**Dictionary Objects :**

**Package : ZPAVAN\_PRACTICE\_924**

**Database Tables**

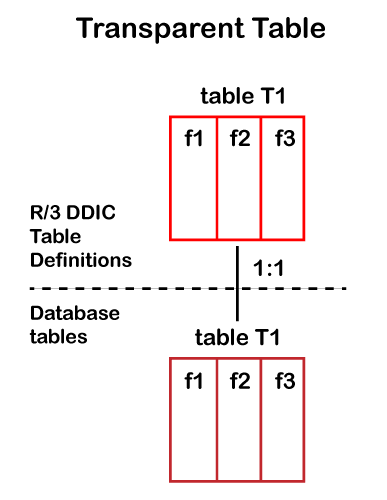
* **Transparent Tables**
* **Pooled Tables**
* **Cluster Tables**

1. Transparent Table

ABAP transparent table contains the application data that represents the **master data** or **transaction data** used by an application.

In transparent tables, the database table has the same name and the same number of fields with field names as the data dictionary table.

The transparent table shows the **one-to-one relationship** with the table definition in the SAP database



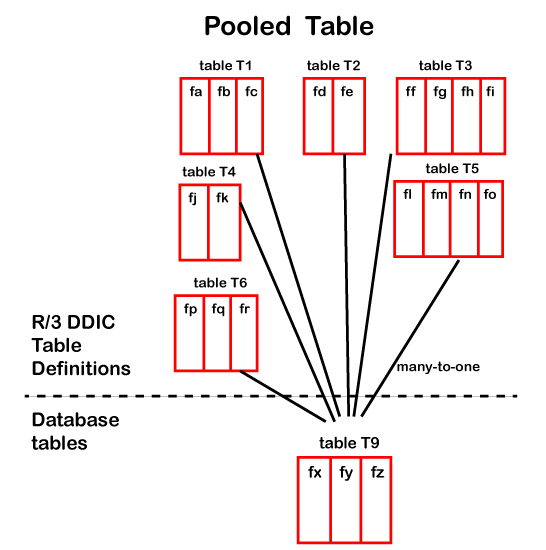
### 2. Pooled Tables

The pooled table in ABAP shows the **many-to-one** relationship with the table definition in the database, which means for a single table defined in the SAP database, there are various tables in the ABAP dictionary.

The names of the tables stored in the Dictionary and the database must be different.

The SAP database stores all the Pooled tables in a single table, which is known as **a table pool.**

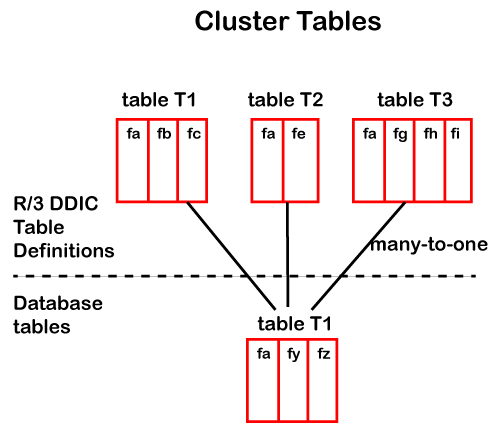
The pooled tables in the data dictionary may or may not have a common primary key field.



### 3. Cluster Tables:

Cluster tables are similar to the Pooled tables, as they also show the **many-to-one** relationship with the table definition in the SAP database. All the cluster tables are stored in a single table in the database, and that table is known as **a table cluster.**

The tables in the data dictionary must have **at least one primary key field in common**, and the table is usually accessed simultaneously.



# **SAP ABAP View**

**Views** In the language of relational databases, a view is a virtual table, that is, a table that is not actually physically stored, instead being derived from one or more other tables.

-database view:1 or more table can join and can create a view \* primary key relationship b/w tables

-projection view: only 1 table and its fields will be projected and can use conditions to filter

-maintenance view:2 or more tables can maintain like TMG to fill table contents automatically when one table adds data the dependent table also add that data in that table as it is linked together by foreign key relationship \* cardinality b/w table should be 1:1 ,mandt should be used

\*instead of tmg for tables to insert data of same fields in foreign table by creating a view with those multiple table so all tables will be inserted with that contents with corresponding fields in that table

-help view: 2 or more tables together can create this view and for creating search helps will be used as data source for those search helps

Delivery class:

Delivery class for controlling data transport of tables for

installation, upgrade, and client copy, and for transporting between

customer systems. The delivery class is also used in extended table

maintenance.

the Data in the table to be transported to clients in read,write modes based in its type of delivery class

-Application table: table with master data, transactional data

-Customer table: Data only maintained by Customer

-table is created to store temporary data

-customer data only insertion and no updates

Data Class: It says about which portion (area) in database (disk) needs to store the table of contents, once activation is done the table is loaded into disk automatically so that that portion can be maintained by a single table space so it has restrictions based on its types of data classes

Simply its for the data in table that to be accessed in database so in that particular area to be chosen for its future accessing

-master data: Rarely changed like customer address

-Transactional data: Frequently Changed like item details

-Organizational data: its about system data as it created and setup with some values and rarely changed and it’s a customized data ,can be changed but it rarely changes as its organization on like country code in a customer data

Size Category: for a table in database there will be initial space for all types of tables and it can be extended further if data space is exceeded of particular data so the size to be mentioned like 100 /1000

**Data Elements**

**Srch Helps**

-elementary search help:by adding database table in selection method we can make a list of fields that to be viewed as a help list to choose possible value for your input

Parameters;

There will be 2 positions that fields be listed one to take input as value to get the list from database(screen position) And 1 to display the loaded data as a list (list position)

Import-when passing the input so that related data be loaded and displays as a list in lit potion

Export-after list is displayed we will select a record in that list for our input help that we are using currently for this search help and that particular field be exported to the field that we expected a value

Sdi – to restrict the fields in screen position not to take any values as input to fetch from database as its import parameter is unselected and default value should be given for it

Modify- to modify the data elements if we use any field the linked data element will be loaded to change it we will use modify to replace other data element

-collective search help-multiple elementary helps together forms this collective help as 2 or more table or search helps will be used as a data source for the list and these 2 or more table will be listed according to its value in different tabs of its list in each tab

Syntax:

PARAMETERS vbeln type zvbeln\_924 MATCHCODE OBJECT zvbeln\_924.

**Data types**

**Type groups:** it’s a type pool managed in data dictionary and its purpose is to visible globally and be uses these data types ,constants, macros by various data objects

**Lock objects:**

For Synchronous access to data objects to multiple users for a same data object ,to protect the data object in multiple users are modifying data simultaneously

If lock definition is called it will trigger a function modules

Enqueue\_<lock\_objectname> =>funtiuon module used to setup lock by se37/se38 for setting up lock

Dequeue\_<lock\_objectname> => for releasing a lock

Sm12 is used to check whether data object is in lock or not

Modes of lock

Read lock: if this lock is called then multiple users can just read the object and mutual lock is called by a shared user for this lock and it will be accepted to read the object

Write lock: Only one user at same transaction can be write the data object by this lock no other shared locks will be accepted

Mutual exclusive lock(shared lock): By multiple users with certain locks can be performed in this lock

Only once this lock is called per transaction

**Cds views - Data Definitions**

**Field Symbols:**

Field Symbols are a place holders to data objects means it allocates memory to the data objects to which it was assigned to place some memory. these symbols are like a pointers which stay at the memory as tightly linked to data objects as if field symbols holds the data objects values that is assigned and if field symbols changes that value then automatically it reflects in data objects and updates it

2 types of field symbol

-Typed field symbols: field symbols should be declared with the same type of data object to get assignment to it of elementary data types or complex ones(structues,work ares)

-Generic field symbols: it will be declared by any type like with no type and it will be assigned to data object of any type , For Dynamic programming this will be used as not to declare a particular type statically it takes at run time and key word for that type is

FIELD-SYMBOLS <FS> TYPE ANY ,

FIELD-SYMBOLS <FS> TYPE ANY TABLE

LOOP AT gt\_data\_table ASSIGNING <fs\_data>.  
 gv\_dynamic\_data = gv\_dynamic\_data + <fs\_data>.  
ENDLOOP.

**Reports => events**

Reports are used to process the business data which are already stored to build the business applications and process those data as per client requirements

Types: Classical reports, Interactive reports, Abap list viewer reports, database logic reports, so on

Events: these events are used in report to control the flow of code by the developers and to improve functionality of reports with these events

Types:

Load-of-program: to load a program into memory for execution and for declarations this event used

Initialization: for Initializing screen values and default values

Start-of-selection: to start the actual logic by taking inputs from user ,Business logic will be here

Top-of-page: heading to every page

End-of-page: footer to every page , reservation be maintained at end Line-count 35(2)

At-selection-screen: Validates user input

At-selection-screen-on field: Validates that particular field in selection screen

At-selection-screen on value request: used to provide f4 help to a field

At-selection-screen on help request: used to provide f1 help to a field

At-selection-screen output : for Dynamic changes of screen in selection screen-manipulations on screen , this event is called actually at in screen event of PBO

To make screen element like active, input in enable or disable to radio button or checkboxes this will be used.

At-line-selection : at selection screen user can interact to get more details of particular record in secondary list like we can hide the details in primary list of employees and if user need to know further about that particular employee then by selecting that record they can view more details

At user-command: it provides the function keys which are there in the screen as it holds those info so based on that we can code in this command like if sy-ucomm = ’back’ then

At personal function: it do use of personal function keys to create buttons or double click, back by the user by clicking or triggering a button

These function code will be used by setting up GUI status by creating pushbuttons, titles, status in menu painter with screen positions like menu bar ->function key->application tool bar

Top-of-page during line selection: this event is used when user interacts in selection screen by interactive events then in secondary list it will give heading in that list.

Exiting events – stop it stops current event, exit in loops it stops current event, check- based on condition

**Modularization Techniques:**

-Include: These will be used to optimize reports with high maintainability and reusability and its not executable programs and the type is ‘I’ and this type will be fetched with the help of F1 help ,mainly declarations will be used here

-Subroutines: same as include these are also for maintainability and reusability purpose and this doesn’t have any separate interface for creation and implementation with keywords as perform, form end form we can create these sub routines. we can create in same main program or else can be created in separate include

These are 2 types :

Internal sub routines : the one that was defined and called in same program

External sub routines : the subroutine I created in one program and it will be used in other program with a syntax like \* perform sum in program zreport2 using input 1 input 2 if found (not to give runtime exception if the base program changes the subroutine name or deletes it in that case this keyword is used to give empty values without error)

Or \* perform sum(zreport2) using input1 input2 if found.

