Change" button
8. Double click include program "ZSAP0000_202_ALV"
9. Modified code of "Call Function"

Before

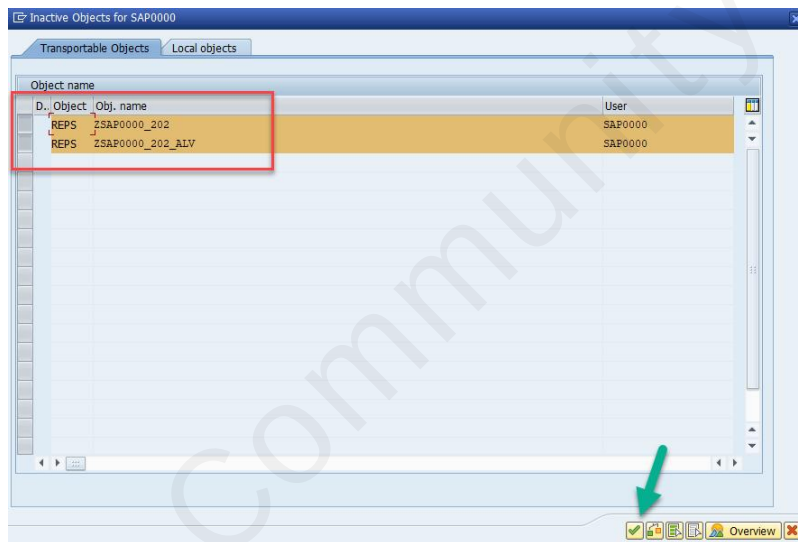
```
CALL FUNCTION 'REUSE_ALV_LIST_DISPLAY'
```

After

```
CALL FUNCTION 'REUSE_ALV_GRID_DISPLAY'
```

10. Click "Activate"

11. Select Rows and click "Continue"



12. Done

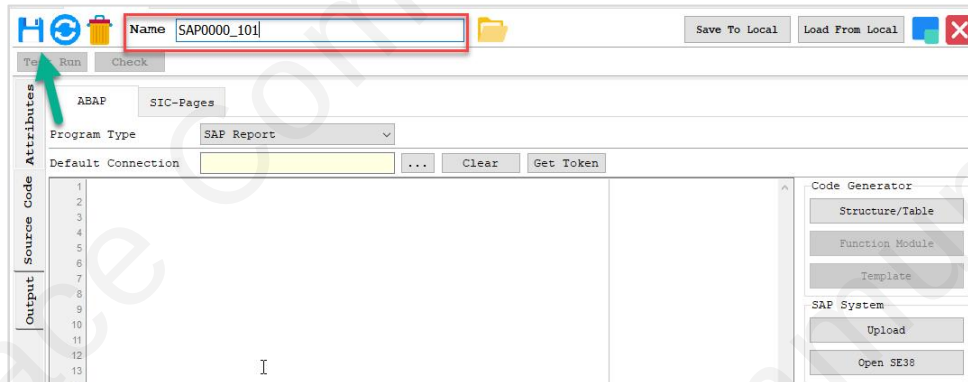
C. Create a Report ARAD - SAP000_101

Report Name : SAP0000_101

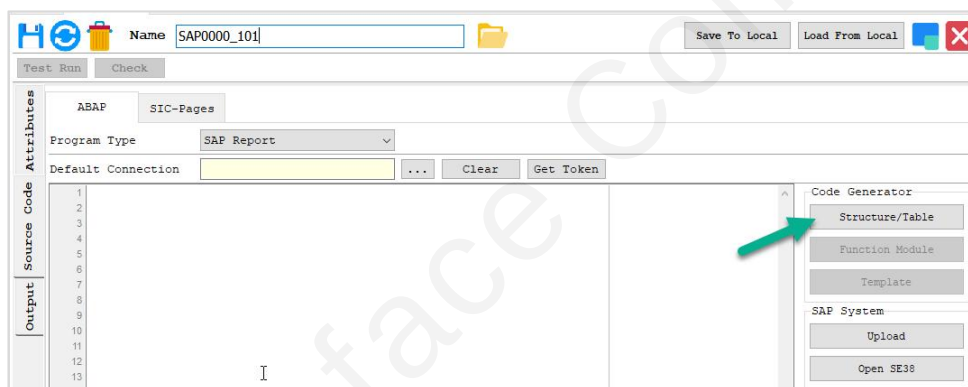
Source : -

Step to create a package:

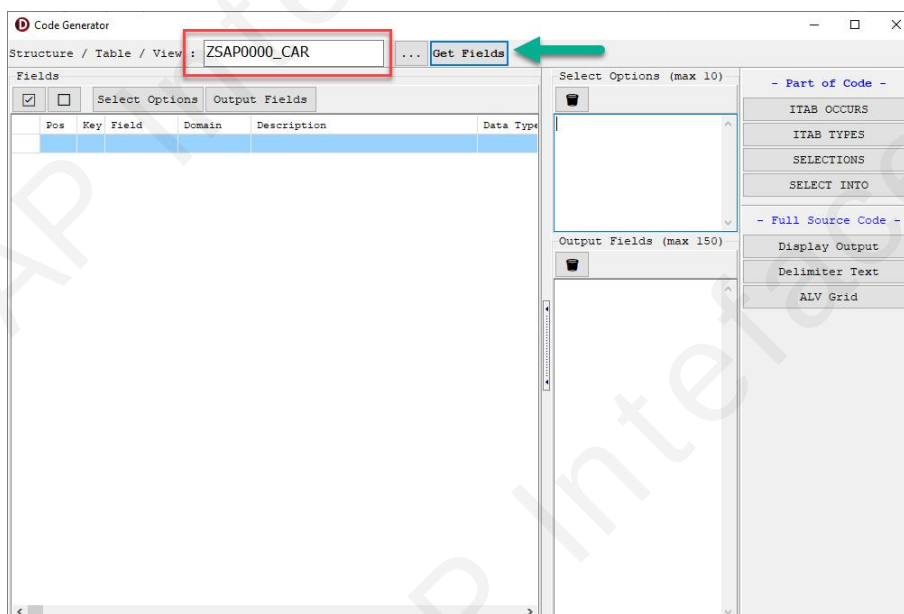
1. Open ARAD Application
2. Open Menu "SAP-Editor"
3. Enter Program Name "SAP0000_101" and click "Save" button



4. Click "Structure/Table"

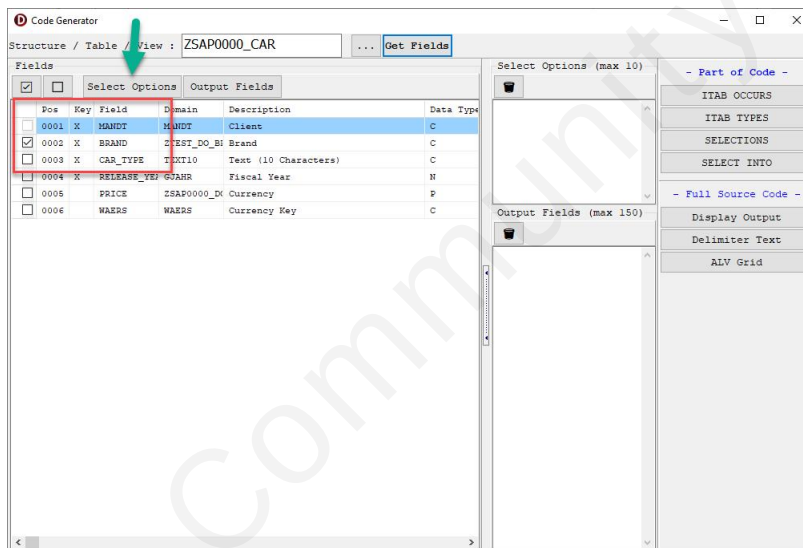


5. Enter "ZSAP0000_CAR" table and click "Get Fields"

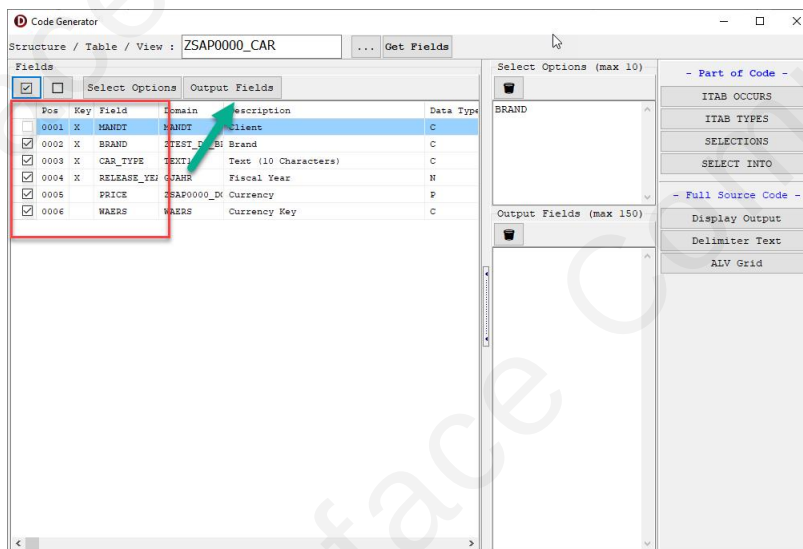


Wait until fields showed

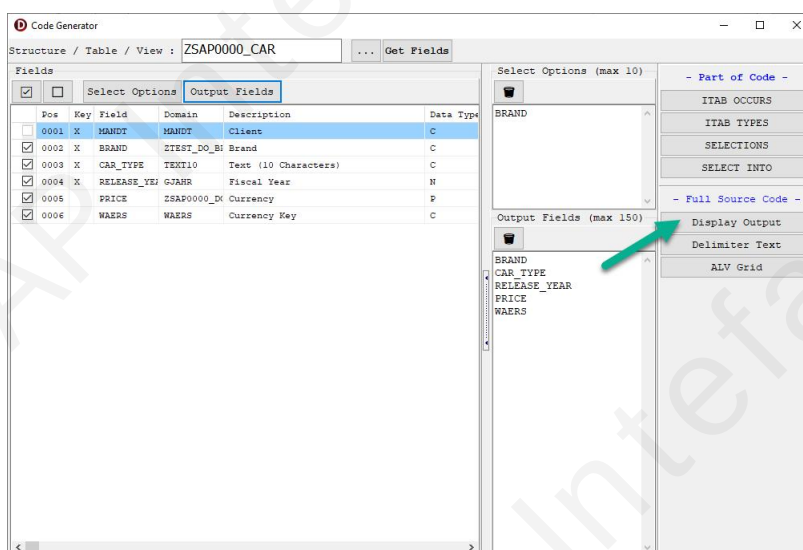
6. Select fields and click "Select-Options"



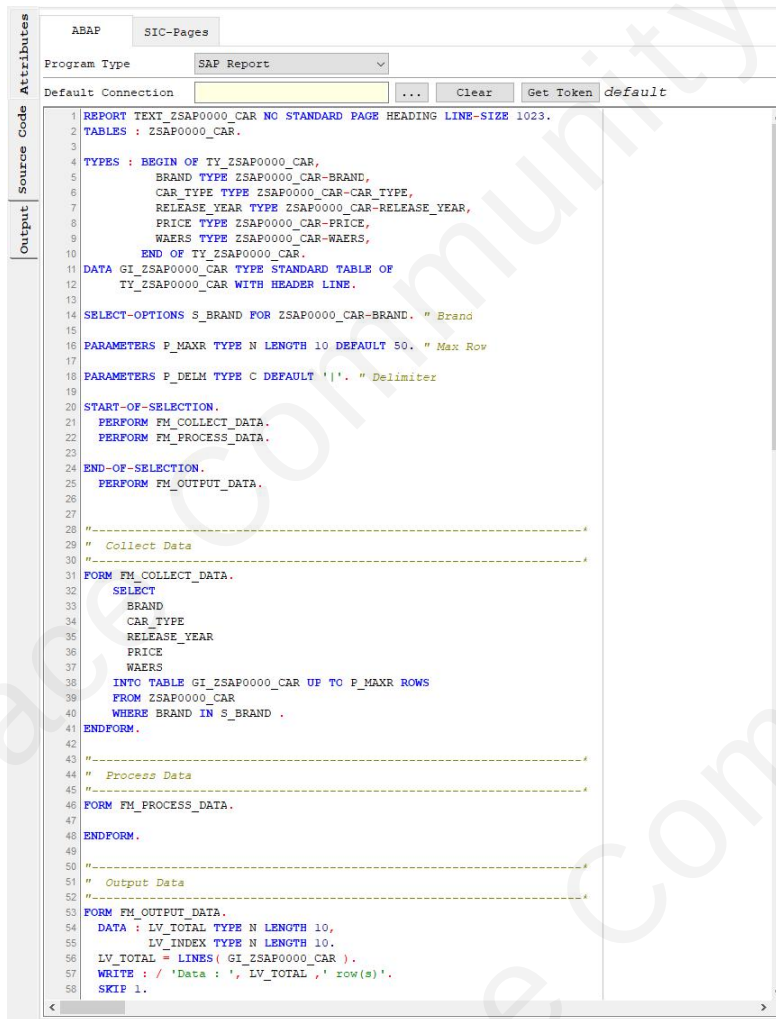
7. Select all fields and click "Output Fields"



8. Click "Display Output"



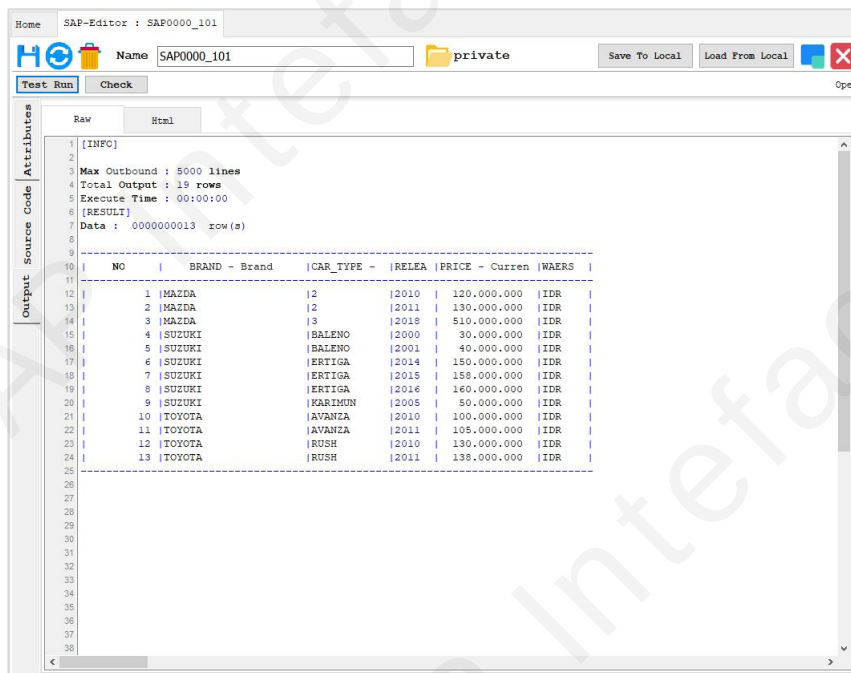
9. ABAP source code automatic generated and click "Save"



```
1 REPORT TEXT_ZSAP0000_CAR NO STANDARD PAGE HEADING LINE-SIZE 1023.
2 TABLES : ZSAP0000_CAR.
3
4 TYPES : BEGIN OF TY_ZSAP0000_CAR,
5         BRAND TYPE ZSAP0000_CAR-BRAND,
6         CAR_TYPE TYPE ZSAP0000_CAR-CAR_TYPE,
7         RELEASE_YEAR TYPE ZSAP0000_CAR-RELEASE_YEAR,
8         PRICE TYPE ZSAP0000_CAR-PRICE,
9         WAERS TYPE ZSAP0000_CAR-WAERS,
10        END OF TY_ZSAP0000_CAR.
11 DATA GI_ZSAP0000_CAR TYPE STANDARD TABLE OF
12        TY_ZSAP0000_CAR WITH HEADER LINE.
13
14 SELECT-OPTIONS S_BRAND FOR ZSAP0000_CAR-BRAND. " Brand
15
16 PARAMETERS P_MAXR TYPE N LENGTH 10 DEFAULT 50. " Max Row
17
18 PARAMETERS P_DELM TYPE C DEFAULT '|'. " Delimiter
19
20 START-OF-SELECTION.
21     PERFORM FM_COLLECT_DATA.
22     PERFORM FM_PROCESS_DATA.
23
24 END-OF-SELECTION.
25     PERFORM FM_OUTPUT_DATA.
26
27
28 "-----#
29 " Collect Data
30 "-----#
31 FORM FM_COLLECT_DATA.
32     SELECT
33         BRAND
34         CAR_TYPE
35         RELEASE_YEAR
36         PRICE
37         WAERS
38     INTO TABLE GI_ZSAP0000_CAR UP TO P_MAXR ROWS
39     FROM ZSAP0000_CAR
40     WHERE BRAND IN S_BRAND .
41 ENDFORM.
42
43 "-----#
44 " Process Data
45 "-----#
46 FORM FM_PROCESS_DATA.
47
48 ENDFORM.
49
50 "-----#
51 " Output Data
52 "-----#
53 FORM FM_OUTPUT_DATA.
54     DATA : LV_TOTAL TYPE N LENGTH 10,
55             LV_INDEX TYPE N LENGTH 10.
56     LV_TOTAL = LINES( GI_ZSAP0000_CAR ).
57     WRITE : / 'Data : ', LV_TOTAL, ' row(s)'.
58     SKIP 1.
```

