Workday Worksheets

Product Summary

December 10, 2025

This content is not part of the Workday Administrator Guide and is subject to further change.

Contents

Summary of User Guide Changes - Worksheets....14

Getting Started with Worksheets....16
Concept: Introduction to Workday Worksheets....16
Concept: Creating Workbooks....16
Concept: Using External References to Refer to Data in Other Workbooks....17
Concept: Conditional Formatting in Workbooks....20
Concept: Protected Ranges and Defined Names in Workbooks....23
Concept: Uploading and Downloading Workbooks....25
Concept: Searching in Workbooks....28
Concept: Visual Cues in the Worksheets User Interface....29
Define Data Validation....31
Merge Workbooks....33
Create Charts in Workbooks....34
Reference: Common Workbook Actions and Usage Notes....34
Reference: Columns, Rows, and Cells....38
Reference: Workbook Limits....40
Reference: Operations on Entire Rows....41
Reference: Workbook Quick Statistics....41
Reference: Automatic Data Updates in Workbooks....43
Reference: Workbook Actions Available Based on Permissions....44
Reference: Worksheets Keyboard Shortcuts....46
Reference: Mobile Features for Worksheets....49

Calculations in Workbooks....49
Concept: Formulas and Formula Bar Actions....49
Concept: Array Formulas in Workbooks....51
Concept: Formula Writer and Explainer....52
Concept: Automatic and Manual Calculation in Workbook Formulas....55
Concept: Goal Seek....56
Concept: Locale, Language, and Time Zone in Workbooks....57
Concept: Dates and Times....58
Concept: Circular References....59
Reference: Formula Editor....60
Reference: Array Formula Keyboard Shortcuts....61
FAQ: Workbook Performance....62

Collaboration in Workbooks....65
Share Workbooks....65
Manage Workbook Versions....66
Manage Workbook Comments....67
FAQ: Collaboration and Security in Worksheets....67

Live Workday Data in Workbooks....69
Concept: Live Data in Workbooks....69

Concept: Global Prompts in Live Data....71
Concept: Structured References in Workbooks....72
Concept: Creating Reports to Insert into Workbooks....75
Concept: Editing Live Data Prompt Values....77
Concept: Editing Live Data Columns....77
Concept: Refreshing Live Data in a Workbook....78
Concept: Single- and Multi-Instance Field Values in Workbooks....80
Add Live Data from Workday Reports to Workbooks....82
Schedule Automatic Live Data Refreshes....85
Example: Use Worksheets with the What's New in Workday Report....86

Workbook Templates....90
Concept: Workbook Templates....90
Create and Distribute Workbook Templates....91
Reference: Template Actions Available Based on Permissions....92

Data Analysis in Workbooks....94
Concept: Data Analysis with Worksheets Functions....94
Concept: Public, Private, and Shared Filters in Workbooks....98
Concept: Automatic Subtotaling and Grouping....99
Create and Edit Pivot Tables in Workbooks....101

Worksheets Function Reference....103
Reference: Worksheets-Unique Functions....103
Reference: Worksheets Functions by Category....106
Reference: Formula Errors....118
Reference: Worksheets Rounding Functions....119
Reference: Measurement Units in Worksheets Functions....121
All Worksheets Functions....131
ABS....131
ACCRINT....131
ACCRINTM....132
ACOS....132
ACOSH....132
ACOT....132
ADD....133
ADDRESS....133
AGGREGATE....134
AMORDEGRC....135
AMORLINC....135
AND....136
ARABIC....136
AREAS....136
ARRAYAREA....137
ARRAYTOTEXT....138
ASC....138
ASIN....138
ATAN....138
ATAN2....139
ATANH....139
AVEDEV....139
AVERAGE....139
AVERAGEA....139

AVERAGEIF ... 140
AVERAGEIFS ... 140
BASE ... 140
BASETONUM ... 141
BESSELI ... 141
BESSELJ ... 142
BESSELK ... 142
BESSELY ... 142
BIN2DEC ... 142
BIN2HEX ... 142
BIN2OCT ... 143
BINOM.DIST ... 143
BINOM.INV ... 143
BINOMDIST ... 144
BINOMINV ... 144
BITAND ... 145
BITLSHIFT ... 145
BITOR ... 145
BITRSHIFT ... 145
BITXOR ... 145
BONDDURATION ... 146
BONDMDURATION ... 146
BYCOL ... 146
BYROW ... 147
CAPPEDVALUES ... 147
CBRT ... 148
CEILING ... 148
CEILING.MATH ... 149
CEILING.PRECISE ... 149
CELL ... 150
CHAR ... 151
CHIDIST ... 151
CHISQ.DIST ... 152
CHISQ.DIST.RT ... 152
CHISQDIST ... 152
CHISQDISTRT ... 153
CHOOSE ... 153
CHOOSECOLS ... 154
CHOOSEROWS ... 154
CLEAN ... 154
CLUSTER.KMEANS ... 154
CLUSTER.KMEANS.CENTROIDS ... 155
CODE ... 155
COLUMN ... 156
COLUMNS ... 156
COMBIN ... 157
COMBINA ... 157
COMPARE ... 157
COMPLEX ... 158
CONCAT ... 158
CONCATENATE ... 158
CONFIDENCE ... 159
CONFIDENCE.NORM ... 159
CONFIDENCE.T ... 159
CONVERT ... 160
CONVERTTZ ... 160

CORREL....161
CORRELATE....161
COS....163
COSH....163
COT....163
COTH....163
COUNT....163
COUNTA....164
COUNTBLANK....164
COUNTIF....164
COUNTIFS....164
COUPDAYBS....165
COUPDAYS....165
COUPDAYSNC....165
COUPNCD....166
COUPNUM....166
COVARIANCE.P....166
COVARIANCE.S....167
COVARIANCEP....167
COVARIANCES....167
CRITBINOM....168
CSC....168
CSCH....169
CUMIPMT....169
CUMPRINC....169
DATE....169
DATEDIF....170
DATESBETWEEN....171
DATESFROM....171
DATETIME....172
DATEVALUE....172
DAY....173
DAYNAME....173
DAYS....173
DAYS360....174
DB....174
DDB....175
DEC2BIN....175
DEC2HEX....175
DEC2OCT....176
DECIMAL....176
DEGREES....176
DELTA....176
DEVSQ....176
DIMENSIONS....177
DISC....177
DISTINCTROWS....178
DISTINCTROWS2....179
DIVIDE....180
DOLLARDE....180
DOLLARFR....180
DROP....180
DUR2DAYS....181
DUR2HOURS....181
DUR2MILLISECONDS....181
DUR2MINUTES....182

DUR2SECONDS..... 182
DUR2WEEKS..... 182
DURATION..... 183
DURATIONVALUE..... 183
E..... 184
EDATE..... 184
EFFECT..... 184
EMAIL..... 185
ENCODEURL..... 185
EOMONTH..... 186
EQ..... 186
ERF..... 186
ERF.PRECISE..... 187
ERFC..... 187
ERFC.PRECISE..... 187
ERROR.TYPE..... 187
EVEN..... 188
EXACT..... 188
EXP..... 189
EXPAND..... 189
EXPM1..... 189
EXPON.DIST..... 189
EXPONDIST..... 190
EXPONENT..... 190
FACT..... 190
FACTDOUBLE..... 190
FALSE..... 191
FILTER..... 191
FIND..... 192
FIXED..... 192
FISHERINV..... 193
FISHER..... 193
FLATTEN..... 193
FLOOR..... 193
FLOOR.MATH..... 194
FLOOR.PRECISE..... 194
FORECAST..... 195
FORECAST.WD.SEASONAL..... 195
FORMULATEXT..... 197
FREQUENCY..... 197
FV..... 197
FVSCHEDULE..... 197
GAMMALN..... 198
GAMMALN.PRECISE..... 198
GAUSS..... 198
GCD..... 198
GEOMEAN..... 199
GESTEP..... 199
GETPIVOTDATA..... 199
GROUPBY..... 199
GROWTH..... 202
GT..... 202
GTE..... 202
HARMEAN..... 203
HEX2BIN..... 203
HEX2DEC..... 203

HEX2OCT....203
HOUR....204
HLOOKUP....204
HSTACK....205
HYPERLINK....205
HYPGEOM.DIST....205
HYPGEOMDIST....206
HYPOT....207
IF....207
IFEMPTY....208
IFERROR....208
IFNA....208
IMABS....209
IMAGINARY....209
IMARGUMENT....209
IMCONJUGATE....209
IMCOS....210
IMCOSH....210
IMCOT....210
IMCOTH....210
IMCSC....211
IMCSCH....211
IMDIV....211
IMEXP....211
IMLN....211
IMLOG10....212
IMLOG2....212
IMPOWER....212
IMPRODUCT....212
IMREAL....212
IMSEC....213
IMSECH....213
IMSIN....213
IMSINH....213
IMSQRT....213
IMSUB....214
IMSUM....214
IMTAN....214
IMTANH....214
IN....215
INDEX....215
INDIRECT....216
INFO....217
INSTANCE....217
INSTANCE.DESCRIPTOR....217
INSTANCE.ID....218
INT....218
INTERCEPT....218
INTRATE....218
IPMT....219
IRR....219
ISBLANK....219
ISBOOLEAN....220
ISERR....220
ISERROR....221
ISEVEN....221

ISFORMULA....221
ISLOGICAL....222
ISNA....222
ISNONTEXT....223
ISNUMBER....223
ISO.CEILING....223
ISOCEILING....224
ISODD....224
ISOMITTED....225
ISOWEEKNUM....225
ISPMT....225
ISREF....226
ISTEXT....226
JOIN....226
KURT....228
LAMBDA....228
LARGE....229
LCM....229
LEFT....229
LEN....230
LET....230
LINEST....230
LN....231
LOG....231
LOG10....231
LOG1P....231
LOGEST....232
LOGINV....232
LOGNORM.DIST....232
LOGNORM.INV....233
LOGNORMDIST....233
LOGNORMINV....233
LOOKUP....234
LOWER....235
LT....235
LTE....235
MAKEARRAY....236
MAP....236
MATCH....236
MATCHCOMPOSITE....238
MATCHEXACT....238
MAX....240
MAXA....240
MAXIFS....240
MDETERM....241
MDURATION....241
MEDIAN....241
MERGECOLUMNS....242
MERGEROWS....242
MHLOOKUP....243
MI.CONTAINS....244
MI.COUNT....244
MI.INDEX....244
MID....244
MIDENTITY....245
MIN....245

MINA....245
MINIFS....246
MINUS....246
MINUTE....247
MIRR....247
MMULT....248
MOD....248
MODE....249
MODE.MULT....249
MODE.SNGL....249
MONTH....249
MONTHNAME....250
MROUND....250
MS.GROUPBY....251
MS.REGEXEXTRACT....251
MS.REGEXREPLACE....252
MS.REGEXTEST....252
MS.SORT....253
MS.UNIQUE....253
MS.UNIQUE2....254
MULTIINST....254
MULTINOMIAL....254
MULTIPLY....255
MUNIT....255
MVLOOKUP....255
N....257
NA....257
NE....257
NEGBINOM.DIST....258
NEGBINOMDIST....258
NETWORKDAYS....258
NETWORKDAYS.INTL....259
NOMINAL....260
NORM.DIST....260
NORM.INV....260
NORM.S.DIST....261
NORM.S.INV....261
NORMDIST....261
NORMINV....261
NORMSDIST....262
NORMSINV....262
NOT....262
NOTIFYIF....262
NOTIFYIFS....266
NOW....267
NOWTZ....267
NPER....268
NPV....268
OCT2BIN....269
OCT2DEC....269
OCT2HEX....269
ODD....269
ODDFYIELD....270
ODDLPRICE....270
ODDLYIELD....270
OFFSET....271

OR... 271
PEARSON... 272
PERCENTILE... 272
PERCENTILE.EXC... 272
PERCENTILE.INC... 272
PERCENTOF... 273
PERCENTRANK... 273
PERCENTRANK.INC... 273
PERMUT... 273
PERMUTATIONA... 274
PHI... 274
PI... 274
PIVOTBY... 274
PMT... 276
POISSON... 276
POISSON.DIST... 276
POW... 277
POWER... 277
PPMT... 278
PRICE... 278
PRICEDISC... 279
PRICEMAT... 279
PRODUCT... 279
PROPER... 280
PV... 280
QUARTILE... 280
QUARTILE.EXC... 281
QUARTILE.INC... 281
QUOTIENT... 281
RADIANS... 282
RAND... 282
RANDARRAY... 283
RANDBETWEEN... 283
RANDCONST... 284
RANK... 284
RANK.AVG... 285
RANK.EQ... 285
RANKAVG... 285
RANKEQ... 285
RATE... 286
RECEIVED... 286
REDUCE... 286
REGEXEXTRACT... 287
REGEXFIND... 287
REGEXMATCH... 288
REGEXPARSE... 289
REGEXREPLACE... 289
REGEXTEST... 290
REMOVECOLUMNS... 290
REMOVEROWS... 292
REPLACE... 293
REPT... 293
RIGHT... 294
RINT... 294
ROMAN... 295
ROUND... 295

ROUNDDOWN....295
ROUNDUP....296
ROW....296
ROWS....296
RRI....297
RSQ....297
SCAN....297
SEARCH....298
SEC....298
SECH....298
SECOND....298
SELECT....299
SEQUENCE....302
SERIESSUM....302
SETUNITS....302
SIGN....303
SIN....303
SINH....303
SLN....304
SLOPE....304
SMALL....304
SORT....304
SORT2....305
SORT3....307
SORTBY....309
SPLIT....309
SQRT....310
SQRTPI....310
STANDARDIZE....310
STDEV....310
STDEV.P....311
STDEV.S....311
STDEVP....311
STDEVS....311
SUBSTITUTE....311
SUBTOTAL....312
SUBTRACT....312
SUM....313
SUMIF....313
SUMIFS....313
SUMPRODUCT....313
SUMSQ....314
SUMX2MY2....314
SUMX2PY2....314
SUMXMY2....314
SWITCH....315
SYD....315
T....315
T.DIST....316
T.DIST.RT....316
T.INV....316
TAKE....317
TAN....317
TANH....317
TBILLEQ....317
TBILLPRICE....318

TBILLYIELD....318
TDIST....318
TDISTRT....318
TEXT....319
TEXTAFTER....320
TEXTBEFORE....321
TEXTJOIN....322
TEXTSPLIT....322
TIME....322
TIMEVALUE....323
TINV....323
TOCOL....323
TODAY....324
TOROW....324
TRANSPOSE....324
TRIM....324
TRIMCOLUMNS....325
TRIMMEAN....325
TRIMRANGE....325
TRIMROWS....326
TRUE....326
TRUNC....327
TRUNCATEMATRIX....327
TYPE....328
UMINUS....328
UNARY_PERCENT....328
UNICHAR....329
UNICODE....329
UNIQUE....330
UNIQUE2....330
UNITS....331
UPLUS....332
UPPER....332
URL....332
URLTEXT....333
VALUE....333
VALUEAT....334
VALUETOTEXT....335
VAR....335
VAR.P....335
VARA....335
VARP....336
VARPA....336
VDB....336
VLOOKUP....336
VSTACK....337
WD.ARRANGECOLUMNS....337
WD.ARRANGEROWS....338
WD.AVERAGEIF....339
WD.AVERAGEIFS....339
WD.COUNTIF....339
WD.COUNTIFS....340
WD.DATEDIF....340
WD.EXCHANGE....342
WD.FULLTOHALF....342
WD.GROUPBY....343

WD.HALFTOFULL....344
WD.LIST.GET....344
WD.LIST.LIST....344
WD.LIST.SIZE....345
WD.LIVEDATA....346
WD.MAXIF....350
WD.MAXIFS....350
WD.MINIF....351
WD.MINIFS....351
WD.MVLOOKUP....352
WD.PIVOTBY....353
WD.SLICE....355
WD.SUMIF....355
WD.SUMIFS....355
WD.VLOOKUP....356
WEEKDAY....357
WEEKNUM....357
WEIBULL....358
WEIBULL.DIST....358
WEIBULLDIST....359
WORKDAY....359
WORKDAY.INTL....360
WRAPCOLS....361
WRAPROWS....361
XIRR....361
XLOOKUP....362
XMATCH....363
XNPV....363
XOR....364
YEAR....364
YEARFRAC....364
YIELD....365
YIELDDISC....365
YIELDMAT....365
Z.TEST....366
ZTEST....366

Summary of User Guide Changes - Worksheets

This table describes updates to the User Guide within the past year. We update the User Guide when new features impact the product user interface and when editorial changes improve content quality.

This page is intended to be a summary of content changes in the User Guide, and is not a full list of new features. If a new feature results in a change to only the Administrator Guide content, that change isn't described here.