Subroutines can be defined with parameters and without parameters

It has actual parameters and formal parameters names can vary

Syntax : perform sum tables/using/changing

Tables : to define internal tables so it can carry from main program to subroutine or else in vice versa so that internal table can be modified its contents in processing .

Using: this type of parameter is to use those parameters based on the type of passing parameters Changing: this can be useful to change the parameters like use and store the result in parameter .like changing that parameter

Type of calling subroutine

Pass by value: with value keyword with parameter this can be used so it purpose is to make a copies of formal parameter to the actual parameter so that actual parameter doesn’t changes its values

Drawback is for internal tables its waste of memory to these copies

So extra memory is created for the pass by value parameters

Pass by reference: it will pass the actual memory address of parameters so it directly changes the actual parameters values without any extra memory by default this mechanism will be done if we won’t use that value keyword with parameters

-Function Modules: same as subroutines but it has a interface to create function module in se37 and this can be executed separately also not only in the program but also individually and major difference b/w subroutine is the exception handling, it should creates in a function group and it must be in activated state when the function group creates it default creates 2 include

1 ztop include : it says under this group we can create multiple function modules so that if we want to write a code that should reflect in all the function modules at once which are there in same group

So that logic will be used in this include

2 zu\_\_\_xx include : this include counts the number of function modules are residing in that particular group initial it has xx this symbols later it increments so a function group must stores only 99 function modules only

Function module contains tabs like :

Attributes: Generic information like user details that created by ,date on, type of function module like regular function module or remote enabled module

Importing: input

Exporting: output

Changing: change and return (modify)

Tables:

Exceptions:

Source code:

Select options will be passed as a exporting parameter from a report as a table type indicating quare braces [ ] and at function module select option will be imported as a type of table type it should be declared as this type by defining sign, option, low, high of structure fields in it.

**RFC:**

* It’s a cross application remote function module to call a function module
* Its an extension to a function module to fetch data and to call the function module from

a external system from remote region

-external systemz may be sap or non sap system

From external it calls rfc that residing in some other sap system it will fetch and gives that data related to that own data in that particular system that this fm presents rather than the in calling system

-pre requisites to create rfc

\*function module -> attributes-> processing type should be -> remote enabled module

\*all parameters = pass by value

Syntax = that sap/non sap system calls this function module

Call function ’function module name’ **destination ‘system name’**

Importing-----------

Exporting----------

Tables------.

* Connection should be there when any system that needs to connect with sap system so for that basis team do that from sm59 tcode in abap connectors they will add sap system and this will be done in which the system that calls this rfc function module

Sap/non sap system(in report or any object it writes code like call function name destination sytem)---🡪 connection should be there that which we need to call in our system RFC CALL ---🡪 system that be called in function module and executed data and fetch in that system and pass that data back to the calling system.

Differences between BAPI and RFC

* Both are cross applications function calls
* BAPI = Business object + RFC
* BAPI is like an oops concept object that calls function module
* BAPI holds the RFC when business object will be called it holds, attributes, interfaces, rfc and in that business object it calls rfc
* So, BAPI is not a standalone RFC to call function module it needs object
* But RFC is standalone and calls function module and it not relates to any oops concepts it directly calls function module

**BAPI:**

Business application programming interface

-cross applications - BAPI, IDOC, RFC

- it’s a standard programming interface so any external system(using any programming language like (sap, java , C++ ,visual studio ) transfer data from that system to sap system via a interface to deal with sap business process in sap system from sap/non sap system it receives.

Generally sap technical objects will be treated into certain objects like RICEFW

Reports, Interfaces, Conversions, Enhancements, Forms, Workflows

-BAPIs are under Interfaces as these are behave as an interface to external system to access the sap system as non sap to sap or sap to sap systems

-Actually the external systems wants to access the business process through this BAPIs itself to reach sap systems, business processes means it has a business tasks with start and end process with its requirement and those are sales ,fico, mm

-to create, delete, change those entries in business process using bapi external systems uses this interface based on its standard bapi available in system

-BAPI = business object + RFC

-this bapi objects will be available in Bapi repository ( tcode BAPI )

PREREQUISITES TO CREATE BAPI

* Create a function mudule
* Name should be – z(custom)BAPI\_BusinessObject\_method
* Parameters type should be – z(custom)BAPI\_name – always use collective fields as structure and these individual fields can be accessed using dot operator if it required at any stage even here also by defining those fields in structure and use these parameters as zbapiemp.eid,zbapiemp.ename so its should use proper datatype for that field
* It should return a parameter in exporting -return and type bapiret2 so it contains some status messages from bapi to external systems as a response from sap system like success, message as item created, or errors
* All parameters be passed as pass by value
* In attributes it should be chosen as remote enabled module in processing type
* In source code write the logic for the existence of this bapi and fill the return variable to pass to external system by creating message class globally and check that return datatype structure and fill those values to return
* Release the function module so that it can be bind to business object
* Function module->release
* Now RFC is created
* Go to tcode swo1 to create business object
* Super type – if any super type objects are there we can use to get inherited or else leave it
* At object type enter object name as per function module as zobjectName – used in function module – ’Internal object reference’ in sap system
* Object name – give object name without ‘’z’’ – external reference this be identified with this name by external systems to access bapi
* Desc,name will be given accordingly
* Program- provide a valid program name with zprefix while creating bapi it will be filled with a code based on the function module and we can enhance the code based on requirement
* Application type – cross application
* Object will be defined – it has interfaces, key fields, attribute, events, methods – indicates similar to oops concepts with object the method and event be called
* At methods portion try to add the method which was declared in function module by using utilities->API methods->add method-> give function module name and under method name provide valid name as per function module(rfc) as method name for BAPI
* Now the method is binded with the business object as business object + rfc
* This rfc can be visible at by clicking method->abap rfc name will be there
* By clicking program button the auto generated program will be available
* Object will be created and maintained into 4 status
* Modelled – the objects will be defined
* Implemented – defined and implemented by adding rfc and attributes ,methods of that object and its for internal testing and can’t used by external as bapi
* Released – it will be used for external use so that this bapi object will be available in bapi explorer using tcode or bapi objects button and this release be done at each component level and entire object and both should be done to get full functionality because for all requirements the entire methods and components will be given to external system so with certain restriction it will be released
* Obsolute – it can be used
* Status can be changed via select the component or object ->edit->change release status-object type or object component-> to release
* Tick mark shows it is release or not
* After that click on generate option to be available at BAPI explorer as business objects and its will be with name external name that was provide at swo1 tcode and it is case sensitive

Tables to identify the existing business objects and methods for a bapi

-TOJTB-Basic data of business object

-SWOTLV- Detailed data of business object

**IDOC**

* IDOC is one of the cross application interface that exchange the data from external system to sap system and act as a interface between 2 systems
* IDOC is a Intermediate Document and it has a unique number and with that it will be identified as idoc = 3443445 -> stores the idoc type, message type, structure , data and all

Suppose 101 -> eid = 100 ,ename = abc

102-> eid = 200, ename = xcv

* It’s a container which transfers data from external system to sap system or vice versa
* IDOC process – inbound process and outbound process
* Inbound process: the data that flows from external system to sap system
* Outbound process: the data that flows from sap system to external system
* When IDoc is generated then it will be appear in 3 records as

This info will be seen in TCode : we02 or we05

* + Control record : it contains info about IDoc type( Basic type ), message type , port info, partner information, direction of data flow = inbound (indicates by 2)or outbound (indicates by 1) and also can seen this info in a table of “EDIDC”
  + Data record : it contains the actual data of that IDoc creation that to be transferred in the form of segments and these will be in multiple columns which stores data for that columns like eid = 100 ename = abc. This info can also seen in table “EDID4”
  + Status record: It contains info about the status of that IDoc as it is in ‘Success’ or ‘Error’ state and each IDoc will undergoes different status while its transferring data and the status info about the data is like from “1 to 42 says outbound direction” and “from 50 to 75 says inbound direction“ . this info can also seen in table “EDIDS”
* Data exchange method / Technologies are ALE and EDI
  + ALE: Application link enabling
    - It’s a sap proprietary technique as this method was introduces in sap itself for the usage
    - This method will transfers the data from only sap to sap systems and the format of the data won’t be changed when transferring to other systems this format means the no of records or else if the sales data is moved from one sap to other sap system sales data then format doesn’t changed or else sales data moved to purchase data then the format is changed.
    - This method is with in business range as the data will flows in sap only
  + EDI: Electronic Data interchange
    - This term and method is there before 1972 of sap existence
    - The data that transfers from non sap to sap or vice versa
    - The format may change while moving the IDoc like sales data will be converted to purchase data in other system
    - This is across the business environment
* Main terminologies to consider when creating IDoc are
  + IDoc type or Basic Type: Tcode is we30
    - It will be created to maintain the structure of IDoc Data as with this structure definition the data will be transferred in segments as found at data records. this IDoc info we can actually seen at control records and structure is like it will create the unique number with that it will create structure as eid ,ename
  + Message type : Tcode is we81
    - This will be created to maintain the meaning or like a valid description to the IDoc that the data is to transfer and it is like “Employee data transfer” or “Sales Data transfer” based on the IDoc we will create this message type
    - After creating IDoc we will see this info generally at control record
  + Segments: Tcode we31
    - These segments are the basic elements to the IDoc type
    - This will store the actual data to that IDoc type that was defined as a structure in IDoc type
    - This will be defined to IDoc type
    - Like it stores eid = 10 ename = ‘abc’
    - When opening the IDoc these segments will be displayed in those segments the actual data will be found

IDoc needs to create a interface to transfer data with all these message types, IDoc types, segments

Tcode : We82 to map message type with IDoc type

Tcode : BDBG to create IDoc Interface as a data source of BAPI - by entering BAPI name here it will automatically create IDoc type, message type, Segments , in bound and outbound function modules

