

목차

1. Crownix Report 요약 함수	4
1.1. 연산자	4
1.1.1. 산술 연산자	4
1.1.2. 관계 연산자	4
1.1.3. 논리 연산자	5
1.1.4. 문자열 연산자	5
1.1.5. 연산자 우선순위	5
1.2. 요약함수	7
1.2.1. ALIGNSTR()	8
1.2.2. ALLDATASUM()	9
1.2.3. AT()	10
1.2.4. ATOI()	11
1.2.5. AVG()	12
1.2.6. AVGEX()	13
1.2.7. CEIL()	14
1.2.8. CHANGEATTR()	15
1.2.9. CHANGEROWATTR()	17
1.2.10. CHANGETEXTATTR()	18
1.2.11. CODE2DATA()	20
1.2.12. CONVDATE()	21
1.2.13. COS()	22
1.2.14. COUNT()	23
1.2.15. CURGRPPAGENO()	24
1.2.16. CURRECINDEX()	25
1.2.17. DATE()	26
1.2.18. DATETOJULIAN()	27
1.2.19. DECIPHER()	28
1.2.20. DELETECOLUMN()	29
1.2.21. DELETEROW()	30
1.2.22. DISTINCT()	31
1.2.23. DIV()	32
1.2.24. DTOC()	33
1.2.25. ENTITYTOSTR()	34
1.2.26. EXEC()	35
1.2.27. FINDSTR()	36
1.2.28. FORMAT()	37
1.2.29. FORMATEX()	38
1.2.30. FULLTOHALF()	40

1.2.31. GETCOLUMN()	41
1.2.32. GETFIELD()	42
1.2.33. GETGLOBAL()	43
1.2.34. GETOBJVALUE()	44
1.2.35. GETPARAM()	45
1.2.36. GETVALUE()	46
1.2.37. GMTIME()	47
1.2.38. GRPCOUNT()	48
1.2.39. GRPTOTALPAGE()	49
1.2.40. HALFTOFULL()	50
1.2.41. HEXSTR()	51
1.2.42. IIF()	52
1.2.43. IIFEX()	53
1.2.44. ISFIRSTROW()	54
1.2.45. ISGRPNEXTPAGE()	55
1.2.46. ISGRPPREVPAGE()	56
1.2.47. ISLASTRECORD()	57
1.2.48. ISNUM()	58
1.2.49. ISVALIDDATE()	59
1.2.50. ITOA()	60
1.2.51. JULIANTODATE()	61
1.2.52. KEY2VALUE()	62
1.2.53. LASTRECINDEX()	63
1.2.54. LEFT()	64
1.2.55. LEN()	65
1.2.56. LOADHTML()	66
1.2.57. LOADIMAGE()	67
1.2.58. LOADIMAGE2()	68
1.2.59. LOADSWF()	69
1.2.60. LOADTEXT()	70
1.2.61. LOCALTIME()	71
1.2.62. LOG()	72
1.2.63. LOG10()	73
1.2.64. MAX()	74
1.2.65. MAXEX()	75
1.2.66. MERGECELLEX()	76
1.2.67. MID()	77
1.2.68. MIN()	78
1.2.69. MINEX()	79
1.2.70. MOD()	80
1.2.71. NUMTOARAB()	81

1.2.72. NUMTOENG()	82
1.2.73. NUMTOKOR()	83
1.2.74. NUMTOTIME()	84
1.2.75. NUMTOVIT()	85
1.2.76. NUMTOVITEX()	86
1.2.77. PAGE()	87
1.2.78. PERIOD()	88
1.2.79. POW()	90
1.2.80. RANK()	91
1.2.81. RANKEX()	92
1.2.82. REPLACESTR()	93
1.2.83. RIGHT()	94
1.2.84. ROUND()	95
1.2.85. RSUBSTR()	96
1.2.86. SETGLOBAL()	97
1.2.87. SETSTRINGFIT()	98
1.2.88. SIN()	99
1.2.89. SQRT()	100
1.2.90. STRCAT()	101
1.2.91. STRLEN()	102
1.2.92. STRMASK()	103
1.2.93. SUBALLDATASUM()	104
1.2.94. SUBSTR()	105
1.2.95. SUM()	106
1.2.96. SYSFORMAT()	107
1.2.97. TAN()	109
1.2.98. TIME()	110
1.2.99. TOLOWER()	111
1.2.100. TOTALPAGENO()	112
1.2.101. TOUPPER()	113
1.2.102. TRIM()	114
1.2.103. TRIMLEFT()	115
1.2.104. TRIMRIGHT()	116
1.2.105. TRUNC()	117
1.2.106. TTOC()	118
1.2.107. VAL()	119
1.2.108. XSUM()	120
1.2.109. NUMTO()	121
1.2.110. MERGECELLEX2()	122

1. Crownix Report 요약 함수

Crownix Report 의 요약은 결과데이터를 바로 적용하지 않고 어떠한 계산이나 데이터를 변환한 값을 적용하고자 할 때 사용합니다. 요약은 또한 데이터변환조건, 서브페이지출력조건 등의 조건에 따른 입력을 필요로 하는 기능에서 사용합니다. 변환되는 데이터는 둘 이상의 결과 값을 가질 수 없기 때문에 요약문은 하나의 문장으로 작성되어야 합니다.

1.1. 연산자

1.1.1. 산술 연산자

산술연산자는 숫자형 값의 연산(덧셈, 뺄셈, 곱셈, 나눗셈)을 위해서 사용합니다. '*', '/' 연산자가 '+', '-' 연산자보다 우선순위가 높습니다.

□ +

양쪽의 값을 더합니다. 또한 양수부호로 사용할 수 있습니다.

□ -

앞의 값에 뒤의 값을 뺍니다. 또한 음수부호로 사용할 수 있습니다.

□ *

양쪽의 값을 곱합니다.

□ /

앞의 값을 뒤의 값으로 나눕니다.

1.1.2. 관계 연산자

□ <=

왼쪽의 피연산자가 오른쪽의 피연산자 보다 작거나 같으면 '참'입니다. 오른쪽이 작다면 '거짓'입니다.

□ <

왼쪽의 피연산자가 오른쪽의 피연산자 보다 작으면 '참'입니다. 오른쪽이 작거나 같다면 '거짓'입니다.

□ **>=**

왼쪽의 피연산자가 오른쪽의 피연산자 보다 크거나 같으면 **'참'**입니다. 오른쪽이 크다면 **'거짓'**입니다.

□ **>**

왼쪽의 피연산자가 오른쪽의 피연산자 보다 크면 **'참'**입니다. 오른쪽이 크거나 같다면 **'거짓'**입니다.

□ **==**

양쪽의 피연산자가 같으면 **'참'**입니다. 양쪽의 피연산자가 다르면 **'거짓'**입니다.

□ **!=**

양쪽의 피연산자가 다르면 **'참'**입니다. 양쪽의 피연산자가 같으면 **'거짓'**입니다.

1.1.3. 논리 연산자

□ **AND, &&**

양쪽의 피연산자가 모두 **'참'**이어야 결과가 **'참'**입니다. 한 쪽이라도 **'거짓'**이면 결과는 **'거짓'**입니다.

□ **OR, ||**

양쪽에 있는 피연산자가 한쪽이라도 **'참'**이면 결과는 **'참'**입니다. 양쪽 모두가 **'거짓'**이어야만 결과가 **'거짓'**입니다.

1.1.4. 문자열 연산자

□ **&**

양쪽의 문자열을 붙여 줍니다. 요약함수에서 `strcat()` 함수와 같은 역할을 합니다.

1.1.5. 연산자 우선순위

연산자에는 (), +(양수 부호), -(음수 부호), *, /, %, &, +, -, >, >=, <, <=, ==, !=, AND, &&, OR, || 등이 있습니다

연산자의 우선순위는 다음 과 같습니다.

연 산 자	우선순위
()	높음
+(양수 부호), -(음수 부호)	
*, /, %, &	
+, -	
>, >=, <, <=, ==, !=	
AND, &&, OR,	낮음

1.2. 요약함수

요약 함수는 괄호 안에 인수를 포함합니다. 인수는 함수마다 각각 다르며 인수의 구별 표시는 첨표(,)를 사용합니다. 스크립트의 내장 함수처럼 함수의 인수로서 함수를 사용할 수도 있으며 중복되어 사용 가능합니다.

1.2.1. ALIGNSTR()

문법

ALIGNSTR(str, num, type)

인수

str
공백에 배치시킬 문자열 또는 문자열 변수

num
공백의 개수를 나타내는 숫자

type
공백에 문자열을 표시할 위치를 나타내는 숫자
0 : 왼쪽맞춤
1 : 오른쪽맞춤
2 : 가운데맞춤

설명

num 수 만큼의 공백 문자열내에 str 의 문자열을 type 에 맞게 배치시킵니다.

예

```
alignstr("teststring", 15, 1)
```

=> 결과는 다음 형태의 문자열이 됩니다.

					t	e	s	t	s	t	r	i	n	g
--	--	--	--	--	---	---	---	---	---	---	---	---	---	---

1.2.2. ALLDATASUM()

문법

ALLDATASUM(fieldname, opt)

인수

fieldname

합계를 낼 숫자형 변수

opt

fieldname 이 주 데이터의 필드명인지 서브페이지 데이터의 필드명인지 구분하는 숫자

0 : 주 데이터

1 : 서브페이지 데이터

설명

fieldname 값의 전체 합계를 구합니다. opt 가 0 이면 주 데이터의 값, 1 이면 서브페이지 데이터의 값입니다.

sum() 함수와 비슷하지만 sum() 함수의 값은 요약문이 실행될 때마다 계산되어 누적되며 alldatasum() 은 처음 요약문이 실행될 때 계산됩니다. sum() 함수를 표의 반복부에 적용하면 반복부가 증가할 때마다 누적되는 값을 볼 수 있지만 alldatasum() 함수를 반복부에 적용하면 처음에 나온 값이 그대로 계속 유지되는 것을 볼 수 있습니다.

예

```
alldatasum(unitprice, 0)
```

참고

SUM(), SUBALLDATASUM()

1.2.3. AT()

문법

AT(fieldname, str)

인수

fieldname
필드명

str
지정문자

설명

지정된 문자가 문자형 데이터의 선두부터 몇 번째 바이트인가를 숫자형 데이터로 취득합니다.

선두에 있는 경우, 1 이 됩니다.

하지만, 지정된 문자가 문자형 데이터가 아닌 경우, 유니코드 데이터로 읽어 들인 후, Ansi 데이터로 변환하여 바이트 수를 계산합니다.

전각 히라카나, 가타카나, 영문, 숫자 : 2 바이트

반각 히라카나, 가타카나, 영문, 숫자 : 1 바이트

예

```
at(" abcd" , " c" ) => 結果値 : 3  
at(" abc 文字" , " 字" ) => 結果値 : 6
```

1.2.4. ATOI()

문법

ATOI(str)

인수

str
숫자로 변환할 문자열 또는 문자열 변수

설명

str의 문자열을 숫자로 변환합니다. 일반적인 C-Language와는 다르게 부동소수까지 변환합니다.

예

```
atoi(unitprice) * 0.1
```

참고

VAL(), ITOA()

1.2.5. AVG()

문법

AVG(fieldname)

인수

fieldname

평균값을 구할 숫자형 변수

설명

fieldname의 평균을 구합니다. fieldname의 총합계에 레코드의 개수를 나눈값이 됩니다. 주로 소계나 합계를 표시하는 표의 요약부에 적용하여 사용합니다.

예

avg(salary)

참고

AVGEX()

1.2.6. AVGEX()

문법

AVGEX(fieldname)

인수

fieldname

평균값을 구할 숫자형 변수

설명

avg() 함수와 동일하게 평균값을 구합니다. 단, 레코드의 필드에 값이 들어 있는 경우만을 계산합니다. 평균값은 총합계에 레코드의 개수를 나눈 값이 되는데, 값이 들어오지 않는 경우는 레코드의 개수에 포함되지 않습니다. 그래서 값이 있는 경우만 계산되어 집니다.

