PARSER MINOSSE

manuale dell'utente

9 settembre 2015

www.linformatica.com

Indice generale

IL PARSER MINOSSE

Introduzione

Nei software de "L'Informatica", viene spesso usato il parser "Minosse", per risolvere le formule e dare così la possibilità di personalizzare le modalità di calcolo ma anche testi e formule di vario tipo.

Semplificando molto, Minosse può essere visto come una specie di calcolatrice che risolve le formule scritte dall'utente.

La sintassi delle formule di Minosse è simile a quella dei software di elaborazione fogli di calcolo, come Excel o OpenOffice Calc, sebbene ci siano sostanziali differenze.

Questa appendice è generica, non legata strettamente al programma nel cui manuale essa è inserita, proprio perché il parser Minosse ha un uso generico, non specifico, così come si può usare Excel per fogli di calcolo del tutto differenti tra di loro ma con la stessa sintassi delle formule; viene riportata per completezza.

Le formule

Le formule accettate dal parser possono andare dalle più semplici alle più complesse. Si possono scrivere quelle aritmetiche come si è abituati, usando le parentesi tonde quando necessario:

```
1 + 1
3 * (2 + 4)
((5 + 2) * ( 6 + 8) + 1) * 9
```

Di solito non si deve mettere il simbolo di "=", davanti alle formule, come si farebbe invece con un foglio di calcolo.

Negli esempi si usano molti spazi, per rendere leggibili le formule; nell'uso non sono necessari ed è possibile evitarli:

```
1 + 2 * (3 + 4)
1+2*(3+4)
```

Le formule, come anticipato, possono contenere sia variabili e costanti che funzioni, con i loro eventuali parametri.

Le costanti

Le costanti sono dei valori che hanno un valore fisso, che non cambia e

che è ben determinato, perché inserito manuamente dall'utente. Le costanti sono essenzialmente di due tipi: numeriche e stringa (testuali).

Le costanti numeriche

Le costanti numeriche rappresentano numeri. Un esempio di costanti numeriche è questo:

```
1 + 2
```

Sia 1 che 2 sono costanti di tipo numerico mentre il + è il normale operatore di somma. Il parser Minosse accetta costanti numeriche intere o con cifre decimali; in quest'ultimo caso, si deve usare il punto (.) decimale e non la virgola:

```
1.25 + 2.5
```

Le costanti negative hanno semplicemente il segno meno:

```
-3.45 * 8.25
```

Per ragioni di compatibilità con vecchi software de L'Informatica tutt'ora in uso, non è ammesso far seguire il – ad un operatore:

```
-3.45 * -8.25 non ammesso
-3.45 * (0 - 8.25) ok
```

Si può usare il simbolo E, per indicare l'esponente a base 10, ma solo se il numero ha il punto decimale:

```
1.52E3 → 1520
1.0E5 → 100000
1.E2 → 100
1E2 → non ammesso (E2 viene ignorato e vale 1!!!)
```

Le costanti stringa

Minosse accetta anche costanti di tipo testo (stringhe), che possono essere delimitate a scelta con l'apice singolo (') o con le virgolette (“):

```
'questo è un testo'
“anche questo lo è!”
```

Nei programmi eminentemente di calcolo (come Tabì) le costanti di tipo stringa verranno raramente usate e questo paragrafo può essere saltato.

Le stringhe che devono contenere il delimitatore devono necessariamente

indicarlo raddoppiato, per non “confondere” il parser stesso, che non saprebbe dove inizia e dove finisce il testo:

'L''Informatica di Gemma Stefano'
'La ricerca del vello d''oro'

In alternativa, è sufficiente delimitare le stringhe con l'altro simbolo:

“L'Informatica di Gemma Stefano”
“La ricerca del vello d'oro”
'Questo simbolo “ si chiama “virgolette” '

Non si deve confondere il doppio apice '' con le virgolette “. Il primo è il simbolo dell'apostrofo ripetuto due volte mentre il secondo è quello che normalmente si trova sopra al tasto “2” delle tastiere italiane (usato sia per iniziare che per terminare una stringa).

Le formule che contengono stringhe possono usare funzioni oppure operatori sul testo ma anche mischiare stringhe e numeri. L'operatore più comune è quello di concatenamento &:

'Questa è una ' & 'stringa'

equivale a:

'Questa è una stringa'

L'utilità dell'operatore di concatenamento è principalmente legata all'uso di funzioni.

