SAP → Systems, Applications & Products used for data Processing

* SAP ABAP TRAINING *

BY- Sailee R. Sarkar (SAP CO-PC)

* DAY 1 – ① Introduction to SAP
② What is ERP?
③ SAP Modules
ⓐ Functional Modules
ⓑ Technical Modules
ⓒ & Admin Modules
④ Role of ABAP Consultant
. RICEF [Details]
⑤ SAP R/3 Architecture
⑥ ABAP Programming Rules.

* SAP - ABAP

* ~~DAY 2~~ – .

Basic ABAP (Core ABAP) · 3 months. | OOPS ABAP · 40 days | Cross-Appl[c] 40 days

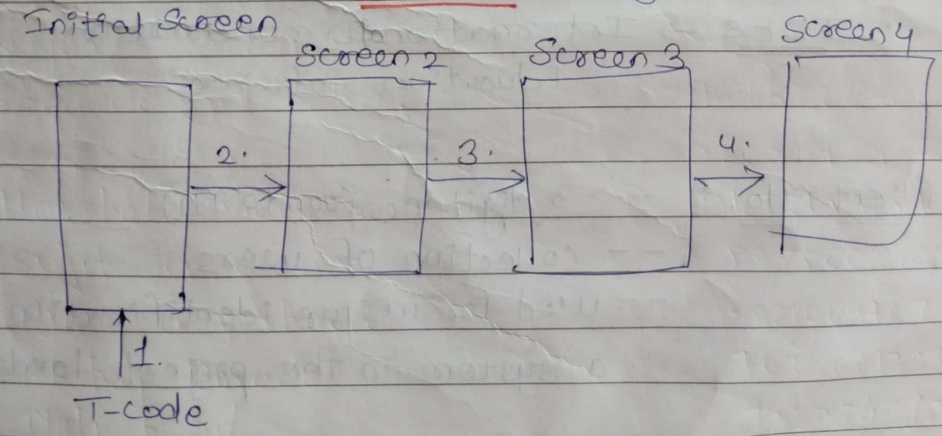
. T-Codes (Transaction Codes)

. What is T-code?

SAP contains different modules for e.g.
- SAP FI/CO — Finance dept. — 100 transactions
- SAP SD — Sales dept.

* Transaction - Collection of screens where you perform business operation.

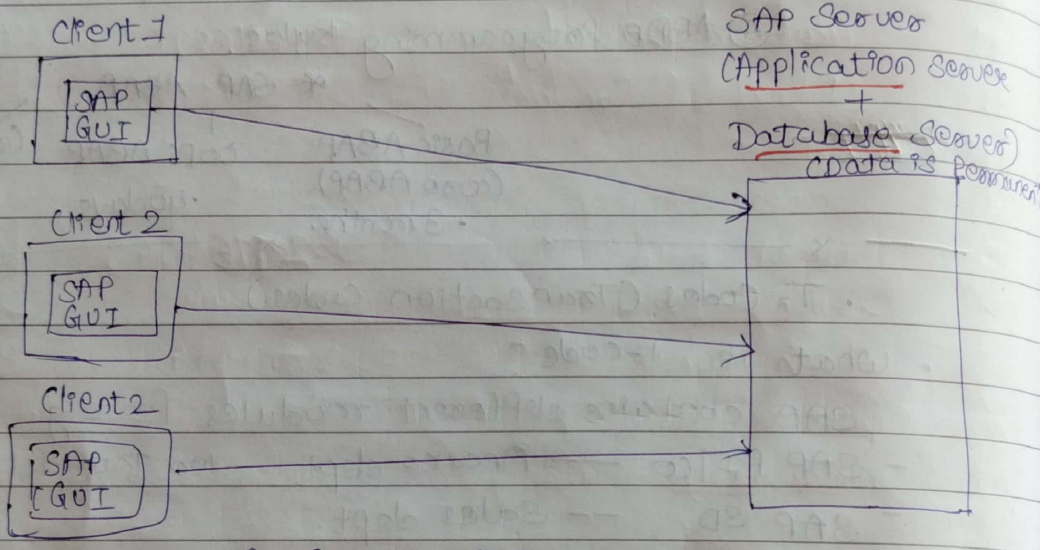
Transaction ⇒ Navigation of screens e.g. ATM screen



- SAP ABAP ⇒ To develop object ABAP is used.
⇒ To develop objects ⇒ requires tools
n no. of tools are in SAP eg. abap editor, screen painter, menu painter

- So for each tool is ABAP 1 transaction code is provided.

  e.g. ABAP editor - se38
  Function Builder - se37
  Screen Painter - se51
  menu painter - se41
  Form Painter - se71

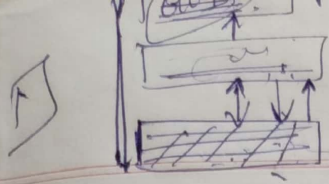


SAP GUI ⇒ SAP Logon Slw (SAP ECC 6.0)

EHP ⇒ Enhancement Pack ECC ⇒ Enterprise Central Component

- Connect to SAP Logon Slw
- Logon
- IDES ⇒ International Demonstration Education Software

- Client No. ? -- 3 digit no. (000 to 999)
  -- collection of users
  -- used for unique identification of a system in the project landscape

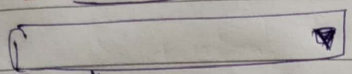


R/3 Architecture

* SAP logon in Multilingual:
  e.g. English - en
  German - de
  * Default language configured is English.

* SAP Easy Access Screen : (Initial Screen)

Menu Bar ⇒ Menu Edit Fav



⇐ Standard tool Bar

Title Bar ⇒ SAP Easy Access

Application Tool Bar ⇒

Screen Area

Status Bar ⇒

* To start writing program open se38 : ABAP editor

*. Standard objects -- Provided by SAP (builtin) --
  starts with other than 'z' | 'Y'
  -- Read & execute -- to modify -- need access key
*. Custom objects | zee objects -- new objects developed by ABAP const. on req. of customer.
  --> starts with 'z' | 'Y'.

* Steps to create a Program :

(1) Name the program with 'Z' (zee) or 'Y'.
(2) Click on 'create' (** use _ as separator / No special char' used in naming **)
(3) Pop-up window appears
   (a) Title ['Title can be given any']
   (b) Created 'Owner'.
   (c) Attributes
      ▽ Type ⇒ Executable program.
   (d) Save

(4) Pop-up window appears.
   (a) Package ⇒ It is a folder or collection of files or objects.
   [Empty]
   (b) Local object -- It stores program by default in package $tmp by SAP.
   * The objects stored in local objects cannot be transported.
   $tmp (Default package provided by SAP).

(5) ABAP Editor opens always
   * Executable program starts with REPORT keyword.
   * Every statement must end with period.
   * ABAP is space sensitive. (Not CASE Sensitive)
(6) To display welcome msg program.

   write 'Welcome to SAP'.
   keyword
   * String in ABAP is written in single quote.

(7) Four steps to execute the program in ABAP.
- (a) Save
- (b) Check for syntax error
- (c) activate [In order to transfer object from one system to other it is needed to activate]
- (d) Execute

(8) The screen on which the output is displayed is called 'LPS' (Liste Processing Screen)

* '/' is used to put statement on next line.

* Executable Program : ABAP Editor SE38

* Program 1 : Perform sum of two numbers.

```
REPORT Z1208PROG1

data a type i.
data b type i.
data c type i.
```

Declaring variables

* data is keyword in ABAP to declare variable. eg: (a,b,c)

* type is keyword in ABAP which indicates what data type is to be declared

: is used to do multiple operations using single keyword

```
data : a type i, b type i,
       c type i.
```

* Shortcut key to comment statements ctrl + <
  uncomment ctrl + >

* " is comment

```
a = 10.
b = 20.
```
} Initializing variables

```
c = a+b. - Performing calculations
write / c.
write : / 'Sum of two numbers is' c left-justified.
```

* Maintain one space in variable.