예

```
avgex(salary)
```

1.2.7. CEIL()

문법

CEIL(num, pos)

인수

num

올림을 할 숫자 또는 숫자형 변수

pos

올림을 할 위치를 나타내는 숫자 (-9 ~ 9)

-1 : 십자리 미만 올림

0 : 소수점 미만 올림

1 : 소수 1 자리 미만 올림

설명

num 인수에 해당하는 값을 자리 수에 따라 올림합니다. 자리 수는 -9 에서 9 까지 유효합니다.

예

i 값이 765.267 인 경우

ceil(i, 0)

=> 결과값: 766 (소수점 미만 올림)

ceil(i, 2)

=> 결과값: 765.27 (소수 2 자리 미만 올림)

trunc(i,-1)

=> 결과값: 770 (십 자리 미만 올림)

참고

ROUND(), TRUNC()

1.2.8. CHANGEATTR()

문법

CHANGEATTR(fieldname, code, value)

인수

fieldname

적용할 데이터 또는 변수, 변수의 형은 관계없음

code

속성을 바꾸기위한 코드를 나타내는 문자열

value

속성을 바꾸기 위한 코드에 해당하는 수행값을 나타내는 문자열

설명

해당 변수명의 데이터를 출력할 때 문자속성이나 텍스트 상자 속성 등을 변경시키는 함수입니다. 수행 코드에 따른 값을 속성으로 적용시킵니다. 보통 `if()` 함수와 함께 어떤 조건을 만족할 경우에 속성을 변경시키고자 할 때 사용됩니다.

수행코드(`code`)가 여러 개라면 '@' 문자를 구분자로 하여 각각의 코드를 이어주며 수행값(`value`)도 또한 마찬가지로 합니다. 수행코드와 값은 다음과 같습니다.

code	설명	Value	
FC	폰트색상	R,G,B(각각 0 ~ 255 사이의 숫자)	
FS	폰트크기	4 ~ 127 사이의 숫자	
FN	폰트명	폰트이름	
FA	폰트속성	B	굵게
		I	기울임
		U	밑줄
		S	가운데선
		R	반전
		DB	굵게 사용안함
		DI	기울임 사용안함
		DU	밑줄 사용안함
		DS	가운데선 사용안함
		DR	반전 사용안함
JP	장평	33 ~ 200 사이의 값, 기본크기는 100	
BC	배경색상	R,G,B(각각 0 ~ 255 사이의 숫자)	
BP	배경무늬	1	
		2	
		3	
		4	
		5	
		6	
		7	

LS	문단의 줄 수	문단의 줄 수를 나타내는 숫자	
HA	가로정렬	1	왼쪽 정렬
		2	오른쪽 정렬
		3	가운데 정렬
		4	양쪽 정렬
		5	배분 정렬
VA	세로정렬	1	상단 정렬
		2	중앙 정렬
		3	하단 정렬
CL	취소선 종류	0	가로
		1	가로 이중(기본값)
		2	세로
		3	세로 이중
CC	취소선 색상	R,G,B(각각 0 ~ 255 사이의 숫자)	
CW	취소선 굵기	0	0,0
		1	0,5
		2	1,0
		3	1,5
		4	2,0(기본값)
		5	2,5
		6	3,0
		7	3,5
		8	4,0
		9	5,0
CS	이중 취소선의 선사이 간격	1 이상의 값. 기본값은 1	

예

```
changeattr(strname, "FC@FA", "255,0,0@B")
=> 결과는 strname 의 값을 객체에 적용하고 글꼴 색상을 빨강으로 하고 글꼴 속성을 bold 로 합니다.
```

참고

CHANGEROWATTR(), CHANGETEXTATTR()

1.2.9. CHANGEROWATTR()

문법

CHANGEROWATTR(fieldname, code, value)

인수

fieldname

적용할 데이터 또는 변수, 변수의 형은 관계없음

code

속성을 바꾸기위한 코드를 나타내는 문자열

value

속성을 바꾸기 위한 코드에 해당하는 수행값을 나타내는 문자열

설명

표의 행 전체의 문자속성이나 텍스트 상자 속성등을 바꿀 때 사용하는 함수입니다. 수행코드에 따른 값을 문자속성으로 적용시킵니다. 보통 `iif()` 함수와 함께 어떤 조건을 만족할 경우, 문자속성을 변경시키고자 할 때 사용됩니다. `changeattr()` 함수는 셀 하나의 속성을 바꿀 때 사용하고 `changerowattr()`은 표의 행 전체를 바꿀 때 사용합니다.

속성은 행 전체가 바뀌지만 `fieldname` 값은 행 전체에 표시되는 것이 아니라 `changerowattr()` 요약함수를 사용한 셀에만 표시됩니다.

수행코드(`code`)가 여러 개라면 '@' 문자를 구분자로 하여 각각의 코드를 이어주며 수행값(`value`)도 또한 마찬가지로 합니다. 수행코드와 값은 `CHANGEATTR()` 함수를 참조하시길 바랍니다.

예

```
changerowattr(customer.custid, "FS@FA@BC", "12@B@0,0,255")
```

=> 전체 행의 글꼴크기를 **12** 로 하고 문자속성을 **진하게** 하며 배경 색상을 **파란색**으로 하며 이 요약함수가 사용된 셀에는 `customer.custid`의 데이터를 표시합니다.

참고

CHANGEATTR(), CHANGETEXTATTR()

1.2.10. CHANGETEXTATTR()

문법

CHANGETEXTATTR(fieldname, str, code, value)

인수

fieldname

적용할 문자열 또는 문자열 변수

str

fieldname 문자열내에서 찾을 문자열 또는 문자열 변수

code

속성을 바꾸기 위한 코드를 나타내는 문자열

value

속성을 바꾸기 위한 코드에 해당하는 수행값을 나타내는 문자열

설명

fieldname 문자열내에서 str 문자열을 찾고 찾은 문자열의 문자 속성만 바꿔주는 함수입니다. changeattr 함수와 사용법이 거의 똑 같고 단지 찾을 문자열(str)이 추가되어 있습니다. str 값에 ""(빈문자열)을 주면 전체 문자열의 속성을 바꿉니다.

여러개의 문자열을 연결하면서 서로 다른 글자속성을 주고 싶을 때에는 아래에 나오는 예제와 같이 changetextattr() 함수를 여러 개 사용하여 문자열을 연결하는 함수를 이용하면 됩니다. 수행코드(code)가 여러 개라면 '@' 문자를 구분자로 하여 각각의 코드를 이어주며 수행값(value)도 또한 마찬가지로 합니다. 수행코드값은 다음과 같습니다.

code	설명	Value	
FC	폰트색상	R,G,B(각각 0 ~ 255 사이의 숫자)	
FS	폰트크기	4 ~ 127 사이의 숫자	
FN	폰트명	폰트이름	
FA	폰트속성	B	굵게
		I	기울임
		U	밑줄
		S	가운데선
		R	반전
		DB	굵게 사용안함
		DI	기울임 사용안함
		DU	밑줄 사용안함
		DS	가운데선 사용안함
DR	반전 사용안함		
JP	장평	33 ~ 200 사이의 값, 기본크기는 100	

주의) changetextattr() 함수의 경우, changeattr(), changerowattr() 함수와 형태가 비슷하지만,

전체 문단의 속성을 바꾸는 것이 아니라 특정 문자열의 속성을 바꾼다는 점에서 차이가 있습니다. 따라서, 위에 적혀진 수행코드 외에 `changeattr()`, `changerowattr()` 함수에서 사용가능한 HA, VA 등의 다른 수행코드들은 `changetextattr()` 함수에서 사용할 수 없습니다.

예

```
changetextattr(CustomerID, "ATR", "FA", "B" ) &  
  changetextattr(OrderID, "", "FA", "R") &  
  changetextattr(CustomerID, "", "FA", "I" )  
=> CustomerID 문자열의 "ATR" 부분만 굵게한 문자열과 OrderID 문자열 전체를 반전한 문자열,  
CustomerID 문자열 전체를 기울인 문자열을 이어놓은 문자열
```

참고

CHANGEATTR(), CHANGEROWATTR()

1.2.11. CODE2DATA()

문법

CODE2DATA(code, section name)

인수

code
코드명

section name
섹션명

설명

CodeMatch.dll 파일을 로드하여 입력한 **section** 에서 **code** 에 해당하는 데이터를 가져옵니다. 단, CodeMatch.dll 파일은 mrd 파일을 기준으로 ""mrd"" 폴더를 찾을때까지 위로 올라간 다음 그 상위 상위 폴더의 하위 폴더중 **common** 폴더 밑에서 찾습니다.

예

```
code2data("1", "ATM 구분 코드")
```

1.2.12. CONVDATE()

문법

CONVDATE(arg)

인수

arg
날짜형으로 바꿀 숫자 또는 문자열, 문자열 변수

설명

숫자 또는 문자 형태의 인수를 날짜형으로 바꿉니다. **Report Designer**에서는 서식을 적용하기 위해서는 데이터가 해당 서식에 맞는 형(**type**)으로 되어 있어야합니다. 데이터가 숫자나 문자열로 구성되어 있는 경우, 이를 날짜형으로 바꾸어 표현해야 날짜 서식 적용할 수 있습니다.

예

```
convdate(19990418)
convdate("1999/04/18")
```

참고

ATOI(), ITOA()

1.2.13. COS()

문법

COS(arg)

인수

arg
radian 형태의 숫자 또는 숫자형 변수

설명

삼각함수 중에서 cosine(코사인) 값을 구하는 함수입니다.

예

```
cos(3.141592/2)
```

참고

SIN(), TAN()

1.2.14. COUNT()

문법

COUNT(fieldname)

인수

fieldname

횟수를 구할 변수. 변수의 형은 상관없음

설명

인수가 반복되어 나오는 레코드의 횟수를 구합니다. 이 함수가 표의 반복부에서 사용되어 질 때에는 숫자가 하나씩 증가되는 것을 볼 수 있고 마지막에 나오는 요약부에서 사용되면 총 레코드의 개수를 표시하게 됩니다.

소계 등을 나타내는 중간에 표시하는 요약부에 적용할 경우, 임의로 다른 작업을 하지 않아도 해당 요약부가 나올 때마다 값이 리셋됩니다.

예

```
count(order)
```

참고

CURRECINDEX(), DISTINCT()

1.2.15. CURGRPPAGENO()

문법

CURGRPPAGENO()

인수

없음

설명

페이지 넘김 조건에 의한, 현재 페이지 번호를 숫자형 데이터로 취득합니다.

* 표 형식의 보고서만 지원하며, 서브 페이지와 서브 레포트는 지원하지 않습니다.

예

curgrppageno()

1.2.16. CURRECINDEX()

문법

CURRECINDEX(opt)

인수

opt

주 데이터인지 서브페이지 데이터인지를 구분하는 숫자

0 : 주 데이터

1 : 서브페이지 데이터

설명

현재 처리중인 주 데이터 또는 서브페이지 데이터의 레코드 인덱스 위치를 구하는데 사용됩니다.

예

```
currecindex(0)
```

```
currecindex(1)
```

참고

COUNT()

1.2.17. DATE()

문법

DATE()

인수

없음

설명

현재 날짜를 구합니다. 현재 보고서를 실행하고 있는 컴퓨터의 설정된 날짜를 돌려주는 함수입니다.

예

```
date()
```

참고

TIME()

1.2.18. DATETOJULIAN()

문법

DATETOJULIAN(date)

설명

날짜로 표현된 문자형 데이터를 Julian 으로 변환합니다.