Date	Notable Content Changes
October 2025	Added information about 20 new functions Worksheets supports that provide compatibility with Excel functionality. • ARRAYTOTEXT • BYCOL • BYROW • ISOMITTED • LAMBDA • MAKEARRAY • MAP • MS.GROUPBY • MS.REGEXEXTRACT • MS.REGEXREPLACE • MS.REGEXTEST • PERCENTOF • PIVOTBY • REDUCE • REGEXTEST • SCAN • TRIMRANGE/Trim References • VALUETOTEXT • WD.GROUPBY • WD.PIVOTBY
July 2025	Feature-related changes: • Concept: Using External References to Refer to Data in Other Workbooks on page 17: Added information about configuring automatic updates to external references. • Concept: Uploading and Downloading Workbooks on page 25 and Reference: Common Workbook Actions and Usage Notes on page 34: Added information about download area and settings options when you download workbook data as a PDF.
March 2025	Concept: Public, Private, and Shared Filters in Workbooks on page 98: Added information about how to save and share private filters in workbooks.
January 2025	Concept: Visual Cues in the Worksheets User Interface on page 29: Updated information about how you are now able to open consumer workbooks that have external references without having to wait for the workbooks to recalculate updates from a producer workbook.
September 2024	Feature-related changes: • Concept: Public and Private Filters in Workbooks (new): Added information on how to add public and private filters in workbooks. • Concept: Global Prompts in Live Data on page 71 (new) and Add Live Data from Workday Reports to Workbooks on page 82 (updated): Added information on how to select a set of prompts to be global prompts, and then apply those prompt values to all live data areas in all sheets. • Share Workbooks on page 65 (updated): Added information on how to set more granular sharing options in the Share dialog. • Concept: Formula Writer and Explainer on page 52 (new): Added information on how to use Formula Writer to generate a workbook formula,

Date	Notable Content Changes
	and how to use Formula Explainer to view an explanation of how a formula works.
June 2024	Feature-related changes: • The new View My Workbook Details report enables you to view metrics for one or more workbooks, where you have View or higher permission for the workbooks. You can find the View My Workbook Details report using Workday's global search.
May 2024	Feature-related changes: • Worksheets now supports live data table names, including renaming your tables, for matrix and composite reports in addition to advanced reports. (Structured references continue to be supported only for advanced reports.) Other notable changes (clarifications and additions related to existing functionality): • Add Live Data from Workday Reports to Workbooks on page 82: If a generated table name would conflict with a workbook address, Worksheets appends an underscore to the generated name. • REGEXPARSE: The regex pattern must match the entire input value.
March 2024	Feature-related changes: • Concept: Uploading and Downloading Workbooks: Updated to describe the enhanced Download to Excel format that preserves supported Excel formulas, while displaying Worksheets-unique formulas with a special identifier along with text-based output. Note that this format doesn't preserve a connection with live data / Workday reports. • Reverted changes to topics related to advanced reports running in the background. The asynchronous advanced report feature was temporarily reverted due to issues that were discovered after release.
November 2023	Feature-related changes: • Concept: Visual Cues in the Worksheets User Interface: Updated to describe the new purple highlighting for array areas.
October 2023	Feature-related changes: • Add Live Data from Workday Reports to Workbooks: Updated to mention that if you add live data to a workbook and you don't manually specify a table name on the Select Options page of the wizard, Worksheets automatically creates a value based on the underlying Workday report name. • Updated topics that mention live data based on advanced reports, because advanced reports now run in the background, and don't require that you wait for them to complete when you refresh live data. • Updated relevant topics to mention that Worksheets now automatically provides structured reference syntax and defined object syntax in the formula bar when you are typing a formula.
August 2023	Feature-related changes: • Concept: Protected Ranges and Defined Names in Workbooks: Added information about editing defined names.
June 2023	Feature-related changes:

Date	Notable Content Changes
	• Concept: Locale, Language, and Time Zone in Workbooks: Clarified how Worksheets creates and uses these settings, and added information about changing the workbook language. • Reference: Common Workbook Actions and Usage Notes: Added information on how to change the workbook language, locale, or time zone.

Getting Started with Worksheets

Concept: Introduction to Workday Worksheets

Worksheets is a spreadsheet component in Workday that enables ad-hoc data exploration, analysis, visualization, and collaboration with live transactional data. Workday users (employees, managers, and executives) can collaborate with secure, live, Workday data as well as external data using the Workday browser application. You can use the Workday mobile app to view workbooks.

On a web browser, you access Worksheets workbooks from Drive by selecting the Drive option in the Workday main menu. On the Workday mobile app, use the Drive worklet.

Worksheets provides real-time user presence information, secure data sharing, and engaging visualizations. For example, you can create a new workbook and then:

- Insert data from a Workday report into the workbook.
- Share the workbook with your colleagues.
- Comment on cells or the entire workbook; in a comment you can tag others so they receive a notification about your comment.
- Collaboratively edit the workbook, in real time, with others who have edit access.
- Create a named version to save a snapshot of a group of changes.
- Download the workbook to a different format, such as XLSX, PDF, or HTML.

In addition to including nearly all well-known spreadsheet functions such as those in Google Sheets™ or Microsoft® Excel®, Worksheets includes many additional Worksheets-unique functions.

Concept: Creating Workbooks

A workbook is the main container, or spreadsheet, in Worksheets. You can create a workbook in several ways, depending on the source data for the workbook:

Workbook Source Data	Steps	Notes
New data.	From Drive, select +New > Workbook.	Worksheets creates a blank workbook.
Imported (uploaded and converted) from a non-Workday file.	To upload and convert a Microsoft® Excel® file, or any file in comma-separated values (CSV) or HTML format, use one of these methods from Drive: • Drag and drop the file onto the main area of the page (not the left pane). • Select +New > Upload.	Workday: • Includes only table data when uploading from HTML. • Doesn't include charts or Visual Basic macros. • Adds a prefix of _XLFN. to any function that Worksheets doesn't support.

Workbook Source Data	Steps	Notes
	When you upload a type of file that Worksheets supports, Worksheets automatically converts it into a workbook.	
Workday application data (grid).	From the Workday action menu or from the Workday Reports page, select Export to Worksheets.	If you have access to Worksheets, the Export to Worksheets icon displays next to the Export to Excel icon. The created workbook: • Contains static data that isn't connected to the original Workday report. • Doesn't display multi-instance field values.
Workday report.	From Drive, select +New > Workbook. Then in the workbook select Data > Add Live Data.	You can insert data only from Workday reports that you have access to. If you select the As Live Data option during the insert, in the future you can manually refresh the workbook based on the report. Workbook owners can also schedule automatic live data refreshes.
Integrating applications such as Payroll, Projects for Resource Forecasting, Mass Actions, Org Studio, and so on.	Follow the workflow steps in the integrating application documentation.	

Concept: Using External References to Refer to Data in Other Workbooks

In a workbook, you can refer to cells or ranges of data that exist in other workbooks. We call these references *external references* or *cross-workbook references*. References to other sheets in the same workbook have a similar syntax so we'll describe them in this topic also. You can make a standalone reference (by using only an equals sign and the reference) or a reference can be an argument in a formula.

A workbook that refers to (brings in) data from another workbook is a *consumer* workbook. A workbook containing data that's being referred to in another workbook is a *producer* workbook. A workbook can be both a consumer and a producer.

Creating External References

To add references to a producer workbook from a consumer workbook, you must have Can Edit permission for the consumer workbook and Can View permission or higher for the producer workbook.

When someone creates or changes a formula with an external reference, Worksheets remembers which user did it. When the formula runs, Worksheets verifies that the user has permission to access the externally referenced workbook.

From a producer workbook, you can copy a complete reference and then paste it into a consumer workbook.

1. Select the data that you want to reference and then right-click and select Copy as External Reference.
2. Open the consumer workbook. In the cell where you want the reference, type an equals (=) sign and paste the reference by pressing Ctrl + V (Windows) or Command + V (Mac).

Alternatively, you can obtain the producer workbook's Workday ID by navigating to File > Info and adding the rest of the external reference manually in the consumer workbook.

From a consumer workbook, you can copy the Workday ID of a producer workbook and then add the defined name or sheet name and cell/range reference manually.

1. In the cell where you want to place the reference, select Data > Get External Reference to see all the workbooks that you have access to.
2. Navigate to the producer workbook and then click Copy Workday ID.
3. In the consumer workbook, type an equals (=) sign and paste the Workday ID by pressing Ctrl + V (Windows) or Command + V (Mac). Then type the rest of the reference, which might be a defined name or a sheet name and cell/range reference.

Updating External References

Consumer workbooks display a gray External References icon on the workbook toolbar. If data in the producer workbook changes, the icon turns green and you can click it to update the data in the consumer workbook. The External References icon is accessible to users with any level of workbook permission. The update that occurs is the same as Recalculate All. There might be a delay of up to 2 minutes, once the data in the producer workbook changes, before the consumer workbook External References icon turns green.

When producer workbook data changes, you can choose to have the external references update automatically when the consumer workbook is opened, or you can open the consumer workbook and click the External References icon to update the external references manually. Click File > Settings and either select or deselect the Updates data from external workbooks each time the workbook is opened check box.

Additionally, you can schedule a live data refresh to prompt the consumer workbook to be recalculated. You can set up a scheduled refresh to automate a recalculation even if the consumer workbook doesn't contain live data.

For example, if you have 3 workbooks joined together (Producer > Consumer/Producer > Consumer) you can automate updates to each workbook by scheduling data refreshes. Additionally, staggering the scheduled refreshes ensures that the data in one workbook is up to date before the data in the next workbook is recalculated.

This approach automates the data flow between the workbooks, eliminating the need for manual updates.

1. Schedule a refresh for the producer/consumer workbook.

   In the producer/consumer workbook, click Data > Schedule Live Data Refresh and set a refresh time such as 12:00 PM. This will update the data pulled in from the producer workbook.
2. Schedule a refresh for the consumer workbook.

   In the consumer workbook, schedule a refresh time after the producer/consumer refresh, such as 12:15 PM. This ensures the consumer workbook always displays the latest recalculated data.

The Schedule Live Data Refresh option is only configurable by the owner of the workbook and scheduled refreshes can only run once per day.

Syntax Guidelines

For external references:

- Enter the workbook's Workday ID inside square brackets, then the sheet name and cell or cell range, or the defined name. Don't include folder/path information for the producer workbook.

For cross-sheet references or external references:

- Start with an equals sign (=) in the cell.
- If the sheet name has either of these properties, surround the name with single quotes ('):
    - The sheet name doesn't start with a letter or an underscore.
    - Any subsequent character in the sheet name isn't a letter, a digit, a period, or an underscore.
- Example cross-sheet references:
    - =Sheet1!A1:A4
    - ='All Employees'!A1:A4
    - ='All Employees'!DataArea1, where DataArea1 is a defined name

This table shows some external reference examples. Workday IDs are shortened for readability.

=[9f9a1]January!A1	Gets data from the workbook with Workday ID 9f9a1, in the sheet January, cell A1.
=SUM([9f9a1]January!A1:B6)	Shows the sum of cells in the workbook with Workday ID 9f9a1, in the sheet January, range A1:B6.
=[9f9a1]DataArea1	Gets data from the workbook with Workday ID 9f9a1, with the defined name DataArea1.
=SUM([9f9a1]Regions!A1:B6)	Shows the sum of cells in the range A1:B6, in the workbook with Workday ID 9f9a1, sheet Regions.
=SELECT("SELECT * FROM ?",ARRAYAREA([9f9a1]'Data Sheet'!A1))	Uses the SELECT function to get an array's data from the workbook 9f9a1, sheet Data Sheet.

Considerations for External References

Keep these tips in mind when working with external references:

- If the data changes in a producer workbook:
    - When you click the green External References icon to update the consumer workbook, the action is the same as Recalculate All.
    - If you change the data placement in a producer, such as by deleting a sheet or inserting a column into a range, Worksheets doesn't update the reference in the consumer workbook.
- If you lose access to a producer workbook or a producer workbook is deleted, you see a #REF! error in workbook cells that depend on data from external references.
- Worksheets reassigns broken external references in workbooks when the references become invalid due to the authoring user's account being deactivated, or when the authoring user no longer has permissions to the producer and consumer workbooks. Worksheets attempts to fix the external references whenever a user who has edit or owner to the consumer workbook, and view access or higher to the producer workbook, clicks the External Reference icon. If Worksheets can't update a reference, a #REF error displays and you need to update the reference manually.

- If you have an external reference that contains a structured reference (live data table/column name), and you change the table name or column name, Worksheets doesn't automatically update the consumer workbook; you need to update the reference manually.
- Formatting in producer workbook references is applied in the consumer workbook, unless the consumer workbook cells are already formatted. If you change the formatting in the referenced producer workbook cells later, the new formatting isn't applied in the consumer workbook. (This same behavior occurs for references in the same workbook.)
- If you hover over the Workday ID in an external reference in the formula bar, Worksheets shows the name of the producer workbook.
- You can't add a reference to a producer workbook chart into a consumer workbook; an error occurs when you place the reference in the consumer.
- If you're the workbook owner, you can prevent users with the Can Edit permission from changing content in the producer workbook by using range protection. However, using range protection in a consumer workbook doesn't prevent updates from propagating from the producer to the consumer.
- On a particular tenant, up to 10,000 consumer workbooks can refer to a single producer workbook. Up to 25 producer workbooks can provide data to a single consumer workbook.
- The functions NOTIFYIF and NOTIFYIFS don't update automatically when producer workbook data changes; you need to update references manually or set up a scheduled live data refresh. (The workbook doesn't need to contain live data; the scheduled refresh performs a recalculate action, which updates the consumer workbook data.)

Concept: Conditional Formatting in Workbooks

Conditional formatting enables you to apply cell formatting selectively and automatically, based on cell content. Conditional formatting is an easy way to quickly identify erroneous cell entries, cells of a particular type, or cells within a range of values. You can use a format (such as a gradient of a red background color) to make particular cells easy to identify. Sometimes conditional formatting is a better way to visualize your data than a chart.

You can apply formatting rules to standard data, unconstrained formula data, live data, and pivot tables.

You can assign single color or color scale (gradient) formatting rules based on formulas, or on preset conditions. Here are a few examples of pre-set conditions:

- Greater than or equal to
- Less than or equal to
- Is equal to
- Has errors
- Text contains
- Text begins with
- Date
- Above or below average
- Cell is blank

To view the existing conditional formatting rules for a range of cells, select the range and then select View > Panels > Conditional Formatting. To display all conditional formatting rules for a sheet, select the top left cell in the workbook data area.

Worksheets doesn't support adding conditional formatting rules for data bars or icon sets. If you upload an Excel workbook containing data bars or icon sets, they aren't preserved in the Worksheets workbook.

Adding a Conditional Formatting Rule

To apply a conditional formatting rule to a cell or range:

1. Select the cell or range, then navigate to Formatting > Conditional Formatting. The Conditional Formatting panel opens and the Apply to Range field contains the range you selected.

2. In the Formatting Style drop-down menu, select Classic for a single color rule or Color Scale for a color gradient rule.

Classic	Select the type of rule, the value to base the rule on, and the formatting for the cell or range if the rule is true.
Color Scale	Select the Minimum, Midpoint, and Maximum value type. If applicable for that value type, type the associated value to the right. Then select a color to associate with that value type. The Formatting example shows a preview of the condition rule's appearance.

3. (Optional) Select the Stop if true check box if you have more than one rule and you want to stop processing the rules that are lower in the list when the first "true" condition is achieved.
4. Click Apply.
5. (Optional) If you have more than one rule for the same range of data, you can drag rules in the Conditional Formatting panel to change their priority, or click the gear icon and then click Move Up or Move Down.