[left-justified, centered] options/additions

* Numeric data types are right justified - They are assigned to space -
* Char data types are left justified - They are assigned to space -

* To know any detailed information about any key word in ABAP click on keyword & press f1. It will direct to ABAP keyword Documentation

* uline keyword is used to separate the output

-

• Core ABAP - Part 06

* How to develop selection-screen ? (To make the customer to select the inputs)

Program 2: Z1208 Prog2

```
REPORT Z1208Prog2
Parameter variable starts with P_
Parameters : P_x type i,
             P_y type i.
```

[Parameter variable name cannot be exceeded than 8]
[No space is used]
[No special character cannot be used]
[Only underscores is used]

* To provide sentence on the selection screen follow the ~~follow~~ below process.

Goto → Text Elements ⇒ Selection Texts

In selection Text enter text

P_x - Enter first No.
P_y - Enter Second No.

Save
Activate

* To provide values on selection screen use Default keyword

```
Parameters : P_x type i Default 10,
             P_y type i Default 20.
```

od to o

~~data~~ = ~~result~~

data v_z type i.

v_z = p_x + p_y.

write :/' sum of two no's is', v_z.

* To use Obligatory : (Obligatory is used to make the field mandatory on the selection screen)

Parameters : p_x type i Default 10 Obligatory,
p_y type i Default 20 obligatory.

* To use string variable

Parameters : p_x type i Default 10 obligatory,
p_y type i Default 20 obligatory,
p_str type string.

write : /' Entered string is', p_str.

* lower case keyword.

* To change the background color of the statements on LPS used keyword color
Color 1 to 7 are only available.

* To switch ON/off Both colors use
format Color 6.

:

Format color off.

* System Fields [These are the fields which are already defined by SAP] hyphen
[They always starts with sy- **

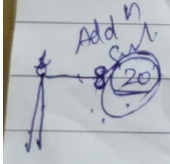
• Core ABAP - Part 07 :

* Program 03: • To put radio buttons on selection screen.
- Addition
- Difference
- Product
- Division
- None

• Validations :
1. By default, the last radiobutton should be selected
2. At any point of time, user should select only one radiobutton
3. On selecting one of the radiobutton & on clk of execute button, identify the selected radiobutton & perform the appropriate operation
4. Both the input fields should be mandatory.

Parameters : p_x type ; default 20,
p_y type ; default 15. (or)

• The above statement gives the o/p on selection screen on extreme left.
• To start the Text on selection screen as required use the following:



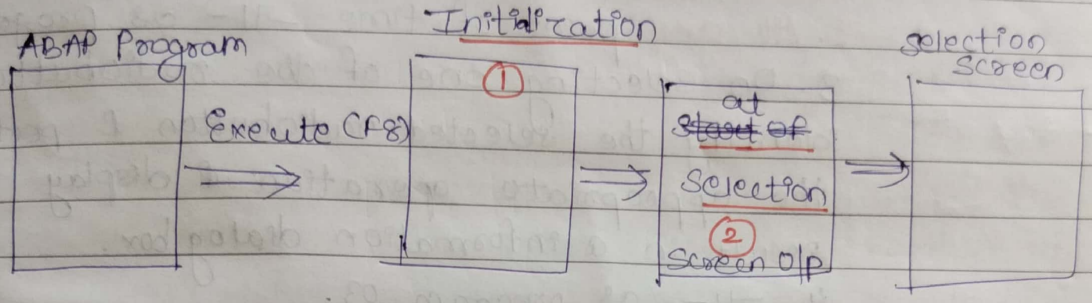
selection-screen begin of line. (can give any name)
selection-screen comment (8)(20) /b1.
Parameters p_x type ; default 20.
selection-screen end of line.

Repeat for second line (p_y type ; default 15: /b2.

· EVENT HANDLING:

Event - Event is an action, which is performed during run-time in SAP ABAP.

· By handling events we can provide dynamic features to our applications.



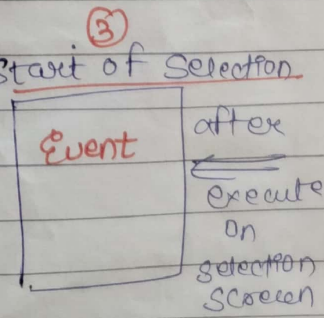
· The first event triggered is Initialization.
· So it is necessary to handle it.
· Events are always handled at the end of program.

```
Initialization.
    lb1 = 'Enter first No.'.
    lb2 = 'Enter second No'.
```

* To design radio buttons:

```
Parameters : p_rb1 radiobutton Group grp1,
             p_rb2
             p_rb3
             p_rb4
             p_rb5
```

· To handle start of selection.



Start of Selection event occur.

* Core ABAP Part 08:

Program 04:

Validations:

1. By default, the last radiobutton should be selected.
2. At any point of time —11— as program 03.
3. On selecting one of the radiobutton identify the selected radiobutton & perform the appropriate operation & display the result in a information dialog box.
4. —11— as program 03.

* As the validation for prog. 03 is same in program 04 except (3) point we can copy same program & edit in that.

· Steps to copy program:

1. ABAP Initial screen (The program should be active.)
2. Click on copy
3. Give name
4. Copy
5. Some checkboxes appear select all
6. Copy
7. Package ⇒ local object.

* Concept of function code:

* To identify the interacted element in program use function code. (means which button is interacted with user during run time to identify that each button is assigned function code).

· When the user click push button on selection screen an event called ~~AT SCREEN~~ "AT SELECTION SCREEN" occurs.

· How to assign function code we need to use the system fields assigned by SAP which starts with sy-

· sy-ucomm = It contains the value of function code stored in it.

· For radio button event triggered is "At selection screen on radiobutton group".

· To assign function code in program use keyword "USER-COMMAND". name of the function code.

* Core ABAP Part 09:

Program 05:

· Validations: checkboxes: (To allow user to select multiple options)

1. Input fields should be defaulted with some initial values & they should be mandatory.
2. On selecting one or more checkboxes & upon click of execute button, perform the appropriate operation & display the same in L.P.S.

- Debugging rules:
  1. Breakpoint can be set only to executable statements & not to the variable declarations. F5 - To go to next line in debug
     F8 - To come out of debug

* Program 06: Checkboxes on selection screen & display the result on execute button.

* Program 07: Pushbuttons on selection screen.

* Core ABAP: Part 10:

- Looping statements:
  1. while - endwhile
  2. do - ~~dowhile~~ enddo
  3. loop - ~~loop at~~ endloop.

- Program 08: looping statements.
  To design a program which gives the result as multiple of 6. excluding 5th Iteration.

  6 * 1 = 6
  6 * 2 = 12
  6 * 3 = 18
  :
  6 * 10 = 60

- Continue keyword: The use of continue statement is used inside loop, the rest of statement will not be executed. (They will be bypassed)

  Exit
  ↓
  After exit the program will come out of the loop Not out of program

* Core ABAP : Part II

Program 09 : Calling Program

To design a program by using SUBMIT keyword.

SUBMIT is a keyword which is used to call a another program from existing program.

Note : Only program can be called but the output shown on L.P.S. is only of one program at a time.

- RETURN keyword is used to return to the calling program.

Program 10 : Called Program.

Program 11 & Prog. 12 :

- The Program 11 & 12 are designed to perform sum of two no's.

Prog 11
↓
will handle I/O operations

Prog 12
↓
will handle Business logic (Calculations)

* Two program are combined & can exchange info. by using keywords.

- SUBMIT
- IMPORT
- EXPORT

* Note : ABAP memory is session specific.

* In ABAP Integer occupies 4 bytes memory & By default value is '0'.
* Characters default value is space ' '.

* Core ABAP : Part 12 ⇒

11

- Concept of Structures: Program 13 : To design a program to store employee infor. using structure.
  - A structure is a user defined data types.
  - A collection of different types of fields.
  - To store interrelated data e.g. employee details structure are used.
  - All the structure fields are stored one after the other in memory, so accessing speed will be faster in structure. (memory allocation)
  - A structure is used to store record type information. [interrelated data]
  - One structure holds one value/record.

- like keyword : To copy fields of two structure
- '=' assigment operator : To copy data between two structure
  - 'move' keyword
  - 'move-corresponding' keyword
- 'clear' keyword : To clear the contents of structure.