Se si cercano di concatenare una stringa ed un numero, quest'ultimo verrà convertito in stringa:

'Il risultato è: ' & (2+2)

equivale a:

'Il risultato è: 4'

Se si somma una stringa ad un numero, sarà invece la stringa ad essere convertita in numero, ignorando i caratteri che non sono convertibili:

2 + '3 volte 6'

equivale a:

2 + 3 = 5

perché ' volte 6' verrà ignorato.

Riassumendo:

- & concatena stringhe di testo
- + somma numeri

Le Variabili

Le variabili sono dati che hanno un valore non pre-determinato o comunque che può variare, durante l'elaborazione. Leggendo semplicemente la formula che le contiene, non si può sapere che valore esse abbiano. Normalmente una variabile rappresenta un dato calcolato in formule precedenti o letto da un archivio.

Le variabili hanno un nome e rappresentano un valore, che può essere sia numerico che stringa (ma anche altri tipi di dato). Il valore di una variabile dipende dal contesto e dal momento in cui viene usata. Nel software Tabì, ad esempio, la variabile chiamata SUPERFICIE rappresenta la superficie virtuale di un'unità immobiliare. In questo contesto, la formula:

SUPERFICIE

ha un valore che dipende dall'unità immobiliare che si sta elaborando. Il software Tabì, infatti, elabora le unità immobiliari ad una ad una ed applica la formula sostituendo il nome SUPERFICIE con il valore che via via esso rappresenta; per l'unità 1 potrà valere 120.50 mentre per la 2 potrà valere 98.75, ad esempio.

Si può pensare ai nomi delle variabili come ai nomi delle celle di un foglio di calcolo. Certamente è più leggibile il nome SUPERFICIE che non il nome F5.

I nomi delle variabili possono essere scritti sia in maiuscolo che in minuscolo e possono contenere i simboli _ e .

CODICE_PRODOTTO
CLIENTI.NOME
_TIPO

I nomi utilizzabili dipendono dal programma, perché è quest'ultimo in genere che crea a valorizza le variabili; alcune variabili possono però essere create dall'utente, ad esempio: quando si assegna il codice di una tabella in Tabì, questo codice diventa il nome della variabile che rappresenta i millesimi di quella tabella.

Le variabili numeriche

Le variabili numeriche rappresentano numeri, come le corrispondenti costanti. Nelle formule si possono usare sia variabili che costanti numeriche:

SUPERFICIE * 1.10

Si possono anche modificare i valori delle variabili, usando la funzione FSET, che vedremo più avanti.

Le variabili stringa

Le variabili stringa rappresentano testi, come le corrispondenti costanti. Nelle formule si possono usare sia variabili che costanti stringa:

'Il nome del cliente è: ' & CLIENTI.NOME

Tutte le considerazioni fatte per le costanti stringa valgono anche per le variabili stringa. I nomi delle variabili stringa non devono però essere racchiusi da apici o da virgolette, come invece avviene per le costanti stringa.

Le Funzioni

Le funzioni sono espressioni che, dati alcuni parametri, forniscono un valore. Si ha sicuramente familiarità con funzioni come la radice quadrata, che calcola appunto la radice quadrata di un numero. In Minosse, ci sono sia funzioni per i numeri che per le stringhe, per effettuare tutta una serie di calcoli o di elaborazioni. Molte di queste funzioni sono riservate ad un uso avanzato o ad altri software, diversi da quello descritto in questo manuale.

Verrà fornita assistenza per l'uso delle funzioni effettivamente utili per questo software, le altre sono riportate solo per completezza ma non dovranno essere utilizzate e sono escluse dall'assistenza.

Nella tabella che segue, i parametri vanno inseriti tra parentesi, dopo il nome della funzione:

FUNZIONE(parametro1, parametro2)

Quando un parametro è opzionale (può quindi essere omesso) si usano le parentesi quadre [,parametro].

Quando uno o più parametri possono essere ripetuti, si usa '…'.