예

```
datetojulian(date()) : 733797 // 요약함수의 경우  
datetojulian(sysdate) : 733797 // Oracle 날짜함수의 경우  
datetojulian(convdate("2010-01-25")) : 733797 // 문자열의 경우
```

1.2.19. DECIPHER()

문법

DECIPHER(encdata, opt)

인수

encdata

암호화된 데이터

opt

복호화 옵션(1)

설명

SEED 알고리즘으로 암호화된 문자열 데이터를 복호화 합니다.

복호화에 사용되는 키를 추가하려면 (주)엠투소프트에 문의 바랍니다.

예

```
decipher("/q13KnkCziB6PXs+08g--", 1) ==> "100100089098"
```

```
substr(decipher(a1,1), 1, 6) & "-" & substr(decipher(a1,1), 7, 7)
```

1.2.20. DELETECOLUMN()

문법

DELETECOLUMN(opt)

인수

opt

열을 삭제한 후, 표의 폭 크기를 늘릴지 아닐지 결정하는 숫자

0 : 해당 열만 삭제

1 : 해당 열을 삭제하고 표의 크기를 삭제된 크기만큼 폭을 늘림

설명

표의 열을 삭제합니다. **opt** 가 0 인 경우는 그냥 해당 컬럼만 삭제하고 **opt** 가 1 인 경우는 삭제된 컬럼의 **width** 만큼 표를 늘려줍니다. 보통 특정 조건에 따라서 이러한 작업을 하기 때문에 **iif()** 함수와 혼용해서 사용합니다.

예

```
iif(flag == 1, deletecolumn(0), coldata1)
```

=> flag 값이 1 이면 표의 열을 삭제하고 그렇지 않으면 coldata1 의 값을 해당 셀에 적용합니다.

참고

DELETEROW()

1.2.21. DELETEROW()

문법

DELETEROW()

인수

없음

설명

적용되는 셀을 포함하는 표의 행을 삭제합니다. 보통 특정 조건에 따라서 이러한 작업을 하기 때문에 `iif()` 함수와 혼용해서 사용합니다.

예

```
iif(flag == 1, deleterow( ), data)
=> flag 값이 1 이면 표의 행을 삭제하고 그렇지 않으면 data 의 값을 해당 셀에 적용합니다.
```

참고

DELETECOLUMN()

1.2.22. DISTINCT()

문법

DISTINCT(fieldname)

인수

fieldname

횃수를 계산할 변수. 변수의 형은 상관없음

설명

인수가 반복되어 나오는 레코드의 횃수를 구합니다. 단, 같은 값을 가지고 있을 때에는 횃수에 포함하지 않으며 인수에 다른 값이 나올 경우에만 카운트합니다. **count()** 함수는 인수의 값에 관계없이 횃수를 계산하지만 **distinct()** 함수는 동일한 값이 연속해서 오는 경우, 횃수에 포함시키지 않습니다.

소계 등을 나타내는 중간에 표시하는 요약부에 적용할 경우, 임의로 다른 작업을 하지 않아도 해당 요약부가 나올 때 마다 값이 리셋됩니다.

예

```
distinct(dept)
```

참고

COUNT()

1.2.23. DIV()

문법

DIV(fieldname1, fieldname2)

인수

fieldname1
숫자 또는 숫자형 변수

fieldname2
숫자 또는 숫자형 변수

설명

fieldname1 을 fieldname2 로 나눕니다. 필드 데이터 값을 가져와서 나누기 계산을 한 후, 결과 값을 적용합니다. '/' 연산자와의 차이점은 div() 함수는 결과값을 적용할 때 뒷만 적용시키고 소수점 이하 값은 버립니다.

예

```
div(totalprice, unitprice)
```

참고

'/' 연산자

1.2.24. DTOC()

문법

DTOC(fieldname)

인수

fieldname
변환할 날짜형 데이터

설명

날짜형 데이터(YYYYMMDD)를 문자형 데이터(YYYY/MM/DD)로 변환합니다.

Fieldname 에 변환할 날짜형 데이터가 들어 있는 필드명을 지정합니다.

예

날짜형 데이터를 문자형 데이터(YYYY/MM/DD)로 표시하는 경우

DTOC("20100811")

=> 결과값: "2010/08/11"

문자형 필드[필드]를 문자형 데이터(YYYY/MM/DD)로 표시하는 경우

DTOC(필드)

=> 결과값: "2010/08/11"

1.2.25. ENTITYTOSTR()

문법

ENTITYTOSTR(fieldname)

인수

fieldname

문자 엔티티를 포함하는 문자열 또는 문자열 변수

설명

문자열내에 HTML 에서 표현하는 문자 엔티티(character entity) 가 있으면 이를 문자로 변환합니다.

예

```
entitytostr("&#60;webmaster@m2soft.co.kr&#62;")  
=> 함수 수행 후 결과값: <webmaster@m2soft.co.kr>
```

참고

HEXSTR()

1.2.26. EXEC()

문법

EXEC(fieldname, code, value)

설명

changeattr() 함수와 같습니다. changeattr() 함수의 설명을 참조하시길 바랍니다.

참고

CHANGEATTR(), CHANGEROWATTR()

1.2.27. FINDSTR()

문법

FINDSTR(fieldname, str, pos)

인수

fieldname

문자열 또는 문자열 변수

str

찾을 문자열 또는 문자열 변수

pos

fieldname 문자열 내에서 str 문자열을 찾기 시작하는 위치를 나타내는 숫자
1 인 경우, 문자열의 처음부터 찾음

설명

문자열내에서 특정 문자열의 위치를 찾습니다. 리턴 값은 찾은 문자열의 위치이며 찾는 문자열이 없으면 0을 리턴합니다.

주의) 유니코드 버전에서는 문자의 위치는 문자의 개수와 연관되고 일반버전에서는 바이트 수와 연관됩니다.

예

```
findstr("abcdefgh", "cd", 1)
결과값: 3

findstr("abcdefgh", "cd", 4)
결과값: 0

findstr("대한민국 서울", "서울", 1)
유니코드 버전의 결과값: 6
일반 버전의 결과값: 10
```

참고

SUBSTR()

1.2.28. FORMAT()

문법

`format(fieldname, str, opt)`

인수

fieldname

특정 서식 문자열로 바꿀 변수(단, 변수형은 변경할 서식과 동일하게 설정)

str

서식을 나타내는 문자열

opt

의미없는 숫자표시 여부를 나타내는 숫자

0 : 의미없는 숫자 표시. 예) 2005-12-04

1 : 의미없는 숫자 표시안함. 예) 2005-12- 4

설명

변수명의 값을 서식 문자열의 형태로 바꾸어 줍니다. 리턴되는 값은 서식 문자열 형식으로 변환된 문자열입니다. 보통 서식맞추기 대화상자에서 대부분의 서식을 모두 설정할 수 있으나 서식맞추기 대화상자에서 설정하여 사용하지 않고 이 함수를 사용하는 경우는 특정 서식을 나타내는 데이터와 다른 데이터를 동시에 한 곳에 표시하고자 할 때입니다. 서식을 문자열로 바꾸어 다른 데이터와 & 연산자 등을 사용하여 문자열을 이어주는 것이 대표적인 예입니다.

opt 의 '의미없는 숫자'의 의미는 날짜형 데이터의 변환된 결과값이 "2005-09-10" 일때, 의미없는 숫자를 표시 안하게 되면 "2005- 9-10"으로 결과값을 리턴하고 숫자인 경우엔 소수점이하의 맨 뒤에 나오는 '0'을 없애줍니다.

fieldname 이 숫자형인 경우, 서식 문자열인 str 은 서식맞추기 대화상자에서 설정할 수 있는 서식문자열만 사용할 수 있습니다.

예

```
format(dateval, "yyyy-mm-dd", 0)
```

날짜형 데이터 dateval 가 '2006년 01월 30일'이라면 결과값: 2006- 1-30

```
format(numval, "#,###.###", 1)
```

숫자형 데이터 numval 가 1234.56 이라면 결과값: 1,234.560

```
format(timeval, "hh:mm:ss", 1)
```

시간형 데이터 timeval 가 '오후 12시정각' 이라면 결과값: 12:00:00

1.2.29. FORMATX()

문법

FORMATX(fieldname, str, opt)

인수

fieldname

서식을 지정할 변수명

str

서식으로 표시할 문자열

opt

0: 숫자형 변수를 서식에 맞추어 표시

1: 숫자형 변수 또는 문자형 변수를 서식에 맞추어 표시

설명

필드명에 서식을 적용할 숫자형 또는 문자형 변수를 지정합니다.

서식의 지정은 아래와 같습니다.

“ nZZZ,ZZ9.99m”

n: 선두 부가문자

▲ 마이너스(-)데이터인 경우 ▲를 표시합니다.

△ 마이너스(-)데이터인 경우 △를 표시합니다.

¥、\$、%、円、기타 임의 문자, 최후 부가문자

서식 문자

Z - 0 을 표시하지 않음

9 - 0 을 표시함

、 - 콤마를 삽입 (콤마 편집)

. - 소수점 위치 (소수점을 표시)

v - 가상 소수점 위치(소수점을 표시하지 않음)

m 최후 부가문자

円、% 등 임의의 문자를 표시합니다.

예

※△는 반각 스페이스를 의미합니다.

예 1) 숫자형 변수 A1 에 -123.456, 부가문자로 ▲를 지정한 경우

FORMATX(A1," ▲ZZ,ZZ9.99" ,0) -> " △△▲123.45"

FORMATX(A1," ▲ZZ,ZZ9.99" ,1) -> " ▲△△123.45"

예 2) 문자형 변수 B1 에 아무런 값이 입력되지 않은 경우

FORMATX(B1," ZZ,ZZ9.99" ,0) -> " △△△△△0.00"

FORMATX(B1," ZZ,ZZ9.99" ,1) -> " "

예 3) 문자형 변수 C1 에 아무런 값이 입력되지 않은 경우

FORMATX(C1," ZZ,ZZ9.99" ,0) -> " △△△△△0.00"

FORMATX(C1," ZZ,ZZ9.99" ,1) -> " "

opt 값이 0 인 경우, 0 과 Null 의 판단은 할 수 없습니다. 따라서, Null 의 경우도 0 이 표시됩니다.

opt 값이 1 인 경우, 0 과 Null 의 판단은 할 수 없습니다. 따라서, 0 은 0 으로 표시, Null 은 " " 로 표시됩니다.

1.2.30. FULLTOHALF()

문법

fulltohalf(str)

인수

str

반각(Half-width)으로 변경할 원본 문자열 또는 문자열 변수

설명

전각(Full-width)의 아스키(ASCII) 문자열을 반각(Half-width)으로 변경하기 위해 사용됩니다.

예

```
fulltohalf(" 1 3 5 5 - 1 9 ") ==> "1355-19"
```

1.2.31. GETCOLUMN()

문법

GETCOLUMN(index, colname)

인수

Index

가져올 컬럼의 인덱스

colname

컬럼이름을 나타내는 문자열

설명

지정한 인덱스 값(시작값은 0)에 해당하는 레코드의 특정컬럼 값을 가져오는데 사용합니다. 리포팅시에 현재 레코드의 데이터 이외에 이전이나 다음 데이터의 참조가 필요할 때 사용합니다.

주의) colname 은 컬럼이름을 나타내게 되는데 반드시 ""로 묶인 문자열을 입력합니다.

예

```
getcolumn(0, "EmployeeName")  
=> 컬럼이름 EmployeeName 의 첫 번째 레코드 값
```

```
getcolumn(1, "CustomerId")  
=> 컬럼이름 CustomerId 의 두 번째 레코드 값
```

참고

GETVALUE()

1.2.32. GETFIELD()

문법

GETFIELD(fieldname, delimiter, pos)

인수

fieldname

구분자를 포함하는 문자열 또는 문자열변수

delimiter

구분자를 나타내는 문자열

pos

구분자 사이의 위치를 나타내는 숫자
1 인 경우, 맨 앞에 위치해 있는 문자열을 가져옴

설명

구분자로 구분되어있는 문자열에서 특정위치의 문자열을 구합니다. Microsoft Excel 의 csv 파일의 데이터 내용과 같이 하나의 데이터 문자열내에서 구분자로 데이터를 분리해 낼 수 있는 경우, **substr()**함수를 사용하지 않고 쉽게 해당 데이터를 얻을 수 있습니다.

