ABAP
Quick Reference

Ver. 2.1

http://help.sap.com
http://www.goudden.com

Predefined Elementary ABAP Types

Data Type	Initial Field	Valid Field	Initial Value	Meaning
Numeric types				
I	4	4	0	Integer (whole number)
F	8	8	0	Floating point number
P	8	1-16	0	Packed number
Character types				
C	1	1 - 65535	' ... '	Text field (alphanumeric characters)
D	8	8	'00000000'	Date field (Format: YYYYMMDD)
N	1	1 - 65535	'0 ... 0'	Numeric text field (numeric characters)
T	6	6	'000000'	Time field (format: HHMMSS)
Hexadecimal type				
X	1	1 - 65535	X'0 ... 0'	Hexadecimal field

Using Variable

```
CONSTANTS: MyConst  LIKE  TableT-Field VALUE 'MOM'.
DATA: MyVariable LIKE  TableT-Field.
```

```
DATA: BEGIN OF PERSON,
      NAME(20),
      AGE TYPE I,
END OF PERSON.

TYPES myPERSONs LIKE PERSON OCCURS 0. "With Header
DATA PERSON1 TYPE myPERSONs. "Type Line of myPERSONs
PERSON1-NAME = 'Michael'.
PERSON1-AGE  = 25.
APPEND PERSON1 TO myPERSONs. "Add record
```

BASIC ABAP Languaje Statements

<operator>	Meaning	<operator>	Meaning
+	Adition	/	Divide
*	Multiplication	DIV	Integral Division without Remainder
**	Exponentiation	MOD	Remainder after integral division
-	Substraction	><	not equal to
EQ	equal to	=	equal to
NE	not equal to	<>	not equal to
LT	less than	<	less than
LE	less than or equal to	<=	less than or equal to
GT	greater than	>	greater than
GE	greater than or equal to	>=	greater than or equal to

Internal Tables "iTable"

```
"Declare  iTable with direct fields
DATA: BEGIN OF iTableA OCCURS n
  field1 TYPE type_field1,
  field2 TYPE type_field2,
END OF iTableA.
```

```
"Declare  iTable from Estructure type
"Create Estructure
TYPES: BEGIN OF MyEstructure,
  field1 TYPE type_field1,
  field2 TYPE type_field2,
END OF MyEstructure.
```

```
"Create iTable with Header
DATA: iTableA TYPE TABLE OF MyEstructure WITH HEADER
LINE.

"Create iTable without Header
DATA: iTableA TYPE TABLE OF MyEstructure.
DATA: waA TYPE LINE OF iTableA.
```

```
"Declare  iTable with estructure from table and add
customs fields
DATA: BEGIN OF iTableA.
 INCLUDE STRUCTURE structure_name. "transparent table
 DATA field1 TYPE type_field1.
 DATA field2 TYPE type_field2.
DATA: END OF iTable1.
```

[Trying Internal Tables]

```
"Insert wa to iTable (single line)
APPEND  wa_A  TO  iTableA.
```

```
"Insert All lines of A to B
APPEND LINES OF iTableA[] TO iTableB[].
"Insert range lines
APPEND LINES OF  iTableA  FROM  nline1 TO nline2
                          TO  iTablebB.
```

```
"Insert wa in correct line of sorted iTable
APPEND  wa_A   TO   iTableA SORTED BY fieldX.
```

```
"Copy identical   iTables
iTableB[] = iTableA[].

"Modify especific Fields from wa to ITable
MODIFY iTableA FROM wa_A TRANSPORTING fieldX
                        WHERE fieldY EQ value.
```

```
"Order Content of iTable by field
"BP: SORT before any READ
SORT iTableA BY fieldA  DESCENDING.

"Read line of table on wa with key
READ TABLE iTableA INTO wa_A WITH KEY fieldX = value1
                                      fieldZ = value2
                                      BINARY SEARH.
"Read line of table on wa with number of line
READ TABLE iTableA INTO wa_A INDEX 3.
```