10. Click "Test Run"

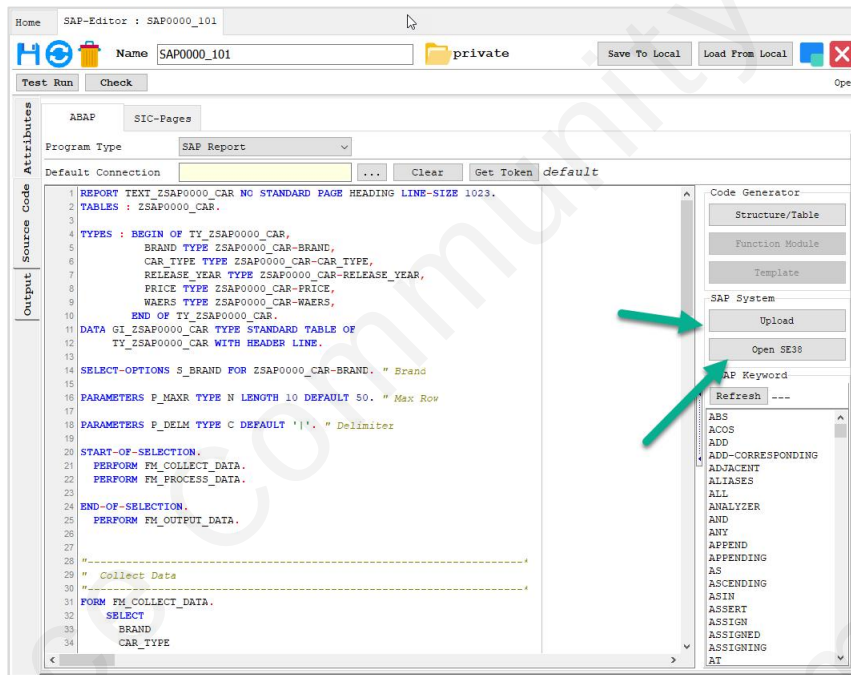
11. Result on "Output" tab



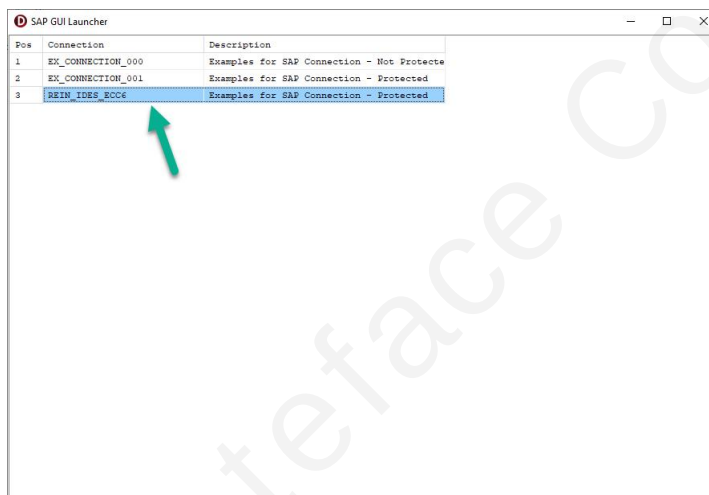
NO	BRAND - Brand	CAR_TYPE -	RELEA	PRICE - Curren	WAERS
1	MAZDA	12	2010	120.000.000	IDR
2	MAZDA	12	2011	130.000.000	IDR
3	MAZDA	13	2018	510.000.000	IDR
4	SUZUKI	BALENO	2000	30.000.000	IDR
5	SUZUKI	BALENO	2001	40.000.000	IDR
6	SUZUKI	ERTIGA	2014	180.000.000	IDR
7	SUZUKI	ERTIGA	2015	150.000.000	IDR
8	SUZUKI	ERTIGA	2016	160.000.000	IDR
9	SUZUKI	KARIMUN	2005	50.000.000	IDR
10	TOYOTA	AVANZA	2010	100.000.000	IDR
11	TOYOTA	AVANZA	2011	105.000.000	IDR
12	TOYOTA	RUSH	2010	130.000.000	IDR
13	TOYOTA	RUSH	2011	138.000.000	IDR

12. Back to "Source Code" tab

13. Click "Upload" and "Open SE38" button

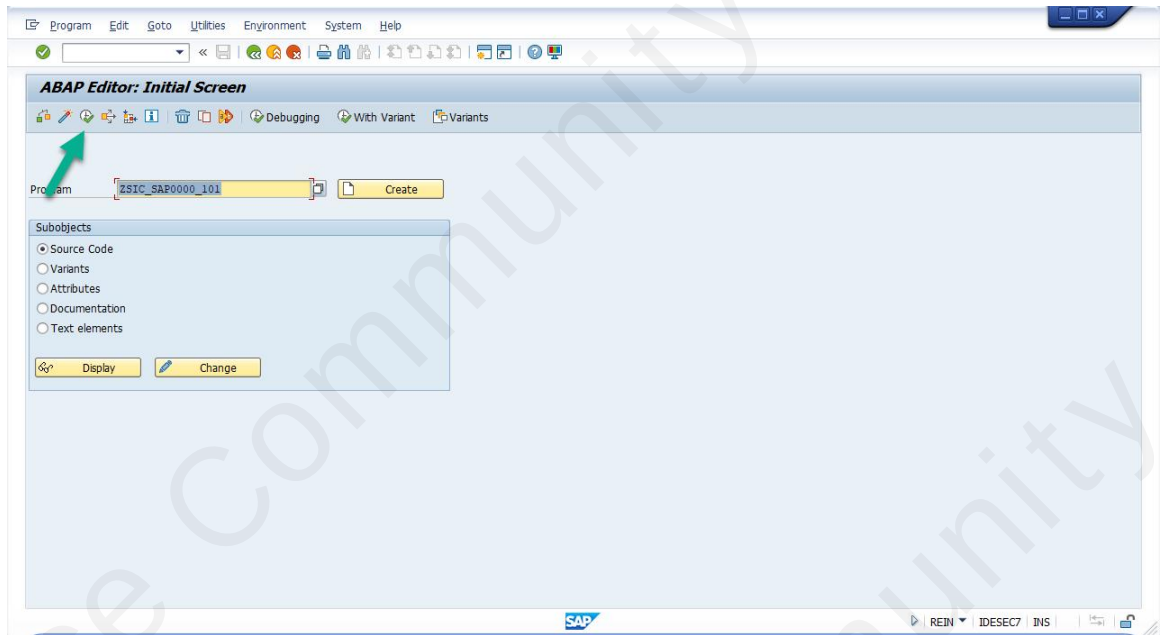


14. Select "SAP Sonnection" and double Click



Wait until SAP GUI showed

15. Click "Execute" button



16. Result

NO	BRAND - Brand	CAR_TYPE -	RELEA	PRICE - Curren	WAERS
1	MAZDA	2	2010	120.000.000	IDR
2	MAZDA	2	2011	130.000.000	IDR
3	MAZDA	3	2018	510.000.000	IDR
4	SUZUKI	BALENO	2000	30.000.000	IDR
5	SUZUKI	BALENO	2001	40.000.000	IDR
6	SUZUKI	ERTIGA	2014	150.000.000	IDR
7	SUZUKI	ERTIGA	2015	158.000.000	IDR
8	SUZUKI	ERTIGA	2016	160.000.000	IDR
9	SUZUKI	KARIMUN	2005	50.000.000	IDR
10	TOYOTA	AVANZA	2010	100.000.000	IDR
11	TOYOTA	AVANZA	2011	105.000.000	IDR
12	TOYOTA	RUSH	2010	130.000.000	IDR
13	TOYOTA	RUSH	2011	138.000.000	IDR

17. Done

D. Create a Report ARAD - SAP000_101A

Report Name : SAP0000_101A

Source : Create new and using "ZSAP0000_CAR_001" table as source

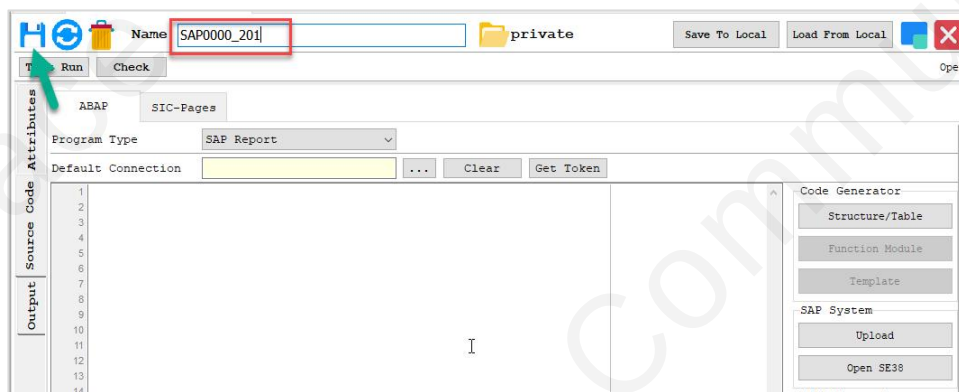
E. Create a Report ARAD - SAP000_201

Report Name : SAP0000_201

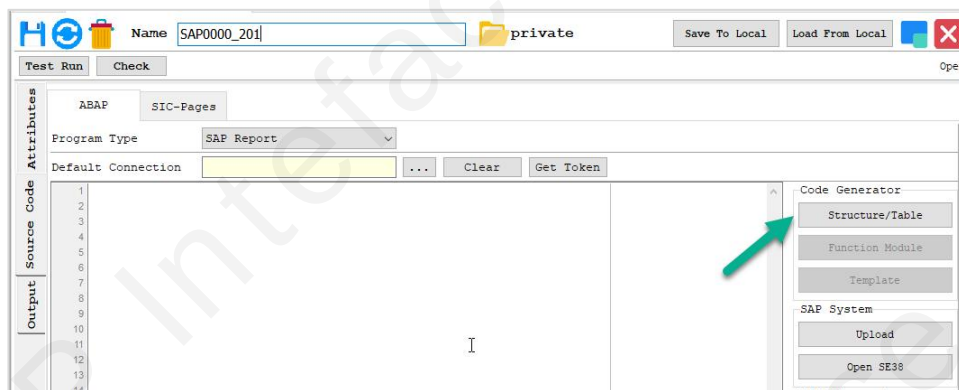
Source : -

Step to create a package:

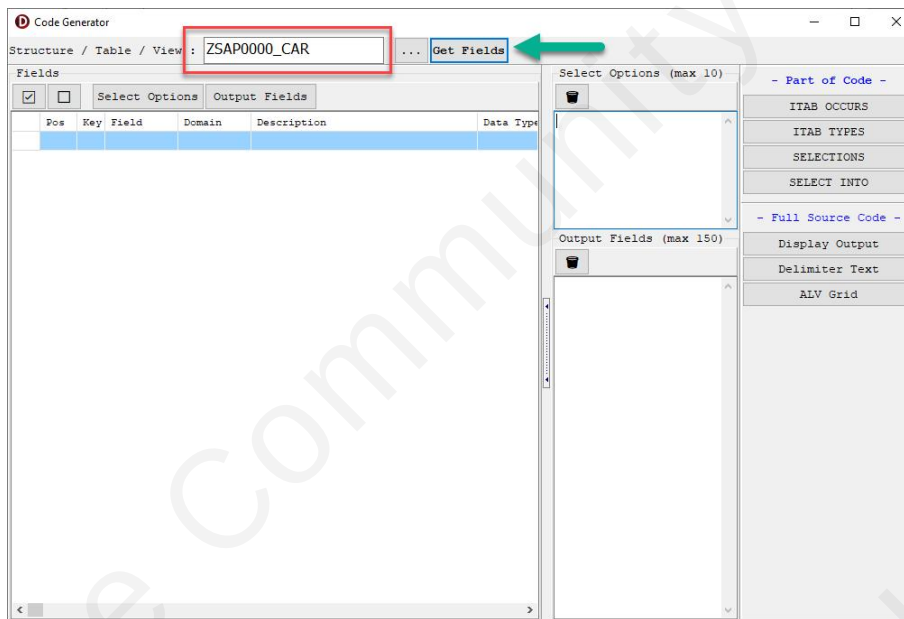
1. Open ARAD Application
2. Open Menu "SAP-Editor"
3. Enter Program Name "SAP0000_201" and click "Save" button