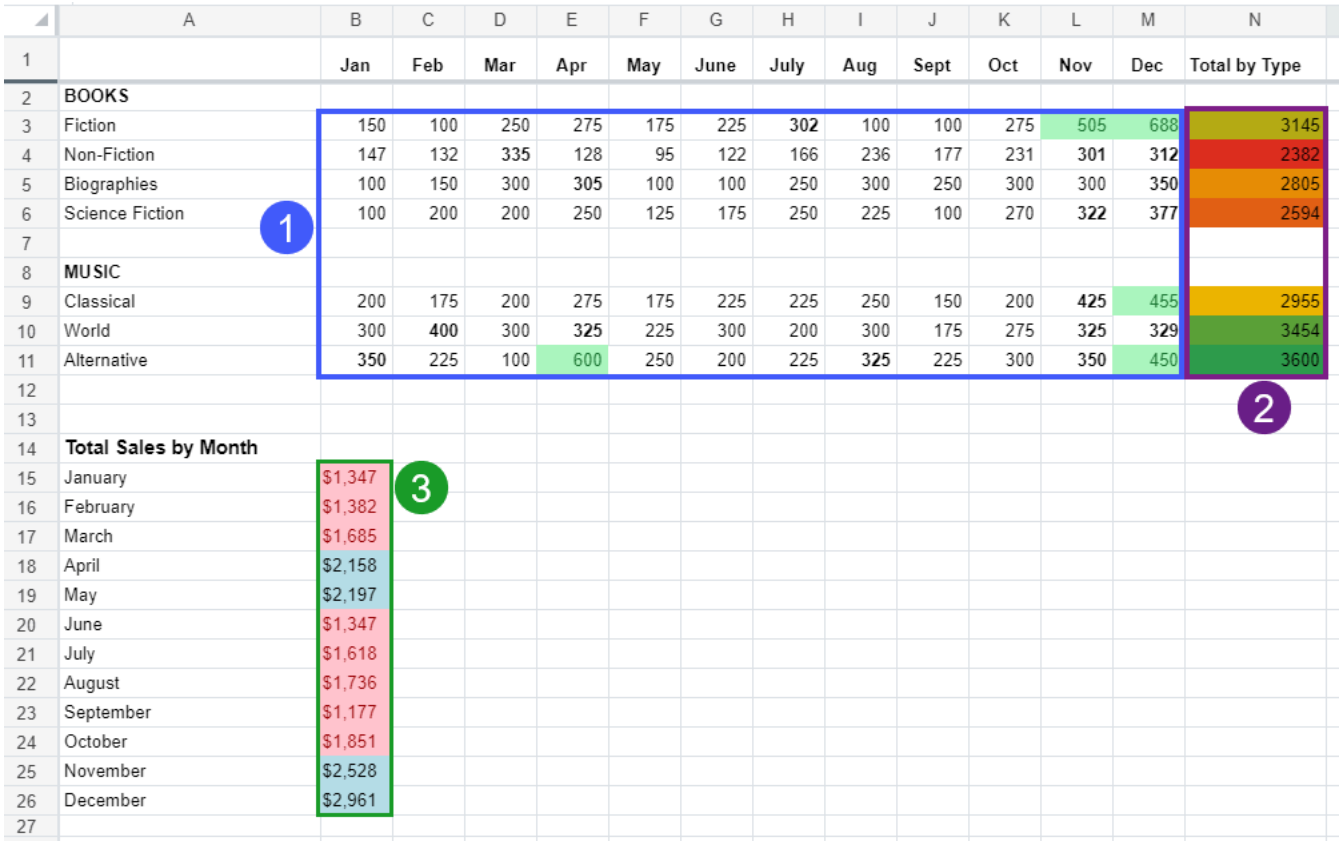
Considerations for Using Conditional Formatting

Keep these tips in mind when using conditional formatting:

- If you apply a classic rule to your data, Worksheets sets the conditions correctly for either numeric or string values in the cells. If you use a color scale in your rule, the cell values must be numbers.
- Avoid using the rule *Greater than 1* in a full-column/row rule, because this rule can cause the header row's formatting to change.
- If you insert rows or columns within a range that already contains conditional formatting, the newly inserted cells get the same conditional formatting.
- When you press Delete to delete the contents of a cell, you don't delete any existing conditional formatting. To remove all conditional formats and all other cell formatting, select the cell and then choose Edit > Clear > All.
- You can associate both classic (single color) rules and color scale (gradient) rules to the same set of cells, but we don't recommend it because you might see unexpected results.
- When you have multiple rules in a priority list, Worksheets doesn't apply the same formatting (such as background colors) again in subsequent rules. For example, if you have a rule to apply formatting when a value is less than 6, then in a later rule you apply formatting when a value is less than 4, Worksheets won't apply the same formatting with that later rule. If you do want that behavior, you need to place the "less than 4" rule higher in the rule list than the "less than 6" rule.
- When you download a workbook, Worksheets doesn't preserve these conditional formatting rule types:
    - Top 10
    - Unique values
    - Duplicate values
    - Contains (text)
    - Does not contain (text)
    - Ends with
    - Begins with
    - Contains blanks
    - Does not contain blanks
    - Contains errors
    - Does not contain errors
    - Time period
    - Above average

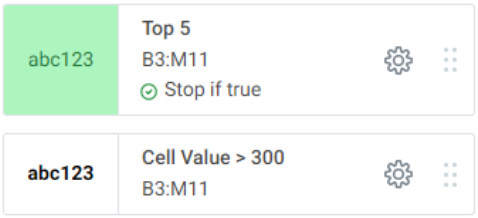
Examples of Conditional Formatting

This workbook sheet has 3 conditional formatting rules. You can experiment using this workbook and the example rules below to see similar results.



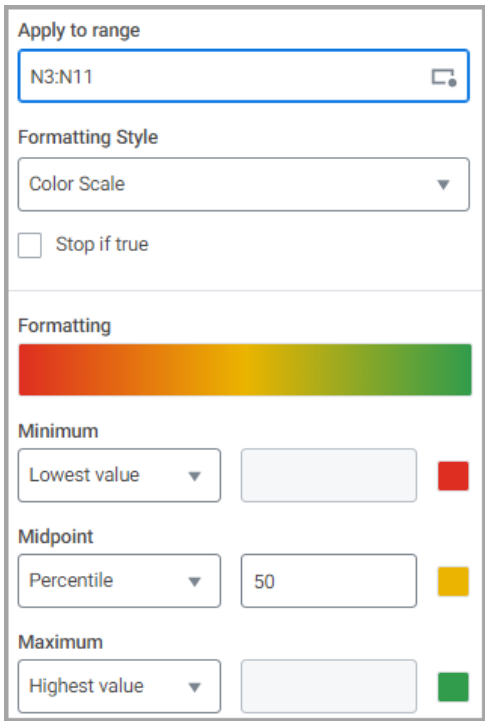
	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1		Jan	Feb	Mar	Apr	May	June	July	Aug	Sept	Oct	Nov	Dec	Total by Type
2	BOOKS													
3	Fiction	150	100	250	275	175	225	**302**	100	100	275	505	688	3145
4	Non-Fiction	147	132	**335**	128	95	122	166	236	177	231	**301**	**312**	2382
5	Biographies	100	150	300	**305**	100	100	250	300	250	300	300	350	2805
6	Science Fiction	100	200	200	250	125	175	250	225	100	270	**322**	377	2594
7														
8	MUSIC													
9	Classical	200	175	200	275	175	225	225	250	150	200	**425**	455	2955
10	World	300	**400**	300	**325**	225	300	200	300	175	275	**325**	**329**	3454
11	Alternative	**350**	225	100	600	250	200	225	**325**	225	300	350	450	3600
12														
13														
14	**Total Sales by Month**													
15	January	$1,347												
16	February	$1,382												
17	March	$1,685												
18	April	$2,158												
19	May	$2,197												
20	June	$1,347												
21	July	$1,618												
22	August	$1,736												
23	September	$1,177												
24	October	$1,851												
25	November	$2,528												
26	December	$2,961												
27														

1: Two Rules, including a Stop If True setting



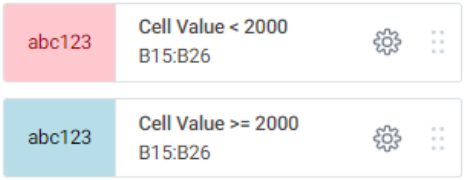
- The first rule changes the font color and cell background color for the top 5 values in the range.
- Because of the Stop if true setting, after identifying the top 5 values, Worksheets doesn't process the second (bold) rule.
- The second rule formats all values greater than 300 in Bold. The top 5 values, even though they're over 300, are not bold because we stopped processing the rules as soon as we identified the 5 values.

2: Color Scale Rule for lowest to highest values



The lowest to highest values in the range are shaded from red, through yellow, to green, with a percentile to indicate the middle of the range.

3: Two classic rules for pink and blue formatting



Concept: Protected Ranges and Defined Names in Workbooks

Workbook owners and editors can create a *defined name* for cells or ranges of cells in workbooks to make formula-writing easier and to make formulas easier to read. Owners can also *protect* ranges to prevent all collaborators, including editors, from changing them.

Defined Names

Action	Steps	Notes
Create a defined name.	Select the cell or range, right-click, and then select Define Name. The Formula field contains the selected cell or range reference by default; to define a name for a formula, constant, or non-contiguous set of cell ranges, edit the field as needed.	Naming rules are consistent with popular spreadsheet products, to maximize compatibility. Names can be between 4 and 255 characters long. Defined names must be unique per workbook, and can't be the same as any pivot table names or live data table names.

Action	Steps	Notes
	If you want to create a defined name containing live data that updates automatically when you refresh the live data, use the ARRAYAREA formula. Example: Specify something like this in the Formula field for the defined name: `=ARRAYAREA('Sheet1'!A1)`	
Insert a defined name into a formula.	When you are typing a formula in the formula bar or in a cell, you don't need to remember all the defined names in the workbook. When you start typing a defined name, Worksheets generates a dropdown list of names for you to select from. When selecting a defined name in the formula drop-down menu using the keyboard, use Tab (not Enter) to select it.	
View all defined names.	Open the Defined Names panel by selecting View > Panels > Defined Names. Alternatively, when you start typing a formula into the formula bar or a cell, defined names for the workbook display in the drop-down menu.	When you select a defined name, its cells are highlighted in the workbook.
Edit a defined name	From the Defined Names panel, select the options (three dot) menu and click Edit.	You can edit either the name or the cell range. If you use this defined name in a Slides presentation and later you change its name, you need to refresh the linked data in the presentation to see the updated data. If you edit the range so that the linked value is no longer included, you need to remove and re-add the linked data.
Delete a defined name.	Hover over the range in the Defined Names panel, then click Delete.	If you delete a defined name, any formulas associated with that name will stop working. Recreating it with the same name will restore functionality in Worksheets, but will not work for the Slides integrating app. In this situation, you'll need to create new links. This is because Slides uses an internally-generated ID instead of the name to identify defined names.

Protected Ranges

Workbook owners can protect ranges to prevent all collaborators, including editors, from changing them.

For protected ranges, in the Protected Ranges panel the workbook owner sees Can Edit below any protected range name; all other collaborators see Can View.

When any user selects a cell in a protected range, the user interface displays a lock indicator on the function bar icon.

Note: When you refresh the live data in a workbook, Worksheets disregards protected ranges when doing those refreshes. You can't prevent a live data refresh from overwriting existing workbook content.

Action	Steps	Notes
Protect a range.	Select the cell or range, right-click, and then select Protect Range.	A user with edit permission can change public filter selections, by clicking the filter icon on a column and then changing the selected checkboxes. As a best practice, if you want to allow users to filter protected data, place a public filter on the workbook before protecting it, and then remind people with edit access to click the filter icon to do their own filtering.
View all protected ranges.	Open the Protected Ranges panel by selecting View > Panels > Protected Ranges.	When you select a protected range, its cells are highlighted in the workbook.
Remove protection for a range.	Hover over the range in the Protected Ranges panel, then click Delete.	

Concept: Uploading and Downloading Workbooks

Worksheets supports uploading, converting, and downloading several types of files. System-wide settings and workbook permission levels determine whether you have access to these actions.

This topic doesn't cover downloads and uploads of tenant migration workbooks (WXF files). Normally only administrators manage WXF files. People with the appropriate access can refer to the Administrator Guide for more information.

Uploading Workbooks

From Drive, select +New > Upload and follow the prompts to select a file.

When uploading XLSX files, keep in mind that this upload action is intended only for workbooks that contain exclusively Excel-supported content; don't use it to upload workbooks that you worked with in Worksheets and then downloaded to Excel using the Worksheets download action.

When you upload a file in the XLS, XLSX, CSV, or HTML file type, Worksheets automatically converts it into a workbook and also keeps the original file.

Worksheets doesn't convert Excel files to workbooks if they require a password to open.

This table summarizes whether or not Worksheets preserves Excel-related content in uploaded workbooks:

Content Type	Preserved?	Notes
Pivot tables	No	Excel pivot tables display as static text (not as pivot tables) in Worksheets.
Formulas	Yes	Worksheets supports most, but not all, Excel functions. A prefix of _XLFN. is added to any function that Worksheets doesn't support.
Data Validation	Yes	
Defined names	Yes	We strongly recommend not uploading workbooks containing: • Defined names that refer to other files. • Defined names that contain #REF errors. • Hidden defined names. (Any time you copy an Excel workbook, all defined names are copied and hidden, so you might need to do additional steps to find and remove them.) Any of these situations can cause the upload to fail.
Data values	Yes	
Conditional formatting	Yes, see notes	If you upload an Excel workbook containing data bars or icon sets, they aren't preserved in the Worksheets workbook.
External references	See notes	When you upload an Excel file containing an Excel-formatted external reference, an error dialog displays. The sheet and cell/range reference are preserved; the workbook name is replaced with WD_EXT_1_.
Charts and other images	No	If you upload an Excel file containing charts and images to Drive, the charts and images aren't preserved in the Worksheets workbook.
Visual Basic macros	No	Worksheets doesn't support macros. If an Excel file contains macros, the conversion to a workbook might fail. Save the Excel file without macros before uploading it.

Oversized Excel Files

For Excel files, check the file size in addition to the number of cells before uploading it. In some cases an Excel file visually appears to be small but the file size is disproportionately large; these files are often referred to as bloated files. A bloated file might upload very slowly or it might fail. You need to eliminate the cause of the oversized file. Here are some common causes and tips:

- Formatting information. Unnecessary formatting is the most common cause of a bloated file. Sometimes formatting rules are applied to a full sheet or to all columns or rows in a sheet; this formatting is invisible but it can dramatically increase the file size. Spreadsheets that were originally created in Google Sheets contain formatting information for all cells up to Z:1000 by default. To determine if formatting is an issue, you can press Ctrl+End to locate the last used cell in a sheet. To eliminate formatting, you can select each data range in the bloated file using Ctrl+Shift+{arrow keys}, and then paste the content into a new XLSX file.
- Hidden sheets. Unhide all the sheets, and remove any that are unnecessary.
- Unused defined names or links to other workbooks in a copied workbook. Excel can retain data associated with defined names and external references even after you break the link to the original workbook. Make sure you delete any unused references.
- Other possible causes of bloat include sharing or revision information, deleted macros, and more.

Downloading Workbooks

In the workbook, select File > Download as and then select an output format.

You can download workbooks into XLSX, PDF, CSV, TSV, HTML, or WXF format.

This table summarizes special considerations, depending on the format you're downloading to:

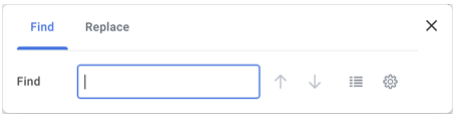
Download Format	Notes
All types	You might need to enable pop-ups in your browser. These Worksheets elements aren't preserved after you download: • Comments • Protected ranges • Filters Data validations: Worksheets allows for some validation definitions that are more robust than Excel; those validations won't work when downloaded to Excel.
PDF	Data might be truncated if the cells aren't big enough. Merge or expand cells as needed before downloading to make sure all your data is displayed in the cells, with no overlap into adjacent cells Click File > Download as > PDF document (.pdf). There are 2 types of options: • Download Area: You can choose to download a range of cells, a single sheet, or multiple sheets in a workbook. • Settings: You can choose page formatting options such as page order, type of layout, page orientation, and other data details.
Excel	Note that live data and the connection to Workday reports isn't preserved in downloaded Excel files. If you need to preserve the live data, download the workbook as a WXF file. There are 3 formatting options: • Download As > Microsoft Excel > Export Formulas. Use this format to preserve supported Excel formulas, while displaying Worksheets-unique formulas with a special identifier along with text-based output. Worksheets-unique formulas show the last calculated result. • Download As > Microsoft Excel > Export Values Only. Use this format when you are interested in working with or reviewing the data but you don't need to preserve formulas. Microsoft Excel as Values: Use this format when you are interested in working with or reviewing the data but you don't need to preserve formulas. The downloaded file contains only values, with no formulas or charts. • Download As > Microsoft Excel Legacy (.xlsx). For backward compatibility, Worksheets provides the legacy Excel download format that includes Worksheets-specific information in JSON format. For Worksheets-unique scalar (non-array) formulas, you see a #NAME error in other spreadsheet products. Wherever a workbook formula or value is Worksheets-unique (not compatible with Excel), Workday prepends the cell content with identifying text in JSON format such as *Workday Unconstrained Array Ext2*: or *V2.Workbook.valueNumber:*. This identifying text is for Workday internal use only. We don't support saving exported workbooks for extended periods of time and then re-importing them into Worksheets, or using custom processes to parse the workbook text. Because Excel doesn't support instance values or multi-instance values, Worksheets downloads this type of cell content as a string (text). If a cell contains a multi-instance value with many instances in it, the content might be greater than Excel's 32,767 character limit for cells; if this occurs, you see an #N/A error in the cell.

Download Format	Notes
Non-Excel	Only the currently selected sheet downloads; the automatically assigned filename is *workbookname_sheetname.filetype*.
WXF	Download As > Tenant Migration File (.wxf). You can download this file type for troubleshooting purposes, to provide to Support. This is not a user-readable file format. Normally only administrators manage WXF files. People with the appropriate access can refer to the Administrator Guide for more information.

