ABAP/4 functions.

Transaction SE37 - ABAP/4 Function Library.
All the function modules below are proved by myself.

Workstation:

Function group name	Function group description	Function module name	Function module description	Description
GRAP	Calling up graphics / File transfer	**DOWNLOAD**	Store Internal Table with Dialog as File on the Presentation Server	Calls WS_QUERY, WS_FILENAME_GET, WS_DOWNLOAD. Dialog like **se38** "Download..."
		WS_DOWNLOAD	Save Internal Table as File on the Presentation Server	
		UPLOAD	Load Data from the Presentation Server into an Internal Table / Dialog	Calls WS_QUERY, WS_FILENAME_GET, WS_UPLOAD. Dialog like **se38** "Upload..."
		WS_UPLOAD	Load File from the Presentation Server into Internal Table	
		WS_FILENAME_GET	Call file Selector	Browse and get file name of the chosen type
		WS_EXECUTE	Execute an External Program on the Presentation Server	
		WS_MSG	Issue the Presentation Server style message	Note. Only I, E and A types. Does not affect events flow logic.
		WS_EXCEL		Download internal table into file and start Excel for it [and upload changed file into the internal table]. See also FTBU_START_EXCEL
		WS_QUERY	Execute query function on the Presentation Server	Query can be **WS**(windowsystem)-e.g.WN32 for WinNT, **CD**(directory), **EN**(environment)-?, **FL**(filelength), **FE**(filename exist)-returns 1(Yes)/0, **FC**(filename creator(Apple)), **FT**(filetype/status(UNIX only)), **DE**(directory exist)-returns 1(Yes)/0, **OS**(operating system)-e.g.NT for WinNT, **WI**(windowID)-?, **XP**(execute-path), **GM**(GMUX-version),**RC**(last returncode).

Functional Group **GRAP** is obsolete in SAP 4.6. Use instead the **SFES** Function Group (Function Modules like **GUI_***).				
IBIP	IBIP: Batch Input/Call transaction: PM	**F4_FILENAME**	F4 for filename / Filemanager support to locate file in a directory	Uses current dynpro value, calls WS_FILENAME_GET
KDUT	Diverse utilities	**KD_GET_FILENAME_ON_F4**	Call file Selector	Browse and get file name of the chosen type. Allows also **File Mask** (e.g. C:\Work\abc*.*)
SFES	Frontend services	**GUI_GET_DESKTOP_INFO**	Delivers Information About the Desktop (client)	TYPE= -1(SAP dir) -2(SAP sys dir) 1(computer name) 2(Win dir) 3(Sys dir) 4(tmp dir) 5(user name) 6(Win OS) 7(Win build No) 8(Win ver) 9(SAP GUI program name) 10(SAP GUI program path) 11(SAP current dir) 12(desktop dir)
		GUI_GET_FILE_INFO		Version, language, single-character flag, size
		GUI_GET_FILELIST_INFO		Like previous, but for list of files
		GUI_HAS_OBJECTS	SAPGUI object models	OBJECT_MODEL= ACTX | JBEAN | OLE | SAP | HTML Returns 'X' if model supported.
		GUI_HAS_ACTIVEX		Calls GUI_HAS_OBJECTS with OBJECT_MODEL= ACTX
		GUI_HAS_JAVABEANS		Calls GUI_HAS_OBJECTS with OBJECT_MODEL= JBEAN
		GUI_IS_ITS		Calls GUI_HAS_OBJECTS with OBJECT_MODEL= HTML (ITS - Internet Transaction Server)
		GUI_GET_OBJECT_MODELS		Returns table within all models supported
		GUI_DELETE_FILE		
		GUI_CREATE_DIRECTORY		
		GUI_REMOVE_DIRECTORY		
		GUI_RUN	Start a File or Program Asynchronously with ShellExecute	Imports: command, parameter, working dir.

		GUI_EXEC	Start a File or Program Asynchronously with WinExec	Imports: command, parameter.
		GUI_UPLOAD		FILETYPE= BIN | DAT | ASC
		GUI_DOWNLOAD		FILETYPE= BIN | ASC
		GUI_IS_AVAILABLE		'X' if yes
BDS_TOOLS	Business Document Service Tools	**BDS_GET_GUI_TYPE**		Returns WIN_GUI | WEB_GUI | JAVA_GUI
SI72	KEN: Info Object editor link	**TMP_GUI_CREATE_DIRECTORY**	Self-explanatory names	No-flush and flush allowed
		TMP_GUI_REMOVE_DIRECTORY		
		TMP_GUI_DELETE_FILE		
		TMP_GUI_GET_FILE_EXIST		File or directory. For file returns also filelength.
		TMP_GUI_READ_DIRECTORY		Imports: directory name, extension (space for all). Returns list of file names.
		TMP_GUI_BROWSE_FOR_FOLDER		Imports: window title, initial folder. Returns folder name.
		TMP_GUI_FILE_OPEN_DIALOG		Multiselection allowed
		TMP_GUI_DIRECTORY_LIST_FILES		Imports: directory (ended with '\'), filter (default '*.*'). Returns: list of files, list of subdirectories.
		TMP_GUI_GET_SYSTEMDIR		
		TMP_GUI_GET_COMPUTERNAME		
		TMP_GUI_GET_TEMPPATH		
C0PC	PC interface (file name checks)	**PC_CHECK_DRIVE**	Self-explanatory names	Checks for correctness only, not existence (can be checked for DOS also)
		PC_CHECK_PATH		
		PC_CHECK_FILENAME		
		PC_CHECK_EXTENSION		
		PC_CHECK_PATH_WITH_DRIVE		
		PC_CHECK_FILENAME_WITH_EXT		

		PC_BUILD_COMPLETE_FILENAME		
		PC_SPLIT_COMPLETE_FILENAME		

Application Server:

Function group name	Function group description	Function module name	Function module description	Description
SAUS	Repository Switch Tools	**SUBST_GET_FILE_LIST**	Return table with file list for the given directory (pattern allowed)	
DX_FILE	File Management	**F4_DXFILENAME_TOPRECURSION**	Popup to select one file from the given application server directory (pattern allowed). (Can be used also for selecting file on presentation server: calls WS_FILENAME_GET) Parameters: I_LOCATION_FLAG={A|P|space}; if I_LOCATION_FLAG is blank then popup to choose Appl./Present.; if I_SERVER='?' then popup to select appl.server.	
THFB	Task handler functions	**TH_SYSTEMWIDE_WPINFO**	Returns table with Work Processes Inf.	
		TH_SYSTEMWIDE_USER_LIST	Returns table with Users Inf.	
		TH_REMOTE_USER_LIST	Returns table with Users on the given Application server.	
		TH_USER_LIST		
		TH_SERVER_LIST	Returns list of application servers.	
		TH_SELECT_SERVER	Popup to select one of available application servers.	