FUNZIONE	PARAMETRI	DESCRIZIONE
GESTIONE VARIABILI		
FSET	variabile, valore	Assegna e ritorna il nuovo valore della variabile FSET('ANNO', 2015) → 2015
FV	nome_variabile	Ritorna il valore della variabile, se esiste, altrimenti ritorna una stringa vuota: FSET('ANNO', 2015) ANNO → 2015 NON_ESISTE → ***ERRORE*** FV('ANNO') → 2015 FV('NON_ESISTE') → ''
FVAR	nome_variabile	Come FV ma ritorna il numero 0 se la variabile non esiste: FSET('SOMMA', 100) FVAR('SOMMA') → 100 FVAR('PRODOTTO') → 0
FVARKO	nome_variabile	Ritorna 0 se la varibile esiste altrimenti ritorna un valore diverso da 0 (è il contrario di FVAROK)
FVARN	nome_variabile	Se la variabile esiste ritorna il suo valore come numero altrimenti ritorna 0 equivale a FN(FVAR('nome_variabile'))
FVARNAMES	nome_variabile, separatore	Ritorna l'elenco delle variabili che hanno il nome che inizia con il nome_variabile; se si aggiunge il separatore, ritorna l'elencio nella forma "nome=valore” ed il separatore: FVARNAMES('X', ' | ') → 'X1=20 | X2=100...' FVARNAMES('X') → 'X1,X2...'
FVAROK	nome_variabile	Ritorna 0 se la variabile non esiste, altrimenti ritorna un valore diverso da 0 (è il contrario di FVARKO)
FUNZIONI NUMERICHE		
FABS	numero	Valore assoluto di un numero esempio: FABS(3-5) → 2
FINT	valore, tipo	Arrotonda all'intero: tipo = 0 → arrotondamento matematico tipo = 1 → arrotonda per eccesso tipo = 0-1 → arrotonda per difetto
FMOD	valore, modulo	Resto della divisione tra due numeri
FROUND	Valore, modulo	Arrotonda il valore in base al modulo esempio: FROUND(15, 2) → 14
FSQR	numero	Radice quadrata del numero (>=0.0)
FUNZIONI TRIGONOMETRICHE E GEOMETRICHE		

FUNZIONE	PARAMETRI	DESCRIZIONE
FACOS	numero	Arco-coseno di un numero → angolo in radianti
FASIN	numero	Arco-seno di un angolo → angolo in radianti FDEG(FASIN(FSQR(2)/2)) → 45 il seno di 45° = radice di 2 fratto 2 l'arco-seno della radice di 2 fratto 2 = 45° i calcoli vengono fatti in radianti: FDEG converte in gradi
FATAN	numero	Arco-tangente di un numero → angolo in radianti
FCOS	numero	Coseno di un angolo in radianti
FDIST	xa, ya, za, xb, yb, zb	Distanza tra 2 punti in 3 dimensioni
FSIN	numero	Seno di un angolo in radianti: FSIN(0.7853981633974) → 0.7071067812 FSIN(FRAD(45)) → 0.7071067812
FTAN	numero	Tangente di un angolo in radianti
FDEG	numero	Converte un angolo da radianti a gradi
FRAD	numero	Converte un angolo da gradi a radianti
FTAN	numero	Tangente di un angolo in radianti
FUNZIONI STRINGA		
FANY	stringa1, separatore1, formula1, separatore2	Estrae i parametri nella stringa1, divisi dal separatore1; per ogni parametro estratto, risolve l'espressione nella formula1 (una stringa), sostituendo ogni occorrenza di #ANY# con l'ennesimo parametro estratto da stringa1; i risultati vengono concatenati con il separatore2. Gli eventuali simboli | contenuti in formula1 vengono sostituiti da apici. Esempio: FANY('139 168 273', ' ', 'FT(|ES04|,|#ANY#|)', ',') è equivalente a: FT('ES04','139') & ',' & FT('ES04','168') & ',' & FT('ES04','273')
FFILL	stringa, numero, carattere	Aggiunge tanti volte il carattere indicato fino a portare la stringa alla lunghezza voluta; se numero>0 la riempie a sinistra, sennò la riempie a destra: FFILL('A', 4, 'X') → 'XXXA' FFILL('A', 0-4, 'X') → 'AXXX'
FFULLSEARCH	tipo, stringa1, stringa2 ...	Come FSEARCH ma il parametro tipo può cambiare la modalità di ricerca: "=", "#<", "#>", "#"
FINSTR	cerca, stringa [,inizio]	Ritorna la posizione di "cerca" nella "stringa", contando i caratteri da 0: FINSTR("B", "A,B,C") -> 2

