

Coal

言語仕様書編

第8章

第 5. 8 版

2014年7月17日

目 次

8. 組み込み関数	1
8.1. 関数一覧	1
8.2. 全般規則	7
8.3. ABS	7
8.4. ADD_MONTHS	7
8.5. ARRAYCPY	8
8.6. ARRAYCLR	9
8.7. ARRAYCMP	9
8.8. ARRAYBXP	10
8.9. ARRAYMAP	10
8.10. ASC	11
8.11. ATAN	11
8.12. ATANH	11
8.13. CBIN	12
8.14. CBRT	12
8.15. CBULK	12
8.16. CCHAR	12
8.17. CDATE	13
8.18. CDEC	13
8.19. CDOUBLE (CDBL)	13
8.20. CEIL	13
8.21. CFLOAT (CFLT)	14
8.22. CHANNEL	14
8.23. CHR	15
8.24. CINT	15
8.25. CLONG (CLNG)	15
8.26. CLOSEDIR	16
8.27. CONCAT	16
8.28. CONDAS	16
8.29. CONS	17
8.30. COS	17
8.31. COSH	17
8.32. COUNTV	17
8.33. CSTRING (CSTR)	18
8.34. DATE_ADD	18
8.35. DATE_DIFF	18
8.36. DECODE	19
8.37. EDIT	19
8.38. ELREAD1	20
8.39. ELWRITE	20
8.40. ELWRITE1	20
8.41. EVAL	20
8.42. EXIT	21
8.43. EXP	21
8.44. FCLOSE	21
8.45. FELREAD1	22
8.46. FELWRITE	23
8.47. FELWRITE1	24
8.48. FF	25
8.49. FGETLINE	25
8.50. FIRST	26
8.51. FL	26
8.52. FLOOR	27

8.53.	FOPEN	27
8.54.	[FP]STAT	28
8.55.	FPUTLINE	29
8.56.	GETARGS	30
8.57.	GETENV	31
8.58.	GETLINE	31
8.59.	GETLOGPARM	31
8.60.	GETMEMUSED	32
8.61.	GETTIME	32
8.62.	GETVAL	33
8.63.	GETWORD	33
8.64.	i IN	35
8.65.	i LIKE	35
8.66.	IN	35
8.67.	INDEXA	36
8.68.	INiRSTR	36
8.69.	INiSTR	36
8.70.	INLIKE	37
8.71.	INRSTR	37
8.72.	INSTR	38
8.73.	i REGEX	38
8.74.	IS	38
8.75.	LAST_DAY	39
8.76.	LEFT	39
8.77.	LEFTB	39
8.78.	LENG	40
8.79.	LENB	40
8.80.	LIKE	41
8.81.	LIST	41
8.82.	LIST_REF	42
8.83.	LOG	42
8.84.	LOG10	42
8.85.	LOGOUT	42
8.86.	LPAD	43
8.87.	LPADB	43
8.88.	MAX	44
8.89.	MIN	44
8.90.	MOD	44
8.91.	NDEF	44
8.92.	NEW	45
8.93.	NVAL	45
8.94.	NOFREE	45
8.95.	OPENDIR	45
8.96.	PCLOSE	46
8.97.	POPEN	46
8.98.	POWER	46
8.99.	PUTENV	47
8.100.	PUTLINE	47
8.101.	RAND1 (DRAND48)	47
8.102.	READDIR	48
8.103.	REGEX	48
8.104.	REP	49
8.105.	REPLIKE	49
8.106.	RESLOGPARM	51
8.107.	REST	51
8.108.	RIGHT	52

8.109.	R I G H T B	52
8.110.	R I N T	53
8.111.	R O U N D	53
8.112.	R P A D	54
8.113.	R P A D B	54
8.114.	S E T _ D A T E _ P A R T	55
8.115.	S E T A R R A Y	56
8.116.	S E T E N V	56
8.117.	S E T L O G P A R M	57
8.118.	S H E L L	57
8.119.	S H U T C T L	57
8.120.	S I N	58
8.121.	S I N H	58
8.122.	S K I P _ O P T	58
8.123.	S O R T	59
8.124.	S Q R T	61
8.125.	S R A N D 1 (S R A N D 4 8)	61
8.126.	S T R I N G S	61
8.127.	S U B S T R (M I D)	62
8.128.	S U B S T R B (M I D B)	62
8.129.	S Y S L O G	62
8.130.	T A N	63
8.131.	T A N H	63
8.132.	T I M E S	63
8.133.	T O	63
8.134.	T O _ B U L K	64
8.135.	T O _ B U L K S	64
8.136.	T O _ C H A R	65
8.137.	T O _ D A T E	65
8.138.	T O _ N U M B E R	66
8.139.	T R I M	66
8.140.	U N L I N K	66
8.141.	U N S E T E N V	67
8.142.	X H A S H	67

8. 組み込み関数

8.1. 関数一覧

表8.1-1 関数一覧（文字列関連）

No	関数名	機能	OP	戻り値	属性
1	I S	文字列の調査	○	調査結果	数値
2	T O	文字列の変換	○	変換結果	注1
3	R E P	文字列の置換	○	置換結果	文字
4	C O N D A S	文字列の条件式	○	条件式の実行結果	文字
5	S U B S T R M I D	文字列を切出す	○	切出し文字列	文字
6	C O N C A T	文字列を連結する	○	連結文字列	文字
7	G E T A R G S	文字列を指定文字で分解する	○	分解数	数値
8	T R I M	文字列の前後の空白文字を削除する	○	削除済み文字列	文字
9	S T R I N G S	文字列を繰り返す	○	繰り返し文字列	文字
10	L E F T	文字列の左側を取り出す	○	切出し文字列	文字
11	R I G H T	文字列の右側を取り出す	○	切出し文字列	文字
12	E D I T	データのフォーマッティング	○	編集結果	文字
13	R E P L I K E	文字列のL I K E置換	○	置換結果	文字
14	L E N G	データの長さまたは文字数を返す		長さまたは文字数	数値
15	L E N B	バイト単位で長さを返す		バイト長	文字
16	S U B S T R B M I D B	バイト単位で文字列を切出す	○	切出し文字列	文字
17	L E F T B	バイト単位で文字列の左側を取り出す	○	切出し文字列	文字
18	R I G H T B	バイト単位で文字列の右側を取り出す	○	切出し文字列	文字
19	G E T W O R D	ワード取り出し		取り出し文字列	文字
20	L P A D	文字列の左端に文字列を連結し、指定桁数にする	○	連結文字列	文字
21	L P A D B	文字列の左端に文字列を連結し、指定バイト数にする	○	連結文字列	文字
22	R P A D	文字列の右端に文字列を連結し、指定桁数にする	○	連結文字列	文字
23	R P A D B	文字列の右端に文字列を連結し、指定バイト数にする	○	連結文字列	文字

(注1) 変換数値の属性。(注) OP欄の"○"は演算子でも使用可を示す。

表8.1-1 関数一覧（演算関連 1/2）

No	関数名	機能	OP	戻り値	属性
1	ASC	文字をコードに変換する	○	文字コード	数値
2	CHR	コードを文字型に変換する	○	変換文字列	文字
3	TO_BULK	各型のデータをそのままバルク型に変換し、バイト単位で切り出す。	○	バルク・データ	BULK
4	TO_BULKS	各型のデータをそのままバルク型に変換し、連結する。	○	バルク・データ	BULK
5	TO_NUMBER	数値への変換	○	変換数値	注1
6	TO_CHAR	文字列への変換	○	変換文字列	文字
7	ABS	絶対値を求める		絶対値	数値
8	MAX	最大値を求める	○	最大値	注2
9	MIN	最小値を求める	○	最小値	注2
10	MOD	余りを求める	○	余り	数値
11	FF	関数名と同じ文字列を関数指定扱いにする		関数名	関数
12	EVAL	式を評価する		式の値	注3
13	NVAL	NUL L値を置き換える		式の値	注3
14	NDEF	未定義値を置き換える		式の値	注3
15	LIKE	パターン・マッチング	○	マッチング結果	数値
16	iLIKE	大文字、小文字を区別しないパターン・マッチング	○	マッチング結果	数値
17	REGEX	正規表現パターン・マッチング	○	マッチング結果	数値
18	iREGEX	大文字、小文字を区別しない正規表現パターン・マッチング	○	マッチング結果	数値
19	IN	文字列比較	○	比較結果	数値
20	iIN	大文字、小文字を区別しない文字列比較	○	比較結果	数値
21	INSTR	文字列サーチ	○	文字列の位置	数値
22	INiSTR	大文字、小文字を区別しない文字列サーチ	○	文字列の位置	数値
23	INRSTR	後ろからの文字列サーチ	○	文字列の位置	数値
24	INiRSTR	後ろからの大文字、小文字を区別しない文字列サーチ	○	文字列の位置	数値
25	INLIKE	文字列のLIKE一致ヶ所数と位置を返す。	○	一致ヶ所数	数値
26	XHASH	ハッシュ処理		ハッシュ表へのポインタ、または、ハッシュ・インデックス	数値
27	SETARRAY	配列に値を設定する		設定数	数値

（注）OP欄の"○"は演算子でも使用可を示す。

（注1）変換数値の属性。（注2）第一項のデータ要素の属性。

（注3）データ要素の属性

表8.1-1 関数一覧（演算関連 2/2）

No	関数名	機能	OP	戻り値	属性
28	ARRAYCPY	配列をコピーする		コピー数	数値
29	ARRAYCLR	配列をクリアする		クリア数	数値
30	ARRAYCMP	配列どうしを比較する		比較結果	数値
31	ARRAYBXP	配列どうしを演算する		演算数	数値
32	COUNTV	配列またはデータ・リストの要素数		定義済み要素数	数値
33	INDEXA	配列の1次元要素位置を求める		1次元要素位置	数値
34	FL	データ・リストを操作する		データ・リスト	注3
35	LIST	データ・リストを作成する		データ・リスト	注3
36	FIRST	データ・リストの最初の要素を参照する		データ・リスト	注3
37	REST	データ・リストの最初の要素を除くリスト参照する		データ・リスト	注3
38	CONS	データ・リストを連結する		データ・リスト	注3
39	LIST_REF	データ・リストの指定番目の要素を参照する。		データ・リスト	注3
40	SORT	配列等のデータをソートする		データ個数	数値
41	ROUND	数値を丸める		丸め結果	注3
42	NEW	インスタンスを生成する		インスタンス	
43	ARRAYMAP	指定インデックス位置にマップする配列を作成する。		配列	
44	CCHAR、 CSTR[ING]	文字型に変換する		変換データ	注3
45	CINT、CBIN	整数型に変換する		変換データ	注3
46	CFLOAT, CFLT, CDOUBLE, CDBL	倍精度2進浮動小数点型に変換する		変換データ	注3
47	CDEC[IMAL]	10進小数点型に変換する		変換データ	注3
48	CBULK	BULK型に変換する		変換データ	注3
49	CDATE	日付型に変換する		変換データ	日付
50	CVARIANT	バリエーション型に変換する		変換データ	
51	SKIP_OPT	文字列をスキップする	○	文字列の位置	数値
52	DECODE	特定の値を別の値に変換する		変換データ	注3

(注) OP欄の"○"は演算子でも使用可を示す。

(注1) 変換数値の属性。(注2) 第一項のデータ要素の属性。

(注3) データ要素の属性

表8.1-1 関数一覧（ファイル関連）

No	関数名	機能	戻り値	属性
1	FOPEN	ファイルをオープンする	ファイル・ポインタ	数値
2	FCLOSE	ファイルをクローズする	リターン・コード	数値
3	FGETLINE	ファイルから1行読み込む	1行文字列	文字
4	FPUTLINE	ファイルに複数の文字列を書き込む	文字列長の合計	数値
5	FELREAD1	ファイルから1データ要素を読み込む	データ要素	注2
6	FELWRITE	ファイルに複数のデータ要素を書き込む	出力データ長の合計	数値
7	FELWRITE1	ファイルに1データ要素を書き込む	出力データ長	数値
8	UNLINK	ファイルを削除する	リターン・コード	数値
9	GETLINE	標準入力から1行読み込む	1行文字列	文字
10	PUTLINE	標準出力に複数の文字列を書き込む	文字列長の合計	数値
11	ELREAD1	標準入力から1データ要素を読み込む	データ要素	注2
12	ELWRITE	標準出力に複数のデータ要素を書き込む	出力データ長の合計	数値
13	ELWRITE1	標準出力に1データ要素を書き込む	出力データ長	数値
14	OPENDIR	ディレクトリをオープンする	ディレクトリ・ポインタ	数値
15	READDIR	ディレクトリ内を読む	ファイル名	文字
16	CLOSEDIR	ディレクトリをクローズする	リターン・コード	数値
17	STAT	ファイル属性を取得する	リターン・コード	数値
18	FPSTAT	ファイル属性を取得する	リターン・コード	数値
19	LPRINT	LPRINTコマンドと同様	リターン・コード	数値
20	PRINT	PRINTコマンドと同様	リターン・コード	数値
21	ECHO	ECHOコマンドと同様	リターン・コード	数値
22	SAY	ECHOと同じ	リターン・コード	数値