- Difference between '=' 'move' & 'move-corresponding'

① '=' assigment operator : It is used to move the data between 2 structure if whether their field names are different. But the data types of fields must be same.

② 'move' ~~stam~~ Statement : Gives the o/p same as '=' assigment operator.

③ 'move-corresponding' statement : Copies the data in structure which has the fields names are same irrespective of operator.

* Core ABAP : Part 13 :

- Nested Structures :
  To declare structure within structure.

* Program 14: To design a program to declare nested structure.

- Including Structures inside other structures:

* Program 15: Including structures.
  - Include structure name can be local or global.

- Standard syntax for declaring structure:

* Program 16: Standard syntax for declaring structure.

Create types declaration
types : Begin of [field-name],
field name 1
field 2
end of [field-name].

* Then access types using work area to assign memory. Because types declaration create template & dont allocate memory. So memory allocation is done using data keyword.

* field-Symbols keyword.
Program 17: Use of field-Symbol variable.

field-Symbol : It is a special ~~variable~~ data structure which can store any kind of data e.g. i, c, str
- < > It is declared in angular bracket.

• whenever you want to assign a variable value to field symbol use assign keyword.
• After assign keyword a pointer link will be established betw reference variable & field-symbol variable.
• Any changes made to either of the variable it will reflect in both variable.

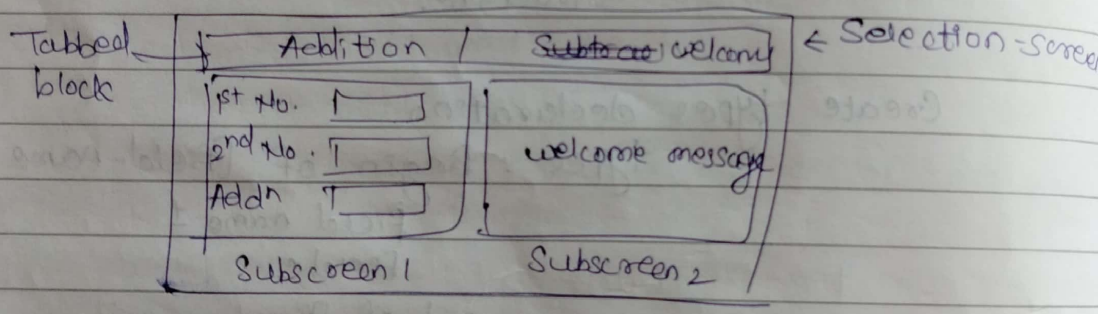
* Core ABAP : Part 14 :

Selection-screen tabbed block:

It is a collection of tab buttons.

Program 18: Selection screen tabbed block

• Tabbed block is a block on selection screen with different tabs & subscreens.



Imp

• Screens should always be given some number between 1 to 999
* 1000 is reserved for selection-screen.

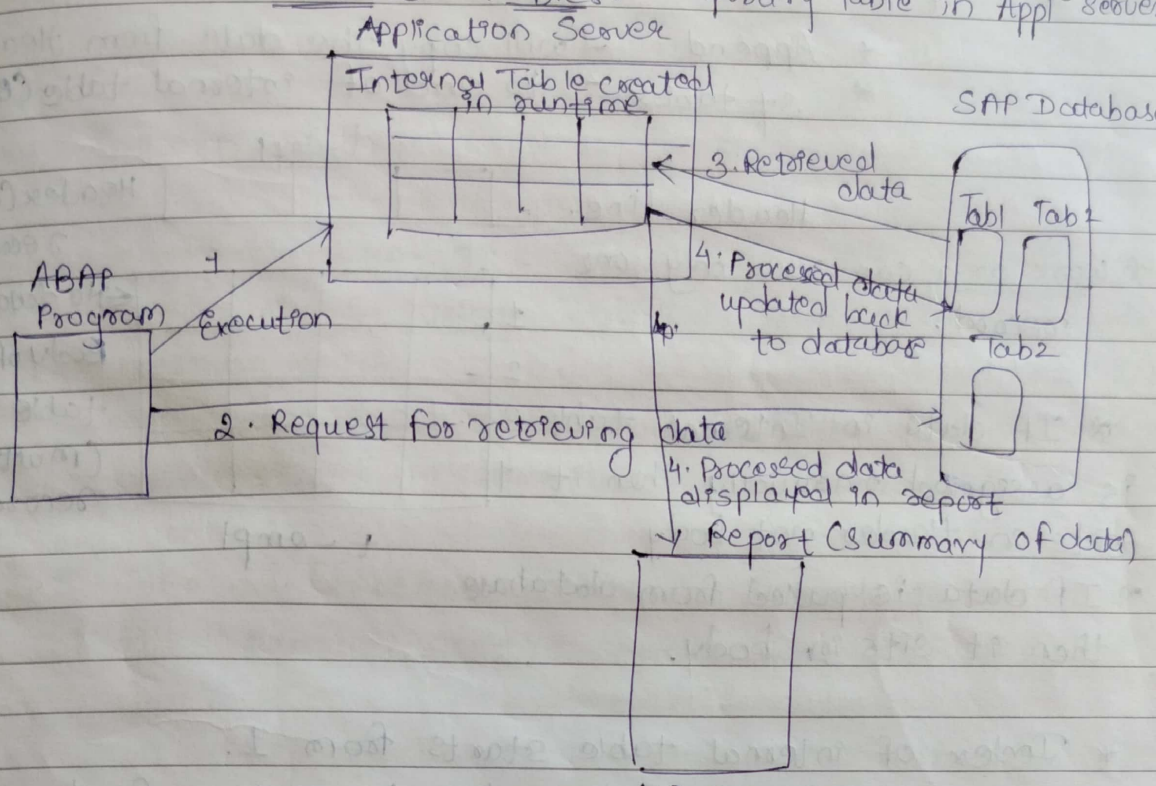
* Properties of tabbed block:

① tbl - activetab - which tab to select by default
② dynnr - which subscreen to select
③ Prog - which program to select

- Why to generate REPORTS?
  By analyzing reports we can take some business decisions.

✱ Core ABAP : Part 15:

- **Internal Tables:** Temporary table in Appl^c server.



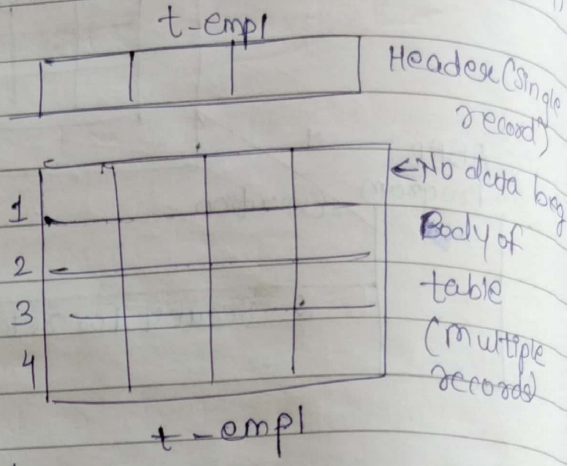
- **Database Tables:** Resides Tables in database permanently to store the data in the form of rows & columns.
- ABAPer needs to develop/generate Report based on the data in database Server.
- By analyzing data business decisions are taken.
- Internal Tables are created during runtime in appl^c server when we execute ABAP program.
- Internal Tables are temporary & holds multiple data

Internal table Classification

- Indexed
  - Standard Internal Table (Default)
  - Sorted Internal Table
- Non-Indexed
  - Hashed Internal Table

** Program 19: To design the employee data using Internal tables.

* Append - will copy the data from Header line to end of internal table (Body)
* sy-tabix - to end of internal table



* work area can save only one record.
* If data in Internal table is assigned manually then it sits in Header not body.
* If data is pulled from database then it sits in body.

* Index of internal table starts from 1.
* We can refer to Index value by system field called sy-tabix which refers to index position in internal table.

* Describe Table : Counts the number of records in internal table & stores count value in system field called sy-tfill.

* Always use Describe Table & sy-tfill one after other in combination.

* Clear - normal variables, work area, int. table

Body
* refresh - only on internal table Clear internal table & doesn't deallocate memory,
* free - only on internal table, clear & deallocate memory.