– in that BAPI rfc logic will be there, and binding the message type with IDoc type

Table : TBDBE – it displays BAPI – ALE IDoc interface components

When to create an interface we need to know few things like any sender or receiver should connect to sap system to send its requests via a communication channel to a fixed ports and these ports are created to various requests that classified a http, https ,ftp requests so connection should be established

* Port : tcode is we21

it’s a technical link that to be maintained with receiver system in order to get requests from that system and this connection will be only maintained to the destination systems where its been setup at sm59 tcode for connection rfc configuration so for that system port is created via system auto generated ports or we can give our naming convention to create that port

* + Those ports are also be created in various ways based in its request as
    - Trfc – transactional rfc
    - File – in order to get a file from external and will place that file in al11 tcode to connect with sap system using BAPI
    - And various multiple ways
* Partner profile : tcode is we20
  + For every business transaction its common to maintain partner profile about the business partner details where the organization is transferring or receiving info
  + There are multiple partner are available in sap like bank, business partner, customer ,vendor ,finance for these business purpose will be created multiple profile to each of its access
  + We can create a profile by selecting it and providing partner number which is existing in current database and the type as organization or finance based on requirement.
  + In the agent tab to be given the username that handle the transactions in business flow can be selected our own or someone else they will be notified if anything went wrong
  + Status will be active if the business transfers are in existence or else inactive state.
* Process codes : it will process the IDoc to start its processing
  + This process code will be done in 2 way
    - Inbound processing code: Tcode is we42

It will process the IDoc and send this IDoc to application means when IDoc binds with message type it will generate a function module in that Bapi resides with specific tasks – this processing says inbound process as processing IDoc it will transfer data through IDoc to application and to sap

* + - Outbound processing code : Tcode is we41

it will take data from application and starts IDoc processing as outbound process

* In this process code generally given as BAPI it will take the function module by entering the message type and methods in inbound/outbound processing
* We57 message type,idoc type,function module of idoc interface

For transferring non-sap data to sap data then

Data migration tools (lsmw) use bapi with idoc ->idoc use function module in process codes to trigger function module that contains bapi->bapi business object that calls the method->rfc->database table(sap)

**LSMW:** Legacy system migration workbench

Tcode is LSMW

It asks for project, subproject, method give as per custom names as your own

It redirects into no of steps to follow to complete the migration of data from non-sap to sap system

Like define attribute in this you have to pass message type by selecting bapi as data source if the non-sap is local file then you have to choose files path also in some other steps

Define structure

Mapping local declared structure to segment structure

define associations

all the steps will be there itself and follow them accordingly

important at project perspective

We19 - test tool for debugging as testing

Not directly involving non-sap system with this tool we can pass sample values as input to check idoc, bapi functionalities in debugging mode also

new idoc is generated as testing but data will be reflected as per logic

only one idoc is processed at a time

not allowed in production system as new idocs generated as customers may confuse

only in quality it will be encouraged

Bd87 - reprocess IDocs if there is any error IDocs are found like duplicate data are any error

So the contents has to edit then use this tcode and change the data records and save and reprocess those idoc in same tcode new idoc process will be carried out and new status you can find as 53 states application process completes as success message

same idoc number will be there as it does not change IDoc number

multiple IDocs will be reprocessed

**for example:**

if you want to transfer the local file (NON-SAP) system to SAP system as a remote location with the help of bapis ,idoc it will be done as bapi is a interface that can call function module(rfc) and rfc is to execute function module in remote location so bapi is a interface it will call rfc from remote location with the help of a business object and in function module you will write logic to store data in sap like table or a directory like al11 so for that you can do like this in this scenario

notepad file in desktop-> as NON-sap system

LSMW creation which has set of predefined steps by setting up new project name you will select bapi so in bapi you have rfc ,message types as it relates to idoc and method name

Lsmw calls the function module that automatically generates a function module by bapi for itself based on our rfc and our function module logic will be found in this new function module

And pass the directory path in subsequent steps in lsmw steps to give input a local file to migrate it into sap system

Declare a local structure in lsmw for input file and match this input file with the segment type in idoc which was already cr eated in idoc with certain type to pass into table

Map and associate these 2 structure and complete the process

Then after this lsmw it will process the idoc by using process code in tcode we42(inbound process code ) or we41 (outbound process code ) here we will use inbound process code

In this process code we will have the function module that was already generated when the bapi creation with the business object creation as in this function module it has rfc logic to store file data into table

This process code invokes the processing of idocs with the help partner profile as it holds the port info where the process is started for inbound or outbound in this we have to choose inbound and pass the inbound info like message types ,idoc types and methods and apart to it in partner profile we have give some info like agent name as our username or some other and group like organization ,logical system and for port setup we have to give the info about the system as file or trfc but our case is file select file and setup those info and choose the system to establish the port link to the system of some other system that are added into connection rfc configuration tcode as se49 by basis team those system is added

By setting up partner profile->port linking

Use tcode as bdbg it will automatically creates the interface between idoc with bapi

As by taking bapi info as external system and object,method name it will setup the idoc type, segment info,message type and it will binds the idoc type and segments in it

You can confirm that binding with the help of tcode we82

Before you have to create the idoc elements like idoc type, message type, segments with that naming conventions

After that use that tcode bdbg so it will bind all together

Create bapi which is business object + rfc with the steps are given above and bind this business object with idoc-> create rfc = function module = create table that to be inserted to this table as the records that flow through this interface and migrated to this sap system from non-sap system as local file from desktop

**DATA MIGRATION TOOLS**

To migrate the business data from non-sap(legacy) to sap or vice versa there are some techniques to do that process: BDC, LSMW, BAPI

Steps for data migration

Data extraction: extracting data from non-sap system to a plain file like into notepad or excel

Data conversion: converting that data in file into compatible to the types of fields in sap system tables after getting those data as extraction process update the values according to sap system

Data importing : impot those data from file to sap system using migrating techniques

Data verifying : check the data is uploaded from legacy to sap system correctly or not

Batch input techniques

BDC – Batch data communication: transfer data from legacy to sap system

methods to implement this technique

call transaction method

session method

direct input method

call transaction method:

steps :

* Record the steps how to transport the data from non-sap as a file from desktop to the sap system into table by taking one sample record
* For example if you want to create the material data from file into mara table then we have tcode as MM01 to create those entries so by taking one sample record and start recording and create the entry and save the recording into a program and this one simple recording will be iterated into millions of times as per the file data
* Be aware if certain things while recording
  + Do not use f4 or f1 help
  + Do not enter wrong values that leads to error so it will also be recorded and perform for every action
  + Make a rehearsal data entry before recording
* For that recording go to tcode SHDB in that create a recording as per name and start recording and enter the tcode from which transaction you want to start recording
* Do the creation part of material data and after creating save the recording and go back and use program button to transfer the recording to a program and choose the radio button as transfer to program by giving a valid name
* Edit the program as per your logic either in call transaction or session method and do not forgot to maintain a file in desktop as legacy data to upload the data

Call transaction and session methods both are same up to the recording part and transfer into the program but from syntax implementation were bit different

Call transaction syntax

Call transaction ‘MM01’ using itable mode ‘A’ or ‘N’ or ‘E’ update ‘A’ or ‘S’ or ‘L’ message into lt\_messtab

Mode A = all screens it will show all the steps that we did previously in recording while creating material for every data

Mode N = No screen it won’t show any screen and it directly process and gives output or creates entries

Mode E = it’s a error screen until unless there is any error it won’t show any screen while creating any error at any screen then at that step it shows the screen with error

Update A = Asynchronous as it won’t wait for receiver update it directly sends all the data to receiver in asynchronously

Update S = Synchronous as it will wait for receiver update as it received the data or not after that it will send the next record until then it will wait

Update L = local update there will be no sender and no receiver in the same system itself as locally it will store and transfer data

It\_mssgtab = internal table that stores the status of the each record as success or error and if any dynamic field values like & in global it will stores that info

Session method :

Same as call transaction but in this the logs that stores the records status info will be available at bdc log status tcode is sm35 and also by running the program it just creates the session of the job it won’t process we should manually process the job group that we created by running the program

Steps

Will Use function module for this session method instead of call transaction after recording ,

Call function ‘bdc\_open\_group’: will open the job for session group creation , it just creates the job as per the name that we pass in this function module parameter

Call function ‘bdc\_insert’

Call function ‘bdc\_close\_group”

Difference between bdc call transaction and session methods.

* Call transaction in the program itself all the processing will be done.
* Session method in program the session group is created by taking file into program and need to process the session in sm35 tcode to get create the material or the program processing based on the logic inside program
* Call transaction method logs means the message status in a way like data transferred successfully or error and those messages will be maintained in mssgtab internal table in the syntax itself.
* Session method the log details will be maintained in sm35 bdc log status tcode in that the log info will be available when we check the keep parameter in bdc\_insert function module while creating logic in bdc session program .
* Call transaction in speed compare to session as call transaction has the update methods of synchronous, asynchronous, local update
* But in session method by default, it has only synchronous updates only so it is bit slow
* In session method the detail log info will be maintained in sm35 but call transaction won’t do that
* There are certain parameters in session group like hold date does before that mentioned date it won’t allow the user to process the session group and keep parameter to maintain session details in sm35 otherwise the session will not there but the logs will be available in logs
* Error handling will be done in call transaction in the syntax of mssgtab internal table and in session method it will be maintained in log info of sm35 tcode

LSMW: Legacy system migration workbench

Tcode is LSMW

This is also same as bdc as to transfer the data from non-sap (legacy) data to sap system

This is a data workbench tool is available at workbench with a tcode

Mainly this type of tool will be used by the functional sap consultants as it doesn’t need much programming interface like BDC migration as there we will build code after recording according to our requirements but for this lsmw the functional sap people use this for the data like “master data”

We can upload or download the lsmw steps says about the migration with all the attribute’s structure, files and all the mapping from one system to other system using import and export options

Lsmw methods

We will use 4 methods to transfer the data from non-sap to sap:

* Direct input method
* Bdc input processing
* Bapi
* idoc

3 things has to provide to create lsmw

Project : like a main project or else it will be a main folder it holds multiple subprojects

Subproject : its subproject under the project and it will holds multiple objects/methods

Object: object is like a business object in bapi or method that will be called for an action from subproject like create ,change, update

Eg: there is a project having master data and client data for migration then we will consider the naming convention to follow the below 3 names to provide when creating lsmw project

project: data migration project: data migration

Sub project: material master data/upload Sub project: material client data /download

Object : mat\_create /change/display Object : mat\_create/change/display

After that there predefined steps available in lsmw tcode to do certain processing like

Define attribute = contains the lsmw methods = direct input method, batch input process,bapi,idoc

`based on this the next steps will be filtered according to that

We can choose batch input processing if we have to record and made this to follow up the same thing so we can make a recording like bdc technique and do the same steps with lesser code involve

Define structure

Assign attribute, structure

Assign file

Map and give relation to source structure with the upload structure

Read data : have to read from file to application layer (sap) has file extension with .read

Convert data : read data will be converted to sap source compatible and stores In application erver like “AL11” tcode like bdc after uploading file it will store the data in internal table here it will store that data file with converted data with extension conv in al11 tcode to process batch input processing

Batch input processing

Session------

After all this as per methods that we follow while migrating data we have to verify data to check whether it is migrated or not in se11 mara table or to any database table.