表8.1-1 関数一覧（シェル実行関連）

No	関数名	機能	戻り値	属性
1	SHELL	システムのコマンド・ラインを実行する	リターン・コード	数値
2	POPEN	パイプをオープンしてコマンド・ラインを実行する	ファイル・ポインタ	数値
3	PCLOSE	パイプをクローズする	リターン・コード	数値

表8.1-1 関数一覧（ログ関連）

No	関数名	機能	戻り値	属性
1	SYSLOG	syslogにログを出力する	リターン・コード	数値
2	LOGOUT	ログを出力する	リターン・コード	数値
3	GETLOGPARM	ログパラメータを取得する	取得項目数	数値
4	SETLOGPARM	ログパラメータを設定する	設定項目数	数値
5	RESLOGPARM	取得済ログパラメータをリストアする	設定項目数	数値

表8.1-1 関数一覧（メモリ管理関連）

No	関数名	機能	戻り値	属性
1	NOFREE	未開放メモリのアドレスをファイルに出力する	未開放メモリ数	数値
2	GETMEMUSED	各種のメモリ使用量を取得する	メモリ使用量	数値

表8.1-1 関数一覧（実行制御関連）

No	関数名	機能	戻り値	属性
1	SHUTCTL	s h u tを制御する	リターン・コード	数値
2	EXIT	スクリプトの実行を終了する	リターン・コード	数値
3	SETENV	環境変数の値を設定する	リターン・コード	数値
4	UNSETENV	環境変数を削除する	リターン・コード	数値
5	PUTENV	環境変数の値を設定する (name=value形式)	リターン・コード	数値
6	GETENV	環境変数の値を取得する	環境変数の値	文字
7	GETTIME	セッション開始からの経過時間を返す	経過時間(ミリ秒まで)	数値
8	TIMES	関数を指定回数繰り返す	関数の戻り値	
9	GETVAL	各種設定値の値を取得する	取得した値	

表8.1-1 関数一覧（通信関連）

No	関数名	機能	戻り値	属性
1	CHANNEL	チャンネルを制御する	リターン・コード	数値

表8.1-1 関数一覧（数値計算関連）

No	関数名	機能	戻り値	属性
1	S Q R T	平方根	関数値	2進浮動小数
2	S I N	$\sin(x)$	関数値	2進浮動小数
3	C O S	$\cos(x)$	関数値	2進浮動小数
4	T A N	$\tan(x)$	関数値	2進浮動小数
5	A T A N	$\text{atan}(x)$	関数値	2進浮動小数
6	L O G	自然対数	関数値	2進浮動小数
7	L O G 1 0	10の対数	関数値	2進浮動小数
8	E X P	eのx乗	関数値	2進浮動小数
9	P O W E R	xのy乗	関数値	2進浮動小数
10	S R A N D 1 S R A N D 4 8	乱数のSEED(種)を設定する	NULL値	文字
11	R A N D 1 D R A N D 4 8	0.0<=、<1.0の間の乱数を求める	関数値	2進浮動小数
12	C B R T	立方根	関数値	2進浮動小数
13	S I N H	$\sinh(x)$	関数値	2進浮動小数
14	C O S H	$\cosh(x)$	関数値	2進浮動小数
15	T A N H	$\tanh(x)$	関数値	2進浮動小数
16	A T A N H	$\text{atanh}(x)$	関数値	2進浮動小数
17	C E I L	$\text{ceil}(x)$	関数値	2進浮動小数
18	F L O O R	$\text{floor}(x)$	関数値	2進浮動小数
19	R I N T	$\text{rint}(x)$	関数値	2進浮動小数

表8.1-10 関数一覧（日付関連）

No	関数名	機能	戻り値	属性
1	A D D _ M O N T H S	月数加算、減算	演算日付	日付
2	D A T E _ _ A D D	時間間隔指定の時間加算、減算	演算日付	日付
3	D A T E _ _ D I F F	時間間隔指定の時間差	時間差	数値
4	L A S T _ _ D A Y	月の最終日を求める	月の最終日付	日付
5	S E T _ _ D A T E _ _ P A R T	時間間隔単位の設定	設定日付	日付
6	T O _ _ C H A R	日付の文字列への変換	変換文字列	文字
7	T O _ _ D A T E	日付への変換	変換日付	日付

8.2. 全般規則

- (1) 関数の引数には、式を指定できる。

8.3. ABS

(1)機能

絶対値を求める。

(2)一般形式

```
result = ABS(var) ;
```

(3)構文規則

なし

(4)一般規則

(A) varが数値でないときは、数値に変換される。

(B) varまたは変換結果が、整数のときは、整数の絶対値を、2進浮動小数点のときは、2進浮動小数点の絶対値を、10進小数点のときは、10進小数点の絶対値を求める。

(C) varが日付のときは、そのまま返す。

8.4. ADD_MONTHS

(1)機能

日付に月数を加算、減算した結果を求める。

(2)一般形式

```
DATE result = ADD_MONTHS(var, months) ;
```

(3)構文規則

なし

(4)一般規則

(A) varが日付でないときは、以下のように日付に変換される。

(a) 文字属性のときは、入力値により以下の形式と見なす。未入力部分は、時間間隔(Y, M, D, H, N, S)の最初の値を補う。'/'には、数字および時間間隔文字以外の半角文字を指定可能。数字または時間間隔文字のときはエラー。時間間隔の値が範囲外のときは、エラー。

4バイトのとき、"YYYY"。

5バイトのとき、"YY/MM"。YYは、>=51のとき19YY，<=50のとき20YYと見なす。

6バイトのとき、"YYMMDD"。

7バイトのとき、"YYYY/MM"。

8バイトのとき、"YYYYMMDD"または"YY/MM/DD"。

10バイトのとき、"YYYY/MM/DD"。

その他のとき、"YYYY/MM/DD HH:MI:SS"

(b) その他のときは、整数に変換し、1900/01/01 00:00:00からの秒数として変換する。

(B) monthsが整数でないときは、整数に変換され、月数として加算または減算される。

8.5. ARRAYCPY

(1) 機能

配列をコピーする。

(2) 一般形式

```
n = ARRAYCPY(array_to, [start_to], array_from, [start_from], [num]) ;
```

(3) 構文規則

(A) array_to または array_from には、配列名または数値を指定する。

(B) array_to または array_from に文字定数が指定されたときは、数値に変換される。

~~(C) array_to には、連想配列は指定できない。~~

(D) 配列が連想配列のときは、両配列共に連想配列でなければならない。

(E) 配列が連想配列のときは、start は指定できない。

(4) 一般規則

(A) array_to または array_from に配列名を指定したときは、配列名 [start] の要素から最大で、配列のサイズまたは num 個までコピーされる。

(B) array_to または array_from に数値が指定されたときは、内部番号変数の \$(指定数値+start) の要素から、最大で配列のサイズまたは num 個までコピーされる。

(C) コピー元要素が未設定のときは、コピー先要素も未設定となる。

(D) 返却値 n には、コピーされた個数が返る。

(E) 配列が連想配列のときは、定義済みキーのデータがコピーされる。

num は定義済み要素が対象となる。

8.6. ARRAYCLR

(1) 機能

配列をクリアする。

(2) 一般形式

```
n = ARRAYCLR(array, [start], [var], [num]) ;
```

(3) 構文規則

- (A) arrayには、配列名または数値を指定する。
- (B) arrayに文字定数が指定されたときは、数値に変換される。
- (C) 配列が連想配列のときは、start、var、numは指定できない。

(4) 一般規則

- (A) arrayに配列名を指定したときは、配列名[start]の要素から最大で、配列のサイズまたはnum個までvarが格納される。
- (B) arrayに数値が指定されたときは、内部番号変数の\$(指定数値+start)の要素から、最大で配列のサイズまたはnum個までvarが格納される。
- (C) 返却値nには、実際に設定された個数が返る。
- (D) 配列が連想配列のときは、要素数がゼロとなる
- (E) デフォルト値は、start=0, var='', num=配列の最大サイズ。

8.7. ARRAYCMP

(1) 機能

配列どうしを比較する。

(2) 一般形式

```
n = ARRAYCMP(array1, [start1], array2, [start2], [num], [cmp]) ;
```

(3) 構文規則

- (A) array1またはarray2には、配列名または数値を指定する。
- (B) array1またはarray2に文字定数が指定されたときは、数値に変換される。
- (C) 配列が連想配列のときは、両方が連想配列でなければならない。このときは、start1とstart2は指定できない。

(4) 一般規則

- (A) array1またはarray2に配列名を指定したときは、配列名[start]の要素から最大で、配列のサイズまたはnum個まで比較される。
- (B) array1またはarray2に数値が指定されたときは、内部番号変数の\$(指定数値+start)の要素から、最大で配列のサイズまたはnum個まで比較される。
- (C) 比較は要素単位に行われ、偽となった時点で終了する。
- (D) 返却値nには、真の要素数が返る。
- (E) cmpには、比較演算子を文字属性で指定する。省略時は、'='で比較する。
- (F) 配列が連想配列のときは、同じキーどうしを比較する。numは同じキーどうしが対象となる。

8.8. ARRAYBXP

(1) 機能

配列どうしを演算する。array_to = array1 演算子 array2

(2) 一般形式

```
n = ARRAYBXP(array_to, [start_to], array1, [start1], array2, [start2], [num], [bxp]) ;
```

(3) 構文規則

- (A) array_to、array1、array2には、配列名または数値を指定する。
- (B) array_to、array1、array2に文字定数が指定されたときは、数値に変換される。
- ~~(C) array_toには、連想配列は指定できない。~~
- (D) 配列が連想配列のときは、3つの配列共に連想配列でなければならない。
- (E) 配列が連想配列のときは、startは指定できない。

(4) 一般規則

- (A) array_to、array1、array2に配列名を指定したときは、配列名[start]の要素から最大で、配列のサイズまたはnum個まで比較される。
- (B) array_to、array1、array2に数値が指定されたときは、内部番号変数の\$(指定数値+start)の要素から、最大で配列のサイズまたはnum個まで比較される。
- (C) 返却値nには、演算した要素数が返る。
- (D) bxpには、演算子を文字属性で指定する。省略時は、'+'が指定されたものと見なす。
- (E) 配列が連想配列のときは、同じキーどうしを演算する。numは同じキーどうしが対象となる。

8.9. ARRAYMAP

(1) 機能

指定インデックス位置を先頭とするマップド配列を返す。

(2) 一般形式

```
array_m = ARRAYMAP(array, [i], [j], [k]) ;
```

(3) 構文規則

- (A) arrayには、配列名を指定する。
- (B) i, j, kには、結果が数値となる式を指定する。

(4) 一般規則

- (A) i, j, kには、先頭としたい、各次元の位置を指定する。
- (B) 各次元の大きさ、または、範囲の上限は、指定位置からの相対的大きさに調整される。
なお、下限は変わらない。

8.10. ASC

(1) 機能

文字列の1文字目をアスキーコードに変換する

(2) 一般形式

```
result = ASC(var) ;
```

(3) 構文規則

(4) 一般規則

(A) 文字は4バイト以下とする。文字コードを4バイト整数で表した値が返される。

(B) データが文字型以外の場合は、文字型に変換された後でアスキーコードに変換される。

8.11. ATAN

(1) 機能

atanを求める。

(2) 一般形式

```
ret = ATAN(x) ;
```

(3) 構文規則

なし。

(4) 一般規則

(A) xが浮動小数でないときは、浮動小数に変換される。

(B) リターン値はラジアンである。

8.12. ATANH

(1) 機能

atanhを求める。

(2) 一般形式

```
ret = ATANH(x) ;
```

(3) 構文規則

なし。

(4) 一般規則

(A) xが浮動小数でないときは、浮動小数に変換される。

(B) リターン値はラジアンである。

8.13. CBIN

- (1)機能
整数型に変換する。
- (2)一般形式
 $\text{ret} = \text{CBIN}(x)$;
- (3)構文規則
なし。
- (4)一般規則
なし。