. Memory management type or performance type 'free' is preferred.

* Core ABAP : Part 16 :

* Program 22 : To design the employee data using Internal table without header.

* Append & Insert copy the data at the end of internal table body.

* The difference between Append & Insert is that, Append always copies data at the end of Internal table whereas
Insert can copy the data at the end of internal table or at specified position by using Index.

* While declaring the internal table we can specify the key field. key can be unique key or Non-unique key.
* Unique key has no duplicate values.
* Non-unique key can have duplicate values.
* Standard Internal table can be designed with key or without key. key is not mandatory.
* For standard Internal table can be given only non-unique key. It does not support unique key.

* Sort keyword : Used to arrange data sequentially.

* Core ABAP : Part 17 :

* Program 23 : Copy of Prog. 22 & only adding the use of additional keywords.

① Read Table : To read specific record from the internal table.

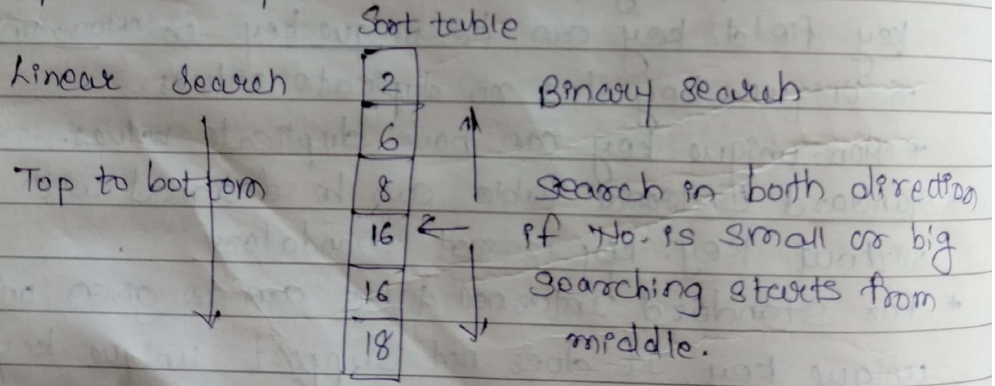
Syntax : Read Table Table name.

② Transporting : To read the table partially means only to read some fields of Internal Table.

③ Sy-Subrc : System field to check whether previous statement is executed. If executed sy-subrc = 0 else sy-subrc = 4.

Syntax : If Sy-Subrc eq 0.
else.
endif.

④ Binary search : To search the record quickly & improve the performance.



⑤ Transporting no fields : To check only the existence of record.

* Note : Sy-tabix : Used only for internal table to known the index position in it.

Sy-index : Used for normal looping statement while & do to known the iteration position.

* Core ABAP : Part 18 :

* Program 24 : Sorted Internal Table : while adding SAP will arrange data in sort sequence.
* Insert statement is used in Sorted Internal table.

Standard Internal Table	Sorted Internal Table
1) Key specification field is optional.	1) key specification field is mandatory.
2) It supports only non-unique key.	2) It supports both unique & non-unique key.
3) Both Append & Insert can be used.	3) Both Append & Insert can be used. But while using append sort sequence & duplicate entries must be checked or it will lead to runtime error.
4) Sort statement can be used.	4) Sort statement cannot be used.
5) Read Table statement can be used.	5) Read Table statement can be used. (supports Index search)
6) Supports Linear search.	6) Supports Linear search
7) Supports binary search.	7) Supports binary search.

* Program 25 : Hashed Internal Table.

- ~~Hars~~ Hashed Internal Table :
  Uses hash algorithm. - indexing is not possible
  - doesn't support explicit / implicit index operations

- implicit index operations :
  Append not supported.
  Read worth binary not supported.

- explicit index operations :
  insert with index not supported
  Read with index not supported.

- Hashed Internal table key specification is mandatory & it supports only UNIQUE key.

- Sort statement supported.
- Linear search supported.

- When to use hashed Internal table :

  In std. Int. Table & sorted internal table as no. of records increases searching time increases.

  But in hased Internal table the searching time is constant & does not depend on the no. of records.

  So for handling huge data use hashed Internal table.

* Core ABAP : Part 19 :

. Modularization Techniques :

Modularization is the process of breaking the program into smaller blocks.

block : set of statements which is defined only once and can be called any no. of times.

. Advantages of modularization :
1. Increases reusability.
2. Maintainance cost is decreased
3. Increases readability.
4. Efficient control on program flow.
* Easy to read & maintain the code.

. ~~Types of~~ modularization can be achieved by using :
1. Subroutines.
2. include programs.
3. function modules.
4. methods.
5. macros (used in HR-ABAP)

1. Subroutines : First Define subroutine & then call subroutine multiple times. [Prog. 26]
. Subroutine is defined at the end of program.
. Subroutine is defined as

```
Form name of subroutine.
      :
      :
Endform.
```

. Subroutine is called by using
Perform name of subroutine.

. Instead of calling repeated statements we can use subroutine & define statements only one & then call multiple times.

- Internal subroutine : Which is defined & called in same program.

- External subroutine : Which is defined in one program & called in other program.

* Changing keyword : To return the values from subroutine.
  - Any no. of parameters we can use to return values from subroutine.

- Actual parameters : Declared in the calling function of subroutine.
- Formal parameters : Declared in the defination of subroutine.
- Both names can be different.

* Pass by value & Pass by reference in subroutines

① Pass by reference : The address of actual parameters is copied to formal parameters. Therefore there is a link between actual parameters & formal parameters.
If actual parameters changes formal parameters also changes.
Syntax : Perform sub using V_x V_Y.

② Pass by value : The actual value of actual parameters is copied to formal parameters. There is no link betw actual & formal parameters.
Syntax : Perform sub using value (K1) value (K2).

* Core ABAP : Part 20 :

- Pass internal table as parameters to subroutines. [Prog. 27]

* Note: To pass internal table as parameters to subroutines use tables keyword.

* Core ABAP : Part 21 :

[Program. 28 & 29]

1) Internal Subroutines : defined & called from same program. [e.g. Prog. 27]

2) External Subroutine : defined in external programs & called from other programs.
external program (another executable program/ subroutine pool) [e.g. prog 28, 29]

- Subroutine pool : It is the collection of one or two subroutines.

- To create Subroutine pool [e.g. progsubpool]
Container of external subroutine
(any no. of external subroutine can be created)

- Type of program - subroutine Pool - starts with PROGRAM keyword.

* Note : Only Executable program can be executed by F8.
Subroutine Pool cannot be executed.

* We can call now subroutine pool in any different program by using following syntax.
e.g. perform sub1 in program [name of subpool] or
perform sub1 (z1208subpool)

2. Include Programs : (2nd Modularization Technique)

- Include programs can contain global variables + subroutine defn + module defn.

- To create include program select Type of object Include program

- Include Program starts with no keyword.
  - To use include program in other executable program use syntax : Include ename of program)
    At the begining of the program always.

- While including subroutine & module pool in include program using event start-of-selection immediat. after include prog.

* Core ABAP : Part 22 :

3. Function Modules :

* FM & RFC
- For FM select option in attributes as normal F.M.
- For RFC select Remote-enabled modu

Function Modules are used to define block only once & can be called 'n' number of times similar to subroutine.

- The difference between subroutine & function module is 'function module is created using function builder tool' (SE37)

- Types of F.M (1) Standard F.M. (Predefined by SAP)
  - Can be called explicitly by developer.
  (2) Custom F.M. - defined & called by developer.

* As part of all the repository objects (executable prog, include prog, subroutine pool) are stored in database table 'TADIR'
* TADIR has standard as well as custom repository object

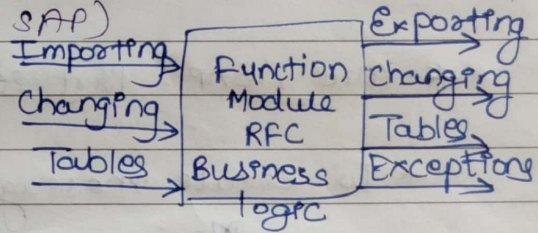
* Function Group : It is used to store 'n' number of function modules.
  - Function modules are always stored in Database table 'TFDIR'.