Direct input method:

This method is faster among all of the bdc methods in transferring non-sap data into sap

There are certain standard programs for each module in sap to import data directly into sap tables

For each module there are programs as for modules like finance ,sd ,mm, ---

BAPI: business application programming interface

Bapi = business object + rfc

To transfer the data by using BAPI is so optimizing method

BAPI is faster among all the migration techniques than bdc to transfer data

For large data volumes it will be good idea to choose

* in bdc methods if the sap standard table creation transactions like mm01 tcode changes it screens by adding few more fields or screens then there will be error as we will run the program as per the trial recording syntax the automates all the data so new screen logic or fields will be not available in recording
* but in bapi we can use the older bapi syntax until there is new bapi is developed
* in bdc it data entry into table will be done by subsequent screens like in mm01 as per recording so its bit slow
* but in bapi it directly added to database table with subsequent steps so its faster

there are predefined bapi function module will be available in sap that are already developed we can use those bapis like material – mara table for creating records we will use = bapi\_material\_save\_data for changing existing data = bapi\_material\_edit

in return table we can get the status of the record that are added into table successfully or not

project wise :

if you want to add custom fields to bapi as the clients are asking to add few more fields in their file to add into bapi and database table then

* we should add those fields into the bapi structure first by seeing into different structures available in that bapi which also has some associate tables in that structure we have to choose the related structure like mara then with this also consider the table that updates the information of table or fields that suffix with ‘x’ like maraxt, marcx
* first in those append the structure in standard bapi’s using global structure as per the fields that are mentioned by clients to add with perfect naming
* for both the bapi-mara and bapi-marax
* later append the fields to the sap database tables of that new fields that will flow like this

file->program->internal\_table->bapi->table

* so add fields to the table as well
* main thing is with the “extensionin” and “extensioninx” parameters we will add those fields to bapi function module from defining into substructure in bapi then have to assign to main bapi extension
* at last as per methods of data migration verify the data in database table

z

**ALV:** ABAP list viewer

To display the contents or list in a format with certain added functionalities in output

Like write statement but having more uses in alv.

Having functionalities like

Sorting – asc, desc

Filtering

Exporting data = like download we can copy the list of output in excel spreadsheet into local disk

Summation = total sum

ALV creation

* create a field catalogue
  + manual field catalogue = by manually giving fields, positions
  + sap field catalogue = with function module – ‘REUSE\_ALV\_FIELDCATALOG\_MERGE’.
* Populate internal table according to your logic to display in alv format
* Pass the internal table to function modules according to list and grid and hierarchical
  + REUSE\_ALV\_GRID\_DISPLAY = each cell will be in grid format while displaying data and this is flexible in expanding the cell tabs
  + REUSE\_ALV\_LIST\_DISPLAY : it will display like a normal list as a table
  + REUSE\_ALV\_HIERSEQ\_DISPLAY : its like a grid display which have header data and item data as 2 different internal table will be display at a time in a hierarchical manner with one key field of both tables at 1st record it will displays header data and in sub list for the same key item it will display item data
* We can create the field catalogue and this mapping the internal table with field catalogue in the same reuse\_alv\_grid\_display itself by passing the global structure here but we sap automatically generated the field catalogue and displays and we don’t the name of field catalogue if we want any manipulations in that field catalogue like we need to change column position or field label like certain parameters data we can’t do if any requirement , if you don’t need to change the field catalogue further you can use one function module
* If you want to add the top of page to alv grid then there is a parameter that needs a subroutine creation in that we have to write logic there by passing name in function module
  + We have to use a function module to pass the text for top of page to : ‘REUSE\_ALV\_COMMENTARY\_WRITE’ in this module we must pass internal table parameter having type = ‘h’ = heading , ‘s’ = selection, ‘a’ = action

Key, info = based on the text format we can choose the type in this parameters

* In parameters the name that starts with “I\_pfstatus” with I then it Is subroutine – (found)
* For interactive alv we will use certain parameter subroutines like pf status and user command events
* Personal function used to create GUI status that has buttons in output screen in alv to behave interactive lists by creating secondary lists as for each button we will create unique identifying function type to write logic as per button action in user command event
  + We can create buttons in menu bar, function keys, title bar, application tool bar
* In user command event we need to write action command to perform what action have to perform based on the clicking on each button
* Use the parameter types that declared in sap standard
* Use the subroutine implementation syntax same as sap by navigating to sap syntax copy and use that code
* Hint : while navigating to get the sap user command subroutine implementation syntax instead of user command parameter search for pf status parameter on inside and inside the function module you will find the user command parameter value in function module that starts with overview\_user\_command below that function module you can find the syntax after searching with this name of sap
* In user command code as per requirement with function code
* Use the using parameter in subroutines for function type and the values that are selected
* Note : the cursor pointed value will be taken as value in using parameter
* To get the entire record be selected and to take the values of that record choose the tabindex field to get those data

Hotspot creation in alv

This option will be available at field catalogue parameter when it is getting created

Its possible to enable in manual field catalogue parameter option like fieldcat1-colpos= 1

So you can activate the hyperlink to that particular field to take the value by sap

There is a parameter in grid function module keep ‘X’ value to enable and notice this as enabled as it will be changed as hyperlink

After enabling the hotspot then code accordingly based on the function code in user command check the function code in debugging that you are getting in the formal parameter column

If you want to call any transaction like va01 then use syntax as

Call transaction ‘va01’

If you want to place automatically the value in that field then

Get the parameter id to which field you want to set this selected value as hotspot from data element option

In data element at further characteristics tab you can find the parameter id and with that you can keep that as place holder the value

In this there is in memory process like abap memory , sap memory concept

The syntax to keep this as

Set parameter id ‘id’ field ‘matnr’.

When you create new pf status event then sap pf status gui buttons will be disabled to get those buttons as well with your new buttons

Then copy those sap gui buttons to you newly created pf event with the help of tcode se41

In this menu painter pass our program name and the pf status name and click on copy so here the sap standard program name is ‘saplsalv’ and chose the standard pf status name with f4 help

**Debugging**

It’s a technique to trace a bug or error to find and resolve that errors.

F5-step by step execution-useful when there are reusable components like subroutines, nested function module, classes, interfaces, if these resides inside a statement then to transfer control to inside statements we will use step by step execution

F6-line by line execution-it executes each statement at once , statement indicate that ends with full stop and it won’t go inside whether there are any nested statements or reusable statements

F7-return-if we go much deeper by transferring control in nested functions, reusable components to return back to main program we will use return statement

F8-continuous execution or exit- this will executes the program at once it will directly process the program to display output without stopping at anywhere and it will only stops when if there is a another break point a part from the current break point or else it will exits the debugger

Static break points – user independent break points , the debugger will be on to every user if it is on

Syntax : break-point.

Dynamic break points – user dependent , break point executes to that particular user who sets it on

Syntax: break uname(user name).

Dynamic break point is most preferable.

* We can go to the statements in debugger that we desire to go by leaving the current execution or in order to skip the next statement in way like
  + By placing the cursor in that statement and debugger->go to statement
* The above statement can be used to backtracking as well to go back of the execution statement that are already executed as by go to statement.

Operations on internal tables in debugging using services of the tool option

* can change the values in internal table.
* Can delete the record or insert new record
* Can delete all rows

Breakpoint at : it will set breakpoints collectively at once by selecting the type at where you wanted to keep , breakpoint ->break point at ->breakpoint at function/subroutine/statement-------

If you select breakpoint at statement and pass the keyword as ‘select’ then it will sets breakpoints in the entire program where the select queries found.

Watch points: to monitor the variables, data objects, work areas we can create watch points to each variable, if that created variable is filled with values or else if the control reaches to that variable then it will give a message like ‘watch point reached to variable’, can see those watch points in breakpoint/watchpoints tab.

You can also keep the condition while creating watch point in case of particular records in internal table to check whether the record getting into table or not if it reaches the it sends a message and condition be like = lt\_data-ono = ‘00000001’.

BREAK POINTS LIMIT = 30

WATCH POINTS LIMIT = 10

Backtracking: if any statement is executed in debugging or you missed any statement the control forwarded to some other statement or you want to delete the internal table by service tool and to debug that again using , place cursor on that statement and debugger->go to statement .

Delete all : breakpoint -> delete all

It will delete the breakpoints after the current breakpoints that are set to execute those breakpoints are deleted permanently again those breakpoints won’t appear.

Deactivate all : breakpoint-> deactivate all

It will deactivate the next breakpoints that are set to execute are deactivates temporarily for the current session. And those breakpoints will work fine in next session.

Debugging with a text file

If you unable to keep breakpoints in any screen by using session breakpoints or any other static, dynamic then write some commands in text file and drag that file into that particular window where you want to start debugging session then you can confirm by a message as ‘debugging is on’

Example : for material master creation for tcode ‘MM01’ then you unable to enter debugging session after entering 2 field details then you can get a pop up window after selecting a field like basic data or ----- after that you wanted to enter into debugging session then after selecting that value you can’t enter /h or else you can’t keep break point then after selecting you want to do is drag the file into that session then you can get message like debugging is on that says the file is properly dragged into the session window.