8.14. CBRT

- (1)機能
立方根を求める。
- (2)一般形式
 $\text{ret} = \text{CBRT}(x)$;
- (3)構文規則
なし。
- (4)一般規則
(A)xが浮動小数でないときは、浮動小数に変換される。

8.15. CBULK

- (1)機能
BULK型に変換する。
- (2)一般形式
 $\text{ret} = \text{CBULK}(x [, \text{size}])$;
- (3)構文規則
なし。
- (4)一般規則
(A)sizeを指定すると固定長に、省略すると可変長となる。

8.16. CCHAR

- (1)機能
文字型に変換する。
- (2)一般形式
 $\text{ret} = \text{CCHAR}(x [, \text{size}])$;
- (3)構文規則
なし。
- (4)一般規則
(A)sizeを指定すると固定長に、省略すると可変長となる。

8.17. CDATE

- (1)機能
日付型に変換する。
- (2)一般形式
`DATE result = CDATE(var [, format]) ;`
- (3)構文規則
なし。
- (4)一般規則
(A)TO_DATE()と同じ。

8.18. CDEC

- (1)機能
10進小数点型に変換する。
- (2)一般形式
`ret = CDEC(x [, size [, scale]]) ;`
- (3)構文規則
なし。
- (4)一般規則
(A)sizeは精度を、scaleは位取りを示す。詳細は、変数のデータ型を参照。

8.19. CDOUBLE (CDBL)

- (1)機能
倍精度2進浮動小数点型に変換する。
- (2)一般形式
`ret = CDOUBLE(x) ;`
- (3)構文規則
なし。
- (4)一般規則
なし。

8.20. CEIL

- (1)機能
指定した倍精度2進浮動小数点数以上の数のうち、最小の整数値を倍精度2進浮動小数点数で返す。
- (2)一般形式
`ret = CEIL(x) ;`
- (3)構文規則
なし。
- (4)一般規則
(A)xが浮動小数でないときは、浮動小数に変換される。

8.21. CFLOAT (CFLT)

- (1)機能
倍精度 2 進浮動小数点型に変換する。
- (2)一般形式
ret = CFLOAT(x) ;
- (3)構文規則
なし。
- (4)一般規則
なし。

8.22. CHANNEL

- (1)機能
チャンネルを制御する
- (2)一般形式

cha = CHANNEL('open', Host, Service, HeadLen, Options, HeadCheck, Exception) ;

ret = CHANNEL('shut', cha, option) ;

ret = CHANNEL('resume', cha, option) ;

ret = CHANNEL('close', cha) ;
- (3)構文規則
- (4)一般規則
(A)

8.23. CHR

(1)機能

コードを文字型に変換する。

(2)一般形式

```
result = CHR(var1[, var2, ...]) ;
```

(3)構文規則

(4)一般規則

(A)コードは、整数値を0xFFでマスクした値が使われる。

(B)データが整数値以外の場合は、整数値に変換された後で文字型に変換される。

(C)指定されたコードは、全てが1バイト文字として連結される。

8.24. CINT

(1)機能

整数型に変換する。

(2)一般形式

```
ret = CINT(x) ;
```

(3)構文規則

なし。

(4)一般規則

なし。

8.25. CLONG (CLNG)

(1)機能

整数型に変換する。

(2)一般形式

```
ret = CLONG(x) ;
```

(3)構文規則

なし。

(4)一般規則

なし。

8.26. CLOSEDIR

(1) 機能

ディレクトリをクローズする。

(2) 一般形式

```
ret = CLOSEDIR(dp) ;
```

(3) 構文規則

(4) 一般規則

(A)ディレクトリ以外は、クローズできない。

8.27. CONCAT

(1) 機能

文字列の連結

(2) 一般形式

```
result = CONCAT(var, [v1], [v2], ... ) ;
```

(3) 構文規則

文字列演算式の第一項以降がそれぞれ第1パラメータ以降に対応する。

(4) 一般規則

文字列演算子を参照。

8.28. CONDAS

(1) 機能

文字列の条件式

(2) 一般形式

```
result = CONDAS(var [, start [, len]], condas) ;
```

(3) 構文規則

(A)文字列演算式の第一項と第二項がそれぞれ第1パラメータと第2パラメータに対応する。

(B)start、lenについては、INSTR()と同じ。

(4) 一般規則

文字列演算子を参照。

8.29. CONS

- (1)機能
データ・リストまたは要素を連結してリストを作る。
- (2)一般形式
$$\text{var} = \text{CONS}([\text{var1}], [\text{var2}], \dots) ;$$
- (3)構文規則
FLと同じ。
- (4)一般規則
FLと同じ。

8.30. COS

- (1)機能
cosを求める。
- (2)一般形式
$$\text{ret} = \text{COS}(x) ;$$
- (3)構文規則
なし。
- (4)一般規則
(A)xはラジアンで指定する。
(B)xが浮動小数でないときは、浮動小数に変換される。

8.31. COSH

- (1)機能
coshを求める。
- (2)一般形式
$$\text{ret} = \text{COSH}(x) ;$$
- (3)構文規則
なし。
- (4)一般規則
(A)xはラジアンで指定する。
(B)xが浮動小数でないときは、浮動小数に変換される。

8.32. COUNTV

- (1)機能
スカラー、配列またはデータ・リストの定義済み要素数を数える。
- (2)一般形式
$$n = \text{COUNTV}(\text{var}) ;$$
- (3)構文規則
なし
- (4)一般規則
(A)オプション番号1の影響は受けない。
(B)スカラー変数が未定義のときは、エラーとなる。定義済みのときは、要素数は、1となる。

8.33. CSTRING (CSTR)

(1)機能

文字型に変換する。

(2)一般形式

```
ret = CSTRING(x [, size]) ;
```

(3)構文規則

なし。

(4)一般規則

(A) sizeを指定すると固定長に、省略すると可変長となる。

8.34. DATE__ADD

(1)機能

時間間隔を指定して時間を加算または減算する。

(2)一般形式

```
DATE result = DATE_ADD(時間間隔文字列, 時間, 日付) ;
```

(3)構文規則

なし。

(4)一般規則

(A)時間間隔文字列には、以下で始まる文字列または数値で指定する。その他はエラー。

YYYY, YEAR or 1 : 年数

M or 2 : 月数

D, Y, W or 3 : 日数

H or 4 : 時間

N, MI or 5 : 分

S or 6 : 秒

Q or 7 : 4半期(3ヶ月)

WW or 8 : 週(7日)

(B)時間には、時間間隔文字列に対応する時間を正または負で指定する。

数値でないときは、数値に変換される。

(C)日付が日付型でないときは、日付に変換される。変換仕様は、ADD_MONTHS()と同じ。

8.35. DATE__DIFF

(1)機能

時間間隔を指定して時間差を求める。

(2)一般形式

```
result = DATE_DIFF(時間間隔文字列, 日付1, 日付2) ;
```

(3)構文規則

なし。

(4)一般規則

(A)時間間隔文字列は、DATE_ADD()と同じ。

(B)時間間隔文字列に対応した(日付1 - 日付2)の時間差を求める。

(C)日付1、日付2が日付型でないときは、日付に変換される。変換仕様は、ADD_MONTHS()と同じ。

8.36. DECODE

(1) 機能

条件制御によって特定の値および値域を別の値に変換する。

(2) 一般形式

```
ret = DECODE(x, s1, r1 [, s2, r2, ...] [, default]) ;
```

(3) 構文規則

なし。

(4) 一般規則

- (A) xがsiに一致するとき、riをそのまま返す。どれとも一致しないときは、defaultをそのまま返す。defaultが省略されているときは、NULL値を返す。
- (B) xとsiの型は、一致していなければならない。

8.37. EDIT

(1) 機能

変数の値を指定したフォーマットへ変換する。

(2) 一般形式

```
result = EDIT(format [, var1, var2, var3 ...]) ;
```

(3) 構文規則

(A)

(4) 一般規則

- (A) formatは、C言語のprintfでの指定と同様(\$, *, #, lは不可)。
- (B) 変換子に対応する変数がないときは、"(N/A)"を出力する。
- (C) 変換できないときは、"(ERR)"を出力する。
- (D) データの編集指定子による編集規則を以下に示す。

	文字	整数	浮動小数点数	BULK
s	文字列	文字列に変換	文字列に変換	文字列に変換
c	先頭1バイト	最下位1バイト	整数に変換し、 最下位1バイト	先頭1バイト
d, i, o, u, x	整数に変換	数値	整数に変換	アドレス
f, e, g, a	浮動小数点数に変換	浮動小数点数に変換	数値	(ERR)
その他	アドレス	アドレス	アドレス	アドレス

(注) 編集指定子は小文字でも同様。

	目付
s	文字列 (YYYY/MM/DD H H24:MI:SS)
c	上記先頭1バイト
d, i, o, u, x	(ERR)
f, e, g, a	(ERR)
その他	アドレス

8.38. ELREAD1

(1)機能

標準入力から1データ要素を読み込む。

(2)一般形式

```
var = ELREAD1(attr, [size], [scale]) ;
```

以下、FELREAD1と同じ。

8.39. ELWRITE

(1)機能

標準出力に複数個のデータ要素を書き込む。

(2)一般形式

```
ret = ELWRITE(v1, v2, ...) ;
```

以下、FELWRITEと同じ。

8.40. ELWRITE1

(1)機能

標準出力に1データ要素を書き込む。

(2)一般形式

```
ret = ELWRITE1(fp, v1, [size], [scale]) ;
```

以下、FELWRITE1と同じ。

8.41. EVAL

(1)機能

文字列で与えられた式を評価し結果を返却する。

(2)一般形式

```
result = EVAL(var[, opt]) ;
```

(3)構文規則

(4)一般規則

(A)varの内容が空のときはエラー。

(B)optは、varの実行でエラーが発生したときの処理を指定する。

=0 または省略: エラーメッセージを出力する。本ブロックからエラーリターンする。

=1: エラーメッセージを出力する。エラーコードをERRORに設定し、正常終了する。NULL値を返却する。

=2: エラーメッセージを出力しない。他は、1と同じ。

8.42. EXIT

(1) 機能

スクリプト実行を中断し、セッションを終了する。

(2) 一般形式

```
ret = exit([error_code]) ;
```

(3) 構文規則

(4) 一般規則

(A) error_codeが指定されているときは、その値がセッションのリターン値となり、その値を返却する。

省略されているときは、ゼロを返却する。

(B) 本関数を含む文の実行が終了した時点で、セッションが終了する。

(C) その他は、LET EXITと同じ。

8.43. EXP

(1) 機能

eのx乗を求める。

(2) 一般形式

```
ret = EXP(x) ;
```

(3) 構文規則

なし。

(4) 一般規則

(A) xが浮動小数でないときは、浮動小数に変換される。

8.44. FCLOSE

(1) 機能

ファイルをクローズする。

(2) 一般形式

```
ret = FCLOSE(fp) ;
```

(3) 構文規則

(4) 一般規則

(A) 標準入力、標準出力、標準エラー出力は、クローズできない。

8.45. FELREAD1

(1)機能

ファイルから1データ要素を読み込む

(2)一般形式

```
var = FELREAD1(fp, attr, [size], [scale]) ;
```

(3)構文規則

(A) attrは、以下の値を指定する。

文字列	: 1
整数	: 2
2進浮動小数点数	: 3
10進浮動小数点数	: 4 (ファイル上のデータは、パック10進と見なす)
バルク	: 5
日付	: 6

(4)一般規則

(A) sizeは、読み取るファイル上の領域の長さ(バイト)を示す。

ただし、10進浮動小数点数の場合は、(size/2+1)バイトのパック10進として読み込む。

sizeがファイルのサイズより大きいときは、ファイルのサイズ分読み込む。

size省略時またはゼロ以下のときは、以下が指定されたと見なす。

- ・文字列 : 1
- ・整数 : 4
- ・2進浮動小数点数 : 8
- ・10進浮動小数点数: 15
- ・バルク : 1
- ・日付 : 日付データ長 内容は、付録9.3を参照