4. Click "Structure/Table"

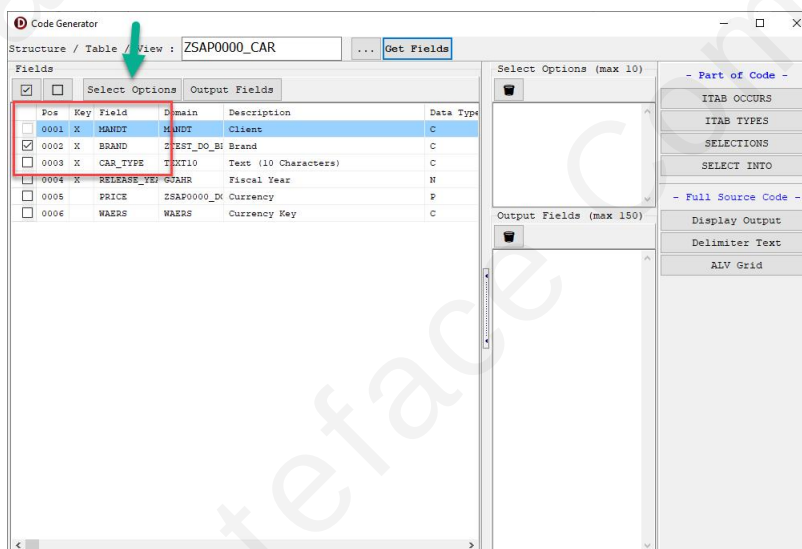


5. Enter "ZSAP0000_CAR" table and click "Get Fields"

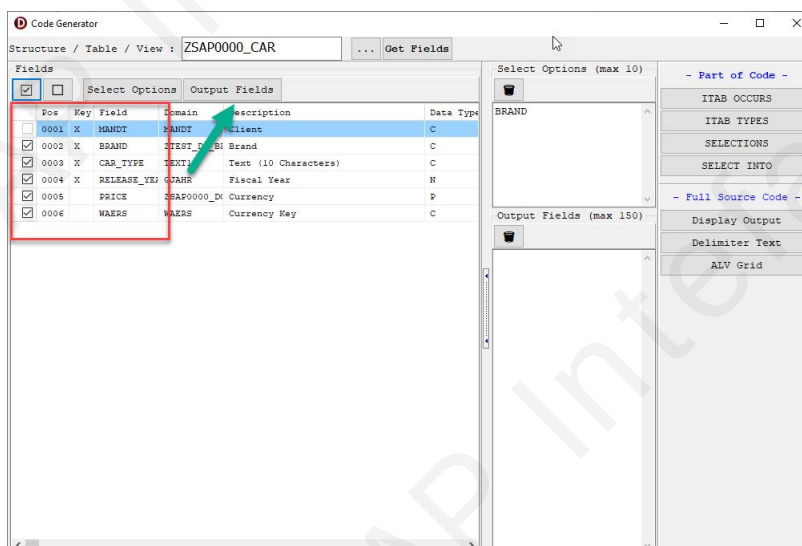


Wait until fields showed

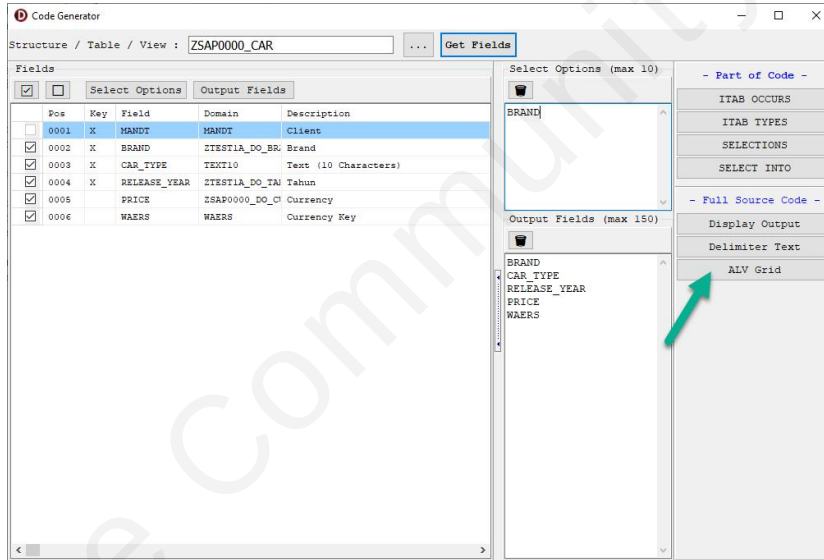
6. Select fields and click "Select-Options"



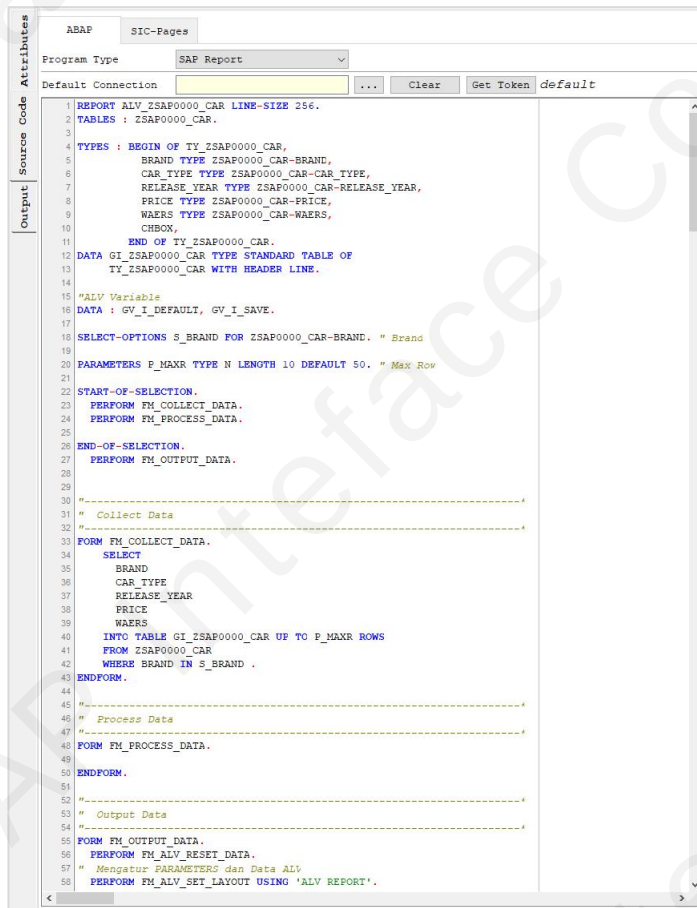
7. Select all fields and click "Output Fields"



8. Click "ALV Grid"

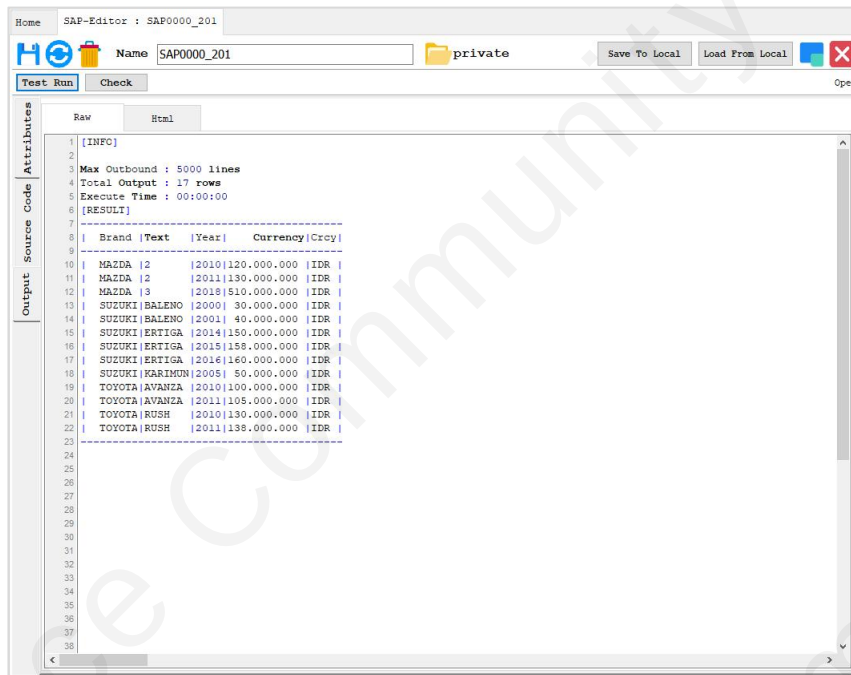


9. ABAP source code automatic generated and click "Save"



10. Click "Test Run"

11. Result on "Output" tab

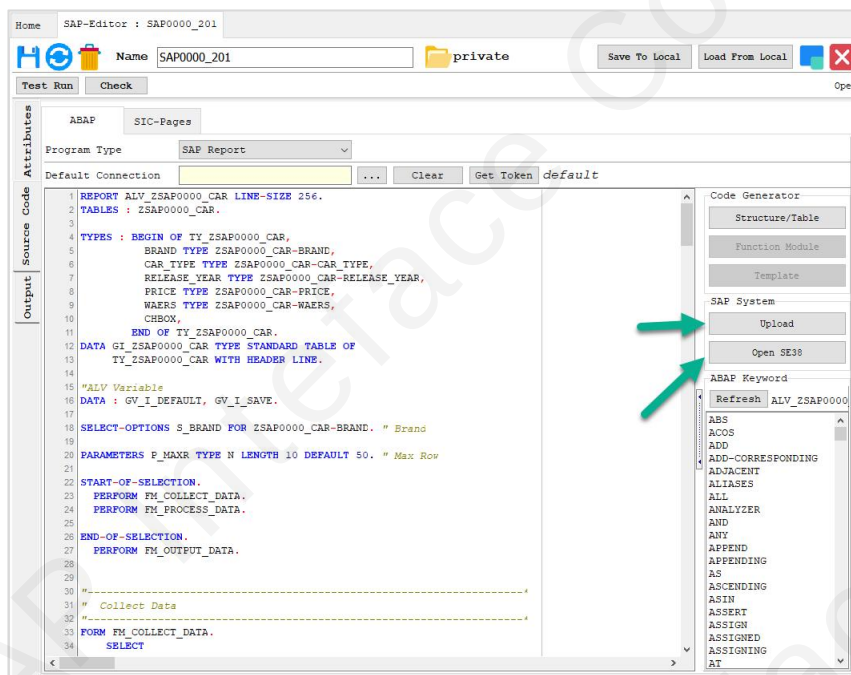


The screenshot shows the SAP-Editor interface with the 'Output' tab selected. The table displays the following data:

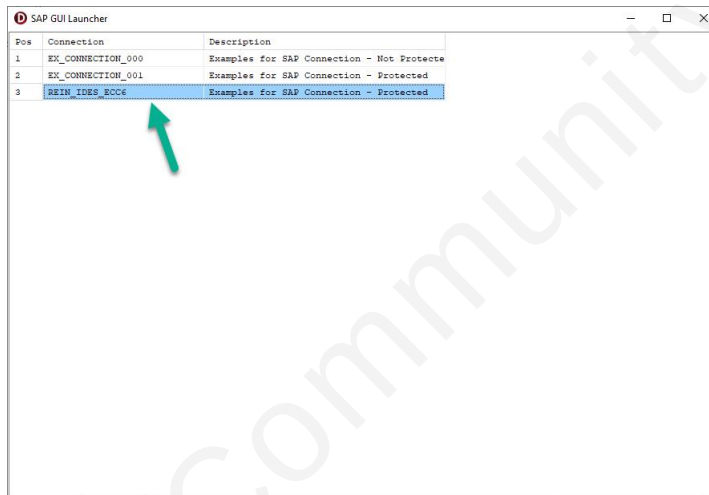
Brand	Text	Year	Currency	Crcy
MAZDA	2	2010	120.000.000	IDR
MAZDA	2	2011	130.000.000	IDR
MAZDA	3	2018	510.000.000	IDR
SUZUKI	BALENO	2000	30.000.000	IDR
SUZUKI	BALENO	2001	40.000.000	IDR
SUZUKI	ERTIGA	2014	150.000.000	IDR
SUZUKI	ERTIGA	2015	158.000.000	IDR
SUZUKI	ERTIGA	2016	160.000.000	IDR
SUZUKI	KARIMUN	2005	50.000.000	IDR
TOYOTA	AVANZA	2010	100.000.000	IDR
TOYOTA	AVANZA	2011	108.000.000	IDR
TOYOTA	RUSH	2010	130.000.000	IDR
TOYOTA	RUSH	2011	138.000.000	IDR