In file you have to write below code

Debugging.txt

[Function]

Command:/H

Type: System Command

**Transport Organizer = se10, se09**

It is used to transfer the abap objects from one system to another system as we have 3 systems development, quality = testing the developed objects, production = live system

Development⬄quality=>production

Development = dev / cust = development /customer system

Development and configurations of system part will be done and will be used and maintained by customer a

Quality = qas / qtst = quality assurance / quality testing system

The developed objects will be testes in this quality checking by functional person

Production = prod / prd = live system

The clients under that project will be used in their business that were developed

Status of transport requests

Modifiable = abap objects can change/editable in that system and can’t transfer to other system

Release = can transfer to other system like dev,qua,pro but can’t editable as it won’t reflect in other system, by releasing it will be indicated by tick symbol beside the subtask and at request

Types of transport requests

Workbench requests = all development sap abap objects will comes under workbench as the development will be done in abap workbench so under this transport request will be created and transfer via this request.

Customizing requests = all configuration details to sap system and their permissions that were handled by basis team usually uses this request to save those configuration in the Tcode = SPRO and also to store table content in this request all other part of tables like domains, tmg, table will be under workbench request after selecting the delivery class with customizing data when creating table content it will ask transport request so there you can choose customizing request.

Transport of copies = for a scenario the development system transfer data from modifiable to release so functional person will test those objects in quality system so that person contacts basis team to export those released objects after testing those objects there are certain changes has to perform so now the developer has to make changes of those objects and he can change the object even though he released those objects to other system that object just transferred but you have the data in your package and that can editable but we already discussed that the released data can’t be edited as under the request we had sent won’t be edited but we can edit in our package and can create a new workbench request by saving and can transfer the request again by releasing after changing the data now the functional person can get object through request

Now, as per above scenario the request count will be same with number of changes that are made to the object under the testing and moving from quality ⬄ development , so certain companies will follow to keep only one request that to be transfer from development->quality->production in that case this transport of copies comes into picture like if there is any changes has to done to development system instead of creating again a workbench request we can create a transport of copies request -> choose the target system to which system that you have to transfer (Quality).

Copy the number of subtask in the transport request that needs to copy -> select the transport request to where to past ->include object -> paste the request number of the subtask -> continue->objects will be copied after that you can-> release the request

It will act as a copy to the main request like a file and the copied one will be transferred

If there will be no changes has to perform and confirmed with quality testing and now you can release the main request that were created by releasing and it will reach to production = live system

Relocation:

Version management : this will holds all the versions list of an object that’s been undergone , version says if an abap object is developed in a workbench request->released->tested->changes in dev system->new workbench request->released

Here there are 2 requests and 2 different copies of code of that object before changes and after changes these are called versions and the versions will be maintained in this version management and it will be helpful that where the changes has done by comparing the codes

Select a object(program)->utilities->versions->version management

Operations can be performs with transport requests

* Create transport request.
* Delete transport request.
* Add new user to the request = by default there will be a owner to the request so for that while creating it will generate a subtask in the request like a folder all the abap objects will be under this subtask that we perform if you want to add a user by adding one more subtask for your request so extra subtask will be added so under that subtask the user 2 can develop the objects and can store there, for certain projects the tasks will be distributed so under one request many users will be there and each can perform their task and can release their subtasks after releasing all subtask the owner of the request will release the request
* Copies the object.
* Simulate release = like syntax check to the transport request while releasing those request if there is any inactive objects or error objects this simulate request detects and notifies after this step you can perform the transport request releasing mechanism

Release/task->complete check->simulate release.

**Enhancements**

Enhancements : enhancing or changing sap functionality in customer name space ‘z/y’

Implementing by adding new sap functionality by creating custom enhancement implementations that’s available in sap logic using various enhancement types

Modifications : enhancing or changing sap logic in sap name space

Implementing the new functionality or Modification will be done in sap namespace

Types :

* Implicit enhancement - Enhancements
* Explicit enhancements - Enhancements
* Customer exits - Enhancements
* BADI’s - Enhancements
* User exits – Modifications

Implicit enhancements:

Enhancements will be performed at begin/ending of the program

Steps:

Open sap program->edit->enhancement operations->show implicit enhancements->will display enhancements that be indicated by “ -> place the cursor and click on spiral symbol it indicate a arrow mark at that line then -> right click->create implementation->

Here you need to give name to custom enhancement implementation by selecting create button

Choose that enhancement that will adds the enhancement syntax with that name you created into the program and you can add your functionality

Activate and run

Explicit enhancements :

Enhancements can be performed at anywhere in the program

If you want create the enhancement option in the program then you can enhance it at anywhere by placing the cursor on any line in change mode of sap program and by these steps

Place the cursor on line edit->create option->select enhancement point/enhancement section ->enter the custom enhancement name to implement now or at future in that block of enhancement ->enter enhancement spot name that stores the various enhancement points that every time user need to create enhancement are stored under this enhancement spot give the custom name and enter

Now you can see a block of code is generated with enhancement point/section keywords

Now you can add code or make in display mode

When in future you need not to do above processes again when you need to implement the enhancement point/section as it is created

In sap standard code the above steps are implemented by developers at their desired line

We need to implement those options

Steps

Place the cursor on enhancement point/section -> click on spiral symbol->indicates arrow mark->right click->create implementation -> create custom enhancement name->continue

Now you can edit and add the code to the program

Enhancement section also follows the same procedure to implement the enhancement section

Difference between enhancement point and section is

Enhancement point -> used to add the code to the existing code and does not replace or edit the previous code were implemented in enhancement point

Enhancement section-> used to replace and add the code as the earlier edited code will allows user to edit the code by replacing or add code in enhancement block

Enhancement spot-> container to enhancement point and section implementation ny the users periodically

Enhancement options -> it is selection made while creating explicit enhancement like enhancement point or section

Customer enhancements:

* Function exits / function module exit
* Menu exit
* Screen exit

Function exit :

These function exits can be find in 3 ways

* Breakpoint at call customer function statement : check the function exit description whether it will be useful to the requirement – you can only find the function name but not the exit name so for that you need the other type of function exit detection

Syntax : call customer function ‘three digit number (000-999)’. => breakpoint

/h->in debugging mode->breakpoints->breakpoint at statement -> call customer function. = check description in attributed tab to the requirement

Function module -> source code -> include ->

* Tcode SMOD = with the help of package we can find exits in this SMOD tcode

For eg VA01 TCODE the package will be checked as

Go to tcode VA01 -> system -> status ->program -> double click and open program->go to -> attributes->package

Pass this package name in SMOD tcode to check the exits available to the VA01 tcode

We can find function exits,menu exits,screen exits at a time by checking the entries available in that corresponding tables as function exit, menu exits, screen exits has the exit name defined there

SMOD tcode -> utilities->find->pass package name here to check exits-> find the exit name based on the description then you can find menu exit,screen exit,function exit

* Tcode se84 = enhancements->function exits->enhancement definition->package name->exits

Function exit implementation

CMOD tcode

Create a custom project -> description->enhancement assignments->exit name (function,menu,screen)->components -> can see all exits = function,menu,screen->double click on function exit->function module -> include-> create a custom include by double click -> write your logic activate the project

3 ways to find Menu exit

* +CU suffix to function code for that menu GUI status that indicates menu exit is there and custom changes can perform

From that screen eg = va01 ->system->status->gui status->double click->menu bar->check for +cu suffixed function code so for that the exits are available then you can add custom menu button(gui status in that function code)

* SMOD tcode

Package name finding is similar as function exit

Utilities->find->package name

* Se84 tcode

Same like function exit path is same to find exit

Enhancement->customer enh->package

By pasting package name it displays list of exits based on the description find out one exit and open it and find there is menu exit entry will be there or not

Finding package is similar like function exit like system status->program->attributes->package and copy and paste

Implementing the menu exit by finding the exit name by above process in CMOD

Same as function exit for implantation

Project creation->enhancement assignments->pasting exit name ->components->change mode->open the function code +cu\_ and edit the function code as per requirement

Adds the button to menu bar

Write the logic for the gui status like user command for pf status

For the action of menu logic will be written in function module

Any how menu exit is there and for this exit we need function exit also for the action to that menu

In the same exit name you will find entry in function exit as well

Open the exit name and open include and create include and write logic to that button

Activate the project

* Screen exit

3 ways to find the screen exit

* Call customer-subscreen ‘subscreen area’ including ‘program name’ subscreen number.

Call customer-subscreen sub including syrepid 1001.

From the screen that you need to add subscreen area by finding the screen exit

Similar to function exit from that screen system->status->screen number->find the above syntax like call customer subsreen --- -> in PBO -> in PAI also such syntax will be ther-> as per the requirement choose the subscreen syntax and exit-> first PBO event will called before input that user has to give

If you find that screen exit will there to that screen

* Smod

Similar to menu exit and function exit find the package based on the desc and then paste the package to implement in CMOD tcode

* Se84

Similar to function,menu exit based on the desc find the exit

Implementing in CMOD

Create the project ->enhancement assignment ->paste exit name ->components->click on the screen to open the subscreen window->at attributes tab add desc and choose screen type as ’ subscreen’ in layout design as per requirement like fields,checkboxex and save the project and layout

Activate the project

Important points:

One exit to be implemented at only one time no multiple implementation by creating multiple projects in Cmod

One exit : one project = 1:1

Smod : to find customer exits

Cmod : to implement customer exits

MODACT – table to find the projects -> by taking input as customer exit name

MODSAP – table to find customer exit name -> by taking input as function module name by which we get it from placing breakpoints in call customer function ‘function name’

BADI – Business ADD ins

One of enhancement technique as we will enhance the sap system as per our requirement in custom name space

BADI are related to oops conceptual enhancement as it will contains the methods, interfaces ,classes

2 types of BADIs

Classic BADI

New BADI or Kernel BADI

Interface contains methods without implementation

Those methods will be implemented in classes by inheriting those interfaces

Tcodes

Se18 = BADI definition via interface methods

Se19 = BADI Implementation via class methods

Finding those BADIs in 3 ways

Classic BADI

* By using cl\_exithandler->get instance method-> break point on the line case statement so by checking exit name in above function module BADI name are displayed by that variable, before placing breakpoint you need to start the screen from which do you want to find the BADI , for eg mm01 tcode to create material then by passing inputs and from where you need to make change before that screen place breakpoint in that class and method and execute the screen then the screen shows in debugging to displays BADIs
  + This is only for classic BADIs
* Se18 – BADI definition

Similar to SMOD in customer exit by getting the package name from that screen like system status program goto attributes package and then in se18->select BADI->use help->information system->paste package and it will display all BADI->find the BADI based on the description you need to find the BADI and create the definition of badi in custom namespace

* + It will generate a class automatically , BADI has interface and method and for that implementation in se19 tcode it will define in this tcode
  + BADI type in properties if
    - sap use is in ticked then that BADI can’t use for implementation only for sap use
    - if multiple use is ticked then we can create multiple implementations to a single BADI also and its not in customer exits as it is 1:1 = project:cust exit
* \* se84 – BADI definition

Similar to customer exit enhancement->badi -> package and find BADIS and define them with your custom namespace

Se19 – BADI implementation

Create implementation for the BADI that is defined

Open the BADI implementation that were defined in se18 tcode and that will be implemented in se19 tcode as the interface and class will be created in the badi by opening the class in the methods our logic to be added as per requirement in that screen so the methods will be displayed as it will be inherited with tilt operator to get those methods to be inherited into the BADI so those methods will be find to where you need to write logic so it will be fulfil the requirement and cross check if you have doubt on particular methos by placing breakpoint in the method and check by executing the screen from where you have real requirement whether debugging is triggering or not.