Concept: Searching in Workbooks

To find values in a workbook sheet, and optionally replace them, select Edit > Find or press Ctrl+F (on Windows) or Command+F (on Mac) to open the Find popup.

You can drag the popup around on the sheet if needed, so you can see the content that you're working with.

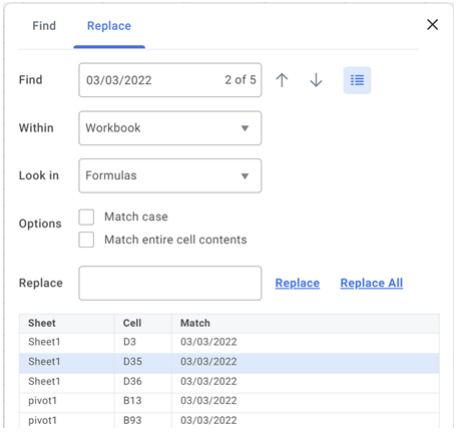


You can type up to 1000 characters of text into the search field. Worksheets finds cells containing the text, as you type.

You can view a list of the results by clicking the List icon.

By default, Worksheets searches the current sheet, and it searches in formulas (not values). You can do more actions by clicking the Find Options icon:

- Search the entire workbook or a selected range.
- Find a formula value (the value as displayed in the formula bar).
- Match the case of the Find field value.
- Limit results to those that exactly match the Find field value.



When you select a range of cells before opening the Find dialog, Worksheets preserves your selection. The Find Options dialog displays, with the Within field and range field auto-filled based on your selected range.

Keep these considerations in mind when using the Find and Replace feature in a workbook:

- You can't replace any protected or read-only values, such as live data, pivot tables, charts, or array formulas.

- Although you can't replace live data values, you can replace live data notes and formula columns in live data.
- Worksheets finds date values, and values that have a units element, based on your selection of either Formulas or Values in the Look In field. Example: If a cell displays January 1, 2022, the formula bar might display 01/01/2022. When searching, select to look in Formulas to find 01/01/2022 or look in Values to find January 1, 2022.
- The first 1,000 results display in the list view. After you scroll through those results, the next set of 1,000 results displays.
- Think about the size of your workbook before using the Replace All action. Performance can degrade with large workbooks, particularly when a replacement causes the workbook to recalculate, such as when a volatile function is involved. If you choose to do a Replace All with more than 10,000 values, Worksheets replaces 10,000 results and returns to the Find dialog. You can then replace more results.
- If you search for a value in a hidden sheet, or in a hidden column or grouped columns, Worksheets doesn't return matches.
- If you replace values for cells containing single-instance or multi instance values, all instances in a cell convert to a string.
- When searching for content, Worksheets supports the ability to look in either values or in formulas. When replacing content, Worksheets supports only looking in formulas. Example: If you want to replace January 1, 2022 with January 1, 2023, you need to search for the value as displayed in the formula bar - typically 01/01/2022 - not January 1, 2022.
- Worksheets returns results for pivot tables and live data areas when the Look in selection is Formulas even though we don't typically think of these values as having data in the formula bar. This is because Worksheets derives those cell values from formulas.

Finding Text Using Wildcard Characters

Worksheets supports using common wildcard characters to find single specific characters, multiple matching characters, and question marks, asterisks, or tilde characters:

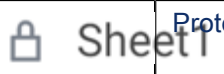
- To find any single character, use the question mark (?). Example: Searching for `nu?c` finds values that start with nu, followed by any character, followed by a c followed by any characters.
- To find any number of characters in a value, use the asterisk (*). Example: Searching for `man*` finds values that start with the string man, followed by any characters.
- To find question marks, asterisks, or other tilde characters, use the tilde (~) followed by ?, *, or ~ . Example: Searching for `fy22~?` finds `fy22?`.

Concept: Visual Cues in the Worksheets User Interface

Icons

This table describes notable icons that indicate certain actions and states in the user interface.

Icon	Type	Notes
	Static formula, array formula, or data cell	Gray fx without braces.
	Constrained formula	Blue fx with curly braces.
	Unconstrained formula	Orange fx with curly braces.

Icon	Type	Notes
	Global prompts	Gray icon in the toolbar that you can use to assign global prompts for a live data area.
	AI Formula Writer	Purple icon that opens the AI Formula Writer.
	External references	Gray icon with arrow pointing out. Indicates that this is a consumer workbook, and data from the producer is up to date.
	External references	Green icon with arrow pointing in. Indicates that this is a consumer workbook and data in the producer workbook was changed. You can click the icon to update the data in the consumer workbook. The update that occurs as a result of the action is the same as Recalculate All. Once the producer workbook's data changes occur, there might be a delay of up to 2 minutes before the consumer workbook External References icon turns green.
	Circular references and iteration limits	A red or yellow icon indicates that Worksheets didn't finish calculating; the workbook is in a state where some formulas didn't fully run and are showing stale values. You can hover the cursor over the icon to see a tooltip with information about the problem. There are two common causes for a workbook to prematurely stop calculating: either the calculation limit was reached, or one or more circular references exist in the workbook. If the problem is caused by a circular reference, you can click the red icon to navigate to the cell location of the reference. If you have more than one circular reference, Worksheets navigates to the location of the first one.
	Circular references and iteration limits	A red or yellow icon indicates that Worksheets didn't finish calculating; the workbook is in a state where some formulas didn't fully run and are showing stale values. A yellow icon indicates that during formula calculations a function such as ARRAYAREA, INDIRECT, or IF has revisited cells more than 5 times. Essentially a kind of chain reaction exists where an unconstrained formula changes data, which causes another unconstrained formula to run, and it repeats until this has continued 5 times. If this occurs, you need to simplify your usage of unconstrained formulas.
	Protected data	A lock indicator displays on the function bar when a user selects a cell or a row in a protected range.
	Protected data	A lock icon displays in the header of a protected column.
	Protected data	A lock icon displays next to the sheet name of a protected sheet.
	Comments	A dot on the Comments icon indicates unread comments. To open the Comments panel and mark all comments as read, click the icon.
	Comments	A triangle in the corner of a workbook cell indicates that a comment exists for that specific cell. To view the comment, right-click the cell and select View Comment.
	Filter	A blue filter icon displays at the top of any column that has an active filter.

Colors and Borders

This table describes notable colors and borders that display around different types of data in specific situations.

Color or Border	Type	Notes
Solid purple border	Array data	Worksheets displays a solid purple border around all the cells of an array created using an array formula.
Solid orange border and banner	Public filter	An orange border and banner indicate an active public filter.
Solid teal border and banner	Private filter	A teal border and banner indicate an active private filter.
Solid purple border and banner	Shared filter	A purple border and banner indicate an active shared filter.
Dashed line blue border	Live data	Worksheets displays a dashed line blue border around all the cells in live data. You can toggle the live data highlighting off and on using the check box in the Live Data panel.
Dashed line purple border	Spill error	If a formula calculation or a data update results in a SPILL error (the results can't display because they would overlap existing data), and then you select the anchor cell of the array, a dashed line purple border appears around the cells that would be populated if the calculation/update could occur. The border includes the cells that are preventing the data from populating.
Dashed line green border	Integrating products	Worksheets displays a dashed line green border around updatable cell ranges from integrating applications such as Workday Payroll and Workday Projects. The formula icon becomes green with a lightning bolt shape at the lower right when a selected workbook cell contains updatable data.
Colored background in column	Live data	• Report data columns are blue. • Note columns are yellow. • Formula columns are green.
Avatar border and cell border	User presence	When more than one user views or edits a workbook simultaneously, Worksheets assigns a different color to each user and shows a cell border in that color to identify the users interacting with the cells. Avatars for the users display above the workbook.

Define Data Validation

Context

You can define rules that determine what data can be entered as values into the cells in your workbook. For example, you might want to make sure users choose a geographic region from a list of valid regions, which your workbook stores in another column.

Notes:

- We don't recommend using resource-intensive functions such as SELECT in a data validation formula. Unexpected behavior might occur.
- Worksheets supports range references, and formulas that result in a list, as data validations.
- As a best practice, be specific with your validation formulas and use absolute references. To make a cell reference absolute, add a dollar sign ($) in front of the column letter and row number in the formula. Examples: $Q$45, $T$67, $A$1:$A$10, etc.
- Uploaded Excel spreadsheets containing validations created in Excel® spreadsheets will continue to work in Worksheets, but you can't view or edit them in Worksheets.
- Worksheets validates *values* that users type into validation cells, but not formulas. For example, if 100 isn't a valid value and a user types it into a validation cell, an error occurs. If the user types =95+5 into the cell, an error doesn't occur.

Steps

1. In the workbook, select the cell or range where you want the data validation to occur, and select Data > Validation.

   The Data Validation dialog displays and the selected cell or range displays in the Cell Range to Validate field.
2. In the List of Values from Formula field, type the formula that determines the values to list in the data validation cell or range.

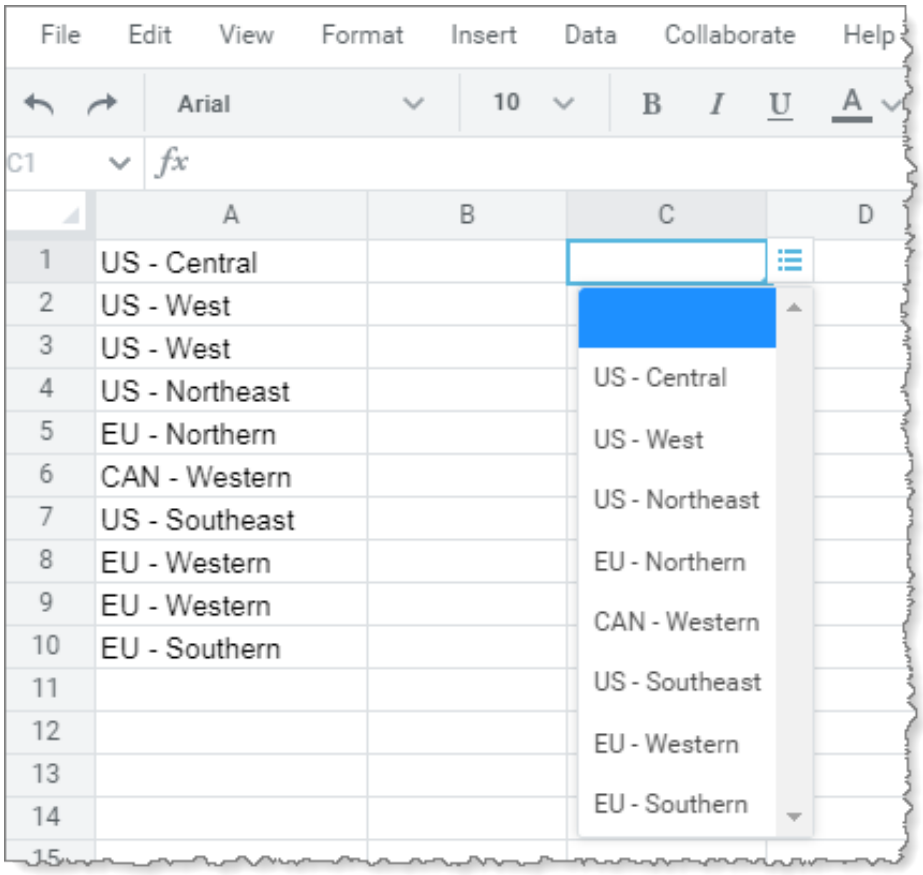
   Alternatively, you can type a list of values surrounded by quotes and separated by commas. Example: "1,2,3".
3. (Optional) With the validation selected, edit your validation by selecting Data > Validation.

Example

In the Cell Range to Validate field, C1 is selected. When the following formula is in the List of Values from Formula field:

=A:A

The unique values in column A display in the data validation list in cell C1:



Merge Workbooks

Context

You can add the sheets from another workbook into an opened workbook if you're the owner or you have edit permission.

When merging workbooks:

- Worksheets doesn't save the original, nonmerged workbook before adding sheets from the second workbook. If you want to keep your original workbook, copy it before merging.
- If updatable data, such as from an entry area, exists in the workbook you're selecting to merge content from, Worksheets copies the values from that area but the data is no longer considered updatable data. This is the same as using a conventional spreadsheet *paste as values* action.

Steps

1. Open the workbook that you want to contain all the content from both workbooks, then select File > Merge Workbook.
2. Navigate to the workbook that you want to merge into the open workbook and click Merge.

Create Charts in Workbooks

Context

You can create a visual representation of data in your workbook using a chart. When you change the corresponding non-array-based workbook data, the chart automatically updates. Optionally, you can select to update the chart when unconstrained array data changes.

If you include dates or times in the chart, make sure you use standard date/time formats; otherwise, Worksheets doesn't recognize the information as dates.

The maximum number of data points that Worksheets can include in a chart is 100,000.

When creating a chart from pivot table data, keep in mind that charts don't understand the structure of a pivot table and you might not be able to create a meaningful chart from a particular set of cells. Additionally, if you have a chart that displays as you want it to, and later you change the rows, columns, or values to include in the pivot, you might not be able to chart it.

Steps

1. Select the cells containing data that you want to use for the chart.
2. Select Insert > Chart.
3. Select the cell where you want the chart to be located, and click OK.

Result

The chart starts at the selected cell and displays in a merged area of cells. The Chart Configuration panel displays

Next Steps

Adjust the chart size as needed by dragging the bottom right corner.

In the Chart Configuration panel, select Auto-update if unconstrained arrays add or remove rows/columns to automatically update the chart if an array size change causes rows or columns to be added or deleted.

Use the panel tabs to customize your chart:

- Appearance: Options include 100% stacked area and 100% bar and column chart types, combination charts, legend location, colors, and more.
- Data: Options include labeling the first row and column, size/color, transposing row/column data, adding a benchmark value, editing/reordering/hiding a data series, and more.
- Axis: Options include x- and y-axis titles and title colors, and hiding axis lines.

If you close the Chart Configuration panel, you can open it again by clicking a chart or by clicking the Chart button in the formula bar.

Reference: Common Workbook Actions and Usage Notes

These notes provide important reminders about day-to-day actions, and describe how to use features in Worksheets that differ from those in other spreadsheet applications.

Action	Notes
Copy a workbook	In the workbook, select File > Copy Workbook. Note: If you have adequate permissions on a workbook and you copy the workbook, you can see the same data as the workbook owner, including

Action	Notes
	Workday data. However, if the copied workbook contains live data from a Workday data source, then as soon as you refresh live data in the workbook, you see only the data you have access to. If you don't have access to one or more of the data source fields, an error occurs when you refresh live data.
Use full-screen mode	Full-screen mode hides the Workday header to display the maximum possible number of workbook rows. To enter full-screen mode, select View > Full Screen. To exit full-screen mode, press Esc or select View > Exit Full Screen. While in full-screen mode, to exit from dialogs, such as the Print dialog, press Alt+` . Workday doesn't support full-screen mode in the Safari browser.
Recover a workbook from the trash	From Drive, select Trash. Select the row containing the workbook and then click the Restore icon. Workday places restored workbooks in the Drive home view. If you remove a workbook that was shared with you (you don't own it), you remove yourself from the list of shared users. The workbook disappears from Drive and it does not display in the Trash folder.
Rebuild a corrupted workbook	If a problem, such as a system error or an interrupted process, causes a workbook to be corrupted, you can return the workbook to a working state using a keyboard shortcut: • Ctrl+Alt+Shift+F9 (Windows) • Command+Option+Shift+F9 (Mac)
Use keyboard shortcuts	Many shortcuts are the same as other spreadsheet products, but because Worksheets is browser-based, several are different. See Help > Keyboard Shortcuts for details; select Help > Keyboard Shortcuts > Print List to view all the shortcuts in a printable format. Keep in mind that keyboard shortcuts are intended for use with US English keyboards. Results might vary with different keyboard layouts.
Undo and redo your changes	To undo a change, select Edit > Undo or press Ctrl+Z (Windows) or Cmd+Z (Mac). To redo a change, select Edit > Redo or press Ctrl+Y (Windows) or Cmd+Y (Mac). Worksheets tracks your most recent 15 changes that you can undo. You can't undo some actions, such as deleting a sheet. If you select to do one of these actions, a confirmation message displays; if you continue with the action, the tracking of your changes resets to that point. You can't undo changes that occurred before this one, unless you previously created a workbook version. Keep in mind that another person editing the workbook might do an action that resets change tracking, which prevents you from undoing your change. You can't undo changes to workbook comments in the Comments panel, and reverting to a previous version doesn't restore comment changes. We recommend creating a workbook version (File > Versions) before doing any of these actions, which can't be undone:

Action	Notes
	• Delete or insert sheet • Change format • Recalculate or Recalculate All • Refresh live data
Hide and unhide rows, columns, and sheets	To hide rows or columns in a workbook sheet, select the rows or columns to hide, then right-click and select Hide. To unhide, click the < > indicator. To hide a workbook sheet, select the triangle icon on the right side of the sheet (tab) to open the menu, then select Hide. To unhide, click the triangle icon at the bottom right of the workbook to display a list of all the sheets, and then click the hidden sheet. Hidden sheets are indicated with a crossed-out eye icon. Note: Hiding content is not a security mechanism. Users with Can View permission for a workbook can use external references to display and use the workbook's hidden content; additionally, they can copy or download the workbook and unhide any content, if you enabled the download or copy option.
Sort columns or rows	Select the columns or rows to sort, then select Data > Sort and then choose a sort order or select Advanced Sort. To use Advanced Sort: 1. Click any cell containing data, or a range of cells, then select Data > Sort > Advanced Sort. If you clicked a single cell, the contiguous range of data surrounding the cell automatically displays in the Range field. 2. Update the Range and Data has headers values if needed. 3. Select Column, Sort On, and Order options in the sort rule that displays. 4. If you want to sort on more than one row, click Add New Sort. Repeat this step to add more sort rules. You can click the Trash icon to delete a sort rule. 5. Click OK. By default, if your workbook sheet contains live data and you select Advanced Sort, Worksheets selects only the live data area for the sort. If you want to include additional columns in the sort, highlight the entire range that you want to sort before selecting Advanced Sort.
General sorting guidelines	General guidelines include: • Unhide columns and rows before sorting. • Use the same data type for all values in a column. • Make sure all rows contain data; don't include blank rows. • Use the alignment selector for cells; don't manually add spaces in front of values.
Sort, when formulas operate on cells that contain references	To keep row data together when sorting, always use absolute cell references. Otherwise, sort results are unpredictable and row data does not stay together. To make a cell reference absolute, add a dollar sign ($) in front of the column letter and row number in the formula. Examples: $Q$45, $T$67, $A$1:$A$10, etc.
Paste content into a workbook	You can use Ctrl+C and Ctrl+V to copy and paste content as values from a desktop spreadsheet application such as Microsoft® Excel® into a workbook, or from one Worksheets workbook to another. However, you can't use the

Action	Notes
	menu option Edit > Paste Special to paste desktop application content into a workbook; Paste Special works only between workbooks that reside in Worksheets. You must upload and convert the desktop spreadsheet into Worksheets by selecting +New > Upload before you can use Edit > Paste Special. The limit when pasting content from one workbook to another is 9 MB.
Rename a sheet	Click the arrow on the sheet tab to rename the sheet or to do other sheet actions. Sheet names can be up to 31 characters long. We recommend using names of 27 characters or less, because when you copy a sheet, Worksheets uses 4 characters to add a space and a numerical increment in parentheses to the sheet name. For example, when you copy the sheet named My Sheet, Worksheets names the copy My Sheet (2).
Open instance details in a new browser tab	When working with live data in a workbook, you can click to view instance details where applicable. To open the instance page in a new browser tab, select the icon in the instance link or use Ctrl+Click (Windows) or Command +Click (Mac).
Auto-fill a formula or value into all cells in a column	You can use a keyboard-based alternative instead of drag-fill: 1. In the cell you want to copy from, enter the formula or value and then press Ctrl+C (Windows) or Command+C (Mac) to copy it. 2. Press Ctrl+Shift+Down Arrow (Windows) or Command+Shift+Down Arrow (Mac) to select all cells in the column, down to the bottom-most cell. 3. Press Ctrl+V (Windows) or Command+V (Mac).
Freeze spreadsheet cells	Drag a freeze handle from the top left corner of the sheet to freeze columns or rows.  Alternatively, you can scroll until the desired column or row is in the first viewable position in the workbook, then freeze at that position by selecting View > Freeze Panes > Top Row. Worksheets freezes at the location of the first visible row in the spreadsheet, which might not be row 1. Similarly, selecting View > Freeze Panes > First Column freezes at the position of the first visible column in the workbook. Note that if your screen display area is too small to show frozen row or column content correctly, you might need to resize your browser window or adjust frozen rows and columns. To unfreeze all columns and rows, select View > Freeze Panes > Unfreeze All.
Define custom cell formats	If you have special requirements for formatting numerical data in a cell, you can use Format > Number > Custom. Worksheets supports the same formatting codes as other popular spreadsheet products. For example, if you want to always display 3 digits to the right of the decimal point in a cell, use the custom formatting string `#.000` .

Action	Notes
Find text in a sheet	Select Edit > Find to open the Find in Sheet panel and search for text on a workbook page.
Break a line in a workbook cell	Double-click the cell in which you want to insert a line break. Click the location inside the selected cell where you want to break the line. Press Alt +Enter (Windows) or Option+Enter (Mac). If you use the workbook formula editor, it removes line breaks, so you need to add the breaks again.
Change a sheet tab color	In the sheet menu, select Tab Color and then choose a color.
Change a workbook font	Worksheets supports a variety of widely available fonts. The default workbook font is Roboto. Keep in mind that Worksheets doesn't support the Calibri font.
Change a workbook language	If you're the workbook owner or you have Edit permission, you can select File > Settings and select a language. If the current setting doesn't match your profile setting, a globe icon displays to the right of the menu bar; hover over or click the globe icon to see the setting.
Change a workbook locale	If you're the workbook owner or you have Edit permission, you can select File > Settings and select a locale. If the current setting doesn't match your profile setting, a globe icon displays to the right of the menu bar; hover over or click the globe icon to see the setting.
Change a workbook time zone	If you're the workbook owner, you can select File > Settings and select a time zone. If the current setting doesn't match your profile setting, a globe icon displays to the right of the menu bar; hover over or click the globe icon to see the setting.
Download workbook data as PDF/Print	Select File > Download as > PDF document (.pdf), select download areas and page formatting options, and click Download.

Reference: Columns, Rows, and Cells

These notes provide important reminders for working with columns, rows, and cells.

Action	Notes
Specify a cell	As in other spreadsheet products, columns are labeled by letter from A to Z, then AA, AB, and so on. Rows are labeled by number starting from 1. Specify a cell by combining its column and its row, such as A1 or E23.
Resize a column	To auto-resize a column based on the amount of text in the cell, double-click the column divider. To manually resize a column, drag the column divider, or select Format > Resize Column and type the desired width. You can type a number with up to 2 digits to the right of the decimal.
Resize a row	To auto-resize a row based on the amount of text in the cell, double-click the row divider. To manually resize a row, drag the row divider.
Select a range of cells	The active cell is shown with a blue border around it. You can select a range of cells by clicking a cell and then dragging to select the range; a blue border

Action	Notes
	then surrounds the range. You can use this feature to place an array formula into the range, or to do an edit operation throughout the range.
Apply a formula to a range of cells	1. Enter a formula into a cell. 2. Hover over the blue dot in the lower right corner of the cell, and then drag to the destination column or row. If a cell contains a formula with a relative cell address (for example, A1), it adjusts automatically for each cell it is dragged to. For example, if a formula in cell C1 is =A1+B2 and you drag-copy it to C2, the formula in C2 becomes =A2+B3. If you drag-copy 1 column to the right, the formula in D2 becomes =B2+C3. If you don't want Worksheets to automatically adjust the references in formulas, use absolute references. To make a cell reference absolute, add a dollar sign ($) in front of the column letter and row number in the formula. Examples: $Q$45, $T$67, $A$1:$A$10, etc. You can do the drag-copy operation on values as well as formulas. If a single cell is selected, its value is copied as is. However, if two consecutive cells are selected, drag-copy fills in cells with a value that is the sum of the second cell plus the difference between the two cells; this works for numbers and dates. You can use the ARRAYAREA formula as a nested formula parameter to pass in arrays (so that the formula updates dynamically as the underlying array changes size). Note: You need to use two operations if you want to use drag-copy to fill a rectangle (range). Example: First fill a column starting from a single cell. Then select the column and use the drag-copy handle to fill the columns.
Clear a cell or range of cells	Select the cells, rows, or columns that you want to clear, then select Edit > Clear > All. It's common to "clear" a cell by typing a space character followed by Enter, Tab, or an arrow key. This doesn't really clear the cell; it leaves a single space character in it, and can cause formulas not to work correctly. Example: Non-empty calls can cause spill errors when calculating formulas, or can cause unexpected results with COUNTBLANK or other issues. There isn't a visual indicator for cells with spaces in them, so when in doubt, manually clear the cells..
Copy a range of data into another workbook	1. Ensure that the range in the workbook where you want to insert the data is blank. 2. In the top left cell of the range in the destination workbook, type a formula similar to this example, which copies the range A1:F50 from the source workbook, where 4d5f6 is the Workday ID of the workbook: =[4d5f6]'Sheet Name'!A1:F50 3. Press Ctrl+Alt+Enter (Windows) or Command+Option+Enter (Mac).
Type into cells in double-byte languages	If you're using a double-byte language such as Chinese, Korean, or Japanese, do one of these steps before starting to type in a workbook cell: • Use the formula bar. • Double-click into the cell. • Press F2.

Reference: Workbook Limits

This table summarizes the primary limits on workbook sizes and data points:

Limit Type	Notes
Workbook overall size	20 million cells or 512 MB, whichever is less. Workbook information about the current percentage of the size limit, and the current number of cells in the workbook, is available by selecting File > Info. It's very important that you take action to reduce the workbook size. Overly large workbooks don't perform well and eventually will fail to open or save. A warning displays when the workbook reaches the size limit. If the workbook reaches 110% of the size limit, a warning message displays and additional changes to the workbook are not saved until you decrease the workbook size. Here are some actions that you can do if your workbook is approaching the limit: • Reduce the number of cells in the workbook that have any data associated with them, such as formulas, values, or formatting. For most cells, the best way to fully clear them is to select the range of cells and select Edit > Clear > All. • For cells in live data, try reducing the number of columns or rows. Consider creating more concise reports to limit the number of rows returned. • For cells in pivot areas, try adjusting the pivot configuration to reduce the overall number of rows and columns produced. • Split large workbooks into two or more smaller workbooks.
Report data exported to a workbook	A specific limit hasn't been established, but the export limit is lower than the overall workbook limit. If your export doesn't work, consider making the report smaller before exporting to a workbook. Instead of exporting to a workbook, consider creating a new workbook and then adding the report data into it using the Add Live Data feature. First you need to enable the report for Worksheets by selecting Enable for Worksheets in the Advanced tab of the custom report. Advanced reports also need to be enabled for Web Services.
Uploads of spreadsheet files (XLSX, XLS, CSV, TSV, JWF, HTML)	20 million cells or 512 MB, whichever is less. Worksheets also follows the Workday overall file upload limit of 30 MB.
Downloads of workbooks	20 million cells or 512 MB, whichever is less.
Report data in a workbook as live data	20 million cells or 512 MB, whichever is less. This limit is based on the amount of data being placed into workbook cells. The *report* can have more than 20 million cells of data; your live data prompt selections and other filters should result in a workbook size that is below the limit.
Report data in a workbook as static values	5 million cells.

Limit Type	Notes
Conditional formatting areas	20 million cells.
Pivot table results	1,000,000
Data points in a chart	100,000
Sheets in a workbook	512
Columns in a workbook	16,384
Rows in a workbook	1,048,576
Characters in a single cell	The limit is generally considered to be 32,767 characters per cell, but it is usually lower in practice. Internally, characters are measured in bytes; each character can require from 1 to 3 bytes. For example, most English characters are 1 byte each, but double-width characters can require up to 3 bytes.
Content copied from workbook to workbook	5 million cells or 9 MB, whichever is less.
Report run time	Advanced reports time out after 30 minutes. Matrix and composite reports don't have this limit.

Reference: Operations on Entire Rows

These notes provide important reminders for doing operations on entire rows.

Action/Feature	Notes
Operate on an entire row	Select a row label (the grid location containing a number to the left of a row). Then select from the Action toolbar or right-click to see a context menu for other actions.
Use the keyboard to cut, copy, or paste a row	• To insert a new empty row, right-click and select Insert Above or Insert Below. Cell references in formulas are automatically renumbered. • To insert a copy of an existing row above or below the current row, right-click the row and select Copy and Insert Above or Copy and Insert Below. Cell references in formulas are automatically renumbered. • To delete a row, select the row, right-click, and select Delete. The row is removed and all other rows are shifted up by 1. Cell references in formulas are automatically renumbered.

Reference: Workbook Quick Statistics

When you select a range of cells in a workbook, including defined names, columns, rows, sheets, and Comments panel references, these formulas display to the right of the formula bar to provide quick reference statistics about your selection:

- SUM
- AVERAGE
- MIN
- MAX
- COUNT

- COUNTA

Based on the types of data in the selected cells, the relevant functions are available in the drop-down list.

The following table lists usage notes for the available functions.

Function	Notes
SUM	Sum of the selected numeric cells. Units in the same dimension (such as length) are converted as needed to determine the sum. Example: If some selected cells use the `ft` unit, and others use the `cm` unit, the SUM function converts the values as needed. The result displays in the units of the topmost selected cell. SUM doesn't evaluate values in different currencies or other units it can't combine, such as `cm` and `liter`.
AVERAGE	Average value of the selected numeric cells. Units in the same dimension (such as length) are averaged. Example: If some selected cells use the `ft` unit, and others use the `cm` unit, the AVERAGE function converts the values as needed. The result displays in the units of the topmost selected cell. AVERAGE doesn't evaluate values in different currencies or other units that it can't combine, such as `cm` and `liter`.
MIN	Minimum value in the selected numeric cells. Units in the same dimension (such as length) are converted as needed to determine the minimum value. Example: If some selected cells use the `ft` unit, and others use the `cm` unit, the MIN function converts the values and shows the result in the units of the topmost selected cell. MIN doesn't evaluate values in different currencies or other units that it can't combine, such as `cm` and `liter`.
MAX	Maximum value in the selected cells. Units in the same dimension (such as length) are converted as needed to determine the maximum value. Example: If some selected cells use the `ft` unit, and others use the `cm` unit, the MAX function converts the values and shows the result in the units of the topmost selected cell. MAX doesn't evaluate values in different currencies or other units that it can't combine, such as `cm` and `liter`.
COUNT	Count of selected numeric cells.
COUNTA	Count of all selected cells containing data. When all cells in the selection contain non-numeric data, then COUNTA is the only available statistic.

Reference: Automatic Data Updates in Workbooks

A workbook can contain these kinds of data:

- Stand-alone data that might or might not originate in Workday.
- Data from one or more Workday report data sources.
- Data from an integrating application such as Workday Planning or Workday Projects, if applicable.

This table describes the data updates that occur, based on the kind of data you're working with:

Source of Workbook Data	Automatic Data Update	Notes
Standalone data, such as: • A new workbook that you create in Worksheets. • A new workbook that you create from an uploaded spreadsheet such as Microsoft® Excel®. • A new workbook that you create when you export Workday application data to a workbook. Create the workbook by selecting the Export to Worksheets icon from a grid.	None.	When you export from a grid into a workbook using Export to Worksheets, Workday doesn't preserve formatting such as bold text.
Data from a Workday report data source (Add Live Data) in a workbook.	Depends on the Insert selection (as Static Values or as Live Data Area).	If you select As Static Values during the insert, Workday: • Doesn't update the workbook based on changes in the report. • Saves changes you make to data that you inserted into the workbook. If you select As Live Data Area during the insert, Workday: • Updates the workbook based on data in the Workday report, either when you manually refresh live data or when a scheduled refresh occurs. • Doesn't save changes that you make later in the inserted data, but does save changes outside the live data range. When Worksheets refreshes live data, it also recalculates any volatile functions, and formulas affected by changed workbook data.
Data from entry areas in integrating applications such as Workday Projects or Workday Payroll.	Updates in both directions, initiated from the integrating application.	

Reference: Workbook Actions Available Based on Permissions

This table summarizes the actions available for workbooks based on the user's workbook permission level. You specify permission levels when you share a workbook.