FUNZIONE	PARAMETRI	DESCRIZIONE
FLEN	stringa	Lunghezza in caratteri della stringa
FMID	stringa, inizio, numero	Ritorna "numero" caratteri della stringa, partendo dal carattere alla posizione "inizio" (1 è il primo carattere)
FN	stringa	Converte la stringa in numero (usa il punto per i decimali)
FCURR	Stringa [,migliaia]	Converte la stringa in numero e lo riconverte in stringa di tipo valuta: FCURR('1000.0') →'1000,00' FCURR('1000.0', 0) →'1000,00' (come sopra) FCURR('1000.0', 1) →'1.000,00'
FNE	Stringa1 [,stringa2...]	Ritorna la prima stringa non vuota
FCASE	stringa, tipo	Converte la stringa a seconda del tipo indicato: 0: in maiuscolo 1: in minuscolo 2: in minuscolo ma con prima lettera in maiuscolo 3: in minuscolo ma con tutte le prime lettere in maiuscolo
FCHAR	numero	Converte il numero nel corrispondente carattere ASCII
FONLYONE	stringa, separatore	Converte ogni sequenza di 'separatore' in uno solo esempio: FONLYONE('1,2,,3,,,4') →'1,2,3,4'
FQ	stringa	Ritorna la stringa racchiusa tra apici, raddoppiando quelli eventualmente contenuti esempio: FQ('L'Informatica') →'L''Informatica'
FS	stringa1 [,stringa2...]	Concatena più stringhe (come l'operatore & ma più veloce) FS('Questa', ' è ', ' una stringa') →'Questa è una stringa'
FSORT	stringa, separatore	Ordina gli elementi contenuti nella stringa, riconoscendoli in base al separatore indicato: FSORT('5/3/1/2', '/') →'1/2/3/5'
FSS	separatore, stringa1 [,stringa2...]	Concatena le stringhe (come FS e &) aggiungendo il separatore indicato: FSS('/', 'gg', 'mm', 'aaaa') →'gg/mm/aaaa'
FSUBST	stringa, cerca, sostituisci	Sostituisce parti di una stringa: FSUBST('ab cd', 'b', '123') →'a123 cd'
FTOKEN	stringa, separatore, indice [,indice_default])	Ritorna l'elemento della stringa indicato da indice (il oprimo è 0); se l'elemento è vuoto, ritorna quello indicato da indice_default: FTOKEN('a:b:c:d', ':', 0) →'a' FTOKEN('a:b:c:d', ':', 1) →'b' FTOKEN('a:b:c:d', ':', 7, 0) →'a'

FUNZIONE	PARAMETRI	DESCRIZIONE
FTOKENL	stringa, separatore	Ritorna l'elemento a sinistra del separatore: FTOKENL('x=123', '=') →'x' FTOKENL('x=123', '+') →'x=123'
FTOKENLW	stringa, separatore, indice	Come FTOKEN ma ritorna tutta la stringa prima dell'indice: FTOKEN('a:b:c:d', ':', 1) →'b' FTOKENLW('a:b:c:d', ':', 1) →'a:b'
FTOKENR	stringa, separatore	Ritorna l'elemento a destra del separatore: FTOKENR('x=123', '=') →'123' FTOKENR('x=123', '+') →''
FTOKENRW	stringa, separatore, indice	Come FTOKEN ma ritorna tutta la stringa dopo l'indice: FTOKEN('a:b:c:d', ':', 1) →'b' FTOKENLW('a:b:c:d', ':', 1) →'b:c:d'
FTRIM	stringa, separatore	Elimina il separatore all'inizio e alla fine della stringa: FTRIM(' A B C ', ' ') →'A B C'
FTRIML	stringa, separatore	Elimina il separatore all'inizio della stringa: FTRIM(' A B C ', ' ') →'A B C '
FTRIMR	stringa, separatore	Elimina il separatore alla fine della stringa: FTRIM(' A B C ', ' ') →' A B C'
FUNZIONI SU TABELLE DI DATI		
FGET	tabella, campo, condizione	Ritorna il valore del campo nella tabella indicata, in base alla condizione
FGETF	tabella, campo, parametro1...	Come FGET ma i vari parametri vengono usati per il parametro FastGet della tabella stessa: FastGet: A1=, B1MIN<=, _<=B1MAX FGETF('TABLE', 'FIELDNAME', FQ(A1), FQ(B1))
FGETX	tabella, campo, parametro1...	Ritorna valori da una tabella con ricerca avanzata: FGETX('TABLE', 'FIELDNAME', 'A1=', FQ(A1), 'B1MIN<=', FQ(B1), 'B1MAX>=', FQ(B1)) FGETX('TABLE', 'FIELDNAME', 'A1=', FQ(A1), 'B1MIN<=', FQ(B1), '_<=B1MAX')
FINTER	tabella, x	Ritorna il campo Valore della tabella, interpolando sulo campo X
FLIST	tabella, campi, condizione, separatore [,filtro:campo, filtro:tabella,	Estrae una lista di valori da una tabella, riunendoli in base al separatore e con un filtro opzionale (inner join sulla seconda tabella)