2進浮動小数点数において、sizeが4未満のときは、エラーとなる。

(B) scale省略時は、ゼロと見なす。

10進浮動小数点数のときは、

少数点の右側(正のとき)または左側(負のとき)の桁数

その他のときは、

scaleは、その大きさをデータとして実際に読み込む長さ示し、正負で位置を示す。

scaleが正またはゼロのときは、以下を示し、負のときは、その逆となる。

- ・文字列 : 前詰(左詰)
- ・整数 : 後詰(右詰)
- ・2進浮動小数点数 : 後詰(右詰)
- ・バルク : 前詰(左詰)
- ・日付 : 前詰(左詰)

scaleがゼロのときは、sizeが設定される。

size < |scale| のときは、scaleには、sizeの値が設定される。

scaleの大きさにより、データとして実際に読み込む長さおよびデータの型は以下である。

No.	attr	scale の値	データ長	データ型	備考
1	文字列	—	scaleバイト	文字列	
2	整数	1	1	char	読み込み後intに変換
		2 ~ 3	2	short	同上
		4 ~ 7	4	int	同上
		8 以上	8	int64	同上
3	2進浮動小数点数	1 ~ 7	4	float	読み込み後doubleに変換
		8 以上	8	double	
4	10進浮動小数点数	少数点桁数	size/2 + 1	パック10進	
5	バルク	—	scaleバイト	バルク	
6	日付	—	scaleバイト	日付	

8.46. FELWRITE

(1) 機能

ファイルに複数個のデータ要素を書き込む。

(2) 一般形式

```
ret = FELWRITE(fp, v1, v2, ...);
```

(3) 構文規則

(4) 一般規則

(A) 変数または定数のデータ長で出力する。

- ・ 文字 : データ長
- ・ 整数 : 4 バイト
- ・ 2進浮動小数点数 : 8 バイト
- ・ 10進浮動小数点数: 有効桁数(ファイル上のデータは、(有効桁数/2+1)バイトのパック10進)
- ・ バルク : データ長
- ・ 日付 : 日付データ長 内容は、付録9.3を参照

(B) retには、出力データ長の合計が返る。

8.47. FELWRITE1

(1)機能

ファイルに1データ要素を書き込む。

(2)一般形式

```
ret = FELWRITE1(fp, v1, [size], [scale]) ;
```

(3)構文規則

(4)一般規則

(A) sizeは、書き込むファイル上の領域の長さ(バイト)を示す。

ただし、10進浮動小数点数の場合は、(size/2+1)バイトのパック10進として書き込む。

size省略時またはゼロ以下のときは、FELREAD1()と同じ。

2進浮動小数点数において、sizeが4未満のときは、エラーとなる。

10進浮動小数点数において、sizeが有効桁数より小さいときは、エラーとなる。

(B) scaleは、以下となる。

10進浮動小数点数のときは、

省略時は、データのまゝ。

設定されたときは、少数点の右側(正のとき)または左側(負のとき)の桁数。

その他のときは、

scaleは、その大きさをデータとして実際に書き込む長さ示し、正負で位置を示す。

scale省略時またはゼロのときは、以下と見なす。

- 文字 : size
- 整数 : size (size>0), 4 バイト(size<=0)
- 2進浮動小数点数 : size (size>0), 8 バイト(size<=0)
- バルク : size
- 日付 : size

size > |scale| のときは、余った部分には、文字ならスペース、その他はゼロが埋められる。

size < |scale| のときは、scaleには、sizeの値が設定される。

文字、バルク、日付において、|scale|>データ長のときは、|scale|バイト内で前詰めされる。

余った部分には、文字はスペース、バルク、日付はゼロが埋められる。

データとして実際に書き込む長さおよび位置は、FELREAD1()と同じ。

8.48. FF

(1) 機能

関数名と同じ文字列、または、関数定義文字列を関数指定の扱いにする。

(2) 一般形式

FF(var) [(本関数の戻り値となった関数名の関数引数)]

(3) 構文規則

(A) 関数の直後に、関数引数を与える' ('がなければならない。

(4) 一般規則

(A) 関数の戻り値は、関数の関数名にのみ使用できる。

(5) 例

```
$x = FF('ABS');  
print $x(-1);
```

8.49. FGETLINE

(1) 機能

ファイルから1行読み込む。

(2) 一般形式

```
line = FGETLINE(fp, [size], [opt]) ;
```

(3) 構文規則

(4) 一般規則

(A) 改行コードまでを1行として読み込む。'¥r', '¥r¥n', '¥n'を改行コードと見なす。

(B) sizeは、1行の有効長を指定する。2以上をバイト単位で指定する。

1行の最初のsize-1バイトが読み込まれ、それより後ろは捨てられる。

デフォルトは、2048。

(C) optは、

0x01 : ONのとき、'¥r', '¥r¥n'は'¥n'に変換される。

0x02 : ONのとき、行末の改行コードを削除する。

0x04 : ONのとき、CSV形式で読み込む(2重引用符中の改行コード以降も読み込む)。

0x08 : ONのとき、CSV形式で読み込み、2重引用符中の改行コードを削除する。

デフォルトは、0x02 + 0x04

8.50. F I R S T

(1)機能

データ・リストの最初の要素を参照する。

(2)一般形式

```
var = FIRST(var1) ;
```

(3)構文規則

F Lと同じ。

(4)一般規則

F Lと同じ。

8.51. F L

(1)機能

データ・リストについて、以下の操作を行なう。

(A)LISTリストを作成する

(B)FIRSTリストの最初の要素を参照する。

(C)RESTリストの最初のvar2個の要素を除くリストを参照する。(var2>=0)
var2省略時は、var2=1と同じ。

(D)CONSリストまたは要素を連結して1つのリストを作る。

(E)LIST_REFリストのvar2番目の要素を参照する。(先頭は0番目)
var2省略時は、var2=0と同じ。

(2)一般形式

```
var = FL('LIST', [var1], [var2], ...);
```

```
var = FL('FIRST', var1);
```

```
var = FL('REST', var1 [, var2]);
```

```
var = FL('CONS', [var1], [var2], ...);
```

```
var = FL('LIST_REF', var1 [, var2]);
```

(3)構文規則

(A)第一パラメータには、機能に示す操作指示文字列を指定する。大文字と小文字は区別されない。

(4)一般規則

(A)全般

①パラメータが省略されたときは、NULL値が追加される。

②データ要素がゼロのときの扱いは、オプション番号5によって異なる。

(B)LIST

①パラメータに指定された変数がデータ・リストのときは、そのデータ・リストが作成されるリストにそのまま追加される。

8.52. FLOOR

(1) 機能

指定した倍精度 2 進浮動小数点数以下の数のうち、最大の整数を倍精度 2 進浮動小数点数で返す。

(2) 一般形式

```
ret = FLOOR(x) ;
```

(3) 構文規則

なし。

(4) 一般規則

(A) xが浮動小数でないときは、浮動小数に変換される。

8.53. FOPEN

(1) 機能

ファイルをオープンする

(2) 一般形式

```
fp = FOPEN(fname [, mode]) ;
```

(3) 構文規則

(4) 一般規則

(A) mode省略時は、'r'。

8.54. [FP]STAT

(1)機能

ファイルの属性を取得する。

(2)一般形式

```
ret = STAT(file [, map_index [, maxval] ] ) ;
ret = FPSTAT(fp [, map_index [, maxval] ] ) ;
```

(3)構文規則

- (A)map_indexには、配列名または数値を指定する。
- (B)map_indexに文字定数が指定されたときは、数値に変換される。

(4)一般規則

- (A)属性値は、以下の順で指定された配列に返却される。

```
st_val[ 0 ] = f_type;
                =0 : それ以外
                =1 : 名前付きパイプ
                =2 : キャラクタ・デバイス
                =3 : ディレクトリ
                =4 : ブロック・デバイス
                =5 : レギュラ・ファイル
                =6 : シンボリック・リンク

st_val[ 1 ] = tStat.st_dev;
st_val[ 2 ] = tStat.st_ino;
st_val[ 3 ] = tStat.st_mode;
st_val[ 4 ] = tStat.st_nlink;
st_val[ 5 ] = tStat.st_uid;
st_val[ 6 ] = tStat.st_gid;
st_val[ 7 ] = tStat.st_rdev;
st_val[ 8 ] = tStat.st_size;          /* 属性は、DECIMAL */
st_val[ 9 ] = tStat.st_atim.tv_sec;
st_val[10] = tStat.st_atim.tv_nsec;
st_val[11] = tStat.st_mtim.tv_sec;
st_val[12] = tStat.st_mtim.tv_nsec;
st_val[13] = tStat.st_ctim.tv_sec;
st_val[14] = tStat.st_ctim.tv_nsec;
st_val[15] = tStat.st_blksize;
st_val[16] = tStat.st_blocks;
```

- (B)map_indexに配列名を指定したときは、属性値は、配列の先頭から最大で、配列のサイズまで格納される。

maxvalの値が、配列のサイズより大きいときは、配列のサイズまで格納される。

- (C)map_indexに数値が指定されたときは、内部番号変数の当該位置から、最大でmaxval個格納される。

maxvalが、内部番号変数の最後の変数を越えるような場合は、エラーとなる。

- (D)map_indexには、連想配列名は指定できない。

8.55. FPUTLINE

(1) 機能

ファイルに複数個の文字列を書き込む。

(2) 一般形式

```
ret = FPUTLINE(fp, v1, v2, ...);
```

(3) 構文規則

(4) 一般規則

(A) 変数または定数のデータ長で出力する。

1つの引数データの出力形式は、以下を除き、SAYコマンドと同じ。

(a) BULKデータは、そのデータ長まで出力する。

(B) 末尾には、改行コード('¥n')を出力する。これは、オプション3で変更できる。

(C) retには、出力データ長の合計が返る。

8.56. GETARGS

(1) 機能

文字列をカンマまたは空白文字または指定文字で分割する。分割の最大値は、128。

(2) 一般形式

```
n = GETARGS(line, map_index, [max_args], [opt], [sep_char]) ;
```

(3) 構文規則

- (A) map_indexには、配列名または数値を指定する。
- (B) map_indexに文字定数が指定されたときは、数値に変換される。
- (C) max_argsのデフォルト値は、128。

(4) 一般規則

- (A) 区切り文字は、空白文字（スペースまたはタブ）またはカンマ。引用符内のときは、区切りとならない。区切られた文字列の前後の空白文字は削除される。
- (B) map_indexに配列名を指定したときは、アークギュメントは、配列の先頭から最大で、配列のサイズまで格納される。
max_argsの値が、配列のサイズより大きいときは、配列のサイズまで格納される。
- (C) map_indexに数値が指定されたときは、内部番号変数の当該位置から、最大でmax_args個格納される。
max_argsが、内部番号変数の最後の変数を越えるような場合は、エラーとなる。
- (D) sep_charが指定されたときは、sep_charで文字列を分割する。
これが、カンマ(,)であり、optが未指定のときは、自動でCSV形式の処理になる。
- (E) max_argsが指定され分割数がこれに満たない場合は、残りには、null値が設定される。
- (F) map_indexには、連想配列名は指定できない。
- (G) optは、以下の通り。

No.	opt	機能	備考
1	0x01	'#'以降を無視する	デフォルト
2	0x02	2ワード目が、'='のときは無視する 1ワード目のときは、NULL値とみなす	デフォルト
3	0x04	','も区切りとする。 ','が現れた時点で0x12=ONの指定を無効にする	デフォルト
4	0x10	': or' :='もワード区切りとする 1ワード目のときは、NULL値とみなす	
5	0x20	'['と']'を引用符と同様に扱う(IPV6対応)	
6	0x40	0x12=1のとき、'=' or' :='が1ワード目のときには、1ワード目として設定する	
7	0x80	0x04=1のとき、','が現れた時点で0x12=ONの指定を無効にしない	
8	0x010000	前後の引用符を残す。また、引用符の中の連続する2つの引用符を1つにしない	
9	0x040000	CSV形式でのワード処理を行う	sep_charがカンマ(,)のときは、デフォルト

8.57. GETENV

(1)機能

環境変数の値を取得する。

(2)一般形式

```
ret = getenv(name) ;
```

(3)構文規則

なし。

(4)一般規則

(5)戻り値

環境変数の値。環境変数が未設定のときは、NULL値が返される。

8.58. GETLINE

(1)機能

標準入力から1行読み込む。

(2)一般形式

```
line = GETLINE([size], [opt]) ;
```

以下、FGETLINEと同じ。

8.59. GETLOGPARM

(1)機能

ログパラメータを取得する。

(2)一般形式

```
ret = GETLOGPARM(logno, map_index, [max_args]) ;
```

(3)構文規則

(A)map_indexは、GETARGS()と同じ。

(B)max_argsのデフォルト値は、7。

(4)一般規則

(A)lognoは、番号またはログ名称で指定する。LET LOGPARMコマンドを参照。

(B)map_indexは、GETARGS()と同じ。

(C)取得されるログパラメータは以下の通り。詳細は、LET LOGPARMコマンドを参照。

LOGNO:ログ番号

FLAG:出力フラグ

LEVEL:出力レベル

SIZE_MAX:ローテーション・ファイル・サイズ(Kbyte)