Activate BADI

New BADI or Kernel BADI

Difference between classic BADI and Kernel BADI is for Classic badi there is no enhancement spot but new BADI there is enhancement spot to store all the BADI definitions , its similar to the concept in explicit enhancements the enhancement spot contains enhancement point and section

Enhancement spot (container) = BADI definition

Enhancement implementation(container) = BADI implementation

3 ways to find the new BADI

* Place breakpoint at ‘CALL BADI’ and ‘GET BADI’ by opening the required tcode screen and place ‘/h’ to switch on debugging and select breakpoint at ->statement-> call BADI and GET BADI

Now it will keep the breakpoints and check the BADI in a way like

As BADI is related to oops concepts GET BADI is the statement for creating a instance object to call the method by taking reference of the BADI interface

You can notice this by the statements that are used in the program level by the syntac of get badi and the object declaration

After checking the method and its description find the BADI that you are looking for your requirement , is that method is related to the requirement so you can confirm the method that contains inside interface and it contains inside enhancement spot

Note down enhancement spot as we have to pass this name while implementing the BADI implementation unlike classic BADI , there we used to pass BADI definition and here we will pass enhancement spot to implement it

* Se18 same as classic BADI find the BADI by seeing method description inside those BADIs
* Se84

Implementing BADI

Using se19 ->new badi->enhancement spot->

By giving enhancement spot and it will ask for badi implementation name and class name we have to give custom name for that unlike classic BADI in classic BADI by giving the custom BADI implementation name it will automatically created a class but here we need to mention the class and implementation name

As enhancement implantation is the container for BADI implantation and select the BADI name

As this BADI name will be displayed as we passed the enhancement spot while creating so it will fetches all the definition available under that spot and pass the custom badi implementation and create it

By opening the class that is created and find the method that needs to implement your logic as per requirement , open and implement the logic

For the confirmation of the method that you chosen is correct or confirmed by placing breakpoint at that method and execute the screens or the tcode whether the method is triggering to implement logic there or not

Activate the BADI implementation.

Deactivation of BADI implementation in classic BADI and new BADI can be done but the code won’t be deleted but it doesn’t work as it is deactivated rather not to miss code and not to work we can deactivate.

Extra learning by creating a BADI definition, enhancement spot = which was generally created by SAP

That we can create custom NEW BADI for practice purpose without hampering sap system

To create New BADI definition we need enhancement spot that is a container for BADI Definitions

In BADI definition we need to create Interface and in Interface we need to create methods and its parameters, and it don’t have implementation

For that we need to implement this definition by inheriting into a class which is in BADI implementation as while giving enhancement spot name to create implementation it will automatically inherit the interface in your class which you created in implementation

in SE19.

Enhancement implementation will be created to store BADI implementation

N no of enhancement implementation can be created in order to create n number of BADI implementation for one BADI definition its like a multiple inheritance as inheriting interface by multiple classes

If we create multiple classes for one BADI definition by default it will overwrite each implementation one after other when the BADI definition is called in a program so the last enhancement logic will be shown in output .

In order to make ordering or to filter to multiple BADI implementations by ranking or condition when to execute that particular implementation can be achieved through BADI with Filters

BADI with filters can be created in SE18

By right click on the BADI definition we can create our custom filter and it asks for the datatype , so based on that type ,it will filter and execute that particular implementation like a condition as a syntax in program be like GET BADI lo\_object FILTERS z\_filter\_924 = ‘1’ or ‘2’ or -------

Before this after creating the filet in SE18 tcode

We should assign those values to each implementation by opening that implementation and open filter->create combination->create folder symbol->select = operator ->give the value according to the type that we declared while creating the filter

Now we can use the filter condition in program after GET BADI statement.

If none of the enhancement is called like the user is selected a wrong filter condition if you kept that in dynamic variable to choose by the user at run time then by default there is a class that can called if we create that class

That class is FALL BACK CLASS

This class will be created in SE18 tcode as similar to the class in SE19

This class will be automatically inherits the methods of interface in that class we can write our own logic, if no implementations are created or user chooses wrong filtering value to execute a non existing implementation.

Steps to create new BADI from BADI definition level for practice.

Go to se18-> create enhancement spot -> desc->give custom name for enhancement spot and desc ->create a BADI Definition of custom namespace->go to interface->provide custom name and create interface-> create the methods for the interface and no implementation is necessary as it is interface and implementation will be done at class level and at BADI implementation -> activate the BADI definition.

BADI definition is completed and

Here you can write the program in similar to sap by using the statements like

GET BADI object name(lo\_object) FILTER zfilter\_924 = p\_filter.

CALL BADI ‘BADI name’

now create BADI implementation in se19

generally we keeps breakpoint here to fetch the methods of BADI definition to check feasibility of methods to change as per requirement in that BADI later for implementation.

Se19->provide the enhancement spot-> provide enhancement implementation of custom namespace-> give class name in custom name space

Here enhancement spot will be the linkage for BADI definition and BADI implementation

As we will create enhancement implementation that contains BADI implementation so enhancement spot will automatically fetches the BADI definition over here to implement that definition in this implementation .

Additional learning

se19 = class – able to implement the method , se18 = interface - with no method implementation

after creating the implementation by your own name space we need to create the class in this

By giving the class name the BADI interface method will be inherited into the class so you can add the logic in that method

As we know logic will be written inside the method of class

If there is filters like multiple implementation for same BADI definition and filter is created inside BADI definition then assign value to the implementation based on the type.

USER EXITS

User exit is a modification unlike before method of enhancements of customer exits,implicit,explicit,BADI’s the user exits are modifications as the user has to enhance or change the sap code with the sap namespace itself, in a way like by getting access key from basis team and by entering in change mode we have to change the code in edit mode has to edit sap code

The changes has to be performed in a dedicated portion in the sap code where that portion says the user exit

User exits will only applicable for Sales and distribution module in sap, vbak, vbap

No other module won’t allows user exits

User exits is residing in subroutine inside a include program in a way of form and end form block

Mostly used include program in their project requirements in MV45AFZZ include

In that include most of the user exits available to fulfil the requirement

Steps to find out the user exit that to edit our code in sap

* In every standard sap program maintains the documentation in the program itself that tells us where the user exits is available with a list of include program
  + Way to enter into that standard sap program is from that screen that you are looking to find user exit for suppose in the screen VA01 from here click system ->status->program screen-> now you can see the documentation having includes that contains user exits
* Use the find button or search for statement like PERFORM USER EXIT
  + It will directly displays number of user exits inside include the thing is we have to check each and every user exit and find the user exit based on the user exit name in the perform statement or in form and end form implementation
* Package VMOD
  + It will give a list of include that contains user exits available in the SD Module
  + Place the package name in se80 and get the include and user exit
  + There will be multiple number of include and finding the include is a bit time consuming but no other option have to search in these ways
  + But any how we can get easier in a way like you have to find the program name from the screen like for VA01 the program name is SAPMV45A so we can get a assumption like most of the user exit available in a include the starts with MV45A as it is a suffix to program name .

After find the place where to write our code then get the access key and change the mode of the program and edit the program in sap namespace .

Key point to know : updktx = is a field name column name or a variable that stores the mode of the program that user is executing like VA01 = create VA02 = change VA03 = display

So based on the tcode the structure or column of vbak that updktx stores the mode = I / U /D = insert , update, display

2 structure will be maintained by standard sap like XVBAK and YVBAK

XVBAK = stores new data – in some cases to get the updated records this table will be chosen

YVBAK = stores old data - in such cases like while creating any delivery order it will be null

BTE – Business transaction events – is a enhancement

Enhancement For FI module with custom namespace

It’s a function module , changes and finding will be done based on this

We will enhance and create a function module and will attach to standard sap program

Which in result it will gives the output by all the sap logic and in the function that we enhanced

Only finance related enhancements will be performed in this enhancement method

Tcode is FIBF

Based on the interface 2 types if enhancement was classified

* Publish and subscribe interface – called informing interface
* Process interface

Publish and subscribe interface :

This type will be used when user do not need to hamper the system by changing data or enhancing or changing sap functionality instead of that user need to add any functionality after processing and the user wants to add features at output by adding certain features like email sending based on the output (email functionality) and sending data to external system

It doesn’t hampers sap system

In this function module it does not have changing parameter because any how it won’t change anything no need such parameter

Ways to find BTE in a sap program any ways you can go to sap program from any screen by system->status->program screen

* Search for the statement using the find button by ‘open\_fi\_perform\_<BTEnumber>\_E’

You can try before bte number also by entering upto perform word the subroutines will be displayed if you search

But the statement syntax will looks like the above syntax as BTE subroutine name it subroutine call statement

The name that ends with E says it is a type of publish and subscribe interface

* Set the breakpoint in function module at beginning line and function module name is : BF\_FUNCTIONS\_FIND

Set the breakpoint in that function module (SE37)and execute the screen like if you want to find the BTE in tcode FB01 then after placing the breakpoint execute this tcode then debugging starts here and in DESKTOP2 you can see the stack structure it says from where to where the flow of program is switching like program->subroutine->subroutine->include

The detailed structure will be there

You can notice the BTE subroutine that is triggering while executing the screen like FB01

So through this you can find the BTE of publish and subscribe interface subroutine

You can see about the subroutine details in attributes tab any how it will maintain the description in german language use google translator to understand it

* Using tcode – FIBF

FIBF->environment ->info system P/S = publish and subscribe interface

It will gives the list of subroutine in the FI module

Based in the description you can find the BTE P/S

Mostly the BTE subroutine will be called before the output that was displayed keep it in mind while searching for BTE p/s based on the requirement

Process interface:

When the user wants to enhance any sap logic with any such requirements like if the finance document should contain a default value with customer name so its not a sap logic the user has to write an extra logic means enhance the code by creating a function module in custom namespace and attach it to sap to add this in sap logic

It hampers sap system

In this function module it have changing and exporting parameters as it changes sap data.