12. Back to "Source Code" tab

13. Click "Upload" and "Open SE38" button

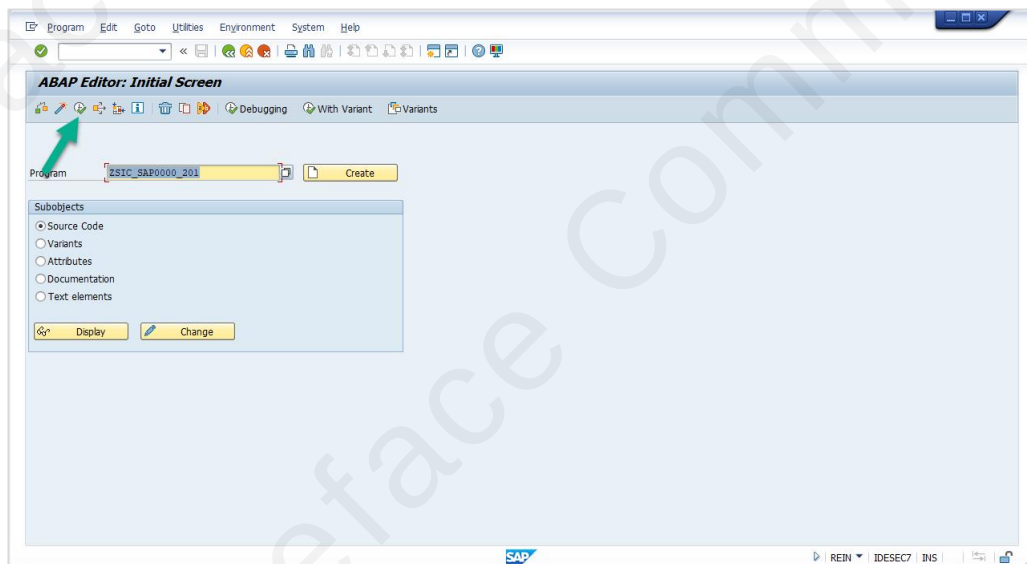


14. Select "SAP Sconnction" and double Click

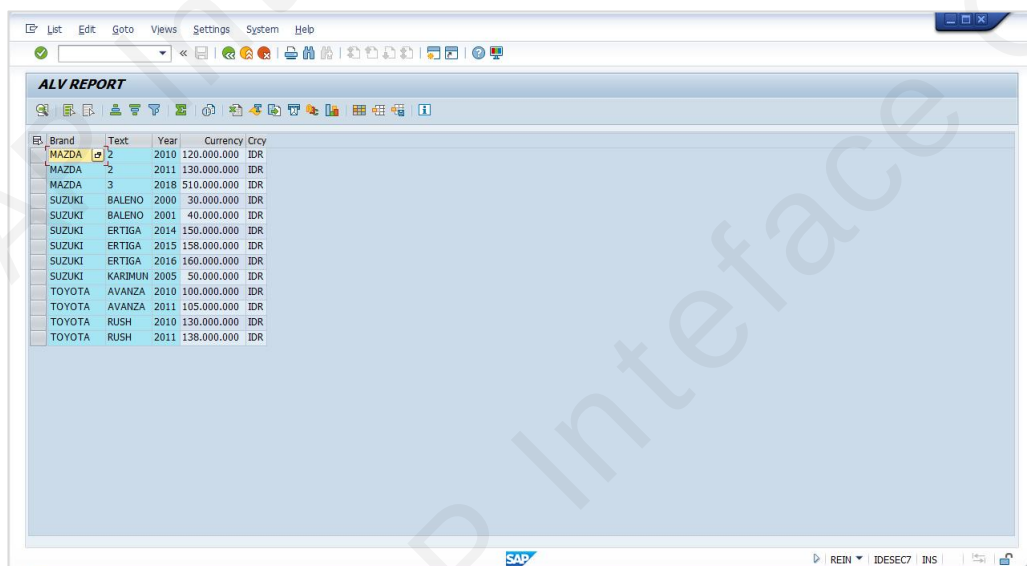


Wait until SAP GUI showed

15. Click "Execute" button



16. Result



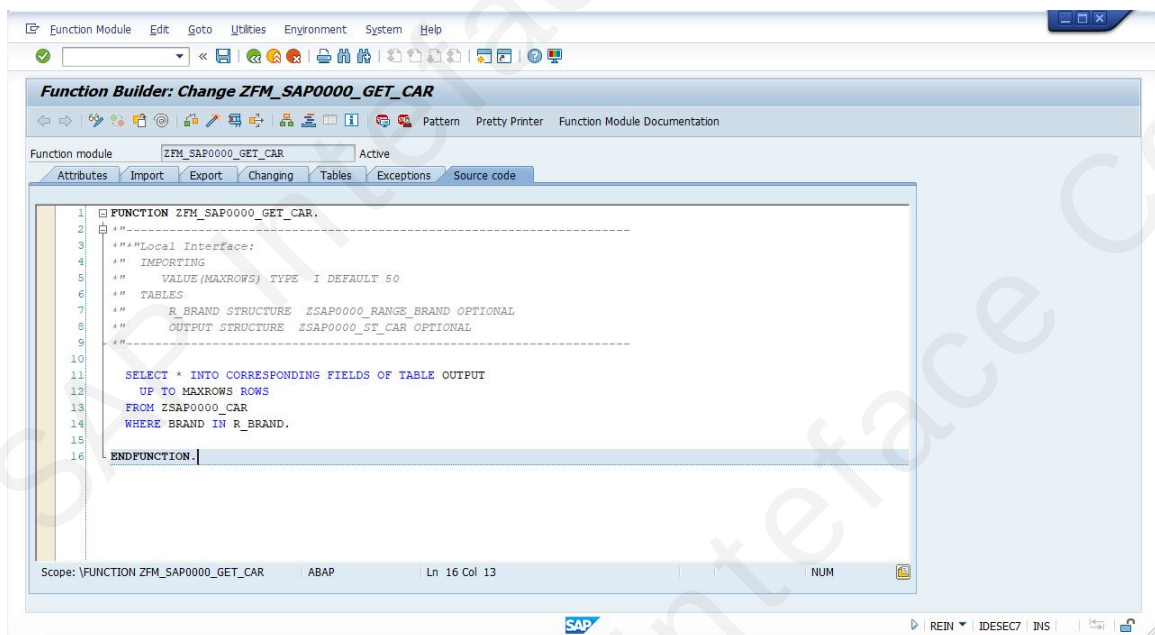
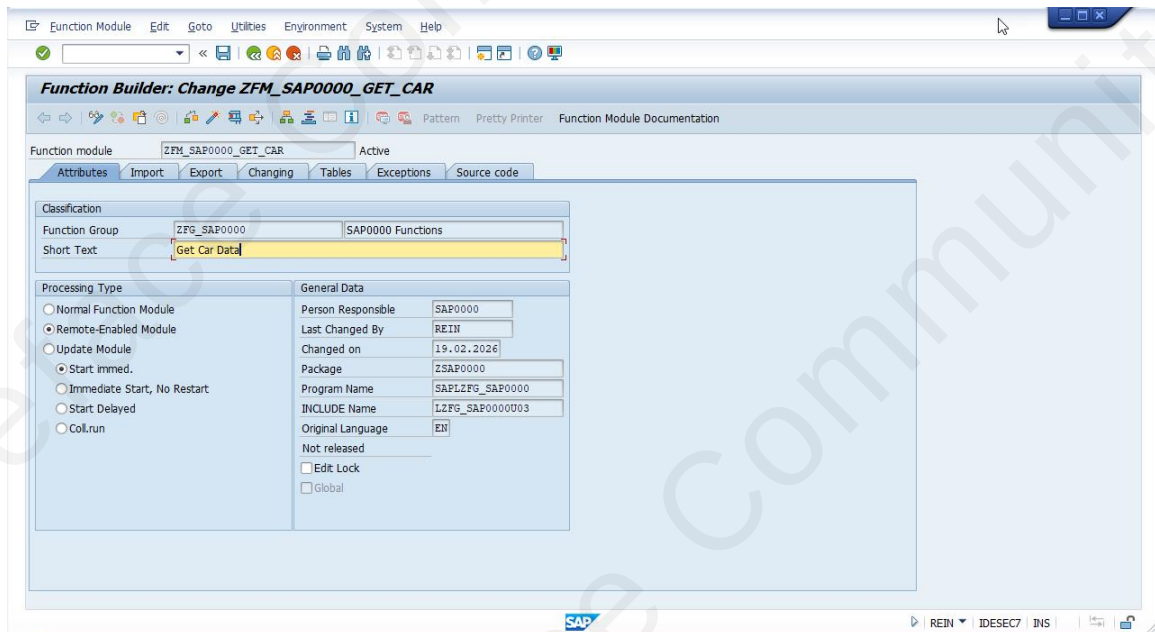
17. Done

Day - 5

TCode	SE11, SE16, SE37, SE38
Target	Understanding Screen Interactive in SAP

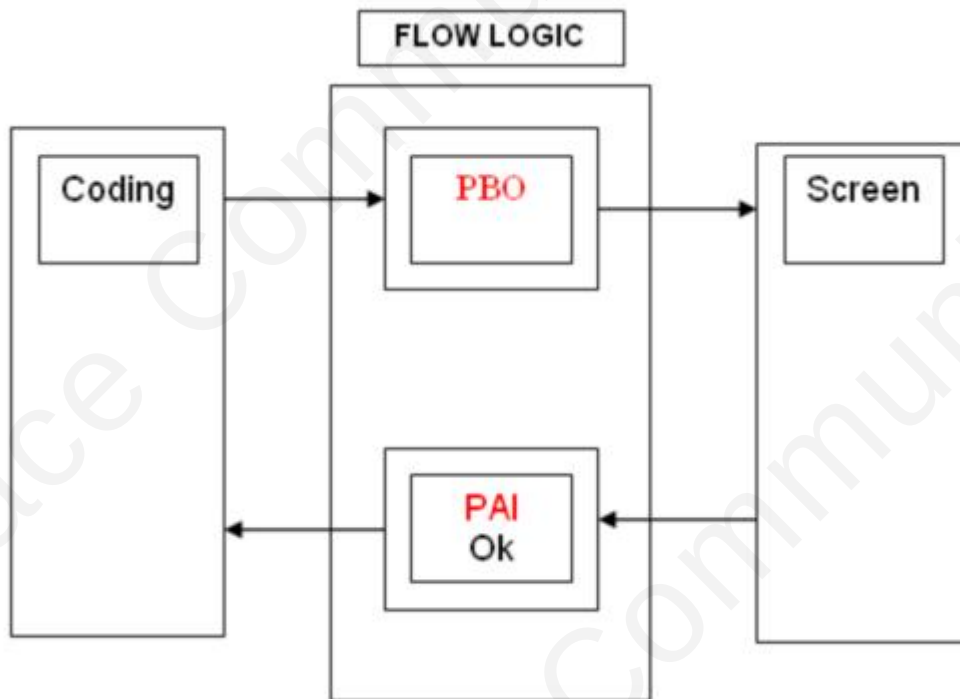
1. Function Module Basic 2

In many projects, function module is used to encapsulate a data collection command. The function module can be called by many ABAP programs from the same system and also from another system.



2. ABAP Dialog Programming

Screen Form in ABAP can be create using "Screen Painter" and only available in SAP GUI for Windows, not in PlatinGUI/ GUI Java. Make sure the component of "Screen Painter" is selected when you installed SAP GUI.

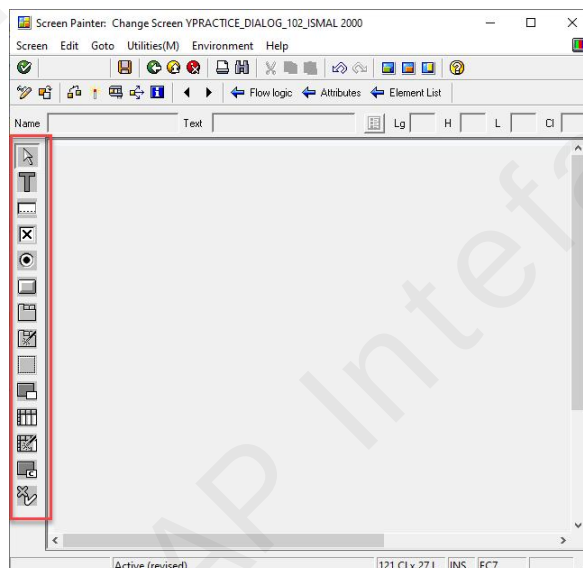


Between Coding and screen is connected by flow logic. In flow logic contains to event, which is PBO (Processes Before Output) and PAI (Processes Before Input).

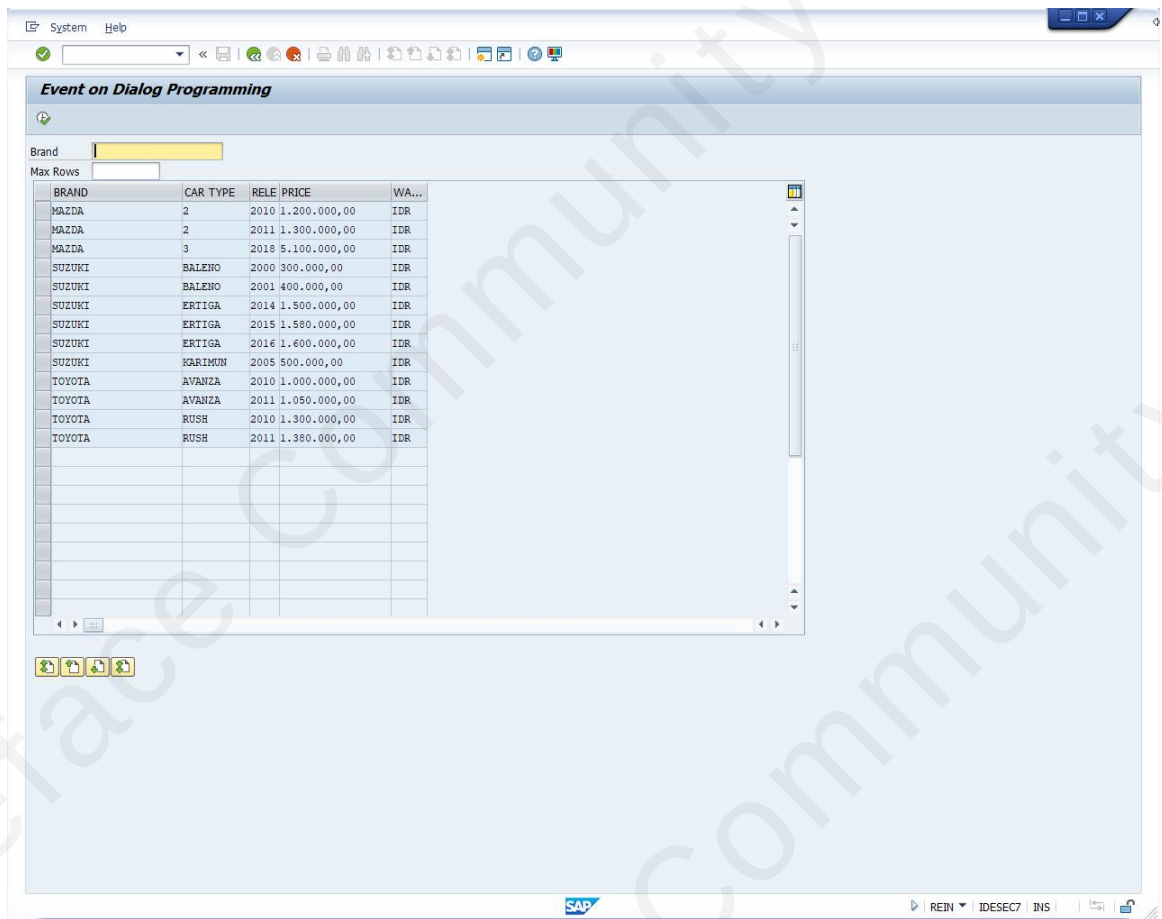
PBO is same with OnShow event at another programming language.

PAI is same with OnClick or OnEnter event at another programming language.

Screen Painter has 13 element/component that can be used for creating a screen.



Examples "Display Car Master Data"



Practice Day - 4

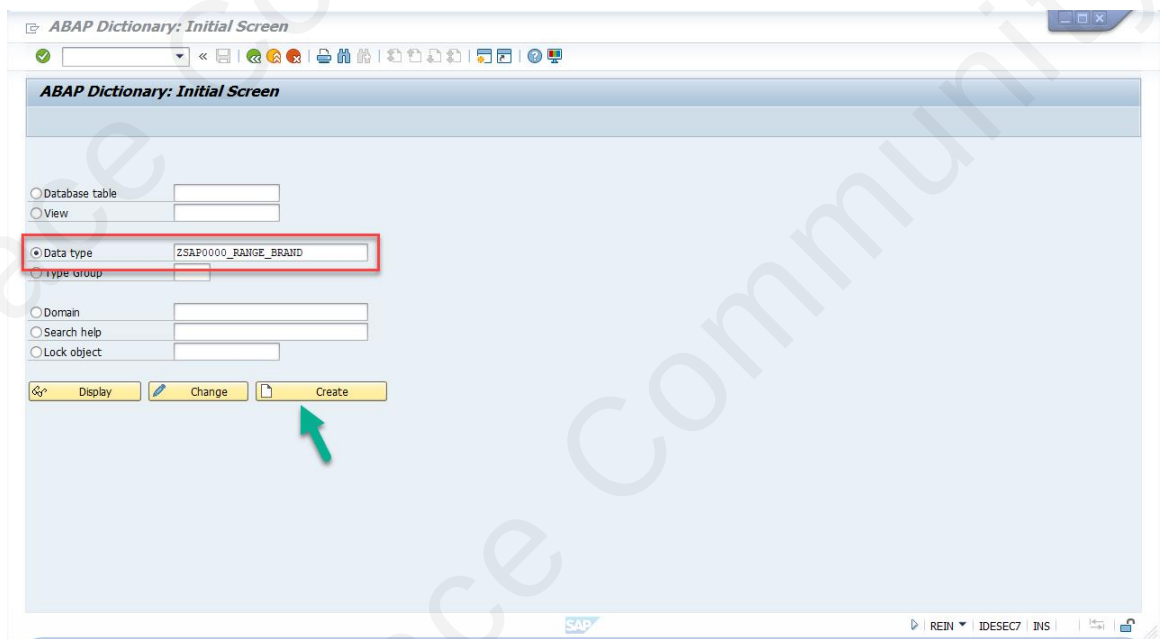
A. Create a Function Module

Function Module : ZFM_SAP0000_GET_CAR

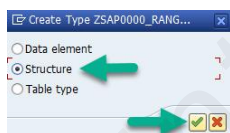
Source : Example Function for Get "Car" Master Data

Step to create a package:

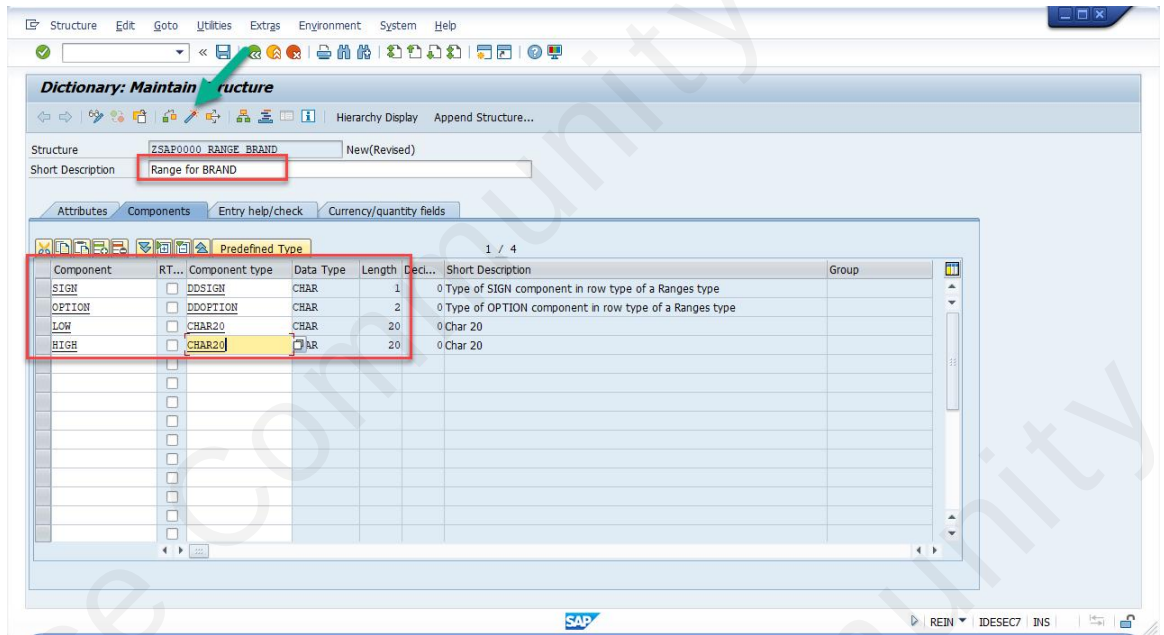
1. Open TCode SE11
2. Select "Data Type", enter value "ZSAP0000_RANGE_BRAND" and then click "Create" button