FILE_MAX:ローテーション・ファイル数

OPTION:ローテーション・オプション

FILE:出力ファイル名

(5)戻り値

取得パラメータ数。

8.60. GETMEMUSED

(1) 機能

各種のメモリ使用量を取得する。

(2) 一般形式

```
ret = GETMEMUSED(map_index, [max_args]) ;
```

(3) 構文規則

(A) map_index は、GETARGS() と同じ。

(B) max_args のデフォルト値は、5。

(4) 一般規則

(A) map_index は、GETARGS() と同じ。

(B) 以下のメモリ使用量を取得する。

- 配列要素の 1 番目：当該スクリプトのソースおよびコンパイル情報。
- 配列要素の 2 番目：当該セッションで使用している定数情報。
- 配列要素の 3 番目：当該スクリプト実行で使用している定数情報。
- 配列要素の 4 番目：現在行の実行で使用しているワーク情報。
- 配列要素の 5 番目：当該セッション実行中に保存している情報。
スクリプト、手続き、関数実行中のみ保存している情報を含む。

(5) 戻り値

取得したメモリ使用量の合計。

8.61. GETTIME

(1) 機能

セッション開始またはシステム開始からの経過時間を秒単位ミリ秒の精度で返す。

(2) 一般形式

```
ret = gettime([opt]) ;
```

(3) 構文規則

なし。

(4) 一般規則

(A) opt

省略または 0：セッション開始から

0以外の数値：システム開始から

(5) 戻り値

10進浮動小数点数で返す。

8.62. GETVAL

(1)機能

各種設定値の値を取得する。

(2)一般形式

```
ret = getval(取得名 [, 要素名または番号]) ;
```

(3)構文規則

なし。

(4)一般規則

(A)取得名と要素名または番号は以下。

(a)OPTION: オプションの設定値。

要素番号: オプション番号

(5)戻り値

取得した値。

8.63. GETWORD

(1)機能

指定された区切り文字で区切ってワードを取り出す。区切り文字もワードとして取り出す。

ただし、空白、タブ、改行コード('¥n', '¥r')はスキップする。(opt指定)

また、引用符、2重引用符は、引用符、2重引用符処理を行う(opt指定)。

(2)一般形式

```
len = getword(line, map_index, [sep], [opt], [wdlen]) ;
```

(3)構文規則

(A)map_indexは、GETARGS()と同じ。

(4)一般規則

(A)map_indexには、以下が設定される。

配列の1番目: ワードのサーチ開始位置。文字列の先頭を0文字目と数える。

サーチ開始位置は、取り出されたワードの次の位置に自動的に更新される。

(注)初回呼び出し時には、設定しなければならない。

配列の2番目: 取り出したワード。

配列の3番目: ワード種別(ワードが区切り文字のときは、区切り文字に0x80をORしたもの)。

1: 名票、数字等のワード

5: 引用符ではじまる

6: 2重引用符ではじまる

配列の4番目: 2バイト文字有りフラグ。1/0=あり/なし。

配列の5番目: 0x01 不完全[2重]引用符のとき、1。

0x02 ワード長オーバーのとき、1。

配列の6番目: [2重]引用符中に連続する[2重]引用符ありのとき、1

配列の個数が6未満のときは、エラーとなる。

(B) sepには、区切り文字を指定する。デフォルトは、以下。

△¥t, ; () [] {} <> = ' " ! ^ * / (注) △は、半角スペース

(C) optは、以下の通り。

No.	opt	機能	デフォルト
1	0x01	1: 先行空白とタブをスキップする。 0: スキップしない。	1
2	0x02	1: 引用符、2重引用符をはずす。 0: はずさない (不完全エラーあり)。	1
3	0x04	1: 引用符、2重引用符の中の連続する2つのそれらを1つにする。 0: 1つにしない。	1
4	0x08	1: 引用符、2重引用符の処理をしない。 0: 処理をする。引用符、2重引用符もワードの区切りとなる。	0
5	0x10	1: 区切り文字以外が現れるまでスキップする。 0: 左記処理をしない。	0
6	0x20	未使用。	—
7	0x40	1: '¥n' および '¥r' を終端としない。 (引用符内のときのみ有効。空白と同じ扱いする) 0: '¥n' または '¥r' を終端とする。	0
8	0x80	未使用。	—
9	0x0100	1: 引用符の処理をしない。	0
10	0x0200	1: 2重引用符の処理をしない。	0

(D) wrlenには、取り出すワードの最大文字数を指定する。

>0: 指定文字数まで格納する。

=<0または省略: 255文字。

(5) 戻り値

以下を返す。

>=0: 取り出したワードの文字数を返す。

<0: 文字列の終端。

8.64. i I N

(1)機能

大文字、小文字を区別しない複数文字列との比較結果を返す。

(2)一般形式

```
result = iIN(var [, start [, len]], str1 [, str2, ...]) ;
```

(3)構文規則

(A)start、lenについては、IN()と同じ。

(4)一般規則

(A)半角について大文字、小文字を区別しない。その他は、IN()と同じ。

8.65. i L I K E

(1)機能

大文字、小文字を区別しないパターン・マッチング結果を返す。

(2)一般形式

```
result = iLIKE(var [, start [, len]], pat1 [, pat2, pat3, ...]) ;
```

(3)構文規則

(A)start、lenについては、INSTR()と同じ。

(4)一般規則

(A)半角について大文字、小文字を区別しない。その他は、L I K Eと同じ。

8.66. I N

(1)機能

複数文字列との比較結果を返す。

(2)一般形式

```
result = IN(var [, start [, len]], str1 [, str2, ...]) ;
```

(3)構文規則

(A)第2、第3引数(start, len)に数値属性が指定されたときは、比較開始位置、文字数と見なす。

比較開始位置は、比較対象文字列(var)の先頭を1文字目として文字単位で数える。

(B)第2引数(start)に範囲指定が指定されたときは、比較開始位置、終了位置と見なす。

(4)一般規則

(A)複数文字列(str1, str2, ...)のどれかと一致したとき、複数文字列の並び順序番号を返す。

どれも一致しなかったとき、0を返す。

8.67. INDEXA

(1)機能

配列の1次元要素位置を求める。

(2)一般形式

```
index = INDEXA(array, index1, [index2], [index3]) ;
```

(3)構文規則

なし。

(4)一般規則

(A)連想配列でないとき、省略された次元のindexは、ゼロと見なされる。

(B)連想配列のときは、キー値のハッシュインデックスが返される。

8.68. INiRSTR

(1)機能

後ろからの大文字、小文字を区別しない文字列サーチ結果を返す。

(2)一般形式

```
result = INiRSTR(var [, start [, len]], str1 [, str2, ...]) ;
```

(3)構文規則

INSTRと同じ。

(4)一般規則

(A)後ろからサーチする以外は、INISTRと同じ。

8.69. INiSTR

(1)機能

大文字、小文字を区別しない文字列サーチ結果を返す。

(2)一般形式

```
result = INiSTR(var [, start [, len]], str1 [, str2, ...]) ;
```

(3)構文規則

INSTRと同じ。

(4)一般規則

(A)半角について大文字、小文字を区別しない。その他は、INSTRと同じ。

8.70. INLIKE

(1) 機能

文字列(var)のLIKE一致ヶ所数と位置を返す。

(2) 一般形式

```
result = INLIKE(map_index, npos, var [, start [, len]], pat_str, [opt_str], [escape]);
```

(3) 構文規則

(A)map_indexについては、GETARGS()と同じ。

(B)第4、第5引数(start, len)に数値属性が指定されたときは、サーチ開始位置、文字数と見なす。

サーチ開始位置は、サーチ対象文字列(var)の先頭を1文字目として文字単位で数える。

(C)第4引数(start)に範囲指定が指定されたときは、サーチ開始位置、終了位置と見なす。

(4) 一般規則

(A)resultには、一致したヶ所数を返す。

(B)map_indexには、先頭から順に、一致した位置と長さをnpos組返す。

位置と長さの単位は文字数。位置はサーチ対象文字列の先頭を1文字目と数える。

位置は、オプションによって、サーチ開始位置が1文字目が変わる。(INSTR()と同じ)

nposがゼロのときには、map_indexには結果を返さない。

nposを省略したときは、最大でmap_indexの配列数/2組の結果をかえす。

(C)pat_str, opt_str, escapeは、REPLIKE()と同じ。

(D)サーチ方法は、INSTR()と同様。

8.71. INRSTR

(1) 機能

後ろからの文字列サーチ結果を返す。

(2) 一般形式

```
result = INRSTR(var [, start [, len]], str1 [, str2, ...]);
```

(3) 構文規則

INSTRと同じ。

(4) 一般規則

(A)一致した文字列中で最も後ろの先頭位置を返す。どれとも一致しなかったとき、0を返す。

(B)後ろからサーチする以外は、上記を除きINSTRと同じ。

8.72. INSTR

(1) 機能

文字列サーチ結果を返す。

(2) 一般形式

```
result = INSTR(var [, start [, len]], str1 [, str2, ...]) ;
```

(3) 構文規則

(A) 第2、第3引数(start, len)に数値属性が指定されたときは、サーチ開始位置、文字数と見なす。

サーチ開始位置は、サーチ対象文字列(var)の先頭を1文字目として文字単位で数える。

(B) 第2引数(start)に範囲指定が指定されたときは、サーチ開始位置、終了位置と見なす。

(4) 一般規則

(A) 一致した文字列中で最も手前の先頭位置を返す。どれとも一致しなかったとき、0を返す。

(B) サーチは文字単位で行い、一致したときの位置は、サーチ対象文字列の先頭を1文字目として文字単位で数える。一致位置は、オプションによって、サーチ開始位置が1文字目が変わる。

(C) スクリプトの仕様で、返却する一致文字列の位置を数える単位が変わる。

旧仕様：バイト単位

新仕様：文字単位

8.73. iREGEX

(1) 機能

大文字、小文字を区別しない正規表現のパターン・マッチング結果を返す。

(2) 一般形式

```
result = iREGEX(var [, start [, len]], pat1 [, pat2, pat3, ...]) ;
```

(3) 構文規則

(A) REGEX()と同じ。

(4) 一般規則

(A) 半角について大文字、小文字を区別しない。その他は、REGEXと同じ。

8.74. IS

(1) 機能

文字列を調査する。

(2) 一般形式

```
ret = IS(var [, start [, len]], check) ;
```

(3) 構文規則

(A) 文字列演算式の第一項と第二項がそれぞれ第1引数と第2引数に対応する。

(B) 第2引数(start)に数値属性が指定されたときは、サーチ開始位置と見なす。

サーチ開始位置は、サーチ対象文字列の先頭を1文字目として文字単位で数える。

(4) 一般規則

(A) 第1引数が文字列に変換されない場合は、サーチ開始位置情報は使用されない。

(B) その他は、文字列演算子を参照。

8.75. LAST_DAY

(1)機能

月の最終日付を求める。

(2)一般形式

```
result = LAST_DAY(var) ;
```

(3)構文規則

なし

(4)一般規則

(A) varがNULL値のとき、または、lenが0以下のときは、NULL値を返す。

8.76. LEFT

(1)機能

文字列の左側を取り出す

(2)一般形式

```
result = left(var, len) ;
```

(3)構文規則

なし

(4)一般規則

(A) varがNULL値のとき、または、lenが0のときは、NULL値を返す。

(B) lenがvarの文字列長より大きいときは、varをそのまま返す。

(C) varが文字属性のときは、スクリプトの仕様で、lenの単位が変わる。

旧仕様：バイト単位

新仕様：文字単位

(D) lenが負のときは、varの文字列長から|len|を引いた長さを取り出す。

8.77. LEFTB

(1)機能

バイト単位で文字列の左側を取り出す

(2)一般形式

```
result = leftb(var, bytes) ;
```

(3)構文規則

なし

(4)一般規則

(A) varがNULL値のとき、または、bytesが0のときは、NULL値を返す。

(B) bytesがvarの文字列長(バイト)より大きいときは、varをそのまま返す。

(C) lenが負のときは、varの文字列長から|bytes|を引いた長さを取り出す。

8.78. LENG

(1)機能

データの長さまたは文字数を返す。

(2)一般形式

len = LENG(var) ;