예

```
getfield("aaa||bbb||ccc", "||", 2)
=> 결과값: bbb

getfield("aaa||bbb||ccc", "||", 3)
=> 결과값: ccc
```

참고

SUBSTR()

1.2.33. GETGLOBAL()

문법

GETGLOBAL(varname)

인수

varname

전역변수명을 나타내는 문자열

설명

전역변수에 설정한 값을 가져옵니다. 전역변수 값 설정은 **setglobal()** 요약함수를 사용합니다.

setglobal() 과 **getglobal()** 함수는 특정 텍스트 상자의 요약함수를 통해서 나타난 결과를 다른 쪽에서 이를 이용하여 또 다른 표현을 하고자 할 때 이용합니다. 서브페이지가 여러 개 있는 경우, 앞의 서브페이지에서 계산한 합계를 뒤에 오는 서브페이지에서 사용하고자 할 때가 대표적인 예입니다.

주의) **setglobal()** 과 **getglobal()** 함수를 함께 사용하고자 할 때는 아래의 예와 같이 **&** 연산자를 사용할 수 있습니다. 하지만, **changeattr()** 과 같이 필드명을 인수로 갖는 다른 함수와 함께 사용하는 경우라면, **setglobal()** 과 **getglobal()** 함수는 반드시 **changeattr()** 함수 안에서 사용해야 합니다.

예

```
getglobal("myTempValue") => myTempValue 전역 변수 값 가져오기
```

```
setglobal("myTempValue", "m2soft") & getglobal("myTempValue") => 함께 사용하기
```

```
setglobal("a1", a1) & changeattr(getglobal("a1"), "FN", "굴림") => 틀린 사용법
```

```
changeattr(setglobal("a1", a1) & getglobal("a3"), "FN", "굴림") => 올바른 사용법
```

참고

SETGLOBAL()

1.2.34. GETOBJVALUE()

문법

```
GETOBJVALUE((varname, option)
```

인수

varname

객체의 변수명

option

옵션 (현재는 지원 안함, 차후 제공 예정)

설명

같은 페이지내에서 지정한 객체명(varname)의 객체를 찾아 해당 내용을 가져옵니다. 페이지의 내용이 바뀔때마다 값을 새로 가져오므로, 페이지번호 객체의 내용 등도 가져올 수 있습니다.

(단, 객체의 요약문이 아닌 페이지 출력 조건이나 객체 삭제 조건 등에서 사용한 경우에는 페이지의 내용이 바뀌어도 값을 새로 가져오지 않습니다.)

예

```
getobjvalue("aaa", "")
```

=> 같은 페이지에서 "aaa" 라는 객체를 찾아 해당 내용을 가져옵니다.

```
iif(getobjvalue("aaa", "") == "", "N", "Y")
```

=> 같은 페이지에 "aaa" 라는 객체가 없거나 해당 값이 없으면 "N"을, 객체의 값이 있으면 "Y"를 출력합니다.

참고

IIF()

1.2.35. GETPARAM()

문법

GETPARAM(arg)

인수

arg
 파라미터로 입력된 변수명을 나타내는 문자열

설명

/rp 나 /rv 파라미터로 입력된 값을 가져옵니다. 하지만 /rv 로 입력된 파라미터는 `getparam()` 함수를 사용하지 않아도 됩니다.

예

```
getparam("$1")  
=> /rp 옵션에 의하여 입력된 값중에서 $1 에 해당하는 값을 가져옵니다.  
  
getparam("custid")  
=> /rv 옵션에 의하여 입력된 값중에서 custid 에 해당하는 값을 가져옵니다. 하지만 단순히  
custid 라고만 사용해도 custid 에 해당하는 값을 가져올 수 있습니다.
```

1.2.36. GETVALUE()

문법

GETVALUE (colname, pos)

인수

colname
컬럼이름을 나타내는 문자열

설명

현재 레코드의 위치에서 **pos** 만큼 떨어진 레코드의 특정컬럼 값을 가져오는데 사용합니다. 리포팅시에 현재 레코드의 데이터 이외에 이전이나 다음 데이터의 참조가 필요할 때 사용합니다.

주의) colname 은 컬럼이름을 나타내게 되는데 반드시 ""로 묶인 문자열을 입력합니다.

예

```
getvalue("EmployeeName", -1)
=> 컬럼이름 EmployeeName 의 이전 데이터

getvalue("CustomerId", 1)
=> 컬럼이름 CustomerId 의 다음 데이터
```

참고

GETCOLUMN()

1.2.37. GMTIME()

문법

GMTIME(fieldname)

인수

fieldname

UTC 를 나타내는 숫자 또는 숫자형 변수

1970 년 1 월 1 일, 0 시정각을 0 으로 해서 초마다 1 씩 증가된 숫자

설명

숫자로 된 UTC(universal time coordinated) 값을 날짜/시간 형태로 변환해주는 함수입니다. 보통 PC 등에 저장되어있는 날짜/시간에 해당하는 값이 UTC 값이며 이를 알아보기 편하도록 변환해주는 함수입니다. localtime() 도 비슷한 기능을 하나 localtime()은 현재 설정되어있는 표준시간대로 시간을 변경시키고 gmtime()은 GMT 시간으로 표시합니다.

예

```
gmtime(393455871)
=> 결과값: 1982-06-20 21:17:51
```

참고

LOCALTIME()

1.2.38. GRPCOUNT()

문법

GRPCOUNT(fieldname)

인수

fieldname

횃수를 계산할 변수. 변수의 형은 상관없음

설명

인수로 들어오는 변수의 그룹카운트를합니다. 즉, 변수의 값이 계속 같은 경우에만 카운트를 하고 변수의 값이 바뀌면 다시 초기화합니다. 반면에 **distinct()** 함수는 동일한 값이 연속해서 오는 경우, 횃수에 포함시키지 않습니다.

소계 등을 나타내는 중간에 표시하는 요약부에 적용할 경우, 임의로 다른 작업을 하지 않아도 해당 요약부가 나올 때 마다 값이 리셋됩니다.

예

grpcount(year)

=> year 라는 필드명의 값에 계속 같은 값이 나오면 횃수를 증가시키고 다른 값이 나오면 초기화 시킨 후 다시 카운트합니다.

참고

COUNT(), DISTINCT()

1.2.39. GRPTOTALPAGENO()

문법

GRPTOTALPAGENO()

인수

없음

설명

페이지 넘김시 마다, 전체 페이지수를 숫자형 데이터로 취득합니다.

* 표 형식의 보고서만 지원하며, 서브 페이지와 서브 레포트는 지원하지 않습니다.

예

grptotalpageno()

1.2.40. HALFTOFULL()

문법

halftofull(str)

인수

str

전각(Full-width)으로 변경할 원본 문자열 또는 문자열 변수

설명

반각(Half-width)의 아스키(ASCII) 문자열을 전각(Full-width)으로 변경하기 위해 사용합니다.

예

```
halftofull("1355-19") ==> " 1 3 5 5 - 1 9 "
```

1.2.41. HEXSTR()

문법

HEXTSTR(str)

인수

str

16 진수 코드값이 들어있는 문자열 또는 문자열변수
문자열은 0~9 의 숫자 A~F 의 문자로 구성

설명

16 진수값을 문자열로 변환한 값으로 반환합니다.

예

```
hexstr("0D0A")  
=> 개행 문자열
```

참고

ENTITYTOSTR()

1.2.42. IIF()

문법

IIF(condition, op1, op2)

인수

condition
조건식

op1
조건식이 '참'일 때 실행되는 연산식

op2
조건식이 '거짓'일 때 실행되는 연산식

설명

특정 조건에 따라 서로 다른 연산식을 수행합니다. **condition** 의 조건식이 '참'일 때, **op1** 연산식을 수행하고 '거짓'일 때, **op2** 연산식을 수행합니다. 연산식에는 다시 **iif()**함수를 재귀적으로 (**recursive**) 사용할 수 있습니다.

주의) 요약문은 매 레코드마다 수행이 되지만 **iif()** 함수는 조건에 따라 **op1** 또는 **op2** 둘 중 하나의 연산식만 실행하게 됩니다. 만약, **op1**, **op2** 연산식 부분에 **sum()**, **count()**, **grpcount()**, **distinct()** 등의 함수를 사용하면 계산값이 잘못될 수 있습니다. 이러한 결과를 막기 위해서는 **alldatasum()**, **suballdatasum()** 등의 함수를 사용하거나 아래의 예제에서처럼 **iif()** 함수를 **sum()**, **count()**, **grpcount()**, **distinct()** 등의 함수의 인수로 사용하십시오.

예

```
iif(fieldname == "code11", count(fieldname), 0)
=> fieldname 변수가 "code11" 문자열인 경우의 카운트를 하기 위한 요약문이지만 마지막 레코드의 fieldname의 값이 "code11" 아니면 최종값은 0으로 출력됩니다. 따라서 아래의 예제와 같이 사용해야 합니다.
```

```
sum(iif(fieldname == "code11", 1, 0))
=> fieldname 변수가 "code11" 문자열인 경우에만 카운트를 하도록 요약문을 수정했습니다.
```

참고

SUM(), COUNT(), ALLDATASUM(), SUBALLDATASUM(), IIFEX()

1.2.43. IIFEX()

문법

IIFEX(condition, op1, op2)

인수

condition
조건식

op1
조건식이 '참'일 때 실행되는 연산식

op2
조건식이 '거짓'일 때 실행되는 연산식

설명

특정 조건에 따라 서로 다른 연산식을 수행합니다. condition 의 조건식이 '참'일 때, op1 연산식을 수행하고 '거짓'일 때, op2 연산식을 수행합니다. 연산식에는 다시 iifex() 함수를 재귀적으로(recursive) 사용할 수 있습니다.

주의) 값을 누적하는 요약함수-avg,avgex,count,distinct,grpcount,max,min,rank,sum,xsum 등-와 함께 사용하는 경우 iif 와 iifex 는 다르게 동작합니다. iif 는 op1 연산식과 op2 연산식 중 하나만 수행하지만, iifex 는 op1 연산식과 op2 연산식 모두를 수행합니다.

예

```
iifex(sum(a3) == 0, sum(a2), sum(a2)/sum(a3))
```

참고

IIF()

1.2.44. ISFIRSTROW()

문법

ISFIRSTROW()

인수

없음

설명

반복부 또는 요약부가 헤더부를 제외하고 표의 페이지별 첫번째 행에 나왔을때, "1"을 리턴합니다.

다단표는 각 다단의 첫번째 행에 나올때 "1"을 리턴합니다

주의) 표문서의 데이터셋 연결 대화상자의 요약작성에서만 사용할 수 있습니다.

(페이지연결조건 등에서 사용못합니다.)

텍스트 상자 또는 크로스탭 표에는 사용할 수 없습니다.

요약작성시 sum(), avg(), count() 등과 같이 값을 누적하는 함수와는 함께 사용할 수 없습니다.

요약부 출력옵션을 ""반복부 앞에 삼입"" 으로 지정한 경우에도 사용할 수 없습니다.

예

```
iif(isfirstrow() == 1, companyName & " 소계", "소계")
=> 반복부 또는 요약부가 표의 페이지별 첫번째 행에 나왔을때는 companyName 컬럼의 데이터와 "소계" 라는 문자열이 나옴. 첫번째 행이 아닌 경우에는 "소계" 라는 문자열만 나옵니다.
```

참고

IIF()

1.2.45. ISGRPNEXTPAGE()

문법

ISGRPNEXTPAGE()

인수

없음

설명

페이지 넘김 조건에 의해 페이지 넘김 발생시, 페이지 넘김 조건 그룹 단위로 최종 페이지인가, 동일 조건에 연속된 페이지인가를 판단합니다.