3. Select "Structure" and click "continue"



4. Enter value like this bellow and click "activate" button

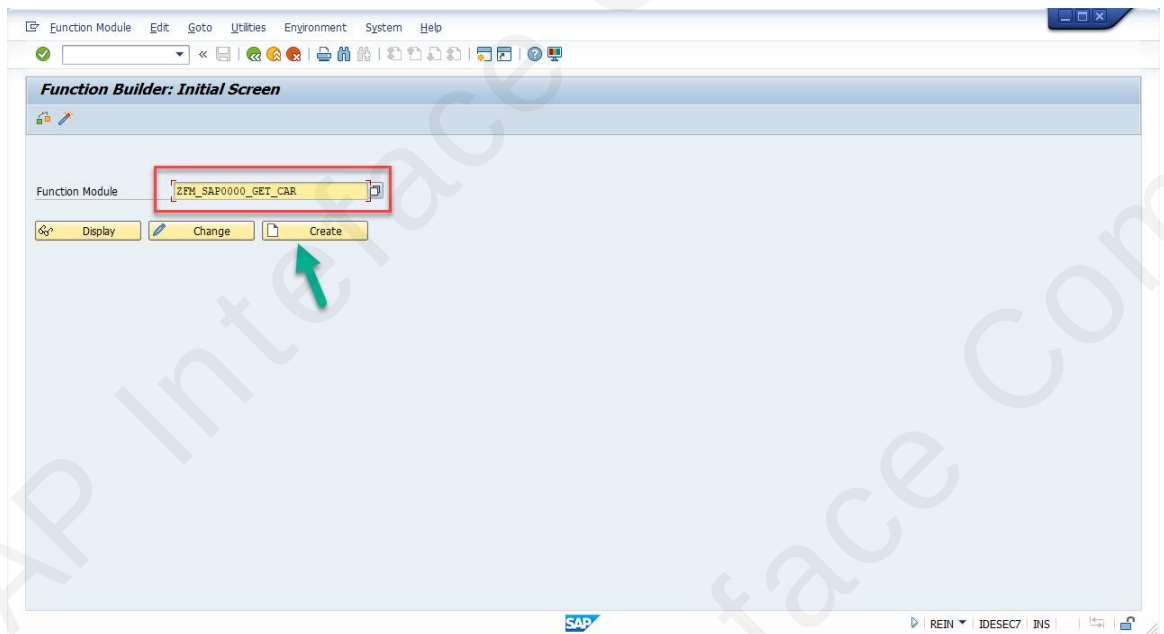


5. Save to package and enter transport request

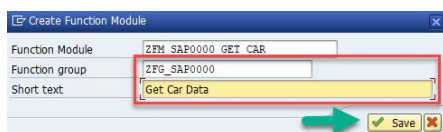
6. Structure "ZSAP0000_RANGE_BRAND" created

7. Open TCode SE37

8. Enter "ZFM_SAP0000_GET_CAR" value as Function Module Name and click "Create" button



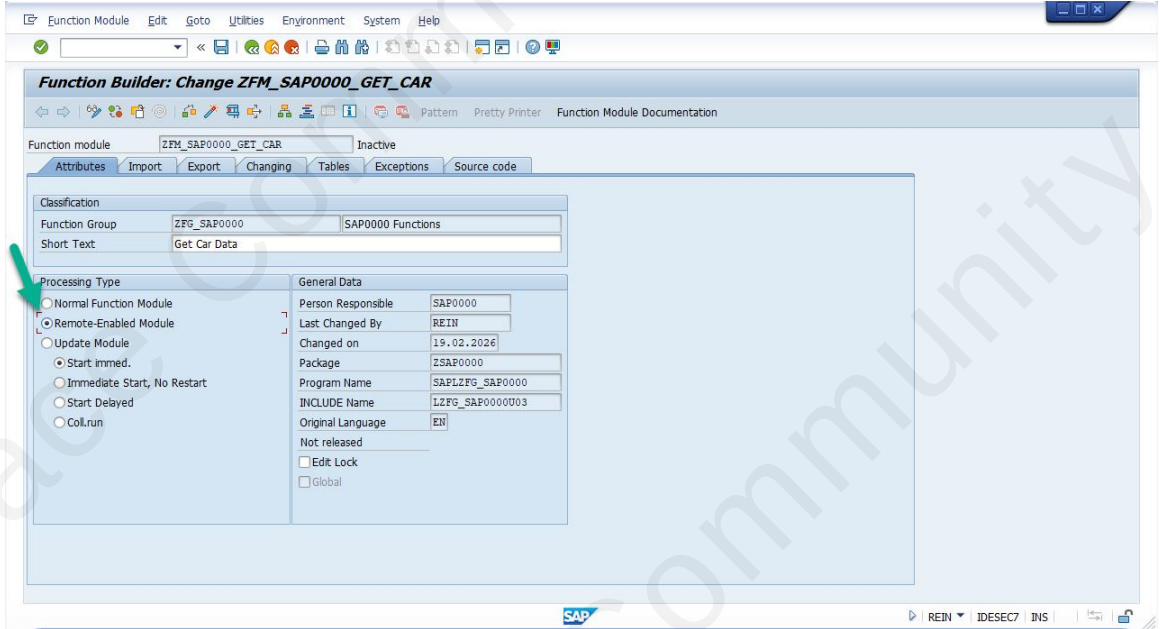
9. Enter value like this bellow and then click "Save"



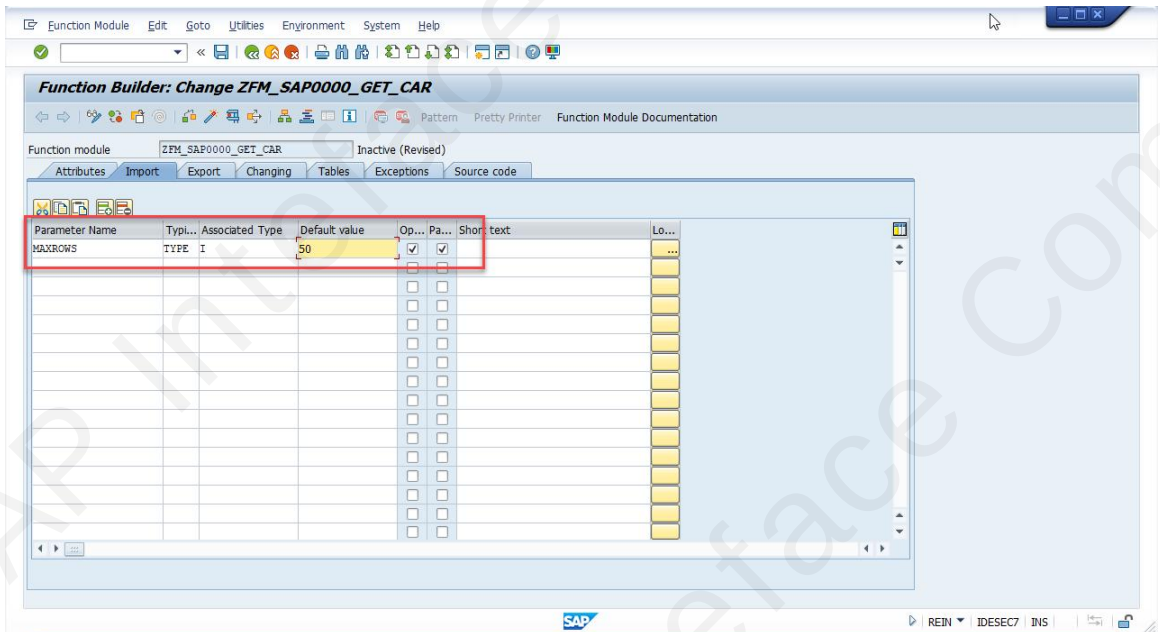
10. Click "Continue"



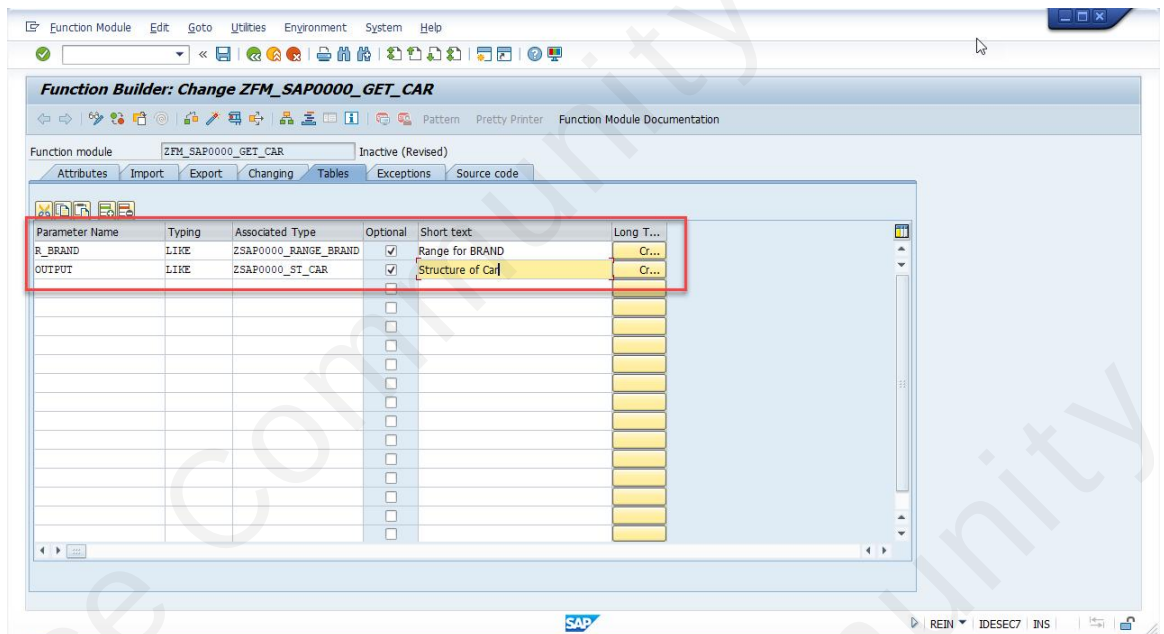
11. Set Attribute "Remote-Enabled Module"



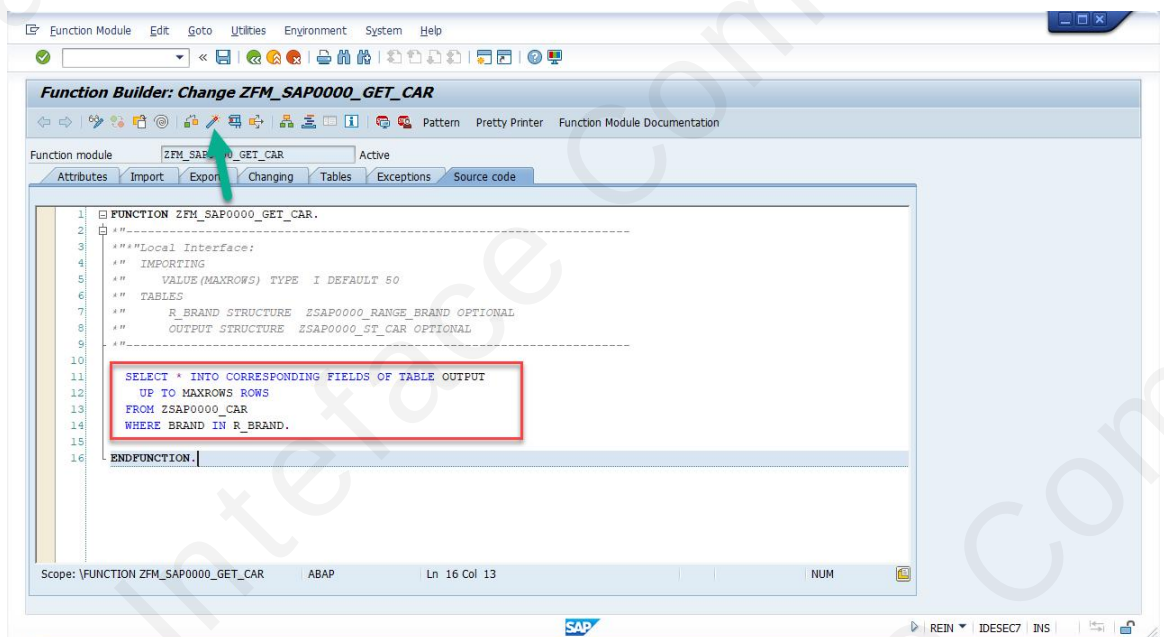
12. Set Import parameter



13. Set Tables parameter



14. Write Source Code like this bellow and click "activate"



15. Done

B. Create a ABAP Dialog Programming 101

Report Name : ZSAP0000_D101

Source : -

Step to create a package:

1. Open TCode SE38
2. Enter Program Name "ZSAP0000_D101" and click "Create"
3. Enter Title "Event on Dialog Programming", Select type "Executable Program" and click "Save"

The screenshot shows the 'ABAP: Program Attributes ZSAP0000_D101 Change' dialog box. The 'Title' field is highlighted with a red box and contains the text 'Event on Dialog Programming'. The 'Type' dropdown menu is also highlighted with a red box and shows '1 Executable program'. At the bottom right, a green arrow points to the 'Save' button.

4. Save to Package and transport request
5. Write Source Code like this bellow

```
REPORT ZSAP0000_D101.

*-----*
*Variable for Screen Command
*-----*

DATA : OK_CODE LIKE SY-UCOMM,
      SAVE_OK LIKE SY-UCOMM.

*-----*
*Set First Screen to Displayed
*-----*
call screen 2000.

*-----*
MODULE STATUS_2000 OUTPUT.
  SET PF-STATUS 'ST_2000'.
  SET TITLEBAR 'TL_2000'.
ENDMODULE.                                "STATUS_2000 OUTPUT

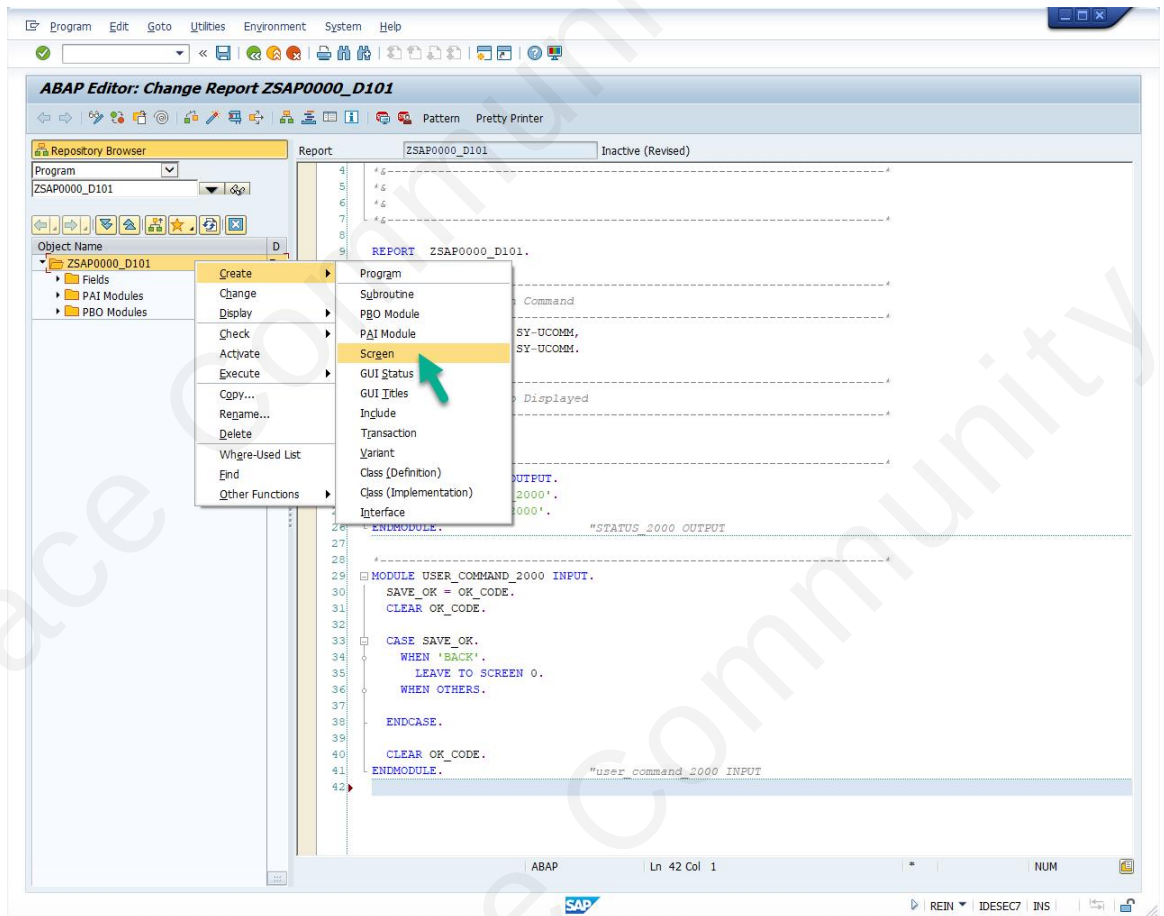
*-----*
MODULE USER_COMMAND_2000 INPUT.
  SAVE_OK = OK_CODE.
  CLEAR OK_CODE.

  CASE SAVE_OK.
    WHEN 'BACK'.
      LEAVE TO SCREEN 0.
    WHEN OTHERS.
  ENDCASE.

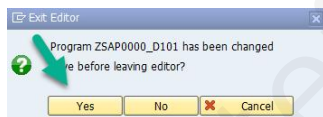
  CLEAR OK_CODE.
ENDMODULE.                                "user_command_2000 INPUT
```