* All the database table in SAP can be accessed by using T-code SE11. (ABAP Dictionary)

- function Group - IBIP - F4-filename
  - SDHI - F4IF_INT_TABLE_VALUE_REQUEST
  - SFES - GUI_UPLOAD
* SE80 - Object Navigator. to check function group.

* To create F.M. First create function group.

* Function group is created in SE80.
whenever we create function group SAP will automatically create 2 include programs.
  (1) End with TOP - top include program
  (2) Ends with 'UXX' - reserved for SAP (internally used by SAP)



* Core ABAP : Part 23:

* function module using returning values.
* Function module using changing parameters.
* function module using exceptions.

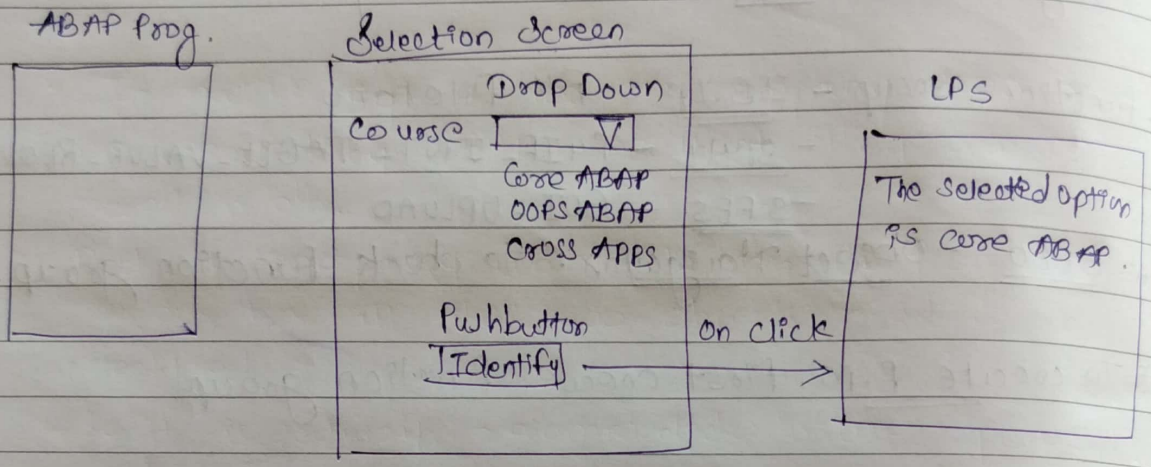
4. Macros : If we want to reuse same set of statements multiple times in a program, we can include them in macro. A macro can be called only in the same program in which it is define. Macro definition should be before in the prog, then the actual macro used.
We can also use parameters inside the macro

* RFC (Remote Enable function modules): Function modules are called within the same sap system, whose it is define, whereas RFC can be called from same or different sap system.
RFC is very easy to use integration option for connecting multiple systems with less efforts to other integration option as IDOC, API etc.

* Core ABAP : Part 24 :

* Standard F.M: Defined by SAP
called by Developer

Program 34:



program use

* To design the above function module VRM_SET_VALUES

* Core ABAP : Part 26 :

* To create message class: Message class is container of msgs
. Message class is created when certain messages are repeatedly used in program.
. T-code SE91 is used to create message class.
. In message class 1000 msgs can be given from msg ID 000-999.
. Syntax for using message class in program

message <type of message> <msg ID> (<msg class name>)

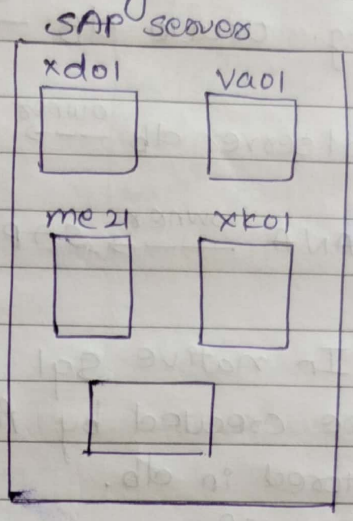
e.g. message I000 (Z1208msg class).

* Core ABAP : Part 27:

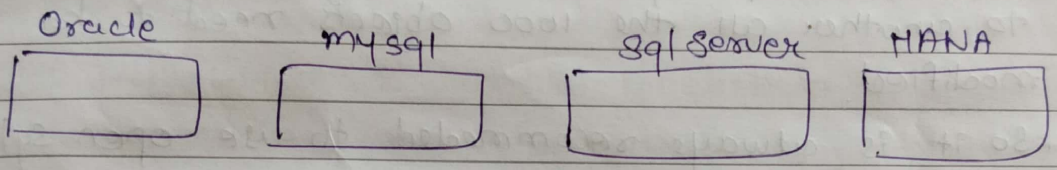
Database Programming.

- xd01 - Creating new customer
- va01 - create sales order
- me21 - create purchase order
- XK01 - create new vendor

→ Predefined Transactions.



- Data captured in any transactions get stored in SAP database permanently.

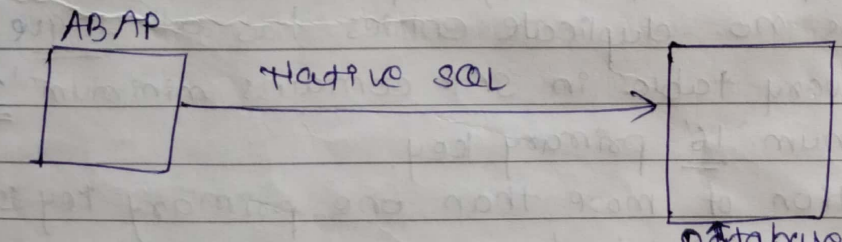


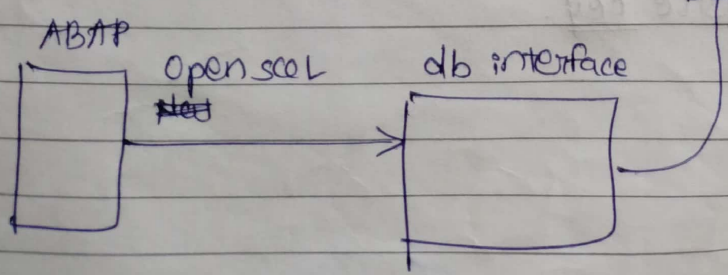
- Customer can choose any database e.g. SAP + oracle
  ← SAP + mysql
  SAP + HANA

Front end server

- ABAP Consultant needs to develop REPORTS. On executing reports data is displayed in summarized format for analysis.
- ABAP Program --→ Interact with database system →→ Performing CRUD operations.
  - C - Create, R - Read, U - update, D - delete

* How to Interact with Database ? SQL - Structured Query Language





* What is open sql & Native sql ?

e.g. oracle db —owner→ oracle corporation —release→ native sql statements

sql server db —owner→ Microsoft —release→ native sql statements

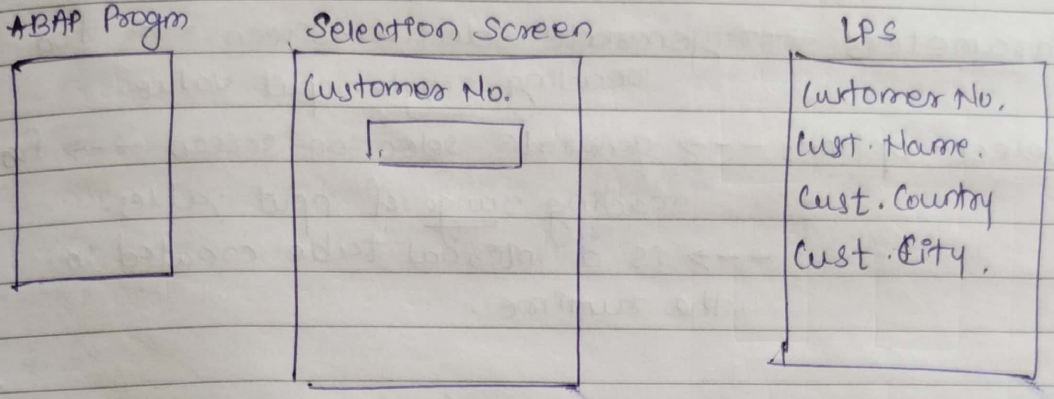
HANA —owner→ SAP → native sql statement

- In native sql if for any company 1000 reports are created by ABAP consultant they are directly stored in db.
- But if company wants to migrate from one db to another all the 1000 objects need to be modified.
- So it is always recommended to use open sql.
- Adv. of open sql is that ABAP consultant need not to worry about db because it is directly converted to native sql by db interface.
- Maintenance cost is less in open sql.