Date and Time:

Function group name	Function group description	Function module name	Function module description	Description

SCAL	Calendar functions	**DATE_CONVERT_TO_FACTORYDATE**	Returns factory calendar date for a date	Calculates and returns factory calendar date for a date (if CorrectOption = '+'); Checks if the date is work day (if CorrectOption = '-').
		DATE_GET_WEEK	Returns week in which a date lies	Import : YYYYMMDD; Export : YYYYNN, where NN is # of week.
		WEEK_GET_FIRST_DAY	Calendar function: Return first day for a week	Import week in format YYYYNN where NN is # of week.
		DATE_COMPUTE_DAY	Returns day of week for a date	Import : YYYYMMDD; Export : 1-Monday, ...
CAPP	Time Sheet: Approval	**DAY_IN_WEEK**		
CADA	Calculate Date	**GET_WEEK_INFO_BASED_ON_DATE**	Import : YYYYMMDD; Export : YYYYNN - week # for the date, Monday and Sunday of the week.	
		LAST_WEEK	Import : YYYYNN (week #); Export : YYYYNN - previous week #, Monday and Sunday of the previous week.	
		NEXT_WEEK	Import : YYYYNN (week #); Export : YYYYNN - next week #, Monday and Sunday of the next week.	
		DATE_TO_DAY	Import : YYYYMMDD; Export : weekday (as word).	
ZDAT	Date format conversion	**DATE_GET_FIRST_DAY_OF_WEEK**	Returns week within which a date lies	Import : YYYYMMDD; Export : YYYYMMDD - Monday in the week.
		CONVERT_DATE_TO_ALPHA_NUMERIC	Converts internal date to DD Mon YY	Import : YYYYMMDD, Language; Export : DD Mon YY.
SCA1	Date: Conversion	**CONVERT_DATE_TO_EXTERNAL** **CONVERT_DATE_TO_INTERNAL**	Converts internal date to user-specific format string and vice	Import/Export : YYYYMMDD; Export/Import : user specific date string. Can be for import either without or within delimiters, but for export only without delimiters. See field **DATFM** in the **USR01** table. Can be set up via *System -> User Profile -> User Defaults* or using **su50**.

<table>
<tr><td></td><td></td><td></td><td>versa.</td><td></td></tr>
<tr><td rowspan="2">VDAT</td><td rowspan="2">Date conversion</td><td>PERIOD_AND_DATE_CONVERT_INPUT</td><td>Input conversion for a date with period.</td><td>Import : country specific date|period string (External Date), External or Internal Periods;
Export : YYYYMMDD (Internal Date - concrete date or first day of period), Internal Period;
<table>
<tr><td>Int.Period</td><td>Ext.Period</td><td>Int.Date|Period, e.g.</td></tr>
<tr><td>1</td><td>D</td><td>MM/DD/YYYY</td></tr>
<tr><td>2</td><td>W</td><td>WW/YYYY</td></tr>
<tr><td>3</td><td>M</td><td>MM/YYYY</td></tr>
<tr><td>4</td><td>P</td><td>?</td></tr>
<tr><td>5</td><td>K</td><td>?</td></tr>
</table></td></tr>
<tr><td>PERIOD_AND_DATE_CONVERT_OUTPUT</td><td>Conversion of a date with period specification with output.</td><td>Import : YYYYMMDD (Internal Date), Language, Country, Internal Period;
Export : country specific date|period string (External Date), External Period, External Period Text.
<table>
<tr><td>Int.Period</td><td>Ext.Period</td><td>Ext.Date, e.g.</td><td>Ext.Per.Text</td></tr>
<tr><td>1</td><td>D</td><td>MM/DD/YYYY</td><td></td></tr>
<tr><td>2</td><td>W</td><td>WW/YYYY</td><td>Week</td></tr>
<tr><td>3</td><td>M</td><td>MM/YYYY</td><td>Month</td></tr>
<tr><td>4</td><td>P</td><td>? space</td><td></td></tr>
<tr><td>5</td><td>K</td><td>? space</td><td>Calndr</td></tr>
</table></td></tr>
<tr><td>BCAT</td><td>Test tool - Function modules</td><td>DATE_IN_FUTURE</td><td colspan="2">Import: date in current external format for user; number of days.
Export: calculated date in future in external format mm/dd/yyyy</td></tr>
<tr><td rowspan="2">SVAV</td><td rowspan="2">Variable date fields in variants Note. Local time for</td><td>RS_VARI_V_1_ACTUAL_MONTH
RS_VARI_V_1_LAST_MONTH
RS_VARI_V_1_NEXT_MONTH</td><td colspan="2">Returns first day of the actual/last/next month.</td></tr>
<tr><td>RS_VARI_V_ACTUAL_MONTH
RS_VARI_V_LAST_MONTH</td><td colspan="2">Returns selection range for the actual/last month.
Output: 1st line in table P_DATETAB</td></tr>
</table>

	user	**RS_VARI_V_DAYS_UP_TO_NOW**	Returns selection range from today-a_days to today+b_days. Input: 1st line in P_INTRANGE (a=P_INTRANGE-LOW, b=P_INTRANGE-HIGH); Output: 1st line in table P_DATETAB	
		RS_VARI_V_WDAYS_UP_TO_NOW	Returns selection range from today-a_work_days to today+b_work_days. Input: a=P_INTRANGE-LOW, b=P_INTRANGE-HIGH (if P_INTRANGE-SIGN <> 'C'); Fact.Calendar Id = P_INTRANGE-OPTION (if P_INTRANGE-SIGN = 'C') Output: 1st line in table P_DATETAB	
		RS_VARI_V_MONTH_XXX_YYY	Returns selection range for the actual month [-a_months +b_months]. Input: 1st line in P_INTRANGE (a=P_INTRANGE-LOW, b=P_INTRANGE-HIGH); Output: 1st line in table P_DATETAB	
		RS_VARI_V_L_LAST_MONTH	Returns last day of last month.	
		RS_VARI_V_MONTH_UP_TO_NOW	Returns selection range for the actual month up to now. Output: 1st line in table P_DATETAB	
		RS_VARI_V_QUARTER1XXXX **RS_VARI_V_QUARTER2XXXX** **RS_VARI_V_QUARTER3XXXX** **RS_VARI_V_QUARTER4XXXX**	Returns selection range for the Quarter of the given year. Input: 1st line in P_INTRANGE (Year=P_INTRANGE-LOW); Output: 1st line in table P_DATETAB	
		RS_VARI_V_TODAY	Today	
		RS_VARI_V_TODAY_X	Returns Today + P_INTRANGE-LOW days	
		RS_VARI_V_TODAY_XWD	Returns Today + P_INTRANGE-LOW work_days Input: work_days = P_INTRANGE-LOW (if P_INTRANGE-SIGN <> 'C'); Fact.Calendar Id = P_INTRANGE-OPTION (if P_INTRANGE-SIGN = 'C')	
		RS_VARI_V_XWD_ACTUAL_MONTH	Returns 1st day of the actual month + P_INTRANGE-LOW work_days Input: work_days = P_INTRANGE-LOW (if P_INTRANGE-SIGN <> 'C'); Fact.Calendar Id = P_INTRANGE-OPTION (if P_INTRANGE-SIGN = 'C')	
DATD	Date format	**DATUMSAUFBEREITUNG**	Country-specific date	Examples of output: MM/YY, MM/YYYY, DD.MM, M/D/YYYY, DD/MM/YYYY, WW/YY, WW/YYYY.