6. Click "Display Object List"

7. Right click on top of Object Name and click "Screen"



8. Click "Yes"



9. Enter screen number "2000" and click "Continue"



10. Enter description "Main Screen" and click "Element List" tab

Screen number: 2000 New(Revised)

Attributes | Element list | Flow logic

Short Description: **Main Screen**

Original Language: EN English

Last changed on/at: 00:00:00

Last Generation: 00:00:00

Screen Type:

- ☒ Normal
- ☐ Subscreen
- ☐ Modal dialog box
- ☐ Selection screen

Settings:

- ☐ Hold Data
- ☐ Switch Off Runtime Compress
- ☐ Template - non-executable
- ☐ Hold Scroll Position
- ☐ Without Application Toolbar

Other Attributes:

Next Screen: 2000

Cursor Position:

Screen Group:

Lines/Columns: Occupied 0 0

Mainten.: 27 120

Context Menu FORM ON CTMENU

Properties

11. Enter element name "OK_CODE" and click "Flow Logic" tab

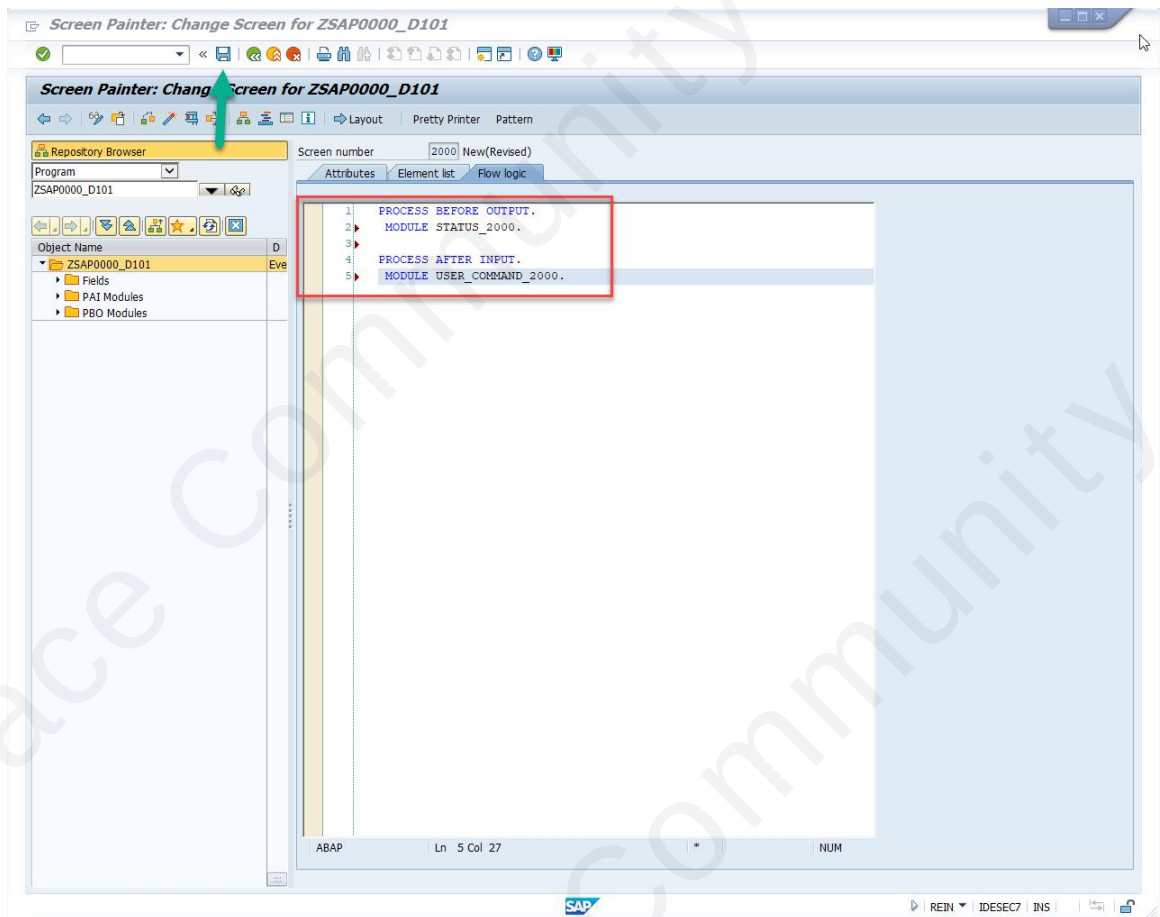
Screen number: 2000 New(Revised)

Attributes | Element list | Flow logic

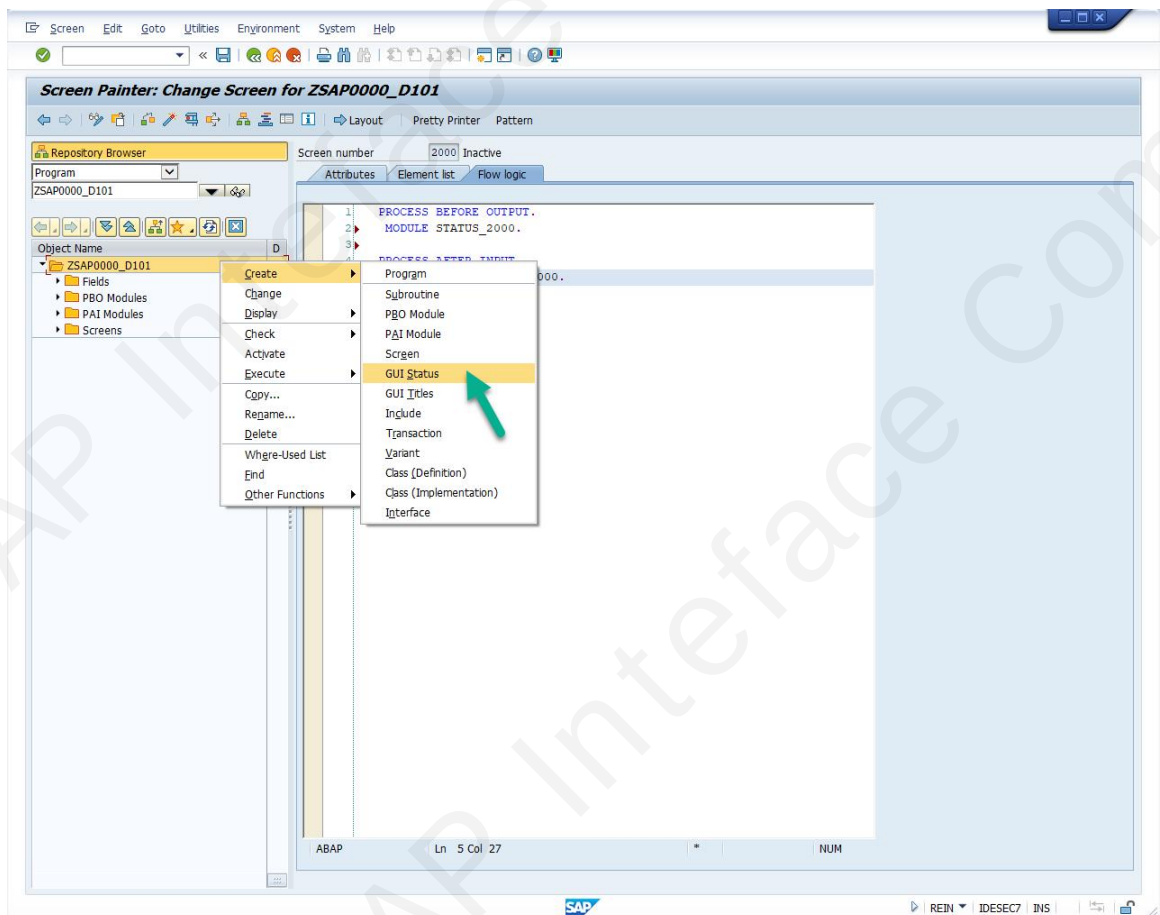
General attr. | Texts/ I/O templates | Special attr. | Display attr. | Mod. groups / functions | References

Element name	Type	Li...	C...	D...	V...	H...	S...	Format	I...	O...	Ou...	D...	Dic...	Property list
OK_CODE	OK	0	0	20	20	1		OK						

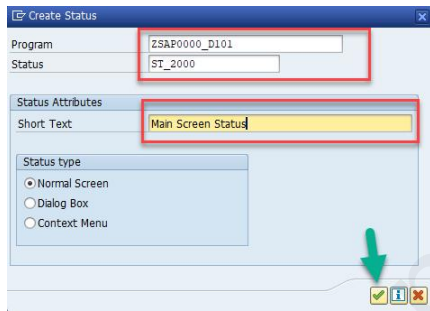
12. Uncomment all line code and click "Save"



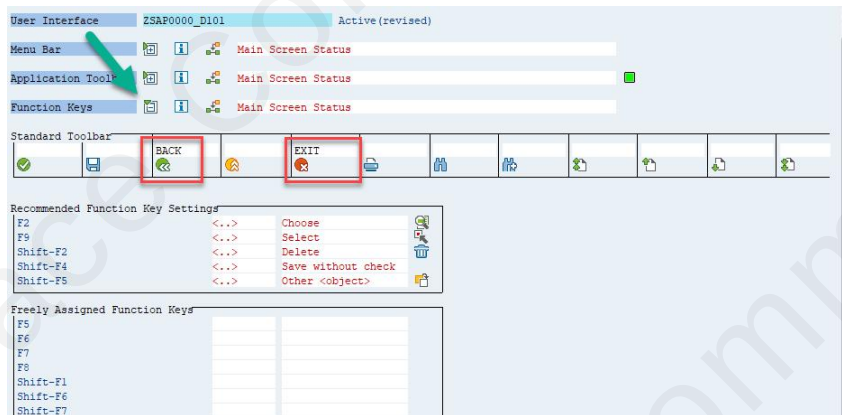
13. Right click on top of Object Name and click "GUI Status"



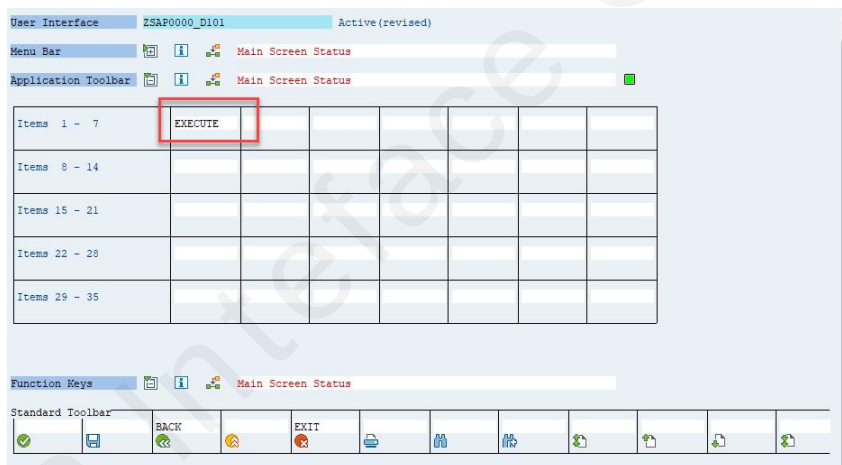
14. Enter value like this bellow and click "continue"



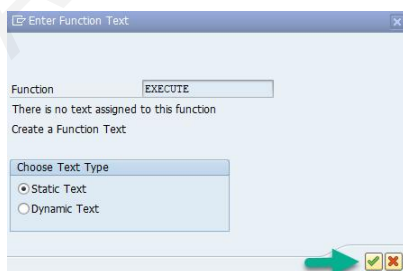
15. Expand "Function Key" and set Toolbar as "BACK" and "EXIT"



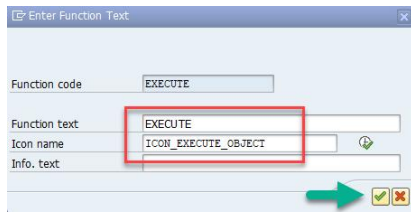
16. Expand "Application Toolbar" and press "Enter"



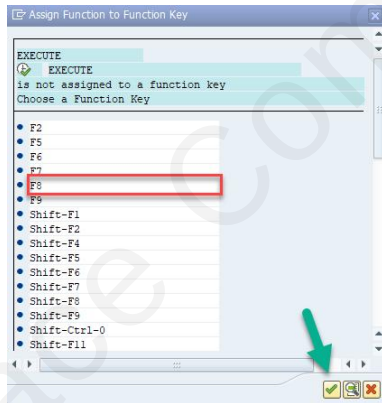
17. Click "Continue"



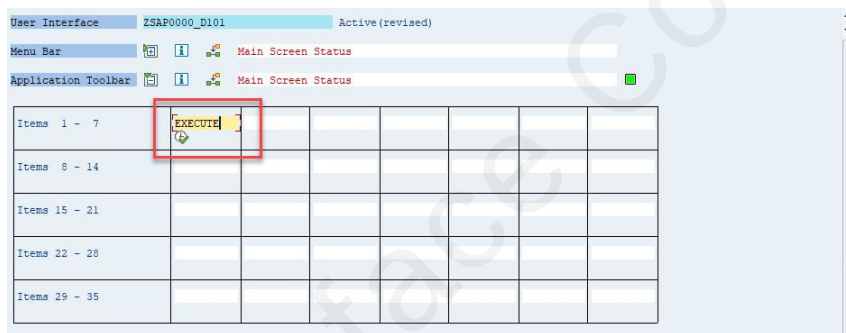
18. Enter value like this bellow and click "Continue"



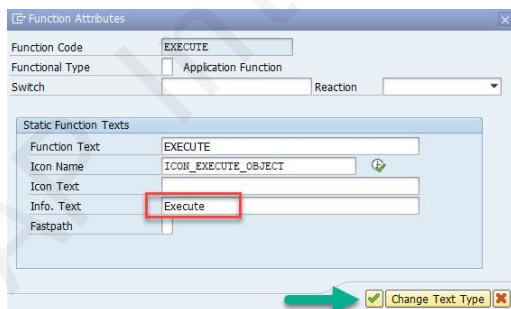
19. Select/click "F8" and click "Continue"