* Enhancement of the table : To plug-in some fields from std. db table to meet the customer requirement is called 'Enhancement'.

* Primary key : The primary key field is the one which has no duplicate entries has all unique entries. Every table in SAP contains minimum '1' & maximum '16' primary key.
- Combination of more than one primary key is called 'Composite key'.

* To develop ABAP Program



* Develop db program using open sql.

* The limitation with the above program is ① the database tables fields length & data type is declared in program itself, then if the data type & length is changed then all the program need to change which will increase maintenance cost. (Hard code the data) * Select Single is open SQL

② On selection screen we need to enter the customer no. but we cannot remember all the customer. So we will require some f4 help of selection screen

So to overcome the disadv. of open sql write program in Native SQL.

* Core ABAP : Part 31 :

- Parameters --> generate selection-screen --> for reading single input value.
- Select-options --> generate selection-screen --> for reading range of input values.
  --> is a internal table created in the runtime.

- Standard tables --> KNA1 (Customer masters data)
  --> MARA (material master data)
  --> VBAK (Sales document header data)

* Watchpoints : Watchpoints are the conditional statements defined in debugging mode.
Whenever watchpoint condition is meet the program execution will be halted.
Maximum 6 watchpoints can be defined.

* Core ABAP : Part 32 :

- VBAK --> Sales document header data
- VBAP --> sales document item data
  Both tables have a common field Vbeln
  Vbak. Vbeln -- Vbap. Vbeln

- Types of Reports : 1. Hierarchical Reports - Parent & (In database program) child

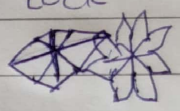
21/11/2023

* Steps to create Database Table : (customized)
  1. Use T-code SE11
  2. Put name of database Table.
  3. Clk create
  4. Put short description i.e. name of table
  5. Delivery class - A
  6. Data Browser - Display/Maintanence Allowed

* Second tab fields -
  Enter field name - 1. MANDT
  Double clk on field name - Save
  7. Enter Package ZPACK.
  8. Enter Transport No.
  9.

22/11/2023 Morning

* Lock Objects : They are used to provide Lock on the database table.



* Search Help : Defines Input Help (F4 Help) for the fields of DB-Table

Types : 1. Elementary Search Help - Single value
  2. Collective Search Help - Group of value

* Views : Retrieves the data from database table.

ZCompDept - ZDept Data
ZECEDept -

1. EmpNO.
2. EmpName of Emp
3. Address
4. Contact No.
5. Email Id
6. Salary
7. Subject
8. Qualification

ZCollege Data

1. Name of Employee
2. Name of College
3.

} Package
function group

SCOE I

SMARTFORMS

* 1) How to select a page type i.e. A4, A3
2) How to set the page orientation portrait / landspace
3) How to create smart style with different font size / colors.
4) Print text ~~use~~ element using smart style.

SMARTFORMS.

SE37 - FM

SE78 - T-code to upload the image in smartforms

---

* ZAFAR K. videos Syllabus:

1. What is SAP? SAP Overview
2. SAP implementation cycle? Role of SAP consultant
3. What is SAP ABAP?
4. SAP ABAP data dictionary SE11 - table overview
5. SAP ABAP data dictionary SE11 - table creation
6. SAP ABAP data dict. create table exercise 2
7. ABAP data dict. create table with trng & transaction code
8. ABAP data dict. foreign keys & check table
9. Views in ABAP data dic. - 04
10. Structure in ABAP data dic.
11. Elementary, fuzzy & collective search help in ABAP
12. Create program in SAP ABAP editor -
13. ABAP Classical Reports
14. ABAP ALV Reports
15. ABAP debugging.

* SAP Applications developed using ABAP

R I C E F W

Reports → Interfaces → Data Conversions → Enhancements → Forms (SAP scripts, Smartforms, Adobe Forms)

Workflows

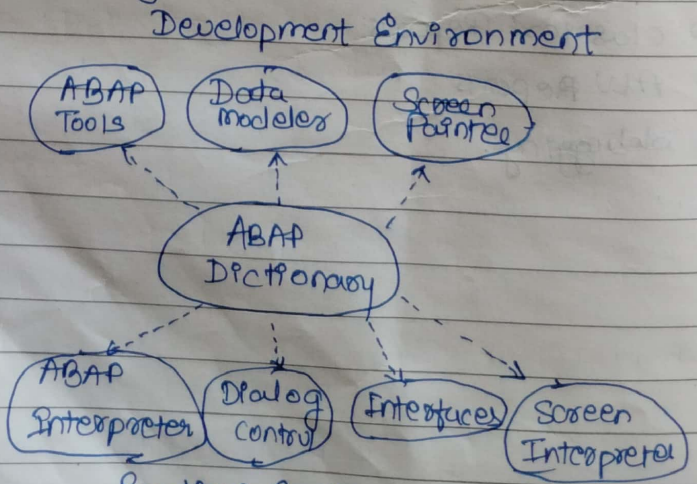
* Module pool programming

* Components of ABAP workbench

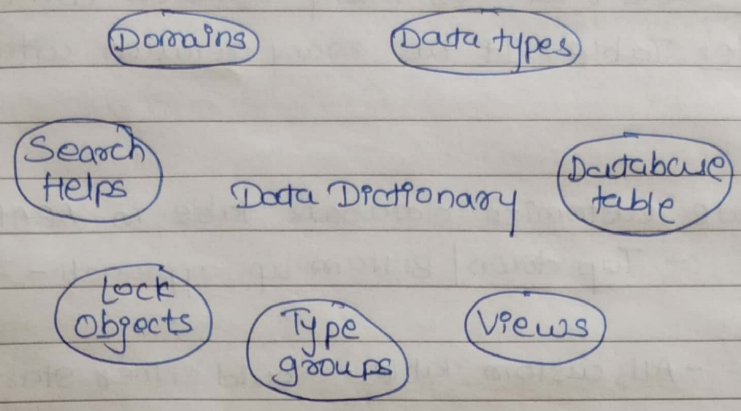
1) ABAP editor - SE38
2) Repository browser - SE80
3) ABAP data dictionary - SE11
4) Screen & Menu Painter - SE51 & SE41
5) Function builder - SE37
6) SAP Script - SE71
7) Smartforms - smartforms

* ABAP data dictionary: T-code SE11

ABAP dictionary is a tool provided by SAP for centrally create, view and maintain data definations related to database. We can use data dictionary to create objects like table & view & it is also use to create & maintain global user define types like data elements, domain, structure, table type etc. which can be used by ABAP program.



* What are the different components of ABAP data dictionary?



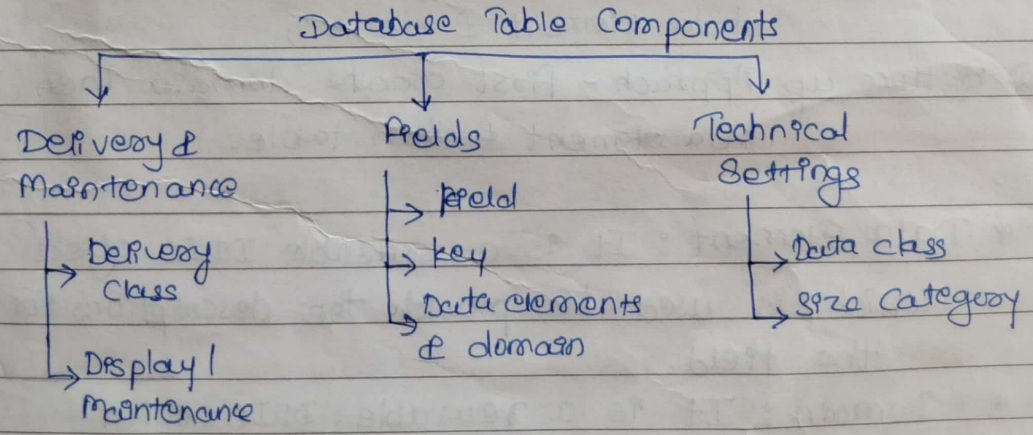
1. Database Tables:

Database Tables are the collection of fields which contain physical data. It is an object that stores data in the form of rows & column.