(3)構文規則

なし。

(4)一般規則

(A)varの属性により返す値は以下となる。

(a)一般変数

No.	属性	返す値
1	文字	スクリプト仕様により長さを数える単位が変わる。 旧仕様：バイト単位。 新仕様：文字単位。
2	整数	4
3	2進浮動小数点	8
4	10進小数点	40
5	BULK	データ長(バイト)
6	日付	14

(B)その他の変数

32を返す。

8.79. LENB

(1)機能

バイト単位で長さを返す。

(2)一般形式

len = LENB(var) ;

(3)構文規則

なし

(4)一般規則

(A)varが文字属性のとき、バイト単位で長さを返す。その他のときは、LENG()と同じ。

8.80. LIKE

(1)機能

パターン・マッチング結果を返す。

(2)一般形式

```
result = LIKE(var [, start [, len]], pat1 [, pat2, ...]) ;
```

(3)構文規則

(A)start、lenについては、INSTR()と同じ。

(4)一般規則

(A)パターンは、SQLのLIKEと同様に指定する。

'%'：NULL値を含む任意の文字列に一致する。

'_'：NULL値を含む任意の1文字に一致する。

'[,]'：これで囲まれた文字列中のどれか1文字と一致する。

この文字列中では、ハイフン('-')で繋げて文字の範囲を複数指定することができる。

範囲指定の文字には、半角と全角の組み合わせを指定することができる。

ただし、左側の文字コード≦右側の文字コードであること。

'¥'：パターン文字に'%'、'_'、'[,]'を含めるとき、それらの前に付ける。

文字定数中で指定するときは、'¥'が文字定数でのエスケープ文字となるため、'¥¥'と指定しなければならない。

(B)パターンのどれかにマッチしたとき、パターンの並び順序番号を返す。

どれもマッチしなかったとき、0を返す。

(C)パターンのどれかがNULL値のときは、無条件にマッチする。

8.81. LIST

(1)機能

データ・リストを作成する

(2)一般形式

```
var = LIST([var1], [var2], ...) ;
```

(3)構文規則

FLと同じ。

(4)一般規則

FLと同じ。

8.82. LIST_REF

(1) 機能

リストのvar2番目の要素を参照する。(先頭は0番目)
var2省略時は、var2=0と同じ。

(2) 一般形式

```
var = LIST_REF(var1 [, var2]) ;
```

(3) 構文規則

Lと同じ。

(4) 一般規則

Lと同じ。

8.83. LOG

(1) 機能

自然対数を求める。

(2) 一般形式

```
ret = LOG(x) ;
```

(3) 構文規則

なし。

(4) 一般規則

- (A) xには、1.0E-6より大きい値を指定する。
- (B) xが浮動小数でないときは、浮動小数に変換される。

8.84. LOG10

(1) 機能

10を底とする対数を求める。

(2) 一般形式

```
ret = LOG10(x) ;
```

(3) 構文規則

なし。

(4) 一般規則

- (A) xには、1.0E-6より大きい値を指定する。
- (B) xが浮動小数でないときは、浮動小数に変換される。

8.85. LOGOUT

(1) 機能

ログを出力する

(2) 一般形式

```
ret = LOGOUT(logno, format [, var1, var2, ..., var5]) ;
```

(3) 構文規則

(4) 一般規則

- (A) lognoは、LET LOGPARMコマンドと同じ。

8.86. LPAD

(1)機能

文字列の左端に文字列を連結し、指定桁数にする。

(2)一般形式

```
result = LPAD(var, len [, pad]) ;
```

(3)構文規則

なし

(4)一般規則

(A)PADを省略するか、NULL値のとき、半角スペースが使われる。

(B)lenがvarの文字列長以下のときは、RIGHT()と同じ。

(C)スクリプトの仕様で、lenの単位が変わる。

旧仕様：バイト単位

新仕様：文字単位

8.87. LPADB

(1)機能

文字列の左端に文字列を連結し、指定バイト数にする。

(2)一般形式

```
result = LPADB(var, len [, pad]) ;
```

(3)構文規則

なし

(4)一般規則

(A)PADを省略するか、NULL値のとき、半角スペースが使われる。

(B)lenがvarの文字列長以下のときは、RIGHTB()と同じ。

(C)LENが全角文字の途中までのときでも、LENバイトを返す。

8.88. MAX

(1)機能

最大値を求める。

(2)一般形式

```
result = MAX(var1[, var2, ...]) ;
```

(3)構文規則

なし

(4)一般規則

- (A) var1が数値のときは、Var2以降はvar1の属性に変換されて比較される。
- (B) var1が文字のときは、Var2以降は文字に変換されて比較される。
- (C) var1が日付のときは、Var2以降は日付に変換されて比較される。

8.89. MIN

(1)機能

最小値を求める。

(2)一般形式

```
result = MIN(var1[, var2, ...]) ;
```

(3)構文規則

なし

(4)一般規則

MAXと同じ。

8.90. MOD

(1)機能

varをradで割った余りを求める。

(2)一般形式

```
result = MOD(var, rad) ;
```

(3)構文規則

なし

(4)一般規則

- (A) varおよびradが整数値でないときは、整数値に変換される。

8.91. NDEF

(1)機能

文字列で与えられた式を評価し、変数が未定義、変数値が未設定、NULLパラメータ属性を持つNULL値のとき、指定された値に置き換える

(2)一般形式

```
result = NDEF(var [, rep]) ;
```

(3)構文規則

varには、評価したい式を文字列で表した式を指定する。【例】\$xを調べたいときは、'\$x'とする。

(4)一般規則

- (A) varの内容が空のときはエラー。
- (B) repが省略されたときは、NULL値が使用される。
- (C) opt=0x01のときは、評価結果(1:変数が未定義等/0:変数が定義済み)を返す。

8.92. NEW

(1) 機能

クラスのインスタンスを生成する。

(2) 一般形式

```
inst = NEW({ クラス名 | var } [,パラメータリスト]) ;
```

(3) 構文規則

なし

(4) 一般規則

- (A) パラメータリストには、実行したいコンストラクタと同じ数のパラメータを指定する。
- (B) varには、クラス名をデータとする文字定数か文字変数を指定する。

8.93. NVAL

(1) 機能

式を評価し、NULL値のときは、指定された値に置き換える

(2) 一般形式

```
result = NVAL(var [, rep]) ;
```

(3) 構文規則

(4) 一般規則

- (A) repが省略されたときは、NULL値が使用される。
- (B) 式の値が未定義またはNULLパラメータのときはエラーとなる。

8.94. NOFREE

(1) 機能

未開放メモリのアドレスをファイルに出力する

(2) 一般形式

```
ret = NOFREE(var) ;
```

(3) 構文規則

(4) 一般規則

- (A)

8.95. OPENDIR

(1) 機能

ディレクトリをオープンする。

(2) 一般形式

```
dp = OPENDIR(dirname) ;
```

(3) 構文規則

(4) 一般規則

- (A) ディレクトリへのポインタを返す。

8.96. PCLOSE

(1)機能

パイプをクローズする。

(2)一般形式

ret = PCLOSE(fp) ;

(3)構文規則

(4)一般規則

(A)パイプ以外は、クローズできない。

8.97. POPEN

(1)機能

パイプをオープンしてコマンド・ラインを実行する

(2)一般形式

fp = POPEN(line [,mode]) ;

(3)構文規則

(4)一般規則

(A)mode省略時は、'r'。

(3)構文規則

8.98. POWER

(1)機能

xのy乗を求める。

(2)一般形式

ret = POWER(x, y) ;

(3)構文規則

なし。

(4)一般規則

(A)x, yが浮動小数でないときは、浮動小数に変換される。

8.99. PUTENV

(1)機能

環境変数の値を設定する(name=value形式)。

(2)一般形式

```
ret = PUTENV(name_value) ;
```

(3)構文規則

なし。

(4)一般規則

(A)本設定は、当該セッションの間のみ有効となり、変更された値は、セッション終了時に元に戻る。

(5)戻り値

0: 正常。-1: エラー。

8.100. PUTLINE

(1)機能

標準出力に複数個の文字列を書き込む。

(2)一般形式

```
ret = PUTLINE(v1, v2, ...) ;
```

以下、FPUTLINEと同じ。

8.101. RAND1 (DRAND48)

(1)機能

0.0<=、<1.0の間の乱数を求める。

(2)一般形式

```
ret = RAND1() ;
```

(3)構文規則

なし。

(4)一般規則

8.102. READDIR

(1)機能

ディレクトリの1エントリを読み込み、ファイル名を返す。

(2)一般形式

```
file = READDIR(dp) ;
```

(3)構文規則

なし

(4)一般規則

(A)エントリを読み切ると、NULL値を返し、\$ERROR=-1となる。

8.103. REGEX

(1)機能

正規表現のパターン・マッチング結果を返す。

(2)一般形式

```
result = REGEX(var [, start [, len]], pat1 [, pat2, pat3, ...]) ;
```

(3)構文規則

(A)start、lenについては、INSTR()と同じ。

(4)一般規則

(A)パターンは、UNIXのREGEX(拡張仕様)と同様に指定する。

'*':

'?':

'[,]': これで囲まれた文字列中のどれか1文字と一致する。

'¥': パターン文字に'*'、'?','[,]'を含めるとき、それらの前に付ける。

文字定数中で指定するときは、'¥'が文字定数でのエスケープ文字となるため、'¥¥'と指定しなければならない。

(B)パターンのどれかにマッチしたとき、パターンの並び順序番号を返す。

どれもマッチしなかったとき、0を返す。

(C)マッチするパターンより前にNULL値のパターンがあるときは、エラーとなる。

(D)パターンのどれかが'()'のときは、無条件にマッチする。

8.104. REP

(1) 機能

文字列の置換

(2) 一般形式

```
result = REP(var [, start [, len]], rep_parm) ;
```

(3) 構文規則

(A) 文字列演算式の第一項と第二項がそれぞれ第1パラメータと第2パラメータに対応する。

(B) start、lenについては、INSTR()と同じ。

(4) 一般規則

文字列演算子を参照

8.105. REPLIKE

(1) 機能

文字列のLIKE置換

rep_patに一致する文字列をrep_strに置換する。

resultには、置換対象文字列(var)の先頭以降の置換された結果の文字列を返す。

オプションによって、サーチ開始以降の文字列を返す。

(2) 一般形式

```
result = REPLIKE(var [, start [, len]], rep_pat, rep_str, opt_str, escape);
```

(3) 構文規則

(A) start、lenについては、INSTR()と同じ。

(4) 一般規則

(A) opt_strに'i'または'I'が含まれるときは、半角英字の大文字、小文字は無視される。

(B) opt_strに'g'または'G'が含まれるときは、rep_patに一致する全ての文字列をrep_strに置換する。

(C) rep_patがNULL値のときは、何も置換しない。

(D) '^'は、先頭に一致することを示す。逆に、'\$'は、末尾に一致することを示す。

(E) rep_patの指定方法は、LIKEと同様である。ただし、先頭または末尾に'^'、'\$'、'%がないときは、'%があるものと見なされる。

以下に、例を示す。

(F) opt_strに'o'または'O'が含まれるときは、rep_patに一致する文字列を抜き出す。

(G) escapeが指定されたときは、それをエスケープ文字とする。escapeが、NULL値のときは、エスケープ処理を行わない。

デフォルトは、'¥'である。使用方法は、LIKE関数を参照のこと。

(H) rep_patが1文字であり、エスケープ文字のときは、エスケープ文字とは見なさない。

No.	rep_pat	置換動作	'g' 指定時
1	'ABC'	初めの'ABC'がrep_strに置換される	全ての'ABC'が置換される
2	'^'	先頭にrep_strが挿入される	同左
3	'\$'	末尾にrep_strが追加される	同左
4	'^\$'	先頭と末尾にrep_strが挿入される	同左
5	'^ABC'	先頭の'ABC'がrep_strに置換される	同左
6	'ABC\$'	末尾の'ABC'がrep_strに置換される	同左
7	'^ABC\$'	varが'ABC'のときのみ置換される	同左
8	'A%C'	初めの'AC'または'A...C'が置換される	全ての'AC'または'A...C'が置換される
9	'%'	varがrep_strに置換される	同左
10	'^%ABC'	'ABC'と同じ	同左
11	'ABC%\$'	'ABC'と同じ	同左
12	'_B_'	初めの真ん中が'B'の3バイトが置換される	全ての左記3バイトが置換される
13	'^%'	'%'と同じ	同左
14	'%\$'	'%'と同じ	同左
15	'\$^'	varが'\$^'のときのみ置換される	同左
16	'\$\$'	先頭の'\$'が置換される	同左
17	'%^'	末尾の'^'が置換される	同左
18			