			formatting for the current user.	
CCMC	Archiving function group	**CCM_GO_BACK_MONTHS**	Module to go back specified number of months	Funny module: 03/31/2002 - 1 month = 02/31/2002 
RPP3	HR-PT: Get levels, rates, etc	**HR_PT_ADD_MONTH_TO_DATE**	HR-PT: Add or subtract months to a date	DMM_OPER can be '+', '-' or '='. if '=', then • DMM_POS = 'BEG' - return 1st day of the DMM_COUNT month of the same year. Use MM format (e.g. 02) otherwise you can get date like '2 /12/2002' for February ☹. • DMM_POS = 'END' - returns the (DMM_COUNT month + 1), (day - 1) for the input date DMM_DATIN. Bug: try DMM_DATIN = MM/31/YYYY, DMM_COUNT = 1 ☹.
FVOZ	RE instalment payments	**RE_ADD_MONTH_TO_DATE**	This module really can add/subtract months to/from date.	
RPDD	HR-D: Payroll Germany	**RP_LAST_DAY_OF_MONTHS**	HR-D: Determine last day of month	

Currency:

Function group name	Function group description	Function module name	Function module description	Description

<table>
<tr><td>FWOS</td><td>Currency translation for flows</td><td>FWOS_CURRENCY_DECIMALS_READ</td><td></td><td rowspan="3">All the currency amounts are stored in SAP tables as CURR(n,2) (the same as DEC(n,2)). So before any arithmetic operation value should be adjusted using the real decimals number for the given currency (stored in TCURX).
Conversion Rates by type and date are stored in TCURR (+factors). Standard type is M. Date is stored in inverted format (the most recent date has the numerically smallest value). ABAP code to convert dates:
convert date p_date into inverted-date w_date.
convert inverted-date w_date into date p_date.
the only difference between CONVERT_TO_LOCAL_CURRENCY and CONVERT_TO_FOREIGN_CURRENCY seems to be the following:
Foreign currency is TCURR-FCURR (From Currency)
Local Currency is TCURR-TCURR (To Currency)
So result will be slightly different for the both functions (two rates stored in the TCURR: e.g. JPY->USD rate is 0.00880, USD->JPY rate is 122.00000). Better to use CONVERT_TO_LOCAL_CURRENCY, because multiplication is more exact operation than division.
Both conversion functions can return also selected rate and factors
Example $$$$$$$$$$</td></tr>
<tr><td rowspan="2">SCUR</td><td rowspan="2">Translation of Currency Amounts etc.</td><td>CONVERT_TO_LOCAL_CURRENCY</td><td></td></tr>
<tr><td>CONVERT_TO_FOREIGN_CURRENCY</td><td></td></tr>
<tr><td colspan="5">Both CONVERT_TO_LOCAL_CURRENCY and CONVERT_TO_FOREIGN_CURRENCY use complicated logic considering contents of the TCURX (Decimal Places in Currencies), TCURR (Exchange Rates), TCURF (Conversion Factors) and other tables, € conversion rules etc. Often functions return error message if information in the tables is inconsistent or not maintained.
To simple currency conversion can be used direct calculation based on the tables TCURR and TCURX.</td></tr>
</table>

See also BAPIs:

- **Currency** (since 4.0A) - to get currencies and decimals
- **ExchangeRate** (since 4.5A) - to get exchange rates and factors for a date

F017	Conversion of amounts to words utility	**SPELL_AMOUNT**	Convert numbers and figures in words	1. convert an amount into words 2. convert a number into words (CURRENCY=space) 3. set decimal point in an amount and return number of decimals for the currency (LANGUAGE=space)

Address:

Function group name	Function group description	Function module name	Function module description	Description
SZA0	Central address management (w/o dialog)	**ADDR_GET**		Get address
SADR	Address formatting	**ADDRESS_INTO_PRINTFORM**	Address format according to Post Office guidelines	see FM documentation. Note. It's enough to import two parameters: • ADDRESS_TYPE = 1 (Firm or Organization, SAP Address) • ADDRESS_NUMBER
		ADDRESS_SHOW_PRINTFORM		

see tables:

- ADRC - Addresses (central address admin.)

Units of measure (UOM):

Function group name	Function group description	Function module name	Function module description	Description
SCV0	Measurement units: Conversion	**CONVERSION_FACTOR_GET**	Measurement unit conversion: Get measurement unit conversion factor	Not for Dimensionless Units of Measure
		UNIT_CONVERSION_SIMPLE	Measurement unit conversion by table T006, with rounding	
		UNIT_OF_MEASURE_SAP_TO_ISO		
		UNIT_OF_MEASURE_ISO_TO_SAP		
MAME	Mat. Master: Determine Units of Measure	**MATERIAL_UNIT_CONVERSION**	Material quantity conversion from Base Unit of Measure to Alternative Unit of Measure and vice versa.	For Dimensionless Units of Measure (Each, Piece, Box etc.) conversion depends on the given Material (see table MARM). For other Units of Measure (Length, Weigth etc.) conversion can be calculated from the T006 table or via CONVERSION_FACTOR_GET.
SCVU	Measurement units: Conversion, F4 help	**CONVERSION_EXIT_CUNIT_INPUT**	Conversion exit for commercial (3-char) measurement unit INPUT	
		CONVERSION_EXIT_CUNIT_OUTPUT	Conversion exit for commercial (3-char) measurement unit OUTPUT	
		CONVERSION_EXIT_LUNIT_INPUT	Conversion exit for technical (6-char) measurement unit INPUT	
		CONVERSION_EXIT_LUNIT_OUTPUT	Conversion exit for technical (6-char) measurement unit OUTPUT	
		UNIT_OF_MEASUREMENT_HELP	Input help for measurement units of a predefined dimension (F4 help)	Allows to browse/select an internal/external UOM from commercial(3-char)/technical(6-char) UOMs, possible with predefined dimension (see FM documentation).

BDC:

Function group name	Function group description	Function module name	Function module description	Description

SBDC	Batch input	**BDC_OPEN_GROUP**		Open batch input session
		BDC_CLOSE_GROUP		Close batch input session
		BDC_INSERT		Add data to batch input session

Popup Windows:

Function group name	Function group description	Function module name	Function module description	Description
KYSY	Global functions	**POPUP_WITH_TABLE_DISPLAY**	Popup to display internal table data	Displays an internal table and returns index of chosen line. Good also for F4 (on value-request) function.
SHL3	Help functions for external use	**HELP_VALUES_GET**	Popup to display default F4 help values for a table field	Does the same as default F4 (on value-request) function. Can be used, for example, if selected value should be used for immediate update another screen fields. **Example**.
		HELP_VALUES_GET_NO_DD_NAME	Standard popup to display F4 help values for a table field as internal table with additional columns (all fields should be active Data Dictionary table fields). Allow selection for each column. (obsolete since 4.0)	**Example**
		TRANSFER_NAMES_TO_FIELDS	Prepare formatted internal table to use as parameter of HELP_VALUES_GET_NO_DD_NAME.	
SDHI	F4 interface module for external	**F4IF_INT_TABLE_VALUE_REQUEST**	F4 help also returning the value to be displayed in internal table	Should be used as standard since 4.0. **Example**