FUNZIONE	PARAMETRI	DESCRIZIONE
	filtro:valore]	
FMAX	tabella, x [,y]	Ritorna il più alto Valore della tabella indicata, in base alle chiavi X ed eventualmente Y
FMIN	tabella, x [,y]	Ritorna il più basso dei valori della tabella indicata, in base alle chiavi X ed eventualmente Y
FRCOUNT	tabella, condizione	Conta le righe della tabella indicata
FSEARCH	chiave [,parametri...]	Funzioni varie di ricerca (riservate) Esempi: Fsearch("table", tabella) Fsearch("tables", cartella) FSearch("tables formula", formula) Fsearch("folder", cartella) Fsearch("valore", campo, valore1 [,value2...])
FT	tabella, x [,y]	Legge il campo Valore dalla tabella indicata, con chiavi X ed eventualmente Y
FTSEARCH	valore1 [, valore2]	Equivale a FSEARCH("value","valore",value1[,value2...])
FTY	tabella, indice, x [,y]	Come FT ma ritorna il valore di un campo array: valore[indice]
FUNZIONI MACRO		
FMACRO	macro, parametro1...	Legge la macro indicata, sostituendo i parametri #1#, #2# ecc. con i corrispondenti parametri passati. Esempio macro = "#1# + #2#” FMACRO("macro", 3, 4) ritorna "7”
FMACROADD	macro, espressione	Salva la macro indicata, con l'espressione come stringa (non risolta)
FMACROCLEAR		Cancella tutte le macro
FMACRODEL	macro	Cancella la macro indicata
FMACROGET	macro, parametro1...	Come FMACRO ma l'espressione viene ritornata con i parametri sostituiti e senza essere calcolata. Esempio macro = "#1# + #2#” FMACRO("macro", 3, 4) ritorna "3 + 4”
FSCRIPT	comandi	Esegue una serie di comandi in linguaggio script-minosse (riservato e non documentato).
FUNZIONI SU LISTE		
FMIX	Stringa1, separatore1, stringa2, separatore2, separatore3, separatore4	Estrae dalla stringa1 i parametri separati da separatore1, idem per stringa2/separatore2; fonde i primi con i secondi, separandoli con separatore3; ogni coppia viene unita dal separatore4. Esempio: FMIX('10-20-30', '-', 'A:B:C', ':', '.', '/') →

FUNZIONE	PARAMETRI	DESCRIZIONE
		'10.A/20.B/30.C'
FMUL	separatoreA, separatoreB, lista1, lista2	Moltiplica i valori estratti dalla lista1 e dalla lista2, in base al separatoreA; crea una nuova lista i cui valori sono separati dal separatoreB; se lista2 e separatoreB sono assenti, ritorna il prodotto dei valori della lista1: FMUL(' ', '/', '1 3 5', '2 3 4') →1*2 3*3 5*4 →'2/9/20' FMUL(' ', '1 3 5') →1*3*5 →15
FNOREP	lista, separatore1 [,separatore2]	Cancella tutti i parametri ripetuti nella lista ed eventualmente somma il primo campo: FNOREP('A,B,A,C', ',') →'A,B,,C' FNOREP('2 A,1 B,3 A,1 C',',',' ') →'5 A,1 B,,1 C'
FSUM	separatoreA, separatoreB, lista1, lista2	Come FMUL ma somma, invece di moltiplicare.
FUNZIONI PER FOGLI DI CALCOLO		
FSUM	foglio.inizio:fine	Calcola la somma di un gruppo di celle: FSUM(FOGLIO1.B4:C6)
FUNZIONI CONDIZIONALI, DI SELEZIONE ED OPERATORI LOGICI		
AND	espressione1, espressione2, [,espressione3...]	Vero se le espressioni sono tutte diverse da 0; si può usare sia come funzione che come operatore: A AND B →vero se A e B sono diversi da 0 AND(A, B,...) →vero se tutti i parametri sono diversi da 0
FEQ	espressione, valore1 [, valore2...]	Vero se uno dei valori corrisponde all'espressione indicata. Ad esempio: FQ(**1**, 2, 3, 4, 5) →falso FQ(**1**, 2, 3, **1**, 6) →vero
FP	indice, parametro1 [,parametro2...]	Ritorna il parametro all'indice indicato esempi: FP(2, 'a', 'b', 'c') →'b' FP(mese, 'gennaio', 'febbraio', 'marzo', 'aprile') → 'marzo' se mese=3
FSELECT	stringa, test1, valore1 [,test2, valore2...]	Ritorna il valore corrispondente al primo testo che corrisponde con la stringa esempio: FSELECT(PIANO, 'PT', 1, 'P1', 2, 'P2', 3) se PIANO='PT' ritorna 1 se PIANO='P1' ritorna 2