0 : 최종 페이지

1 : 뒤에 연속해서 출력될 페이지가 있음

* 표 형식의 보고서만 지원하며, 서브 페이지와 서브 레포트는 지원하지 않습니다.

예

isgrpNextpage()

1.2.46. ISGRPPREVPAGE()

문법

ISGRPPREVPAGE()

인수

없음

설명

페이지 넘김 조건에 의해 페이지 넘김 발생시, 페이지 넘김 조건 그룹 단위로 선두 페이지인가, 동일 조건에 연속된 페이지인가를 판단합니다.

0 : 선두 페이지

1 : 앞 페이지로부터 연속된 페이지

* 표 형식의 보고서만 지원하며, 서브 페이지와 서브 레포트는 지원하지 않습니다.

예

isgrpprevpage()

1.2.47. ISLASTRECORD()

문법

ISLASTRECORD(opt)

인수

opt

주 데이터인지 서브페이지 데이터인지를 구분하는 숫자

0 : 주 데이터

1 : 서브페이지 데이터

설명

현재 처리중인 레코드가 마지막 레코드인지를 나타냅니다. 마지막 레코드이면 1 을 리턴하고 아니면 0 을 리턴합니다.

예

```
iif(islastrecord(0), deleterow(), strvalue)
```

=> 마지막 레코드이면 적용된 셀을 포함하는 표의 행을 지우고 그렇지 않으면 셀에 strvalue 값을 적용합니다.

참고

CURRECINDEX(), LASTRECINDEX()

1.2.48. ISNUM()

문법

ISNUM(str)

인수

str
정수형 문자

설명

숫자(정수)형 문자인지 아닌지를 판별하여 리턴합니다.
판단기준은 0 ~ 9 까지의 문자일 경우 1 을 리턴하고 그 외의 문자가 포함되어 있을 경우 0 을 리턴합니다

예

```
isnum("1218")  
=> 1 반환  
isnum("1.23")  
=> 0 반환  
isnum("-1.929")  
=> 0 반환  
isnum("Love79")  
=> 0 반환
```

1.2.49. ISVALIDDATE()

문법

ISVALIDDATE(str, opt)

인수

str

검사 대상 문자열

opt

검사 옵션

0 : 년(4 자리) + 월(2 자리) + 일(2 자리). 날짜의 정합성 체크

1 : 년(4 자리) + 월(2 자리). 년은 4 자리 숫자, 월은 01~12 까지의 숫자

2 : 년(4 자리). 년은 4 자리 숫자

3 : 월(2 자리). 월은 01~12 까지의 숫자

4 : 일(2 자리). 일은 01~31 까지의 숫자

설명

날짜값인지 검사옵션에 따라 판별하여 리턴합니다

예

```

isvaliddate("2009103", 0)
=> 0 (월(2 자리) 형식이 아니어서 날짜형식에 맞지 않습니다)
isvaliddate("20090230", 0)
=> 0 (2월 30일은 날짜의 정합성이 맞지 않습니다)
isvaliddate("200900", 1)
=> 0 (월은 01~12 까지의 숫자. 00은 날짜형식에 맞지 않습니다)
isvaliddate("200911", 2)
=> 0 (년은 4 자리 숫자. 날짜형식에 맞지 않습니다)
isvaliddate("21", 3)
=> 0 (월은 01~12 까지의 숫자. 날짜형식에 맞지 않습니다)
isvaliddate("50", 4)
=> 0 (일은 01~31 까지의 숫자. 날짜형식에 맞지 않습니다)
isvaliddate("20090310", 0)
=> 1 (날짜형식에 맞습니다)

```

1.2.50. ITOA()

문법

ITOA(fieldname)

인수

fieldname

문자열로 변환할 숫자 또는 숫자형 변수

설명

숫자형을 문자형으로 변환합니다. 단 소숫점 이하 부분은 버립니다.

예

`itoa(1234.56)`

=> 숫자 1234.56 를 문자열 "1234"로 변환

참고

ATOI()

1.2.51. JULIANTODATE()

문법

JULIANTODATE(julian)

인수

julian

Julian 값을 나타내는 숫자 또는 숫자형 변수

설명

Julian 으로 표현된 숫자형 데이터를 날짜형으로 변환합니다. julian 값이 0 인 경우, '1899 년 12 월 30 일' 날짜형으로 변환됩니다. 다음 숫자인 1 은 다음 날짜인 '1899 년 12 월 31 일' 로 변환되어 표시됩니다.

convdate() 함수도 숫자형 데이터를 날짜형으로 변환하지만, convdate()는 19990101 과 같은 숫자 형식을 날짜형으로 변환할 때 사용합니다.

예

```
juliantodate(34157)
=> 1993-07-07 의 DATE 형값 반환

juliantodate(datetojulian(date()) - 693594)
=> 오늘 날짜 반환
```

참고

CONVDATE()

1.2.52. KEY2VALUE()

문법

KEY2VALUE(“ keyname” , “ section name” , “ picking file path”)

인수

keyname

검색할 키명

section name

데이터의 그룹명

picking file path

데이터의 정보를 포함하고 있는 픽킹 파일 경로 (웹 경로, 로컬 경로, 상대 경로 지원)

설명

key=value 데이터 정보를 포함하고 있는 픽킹 파일을 읽어 들여, 지정 Key 에 해당하는 값을 취득합니다.

* 픽킹 파일 작성 방법

- 1) 정의 파일 생성(mytable.ini)
- 2) Section, Key, value 작성

```
[section_name01]
```

```
key01 = XXX.jpg
```

```
key01 = XXX.jpg
```

```
[section_name02]
```

```
key01 = XXX
```

```
key02 = XXX
```

예

```
Key2value(“ 0100” , “ emp_name” , “ http://myserver/mytable.ini” )
loadimage(key2value(“ 0100” , “ emp_photo” , “ http://myserver/mytable.ini” )
```

1.2.53. LASTRECIINDEX()

문법

LASTRECIINDEX(opt)

인수

opt

주 데이터인지 서브페이지 데이터를 구분하는 숫자

0 : 주 데이터

1 : 서브페이지 데이터

설명

해당쿼리의 결과레코드 중에서 마지막 인덱스값을 리턴합니다. 레코드의 개수를 구할 때 사용할 수 있습니다.

예

```
iif(currecindex() == (lastrecindex()-1), deleterow(), strvalue)
```

=> 현재 처리하고 있는 레코드의 위치가 마지막레코드의 바로 이전레코드이면 표의 행을 삭제하고 그렇지 않으면 strvalue 를 셀에 적용합니다.

참고

CURRECIINDEX(), ISLASTRECORD()

1.2.54. LEFT()

문법

LEFT(fieldname, index)

인수

fieldname
필드명

index
데이터의 선두로부터 취득할 바이트 수

설명

선두부터 지정한 바이트 수만큼, 문자를 취득합니다.

Fieldname 에 취득할 데이터가 들어 있는 문자형 필드명을 지정합니다.

index 에 취득한 바이트 수를 숫자로 지정합니다.

지정된 fieldname 에 들어 있는 문자형 데이터의 선두부터 index 바이트 수만큼, 취득한 결과가 출력됩니다.

예

문자형 필드 S1 에 "東京都渋谷区"라는 데이터가 들어 있는 경우

```
LEFT(S1, 6)
=>결과값 : "東京都"
```

1.2.55. LEN()

문법

LEN(fieldname)

인수

fieldname
필드명

설명

문자형 데이터의 바이트 수를 숫자형 데이터로 취득합니다.

Report Designer 일본어 버전의 경우, 유니코드 데이터로 읽어 들인 후, Ansi 데이터로 변환하여 바이트 수를 계산합니다.

전각 히라카나, 가타카나, 영문, 숫자 : 2 바이트
반각 히라카나, 가타가나, 영문, 숫자 : 1 바이트

예

len(" abcd") => 결과값 : 4 len(" abc 文字 d") => 결과값 : 8
--

1.2.56. LOADHTML()

문법

LOADHTML(path)

인수

path

html 파일이 위치하는 경로를 나타내는 문자열 또는 문자열 변수
로컬 PC의 경로, HTTP URL 경로, 상대 경로 등으로 지정할 수 있음

설명

이미지 파일을 표시합니다. 데이터로 이미지 파일의 위치가 저장되어 있을 때 주로 사용하며 path 경로에 있는 이미지 파일을 읽어서 이 요약함수가 적용된 객체에 표시합니다.

예

```
loadhtml("test.htm")  
loadhtml("c:/test.htm")  
loadhtml("http://m2soft.co.kr/report/bmr/test.htm")
```

1.2.57. LOADIMAGE()

문법

LOADIMAGE(path)

인수

path

이미지 파일이 위치하는 경로를 나타내는 문자열 또는 문자열 변수
로컬 PC의 경로, HTTP URL 경로, 상대 경로 등으로 지정할 수 있음

설명

이미지 파일을 표시합니다. 데이터로 이미지 파일의 위치가 저장되어 있을 때 주로 사용하며
path 경로에 있는 이미지 파일을 읽어서 이 요약함수가 적용된 객체에 표시합니다.

예

```
loadimage(employee.imagepath)
```

```
loadimage("http://www.m2soft.co.kr/testimage.jpg")
```

1.2.58. LOADIMAGE2()

문법

LOADIMAGE2(path, XResolution, YResolution)

인수

path

이미지 파일이 위치하는 경로를 나타내는 문자열 또는 문자열 변수
로컬 PC의 경로, HTTP URL 경로, 상대 경로 등으로 지정할 수 있음

XResolution

X 축 해상도(X-Dpi)

YResolution

Y 축 해상도(Y-Dpi)

설명

LOADIMAGE의 확장입니다. 이미지파일의 해상도를 지정할 수 있습니다.

예

```
loadimage2(employee.imagepath,100,200)
```

```
loadimage2("http://www.m2soft.co.kr/testimage.jpg",300,300)
```

1.2.59. LOADSWF()

문법

LOADSWF(filename)

인수

filename
swf 파일명.

설명

파일명에 해당하는 shockwaveflash 파일을 읽어서 해당하는 객체에 넣습니다.

예

```
loadswf("shockwaveflash.swf")
```

1.2.60. LOADTEXT()

문법

LOADTEXT(keyname, defaultstr)

인수

keyname
리소스키

defaultstr
기본문자열.

설명

지정한 리소스 XML 파일에서 리소스 키(keyname)값을 가져와서 문자열을 매핑합니다.
리소스 키가 XML 파일에 존재 하지 않을 경우 기본 문자열(defaultstr)을 매핑합니다

예

```
loadtext("title", "거래업체목록")  
=> 리소스 키가 있을 경우 '지역별거래 업체목록'으로 없을 경우 '거래업체목록'으로 매핑됩니다  
loadtext("company", "회사")  
=> 리소스 키가 있을 경우 '업체명'으로 없을 경우 '회사'로 매핑됩니다
```

1.2.61. LOCALTIME()

문법

LOCALTIME(fieldname)

인수

fieldname

UTC 를 나타내는 숫자 또는 숫자형 변수

1970 년 1 월 1 일, 0 시정각을 0 으로 해서 초마다 1 씩 증가된 숫자.

설명

숫자로 된 UTC(universal time coordinated) 값을 날짜/시간 형태로 변환해주는 함수입니다. 보통 PC 등에 저장되어있는 날짜/시간에 해당하는 값이 UTC 값이며 이를 알아보기 편하도록 변환해주는 함수이다. `gmtime()` 도 비슷한 기능을 하나 `localtime()`은 현재 설정되어있는 표준시간대로 시간을 변경시키고 `gmtime()`은 GMT 시간으로 표시합니다.

예

```
localtime(393455871)
```

```
=> 지역이 한국/일본(GMT+09:00)인 경우, 결과값: 1982-06-21 06:17:51
```

참고

GMTIME()

1.2.62. LOG()

문법

LOG(num)

인수

num

자연로그 값을 구할 숫자 또는 숫자형 변수
0 이 되면 안됨

설명

자연로그 값을 구합니다. 일반적인 사용보다는 수학적 통계나 수치의 연산을 위해서 사용됩니다.