	appl.	**F4IF_FIELD_VALUE_REQUEST**	F4 help for fields that are only known at runtime	Standard F4 help for a Data Dictionary help
SPO1	Dialog box for save prompts	**POPUP_TO_CONFIRM_STEP**	Popup a question (two lines of text) with buttons Yes, No [,Cancel]	Returns 'J' for Yes, 'N' for No, 'A' for Cancel.
		POPUP_TO_CONFIRM_WITH_MESSAGE	Popup a diagnostic message (two lines of text) and a question (two lines of text) with buttons Yes, No [,Cancel]	Returns 'J' for Yes, 'N' for No, 'A' for Cancel.
		POPUP_TO_CONFIRM	Popup a question with two customized buttons (e.g. Yes, No) and [Cancel]	Returns '1' and '2' for 1st and 2nd buttons, 'A' for Cancel.
		POPUP_TO_CONFIRM_LOSS_OF_DATA	Popup a 'Data will be lost' and question (two lines of text) with buttons Yes and No	Returns 'J' or 'N'.
SHI5	General Functions	**POPUP_TO_CONFIRM_DATA_LOSS**	Calls POPUP_TO_CONFIRM_STEP with 'Changed data will be lost. Save?'	
SEUO	Repository Info System - background	**POPUP_TO_DECIDE_INFO**	Popup a question (two lines of text).	Returns Answer='J'(Yes) and 'A'(Cancel).
SPO2	Dialog box for deciding ages	**POPUP_TO_DECIDE**	Popup a question (three lines of text) with two customized buttons and [Cancel]	Returns '1' and '2' for 1st and 2nd buttons, 'A' for Cancel.
		POPUP_TO_DECIDE_WITH_MESSAGE	Popup a message (three lines of text) and a question (three lines of text) with two customized buttons and [Cancel]	Returns '1' and '2' for 1st and 2nd buttons, 'A' for Cancel.
SPO3	Dialog box for entering data	**POPUP_TO_GET_VALUE**	Popup to request for a field of the given dictionary table.	Returns Answer=' ' if the value changed, 'C' if not.
		POPUP_TO_GET_ONE_VALUE	Popup to request for a string value.	Returns Answer='J' if the value entered, 'A' if not.
SPO4	Dialog box for display and request	**POPUP_GET_VALUES**	Dialog box for the display and request of values, without check	One or more DB table/view fields (Medium field label for data element used
		POPUP_GET_VALUES_DB_CHECKED	Dialog box for requesting values, check against the DB table/view	

		POPUP_GET_VALUES_USER_CHECKED	Dialog box for requesting values, check by user exit (import parameters FORMNAME and PROGRAMNAME)	as prompt text). Returncode=' ' if the value entered, 'A' if cancel. **Example**
		POPUP_GET_VALUES_USER_HELP	Dialog box for requesting values, call of user exits (import parameters FORMNAME and PROGRAMNAME) and help (import parameters F1_FORMNAME and F1_PROGRAMNAME, F4_FORMNAME and F4_PROGRAMNAME)	
		POPUP_GET_VALUES_USER_BUTTONS	Dialog box for requesting values and offering user pushbuttons	
GLU2	FI-LC: Currency translation customizing	**POPUP_TO_MODIFY_TEXT**	Text field has length 45.	Returns Answer='J' if modified, 'A' if not (and empty field).
		POPUP_CONTINUE_YES_NO		Returns 'J' or 'N'.
		LC_POPUP_RADIO_5		Like this: ◉ Value 1 ○ Value 2 ○ Value 3 ○ Value 4 ○ Value 5 Returns No of line or 'A' if cancelled.
SPO5	Dialog box for selecting from lists	**POPUP_TO_DECIDE_LIST**	Displays internal table as radiobuttons.	Like this: ◉ Value 1 ○ Value 2 ○ Value 3 ○ Value 4 Returns No of line or 'A' if cancelled.
RHRD	Shift Planning: Rqmnts	**POPUP_TO_DISPLAY_TEXT**	Displays two lines of text like (I) message.	

	Definition Maint.			
STMO	Monitoring statistics	**POPUP_TO_INFORM**	Displays four long lines of text.	
SLST	List processing - additional functions	**POPUP_DISPLAY_MESSAGE**	Displays 'Action not possible' string and formatted message (input: Message Id, Number and parameters).	
ALDB	Logical databases in ABAP/4	**COMPLEX_SELECTIONS_DIALOG**	External Call 'Multiple Selection'	Displays standard 'Multiple Selection' window (like after pressing button on selection screen) for a range or select-option (table parameter RANGE).

See also:

- F4 popup screen for Units of Measure (UOM).

Lists (Reports):

Function group name	Function group description	Function module name	Function module description	Description
ALDB	Logical databases in ABAP/4	**PRINT_SELECTIONS**		Builds nice ready to print table within Report Id, variant, date/time and selected parameters / selection options. Receives Report Id (sy-cprog), variant (sy-slset). Note. Parameter MODE is not used. Returns internal table (parameter INFOTABLE). **Example**.
SVAP	Variant maintenance	**RS_COVERPAGE_SELECTIONS**	Internal use: Generate table for	Called from PRINT_SELECTIONS

			printing cover sheet	
FMRE	FIFM: Reporting	**FM_SELECTION_CRITERIA_PRINT**		Just prints selection criteria for the report (sy-cprog). **Example**.
SPRI	Read, determine, change spool print parameters and archive parameters	**GET_PRINT_PARAMETERS**	Read, determine, change spool print parameters and archive parameters	**Only** correct way to modify a print or archive parameter record (see FM documentation).

SLST	List processing - additional functions	**LIST_FROM_MEMORY**	Prepared list import from memory	`report report01.`
				`...`
				`data listtab like listobject occurs 1.`
				`...`
				`* Either other report`
				`* has exported the list to memory, or`
				`submit report02...`
				`  exporting list to memory and return.`
				`call function 'LIST_FROM_MEMORY'`
				`  tables listobject = listtab.`
				`* process listtab (see below)`

		LIST_TO_MEMORY	Exports list of SY-LSIND to memory	
		WRITE_LIST	Display a (saved) list object	Writes list (e.g. imported from memory), not encapsulated in a CALL SCREEN. Can be used after or before WRITE statement to add list data to the current screen
		DISPLAY_LIST	List object display (ABAPLIST)	Displays list (e.g. imported from memory) in full screen or dialog box.
		SAVE_LIST	Save list	Saves list of SY-LSIND as internal table
		LIST_TO_ASCI	Saves list of SY-LSIND as text internal table (without line breaks, icons, colors etc)	
		LIST_TO_TXT		
SYSF	Internal system functions (lists)	**LIST_DOWNLOAD**	ABAP list download	Like *List->Save->Local file*. METHOD = RTF | DAT | HTML | NOCO(no conversion) | space(selection dialog)
		DOWNLOAD_LIST	Download list in ASCI format (called from LIST_DOWNLOAD when METHOD = NOCO)	

		LIST_DOWNLOAD_HTML	Download list in HTML format (called from LIST_DOWNLOAD when METHOD = HTML)	

		LIST_SCROLL_LINE_TOPMOST	Scroll List Display: Selected Line Becomes First Line	E.g. to place an interactively-selected line (SY-LILLI) directly after the page header:
				`call function 'LIST_SCROLL_LINE_TOPMOST'`
				`exporting list_line = sy-lilli.`
				See FM documentation.
				See also SCROLL statement.

		LIST_SCAN	Calls standard Find window.	