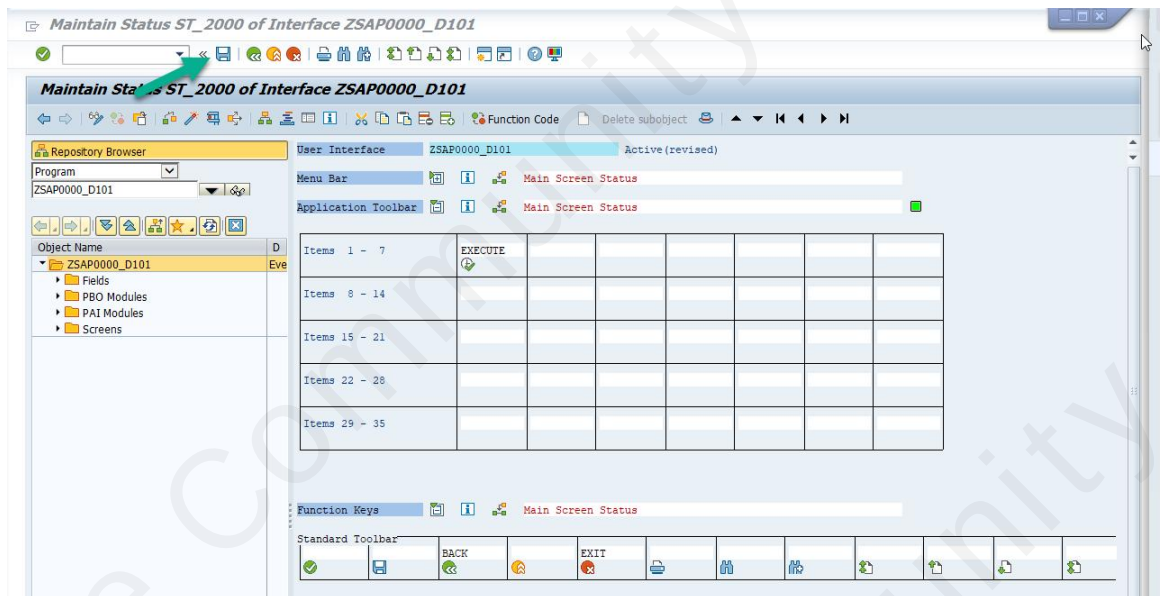
20. Double click "EXECUTE"



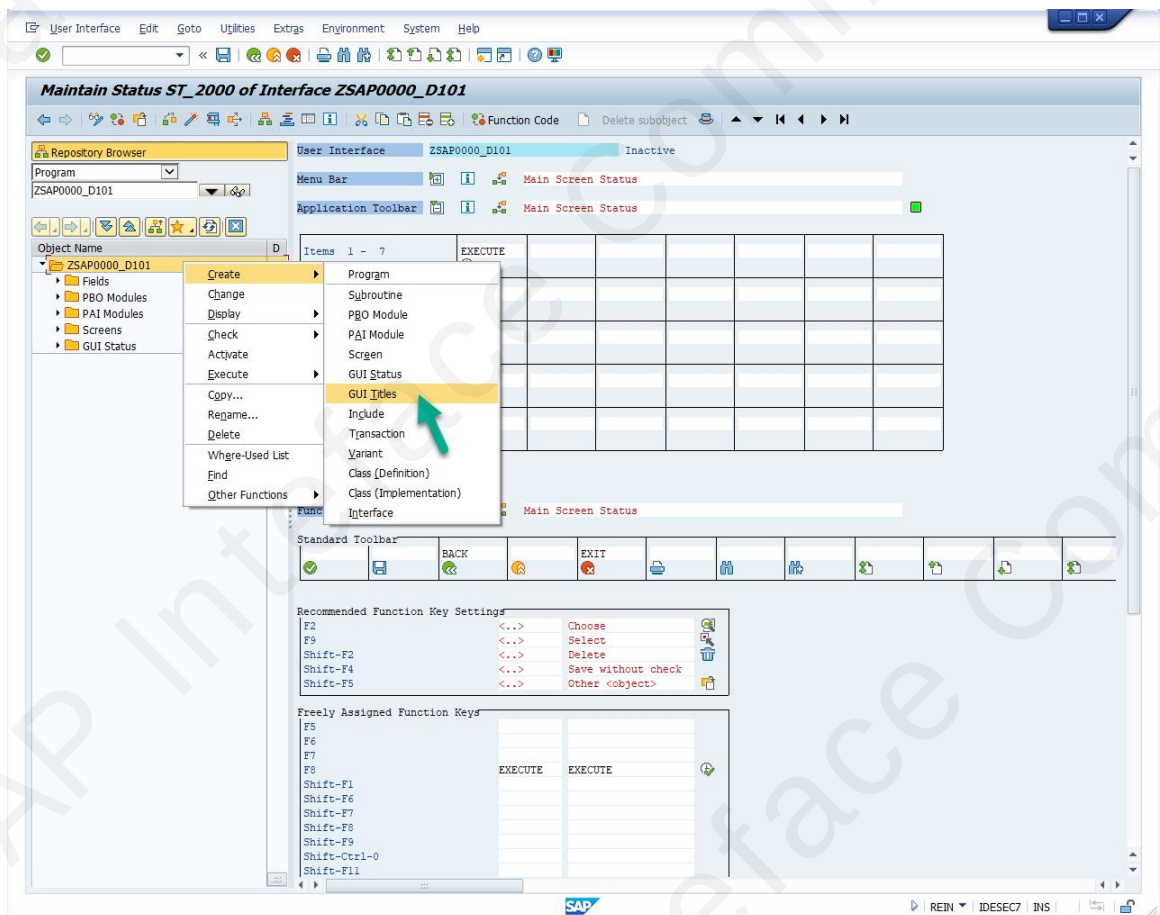
21. Enter info text "Execute" and click "Continue"



22. Click "Save" button



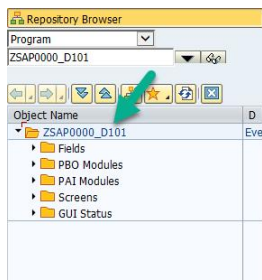
23. Right click on top of Object Name and click "GUI Title"



24. Enter title code and title like bellow and then click "Continue"



25. Double click on top of Object Name



26. Click "Yes" to save last object



27. Edit module "USER_COMMAND_2000"

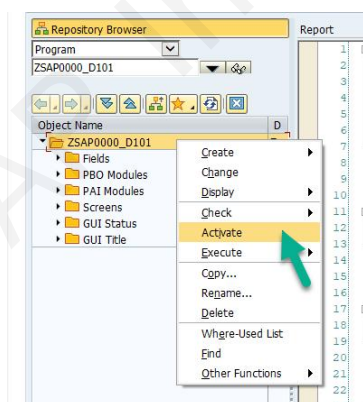
Before

```
CASE SAVE_OK.  
  WHEN 'BACK'.  
    LEAVE TO SCREEN 0.  
  WHEN OTHERS.  
  
ENDCASE.
```

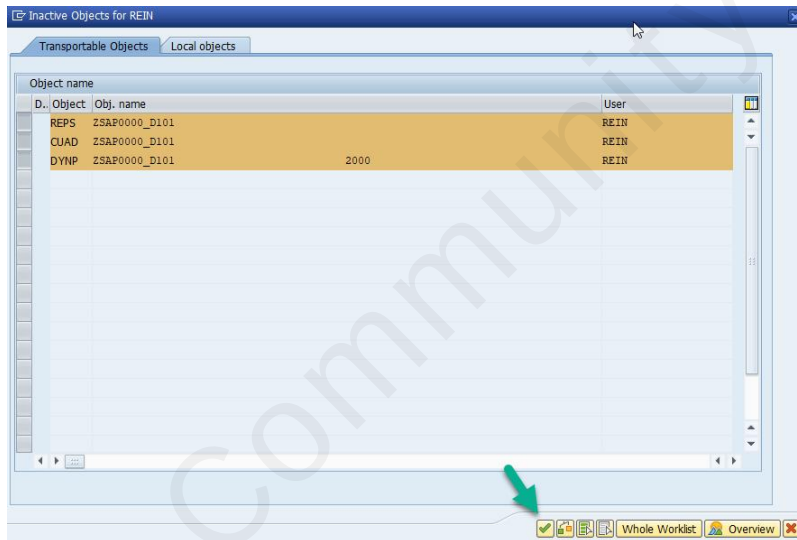
After

```
CASE SAVE_OK.  
  WHEN 'BACK'.  
    LEAVE TO SCREEN 0.  
  WHEN 'EXIT'.  
    LEAVE PROGRAM.  
  WHEN 'EXECUTE'.  
    MESSAGE 'Hello world' TYPE 'I'.  
  WHEN OTHERS.  
  
ENDCASE.
```

28. Right click on top of Object Name and click "activate"



29. Click "continue"



30. Done

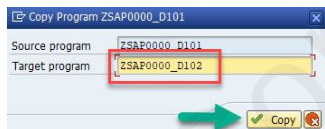
C. Create a ABAP Dialog Programming 102

Report Name : ZSAP0000_D102

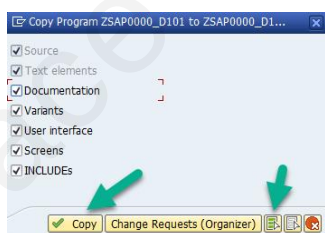
Source : -

Step to create a package:

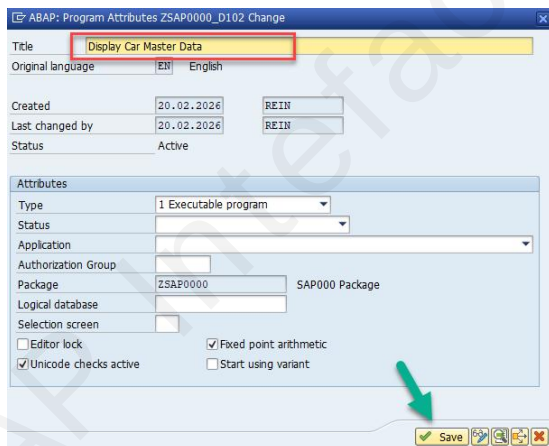
1. Open TCode SE38
2. Enter Program Name "ZSAP0000_D101" and click "Copy"
3. Enter target program "ZSAP0000_D102" and click "copy"



4. Click "Select All" and click "copy"



5. Save to Package and transport request
6. Click "Change" button
7. Click menu "Go to - Attributes"
8. Change Title to "Display Car Master Data" and click "Save"



9. Declare new variable

Before

```
REPORT ZSAP0000_D102.

*-----*
*Variable for Screen Command
*-----*

DATA : OK_CODE LIKE SY-UCOMM,
      SAVE_OK LIKE SY-UCOMM.
```

After

```
REPORT ZSAP0000_D102.

TABLES : ZSAP0000_CAR.

TYPES : BEGIN OF TY_DATA.
        INCLUDE STRUCTURE ZSAP0000_ST_CAR.
TYPES :
        CHBOX,
        END OF TY_DATA.

DATA : GI_DATA TYPE STANDARD TABLE OF TY_DATA WITH HEADER LINE.

DATA : GV_MAXROWS TYPE I,
      GV_BRAND LIKE ZSAP0000_CAR-BRAND.

*-----*
*Variable for Screen Command
*-----*

DATA : OK_CODE LIKE SY-UCOMM,
      SAVE_OK LIKE SY-UCOMM.
```

10. Modified module "USER_COMMAND_2000" like this bellow

Before

```
CASE SAVE_OK.
  WHEN 'BACK'.
    LEAVE TO SCREEN 0.
  WHEN 'EXIT'.
    LEAVE PROGRAM.
  WHEN 'EXECUTE'.
    MESSAGE 'Hello world' TYPE 'I'.
  WHEN OTHERS.

ENDCASE.
```

After

```
CASE SAVE_OK.
  WHEN 'BACK'.
    LEAVE TO SCREEN 0.
  WHEN 'EXIT'.
    LEAVE PROGRAM.
  WHEN 'EXECUTE'.
    PERFORM FM_COLLECT_DATA.
  WHEN OTHERS.

ENDCASE.
```

11. Create new subroutine at the end of source code

```
*-----*
FORM FM_COLLECT_DATA.
  DATA : LI_DATA LIKE STANDARD TABLE OF ZSAP0000_ST_CAR WITH HEADER LINE,
          LI_BRAND LIKE STANDARD TABLE OF ZSAP0000_RANGE_BRAND WITH HEADER LINE.

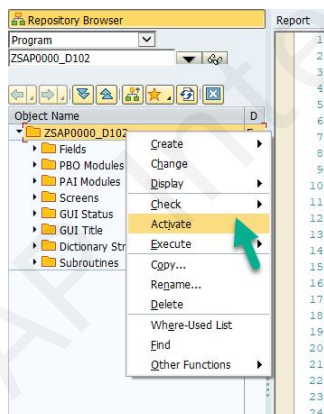
  IF GV_BRAND CS '*'.
    LI_BRAND-SIGN = 'I'.
    LI_BRAND-OPTION = 'CP'.
    LI_BRAND-LOW = GV_BRAND.
    LI_BRAND-HIGH = ''.
    APPEND LI_BRAND.
  ELSE.
    IF GV_BRAND NE ''.
      LI_BRAND-SIGN = 'I'.
      LI_BRAND-OPTION = 'EQ'.
      LI_BRAND-LOW = GV_BRAND.
      LI_BRAND-HIGH = ''.
      APPEND LI_BRAND.
    ENDIF.
  ENDIF.

  CALL FUNCTION 'ZFM_SAP0000_GET_CAR'
    EXPORTING
      MAXROWS = GV_MAXROWS
    TABLES
      R_BRAND = LI_BRAND
      OUTPUT = LI_DATA.

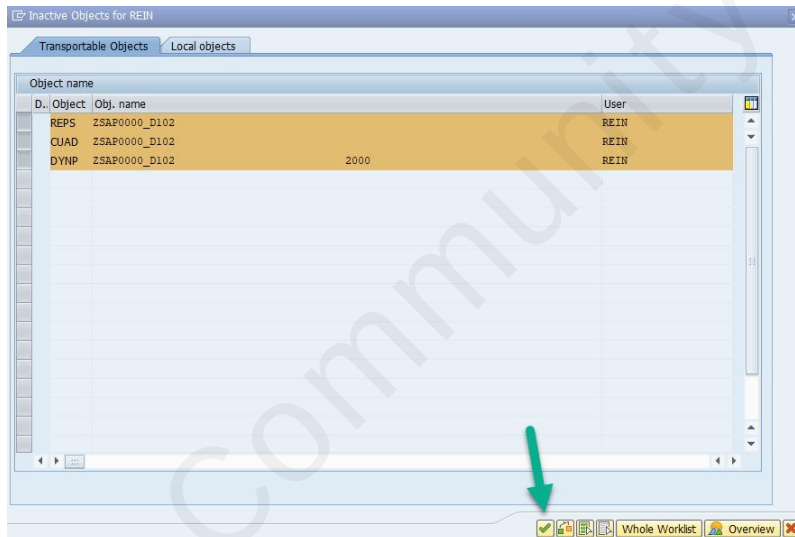
  "Copy result to main internal table
  REFRESH GI_DATA.
  LOOP AT LI_DATA.
    CLEAR GI_DATA.
    MOVE-CORRESPONDING LI_DATA TO GI_DATA.
    APPEND GI_DATA.
  ENDLOOP.
ENDFORM.                    "FM_COLLECT_DATA
```