예

log(2.717)

참고

LOG10()

1.2.63. LOG10()

문법

LOG10(num)

인수

num

밑이 10 인 로그 값을 구할 숫자 또는 숫자형 변수
0 이 되면 안됨

설명

밑이 10 인 로그 함수의 값을 구합니다. 일반적인 사용보다는 수학적 통계나 수치의 연산을 위해서 사용합니다.

예

log10(b/3.14)

참고

LOG()

1.2.64. MAX()

문법

MAX(fieldname)

인수

fieldname
최대값을 구할 숫자형 변수

설명

fieldname 인수의 최대값을 구합니다. 반복되어 나오는 변수의 값 중에서 최대값을 구하는데 사용됩니다.

예

```
max(salary)
```

참고

MIN(), AVG(), AVGEX()

1.2.65. MAXEX()

문법

MAXEX(fieldname)

인수

fieldname
필드명

설명

필드명이나 변수명에 해당하는 값들의 최대값을 구합니다.
(단, 값이 들어있는 경우의 필드들만 해당됩니다.)

예

```
maxex(prodname)
```

참고

MAX(), MIN(), MINEX()

1.2.66. MERGECELLEX()

문법

MERGECELLEX(fieldname, showdataopt)

인수

fieldname

셀 병합을 적용할 원본 문자열 또는 문자열 변수

showdataopt

0 – 병합된 셀이 다음 페이지까지 넘어가 표시되는 경우, 다음 페이지의 병합된 셀에는 데이터를 표시하지 않음.

1 – 병합된 셀이 다음 페이지까지 넘어가 표시되는 경우, 다음 페이지의 병합된 셀에도 데이터를 표시함.

설명

[데이터 셋 연결] – [셀 병합] 옵션이 적용된 셀에 사용해야 하며, 셀이 병합된 여러 행의 높이를 균등하게 맞추고자 할 때 사용합니다.

참고)

- ① mergecellex()를 적용한 셀에는 ‘셀 크기 고정’ 옵션이 적용되지 않습니다.
- ② mergecellex()를 적용한 셀은 ‘옆 셀 병합’ 옵션이 적용되지 않습니다.
- ③ mergecellex()는 다른 요약 함수에 포함하여 사용할 수 없으며, 최상위에 위치해야 합니다. (예를 들어, iif() 안에 사용하면 안 됨.)
- ④ mergecellex()가 적용되어 병합되는 셀의 데이터가 같은 경우에만 셀 높이가 늘어나지 않고 병합됩니다.

예

```
mergecellex(prodname, 1)
```

1.2.67. MID()

문법

MID(fieldname, startposition, index)

인수

fieldname
필드명

startposition
취득 시작 바이트 수 (선두 : 1)

index
취득할 바이트 수

설명

문자형 데이터의 시작 바이트 위치부터 지정한 바이트 수만큼 취득합니다.

Fieldname 에 취득할 데이터가 들어 있는 문자형 필드명을 지정합니다.
startposition 에 취득할 데이터의 시작 바이트 위치를 숫자로 지정합니다.
index 에 시작 바이트 위치부터 취득할 바이트 수를 숫자로 지정합니다.

예

문자형 필드 S1 에 "東京都渋谷区"라는 데이터가 들어 있는 경우

MID(S1, 7, 4)
=>결과값 : "渋谷"

1.2.68. MIN()

문법

MIN(fieldname)

인수

fieldname
최소값을 구할 숫자형 변수

설명

fieldname 인수의 최소값을 구합니다. 반복되어 나오는 변수의 값 중에서 최소값을 구하는데 사용됩니다.

예

```
min(salary)
```

참고

MAX(), AVG(), AVGEX()

1.2.69. MINEX()

문법

MINEX(fieldname)

인수

fieldname
필드명

설명

필드명이나 변수명에 해당하는 값들의 최소값을 구합니다.
(단, 값이 들어있는 경우의 필드들만 해당됩니다.)

예

```
minex(prodname)
```

참고

MAX(), MIN(), MAXEX()

1.2.70. MOD()

문법

MOD(val1, val2)

인수

val1
val2 로 나눌 값

val2
val1 을 나누는 값

설명

val1 값을 val2 값으로 나눈 나머지 값을 구합니다.

예

```
mod(numdata/2)
```

참고

DIV()

1.2.71. NUMTOARAB()

문법

NUMTOARAB(num, opt)

인수

num
아랍어 문자열로 바꿀 숫자

opt
0.
opt 값은 향후 확장성을 고려하여 넣은 값으로 현재는 0으로 넘깁니다.

설명

숫자를 숫자에 해당하는 아랍어 문자열로 바꿔줍니다.

예

```
numtoabab(1230, 0)
```

참고

NUMTOKOR(), NUMTOVIT(), NUMTOENG(), NUMTOVITEX()

1.2.72. NUMTOENG()

문법

NUMTOENG(num, opt)

인수

num

영어 문자열로 바꿀 숫자

opt

0 또는 1. 소수점 자리 표시 옵션.

숫자를 영어 문자열로 바꿀 때 소수점 자리까지 표시할 것인지 여부. 값이 1 인 경우 소수점 자리까지 표시함.

설명

숫자를 숫자에 해당하는 영어 문자열로 바꿔줍니다.

예

```
numtoeng(151, 0) => "one hundred one"
```

```
numtoeng(1000.90, 1) => "one thousand and 90/100"
```

참고

NUMTOKOR(), NUMTOVIT(), NUMTOARAB(), NUMTOVITEX()

1.2.73. NUMTOKOR()

문법

NUMTOKOR(num)

인수

num
한글 문자열로 바꾸어줄 숫자 또는 숫자형 변수

설명

숫자를 한글 문자열로 바꿔 줍니다. 인수로 오는 숫자는 소수점이하는 처리하지 않습니다.

예

```
numtokor(1501)
=> 결과값: "일천오백일"
```

참고

NUMTOENG(), NUMTOVIT(), NUMTOARAB(), NUMTOVITEX()

1.2.74. NUMTOTIME()

문법

NUMTOTIME(num, opt)

인수

num

시간 서식으로 바꿀 숫자

opt

의미 없는 [0]값 표시 여부

0 : 의미 없는 [0]값을 표시 한다. 예)06:30

1 : 의미 없는 [0]값을 표시 하지 않는다. 예)6:30 설명

설명

숫자(분 데이터)를 시간 서식으로 바꿔줍니다.

예

```
numtotime(350,0)=> 6:50
```

```
numtotime(350,1)=> 06:50
```

1.2.75. NUMTOVIT()

문법

NUMTOVIT(num, opt)

인수

num

베트남어 문자열로 바꿀 숫자

opt

0 또는 1. 소수점 자리 표시 옵션.

숫자를 베트남어 문자열로 바꿀 때 소수점 자리까지 표시할 것인지 여부. 값이 1 인 경우 소수점 자리까지 표시함.

설명

숫자를 숫자에 해당하는 베트남어 문자열로 바꿔줍니다.

예

```
numtovit(151, 0) => "mot tram nam moui mot"
```

```
numtovit(1000.90, 1) => "mot ngan cham chin moui"
```

참고

NUMTOKOR(), NUMTOENG(), NUMTOARAB(), NUMTOVITEX()

1.2.76. NUMTOVITEX()

문법

NUMTOVITEX(num, opt)

인수

num

베트남어 문자열로 바꿀 숫자

opt

0 또는 1. 소수점 자리 표시 옵션.

숫자를 베트남어 문자열로 바꿀 때 소수점 자리까지 표시할 것인지 여부. 값이 1 인 경우 소수점 자리까지 표시함.

설명

숫자를 숫자에 해당하는 베트남어 표준 문자열로 바꿔줍니다.

예

```
numtovit(151, 0) => "mot tram nam moui mot"
```

```
numtovit(1000.90, 1) => "mot ngan cham chin moui"
```

참고

NUMTOKOR(), NUMTOENG(), NUMTOARAB(), NUMTOVIT()

1.2.77. PAGE()

문법

PAGE()

인수

없음

설명

page 번호를 표시하는 함수입니다.

보고서를 작성하는 시점에서의 **page** 번호를 표시하는 함수입니다. 일반적으로 페이지번호를 표시하는 경우, 이 요약함수를 이용하지 않고 텍스트 상자 속성의 페이지 객체를 설정해서 사용합니다. 이 함수는 보통 텍스트 상자에 적용하며 표문서에서 표의 반복부나 요약부에 사용하는 경우, 요약함수가 처리되는 시점에 따라 값이 틀리게 나올 수도 있습니다.

예

PAGE()

1.2.78. PERIOD()

문법

PERIOD(sdate, edate, opt)

인수

sdate

시작일을 나타내는 날짜 또는 날짜형 변수. 날짜를 나타내는 문자열도 가능

edate

종료일을 나타내는 날짜 또는 날짜형 변수. 날짜를 나타내는 문자열도 가능

opt

결과값을 어떠한 문자열로 표현할 것인지에 대한 숫자 또는 문자열

0 : 결과값을 “yyyy/mm/dd” 형태로 리턴

1 : 결과값을 “mm/dd” 형태로 리턴 - 년을 모두 월로 바꾸어 계산

2 : 결과값을 “dd” 형태로 리턴 - 년과 월을 모두 일로 바꾸어 계산

문자열 : yyyy, yy, mm, dd 등의 문자열을 포함.

설명

특정 날짜 사이의 기간을 구합니다. 시작날짜 및 종료날짜에는 날짜형의 상수와 변수가 올 수 있으며 문자열도 올 수 있는데, “19980514”, “1998/05/14”의 형태가 가능합니다. 옵션은 숫자 또는 문자열이 올 수 있으며 period 가 리턴하는 문자열의 형태를 나타냅니다.

예

```
period(“1997/5/14”, “1998/5/18”, 0)
```

```
=> 결과값: “0001/00/04”
```

```
period (“1997/5/14”, “1998/5/18”, 1)
```

```
=> 결과값: “12/04”
```

```
period (“1997/5/14”, “1998/5/18”, 2)
```

```
=> 결과값: “369”
```

```
period (“1997/5/14”, “1998/5/18”, “yyyy 년 mm 개월”)
```

```
=> 결과값: “1 년 0 개월”
```

```
period (“1997/5/14”, “1998/5/18”, “yy 살”)
```

```
=> 결과값: “1 살”
```

```
period (“1997/5/14”, “1998/5/18”, “mm 개월”)
```

```
=> 결과값: “0 개월”
```

```
=> 지금과 같이 문자열에 mm 만 적용하게되면 년도에 해당되는 값이 계산에 반영되지 않으므로 주의하십시오.
```

period (“1997/5/14”, “1998/5/18”, “dd 일”)

=> 결과값: “4 일”

=> 지금과 같이 문자열에 dd 만 적용하게되면 년도 및 월에 해당되는 값이 계산에 반영되지 않으므로 주의하십시오.

1.2.79. POW()

문법

POW(num, exp)

인수

num

지수값을 계산할 때 밑(base)을 나타내는 숫자 또는 숫자형 변수

exp

지수값을 계산할 때 지수(exponent)를 나타내는 숫자 또는 숫자형 변수

설명

지수값을 계산합니다. 10의 제곱과 같은 수치를 계산할 때 사용합니다. num이 밑(base)이 되고 exp가 지수(exponent)가 됩니다. num이 0이면서 exp가 음수가 올 수 없습니다.

예

```
pow(3,5), pow(b/3, a/4)
```

참고

SQRT()

1.2.80. RANK()

문법

RANK(fieldname)

인수

fieldname

순위를 구할 때 사용하는 숫자형 변수

설명

순위를 계산하는데 사용되는 함수입니다. 적용된 필드 순으로 레코드가 반드시 정렬되어 있어야 합니다. 만약, 레코드가 정렬되어있지 않거나 그룹별 순위, 요약부에서의 순위 등의 기능이 필요하다면 `rankex()` 함수를 사용해야 합니다.