Programs:

Function group name	Function group description	Function module name	Function module description	Description
SVAR	Variant maintenance	**RS_REFRESH_FROM_SELECTOPTIONS**	Current contents of selection screen	Allows submit another report with parameters entered for the current report. **Example**.
		RS_VARIANT_CONTENTS	Input: report/variant; Output: variant contents	
		RS_COPY_SELECTION_SETS	Copy all variants from one report to another report	
		RS_REPORTSELECTIONS_INFO	Returns list of selection screen parameters and select-options, including default values	
		RS_VARIANT_INFO	Returns list of variants for report(s), additional selection criteria allowed	
		RS_VARIANT_TEXT	Returns variant short text	
		RS_SUPPORT_SELECTIONS	Assigns the variant values to a **loaded** report	
		RS_VARIANT_EXISTS	R_C = 0 if variant exists	
		RS_VARIANT_CATALOG	Standard pop-up window to select a variant for the report	
		RS_VARIANT_LIST	Maintain report variants: STATUS=LIST to Display|Delete|Print STATUS=LIS1 to Choose	

SLDB	FMs for logical databases	**RS_SUBMIT_INFO**	Gives information about the mode of the current SUBMIT	is report sel.screen processing under run|print|variant maint|submit via job etc.?

SAPscript:

Function group name	Function group description	Function module name	Function module description	Description
STXC	SAPscript composer ⓘ Example	**OPEN_FORM**	SAPscript: Open form printing	• to only preview the form: options-tdpreview='X' (options-tdnoprint forbids print even from preview) • to get OTF data instead of printing: options-tdgetotf='X' • to output OTF data to memory buffer instead of printing: device='OTF_MEM'
		CLOSE_FORM	SAPscript: End layout set printing	• spool #: result-tdspoolid • OTF data: otfdata (when options-tdgetotf='X' in OPEN_FORM)
		WRITE_FORM	SAPscript: Output text element in form window	
		CONTROL_FORM	SAPscript: Control form output	issue command to the form, e.g. `exporting command = 'NEW-PAGE'`
		CONVERT_OTF_MEMORY	SAPscript: Convert OTF from memory into text format	output: table of structure TLINE (OTF saved to memory buffer by OPEN_FORM)
STXW	OTF screen output	**SAVE_OTF_TO_MEMORY**	Write OTF format to memory	export|import OTF internal table to|from memory id ...
		READ_OTF_FROM_MEMORY	Read OTF from memory	

		PRINT_OTF		print OTF from internal table
		DISPLAY_OTF	Display an OTF table on the screen	
		DISPLAY_POSTSCRIPT	Display a postscript table on the screen	`tables postscript structure itcps`
		CONVERT_OTF	Convert OTF format to various formats (TLINE table)	ASCII or PDF
		CONVERT_OTF_2_PDF	Convert OTF to PDF (TLINE table). OTF can be filled used archivelink. Calls CONVERT_OTF.	
		CONVERT_OTF_2_PDF_ARCHIVELINK	Convert OTF to PDF (TLINE table). Calls CONVERT_OTF. Looks like the function names for these two functions are mixed up 	
		CONVERT_OTF_AND_FAX		
		CONVERT_OTF_AND_MAIL		
		CONVERT_OTFSPOOLJOB_2_PDF	Input: spool # (SAPscript: tsp01-rqdoctype='OTF'); Output: PDF as internal table (TLINE) **Example**	
		CONVERT_ABAPSPOOLJOB_2_PDF	Input: spool # (ABAP listing: tsp01-rqdoctype='LIST'); Output: PDF as internal table (TLINE) **Example**	

Messages:

Function group name	Function group description	Function module name	Function module description	Description
SLG9	Application log: Auxiliary functions	**MESSAGE_PREPARE**	Read T100 message and format message with parameters	Import: Language; Message Id, # and variables; Export: formatted message text.
V12B	RV Condition Generation - General	**MESSAGE_TEXT_BUILD**	Set up a message with parameter	Import: Message Id, # and variables; Export: formatted message text. Since 4.x can be replaced by: `MESSAGE ... INTO f.`

BATG	BAPIs for Reading Documentation	**BAPI_MESSAGE_GETDETAIL**	Read long text of error message	New in 4.5a. Import: Message Id, #, variables and formats; Export: formatted message text and long text.

Data Dictionary and Data Processing:

Function group name	**Function group description**	**Function module name**	**Function module description**	**Description**
MGF2	Accesses for Generator & For. Key Checks	**CHECK_DOMAIN_VALUES**	Checks if a fixed value is valid for domain	Exceptions: WRONG_VALUE, DOM_NOT_FOUND
		GET_DOMAIN_VALUES	Read out fixed values of a domain	With text descriptions: **Example1** **Example2**
SDEX	ABAP/4 Dict.: official ext. interfaces	**DD_DOMVALUES_GET**	External interface for reading the domain fixed values	Called from GET_DOMAIN_VALUES. Allows use of any Language for text descriptions.
SDIF	Interfaces for the ABAP/4 Dictionary	**DDIF_FIELDINFO_GET**	DD: Interface for retrieving information about DD field	Length, Texts etc. **Example** - print current values of the system fields (SY-...)
		DDIF_TABL_GET	DD: Interface to Read a Table from the ABAP Dictionary	Length, Texts etc.
		DDIF_NAMETAB_GET	DD: Interface to Read a Runtime Object from the ABAP Dictionary	Length etc., but not Texts (called from GET_FIELDTAB).
SDD3	Dictionary interfaces	**GET_FIELDTAB**	Read fields of a table in the format DFIES	Obsolete. Length, Texts etc. **Example**
SVIX	Editing tabular data	**VIEW_AUTHORITY_CHECK**	Check authority for table/view.	Check Authorization Objects S_TABU_DIS (for whole table class according to the table TDDAT) and S_TABU_CLI (if client-independent table).

		VIEW_ENQUEUE	Lock/Unlock data in table/view.	Whole table or selection
/1BCDWBEN/SENQ		**ENQUEUE_E_TABLEE** **DEQUEUE_E_TABLEE** **ENQUEUE_E_TABLES** **DEQUEUE_E_TABLES**	Lock/Unlock data in table.	Whole table or selection (called from VIEW_ENQUEUE) **E**xclusive | **S**hared
/1BCDWBEN/SEN4		**ENQUEUE_E_TABLE** **DEQUEUE_E_TABLE**		Whole table or selection
LOIK	LOI: Data Conversion	**CLOI_PUT_SIGN_IN_FRONT**	Put sign in front.	And condense result

Screens:

Function group name	Function group description	Function module name	Function module description	Description
SHL2	Help functions	**DYNP_VALUES_READ**	Reads a screen field	**Example**.
		DYNP_VALUES_UPDATE	Updates a screen field	
SLDB	FMs for logical databases	**SELECT_OPTIONS_RESTRICT**	Make use of SELECT-OPTIONS easier on the selection screen	Powerful function module to control SELECT-OPTIONS (see FM documentation). E.g. fully forbids intervals in SELECT-OPTIONS: **Example**.
ICON	General functions for graphical elements	**ICON_CREATE**	Transfer Icon Name and Text to a Screen Field	Combines into one character field: icon, text, additional text that will appear on mouse over event. This field can be used for screen field or for list output field as well. **Example**: 
SVSM	Value Request Manager	**VRM_SET_VALUES**	Transfer values to listbox parameter	Populates listbox parameter (like ) with values for selection. **Example**
SLDB	FMs for logical databases	**RS_SET_SELSCREEN_STATUS**	Sets own GUI status on selection screens	Can be called while INITIALIZATION to set needed PF-STATUS before selection screen output.