JUDGMENTS

```
IF gv_var > 0.
  " statements here
ELSEIF gv_var = 0.
  " statements here
ELSE.
  " statements here
ENDIF.
```

```
-- IF result IS NOT INITIAL.

-- IF NOT (a=1 OR b=2) AND c = 3.
```

```
CASE My_var.
  WHEN 'aa' .
    "statements here
  WHEN 'bb' .
    "statements here
  WHEN OTHERS .
    "statements here
ENDCASE.
```

```
DO.
  "Statements here...
  sy-index. "LOOP Counter
  IF abort_condition .
    EXIT.
  ENDIF.
ENDDO.
```

```
DO n TIMES.
  "statements here...
  sy-index. "loop counter
  IF abort_condition.
    EXIT.
  ENDIF.
ENDDO.
```

[Browse rows of internal table]

```
"BP: SORT it before loops
LOOP AT iTable INTO wa.
    "Statements here
ENDLOOP.
```

```
"BP: Love WHILE before DO
WHILE Value1 EQ Value2.
    "Statements here
ENDWHILE.
```

MESSAGE

[identify the message in table T100] - [Message Maintenance - SE91]

Type	Meaning	Behavior	Message appears in
s	Status message	Program continues without interruption	Status line in next screen
i	Information	Program continues after interruption	Modal dialog box
w	Warning	Context dependent	Status bar
e	Error	Context dependent	Status bar
a	Termination	Program Aborted	Modal dialog box
x	Short dump	Runtime error MESSAGE_TYPE_X is triggered	Short dump

```
                [ First Declare Class ]
REPORT name MESSAGE-ID class.
        [ Specifying the Message Statically ]
MESSAGE t nnn(id) [WITH f1 ... f4] [RAISING
exc].
        [ Specifying the Message Dynamically ]
MESSAGE ID id TYPE t NUMBER n [WITH
f1 ... f4] [RAISING exc].
        [ Filling Message Texts Dynamically ]
MESSAGE ... WITH f1 ... f4.
            [ Messages and Exceptions ]
MESSAGE..... RAISING exc.
```

Messages on help.sap.com

Common Transactions

SE37	Function Builder
SE38	ABAP Editor
BAPI	Business Object Repository
SE11	Abap Dictionary
SE16	Data Browser
SCID	Code Inspector
SM04	User Overview
SM51	AS Abap Instances
SM50	Overview Work Process
DWDM	Sample code in standard
SE10	Tools Transport Organizer
SM12	Lock Entries
SP02	List of Spool Requests
SM36	Define Background Job
SM59	RFC Connections
ST22	View Short DUMPS
SE16	Data Browser
SM30	Maintain Table Views
SE41	GUI Painter
SE24	Class Builder
SE39	ABAP Splitscreen Editor
SHMA	Shared Objects
SPRO	Cusrtomizing (press F5)
CMOD	Project M anag Enhancements
SE18	BAdI Builder
SE19	BAdI Implementations
SE95	Modification Browser Trasport
SE91	Message Class

[More Transaction Codes]

command	transaction
/oXXX	Call transaction, new session
/nXXX	Call transaction, same session
/o	Overview of sessions
/nend	End logon session
/nex	End logo session without save
/:	Delete current session
/ns000	To end the current transaction and return to the starting menu

System Fields

System Field	Field Meaning
sy-subrc	Return Code 0 = OK
sy-mandt	Logon Client
sy-uname	Logon Name user
sy-langu	Logon Language user
sy-datum	Server Date ABAP System
sy-uzeit	Server Time ABAP System
sy-tcode	Current Transaction Code
sy-repid	Name Current Program
sy-index	Loop Counter DO & WHILE
sy-tabix	record count of internal table
sy-listi	index of current list
sy-lsind	List processing, details list index

[More System Fields...]