Way to find BTE of process interface

* Search for the statement using the find button by ‘open\_fi\_perform\_<BTEnumber>\_P’

You can try before bte number also by entering upto perform word the subroutines will be displayed if you search

But the statement syntax will looks like the above syntax as BTE subroutine name it subroutine call statement

The name that ends with P says it is a type of publish and subscribe interface

* Same as the publish and subscribe interface method to find the subroutine but it has a different subroutine to keep the breakpoint but mechanism is same and name of the function module -> in se37-> is PC\_FUNCTION\_FIND
* Same as publish and subscribe interface

Using tcode – FIBF

FIBF->environment ->info system Process = process interface

It will gives the list of subroutine in the FI module

Based in the description you can find the BTE P/S

Based on the requirement debug carefully subroutine will be called before or after the screen that you are currently opened and looking for BTE process

Practical:

If the requirement is related to process interface (to enhance sap logic in program) or publish and subscribe interface(to add any functionality at output level like sending data to other system or sending email )

Now will see a requirement related to process interface that indicates a enhancement in sap logic like as this BTE is for Finance accounting module so the tcodes is fb01 = create accounting document , fb02 = to change accounting document, fb03 = to display the accounting document

Requirement is like in changing accounting document by taking user inputs based on this we have to disable the input options for certain fields to change or edit those details for the chosen accounting document as similar to sales order , to finance document it has header table and item table for 1 header there are multiple item data 1:N relationship

By giving header data as input to change the details again we have to select the item to change as there are N items in one header so choosing one item we can change those item details

The requirement is when if suppose user gives the BUKRS as DE11 then we have to disable the input for the item text field for that selected item so for this

Find the BTE that matches the requirement based on the description and create a function module similar to BTE by custom name and attach that custom function module to BTE to execute our logic that we writes in our function module as of BTE executes our logic will also executes

Ways to find the BTE – Process interface

* By setting the breakpoint in the function module se37 name is pc\_function\_find , at the beginning of the module and execute the tcode that you looking to change sap code as tcode is fb02 so run the tcode by giving required input so the function module breakpoint will triggers and find out the BTE name by checking in the debugging like BTE is shown in stack flow desktop in desktop 2 in debugging screen

Syntax for BTE – process interface statement = open\_fi\_perform-BTEnumber\_eventnumber\_P

For process interface

* Copy the process sample interface BTE as it is a function module – with custom name space copy it and give a name to it before that create a function group to store this function module
* Go to tcode – fibf = execute it will give list of all BTEs so find the BTE that you find in debugging screen and select that and click on pattern function module
* It will take you to se37 tcode and use copy button and copy with your custom name so you can write your logic there
* Write the code in source code of the function module based on the importing parameters of the function module in debugging you can know what are the values are there in those parameters so according to that write you logic
* Keep the breakpoint inside the BTE that you find out so you can understand the values that flows in those parameters
* You can find the field names and table name from the technical information (F1 help) from the screen when you need to know about the field name that you are giving input in a screen
* If there is tables parameter so those table parameter is used to export your values, check according to the requirement declare those table in your logic and append the values when you need to change those values like tablename-fname = ‘tablename-fieldname’ so that functionality is reflected in the execution like
  + T\_noInput-fname = ‘BSEG-SGTXT
  + T\_invisible -fname = ‘BSEG-SGTXT’
* After logic is written activate the function module and function group
* Now link this custom function module to BTE so
* Create product and activate product = link the function module with event (suffix number of BTE number)
* Goto tcode fibf->setting->create product->as a customer ->create new entry->custom product name->description->enable check box Activate a->Save
* When save you will be asked for new transport request = the request is Customizing request as it is linking the function module and BTE so it is related to configuration so the request is customizing request and you can create your own request
* In SE11 - TBE24 that stores all the product details that were created in the fibf tcode you can check that when you create a product of your own by checking the table contents of TBE24 your product is added or not
* Event name is BTE number suffix numeric number
* To link function module with event in BTE so need to Create module – process module
* Goto tcode FIBF->setting ->create process module->as a customer->add a entry->use f4 to add entry->choose the BTE number in process interface->add your custom function module name here to link you logic ->add your custom product that you created use f4 help can get your product info -> no need to give certain values like country specific or application fields it will take by default->Save->in customizing request
* Similar to the product entry the table is also there for the process module entries that will be added in this se11 table for process module = TPS34 publish and subscribe module = TBE34
* Now the requirement is process module so check in TPS34
* Now your custom function module is linked to BTE
* Whenever the BTE is called from the screen of FB02 of finance module
* Then your code will be executed and sap code will be enhanced
* Check the entire flow in debugging screen that how the FB02 is calling the function module of pf\_function\_find and BTE function module and then how your function module is triggering and how the data is flowing like TPS34 table,TBE24 table how sap is getting these values to used in you custom function module check all in debugging for the better understanding
* If you want to disable you logic that is triggering from the sap of custom function module that you liked to BTE
* Just unselect the activate field in product that you had created in FIBF tcode ->settings->product-> as customers->select your product and deselect activate field and save
* Check it is deactivate in TBE24 table as it stores the product information so custome logic won’t be triggered when sap do not want your code to run

**Module pool programming**

Module pool programming is a dialog programming where the pool of modules are used modules = includes,screens,transactions

To execute the module pool program transaction should be created to that program

Include programs = to store the modules = pbo,pai,poh,pov

Execution flow

Pbo -> pai

If 2 screen are there then

before screen appears -> 1st screen pbo -> before screen appears -> user clicks on button -> 1st screen PAI -> 2nd screen PBO = before displays 2nd screen->button clicks -> 2nd screen PAI

if 1 screen is there then

before screen appears->PBO event called->button clicks -> PAI event called-> to again stays into the same previous screen of 1st **screen PBO of the same screen is called** again if no next screen is available and to stay back to original postion

Types of screens

Normal screen

Sub screen = CALL SUBSCREEN SUB INCLUDING SY-REPID '0200'.

Subscreen area = position where to call

Sub screen number = actual screen that to be called in that area like SUB in normal screen

Modal Dialog screen = like a pop window or dialog window

      CALL SCREEN '0200' STARTING AT 10 20  
                      ENDING AT 50 60.

Starting at and ending at indicates the co ordinates to the dialog screen that means top left edge and bottom right edge positions to the dialog box to be popped out over main screen

Selection screen /selection dynpro

Leave to screen ‘0’ = this means it will leave to the previous screen from the screen stack

Tab strip screen.

In a normal screen type it we need to create multiple tabs then we will use tab strip screen

In the same screen we will use tab strip with wizard we will use to create tab strip by providing the tabs in that wizard process

Then tabs will be created with SUBSCREEN type each tab will be in sub screen by default the tab strip wizard writes the code that calls these sub screen in the main screen

We can write our logic in the screen of sub screen as per our requirement like to display header data in tab1 and item data in tab2

. If one record to be displayed in the screen then better choose the dictionary button in the layout in that particular screen to design the fields as per the type of the table like input field 1 and input field 2 , 3 ----

Then select and double click that input field properties will open then disable the input check box to use that box for output

. If more than one record is there to display in out put then choose table control with wizard so it will maps the layout fields with internal table data.

. Radio buttons also can be used in layouts to design in screen the important note for such case is by taking one radio button in default it will assume like it in individual group but we need to group all the radio button after design in layout then after select all the radio button ->edit->grouping->raio button->define

Then it will form as a group

. if some cases if we do not give input to the input field in the selection screen it wont give warning or error reminding to give input to the user

In such case if we want show some error message then go to layout->select the button which is a button used by the user to show the data as per input field the by selecting and doublt clicking the button and in properties besides input there is dropdown ->button ’required’.

It will act as a obligatory feature to the parameter variable

. the limitation to the above option is if we click any button without a input to the input field then it gives error message to user

For suppose the user clicks back button in such case user do not want the data to display and the user wants to go back without giving input field in such case

We have to use exit command type

EXIT COMMAND :

In layout of that screen click on the button to which we need to provide exit command ->properties will open->function type-> exit

By giving the exit type to the button whenever user clicks the button as it is a exit type button then in the program it will checks for a module with exit command event like

Module BACK AT EXIT COMMAND.

This syntax in PAI block so we can write whatever logic in that module for the action of that button.

the button wont check for input validation whether input is there or not in input field , it will perform the action that we writes in the module with at exit command.

The above process is for button – if we want to give error to the user when no input is there and clicks the button , its button level using function type = exit – and action as module exit command

Now , field level:

If user enters wrong input for the input field which is not exists in the data base to fetch the records and to display then the requirement is like to give a message like no entries exits

If we use that in a user command module at PAI block by a statement of message class like

message E001(zmsge\_924)

then the message shows without fail and the problem is after the pop error or the message shown

it disables the input field to change the input value to give a correct value to the user

so for this a concept call chain and end chain.

CHAIN and END CHAIN

In this block the fields will be validated in a manner like if you give a wrong input value to those fields that you declared in these block and then it will validate those values and make that field in enable mode itself without restricting any input from user

Here after by declaring those field in that block you can write the message class or whatever you wants to write the logic inside the a module of that field like

Chain

Field :zvbak\_924-vbeln module validate.