예

```
rank(salary)
```

참고

RANKEX()

1.2.81. RANKEX()

문법

RANKEX(fieldname, reference, opt)

인수

fieldname

순위를 구할 숫자형 변수

reference

그룹별로 순위를 적용할 경우에 사용하는 참조할 데이터의 변수. 그룹적용을 안하는 경우는 "" (빈 문자열)을 주면 됨

opt

주 데이터/서브데이터, 오름차순/내림차순, 요약부에서의 사용 등을 나타내는 숫자

- 1 - 주 데이터 & 오름차순
- 2 - 주 데이터 & 내림차순
- 3 - 서브 데이터 & 오름차순
- 4 - 서브 데이터 & 내림차순
- 11 - 요약부 & 주 데이터 & 오름차순
- 12 - 요약부 & 주 데이터 & 내림차순
- 13 - 요약부 & 서브 데이터 & 오름차순
- 14 - 요약부 & 서브 데이터 & 내림차순

설명

순위를 계산하는데 사용합니다. 기존 **rank()** 함수와는 달리 데이터가 순서대로 정렬되어 오지 않더라도 순위를 알아오는 기능을 수행합니다. 그룹별로 따로 순위를 매기는 것도 가능하며 요약부 사이의 순위를 매기는 기능도 **opt** 인수를 통해서 처리합니다. 요약부에서의 순위를 얻기 위해서는 **opt**의 값이 11~14 사이여야 합니다.

예

```
rankex(salay, dept, 1)
=> salay 가 직원의 월급이고 dept 가 부서명을 나타낸다면 각 부서별로 월급의 순위를 오름차순으로 나타내는 기능을합니다.
```

참고

RANK()

1.2.82. REPLACESTR()

문법

REPLACESTR(fieldname, str1, str2)

인수

fieldname

원본 문자열 또는 문자열 변수

str1

찾을 문자열

str2

찾은 문자열을 대체할 문자열

설명

fieldname 의 문자열에서 str1 문자열을 찾아서 str2 문자열로 바꾸어줍니다.

예

```
replacestr("Seoul, Korea", "Seoul", "Busan")  
=> 결과값: "Busan, Korea"
```

1.2.83. RIGHT()

문법

RIGHT(fieldname, index)

인수

fieldname
필드명

index
데이터의 뒤에서부터 취득할 바이트 수

설명

데이터의 뒤에서부터 지정한 바이트 수만큼의 문자를 취득합니다.

Fieldname 에 취득할 데이터가 들어있는 문자형 데이터 필드명을 지정합니다.

index 에 뒤에서부터 취득할 바이트 수를 숫자로 설정합니다.

지정된 fieldname 에 들어있는 문자형 데이터에서, 뒤에서부터 index 바이트만큼 취득한 결과가 출력됩니다.

예

```
right("東京都渋谷区", 6)
```

```
=> 결과값 : "渋谷区"
```

```
right("東京 A12", 5)
```

```
=> 결과값 : "京 A12"
```

1.2.84. ROUND()

문법

ROUND(num, pos)

인수

num

반올림할 숫자 또는 숫자형 변수

pos

반올림할 위치를 나타내는 숫자 (-9 ~ 9)

-1 : 십자리 미만 반올림

0 : 소수점 미만 반올림

1 : 소수 1 자리 미만 반올림

설명

num 인수에 해당하는 값을 자리 수에 따라 반올림합니다. 자리 수는 -9 에서 9 까지 유효합니다.

예

i 값이 765.267 인 경우

round(i, 0)

=> 결과값: 765 (소수점 미만 반올림)

round(i, 2)

=> 결과값: 765.27 (소수 2 자리 미만 반올림)

round(i,-1)

=> 결과값: 770 (십 자리 미만 반올림)

참고

CEIL(), TRUNC()

1.2.85. RSUBSTR()

문법

RSUBSTR(str, pos, num)

인수

str

일부 문자열을 뽑아낼 원본 문자열 또는 문자열 변수

pos

원본 문자열의 마지막을 기준으로 하는 위치를 나타내는 숫자
마지막 위치는 1 부터 시작

num

뽑아낼 문자열의 길이를 나타내는 숫자

설명

문자열내에서 특정위치의 문자열을 구합니다. str 문자열의 끝을 기준으로 pos 위치에서부터 num 개수 만큼 좌측 문자열을 발체합니다.

주의) 유니코드 버전에서는 문자의 위치 및 길이를 각 문자의 개수와 연관되며 일반 버전에서는 바이트 수와 연관됩니다. 한글 한 글자인 경우, 유니코드 버전에서는 길이가 1 이지만 일반 버전에서는 2 가 됩니다.

예

```
rsustr("BlueSky",4,4)
=> 결과값: "Blue"

rsustr("대한민국서울",3,4)
=> 유니코드 버전에서의 결과값: "대한민국"
=> 일반 버전에서의 결과값: "국서"
```

참고

SUBSTR()

1.2.86. SETGLOBAL()

문법

SETGLOBAL(varname, value)

인수

varname

전역변수명을 나타내는 문자열

value

저장할 값

설명

전역변수에 값을 저장합니다. 저장된 전역변수 값을 가져오려면 `getglobal()` 함수를 사용합니다.

`setglobal()` 과 `getglobal()` 함수는 특정 텍스트 상자의 요약함수를 통해서 나타난 결과를 다른 쪽에서 이용하여 또 다른 표현을 하고자 할 때 이용합니다. 서브페이지가 여러 개 있는 경우, 앞의 서브페이지에서 계산한 합계를 뒤에 오는 서브페이지에서 사용하고자 할 때가 대표적인 예입니다.

예

```
setglobal("myTempValue", "m2soft")  
=> myTempValue 전역변수의 값으로 m2soft 설정
```

참고

GETGLOBAL()

1.2.87. SETSTRINGFIT()

문법

SETSTRINGFIT(fieldname, varname, option)

인수

fieldname
필드명

varname
global 변수명

option
옵션(현재는 지원 안함, 차후 제공 예정)

설명

표의 셀 또는 텍스트 상자 객체의 사이즈 보다 출력되는 문자열이 긴 경우, 출력되지 못한 문자열을 Global 변수에 저장합니다.

setstringfit() 요약함수가 적용된 셀 또는 텍스트 상자는 텍스트 상자 속성의 "셀 크기 고정" 옵션이 자동 설정됩니다.

예

```
setstringfit(description, "remainder", "")
=> description 이란 필드값을 출력하고, 출력하지 못한 문자열은 remainder 라는 global 변수에 저장
```

1.2.88. SIN()

문법

SIN(arg)

인수

arg
radian 형태의 숫자 또는 숫자형 변수

설명

삼각함수 중에서 sine(사인) 값을 구하는 함수입니다.

예

```
sin(3.141592/2)
```

참고

COS(), TAN()

1.2.89. SQRT()

문법

SQRT(num)

인수

num

제곱근을 구할 숫자 또는 숫자형 변수.
num 은 0 보다 크거나 같음

설명

제곱근 (root)를 구하는 함수입니다. num 은 음수가 아니어야 합니다.

예

```
sqrt(2.4)
```

```
sqrt(sum(val))
```

참고

POW()

1.2.90. STRCAT()

문법

```
STRCAT(str1, str2)
```

인수

str1

이어진 문자열에서 앞 부분에 나올 문자열 또는 문자열 변수

str2

이어진 문자열에서 뒷 부분에 나올 문자열 또는 문자열 변수

설명

두 개의 문자열을 이어 하나의 문자열로 만들어 줍니다. '&' 연산자와 같은 기능을합니다. 여러 개의 문자열을 동시에 이어주는 경우, `strcat()` 함수를 사용하는 것보다 '&' 연산자를 사용하는 것이 더 편리합니다.

예

```
strcat("abc", b)

strcat ("m2soft ", "Report Designer")

strcat ("m2soft", strcat (b, strcat (c, "Report Designer")))
```

참고

& 연산자

1.2.91. STRLEN()

문법

STRLEN(str)

인수

str
길이를 구할 문자열 또는 문자열 변수

설명

문자열 길이를 구하는 함수입니다.

주의) 유니코드 버전에서는 각 문자의 개수로 문자열의 길이를 구하며 일반 버전에는 문자열의 바이트 수로 문자열의 길이를 구합니다. 따라서 같은 문자열이라도 유니코드 버전인지 아닌지에 따라서 길이가 달라질 수 있습니다.

예

```
strlen("ABCDEFGHIKL")  
=> 결과값: 11  
  
strlen("한국")  
=> 유니코드 버전 결과값: 2  
=> 일반 버전의 결과값: 4
```

1.2.92. STRMASK()

문법

```
STRMASK(str, mask)
```

인수

str
바뀌어야 할 원본 문자열 또는 문자열 변수

mask
어떻게 변환할 것인지를 나타내는 문자열

설명

str의 문자열을 마스크(**mask**)를 씌워 다른 문자열로 변환시킵니다. **substr()**, **strcat()** 등의 함수를 조합해서 사용하는 불편함을 해소할 수 있습니다.

mask는 문자열에 'x' 문자를 포함하고 있으며 각각의 'x' 문자가 원본 문자열에서의 한 문자와 대응되며 해당부분을 치환해 줍니다. 아래의 예제를 보면 쉽게 알 수 있습니다.

예

```
strmask("123456", "xxx-xxx")
=> 결과값: "123-456"

strmask("123456", "xxx-xxx-x-")
=> 결과값: "123-456- -"

strmask("123456", "xx-xx")
=> 결과값: "12-34"
```

참고

SUBSTR(), STRCAT()

1.2.93. SUBALLDATASUM()

문법

SUBALLDATASUM(fieldname, reference, opt)

인수

fieldname

소계를 구할 숫자형 변수

reference

그룹별로 소계를 계산하기 위해 사용하는 참조할 데이터의 변수

opt

주 데이터와 서브 데이터의 구분을 나타내는 숫자

0 : 주 데이터

1 : 서브 데이터

설명

소계를 구합니다. **reference** 의 값이 바뀔 때까지의 **fieldname** 값의 합계를 구합니다.

주의) 크로스탭을 적용한 보고서에서는 사용할 수 없습니다.

예

```
suballdatasum(unitprice, categoryid, 0)
```

참고

ALLDATASUM(), SUM()

1.2.94. SUBSTR()

문법

SUBSTR(str, pos, num)

인수

str

일부 문자열을 뽑아낼 원본 문자열 또는 문자열 변수

pos

원본 문자열에서의 위치를 나타내는 숫자
첫 위치는 1 부터 시작

num

뽑아낼 문자열의 길이를 나타내는 숫자

설명

문자열내에서 특정위치의 문자열을 구합니다. str 문자열의 pos 위치에서부터 num 개수 만큼을 발채합니다.

주의) 유니코드 버전에서는 문자의 위치 및 길이를 각 문자의 개수와 연관되며 일반 버전에서는 바이트 수와 연관됩니다. 한글 한 글자인 경우, 유니코드 버전에서는 길이가 1 이지만 일반 버전에서는 2 가 됩니다.

예

```
substr("BlueSky",5,3)
=> 결과값: "Sky"

substr("대한민국서울",5,2)
=> 유니코드 버전에서의 결과값: "서울"
=> 일반 버전에서의 결과값: "민"
```

참고

RSUBSTR()

1.2.95. SUM()

문법

SUM(fieldname)

인수

fieldname

합계를 구할 숫자 또는 숫자형 변수

설명

반복해서 나오는 변수의 합을 구합니다. 이 함수를 표의 반복부에서 사용하게 되면 각 반복부마다 해당변수의 값을 누적해서 볼 수 있습니다. 또한 마지막 페이지에 나오는 요약부에서 사용하면 총합계를 조건을 줘서 중간에 나오는 요약부에서 사용하면 소계를 구할 수 있습니다.