SAPGUI, RFC:

Function group name	Function group description	Function module name	Function module description	Description
SGUI	SAPgui interface	**SAPGUI_PROGRESS_INDICATOR**	Set 'Progress Indicator' in Current Window	SAPGUI ver. 4.0B: SAPGUI ver. 4.6D:
SRFC	RFC administration	**RFC_SYSTEM_INFO**	Returns System Information.	If destination = 'SAPGUI" then returns: - SAP character set; - SAPGUI version; - Frontend computer Name, Operating System and IP-address etc. **Example**.
THFB	Task handler functions	**TH_USER_INFO**	Returns User Information.	IP address, computer name, sessions. **Example1** (3.1), **Example2** (4.6).
		TH_LONG_USR_INFO	Returns User Sessions Information.	
		TH_POPUP	Send message to a SAP user.	The message will appear in SAPGUI popup window. The user can be in other mandant (parameter CLIENT) or SAP system (`call ... destination DEST ...`)
		TH_REM_TRANSACTION	Call transaction [using bdctab]. Remotely if `call ... destination DEST ...`	
		TH_REMOTE_TRANSACTION	Login and Call transaction [using bdctab] on remoted system (parameter DEST).	
		TH_SAP_LOGIN	Login and Call transaction [using bdctab] on remoted system (`call ... destination DEST ...`). Called from TH_REMOTE_TRANSACTION	
PHOH	SAPphone: Help functions	**TERMINAL_ID_GET**	Returns IP-address and Terminal(Computer) Id for the particular SAP User.	
SDTX	Desktop Access	**RFC_READ_TABLE**	External access to R/3 tables via RFC	As of 4.6c, not Released to Customers

OLE:

Function group name	Function group description	Function module name	Function module description	Description
RPPC	HR/PC interface	**EXCEL_OLE_STANDARD_DAT**		Just calls MS_EXCEL_OLE_STANDARD_DAT
SLPC	OLE - PC interface	**MS_EXCEL_OLE_STANDARD_DAT**		Downloads internal table and opens it in MS Excel. **Example**.
KCDE	Data transfer from PC files	**KCD_EXCEL_OLE_TO_INT_CONVERT**		Uploads an *.xls file to internal table (max cell length = 32). **Example**. This function uses a range selection and copy-paste technique, therefore it quite fast.✈ There is another **Example 2** that reads data from Excel file cell by cell.🚲
ALSMEX	FI-AA Legacy Data Transfer w/ Excel	**ALSM_EXCEL_TO_INTERNAL_TABLE**		the same as KCD_EXCEL_OLE_TO_INT_CONVERT but max cell length = 50
FTBU_CONV	General conversion	**FTBU_START_EXCEL**		just [download internal table to file and] start Excel (w/o OLE). See also WS_EXCEL

Jobs:

Function group name	Function group description	Function module name	Function module description	Description
BTCH	Background processing	**GET_JOB_RUNTIME_INFO**		Job, Step, Event etc.
		JOB_OPEN	Open Job Scheduling Without Dialog (Including COMMIT WORK)	Open background job and obtain the job Id (JOBCOUNT)
		JOB_SUBMIT	Insert Background Task in Background Request with COMMIT WORK	Add step to the opened job
		JOB_CLOSE	Close Background Request With COMMIT WORK	Release the job

System and Transports:

Function group name	Function group description	Function module name	Function module description	Description
SI23	KEN: Transport Info Objects	**IW_C_CREATE_TRANSPORT_REQUEST**		Create request (Workbench, Transport etc.), assigning next request # w/o dialog
		IW_C_APPEND_OBJECTS_TO_REQUEST		Add objects to request (Workbench, Transport etc.) If all objects are unlocked, then dialog to choose request. If one of objects is locked in a request, then remaining objects added to the same request
SVRY	Version Management: External Interfaces	**SVRS_AFTER_CHANGED_ONLINE_NEW**		Generate new version. Only for the objects that: • versionable • exist in TADIR • locked or local
SEDA	Program parsing	**RS_GET_ALL_INCLUDES**		
		RS_GET_MAINPROGRAMS		
SEUF	Interfaces to Function Builder	**RS_FUNCTION_POOL_CONTENTS**	Get Function Modules in a Function Group	Import: function group name. Export: table of function modules (function module name <->ABAP program name)
		FUNC_GET_OBJECT	Get function module information (main program, include number, parameters and source code.	

SUNI	Function Library program interface	**FUNCTION_INCLUDE_SPLIT**	Get Function Modules and Includes Belonging to a Function Group	Depending on the import parameter, returns the following data:
				
				• PROGRAM - namespace, function group
				• COMPLETE_AREA - namespace, function group
				Example: find main program for function group.
				• INCLUDE -
				  ○ namespace, function group
				  ○ include (without namespace)
				  ○ include number (for function module includes)
				  ○ 3-character suffix,
				  ○ function module name (for function module includes, and if SUPPRESS_SELECT=space)

		FUNCTION_INCLUDE_INFO		Depending on the import parameter, returns the following data: • FUNCNAME - namespace, function group, include name • GROUP - namespace, table of function modules (function module<->include name) • INCLUDE - namespace, function group, function module name (for function module includes)
		FUNCTION_INCLUDE_CONCATENATE	Determine the Name of a Function Group / Main Program by Namespace	Import: include number, complete area, namespace Export: function group, names of main program, top include, UXX include, include with the given number.

See also tables:

- TLIBG - Person responsible for function class (list of all function groups)

- TFDIR - Function Module
- ENLFDIR - Additional Attributes for Function Modules
- FUPARAREF - Parameters of function modules

Application Log:

Function group name	Function group description	Function module name	Function module description	Description
SLG1	Application Log (old): Read	**APPL_LOG_READ_DB_WITH_LOGNO**	Read Application Log by Log Number	Returns Log(s) header and messages into internal tables. E.g. all the messages for an IDoc # can be read from application log if the EDIDS-APPL_LOG contains the log number.
SLG3	Application Log (old): Display	**APPL_LOG_DISPLAY_WITH_LOGNO**	Display Application Log by Log Number	Displays Log(s) like transaction **SLG1**.
Functional Groups **SLG*** exist since 3.0. New more flexible function modules exist since 4.6 (Function Groups **SBAL_***, Function Modules like **BAL_***).				
BATG	BAPIs for Reading Documentation	**BAPI_APPLICATIONLOG_GETDETAIL**	Read Details of Entries in Application Log	New in 4.5a. Import: Log Numbers / [Message Numbers]; Export: formatted messages. (calls APPL_LOG_DISPLAY_WITH_LOGNO)
SBAL	Application Log: Collect Messages	**BAL_LOG_CREATE**	Application Log: Log: Create with header data	Created log can be referred by Log Handle. (log handle can be skipped if the program itself created and is using the log). **Example**.
		BAL_LOG_MSG_ADD	Application Log: Log: Message: Add	

SBAL_DISPLAY	Application Log: Display Log	**BAL_DSP_LOG_DISPLAY**	Application Log: Fullscreen log output (like transaction SLG1)	
Logs in memory and in the database are referred to in the new Application Log by the log handle (LOG_HANDLE), but the previous LOGNUMBER, which is assigned from number range interval 01 of number range object APPL_LOG when you save, still exists. A lot of applications have a reference to this LOGNUMBER in their structures, so it is still supported. The LOGNUMBER is also more understandable for users than the LOG_HANDLE. There is a 1:1 relationship between LOG_HANDLE and LOGNUMBER.				