Action	Can View	Can Comment	Can Edit	Owner	Notes
View workbook	X	X	X	X	
View workbook information	X	X	X	X	
Copy workbook	X	X	X	X	Applies for the Can View and Can Comment permissions only if the workbook owner selected the Commenters and viewers can copy, download, and print option when sharing the workbook.
Download workbook (if enabled in tenant settings)	X	X	X	X	Applies for the Can View and Can Comment permissions only if the workbook owner selected the Commenters and viewers can copy, download, and print option when sharing the workbook.
Print workbook (if enabled in tenant settings)	X	X	X	X	Applies for the Can View and Can Comment permissions only if the workbook owner selected the Commenters and viewers can copy, download, and print option when sharing the workbook.
View pivot table details	X	X	X	X	You can access the details for a single pivot table value from the context (right-click) menu. Select Show Details. The Create Sheet button displays in the details dialog only if you have edit permission.
View workbook user presence	X	X	X	X	
Use a private filter	X	X	X	X	
Use a shared filter	X	X	X	X	
View and add comments		X	X	X	
View list of users that the workbook is shared with		X	X	X	
Remove (self) from shared workbook access	X	X	X		

Action	Can View	Can Comment	Can Edit	Owner	Notes
Add/edit external references			X	X	Applies for the Can Edit permission only if the workbook owner selected the Editors can share option for the workbook when sharing it.
Define names for workbook cells or ranges			X	X	
Share workbook or change share permissions			X	X	Applies for the Can Edit permission only if the workbook owner selected the Editors can share option for the workbook when sharing it.
Edit content including reverting changes			X	X	
Sort workbook content			X	X	
Use a public filter			X	X	
Merge workbooks			X	X	
Recalculate data			X	X	
Rename workbook			X	X	
Refresh live data			X	X	Applies for the Can Edit permission only if the workbook owner didn't select the Only owner can refresh live data option in File > Settings. Note that during a live data refresh, the workbook isn't available to you or anyone you shared with workbook with.
Define data validation rules			X	X	
Copy pivot data formula			X	X	From the context (right-click) menu, you can copy the formula that produces a single pivot table value. Select Copy Pivot Data Formula.
Schedule live data refresh				X	
Protect workbook ranges				X	
Change workbook owner settings				X	
Remove to trash and restore from trash				X	

Reference: Worksheets Keyboard Shortcuts

Keyboard shortcuts are intended for use with US English keyboards. Results might vary with different keyboard layouts.

Note: In addition to standard keyboard shortcuts, we provide access to Worksheets functionality using a user interface overlay of action keys typically referred to as KeyTips. You enable KeyTips by pressing the Alt key on Windows or Option on Mac. After you press Alt or Option, small icons depicting keyboard keys are overlaid in the workbook. Press a key to do the displayed action.

- Edit on page 46
- Format on page 47
- Data on page 47
- Panels on page 48
- Selection on page 48
- Navigation on page 49

Edit

A double asterisk (**) in a cell indicates that the default Mac OS key assignment conflicts with this shortcut. To change your settings, select Apple > System Preferences > Keyboard > Shortcuts. Click Mission Control, then clear the shortcut option.

Action	Windows	Mac
Cut	Ctrl+X	Command+X
Copy	Ctrl+C	Command+C
Copy as External Reference	Ctrl+Alt+R	Command+Option+R
Copy Workbook ID	Alt+F3	Option+F3
Find	Ctrl+V	Command+F
Find and Replace	Ctrl+H	Command+Shift+H
Paste	Ctrl+V	Command+V
Undo	Ctrl+Z	Command+Z
Redo	Ctrl+Y	Command+Y
Insert Row Above	Ctrl+Alt+=	Command+Option+=
Insert Column Left	Ctrl+Alt+=	Command+Option+=
Insert New Sheet Right	Shift+F11 or Alt+F2	Shift+F11** or Option+F2
Delete Row	Ctrl+Alt+-	Command+Option+-
Delete Column	Ctrl+Alt+-	Command+Option+-
Download as PDF/Print	Ctrl+P	Command+P
Paste Into Selected Cells	Ctrl+Enter	Command+Enter
Break Line in Cell	Alt+Enter	Option+Enter
Submit Unconstrained Array Formula	Ctrl+Alt+Enter or Ctrl+Alt+Shift +Enter	Command+Option+Enter

Action	Windows	Mac
Submit Constrained Array Formula	Ctrl+Shift+Enter	Command+Shift+Enter

Format

A double asterisk (**) in a cell indicates that the default Mac OS key assignment conflicts with this shortcut. To change your settings, select Apple > System Preferences > Keyboard > Shortcuts. Click Mission Control, then clear the shortcut option.

Action	Windows	Mac
Clear Formatting (Example: bold text)	Ctrl+\	Command+\
Bold	Ctrl+B	Command+B
Italic	Ctrl+I	Command+I
Strikethrough	Ctrl+Shift+X	Command+Shift+X
Underline	Ctrl+U	Command+U
Wrap	Alt+W	Option+W
Center Align	Ctrl+E	Command+E
Apply Outside Border	Ctrl+Shift+7	Command+Shift+7
Remove Borders	Ctrl+Shift+-	Command+Shift+-
Default Number	Ctrl+Shift+1	Command+Shift+1
Default Currency	Ctrl+Shift+4	Command+Shift+4** or Command+Option+4
Default Percent	Ctrl+Shift+5	Command+Shift+5** or Command+Option+5

Data

A double asterisk (**) in a cell indicates that the default Mac OS key assignment conflicts with this shortcut. To change your settings, select Apple > System Preferences > Keyboard > Shortcuts. Click Mission Control, then clear the shortcut option.

Action	Windows	Mac
Show All Formulas	Ctrl+`	Ctrl+`
Copy Workbook ID	Alt+F3	Option+F3
Recalculate	F9	F9
Recalculate All	Ctrl+Alt+F9	Command+Option+F9
Refresh All Live Data	Ctrl+Alt+F5 or Alt+F1	Command+Option+F5** or Option+F1
Fill from Above Cell	Ctrl+Shift+6	Alt+F6
Fill from Left Cell	Ctrl+Shift+,	Command+Shift+,
Insert Above Formula	Ctrl+‘	Command+‘

Action	Windows	Mac
Auto Sum	Alt+=	Option+=
Enter Current Date	Ctrl+;	Command+;
Enter Current Time	Ctrl+Shift+;	Command+Shift+;
Toggle Reference Display	F4 or Ctrl+Alt+T	F4** or Command+Option+T
Open Prompts in Cell	Alt+Down Arrow	Option+Down Arrow

Panels

Action	Windows	Mac
Open Versions Panel	Ctrl+F8	Command+F8
Open Chart Panel	Ctrl+F11	Command+F11
Open Comments Panel	Ctrl+F2	Command+F2
Open Context Menu	Shift+F10	Shift+F10
Open Errors Panel	Ctrl+Shift+F1	Command+Shift+F1
Open Formula Editor	Ctrl+F1	Command+F1
Open Functions Library Panel	Ctrl+F9	Command+F9
Open Live Data Panel	Ctrl+F12	Command+F12
Open Defined Names Panel	Ctrl+F3	Command+F3
Open Pivot Table Panel	Ctrl+F7	Command+F7
Open Protected Ranges Panel	Ctrl+F10	Command+F10

Selection

A double asterisk (**) in a cell indicates that the default Mac OS key assignment conflicts with this shortcut. To change your settings, select Apple > System Preferences > Keyboard > Shortcuts. Click Mission Control, then clear the shortcut option.

Action	Windows	Mac
Select to Region Edge	Ctrl+Shift+Arrow	Command+Shift+Arrow
Select Column	Ctrl+Space or Ctrl+Alt+6	Command+Space** or Command+Option+6
Select Row	Shift+Space	Shift+Space
Select All	Ctrl+A or Ctrl+Shift+Space	Command+A or Command +Shift+Space
Extend Selection to Final Cell	Ctrl+Shift+End	Command+Shift+End
Select Active Cell's Array	Ctrl+/	Command+/
Select Only Active Cell	Ctrl+Backspace	Command+Backspace
Hide Selected Rows	Ctrl+Alt+9	Command+Option+9
Hide Selected Columns	Ctrl+Alt+0	Command+Option+0

Action	Windows	Mac
Unhide Selected Rows	Ctrl+Alt+Shift+9	Command+Option+Shift+9
Unhide Selected Columns	Ctrl+Alt+Shift+0	Command+Option+Shift+0
Open Instance Link Tooltip	Alt+Enter	Option+Enter

Navigation

Action	Windows	Mac
Enter Full-Screen Mode	Alt+F11	Option+F11
Close Dialog When in Full-Screen Mode	Alt+`	Alt+`
Access KeyTips Overlay	Alt	Alt
Jump to A1	Ctrl+Home	Command+Home
Jump to Final Cell	Ctrl+End	Command+End
Jump to Region Edge	Ctrl+Arrow	Command+Arrow
Jump to Next Sheet	Ctrl+Shift+Page Down	Command+Shift+Page Down
Jump to Previous Sheet	Ctrl+Shift+Page Up	Command+Shift+Page Up
Open Context Menu	Shift+F10	Shift+F10

Reference: Mobile Features for Worksheets

Workday provides an iOS app and an Android app. Workbooks are available for viewing in the mobile apps, from the Drive worklet.

Calculations in Workbooks

Concept: Formulas and Formula Bar Actions

To edit cells, you can select a cell and click inside the formula bar, or double-click a cell and edit the data directly in the cell. We recommend using the formula bar as the main data entry area.

Action/Feature	Notes
View reference information for all available formulas	Click the Function icon (fx) to open the Functions Library panel.
Enter a formula	You can: • Double-click in a cell and type the formula into the cell, starting with = (the equals character). • Place the cursor in the formula bar and start typing the formula. When you type a function name that Worksheets recognizes, a function description displays; when you start typing parameters, syntax information displays.

Action/Feature	Notes
	• From the Functions panel, find the function you want and then click the + to its left to insert it into the formula bar. • Click the Formula Editor icon to open the interactive formula editor, and enter your formula. Any existing content in the formula bar displays in the editor. Important note: The editor icon doesn't display when a live data formula is in the formula bar and the formula has an active connection to the Data Wizard, or when the active cell is in a pivot table. • The formula editor doesn't support constrained array formulas. To make Worksheets handle the content of the cell as text instead of a formula, type a ' (single quote character) before the = character; everything after the ' is treated as plain text. The ' character doesn't display in the cell but it displays in the formula bar.
Select a range to enter into a formula	When entering a formula argument, you can click a cell or range of cells to insert its address into the formula. Then type the next character for the formula, which is usually a colon when entering a range reference, or a comma when separating arguments. For example, for =SUM(A1:C5) you can drag the selection to automatically include the range from A1 to C5 in the formula. If the cell you want to insert in a formula is on another sheet, click the tab for that sheet, click the cell in that sheet, then navigate back to the original sheet. When you use the auto-fill feature in the formula bar, Worksheets automatically adds references from different workbook *sheets* into the formula. However, the formula bar doesn't support opening two workbooks in separate browser tabs and auto-filling external references.
Submit (run) a formula from the formula bar or the cell containing the formula	If you expect the formula to return a single value, press Enter to run the formula. If you expect the formula to return multiple values (it's an array formula), use the appropriate keyboard shortcut: • For an unconstrained array formula (Worksheets dynamically determines the output range), press Ctrl+Alt+Enter (Windows) or Command+Option+Enter (Mac). • For a constrained array formula (you select the output range), press Ctrl+Shift+Enter (Windows) or Command+Shift+Enter (Mac).
Submit (run) a formula from the formula editor	When you open the formula editor for a formula, the editor detects if the formula is a standard scalar (single value, non-array) formula or unconstrained array formula, and submits it appropriately when you click Save & Close. The formula editor doesn't support constrained array formulas. If you select Save & Close without using Evaluate first, the Evaluate action runs before closing the panel. When you close the editor panel, the formula displays again in the formula bar as a single line.

Action/Feature	Notes
Enter numbers into a formula	You can enter numbers in several formats, such as: • A standard format such as 123, -4.5, 27.0001. • Scientific format, such as 1E10, 12.4e-4, 8.2E+34. • Accounting format for negative numbers, such as (12.45), (12E2). • Formatted with thousands separators, such as 12,234.6789. • Formatted as a percentage, such as 12.3%, -.23E3%. • Formatted as a currency, such as $12.45, $50, $34.33, ($1,456.99), ($12). • Dates and times. The sheet might not display the number exactly as it was typed, depending on the current cell's formatting rules and other factors, but the value is preserved. To make Worksheets handle the format of the cell as text, type a ' (single quote character) before the = character; everything after the ' is treated as plain text. The ' character doesn't display in the cell but it displays in the formula bar. This is especially useful when you are entering dates and you want to preserve the formatting you typed instead of displaying it in a format such as DD/MM/YYYY.

Concept: Array Formulas in Workbooks

An array is a set of data contained in more than 1 spreadsheet cell. An array visually looks like a simple range of cells, but it's actually a special structure that you work with as a unit. From a Worksheets perspective, live data areas and entry areas are examples of arrays. A formula that deals with a single cell, not an array, is a scalar formula; the formula returns a result in a single cell.

An array can be one-dimensional, containing values in a single row or column; or two-dimensional, containing a rectangular set of values in both columns and rows. You can see horizontal arrays represented in text with braces around them and commas between the values, such as in the formula ={1,2,3,4}. Vertical arrays have semicolons between the values, such as ={5;6;7;8}. Two-dimensional arrays have both, such as ={1,2,3;4,5,6}.

An array formula is a single formula that does calculations for the values in an array, and then returns an array of results. The array result area is called the spill area. Array formulas use the same syntax as regular formulas. You can:

- Submit a formula to output the results into an array. All the cells in the array are a single unit that you work with as a single entity. This is a multicell array formula.
- Submit a formula in a single cell to operate on an array, while containing the result in a single cell. This is a single cell array formula. Single cell array formulas are much less commonly used compared to multicell array formulas.

You can use many common functions, such as SUM and COUNT, in an array formula. Several functions, such as TRANSPOSE, are meaningful only when they operate on an array.

Keep these considerations in mind when using array formulas:

- Array formulas perform better. It's faster to use a 10x100 multicell array formula instead of 1,000 separate formulas.
- Using a multicell array formula helps prevent you or other users from accidentally overwriting a cell formula.

- You can't edit an individual cell of an array, except for the top-left cell (also called the root cell) in the spill area. If you select another cell in the spill area, the formula is visible in the formula bar, but you can't change it. To update the formula, select the root cell, then change it as needed. Worksheets automatically updates the rest of the spill area when you press Enter.

Constrained and Unconstrained Array Formulas

In a formula, if you specify the exact range of cells to put results in, you're using a constrained array formula. Use a constrained array formula only if you know the specific range that should contain the result set. If you add or remove data in the range, you must modify the array formula to account for it.

Worksheets offers a more flexible method of working with array data. If you're working with live data, such as Workday report data or entry area data, the data array can change size whenever you refresh the live data. The array formula needs to adapt to those changes. Worksheets handles variations in array size using unconstrained array formulas. When you submit a formula without specifying the specific array size for the output, allowing Worksheets to determine the array result set dynamically, you're using an unconstrained array formula.

We use the term unconstrained in a few different contexts:

- We call a formula such as GROUPBY an unconstrained array formula because it's intended to have variable output.
- A live data array such as a Workday report data set is an unconstrained array because the array size can grow or shrink when the live data refreshes.
- A range reference such as B:B is an unconstrained column reference because you're not specifying the number of cells containing data in the column.

Submitting Array Formulas

Submit array formulas from either of these locations:

- The formula bar.
- The cell containing the formula. Before submitting, double-click in the cell to make sure it is active.

Submit array formulas using these keyboard shortcuts:

- To submit an *unconstrained* array formula, press Ctrl+Alt+Enter (Windows) or Command+Option +Enter (Mac). Results display in all required cells in the range. If there aren't enough empty cells to display the complete results, an error occurs. Make sure that enough empty cells are available to contain the result.
- To submit a *constrained* array formula, press Ctrl+Shift+Enter (Windows) or Command+Shift+Enter (Mac). Results display only within the selected range. You see the same behavior in other common spreadsheet products.

Concept: Formula Writer and Explainer

The Worksheets Formula Writer generates a workbook formula based on your description of what you want the formula to do. You can insert the resulting formula into the workbook, or you can use the generated formula as a guide and make edits to it. You need edit access to the workbook to use the Formula Writer.

Generative AI and How Worksheets Uses It

Generative artificial intelligence (Gen AI) encompasses a range of AI systems that create many kinds of content. Worksheets uses a generative AI model when generating and explaining formulas so that you receive the most accurate results possible. Formula generation is more like an art than a science so you should test the results that you receive from the Formula Writer. You can insert the formula and run it, interpret the results, and then edit and resubmit the formula description if needed, to get the formula

you want. You can also use the generated formula as a starting point or an idea generator for your own formula.

To get the best results from the Formula Writer:

- Include as much detail as possible in the formula description.
- Include structured references, table and column names, column/row/cell references, and so on.
- Use quotation marks around any strings (text).

Formula Explanations

You can have Worksheets explain to you how a formula works. To better understand formulas in your workbook, select a cell containing a formula, then right-click and select Explain Formula.

Create a Formula Using the Formula Writer

The Formula Writer enables you to describe what you want to do in a Worksheets formula and then it generates a formula that you can insert into the workbook.

To use the Formula Writer, Worksheets must be enabled for GenAI features. Additional setup steps might be required depending on your organization's subscription service agreement, and the Enable Worksheets GenAI Features option must be enabled in the Workday Conversation Settings section on the Edit Tenant Setup - System task.

1. Click the cell where you want to insert the generated formula.

   You can’t place a generated formula into a Live Data area.
2. Open the Formula Writer using one of these methods:
   - Click the Formula Writer sparkle icon ✨ in the toolbar.
   - Navigate to View > Panels > Formula Writer.
   - Right-click the cell and then select Formula Writer.
3. Type instructions for the formula you want into the Formula Description field.
4. Click Generate. Worksheets generates a formula and displays it in the Result area and an explanation of the formula displays below the results.