TIPS

make modifications
On transparent Table

1. ***TX: Se16n***
2. ***Active debuging***
3. ***GD-EDIT = 'X'***
4. ***GD-SAPEDIT = 'X'***

Demo PROGRAM USING Macro

```
DEFINE PRINT.
  WRITE:/ 'Hello Macro'.
END-OF-DEFINITION.

"Use
WRITE:/ 'Before Using
Macro'.
PRINT.
```

OpenSQL

```
"Result of SQL query from transparent table
SELECT *      "BP:  use TableT~Field1...
INTO TABLE iTableA   "declare  with same structure
FROM TableT          " Transparent Table
WHERE TableT~Field1  IN  Values1
AND  TableT~Field2  =    Value2
AND ( TableT~Field3  LIKE '006%'
OR   TableT~Field3  LIKE '009%' ).
```

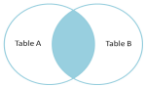
```
"Adding files to internal table without delete old files
SELECT *      "BP:  use TableT~Field1...
APPENDING TABLE  iTableA
FROM TableT              " Transparent Table
WHERE TableT~Field1  = <Value1>
AND TableT~Field2  = <Value2>.
```

```
" Select without LOOP on internal table
SELECT *  "BP:  use TableT~Field1...
FROM TableT
INTO TABLE iTableB
FOR ALL ENTRIES   IN iTableA
WHERE TableT~Field1  = iTableA-Field8
AND TableT~Field2  = Value6 .
```

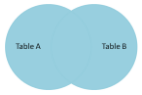
```
"Select from result of select "BP: Cute but poor
SELECT SINGLE TableT~Field1
FROM TableTa  AS T1
WHERE Field1 = Value1
AND exists ( SELECT TableT~Field5 FROM TableTb
WHERE Field3 = T1~Field3  AND "BP: Love the WHERE
Field2 = Value2  AND
Field1 = Value3 ).
```

```
"Update Transparent Table
UPDATE TableTa SET Field1 = value1 "Value to Save
WHERE Field2 = value2
AND Field3 = value3.
```

Inner join produces only the set of records that match in both Table A and Table B.



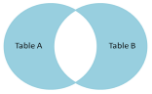
Full outer join produces the set of all records in Table A and Table B, with matching records from both sides where available. If there is no match, the missing side will contain null.



Left outer join produces a complete set of records from Table A, with the matching records (where available) in Table B. If there is no match, the right side will contain null.

To produce the set of records only in Table A, but not in Table B, we perform the same left outer join, then exclude the records we don't want from the right side via a where clause.

To produce the set of records unique to Table A and Table B, we perform the same full outer join, then exclude the records we don't want from both sides via a where clause.



[A Visual explannation of SQL JOINS Reference]

Subroutines

```
"internal subroutines
PERFORM MyPerf_name   TABLES iTable1 iTable2
                      USING Value1 Value2
                      CHANGING Value3.
```

```
"external subroutines
PERFORM  MyPerf_name('MYPROGRAM') TABLES iTable1 iTable2
                                  USING Value1 Value2
                                  CHANGING Value3
                                  IF FOUND.
```

```
"implementation
FORM MyPerf_name TABLES iTable1 iTable2
                 USING VALUE Value1 Value2
                 CHANGING Value3.

   "statements here...
ENDFORM.
```

FIELD-SYMBOLS

```
FIELD-SYMBOLS <FS> [<type>|STRUCTURE <s> DEFAULT <wa>].
```

```
" Create field symbol
FIELD-SYMBOLS:  <fs_object> LIKE LINE OF iTableA.

"Desasignar valores actuales del objeto fs
UNASSIGN <fs_object>.

"Add line on iTable from fs
APPEND INITIAL LINE TO iTableA  ASSIGNING <fs_object>.

"Assing value to fs field
<fs_object>-field1 = lvalue .

"Read Table line to field symbol object line
READ TABLE iTableB WITH KEY fieldZ = <fs_objectA>-fieldZ
                   ASSIGNING <fs_objectLineB>.
```

[More about FieldSymbols]

Programs to remember

program	description
Reptran	Backup a Program on TXT
RPINCL10	Seek string on Programs
EWK1	Its a Transaction TX: EWK1 Seek on programs
RSTXSCRP	Upload/download objets SAPscript
SLIN	BP Its a Transaction - Extended syntax check

BP: Best-Practices