ALE, EDI, IDocs:

Function group name	Function group description	Function module name	Function module description	Description
EDI1	EDI: Processing of one IDoc	**I. Opening Idoc to process:**		
		EDI_DOCUMENT_OPEN_FOR_CREATE	EDI interface: Open IDoc for create	Initiates the generation of an IDoc. <-- IDENTIFIER - temporary Id for referring in following FMs (multiprocessing allowed)
		EDI_DOCUMENT_OPEN_FOR_FORCE	EDI interface: Open IDoc for create with few/no checks	If the EDI_DOCUMENT_OPEN_FOR_CREATE failed but the IDoc **must** be passed to the database (multiprocessing allowed)
		EDI_DOCUMENT_OPEN_FOR_TRACE	EDI interface: Open IDoc for trace	Simulates the creation of an IDoc. No number specification and database operations (multiprocessing allowed).
		EDI_DOCUMENT_OPEN_FOR_EDIT		open in edit mode with locking of the IDoc
		EDI_DOCUMENT_OPEN_FOR_PROCESS	Open IDoc for processing	To change status records and control record (multiprocessing allowed)
		EDI_DOCUMENT_OPEN_FOR_READ	EDI interface: Open IDoc for reading	status of the IDoc cannot be changed (multiprocessing allowed)

II. Processing Idoc (after step I):		
EDI_SEGMENT_GET_NEXT	EDI interface: Sequential accesses to data record	can be called in READ and PROCESS modes
EDI_SEGMENT_GET	EDI Interface: Direct Access to Data Record	
EDI_SEGMENTS_GET_ALL	EDI interface: Read all data records for IDoc	
EDI_SEGMENTS_GET_RANGE	EDI interface: Read part of section from data records for IDoc	
EDI_DOCUMENT_READ_ALL_STATUS	IDoc interface: Read all status records for one IDoc	
EDI_DOCUMENT_READ_LAST_STATUS	IDoc interface: Read last/current status record for IDoc	
EDI_DOCUMENT_STATUS_SET	EDI interface: Set status	in CREATE or PROCESS mode
EDI_STATUS_ADD_BLOCK	EDI interface: Insert block of status records	
EDI_SEGMENT_ADD_NEXT	EDI interface: Sequential insertion of data record	can only be called in CREATE mode
EDI_SEGMENTS_ADD_BLOCK	EDI interface: Add block of data records	

		III. Closing IDoc after Processing (after step II):		
		EDI_DOCUMENT_CLOSE_CREATE	EDI interface: Close IDoc after insertion	
		EDI_DOCUMENT_CLOSE_CREATE_TAB	EDI interface: Close IDocs after insertion with EDIDD table	Returns data of the created IDoc in the TABLES EDIDD parameter for furhter processing in parent program.
		EDI_DOCUMENT_CLOSE_FORCE	EDI interface: Close IDoc after open for force	Stored with the status 'Created incorrectly'
		EDI_DOCUMENT_CLOSE_TRACE		No IDoc number is specified nor is the IDoc written onto the database.
		EDI_DOCUMENT_CLOSE_EDIT		
		EDI_DOCUMENT_CLOSE_PROCESS	EDI interface: Close IDoc after processing	New status records and a new status value in the control record - are passed on to the database
		EDI_DOCUMENT_CLOSE_PROCESS_UPD	EDI interface: Close IDoc after processing in update task	
		EDI_DOCUMENT_CLOSE_READ	EDI interface: Close IDoc after reading	
		IV. Standalone function modules:		
		IDOC_READ_COMPLETELY		Read all control, data and status records for the IDoc
EDI5	EDI: display data, status, ctrl record	**EDI_DOCUMENT_STATUS_DISPLAY**	Display last status record for current IDoc	like in WE02

XML:

Function group name	Function group description	Function module name	Function module description	Description	
SDIXML	(Since 4.6D, in 4.6C - beta)	**SDIXML_DATA_TO_DOM**		Convert SAP data (elementary/structured/table types) into DOM (XML)	Example
		SDIXML_DOM_TO_XML		Convert DOM (XML) into string of bytes that can be downloaded to PC or application server	

		SDIXML_DOM_TO_SCREEN		Display DOM (XML)	
		SDIXML_DOM_TO_DATA			
All this can be made using the objects except of SDIXML_DOM_TO_DATA and SDIXML_DATA_TO_DOM (ABAP OO-versions will be soon). **Example** in OO-style.					
EDIN	IDoc: Inbound processing	**IDOC_XML_FROM_FILE**	standard program for inbound processing that imports an XML file from the file system, converts it into IDoc format, and passes it to ALE layer.		

Miscellaneous:

Function group name	Function group description	Function module name	Function module description	Description
SAL2	Access routines for RZL storage	**RZL_SLEEP**	Sleep	Up to 5 seconds
SENT	ENQUEUE utilities and test	**ENQUE_SLEEP**	Sleep	Without restrictions
QF05	Random number generators	**QF05_RANDOM**	Random number generator	Returns random number between 0 and 1
		QF05_RANDOM_INTEGER	Random (whole) number	Returns random integer number between *min* and *max*

ALFA	ALPHA conversion	**CONVERSION_EXIT_ALPHA_INPUT**	Conversion exit ALPHA, external->internal	Used especially with account numbers. Numeric fields (only numbers, can be spaces before and after them):
				`"  1234    " -> "0000001234"`
				`"0000001234" -> "1234      "`

		CONVERSION_EXIT_ALPHA_OUTPUT	Conversion exit ALPHA, internal->external	

Miscellaneous: Static Methods of ABAP Objects:

Class	Class description	Static method	Static method description	Comments
cl_abap_math	ABAP: Math Library	**round_f_to_15_decs**	round variable of type f to 15 decimals	Sometimes type F (float) is used to represent exact data (Amounts, Quantities): • if values are to big to be represented as Packed (> 31 decimals) • to significally improve performance of calculations, e.g. in statistical reports This method helps to round float data correctly: **Example**
cl_abap_memory_utilities	Utilities for ABAP Memory Management	**do_garbage_collection**	Execute Garbage Collection	Called automatically by SAP to free memory for objects created by CREATE OBJECT ... and CREATE DATA ... statements
since 4.6				

SD:

Function group name	Function group description	Function module name	Function module description	Description
V05C	RV SD Document Flow Cumulative Qties	**RV_ORDER_FLOW_INFORMATION**	Reads sales document flow of sales document after delivery and billing	**Example** Don't forget also check direct reference documents for the both starting document # and preceding/following document types that you are searching for (For example, if you search delivery for the given SO, check also LIPS-VGBEL and LIPS-VGPOS, if it's possible with regard to performance).
V45A	SD Functions for Copying Documents	**SD_SALES_DOCUMENT_READ**		Reads sales document header and business data: tables VBAK, VBKD and VBPA (Sold-to (AG), Payer (RG) and Ship-to (WE) parties)