   If your formula description isn't detailed enough to generate a formula, a message displays asking you to change the description.
5. Review the formula and edit it as needed.

   Generated formulas often don't include correct cell references and might have other issues. The process of generating formulas is iterative, and relies on you to assess the output and edit the formula as needed.
6. Click Insert into <Cell> to place the formula into the selected cell and automatically run it.

Example Formula Descriptions

This table shows examples of well-written formula descriptions, along with their resulting formulas:

Formula Description	Formula
What is the average of column A, rounded to two decimal places.	ROUND(AVERAGE(A:A),2)
Count how many cells have the text "Ready for Promotion" in column S.	COUNTIF(S:S,"Ready for Promotion")
Calculate the sum of the product of values in cells A1:A5 and B1:B5.	SUMPRODUCT(A1:A5,B1:B5)

Formula Description	Formula
In A1:A5 I have a list of names and in column B1:B5 I have a list of corresponding ages. What is the name of the person in D1 based on my list of names and ages?	INDEX(A1:A5, MATCH(D1, B1:B5, 0))
Give me a list of unique values from column A.	MS.UNIQUE(A:A)
I have a list of fruit names in column A and their corresponding prices in column B. How do I calculate the price of an apple?	VLOOKUP("Apple",A:B,2,0)
Divide A1 by A2. If there is an error, return 0.	IFERROR(A1/A2,0)
What is the last day of the current month?	EOMONTH(TODAY(),0)
Find the "Employee Name" of the employee with the lowest "Prior Bonus" in the dataset "CompDetails."	INDEX(CompDetails[Employee Name],MATCH(MIN(CompDetails[Prior Bonus]),CompDetails[Prior Bonus],0))
Find the "Employee Name" of the employee with the lowest "Salary Increase" among those hired after January 1, 2010, in the dataset "CompDetails."	INDEX(CompDetails[Employee Name],MATCH(MINIFS(CompDetails[Salary Increase],CompDetails[Hire Date],">1/1/2010"),CompDetails[Salary Increase],0))
Calculate the average of the "New Salary" column to find the average salary of all employees.	AVERAGE(CompDetails[New Salary])
Calculate the average of the "Proposed Bonus %" column to compute the average proposed bonus percentage for all employees.	AVERAGE(CompDetails[Proposed Bonus %])
Calculate the total sum of the "Prior Bonus" column to determine the total prior bonus for all employees.	SUM(CompDetails[Prior Bonus])
Find the total sum of the "Salary Increase" column to determine the total salary increase for all employees.	SUM(CompDetails[Salary Increase])
Identify the highest value in the "New Salary" column to find the highest new salary among all employees.	MAX(CompDetails[New Salary])
Return the list of "Employee Name" where the cost center is "41200 Payroll".	FILTER(CompDetails[Employee Name],CompDetails[Cost Center]="41200 Payroll")
Find the "Employee Name" and "Supervisor" for employees with a "New Salary" greater than 100,000.	FILTER(CompDetails,CompDetails[New Salary]>100000)
Determine the "Employee Name" and "Supervisor" for employees with a "Proposed Bonus %" greater than 0.02.	FILTER(CompDetails,CompDetails[Proposed Bonus %]>0.02,CompDetails[Employee Name],CompDetails[Supervisor])
Use Total Base Pay in cell C2 and Time Type in cell D2 to calculate the Hourly Rate. If the value in cell D2 is "Part time", the formula divides by 4.	C2/IF(D2="Part time",4,IF(D2="Full time",8))

Formula Description	Formula
If the value in cell D2 is "Full time", the formula divides by 8.	
Calculate the median price from column R for the values in column Q that are "Electronics".	MEDIAN(IF(Q:Q="Electronics",R:R))

Concept: Automatic and Manual Calculation in Workbook Formulas

It's useful to understand the actions that cause Worksheets to calculate or recalculate the formulas in your workbook.

Calculations depend on the formula type and the action you're doing. Recalculation time includes the time spent running reports, executing formulas, and applying conditional formatting.

This table summarizes whether Worksheets recalculates workbook data or not:

Recalculation Setting	Formula/Workbook Type	Event/Action	Recalculate Formula?
Automatic	Volatile • INDIRECT • NOW • NOWTZ • OFFSET • RAND • RANDBETWEEN • TODAY	Change any cell in workbook.	Yes.
Automatic	Non-Volatile	Change any cell that this cell depends on.	Yes.
Automatic	ARRAYAREA	A formula uses ARRAYAREA and the array referenced by ARRAYAREA changes. The anchor cell (top left cell) of the array referenced by ARRAYAREA changes.	Yes.
Automatic	RANDCONST		No. This non-volatile function is recalculated only when: • A scheduled live data refresh runs. • You select Data > Recalculate. • You select Data > Recalculate All.
Automatic	SELECT (with live data)	Manually refresh live data range. Live data refresh schedule runs.	Yes.
Automatic	SELECT as a generic formula	Change any cell that the formula references.	It depends.

Recalculation Setting	Formula/Workbook Type	Event/Action	Recalculate Formula?
			If you use the parameter section to reference data, the formula runs when the data changes. If you change data in the SELECT statement portion, the formula doesn't run again.
Automatic	ONCE	Place (wrap) any formula in the ONCE function.	No.
Automatic	Copied workbook	First time you open copied workbook.	No. Select Data > Recalculate All to update the workbook.
Automatic	Converted Microsoft Excel file	First time you open the uploaded/converted workbook.	No. Select Data > Recalculate All to update the workbook.
Manual	Any	Any.	No. Worksheets never recalculates formulas unless you manually initiate a recalculation. This is applicable for all formulas, including volatile formulas. (New formulas are calculated when you first submit them but they're not recalculated.)

The default recalculation setting in workbooks is Automatic.

You might want to prevent automatic recalculation in some situations. Example: If you're making many changes to a workbook with complex formulas, you might want to wait until you finish all your changes before recalculating the formulas.

To specify a calculation option, select File > Settings.

To manually recalculate formulas, use one of these actions:

- Data > Recalculate calculates any formulas in the workbook where the data has changed.
- Data > Recalculate All calculates all formulas in the workbook, whether or not the data has changed since the last calculation. Recalculate All also updates external references in consumer workbooks

Notes:

- When you do an action that doesn't cause a recalculation, such as uploading a Microsoft Excel file to Worksheets and opening the converted workbook the first time, select Data > Recalculate All to update the workbook.
- Live data in a workbook isn't affected when you use the recalculation settings or actions. You can refresh live data manually by selecting Data > Refresh All Live Data, by selecting Refresh Now in the Live Data panel, or by setting up a scheduled refresh using Data > Schedule Live Data Refresh.

Concept: Goal Seek

Goal Seek is a type of optimization tool; it determines the value that you need to enter in a single input cell to produce the result that you want in a dependent (formula) cell.

Example: You are buying a car and you can afford a loan payment of $500 per month. Goal Seek can tell you the purchase price that you can afford.

Enter beginning values and a payment formula into the workbook:

	A	B	Notes
1	Purchase Price	29900	The amount you would like to spend.
2	Loan Term in Months	48	
3	Interest Rate	4.25%	
4	Payment	=PMT(B3/12,B2,-B1)	Formula to calculate the payment.

Navigate to Data > Goal Seek and enter these values:

- Set Cell: B4
- To Value: 500
- By Changing Cell: B1

Optionally you can enter a minimum and/or a maximum value that you want Goal Seek to return.

When you click OK, Goal Seek changes the value in B1 to the purchase price value that causes the payment in B4 to equal $500.

You can discover more about the goal seek functionality on the Internet, along with information about the golden section, golden ratio, and tau.

Concept: Locale, Language, and Time Zone in Workbooks

Worksheets uses a combination of the settings in your Workday user profile, along with workbook-specific settings, to determine how to display information.

If you're the workbook owner or you have edit permissions, you can change the locale or language of an open workbook in the File > Settings menu. To change the time zone, you must be the workbook owner. Worksheets supports only the values listed in the drop-down menu.

If the current workbook locale, language, or time zone doesn't match your profile setting, a globe icon displays to the right of the menu bar; hover over or click the globe icon to see the settings.

Keep in mind that when someone puts a value into a cell, it will stay in that same language. If other users with different language settings view the cell, they will see the value in the language that it had when it was originally entered in the cell.

This table summarizes the settings and which values they affect:

Setting	Effects
Workday Settings: Account > Preferences > User Profile Display Language User Profile Locale User's Time Zone	Language, locale formatting, and time zone of elements external to the values in the cells of the workbook. Examples: Text and values in menu items, dialogs, and side panels.
File > Settings > Language	Language of user interface elements related to the values in the cells of the workbook. Examples:

Setting	Effects
	• Tooltips, including tooltip text next to error values (such as #VALUE). • Informational and error messages that display in the workbook.
File > Settings > Locale	Formatting of values in the cells within the workbook. Examples: • Numerical formatting, such as comma or period separators for thousands and decimals. • Formatted date values, such as DD/MM/YY or MM/DD/YY.
File > Settings > Time Zone	Time zone of the workbook, which can affect data such as live data values, or results for certain functions such as NOW(), which returns the current date and time.

Concept: Dates and Times

Note: When possible, Worksheets automatically displays dates using standard date formatting. In some cases, such as formula results, Worksheets shows the date as a serial number; you can use Format > Number > Date to display the result using date formatting.

In the US English locale, you can specify dates as described in this section. Other locales might have different entry formats.

You can specify dates as shown in the table below, where:

- MM represents a month number from 1-12.
- DD represents a day number in a month from 1-31 depending on the month.
- YY represents a 2 digit year where 30 = 1930, 29 = 2029, and 00 = 2000.
- YYYY represents a 4 digit year (from 1899 to 9999).
- MMM represents the abbreviated month name (such as Jan, Feb, and Mar).
- MMMM represents the full month name (such as January, February, and March).

Format	Date
MM/DD/YY	7/16/18
MM/DD/YYYY	7/6/2018
MM-DD-YY	7-6-18
MM-DD-YYYY	7-5-2018
YYYY-MM-DD	2014-07-08
DD-MON-YY	12-JAN-12, 1-DECEM-99
DD-MON-YYYY	12-MAY-2012, 14-MARCH-1998
MON-DD	Jul-4, Feb-28

You can specify times as shown in the table below, where:

- hh represents an hour of the day from 0-23
- mm represents a minute from 0-59
- ss represents a seconds value from 0-59
- ms represents milliseconds from 0-999

- am/pm is AM or PM (case insensitive)

You can append am or pm to the time. Example: 11:45am, 4:55pm.

Format	Example
hh:mm	22:30, 11:45
hh:mm:ss	13:14:15, 9:00:0

If the AM/PM designator is missing, Worksheets defines times before 12:00 as AM and times after 12:00 as PM. If the hour value is 13 or higher, Worksheets defines the time as 24-hour (military) time and does not allow the AM/PM designator.

You can combine any of the date formats with any of the time formats to get a specific date and time combination such as 7/4/2014 13:15 or 09-Jul-2000 10:26:46 am.

Note: Normally if you type a date such as Dec 1, 2018 into a cell, Worksheets converts it into the datetime value 12/01/2018. To display the value as text in the cell, type a single quote (') before the value: `'Dec 1, 2018`.

Concept: Circular References

Caution: In virtually all cases, a circular reference indicates an error that you must correct. We strongly recommend not enabling iterative calculations, particularly in workbooks that contain live data; doing so obscures data errors that can lead to poor performance and unexpected behavior.

Circular references occur in a workbook when a cell refers to itself, or to another cell that references the original cell. The formula requires a result that is dependent on the original cell so an error occurs.

A red icon  indicates that Worksheets didn't finish calculating; the workbook is in a state where some formulas didn't fully run and are showing stale values. You can hover the cursor over the icon to see a tooltip with information about the problem. There are two common causes for a workbook to prematurely stop calculating: either the calculation limit was reached, or one or more circular references exist in the workbook. If the problem is caused by a circular reference, you can click the red icon to navigate to the cell location of the reference. If you have more than one circular reference, Worksheets navigates to the location of the first one.

In some rare situations, you might want to allow circular references.

Example: You have a basic cost of $1,000,000 for a project. A consultant earns a 5% fee based on the total project cost, which is $1,000,000 plus the consultant fee. Because the fee is part of the calculation, it's a circular reference in the spreadsheet, so you need to allow iterative calculation to occur.

A	B	C
Basic Cost	$1,000,000.00	
Consult Fee	=B3*C2	5%
Total Cost	=SUM(B1:B2)	

To enable circular references in a workbook to calculate:

1. Select File > Settings.
2. Select Enable Iterative Calculation and enter values for:
    - Maximum iterations: Enter a value of 100 or fewer iterations.
    - Maximum change: Enter a maximum change value. The default is .001. Worksheets does the circular calculation for the number of iterations that you specified, or until the result changes by less than this value.

If you enable iterative calculation but the iteration calculation limit is reached, a yellow icon displays in the workbook.

If you open a workbook containing circular references, and your recalculation setting is Manual, a message displays; select Data > Recalculate All to update the formula.

Reference: Formula Editor

The formula editor is an interactive editor that enables you to quickly build, evaluate, and debug complex nested formulas. Example: You can change a function argument and then see the effect it has on the results.

Open the editor by clicking the Formula Editor icon at the right side of the formula bar.

Note:

- The editor icon doesn't display when there is an active connection to the Data Wizard. (You can't use the formula editor for live data formulas.)
- The formula editor doesn't support constrained array formulas.

This table summarizes the key features of the editor:

Feature	Notes
Formatting	The row you're actively editing displays in a different color. Syntax highlighting colors are: • Functions: green • Quoted strings: purple • Cell references: blue • Operators: brown • Error text: red • Default: black Indentation of lines and creating new lines is automatic, based on rules such as whether a formula contains nested formulas. When an argument extends beyond the width of the panel, Worksheets wraps the line, and indents lines after the first line by 1 space.
Evaluations	When you click Evaluate, the editor recalculates values and reformats the lines in the formula as needed, such as to redo indentations. The default is to evaluate a formula as the type you initially opened in the editor; you can evaluate a formula as a scalar (non-array) or as an unconstrained formula by selecting the appropriate option from the drop-down menu next to the Evaluate button. The calculated results of the main formula and nested formulas display in a green indicator at the top of the formula. The workbook remains available for editing, such as for dependent cells, while you are editing a formula. The active formula remains in the panel. Example: If you're editing a formula in cell A1, and the results depend on a value in B1, you can click into the B1 cell and change the value; the formula editor continues to display the formula in A1 in the panel. Click Evaluate to update the results based on the change you made in B1.
Error assistance and prevention	To help prevent errors, Worksheets automatically inserts matching grouping characters such as parentheses, braces, and quotes. If you

Feature	Notes
	place the cursor to the left or right of one of these characters, the matching character is highlighted. A green indicator at the top of the panel shows the active cell and its calculated value. If an error exists, the indicator becomes red and displays an error message. To view help text about the error, hover over the message. If a syntax error exists in a line of the formula, an exclamation mark icon displays to the left of that line, and the line is highlighted.
Submit the formula and close the panel	Click Save & Close. Notes: When you open the formula editor for a formula, the editor detects if the formula is a standard scalar (single value, non-array) formula or an unconstrained array formula, and submits it appropriately when you click Save & Close. The editor prevents you from losing unsaved changes if you select to close the panel; you can choose to save the changes or discard them. If you select Save & Close without using Evaluate first, the Evaluate action runs before closing the panel. When you close the editor panel, the formula displays again in the formula bar as a single line.

Reference: Array Formula Keyboard Shortcuts

Use these keyboard shortcuts for array formulas in Worksheets:

Keyboard Shortcut	Description

Ctrl+Alt+Enter or Ctrl+Alt+Shift +Enter (Windows)
 	
Command+Option+Enter (Mac)
	Use for unconstrained array formulas.
	
	To place results for a formula in all applicable cells, select a single cell and submit the formula using the shortcut.
	
	If the workbook doesn't include enough empty cells to display the complete results, an error occurs.
	
	Note: Use the keyboard shortcut to submit changes that you make in a pivot table formula. A pivot table is an example of an unconstrained array formula because it can shrink or grow depending on the underlying data.
	
	Example:
	
	1. Select cell A1.
	2. In the formula bar, type `={1,2;3,4}`.
	3. Press Ctrl+Alt+Enter (Windows) or Command+Option+Enter (Mac).
	
	Workday shows these 4 values in 2 rows and 2 columns:
	
	`1 2`
	`3 4`

Keyboard Shortcut	Description
	You can't edit individual cells in the results range.