Endchain

Module validate input.

Message E001(zmsge\_924).

End module.

So it keeps enable mode of the input field even after entering wrong values and validate by message class

**PROCESS ON VALUE REQUEST:**

process ON VALUE-REQUEST .  
   field : vbeln module valuehelp.

FROM INTERNAL TABLE THE FIELD WHICH NEEDS VALUE REQUEST IT WILL TAKE VALUES FROM TABLE THROUGH THIS FUNCTION MODULE ………

  select vbeln from ZSALES\_924 into table lt\_help.  
  
CALL FUNCTION 'F4IF\_INT\_TABLE\_VALUE\_REQUEST' *" binds data to the value help field from internal table*  
  EXPORTING  
*\*   DDIC\_STRUCTURE         = ' '*  
    retfield               = 'VBELN' “”INTERNAL TABLE FIELD NAME  
*\*   PVALKEY                = ' '*  
   DYNPPROG               = SY-REPID  
   DYNPNR                 = SY-DYNNR  
   DYNPROFIELD            = 'VBELN' ”LAYOUT FIELD NAME WHILE DESIGN SCREEN  
*\*   STEPL                  = 0*  
*\*   WINDOW\_TITLE           =*  
*\*   VALUE                  = ' '*  
   VALUE\_ORG              = 'S'  
*\*   MULTIPLE\_CHOICE        = ' '*  
*\*   DISPLAY                = ' '*  
*\*   CALLBACK\_PROGRAM       = ' '*  
*\*   CALLBACK\_FORM          = ' '*  
*\*   CALLBACK\_METHOD        =*  
*\*   MARK\_TAB               =*  
*\* IMPORTING*  
*\*   USER\_RESET             =*  
  tables  
    value\_tab              = lt\_help  
*\*   FIELD\_TAB              =*  
*\*   RETURN\_TAB             =*  
*\*   DYNPFLD\_MAPPING        =*  
 EXCEPTIONS  
   PARAMETER\_ERROR        = 1  
   NO\_VALUES\_FOUND        = 2  
   OTHERS                 = 3  
          .

**PROCESS ON HELP REQUEST**

  PROCESS ON HELP-REQUEST.  
   FIELD: VBELN MODULE TECHELP.

BEFORE TO FETCH THE TECHNICAL DOCUMENTATION NEED TO CREATE A DOCUMENT CLASS FROM TCODE **SE61** by choosing the type as general text and type is TX and create a class and edit a text in word and pass that details in below function module.

CALL FUNCTION 'HELP\_OBJECT\_SHOW'  
  EXPORTING  
    dokclass                            = 'TX'  
*\*   DOKLANGU                            = SY-LANGU*  
    dokname                             = 'Z\_VBELN\_924'  
*\*   DOKTITLE                            = ' '*  
*\*   CALLED\_BY\_PROGRAM                   = ' '*  
*\*   CALLED\_BY\_DYNP                      = ' '*  
*\*   CALLED\_FOR\_TAB                      = ' '*  
*\*   CALLED\_FOR\_FIELD                    = ' '*  
*\*   CALLED\_FOR\_TAB\_FLD\_BTCH\_INPUT       = ' '*  
*\*   MSG\_VAR\_1                           = ' '*  
*\*   MSG\_VAR\_2                           = ' '*  
*\*   MSG\_VAR\_3                           = ' '*  
*\*   MSG\_VAR\_4                           = ' '*  
*\*   CALLED\_BY\_CUAPROG                   = ' '*  
*\*   CALLED\_BY\_CUASTAT                   =*  
*\*   SHORT\_TEXT                          = ' '*  
*\*   CLASSIC\_SAPSCRIPT                   = ' '*  
*\*   MES\_PROGRAM\_NAME                    = ' '*  
*\*   MES\_INCLUDE\_NAME                    = ' '*  
*\*   MES\_LINE\_NUMBER                     =*  
*\*   MES\_EXCEPTION                       = ' '*  
*\* TABLES*  
*\*   LINKS                               =*  
 EXCEPTIONS  
   OBJECT\_NOT\_FOUND                    = 1  
   SAPSCRIPT\_ERROR                     = 2  
   OTHERS                              = 3  
          .  
IF sy-subrc <> 0.  
*\* Implement suitable error handling here*  
ENDIF.

Call screen ‘0200’.

Leave to screen ‘0200’

Difference between these to statements is

Call screen adds the screen 200 in the system stack

Call screen replaces the screen 200 in the system stack from current screen

0100 = call screen 0200 = 0200

Stack will be 0100 -> 0200

0100 = leave to screen 0200 = 0200

Stack will be 0200

In stack previous screen will remove and new screen is added

Stack will be checked at debugging mode in desktop 2

**Leave to screen 0 = it will leaves back to previous screen and current screen number is removed from the stack**

**Current screen is 0100 = used this = Call screen 0200 -> leave to screen 0 -> it says**

**Leaves to previous screen as 0200**

**Stack has 0100 as call screen adds screen number**

**Current screen is 0100 = used this = Leave to screen 0200-> leaves to screen 0 -> it says**

**It doesnot return back to previous screen**

**As leave screen replaces the current screen of 0100 to 0200 and so it does not return back as in stack there is no such 0100 screen the same screen number as 0200 it stores in stack**

**s**

To use multiple ALVs in the list display, first you to initialize the ALV using,

REUSE\_ALV\_BLOCK\_LIST\_INIT.

Then in LOOP, use your FM REUSE\_ALV\_BLOCK\_LIST\_APPEND for different tables which you want to use.

After ENDLOOP. finally use the Function Module, REUSE\_ALV\_BLOCK\_LIST\_DISPLAY.

**Performance optimisation tools**



Last pages – ST05,SAT,SE30,ATC





****

**Output type of adobe form = pdf forms**

**Output type of smart form = ?**

output type = standard layout = sap has its own format in standard system it displays the form and pdf convertible

xsf layout = xml fomat , show the contents and hides the layout and still process

xsf + html layout = in web browser it shows layout as htmls, and similar to xsf as it shows only content

**OOPS**

**Forms**

**SAP HANA – Open SAP , CDS views , ADBC, AMDP**

**Performance tools**

**ODATA**

**Web dynpro**

**Project case studies**

**String Operations:**

****

**Syntax difference between abap sql and open sql**

**Open Sql**

***SELECT vbeln, matkl,  
    CASE MATKL  
      WHEN ‘L001’ THEN matnr  
      WHEN ‘L004’ THEN arktx  
      ELSE ‘ ‘  
    END AS material\_id  
   FROM vbap  
   INTO TABLE @gt\_outtab  
   WHERE vbeln IN @so\_vbeln[].***

***Abap sql :***

**CASE <field>.**

**WHEN <abc>.**

**<statement block>.**

**WHEN <def>.**

**<tatement block>.**

**WHEN <pqr>.**

**<statement block>.**

**......**

**......**

**......**

**WHEN <xyz>.**

**<statement block>.**

**WHEN OTHERS.**

**<statement block>.**

**ENDCASE.**

**Open sql**

**LOOP AT lt\_join INTO DATA(wa\_join).  
  
READ TABLE lt\_ser INTO DATA(wa\_ser) WITH KEY sernr = wa\_join-serial.**

**Abap sql ;**

**READ TABLE <internal\_table> <key> INTO <work\_area\_itab>**

**[COMPARING <F1> <F2>...<Fn>].**

**READ TABLE <internal\_table> WITH KEY = <internal\_tab\_field>.**

**SAP HANA**

High performance and analytic appliances

Combination of software and hardware components to integrate sap components

Can analyse huge data volumes of real time data

Having micro processors to manage real time business applications and to analyse huge data

As it is having Inmemory database – RAM which data will be stored in main memory

To access, store , to compute the data much faster for applications

Having multiple servers in hana database to satisfy business trade decisions as – deep,broad,simple,speed,real time on these five factors at once without compromising on any factor

The servers are – name server,index server, statistic server, pre processor server,persistence layer

Name server= have the info about data flow where and what the data is doing, what the resources are working on what tasks such brief info maintained here

Index server = it stores actual data and use the resources of hardware and software for processing

Persistence layer = confirms the database process or transactions completed or committed or revoked completely without incompletion = tells durability and atomicity

Statistic server= have all the data about the data flow, system status, data base performance , resource usage

Pre processor sever= works with index server, index servers uses this server for analysing and pre processing workings

Hana features

In memory data base

Multi processor

Multiple server works on parallely on different tasks

Row based store, column based store(introduced by hana on analysing views like information views) , object based store

Code push down – code to data

Data compression – compressing the actual data into RAM with less storage usage with the help of some algorithms to occupy less space like dictionary nbased algortiht – says storing the values of column based to neared values with less space in the memory, attribute vector algortihtm- says storing column based values into some index values and grouping into index values and accessing them with the address like a vector ----- ------ ------ ------ ------ ------

Table partitioning – says partitioning table columns into minor colums by breaking them and storing it in memory and accessing them with some hash algorithms so can achieve some performance like load balancing by breaking the huge to chunks of colums , parallel processing= says searching multiple columns of same table with multiple threads to finish task fast , size limitation= normal table should be less than 2 billion records so by this we can exceed the limit

Avoidance of aggregate tables = storing the data into highest level of calculation easy way calulating the data by accessing the data from table and there itself calculating the values and serving to the user

Storing in delta and moving to main – as in hana data will be compressed and stored by undergoing lots of processing to store data in memory, so for each insertion and updation and deletion the same process can’t be performed as its high costlier so instead of modify the main table , inserting data to delta which is uncompressed and over a period this data will be merged to the main table as here over the period lots of tasks is performed once instead of doing one by one operation like delete,insert update , for delet also setting up the flag in delta table and merging it into main table

And doing group of operations at once so it save the performance and it I optomised

Data provisioniong = transferring data from non sap, sap into sap system

There are sap tools for migrating the data into sap

And external tools for the same provisioning the data into sap

Sap tools – falt file upload,smart sap smart data access,smart data stream

External tools – slt,srs,dxc,sds

Sap landscape transformation – its mainly to migrate the hana data from non sap

Sap replication server – mainly for Sybase database as transaction data of ddl , dml data into data base from non sap, with zero down time, low impact