소계 등을 나타내는 중간에 표시하는 요약부에 적용할 경우, 임의로 다른 작업을 하지 않아도 해당 요약부가 나올 때 마다 값이 리셋됩니다.

alldatasum() 또는 suballdatasum() 과의 차이점은 alldatasum()은 반복부에서 사용하더라도 누적이 되면서 보이는게 아니라 최종합계나 최종소계를 나타냅니다.

예

```
sum(salary)
```

```
sum(iif(deptcode == "AA01", 1, 0))
```

=> iif() 함수와 조합해서 사용하여 특정 그룹의 카운트를 할 수 있습니다.

참고

ALLDATASUM(), SUBALLDATASUM()

1.2.96. SYSFORMAT()

문법

SYSFORMAT(fieldname, type, nooverride, opt)

인수

fieldname

특정 서식 문자열로 바꿀 변수. 변수형은 바꿀 서식에 맞게 설정

type

변경할 서식의 형식을 나타냄

- 1 : 숫자
- 2 : 통화
- 3 : 날짜
- 4 : 시간

nooverride

사용자 서식변경 불가를 나타냄

- 0 : 사용자 서식변경 가능
- 1 : 사용자 서식변경 불가

opt

type 이 3(날짜) 또는 4(시간)인 경우에만 적용됨

type 이 3 인경우

- 0 : short date format
- 1 : long date format

type 이 4 인경우(숫자 4 자리로 구성되어 각 자리의 숫자가 플래그를 나타냄)

- 일의 자리 : No minutes or seconds
- 십의 자리 : No seconds
- 백의 자리 : No time maker
- 천의 자리 : 24-hour format

설명

변수명의 값을 시스템 서식 문자열의 형태로 바꾸어 줍니다. 리턴되는 값은 서식 문자열 형식으로 변환된 문자열입니다. 보통 서식맞추기 대화상자에서 대부분의 서식을 모두 설정할 수 있습니다. 서식맞추기 대화상자에서 설정하여 사용하지 않고 이 함수를 사용하는 경우는 시스템 서식을 나타내는 데이터와 다른 데이터를 동시에 한 곳에 표시하고자 할 때입니다. 서식을 문자열로 바꾸어 다른 데이터와 & 연산자 등을 사용하여 문자열을 이어주는게 대표적인 예입니다.

숫자 서식인 경우, **Report Designer** 에서 기본적으로 제공하는 것은 자리수는 (,)로 나타내고 소수점은 (.)으로 나타내게 되는데, 특정 지역 및 국가에서는 자리수는 (.)로 나타내고 소수점은 (,)로 나타냅니다. 이러한 서식의 특징은 윈도우시스템이 가지고 있게 되는데 이렇게 특정 시스템이 가지고 있는 서식의 특징을 그대로 구현하기 위해서 **sysformat()** 함수를 사용합니다.

시스템 서식이 아닌 다른 서식들은 `format()` 함수를 사용하시길 바랍니다.

예

사용자 시스템의 지역설정이 한국인 경우

`sysformat(numval, 1, 1, 1)`

=> 숫자형 데이터 `numval` 가 1234.567 이라면 결과값: 1,234.57

`sysformat(numval, 2, 1, 1)`

=> 숫자형 데이터 `numval` 가 1234.56 이라면 결과값: \1,235

`sysformat(dateval, 3, 0, 0)`

=> dateval 가 '2006년 01월 30일' 이라면 결과값: 2006-01-30

`sysformat(dateval, 3, 0, 1)`

=> dateval 가 '2006년 01월 30일' 이라면 결과값: 2006년 1월 30일 월요일

`sysformat(timeval, 4, 0)`

=> 시간형 데이터 `timeval` 가 '13:13:13' 이라면 결과값: 오후 1:13:13

`sysformat(timeval, 4, 1)`

=> 시간형 데이터 `timeval` 가 '13:13:13' 이라면 결과값: 오후 1

`sysformat(timeval, 4, 10)`

=> 시간형 데이터 `timeval` 가 '13:13:13' 이라면 결과값: 오후 1:13

`sysformat(timeval, 4, 101)`

=> 시간형 데이터 `timeval` 가 '13:13:13' 이라면 결과값: 1

`sysformat(timeval, 4, 1010)`

=> 시간형 데이터 `timeval` 가 '13:13:13' 이라면 결과값: 오후 13:3

참고

FORMAT()

1.2.97. TAN()

문법

TAN(arg)

인수

arg
radian 형태의 숫자 또는 숫자형 변수

설명

삼각함수 중에서 **tangent**(탄젠트) 값을 구하는 함수입니다.

예

```
tan(3.141592/2)
```

참고

SIN(), COS()

1.2.98. TIME()

문법

TIME()

인수

없음

설명

현재 보고서를 실행하고 있는 컴퓨터의 설정된 시간을 구합니다.

일반적으로 이 함수를 사용하는 것 보다는 텍스트 상자 속성의 시간객체를 이용하는 것이 편리합니다.

예

```
time()
```

참고

DATE()

1.2.99. TOLOWER()

문법

TOLOWER(str)

인수

str
소문자로 바꿀 문자열

설명

str 문자열 내의 대문자를 소문자로 바꿔줍니다.

예

```
tolower("AB cd") => "ab cd"
```

참고

TOUPPER()

1.2.100. TOTALPAGENO()

문법

TOTALPAGENO()

인수

없음

설명

전체 페이지 수를 숫자형 데이터로 취득합니다.

* 표 형식의 보고서만 지원하며, 서브 페이지와 서브 레포트는 지원하지 않습니다.

예

totalpageno()

1.2.101. TOUPPER()

문법

TOUPPER(str)

인수

str
대문자로 바꿀 문자열

설명

str 문자열 내의 소문자를 대문자로 바꿔줍니다.

예

```
toupper("AB cd") => "AB CD"
```

참고

TOLOWER()

1.2.102. TRIM()

문법

TRIM(str)

인수

str

좌우 양쪽의 공백을 제거할 문자열 또는 문자열 변수

설명

문자열의 좌우 양쪽의 공백을 지웁니다.

예

```
trim(" test string ")  
=> 결과값: "test string"
```

참고

TRIMLEFT(), TRIMRIGHT()

1.2.103. TRIMLEFT()

문법

TRIMLEFT(str)

인수

str
좌측의 공백을 제거할 문자열 또는 문자열 변수

설명

문자열의 좌측 공백을 지웁니다.

예

```
trimleft(" test string ")  
=> 결과값: "test string "
```

참고

TRIM(), TRIMRIGHT()

1.2.104. TRIMRIGHT()

문법

TRIMRIGHT(str)

인수

str
우측의 공백을 제거할 문자열 또는 문자열 변수

설명

문자열의 우측 공백을 지웁니다.

예

```
trimright(" test string ")  
=> 결과값: " test string"
```

참고

TRIM(), TRIMLEFT()

1.2.105. TRUNC()

문법

TRUNC(num, pos)

인수

num

버림을 할 숫자 또는 숫자형 변수

pos

버림을 할 위치를 나타내는 숫자 (-9 ~ 9)

-1 : 십자리 미만 버림

0 : 소수점 미만 버림

1 : 소수 1 자리 미만 버림

설명

num 인수에 해당하는 값을 자리 수에 따라 버림합니다. 자리 수는 -9 에서 9 까지 유효합니다.

예

i 값이 765.267 인 경우

trunc(i, 0)

=> 결과값: 765 (소수점 미만 버림)

trunc(i, 2)

=> 결과값: 765.26 (소수 2 자리 미만 버림)

trunc(i, -1)

=> 결과값: 760 (십 자리 미만 버림)

참고

CEIL(), ROUND()

1.2.106. TTOC()

문법

TTOC(fieldname)

인수

fieldname
변환할 시간형 데이터

설명

시간형 데이터(HHMMSS)를 문자형 데이터(HH:MM:SS)로 변환합니다.

Fieldname 에 변환할 시간형 데이터가 들어 있는 필드명을 지정합니다.

예

시스템 시간을 문자형 데이터(HH:MM:SS)로 표시하는 경우

TTOC(time())

=> 결과값: "13:20:30"

문자형 필드[필드 1]를 문자형 시간 데이터(HH:MM:SS)로 표시하는 경우

TTOC(필드 1)

=> 결과값: "13:20:30"

1.2.107. VAL()

문법

VAL(arg)

인수

arg
숫자형으로 바꿀 문자열 또는 문자열 변수

설명

문자열을 숫자형으로 변환합니다. atoi() 함수와 같습니다.

예

```
val("1234.56")
```

참고

ATOI(), ITOA()

1.2.108. XSUM()

문법

XSUM(fieldname1, fieldname2)

인수

fieldname1
합계를 구할 숫자 또는 숫자형 변수

fieldname2
합계

설명

필드값의 이전값과 현재값을 비교하여 **fieldname1** 값의 합계를 냅니다. **fieldname1** 과 **fieldname2** 중 하나라도 이전 레코드값과 같은 값을 가지면 해당 **fieldname1** 의 값은 합계에서 제외됩니다.

예

```
xsum(salary, code)
```

=> salary 값과 code 값이 모두 이전 레코드의 값과 다를 때만 salary 값을 더해줍니다.

참고

SUM(), ALLDATASUM()

1.2.109. NUMTO()

문법

NUMTO(num, opt, type)

인수

num

해당 type 의 문자열로 바꿀 숫자

opt

표시 옵션

type

1: 한글(numtokor), 2: 아랍어(numtoarab), 3: 중국어(numtochn), 4: 영어(numtoeng),
5: 시간서식(numtotime), 6: 베트남어(numtovit), 7: 베트남어(numtovitex) 8: 로마

설명

숫자를 해당 타입의 문자열로 바꿔줍니다. 1 ~ 7 까지의 type 은 기존 요약함수들과 동일하게 동작합니다. 추가된 로마 숫자는 opt 가 0 일 때는 대문자로, opt 가 1 일때는 소문자로 출력됩니다.

예

```
numto(12340, 0, 8)
```

```
=> 결과값: "FFMMCCCXL"
```

참고

NUMTOKOR(), NUMTOARAB(), NUMTOCHN(), NUMTOENG(), NUMTOTIME(),
NUMTOVIT(), NUMTOVITEX()

1.2.110. MERGECELLEX2()**문법**

MERGECELLEX2(fieldname, reference fieldname, showdataopt)

인수**fieldname**

셀 병합을 적용할 원본 문자열 또는 문자열 변수

reference fieldname

셀 병합을 적용하기 위한 참조 필드명. 지정한 필드의 값이 같은 경우에만 셀 병합이 적용됨.

showdataopt

0 – 병합된 셀이 다음 페이지까지 넘어가 표시되는 경우, 다음 페이지의 병합된 셀에는 데이터를 표시하지 않음.

1 – 병합된 셀이 다음 페이지까지 넘어가 표시되는 경우, 다음 페이지의 병합된 셀에도 데이터를 표시함.

설명

[데이터 셋 연결] – [셀 병합] 옵션이 적용된 셀에 사용해야 하며, **reference fieldname** 으로 지정한 필드의 데이터가 같은 경우에만 셀 병합이 적용됩니다. 셀이 병합된 여러 행의 높이를 균등하게 맞추고자 할 때 사용합니다.

참고)

- ① mergecellex2()를 적용한 셀에는 ‘ 셀 크기 고정’ 옵션이 적용되지 않습니다.
- ② mergecellex()를 적용한 셀은 ‘ 옆 셀 병합’ 옵션이 적용되지 않습니다.
- ③ mergecellex2()는 다른 요약 함수에 포함하여 사용할 수 없으며, 최상위에 위치해야 합니다. (예를 들어, iif() 안에 사용하면 안 됨.)
- ④ mergecellex2()가 적용되어 병합되는 셀의 데이터가 같은 경우에만 셀 높이가 늘어나지 않고 병합됩니다.

예

```
mergecellex2(prodname, reference, 1)
```