FUNZIONE	PARAMETRI	DESCRIZIONE
		FSELECT(GIORNO, 'lunedì', 1, 'martedì', 2...) se GIORNO='martedì' ritorna 2

FVALIDATE	expr1, valore1 [,expr2, valore2 ...]	Ritorna il primo valore in cui l'espressione corrispondente dà un valore diverso da zero:
		
		FVALIDATE(A1='100', 'X100', B1='Z', 'XZ')
		
		equivale a:
		
		IF(A1='100', 'X100', IF(B1='Z', 'XZ', ''))
		
		ed equivale a:
		
		FVALIDATE("A1='100'", 'X100', "B1='Z'", 'XZ')
		
		Si può usare il simbolo $ per contenere il parametro:
		
		FVALIDATE(" $ #'+' ", 'ABCD+') → 'ABCD+'
		FVALIDATE(" $ #!'+' ", 'ABCD+') → ''
		
		FVALIDATE(" $ #!'+' ", 'ABCD+', "FLEN($)=3", 'AAA') → 'AAA'
		FVALIDATE(" $ #'+' ", 'ABCD+', "FLEN($)=3", 'AAA') → 'ABCD+'
		
		IF(A,B,'') equivale a FVALIDATE(A,B)
		
		IF(A,B,IF(C,D,'')) equivale a FVALIDATE(A,B,C,D)

NOT	espressione1 [espressione2,...]	Ritorna vero (un valore diverso da 0) se tutte le espressioni sono uguali a 0, altrimenti ritorna 0: Si può usare come funzione: NOT(1, 0, 3) → falso → 0 NOT(0, 0, 0) → vero → 1 o come operatore: NOT 123 → falso → 0 NOT 0 → vero → 1 NB: 'vero' può essere un qualsiasi valore diverso da 0, non necessariamente 1
OR	espressione1, [,espressione2...]	Vero se almeno un'espressione è diversa da 0; si può usare sia come funzione che come operatore: A OR B → vero se A o B o entrambi sono diversi da 0 AND(A, B,...) → vero se almeno un parametro è diverso da 0
FUNZIONI SPECIALI		
FEXECUTE	espressione	Risolve l'espressione stringa in una sessione separata del

FUNZIONE	PARAMETRI	DESCRIZIONE
		parser, come se fosse una formula: FE('1+1') → 2
FPARSE	tipo, formula	Esamina la formula e ritorna valori dipendenti dal tipo: tipo = 'P' → lista dei parametri Esempio: FPARSE('A+2*FINT(5.3)') → 'A,2,5.3'
FPROFn		Funzioni riservate
FSETOPT	opzione, valore	Assegna il valore all'opzione (riservato)
FSETVARS	variabili, separatore, azzera	Assegna i valori delle variaqbili contenute nella stringa, delimitate dal separatore indicato. Se 'azzera' è diverso da zero, le cancella, prima di assegnarle: FSETVARS('A1=ABCD|B1=XYZ', '|', 1)
FSWAP	variabile, valore	Come FSET ma ritorna il valore precedente: FSET('ANNO', 2010) → 2010 FSET('ANNO', 2015) → 2015 FSWAP('ANNO', 2020) → 2015
FTEXT	file	ritorna il contenuto del file di testo indicato in una stringa
MACRO RISERVATE		
FE	funzione_esterna	Valore della funzione esterna indicata
FNOTA	nota1 [,notan...]	Vero se esiste una delle variabili NOTA_notaN