The different types of tables are

1) Transparent Tables
2) Pooled Tables
3) Cluster Tables

. key field define the unique combination of a record, which get stored in a database table. A maximum of 16 key field we can define in a table.



* We can create the field of database table without data elements & domain using pre-defined types.
But reusability is not possible by using pre-defined type.

1) Transparent Table : It has one relation with table in database.
2) Pool Table : It has many relation with table in db.
3) Cluster Table : It has many relation with table in db.

- To create customize database table in ABAP dictionary.
  - Top down / Bottom up approach - ZEMPLOYEE_MASTER
  - All custom table - should either start with 'z' or 'Y'

- key points while creating database Table :
  1) Database Table Contains fields ~~& Data Elements~~.
  ~~Data Elements~~ = Data & fields is containing
  Data Element (Description) + Domain (properties of field)

① Data type
  - Elementary Type
  - Predefined Type

② Field label
  - S, M, L

Domain
- ① Data type
- ② length
- ③ Value range (Fixed value, lowere limit - Upper limit)

1) Top down Approach - First create table & then data element & domain
2) Bottom up Approch - first create domain then data element & then table.

* Data Element : It is a reusable DDIC object which is used to provide the descriptions for the field.
* Domain : It is a reusable DDIC object which stores the technical properties of the field like data types & size

* In TMG Maintenance screens → maintenance type
1) One Step - For single screen display only
2) Two Step - For two screen overview screen - in that we can read, delete & update, & single screen - create record

* Table Maintenance Generator (TMG):

TMG is used in database table to maintain (change, create, delete) the records. (SM30 - is the T-code to maintain TMG)

TMG is given to end user in production system for maintaining the data.

TMG is given with the custom T-code.

The T-code is created in SE93

* Events in TMG: Total we have 39 events in TMG.

* Foreign keys and check table:

We use foreign keys to define relationships between tables in the ABAP Dictionary, create value checks for input fields. 1) How to create TMG 2) Screen field modifications 3) Events implementation 4) Assign a T-code to TMG

* Views in ABAP Data Dictionary:

A view is the logical table, virtual table (imaginary table) which contains no physical data, but is used to hold data during runtime, on multiple tables.

· Types of Views:

① Database views
② Maintenance view
③ Help view
④ Projection view

① Database views: A database view is created on two or more tables using inner join concept, we can only read the data from database view.

· For joining two tables using database views it is mandatory to have same datatype & length for the common field.

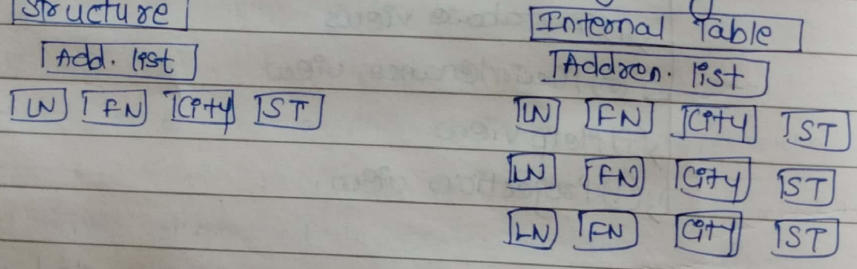
② Help views: Help view are created on two or more tables using outer join concept, we cannot directly execute help views, we have to include the help views inside the search help, & it is displayed when user press f4 help.

③ Maintenance view: A maintenance view are used for a sap internal purpose, in real time, we do not create maintenance views.

④ Projection view: A projection view is created on single table, it is used in case if we have large number of field & we want to provide only some of the fields as interface, in that case we use projection view.

* Structure in SAP ABAP data dictionary:

A structure is a group of components under a name. A structure can hold the data a single record only at time in program run time it acts as a work area, but it cannot store the data like a table. * Structure dont require primary key



- we can use it to perform internal table operation at program run time.
- It can also be used to display the data on the screen.

- Structure can be define with three types:
1) Flat structure: define with elementary types, individual comp one after
2) Nested structure: Define with elementary type alongwith anoth
3) Deep structure: Define with table types

* Elementary, Collective and fuzzy search help in ABAP data dictionary:

There are 2 types of search help

1) Elementary search help: specify a single help, it will give only single tab of selection.
2) Collective search help: It is collection of multiple search, with this on pressing F4, we get multiple tabs for providing selection criteria for a field value.

* ABAP Editor: SE38

keywords : 1) Write - To point the op in the program
2) NO Standard Page heading - To delete display the heading in op
3)

* Our own T-code can be given to a program (Report) in SE93, select option (Program & selection screen(report transaction))

* Variables, Constant ABAP program:

- Variables are data object that we have to declare in program, and when we execute the program a memory is allocated to the variables and that stored the data, & as the name indicate, the value of a variables can change inside a program.
- Variables we can declare using predefined types as C, I, N, P etc. or we can also declare using data elements or by giving a table with field name.
- Constants are use to assign a fix value to a data another object, value of constant does not change.

nts

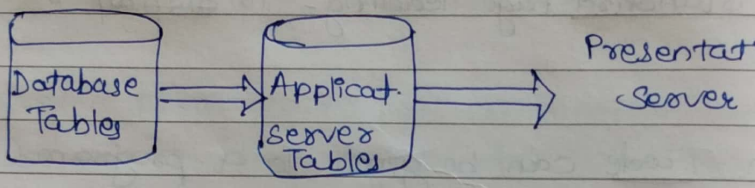
structure

- System-variables we can access the system variables like date, time, user, program name etc. using structure called 'SY'. SY-datum, SY-uzeit, SY-uname

* Internal Tables and structure (work Area) :

Internal tables are temporary table, which gets created on application server at program run time, that can stored any number of records, which we can select either from database or the records in an internal table can be generated during program run time.

Work area are used to access the record of an internal table, and it is use to perform different logical operations on a table record, an structure or work area can hold a single record of data, but they cannot store the data like table.



- Internal table operations -

1) Add record using append (it will add record at the end of the table)
2) Insert will add record at the end or if we give index, it will add the record at particular row.
3) Changing lines of internal table with modify with or without index, single or all values.
4) Delete - delete the records from an internal table.
   - Describe table - to count no. of records
   - Sort - Sort the record alphabetically
   - Delete adjacent duplicate

- Control Statements:
  IF, ELSE, ELSEIf, CASE

- Loop Statements: DO-ENDDO - To execute certain line of code at specified no. of times
  - DO-ENDDO, DO N times-ENDDO, While-ENDwhile.
  EXIT - to come out of loop
  Continue - to go to next record.
  } while-endwhile - this type of loop will continue process till the condition is true

SAP

* To see all the fields inside structure press Ctrl + Space
* To copy the fields of internal table press Ctrl + Y.

- String operations:
  1) Concatenate
  2) Split
  3) Find the length of the string - syntax: length variable = strlen (Name of variable)
  4) Replace charater - replace keyword or replace all occurences of
  5) Shift - shift keyword

* Control Break Statement:

Control break statements are used in internal table between loop & Endloop.

Control Break statements are

1. At First / End At : This block is executed at the first

* Open Sql in abap program:

Open Sql statements are used in abap to acces the data base tables declared in the abap dictionary.

- All dB have their own syntax to write a code.
- Open Sql supports all languages.

- Open Sql statements are
  1) Select : To read the data from dB
  2) Insert : To add the new records in dB
  3) Modify : If record exists with same primary key it will be modified else new record will be created
  4) Update : This is use to update the some fields value of an existing record
  5) Delete : To delete the record from dB.