12. Click "Display Object List"

13. Right click on top of Object Name and click "activate"



14. Click "continue"

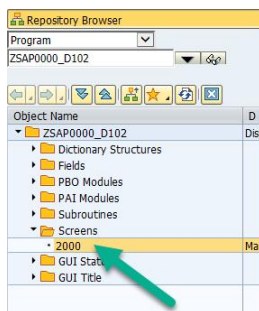


15. Enter command "/NSE38"

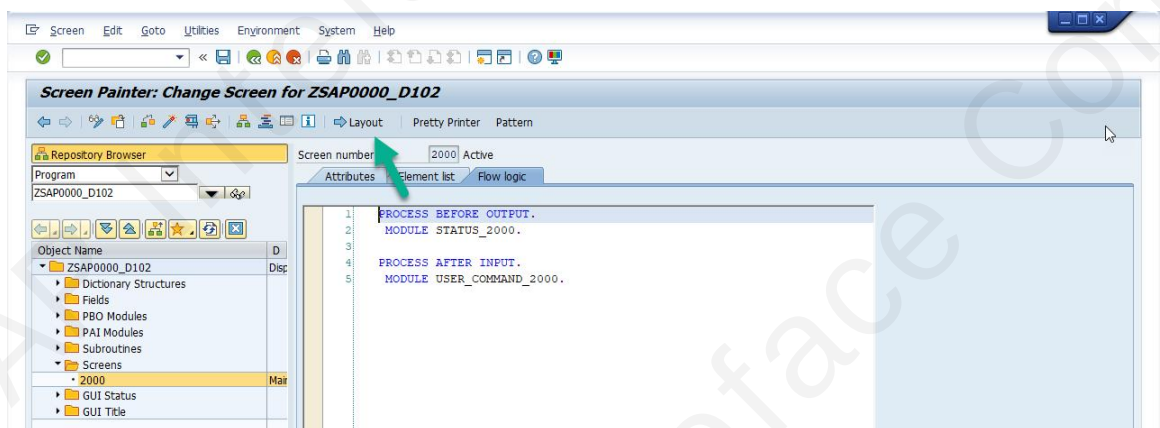
16. Click "Change" button

17. Click "Display Object List"

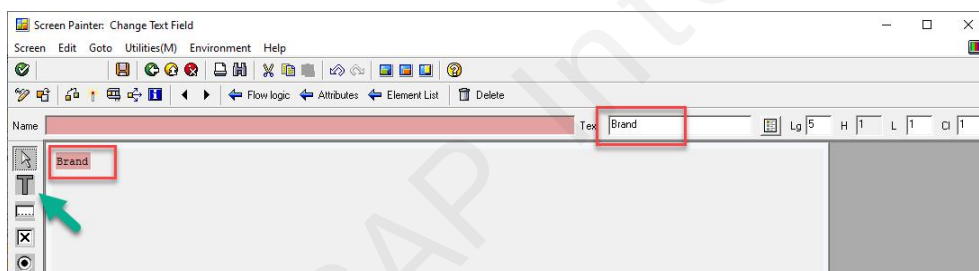
18. Expand "Screens" tree and double click screen "2000"



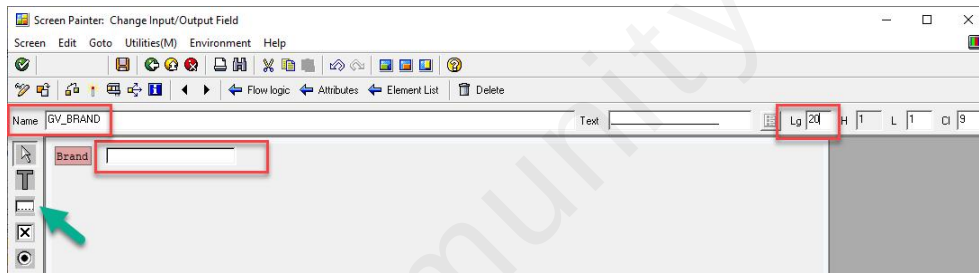
19. Click "Layout" and wait until "Screen Painter" showed



20. Create "Text Field" element and enter text "Brand"

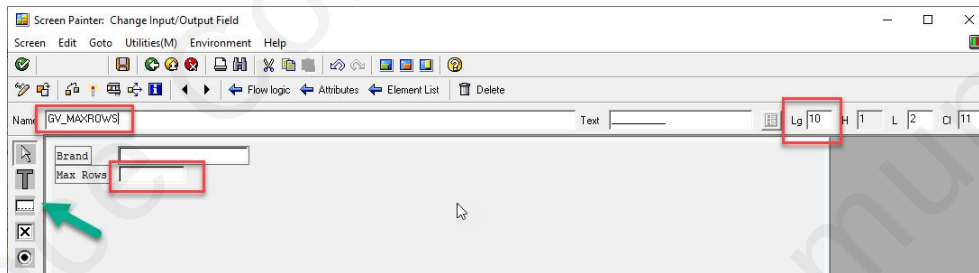


21. Create "Input/Output Field" and enter name "GV_BRAND"

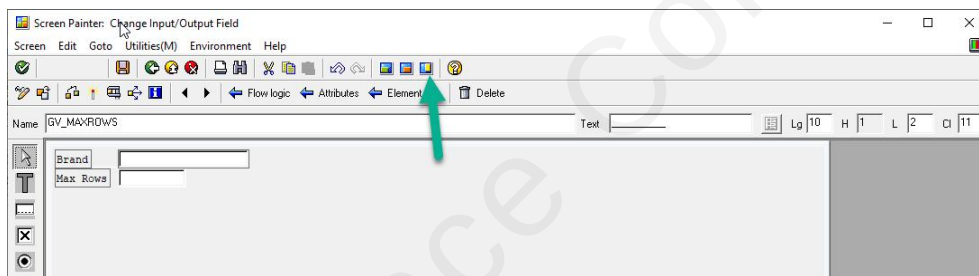


22. Create "Text Field" element and enter text "Max Rows"

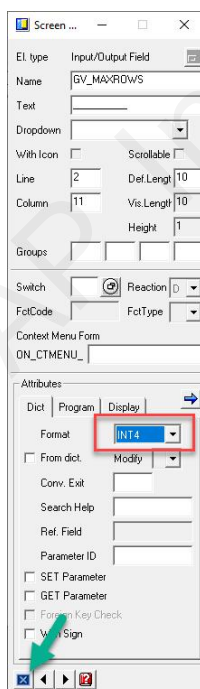
23. Create "Input/Output Field" and enter name "GV_MAXROWS"



24. Click "Attribute Windows"

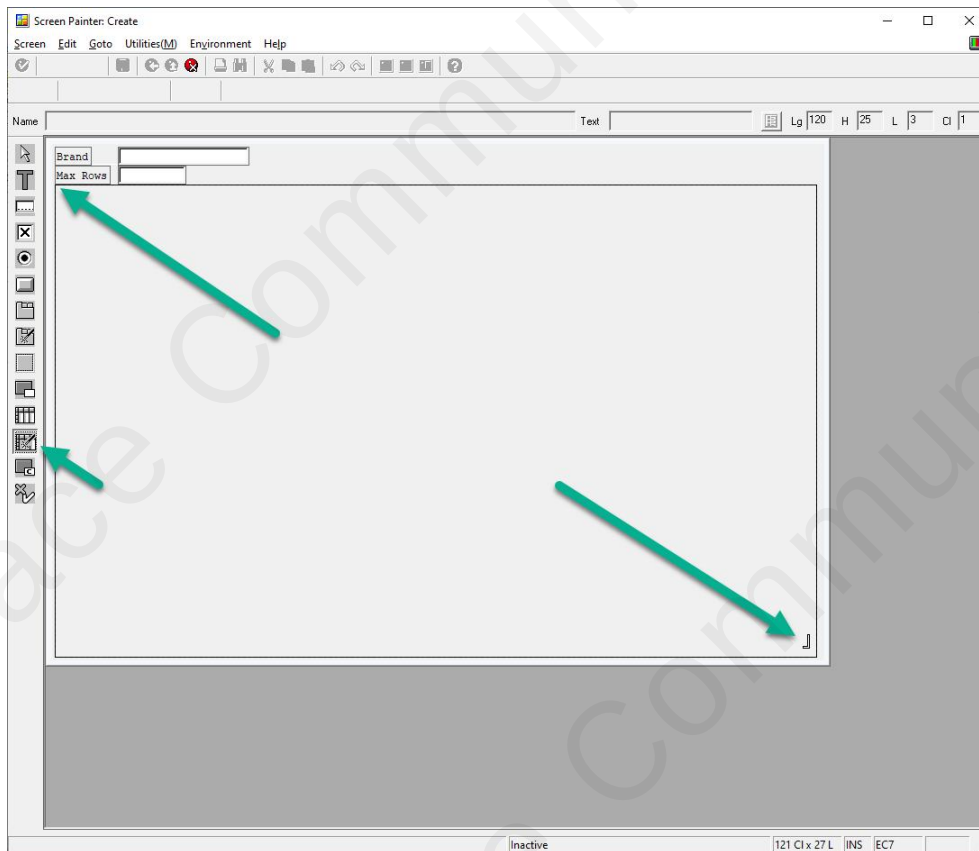


25. Set format "INT4" and click "Close"

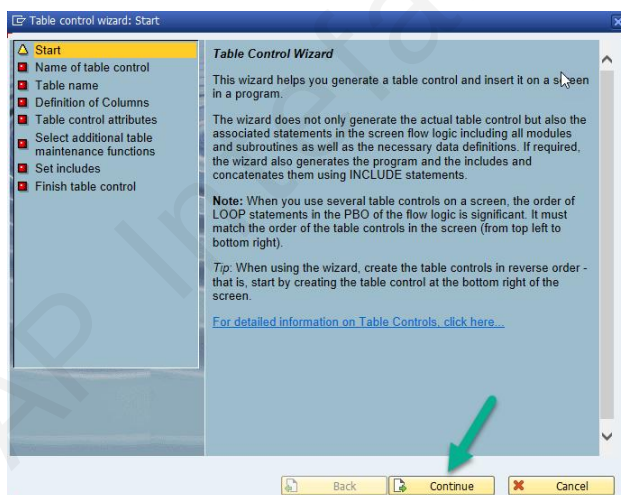


26. Click "Save" button

27. Click "Table Control (with wizard)" and click-drag on screen painter from top left corner to right bottom.



28. Click "Continue"



29. Enter table control name "TC_DATA" and click "continue"

The screenshot shows the 'Table control wizard: Name of table control' dialog box. The left sidebar has a tree view with 'Name of table control' selected. The main area contains instructions and input fields. The 'Program name' field is 'ZSAP0000_D102', the 'Screen number' is '2000', and the 'Name of table control' field is 'TC_DATA' (highlighted with a red box). A green arrow points to the 'Continue' button.

30. Select "Internal program table" radiobutton and enter name "GI_DATA", after that click "continue"

The screenshot shows the 'Table control wizard: Name of table' dialog box. The left sidebar has a tree view with 'Name of table' selected. The main area contains instructions and input fields. The 'Dictionary table' field is empty, the 'Internal program table' radiobutton is selected, and the 'GI_DATA' field is highlighted with a red box. The 'Table work area' field is empty, and the 'From active program' field is 'ZSAP0000_D102'. A green arrow points to the 'Continue' button.

31. Select fields (except "CHBOX") and click "continue"

The screenshot shows the 'Table control wizard: Column definition' dialog box. The left sidebar has a tree view with 'Definition of Columns' selected. The main area contains instructions and a list of fields. The 'Definition of Columns' section is highlighted with a red box, showing a list of fields including 'BRAND', 'CAR_TYPE', 'RELEASE_YEAR', 'PRICE', 'WAERS', and 'CHBOX'. A green arrow points to the 'Continue' button.

32. Set "Check" of "Line Selection col." and enter "CHBOX", after that click "continue"

Table control wizard: Table control attributes

Here you can specify several essential attributes which control how the table control looks like and how it works.

It is possible to create the table control displaying data only, or to allow entries for all column fields.

You can choose whether the columns should have headings.

If you want to include a selection column, you can specify if you want to allow single-line or multiple-line selection.

You can then maintain additional attributes for the table control using the Screen Painter.

Input/output attributes: ☐ Output only ☐ Input control

Columns: ☒ With column headers

Line selectability: ☒ Line selection col. ☐ Selection col. fld

Selection col. fld: CHBOX

☐ Single ☐ Multiple

Back Continue Cancel

33. Set "Scroll" and click "continue"

Table control wizard: Table maintenance functions

In this step, you can add several pushbuttons to the table control which initiate standard functions for table maintenance.

The following functions are provided, if applicable:

- Scroll: Four scroll buttons below the table control that can be used to scroll the table display.
- Insert/delete line: Two buttons for inserting an empty line before the current line and for deleting selected lines. This function can only be used if the table control can be selected and if entries are allowed.

Note: 'SAPSTED' or 'HASHED' internal tables can cause the code.

☒ Scroll ☐ Insert/delete line ☐ Select/deselect all

Back Continue Cancel

34. Set all value "ZSAP0000_D102_T01" and click "continue"

Table control wizard: Set includes

Here you must specify the names of program includes in which the data declarations, the input/output modules and the subroutines are to be created.

- Include for the data definitions
- Include for PBO modules
- Include for PAI modules
- Include for subroutines

Includes that do not yet exist are created when the table control is

Include for data: ZSAP0000_D102_T01

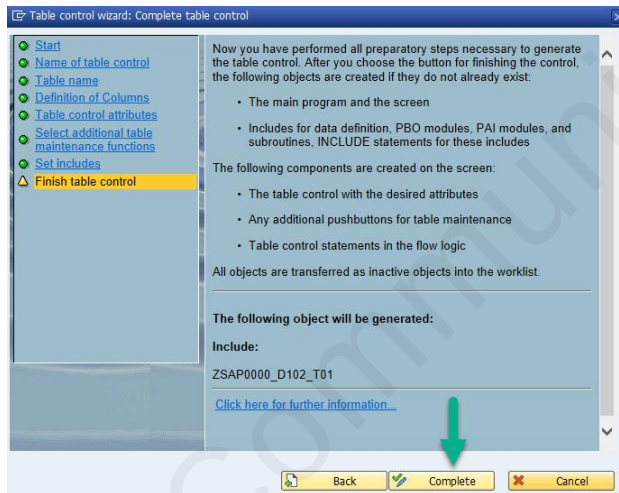
Include for PBO Modules: ZSAP0000_D102_T01

Include for PAI Modules: ZSAP0000_D102_T01

Include for subprograms: ZSAP0000_D102_T01

Back Continue Cancel

35. Click "Complete"

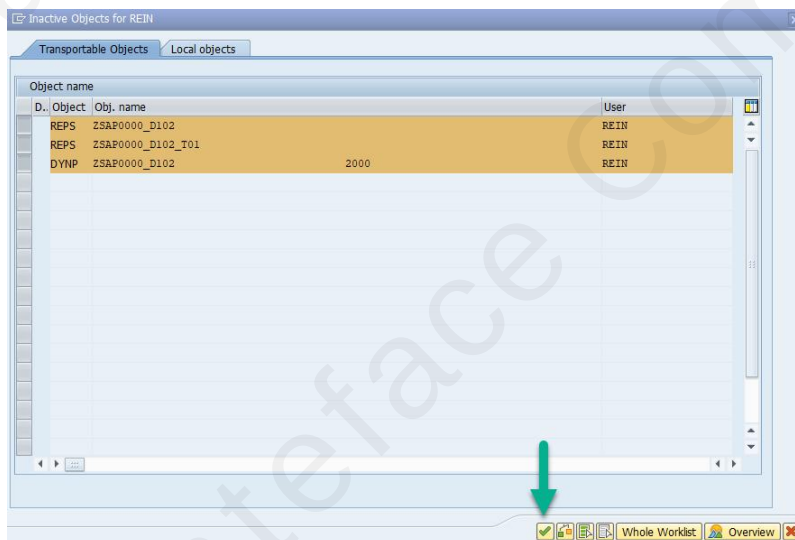


36. Click "Save" and "Close"

37. Double click on top of Object Name

38. Right click on top of Object Name and click "activate"

39. Click "Continue"



40. Done