8.106. RESLOGPARM

(1)機能

取得済ログパラメータをリストアする。

(2)一般形式

```
ret = RESLOGPARM(map_index, [max_args]) ;
```

(3)構文規則

(A)map_indexは、GETARGS()と同じ。

(B)max_argsのデフォルト値は、7。

(4)一般規則

(A)map_indexは、GETARGS()と同じ。

(B)リストアされるログパラメータは、GETLOGPARMコマンドを参照。

(5)戻り値

設定パラメータ数。

8.107. REST

(1)機能

データ・リストの最初のvar2個の要素を除くリストを参照する。(var2>=0)

var2省略時は、var2=1と同じ。

(2)一般形式

```
var = REST(var1 [, var2]) ;
```

(3)構文規則

FLと同じ。

(4)一般規則

FLと同じ。

8.108. R I G H T

(1)機能

文字列の右側を取り出す

(2)一般形式

```
result = RIGHT(var, len) ;
```

(3)構文規則

なし

(4)一般規則

(A) varがNULL値のとき、または、lenが0のときは、NULL値を返す。

(B) lenがvarの文字列長より大きいときは、varをそのまま返す。

(C) varが文字属性のときは、スクリプトの仕様で、lenの単位が変わる。

旧仕様：バイト単位

新仕様：文字単位

(D) lenが負のときは、varの文字列長から|len|を引いた長さを取り出す。

8.109. R I G H T B

(1)機能

バイト単位で文字列の右側を取り出す

(2)一般形式

```
result = RIGHTB(var, bytes) ;
```

(3)構文規則

(4)一般規則

(A) varがNULL値のとき、または、bytesが0のときは、NULL値を返す。

(B) bytesがvarの文字列長(バイト)より大きいときは、varをそのまま返す。

(C) lenが負のときは、varの文字列長から|bytes|を引いた長さを取り出す。

8.110. RINT

(1)機能

指定した倍精度2進浮動小数点数に最も近い整数値を倍精度2進浮動小数点数で返す。

(2)一般形式

```
ret = RINT(x) ;
```

(3)構文規則

なし。

(4)一般規則

(A)xが浮動小数でないときは、浮動小数に変換される。

8.111. ROUND

(1)機能

数値を丸める。

(2)一般形式

```
result = ROUND(var, [scale], [opt]) ;
```

(3)構文規則

(4)一般規則

(A)scale桁に丸める。省略時は、ゼロと見なす。

scale \geq 0 : 小数点以下scale桁に丸める。

scale $<$ 0 : 小数点以上-scale桁目を丸める。

【例】 scale=-1 opt=0 var=125 ==> result=130

(B)optの下2ビットで丸め方を指定する。省略時は、ゼロと見なす。

opt = 0x00 : 四捨五入。

opt = 0x01 ビット0N : 切捨て。(0x02ビット0Nでも優先する)

opt = 0x02 ビット0N : 切上げ。

(C)varが文字列のときは、数値に変換される。

(D)数値の属性は保存される。

8.112. RPAD

(1)機能

文字列の右端に文字列を連結し、指定桁数にする。

(2)一般形式

```
result = RPAD(var, len [, pad]) ;
```

(3)構文規則

なし

(4)一般規則

(A)PADを省略するか、NULL値のとき、半角スペースが使われる。

(B)lenがvarの文字列長以下のときは、LEFT()と同じ。

(C)スクリプトの仕様で、lenの単位が変わる。

旧仕様：バイト単位

新仕様：文字単位

8.113. RPADB

(1)機能

文字列の右端に文字列を連結し、指定バイト数にする。

(2)一般形式

```
result = RPADB(var, len [, pad]) ;
```

(3)構文規則

なし

(4)一般規則

(A)PADを省略するか、NULL値のとき、半角スペースが使われる。

(B)lenがvarの文字列長以下のときは、LEFTB()と同じ。

(C)LENが全角文字の途中までのときでも、LENバイトを返す。

8.114. SET_DATE_PART

(1)機能

日付varを時間間隔単位に設定する。

(2)一般形式

```
DATE result = SET_DATE_PART(var, val, format) ;
```

(3)構文規則

なし。

(4)一般規則

(A) varが日付型でないときは、日付型に変換する。(ADD_MONTHS()参照)

(B) valに時間間隔単位の値を文字型で指定する。文字型でないときは、文字型に変換される。

各値は、formatに合わせて指定する。値が数字のときは、時間間隔文字の繰り返し文字数に一致しなければならない。ただし、時間間隔文字が、時間間隔文字以外の文字で区切られている場合は、この限りではない。

(C) formatが文字型でないときは、文字型に変換される。

formatには、以下の時間間隔文字を混合して指定できる。(大文字小文字区別なし)

YYYY	西暦年 4桁
YY, Y	西暦年 2桁(上2桁は、>=51:19, <51:20)
MM, M	月2桁
DD, D	日2桁
HH24, H24	時2桁(24時間制)
HH, H, HH12, H12	時2桁(12時間制)
MI, NN, N	分2桁
SS, S	秒2桁
DY, DAY	英語曜日名 (省略名または完全な名前)
MON, MONTH	英語月名 (省略名または完全な名前)。
P	"AM", "PM"
DDD	その年の通算日数(1月1日が0)

(D) format中には、エスケープ文字を指定することができる。

'¥'または'`'が使用できる。

また、2重引用符で囲まれた部分は、2重引用符が取り除かれそのまま出力される。

この中に2重引用符を入れるときは、2個続けて指定する。これは1で出力される。

(5)例

月(5)と日(14)を設定する。 SET_DATE_PART(SYSDATE, '5 14', 'MM DD')

時刻をPMで設定する。 SET_DATE_PART(SYSDATE, 'PM8', 'PHH')

月日を年通算日数で設定する。 SET_DATE_PART(SYSDATE, 99, 'DDD') 1月1日から100日目

8.115. SETARRAY

(1) 機能

配列にデータを設定する。

(2) 一般形式

```
n = SETARRAY(array, start, [var1], [var2], ...);
```

(3) 構文規則

(A) arrayには、配列名または数値を指定する。

(B) arrayに文字定数が指定されたときは、数値に変換される。

(C) arrayに連想配列名を指定したときは、varには、キー、データの順に指定する。

(4) 一般規則

(A) arrayに配列名を指定したときは、配列名[start]の要素から最大で、配列のサイズまで格納される。

(B) arrayに数値が指定されたときは、内部番号変数の\$(指定数値+start)の要素から、最大で配列のサイズまで格納される。

(C) 設定データが省略された位置に対応する要素への設定はスキップされる。

(D) 返却値nには、実際に設定されたデータ個数が返る。

(E) 連想配列で、キー省略されたデータはスキップされる。同じキーのときは、後が有効となる。

8.116. SETENV

(1) 機能

環境変数の値を設定する。

(2) 一般形式

```
ret = SETENV(name, value [, overwrite]);
```

(3) 構文規則

なし。

(4) 一般規則

(A) overwriteが0のときは、既存の環境変数の値を上書きしない。0以外のときは、上書きする。
overwrite省略時は、1と見なされる。

(B) 本設定は、当該セッションの間のみ有効となり、変更された値は、セッション終了時に元に戻る。

(5) 戻り値

0: 正常。-1: エラー。

8.117. SETLOGPARM

(1)機能

ログパラメータを設定する。

(2)一般形式

```
ret = SETLOGPARM(log_no, flag, [level], [size_max], [file_max], [option], [log_file]) ;
```

(3)構文規則

(A)後続の引数を省略するときは、不要なカンマも省略する。

(4)一般規則

(A)引数については、LET LOGPARAMコマンドを参照。

8.118. SHELL

(1)機能

システムのコマンド・ラインを実行する。

(2)一般形式

```
ret = SHELL(line) ;
```

(3)構文規則

(4)一般規則

(A)

8.119. SHUTCTL

(1)機能

s h u t 状態を返却する。

(2)一般形式

```
ret = SHUTCTL([mode, var]) ;
```

(3)構文規則

(4)一般規則

(A) s h u t 状態であり、s h u t が保留されていないとき、0以外を返す。

8.120. SIN

- (1) 機能
sinを求める。
- (2) 一般形式
ret = SIN(x) ;
- (3) 構文規則
なし。
- (4) 一般規則
 - (A) xはラジアンで指定する。
 - (B) xが浮動小数でないときは、浮動小数に変換される。

8.121. SINH

- (1) 機能
sinhを求める。
- (2) 一般形式
ret = SINH(x) ;
- (3) 構文規則
なし。
- (4) 一般規則
 - (A) xはラジアンで指定する。
 - (B) xが浮動小数でないときは、浮動小数に変換される。

8.122. SKIP_OPT

- (1) 機能
文字列をサーチし、スキップした文字数を返す。
- (2) 一般形式
pos = SKIP_OPT(str, pat [, opt]) ;
- (3) 構文規則
なし。
- (4) 一般規則
 - (A) optは以下を組み合わせで指定する。

```
0x01; /* ignore case */
0x02; /* reverse */
0x08; /* 0/1=in/to */
```
 - (B) str中でpat内のどれかの文字を対称として、optのスキップ指定に合致するスキップ文字数を返す。
 - (C) サーチは文字単位で行い、一致したときの位置は、サーチ対象文字列の先頭を1文字目として文字単位で数える。
 - (D) スクリプトの仕様で、返却する一致文字列の位置を数える単位が変わる。
旧仕様：バイト単位
新仕様：文字単位

8.123. SORT

(1)機能

配列等のデータをソートする。

(2)一般形式

```
ret = SORT(array_d, array_s, number, [option], [ソート位置指定のリスト]) ;
```

(3)構文規則

(A)array_d および array_s には、配列名または数値を指定する。

文字データが指定されたときは、数値に変換される。

(B)option

数値または文字列で指定する。

指定形式	値	意味
数値指定	0x01ビット	0:バイト単位での範囲指定(省略時) 1:区切り文字で区切られた項目指定
	0x02ビット	0:データをソート対象とする 1:連想配列のキー値+データ値をソート対象とする
	0x04ビット	1:連想配列のキー値のみをソート対象とする
文字列指定	Fixed または Ranged	バイト単位での範囲指定(省略時)
	Item[=X] または Column[=X]	区切り文字で区切られた項目指定 X:半角1バイトの区切り文字 省略時の区切り文字は、空白文字またはカンマ
	Option=opt	区切り文字に関するオプション
	Key[=Only]	連想配列のキー値とデータ値を区切り文字で連結したものをソート対象とする。 Only が指定されたときは、キー値のみをソート対象とする。

(注)文字列指定時は、先頭文字のみで判定する。上記以外の場合は数値に変換する。

区切り文字に関するオプション(opt)
0x01=1:'#'以降を無視する
0x02=1:2ワード目が、'='のときは無視する。1ワード目のときは、NULL値とみなす。
0x04=1:', 'も区切りとする。', 'が現れた時点で0x12=0Nの指定を無効にする。
0x10=1:2ワード目が、': ' or ':='のときは無視する。1ワード目のときは、NULL値とみなす。
0x20=1:'['と']'を引用符と同様に扱う(IPV6対応)
0x40=1:0x12=1のとき、'= ' or ': ' or ':='が1ワード目のときには、1ワード目として設定する。
0x80=1:0x04=1のとき、', 'が現れた時点で0x12=0Nの指定を無効にしない
0x010000=1:引用符の中の連続する2つの引用符を1つにしない

(注)区切り文字が指定されたときは、0x02, 0x04, 0x10, 0x40, 0x80 は、0となる。

(C) ソート位置指定のリスト

範囲指定: pos1, len1, order1, pos2, len2, order2, ...