- Open Sql statements follow 2 system variables
  sy-subrc -
  sy-dbcnt - No. of dB lines processed.

• Select query - ABAP Program

* select <fields> from <source> into <target> where <condition>

Fields - Here we have to mention the columns which we want to fetch, we use * to select all

Source - Database table name from which data to be select

Target - Internal table or work area in which data to be stored after selection from dB

Where - We have to give conditions here, on which basis data to be selected.

- There are various form of select statements
  - Select - endselect
  - Select *
  - Select field1 field2 field3
  - Select single ...
  - Select upto n rows
  - Select distinct - unique records of a column
  - Select using join (inner & outer join)
  - Select into and select into corresponding fields

* Insert <into db table> from table <internal table>
* Insert <into db table> from <work area>

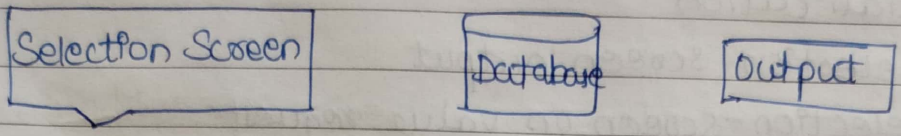
* Update <db table> set field1 = value1
  field2 = value2
  where xyz = abc

* Modify <db table> from <work area>

* Delete <db table> where xyz = abc
* Delete <db table> from <work area>

* ABAP Reports:

A report is a presentation of data in an organized structure. It is a one way communication based on the selection criteria, given data is fetch from database and display on the screen.



1. Classical Report
2. ALV report
3. Classical | ALV Interactive report

1. Classical Report: A classical report is created by presenting the data in output screen using WRITE statement. It is very simple & basic type of report without much functionality like download, sort, filter etc.

2. ALV Report: ABAP List Viewer, sap provides the table. function modules for displaying of the data, a ALV report has lot of inbuilt functionalities like sort, filter, download, ascending, desending, sum at different levels etc.

* Interactive report:

SE41 - Menu Painter
↓
Status - ZPF

* ABAP REPORT EVENTS:

ABAP programs are executed based on events. Following is the list of events which get triggered in a sequence, while program execution.

- Load-of-program
- Initialization
- At selection-screen output
- At selection-screen on value request
- At selection-screen on help request
- At selection-screen
- Start-of-selection
- End-of-selection
- Top-of-page
- End-of-Page.

* How to connect 2 Sap system with RFC.
* How to call RFC of one Sap system from another.

* Create RFC destination with SM59
* Create ABAP program.

* SAP ABAP ALV Reports:

- Function Modules in ALV Reports: We can make use of SAP standard function module for ALV report example as below:

1. REUSE_ALV_GRID_DISPLAY → used to display data in grid format
2. REUSE_ALV_LIST_DISPLAY → used to display data in list format.

- ALV reports include below in-built functions:
  - Sorting of records
  - filtering of records
  - Totaling of records when there is a quantity/Amount columns
  - Sub totaling of record
  - Hiding columns
  - Rearrange order of columns.
  - Downloading report in Excel or HTML format
  - Save different output layouts.

* Types of ALV Reports in SAP:
  1) Simple ALV report
  2) Interactive
  3) Editable
  4) Dynamic
  5) Blocked
  6) Hierarchical

* SAP Standard Reports:
  - MM60 - Material list
  - VA05 / VA06 - Sales order report
  - ME2L / ME2M - Purchasing document by vendor
  - FBL5N -

* Indexes on db tables: Index is the process of arranging the data in an order.
  - If the data is in sorted order then the searching would be fast.
  - If huge amount of data is there & if we want to search record it can be sorted by using Index.
  - Types of Index
    1) Primary Index - created by SAP (Primary
    2) Secondary Index - created by ABAP consultant (non-primary key fields
       a) Unique Secondary Index - No duplicate va
       b) Non-Unique -||- - Duplicate values

* When button is on selection screen application tool bar the event triggered is at selection-scree

* When button is on LPS event triggered is at USER-Command.

* Set PF-Status - is the command use to set the ~~insert~~ GUI buttons on Menu bar, Application tool bar or change or delete the existing buttons on standard tool bar.

* We create foreign key relationships between two tables to maintain data consistancy.
  Whenever the data on one table has to be
  * To create f.k. rela the two tables should have atleast one field which have same domain the field name may be different.

* In one of the
* 2) To create foreign key between two tables one is check table & other is foreign key table
3) In one of the table the (common) key field should be primary key which is called check table (Parent table) where we maintain unique data.
And the other table we create the foreign key which is called foreign key table.

key)

* When the foreign key is created between two tables it will not check the existing entries, it will check the (take care of) new entries
* Whenever we create foreign key relationship between two tables SAP will maintain referential integrity between two tables.

* Table Type: It is a reusable DDIC object which is an internal table structure created at Database level. Whenever we want to use same internal table structure across the program we use Table Type
e.g. of std. Table type LVC_T_DROP

* Parallel Cursor method:
In abap, the consultant use where clause in nested loop, this is very common but there is big issue with performance when using where clause
The problem with this method is for each record in the first internal table, when system checking for matching record in second table, it searches all records starting from index 1 to the last record.
To overcome the performance issue, there is a method called parallel cursor, in this method before the loop in second table, first we read the index of the matching record in second table, and then loop get

Starts from the index only and exit, when record not match. In this method for each record of first table system check & perform logic only for the match records of second table.

* Traditional method:

```
loop at gt_ekko into gs_ekko
      :
      :
loop at gt_ekko into gs_ekpo where
ebeln = gs_ekko-ebeln.
      :
      :
clear: gs_ekpo
endloop.
clear: gs_ekko.
endloop.
```

* Parallel cursor:

```
loop at gt_ekko into gs_ekko.
      :
      :
Read table gt_ekpo into gs_ekpo with key
   ebeln = gs_ekko-ebeln
if sy-subrc = 0.
loop at gt_ekpo into gs_ekpo from sy-tabix
if gs_ekko-ebeln ne gs_ekpo-ebeln.
 exit.
 end if.
      :
      :
clear
 endloop
endif
```

or optimizing the ABAP performance.

* To increase the performance of ~~SAP~~ ~~ABAP~~ program:

1) Use Inner join in your Select statement to retrieve the matching records at one shot

2) Avoid using select *

3) Avoid using ORDER BY in select statements because it may add additional work to the dB,

4) Avoid using nested loop using where clause instead we can use parallel processing

5) Can use Binary Search to speed up the search.

6) Avoid using select statements inside loop. because the no. of times select statement inc. n/w traffic.

* MODIF ID is used to group the selection screen elements.

* To associate custom f4 Help we need to use funtion Module F4IF_INT_Table_Value_Request

*

* Interactive classical Reporting:

* What is the purpose of module pool prog?
⇒ In TMG if we want to enable or disable any field or write code for TMG we use MPP.
To control the appearance of TMG the changes has to be done in its dependent objects

* Module Pool Programming:

Screen Painter (SE51):

- To create a screen in SE51 first we need to create a program in object Navigator (SE80).
- The program must be Top include one so that we can declare Global variables in it which will be visible to all screens.
- In type of program use Module Pool:
- Right clk on program name select create & then screen, give screen no. & save.
- Then Give short Description in Attributes & clk on layout of screen painter
- Start Designing
- Module Pool Program cannot be executed directly, it requires T-code. So we need to create T-code. (In creating T-code select option ① Program & Dialog)
- To activate the button on selection screen we need to write the flow logic.
- In MPP the events which occur after pressing button on selection screens are Process After IIP (PAI).
- Every Event in MPP is associated with one or more Module. Inside Module Defination the business logic related to event is written.

(Z1208MPP)

* BY using PBO & PAI we can control the appearance of TMG using module programming.
* We can make any changes in TMG by using module pool content.
* In MPP before the selection screen appears the event called 'Process Before OIP' is triggered.

* NOTE : To make the input fields invisible on Screen set SCREEN-INVISIBLE = '1'.
SCREEN-INPUT = '0'.
Both are the fields of 'structure' SCREEN