		SD_SALES_DOCUMENT_READ_POS		Reads sales document header and item material: tables VBAK, VBAP-MATNR
		SD_DOCUMENT_PARTNER_READ		partner information including address. Calls SD_PARTNER_READ
		SD_DETERMINE_CONTRACT_TYPE		In: at least VBAK-VBELN Exceptions: NO CONTRACT | SERVICE_CONTRACT | QUANTITY_CONTRACT
		SD_SALES_DOCUMENT_COPY	copy Sales Doc into new one with the required Sales Doc Type (VBAK-AUART) for further creating.	**Example** - create subsequent document
		SD_SALES_DOCUMENT_SAVE	create Sales Doc from the copied document	
V45K	Sales Order Processing Functions	**SD_SALES_DOCUMENT_ENQUEUE**		to dequeue use DEQUEUE_EVVBAKE
V05I	SD Index Update	**SD_PARTNER_READ**		all the partners information and addresses
V05O	SD Data Collection for Dlv Note Printing	**RV_DELIVERY_PRINT_VIEW**	Data provision for delivery note printing	
		SD_PACKING_PRINT_VIEW		
		SD_DELIVERY_VIEW	Data collection for printing	called from RV_DELIVERY_PRINT_VIEW, SD_PACKING_PRINT_VIEW
V05N	SD Data Collection for Bill Doc Printing	**RV_BILLING_PRINT_VIEW**	Data Provision for Billing Document Print	

MM:

Function group name	Function group description	Function module name	Function module description	Description
OMCV	Material Number Conversion	**CONVERSION_EXIT_MATN1_OUTPUT**	Material number conversion (OUTPUT)	E.g. 00001234 -> 1234
		CONVERSION_EXIT_MATN1_INPUT	Material number conversion (INPUT)	E.g. 1234 -> 000000000000001234
For material numbers conversion, in contrast to simple ALFA conversion, more complicated logic is used including customer exit MGA00003 (see transaction SMOD).				
1001UEB	BAPIs for Creating/Changing Matl Data	**BAPI_MATERIAL_SAVEDATA**	Create and Change Material Master Data	Very useful function module to create new material master data or to change existing material master data, although it is not used directly as method in any BAPI object, and therefore cannot be found via transaction **BAPI**. See *Function Module Documentation*
MGDS	Screen Sequence Control: Material Master	**SELECTION_VIEWS_FIND**	Determine Views for View Selection or View Configuration	One of functional modules for those who still has to use BDC for Material Master transactions (MM01 and MM02). Allows to determine line number for desired view in the "Select View(s)" window, to prepare BDC for the view access. See **skeleton** as complete example.
EINR	Read Purchasing Document	**ME_READ_HISTORY**	Read History of Purchasing Document	**Example**
MESHOW	Display Purchasing Documents	**ME_DISPLAY_PURCHASE_DOCUMENT**	Will be called: I_EDIT='X' => ME22 I_DISPLAY_ONLY='X' or default => ME23 I_ENJOY='X' => new transactions: ME23N / ME22N I_PREVIEW='X' => screen preview for default output type	
MEWP	BAPIs Purchase order	**BAPI_PO_CREATE**	Create purchase order	Both BAPIs create an PO, but BAPI_PO_CREATE1:

2012	BAPI in the Enjoy Purchase Order	**BAPI_PO_CREATE1**		• allows much more things: entering conditions, test mode, hold uncomplete POs etc. • not listen in **BAPI** transaction (4.6C), because this method of **2012** business object is not released

MRP:

Function group name	**Function group description**	**Function module name**	**Function module description**	**Description**
CSS4	BOM explosions	**CS_BOM_EXPLOSION_MAT**	BOM explosion for material	old version (before 3.0)
		CS_BOM_EXPL_MAT_V2		Explode BOM for Production. Mostly necessary input parameters: • CAPID (Application Id): 'PP01' (Production - general) • DATUV (Validity date) • MTNRV (Material) • WERKS (Plant)
		CS_BOM_EXPL_KND_V1		Explode BOM for Sales and Distribution. Additional input parameters: • CAPID (Application Id): 'SD01' (Sales and Distribution) • VBELN (Sales order number) • VBPOS (Item in sales order)
		CS_BOM_EXPLOSION	General BOM explosion	Internal function module, called by all FMs from function group CSS4 (for material as well as for equipment, Project Position, Document etc.)

CSS5	BOMs: where-used list	**CS_WHERE_USED_MAT**	BOMs: where-used list	returns BOM##, parent materials and other information for component. Additional criterias: valid dates, item category and BOM usage, plant (can be '*' to select all plants).
		CS_WHERE_USED_MAT_VIA_CLA	Bills of material; where-used list via classes	
		CS_WHERE_USED_MAT_ANY	Bills of material; where-used list as material or class item	
In SAP 4.6C this function group is used in report RCS15001 (Material Where-Used List), which called from transaction CS15 (Single-Level Where-Used List)				
CSEP	BOM - APIs for Enterprise Portal	**CSEP_MAT_BOM_SELECT_WHERE_USED**	API Bills of Material: Select BOM(s)	New and simplified function module. Input: only component name (material number). Return the same as function modules above.

FI-CO:

Function group name	Function group description	Function module name	Function module description	Description
KAB2	CO Reporting: General Functions	**K_GROUP_SELECT**	Searching for cost-object groups via a pop-up window	Good for F4 (on value-request) function. Imports a field name and returns chosen Group name for the field: KSATR – Cost element group; RACCT – Account group PRZNR – Business process group; KSTRG – Cost object group KOSTL – Cost center group; AUFNR – Order group LSTAR – Activity type group; STAGR – Statistical key figure group SKOST – Sender cost center group; SLSTA – Sender activity group PRCTR RPRCTR – Profit center group; VORNR – Network activity group POSID **Example**.

		K_DOCUMENT_SELECT	CO: searches a document number according to certain criteria	also good for F4
KKHI	Group Maintenance	**K_F4_STANDARD_HIERARCHY_VALUE** **K_F4_STANDARD_HIERARCHY_NODE**	Displays hierarchy tree for the given SETCLASS and Valuation Area (requested in separate dialog window), returns selected value | node. Import parameter SETCLASS: values see in GSETC type-pool. Following standard hierarchies allowed in the FMs: • 0101 - Cost Center • 0106 - Profit Center • 0107 - Business Process **Example**	

FI-SL Sets:

Function group name	Function group description	Function module name	Function module description	Description
GSGE	SETS: Generating and reading sets	**G_SET_GET_ALL_VALUES**	Read All Values in a Set Hierarchy	Imports a set Id and table name and returns an internal table filled with elements of the set. 3.1H **Example** (set Id = 0H + Group name). 4.6C **Example** (retrieve set Id using G_SET_GET_ID_FROM_NAME).
GSAC	SETS: access to set tables	**G_SET_TREE_IMPORT**	Importing a Set Hierarchy	Imports a set Id and table name and returns two internal tables filled with 1. elements of the set; 2. hierarchy tree of the set. 3.1H **Example** (set Id = 0H + Group name). 4.6C **Example** (retrieve set Id using G_SET_GET_ID_FROM_NAME).

		G_SET_GET_ID_FROM_NAME	Determining a Set ID from the Set Name and Other Information	Derives the internal set Id from the name that appears on the user interface, e.g. in order then to import the set (see the function group and function module documentation). To take in account changes in SET concept in 4.6C: see examples to G_SET_GET_ALL_VALUES and G_SET_TREE_IMPORT.
GSSM	SETS: Set-Manager			

See also example reports:

- RGSEX001 - Read-only access with G_SET_TREE_IMPORT
- RGSEX000, RGSEX010, RGSEX020 - Access with modules from the function group GSSM
- RGSEX030 - Create a set

Tables:

- T800S - FI-SL Set Table (before Release 4.0)