項目指定: col1, order1, col2, order1, ...

posは先頭を1バイト目と数える。

lenはバイト単位で数える。

len省略時は、posから末尾までが範囲となる。

colは先頭を1項目目と数える。

order は、数値または文字列で指定する。

文字列のときは、各キーワードを空白文字で区切って指定する。

指定形式	値	意味
数値指定	0x01ビットが、0	昇順(省略時)
	0x01ビットが、1	降順
	0x02ビットが、0	文字列として比較する(省略時)
	0x02ビットが、1	数値として比較する
文字列指定	A sc	昇順(省略時)
	D esc	降順
	C har または S tring	文字列として比較する(省略時)
	N umber または I nteger	数値として比較する Iのときは、整数値として比較する

(注) 文字列指定時は、先頭文字のみで判定する。上記以外のときは数値に変換する。

(4) 一般規則

(A) array_dに配列名を指定したときは、配列の先頭要素から、ソート結果が格納される。

数値が指定されたときは、内部番号変数の\$(指定数値)の要素から、ソート結果が格納される。

配列には、パラメータ変数または検索変数へのMAPPEDARRAYは、指定できない。

(B) array_sに配列名を指定したときは、配列の先頭要素からが、ソート対象データとなる。

数値が指定されたときは、内部番号変数の\$(指定数値)の要素からが、ソート対象データとなる。

(C) option以降が省略された場合は、全データが昇順のソート対象となる。

(D) 数値変換は、先頭空白文字をスキップし、整数または浮動小数点形式以外の文字が現れた時点で終了し、それまでの変換結果が数値として扱われる。変換エラーのときは、0と見なされる。

(E) ソート対象データの属性が数値であり、ソート位置が先頭のみで数値で比較するときは、数値データがそのままソートに使われる。その他のときは一旦文字列に変換され、ソート位置指定にしたがって変換されソートに使われる。

ただし、ソート結果には元のデータが返される。

8.124. S Q R T

- (1)機能
平方根を求める。
- (2)一般形式
`ret = SQRT(x) ;`
- (3)構文規則
なし。
- (4)一般規則
 - (A)xが浮動小数でないときは、浮動小数に変換される。
 - (B)xが負のときは、エラー。

8.125. S R A N D 1 (S R A N D 4 8)

- (1)機能
乱数のSEED(種)を設定する。
- (2)一般形式
`ret = SRAND1(x) ;`
- (3)構文規則
なし。
- (4)一般規則
 - (A)xが整数型でないときは、整数型に変換される。

8.126. S T R I N G S

- (1)機能
文字列を繰り返す
- (2)一般形式
`result = STRINGS(var, repeat) ;`
- (3)構文規則
なし
- (4)一般規則
 - (A)varがNULL値のとき、または、repeatが0以下のときは、NULL値を返す。

8.127. SUBSTR (MID)

(1)機能

文字列を切り出す。

(2)一般形式

```
result = SUBSTR(var, [pos], [len]) ;
```

(3)構文規則

文字列演算式の第一項～第三項がそれぞれ第1パラメータ～第3パラメータに対応する。

(4)一般規則

(A) varが文字属性のときは、スクリプトの仕様で、posとlenの単位が変わる。

旧仕様：バイト単位

新仕様：文字単位

(B)その他は、文字列演算子を参照。

8.128. SUBSTRB (MIDB)

(1)機能

バイト単位で文字列の切り出し。

(2)一般形式

```
result = SUBSTRB(var, [pos], [len]) ;
```

(3)構文規則

文字列演算式の第一項～第三項がそれぞれ第1パラメータ～第3パラメータに対応する。

(4)一般規則

文字列演算子を参照。

8.129. SYSLOG

(1)機能

syslogにログを出力する

(2)一般形式

```
ret = SYSLOG(syspri, format[, var1, var2, ..., var5]) ;
```

(3)構文規則

(4)一般規則

(A)

8.130. TAN

- (1) 機能
tanを求める。
- (2) 一般形式
ret = TAN(x) ;
- (3) 構文規則
なし。
- (4) 一般規則
 - (A) xはラジアンで指定する。
 - (B) xが浮動小数でないときは、浮動小数に変換される。

8.131. TANH

- (1) 機能
tanhを求める。
- (2) 一般形式
ret = TANH(x) ;
- (3) 構文規則
なし。
- (4) 一般規則
 - (A) xはラジアンで指定する。
 - (B) xが浮動小数でないときは、浮動小数に変換される。

8.132. TIMES

- (1) 機能
関数または式文字列を指定回数実行する。関数の返却値または式文字列実行結果を返す。
- (2) 一般形式
result = TIMES([num], [func, p1, p2, ...]) ;
result = TIMES([num], [式文字列 [, opt]]) ;
- (3) 構文規則
関数(func)のパラメータ(p1, p2, ...)は、関数に合わせて指定する。
- (4) 一般規則
 - (A) 回数(num)省略時は、\$MAX_LOOP_WHILE回繰り返す。
 - (B) 関数または式文字列を省略した場合は、何もしないで、NULL文字を返す。
 - (C) 式文字列を指定した場合は、EVAL()関数と同じ。

8.133. TO

- (1) 機能
文字列の変換
 - (2) 一般形式
result = TO(var [, start [, len]], trans) ;
 - (3) 構文規則
 - (A) 文字列演算式の第一項と第二項がそれぞれ第1パラメータと第2パラメータに対応する。
 - (B) start、lenについては、INSTR()と同じ。
 - (4) 一般規則
文字列演算子を参照
-

8.134. TO_BULK

(1)機能

各型のデータをそのままバルク型に変換し、バイト単位で切り出す。

(2)一般形式

```
result = TO_BULK(var, [pos], [len], [opt]) ;
```

(3)構文規則

カンマ以降のパラメータを全て省略する場合は、カンマを省略可能。

(4)一般規則

(A)切り出しの規則は、SUBSTRBと同じ。

(B)以下のように変換される。

データ型	opt	変換
数値	0 または、省略	ネットワーク・バイト・オーダーとなる
	1	ホスト・バイト・オーダーとなる
文字	0 または、省略	変換しない
	1	1 6進文字列とみなし、2 バイトずつ数値に変換する
日付	—	日付データをそのまま変換する(内容は付録を参照)
BULK	無視	変換しない

8.135. TO_BULKS

(1)機能

各型のデータをそのままバルク型に変換し、連結する

(2)一般形式

```
result = TO_BULKS(var1[, var2, ...]) ;
```

(3)構文規則

(4)一般規則

(A)数値データは、ネットワーク・バイト・オーダーとなる。

(B)文字型、日付型、バルク型はそのまま使われる。

8.136. TO_CHAR

(1) 機能

文字型に変換する。

(2) 一般形式

```
CHAR result = TO_CHAR(var [, format]) ;
```

(3) 構文規則

なし。

(4) 一般規則

(A) varが日付型のときは、formatに従って文字列に変換する。

(B) formatが文字型でないときは、文字型に変換される。

formatには、UNIX形式またはSQL形式が使用できる。ただし、混合はできない。
SQL形式において、時間間隔文字が1桁のときは、ゼロサプレスされる。

UNIX形式 SQL形式(大文字小文字区別なし)

%Y	YYYY	西暦年 4桁
%y	YY, Y	西暦年 2桁
%m	MM, M	月2桁
%d	DD, D	日2桁
%H	HH24, H24	時2桁(24時間制)
%I	HH, H, HH12, H12	時2桁(12時間制)
%M	MI, NN, N	分2桁
%S	SS, S	秒2桁
%a, %A	DY, DAY	英語曜日名 (省略名または完全な名前)
%b, %B, %h	MON, MONTH	英語月名 (省略名または完全な名前)。
%p	P	"AM", "PM"
%j	DDD	その年の通算日数(1月1日が0)

(C) formatが省略されたときは、'YYYY/MM/DD HH/MI/SS' が使われる。

(D) format中には、エスケープ文字を指定することができる。

(a) UNIX形式

'%'を使用できる。

(b) SQL形式

'¥'または'`'が使用できる。

また、2重引用符で囲まれた部分は、2重引用符が取り除かれそのまま出力される。

この中に2重引用符を入れるときは、2個続けて指定する。これは1で出力される。

(E) varが日付型でないときは、ADD_MONTHS()と同じ。

8.137. TO_DATE

(1) 機能

日付型に変換する。

(2) 一般形式

```
DATE result = TO_DATE(var [, format]) ;
```

(3) 構文規則

なし。

(4) 一般規則

(A) varが文字型でないときは、文字型に変換される。

(B) formatは、TO_CHAR()と同じ。

(C) 各時間間隔に対応する数字列は、formatがUNIX形式のときは、ゼロサプレスされていても良い。

SQL形式のときは、SET_DATE_PART()と同じ。

(D) 未指定の上位の時間間隔は、現在日時の値となる。

(E) 未指定の下位の時間間隔は、その最初の値になる。

8.138. TO_NUMBER

(1)機能

数値への変換

(2)一般形式

```
num = TO_NUMBER(var, attr, [size], [scale], [opt]) ;
```

(3)構文規則

(A)attrは、以下の値を指定する。

整数 : 2 (\$BIN[ARY], \$INT[EGER])

2進浮動小数点数 : 3 (\$FLO[AT], \$DOU[BLE])

10進浮動小数点数: 4 (\$DEC[IMAL])

(4)一般規則

(A)size, scaleは、10進浮動小数点数のときのみ有効。

sizeを指定したときは、10進固定小数点数に変換する。

(B)optは、

0x01: 数字列の後ろに数字(. +EDを含む)以外があってもエラーとしない。

0x02: 数字列中のカンマを無視する。(10進浮動小数点数のみ)

8.139. TRIM

(1)機能

文字列の前後の空白文字を削除する。

(2)一般形式

```
result = TRIM(var [, opt]) ;
```

(3)構文規則

(4)一般規則

(A)optを省略したときは、文字列の前後の空白文字を削除する。

(B)opt<=1を指定したときは、文字列の後の空白文字を削除する。

(C)opt>=1を指定したときは、文字列の前の空白文字を削除する。

8.140. UNLINK

(1)機能

ファイルを削除する。

(2)一般形式

```
ret = UNLINK(fname) ;
```

(3)構文規則

(4)一般規則

8.141. UNSETENV

(1) 機能

環境変数を削除する。

(2) 一般形式

```
ret = unsetenv(name) ;
```

(3) 構文規則

なし。

(4) 一般規則

(A) 本設定は、当該セッションの間のみ有効となり、変更された値は、セッション終了時に元に戻る。

(5) 戻り値

0: 正常。 -1: エラー。

8.142. XHASH

(1) 機能

ハッシュ処理を行う。

(2) 一般形式

(A) 新規にハッシュ表を作成する

```
hp = XHASH('New', $sKeyLen, $lMaxReg, [$lPreReg], [$DataFlag]) ;
```

hp : = 0 : エラー

<>0 : ハッシュ表へのポインタ

\$sKeyLen : キー定義長 (バイト)

>0 : 指定の固定長キー。内容は任意。キー定義長は 1 ~ 32,759 バイト。

キーのデータ型が文字と BULK 以外は、データ長以上を指定すること。

データ長については、付録「データ属性の値」を参照。

=0 : NULL 終端文字列のキー。

\$lMaxReg : 最初に取りられるエントリ数。(>2)

\$lPreReg : 第一割り当て数。0 か、省略時は自動計算。

\$DataFlag : >0 のとき、データを保存。<=0 か、省略のとき、データを保存しない。

(B) ハッシュ表を削除する

```
ret = XHASH($hp, 'Free') ;
```

ret : 常に 0

\$hp : ハッシュ表へのポインタ

(C) 登録/検索/削除

```
index = XHASH($hp, $Func, $cKey, [$Data]) ;
```

index : < 0 : エラー

= 0 : 空きなし

> 0 : ハッシュされたエントリへのインデックス。

\$hp : ハッシュ表へのポインタ

\$Func : コマンド

'S' : 登録

'R' : 検索

'D' : 削除

\$cKey : ハッシュ・キー

キーには、一般データを指定できる。

固定長キーで、

a) ハッシュ・キー長 > キー定義長 のとき、

- ・文字、BULK属性：余った部分は切り捨てられる。
- ・その他のデータ属性：エラーとなる。

b) ハッシュ・キー長 < キー定義長 のとき、不足部分は半角スペースが入る。

\$Data : 登録のとき、指定されたデータを登録する。

検索、削除のとき、以下の指定で変数にデータを返却する

整数値 : 整数値を番号とする内部番号変数が対象

配列変数名 : 配列変数名[0]が対象

(D) チェック

```
ret = XHASH($hp, 'K', $index, [$cKey], [$Data]) ;
```

ret : < 0 : エラー

= 0 : 未使用

> 0 : 使用中 (ハッシュされたエントリへのインデックス)

\$hp : ハッシュ表へのポインタ

\$index : チェックするエントリへのインデックス

\$cKeyRet : ハッシュ・キーを返却する以下の変数指定

整数値 : 整数値を番号とする内部番号変数が対象

配列変数名 : 配列変数名[0]が対象

\$DataRet : データを返却する変数指定

指定方法は、\$cKeyRetと同じ

(E) 登録数/登録インデックスの最大値

```
ret = XHASH($hp, $Func) ;
```

ret : < 0 : エラー

その他 : 登録数/登録インデックスの最大値

\$hp : ハッシュ表へのポインタ

\$Func : コマンド

'U' : 登録数

'M' : 登録インデックスの最大値

(3) 構文規則

(4) 一般規則

(A)