Ctrl+Shift+Enter (Windows)
Command+Shift+Enter (Mac)
	Use for constrained array formulas. This shortcut is the same in Worksheets and Excel.
	Select a range of cells, then submit the formula using the shortcut.
	Workday displays results only in the selected range.
	Don't use this shortcut when you're working with Workday live data, entry area data, or an array with an undefined or unknown size. Use the shortcut for unconstrained arrays instead.
	When you select more cells than necessary, Workday displays an #N/A error in the extra cells. When you don't select enough cells to display the complete results, the remaining results don't display.
	Example:
	1. Select cells A1-B3.
	2. In the formula bar, type `={1,2;3,4}`
	3. Press Ctrl+Shift+Enter (Windows) or Command+Shift+Enter (Mac).
	Workday shows these values in 3 rows and 2 columns:
	`1 2`
	`3 4`
	`#N/A #N/A`
	Workday displays the #N/A error because you selected more cells than the number of results.
	You can't edit individual cells in the range.

Ctrl+Enter (Windows) Command+Enter (Mac)	You can't use the Ctrl+Enter keyboard shortcut in entry areas. This shortcut is the same in Worksheets and Excel. Technically the shortcut isn't an array formula shortcut; it's similar to a paste: it places the same formula or data into a range of cells. Unlike array formulas, using Ctrl+Enter doesn't prevent you from editing individual cells in the range. Example: 1. Select cells A1-B3. 2. In the formula bar, type `={1,2;3,4}`. 3. Press Ctrl+Enter (Windows) or Command+Enter (Mac). Workday shows these 6 values in 3 rows and 2 columns: `1 1` `1 1` `1 1`

FAQ: Workbook Performance

What's the upper limit on workbook size, and what affects this limit?

The overall limit is 20 million cells per workbook. Keep in mind that performance impacts are much more dependent on workbook formulas and dependencies than on workbook size. For a complete list of workbook limits, see Reference: Workbook Limits on page 40.

This table lists workbook characteristics that can have a significant impact on performance, along with some tips for mitigating that effect:

Performance Impacts	Tips
Using many instances of the same formula instead of an array formula.	Instead of using drag-fill to place many occurrences of the same formula in cells, use an unconstrained formula when possible. Remember to submit the formula using the unconstrained formula keyboard shortcut. For details, see Concept: Array Formulas in Workbooks on page 51.
Dependencies within and between workbooks. A change in 1 cell might cause calculations in many other cells. If more than 1,000 cells are directly or indirectly affected, performance might degrade	When using external references, minimize the use of producer workbooks that are referenced by consumer workbooks, particularly if the producer workbooks are large.
Calculation-intensive functions, such as VLOOKUP, SUMIFS, and so on, when changing a range value.	Worksheets provides indexed functions that are optimized for performance. These WD functions create a b-tree index for the specified range or array and provides better search performance for repeated use on the *same* data. If you're doing a lot of lookups on the *same* set of data, the b-tree index is cached and used again and again. We recommend using these functions instead of their non-indexed counterparts whenever possible: • WD.AVERAGEIF / WD.AVERAGEIFS • WD.COUNTIF / WD.COUNTIFS • WD.MAXIF / WD.MAXIFS • WD.MINIF / WD.MINIFS • WD.MVLOOKUP • WD.SUMIF / WD.SUMIFS • WD.VLOOKUP To optimize workbook performance, we recommend when you reference a range in the WD functions that you write your formulas and specify the correct lookup ranges using absolute references. This ensures that less b-tree indexes will need to be created and cached. Absolute references ensure cell references in formulas remain constant regardless of where the formulas are copied or filled. To make a cell reference absolute, add a dollar sign ($) in front of the column letter and row number in the formula. Examples: $Q$45, $T $67, $A$1:$A$10, etc.
Volatile functions that run any time Worksheets recalculates, such as RAND, NOW, TODAY, OFFSET, INDIRECT, and INFO. Actions that cause the workbook to recalculate include:	You can wrap a volatile function in the ONCE function to prevent recalculation. Don't create a defined name that uses a volatile function and then use that defined name in many places. Instead, place the volatile function in a cell, and create a defined name as a reference to that cell.

Performance Impacts	Tips
• Changing cell values or formulas. • Inserting or deleting rows or columns. • Adding or changing defined names. • Filtering. • Hiding or unhiding content.	Example: To set up and use a defined name for the current date: 1. Place the formula `=EDATE(TODAY(),0)` in cell A3 of Sheet1. 2. Create the defined name todays_date. In the formula field type `=Sheet1!$A$3`. 3. In formulas where you want to insert the current date, use the defined name todays_date.

These workbook characteristics typically have a minimal effect on performance

- Workbook total size
- Number of cells
- Number of sheets (tabs)
- Cross-sheet formulas
- Formula sequence
- Summary tabs

Sometimes I see an alert telling me to come back to my workbook later. Why?

When Worksheets takes more than several seconds to complete an operation, an alert like this displays. For example, it might display while inserting or refreshing live data, or when recalculating a lot of formulas. The alert enables you to dismiss the workbook so you can do other work while the action completes. If you open a workbook that's already running a refresh or a Data Wizard process, you also see the alert.

Sometimes I see a Processing... indicator for a while, in the Drive file list. What determines whether I see the indicator or not?

When you create a workbook from existing data - such as from a report or an integrating application - Worksheets displays a Processing... indicator while adding the data into the workbook. The time required can vary from seconds to hours, depending on several factors, including:

- The number of calculations and formulas.
- The size of the file or Workday report.
- The complexity of the integrating application generation process.
- Whether the workbook contains live data from any Workday reports, or other resource-intensive processes that the tenant might be running. (Aside from live data, Worksheets processes are independent of any other tenant processes.)

After Worksheets starts generating the workbook, you can't cancel it or start generating a new integrating application workbook. However, if a workbook is in a Processing state for more than 24 hours, Worksheets automatically cancels it, enabling you to move the workbook to the trash or regenerate it.

My workbook calculations are too slow. What improvements can I make?

- If you're making many changes to a workbook with complex formulas, you might want to prevent recalculations until you finish all your changes. You can temporarily change the recalculation mode to Manual from the File > Settings dialog.
- Remove any data that you don't need for analysis.
- Minimize external references among workbooks.
- Minimize the number of formulas.
- Use whole-column references (such as A:A) sparingly.
- If you're using lookup formulas, try to organize your data so the calculations are on sorted data. Lookup functions are much faster with sorted data, and even faster when you can avoid using the exact match option.

How long should it take to upload an Excel spreadsheet?

Although most uploads finish within a few seconds, extremely large Excel spreadsheets might require several minutes. The progress bar in Drive lets you know that the process is occurring. You can work in other workbooks while the upload continues in the background.

Worksheets seems to run faster or slower depending on which computer I run it on. Why?

Rich web applications such as Worksheets can reduce performance if a computer is running other resource-intensive applications concurrently, or if processing capacity is low for another reason. Although it's not possible to recommend specific configurations because of the number of variables involved, you might find that in your environment you need to establish your own hardware configuration requirements.

Collaboration in Workbooks

Share Workbooks

Workbook owners and authorized editors can share workbooks with specific users or groups, or turn on link sharing and then give users a link (URL) to the workbook. You can share workbooks only with users who have access to the Worksheets application.

For more information about sharing workbooks, see the Drive User Guide.

Sharing Type	Description
With unspecified users, using a link (URL)	Note: If you enable link sharing for a workbook, any user with access to the Worksheets application can access the workbook if they have the URL. You can share workbooks from Drive or by clicking the Share button in the workbook. Click the desired sharing permission, then click the Link Sharing toggle to enable link sharing, and copy the URL to give to other users. If you turn off link sharing for a workbook, the change takes effect immediately. Cross-workbook formulas that had access to the reference only through link sharing change to #REF errors. Unlike sharing with individual users, when you share an item using a link, Workday doesn't send notifications. We recommend that you notify people that you shared an item with them; for example, you can email them a link to the workbook.
With specific users	You can share workbooks from Drive or by clicking the Share button in the workbook. When you share an item, the user receives a Workday notification and an email (if the Workday administrator enabled email notifications).
With groups	You can share workbooks from Drive or by clicking the Share button in the workbook.

Sharing Type	Description
	Unlike sharing with individual users, when you share an item with a group, Workday doesn't send notifications. We recommend that you notify people that you shared an item with them; for example, you can email them a link to the workbook.

As soon as you share a workbook, the user interface adapts to show only the actions that users can do, based on their permission level.

Workbook owners have additional options in the Advanced tab when sharing workbooks with specific users:

Option	Description
Edit: Provides Share, Copy, and Download/Print options. Comment: Provides Copy and Download/Print options. View: Provides Copy and Download/Print options.	By default, only the owner can share, copy, and download/print workbooks. Notes: • If you hid content in the workbook before sharing it, remember that users who have only View or Comment permission can copy or download the workbook and unhide the content, if you enable the copy or download/print option. • A tenant setting can disable the ability to download and print workbooks for all users.

Users that you share the workbook with can initially see any unhidden data included in the workbook, including Workday data. If the shared workbook is based on live data from a Workday data source, updates to the workbook's live data affect what those users can see:

- If the user has permission to refresh live data, then as soon as they do the refresh, they see only the data they have access to. Additionally, all other users see the live data based on the security associated with the user who refreshed the data. If the user doesn't have access to one or more of the data source fields, an error occurs when they try to refresh the live data.
- Whenever the workbook owner does the Refresh All Live Data action, or whenever a scheduled live data refresh occurs, all live data in the workbook refreshes. The owner's view of the live data is available again to users that the workbook is shared with.

Note: Sharing occurs at a workbook level; you can't share individual sheets or ranges in workbooks. Workbook owners can prevent other users from editing ranges within workbooks by protecting them.

Manage Workbook Versions

You can create a named version of a workbook to save a snapshot of accumulated changes up to that point.

To create a named version, select File > Versions and enter a name in the Add Version to Workbook dialog.

To view a list of created versions, open the Versions panel by selecting View > Panels > Versions.

To revert a workbook to a particular named version, select View > Panels > Versions, then click Revert on the version you want to return to. Worksheets automatically creates a new named version with an incremented version number at the top of the version list.

You can't view a previous version of the workbook without reverting to that version.

When you revert to a named version, tracking of your individual changes is reset; that is, tracking of the 15 changes that you can undo starts again at zero after you revert to a named version.

Workbook comments are independent of any versioning status: the user sees in the Comments panel all comments entered for the workbook, regardless of the version where the comment was added. Restoring to a previous version of the workbook doesn't undo or change the Comments panel data.

We recommend creating a named version before doing any of these actions, which can't be undone:

- Delete or insert sheet
- Change format
- Recalculate or Recalculate All
- Refresh live data

Note: You can't use a named version to revert changes to entry areas.

Manage Workbook Comments

Workbook owners, and users with Can Edit or Can Comment access, can add comments to a workbook.

To add a general comment about the workbook, click the Comments icon, type your comment, and click Post. Adding a comment this way creates a comment without adding a cell reference.

To associate a comment with a particular cell in the sheet, right-click the cell that you want to comment on and select Insert Comment. After you enter the comment, a View Reference link displays below the comment. You can use this link to quickly find the cell where the comment was added.

If you right-click a cell that contains an existing reference, View Comment displays in the right-click menu instead of Insert Comment.

To tag a person in a comment, type the @ character and start typing the user's name in the comment, then select the user. You can tag users only if you already shared the workbook with them. When you tag a person in a comment, they receive a Workday notification, and (if enabled) an email.

To edit or delete a comment, select an option from the comment options menu (three dots) on the right side of the comment. You can edit or delete only your own comments.

You can use a subset of Markdown syntax in comments.

Text Type	Markdown Example
bold	**This text is bold.**
italic	_This text is italicized._
strikethrough	~~This text is marked out.~~
hyperlink	[This text is blue and links to example.com](http://example.com)

FAQ: Collaboration and Security in Worksheets

How do I test my security configuration for Worksheets?	To see which data displays in workbooks, we recommend signing in as multiple users with different security settings. Then use the Data Wizard to insert report data into workbooks and share the workbooks among the users.

	Proxy is supported in Worksheets. But keep in mind that in an application workflow such as HCM, where the application creates the workbook (not Drive), proxy usage might not be supported. Worksheets supports workbooks created or edited as a proxy user (or any other non-Worker user type) in one tenant and then migrated as a WXF file to another tenant.
Is there a global configuration setting that enables some users to view workbooks and others to modify them?	No. Workbook sharing settings are for individual workbooks. The security group's View and Modify settings that you assign when initially configuring Worksheets aren't related to viewing and editing permissions.
Can I share a Worksheets workbook with anyone inside Workday even if they can't access the original data?	Yes, if they have access to the Worksheets application (they are in the *Worksheets* security domain). Users that you share the workbook with can initially see any unhidden data included in the workbook, including Workday data. All users that you share with can see the same data that you see in the workbook. This functionality exists because Worksheets is intended as a collaboration tool where group of users solve business problems based on the same set of data. However, if the shared workbook is based on live data, refreshing the workbook affects what other users can see: • As soon as a user refreshes live data in the workbook, they see only the data they have access to. Additionally, all other users see the live data based on the security associated with the user who refreshed the data. • Whenever a user does the Refresh All Live Data action, or whenever a scheduled live data refresh occurs, all live data in the workbook refreshes. The live data is visible again to other users, based on the security associated with the user who refreshed the data or who set up the schedule. • If a user has permission to refresh live data but they don't have access to one or more of the data source fields, an error occurs when they try to do the refresh. If you share a workbook containing live data with a user and they make a copy of the workbook, as soon as they refresh live data in the workbook they see only the data they have access to. To ensure that a workbook always displays only the data that a user has access to, you can create a workbook template and then distribute the template to users.

Can I hide content in a workbook to prevent anyone from seeing it in any situation?	No. Hiding content is useful to reduce distractions in your workbook, such as hiding a sheet when you're using its data for calculations in other sheets. However, hiding content isn't a security mechanism. Users with Can View permission for a workbook can use external references to display and use the hidden content. Additionally, they can copy the workbook and unhide any content, if you enabled the Commenters and viewers can copy, download, and print option.
What are the colored cell borders in the workbook?	Colored cell borders are called presence indicators. When users view or edit a workbook simultaneously, Worksheets assigns different colors to the users. Worksheets highlights cells with that color as the users interact with the cells. User avatars also display for any users currently in the workbook.
When I insert Workday report data into a workbook, is the workbook auto-shared with the people I shared with report with?	No, only you have access to the workbook, and it contains the data you have access to from the report.

Live Workday Data in Workbooks

Concept: Live Data in Workbooks

When you add report data to a workbook and you choose to keep the workbook in sync with the original report data, that data is called live data. You can also add report data to a workbook without keeping the data in sync; we refer to this data as static data.

Option	Description
Live Data	The inserted data in the workbook maintains a connection to the Workday report, providing 1-way updates from the report to the workbook. (This setting doesn't enable writing data from the workbook to the report.) You can't manually edit the inserted live data in the workbook.
Static Values	Inserted data in the workbook doesn't maintain a connection to the original report. The data is a snapshot of the report data, as of the time you insert it. You can edit the inserted data in the workbook. The maximum number of cells that Worksheets can insert as static values is 5 million (5,000,000).

Live Data Tables and Structured References

Live data areas in Worksheets workbooks are similar to Excel tables. When Worksheets creates a live data area, it assigns a table name to the area. Optionally you can rename the table. Using a special syntax, you can create references to the data in a live data area using the table and column name; these references are called *structured references*, and the range of cells is automatically updated whenever the live data is refreshed. For more details about structured references, see Concept: Structured References in Workbooks on page 72.

Formula Columns and Note Columns in Live Data

To work with formula columns:

- To add a new formula column without using the data wizard, right-click the header of any live data column to add a new formula column to the right or left. Adding a formula column using this method causes a live data refresh.
- To add a new formula column without using the data wizard, type the formula into the 2nd row of a column adjacent to the live data area and press Enter. Worksheets automatically adds the formula into the live data area and applies the formula to all rows in the live data area, without doing a live data refresh. (Skip a column if you don't want to add the formula into the live data as a formula column.)
- Type a formula into the cell below the header and press Enter; Worksheets applies the formula to all rows in the live data area.
- Double-click the column header cell to edit its name.

To work with note columns in the live data area:

- You must have already set up a key column (field) using the data wizard to see this option in the menu. A key uniquely identifies the data in each row. The key column that you select must contain unique values, such as Employee IDs. The value must not change over time, and the value must exist in *only one row* of the data. Worksheets uses the key to match note rows to live data rows. Make sure your key is unique; if it isn't, notes will be lost.
- Type notes as desired into the individual cells. To paste text into a cell, double-click in the cell first.
- Double-click the column header cell to edit its name.
- You can right-click the header of any live data column to add a new note column to the right or left. Adding a note column using this method causes a live data refresh.
- Keep in mind that during a live data refresh, Worksheets doesn't preserve formatting for note columns.
- If rows are added during a live data refresh, Worksheets preserves the alignment of notes with their corresponding rows. If rows are removed during a live data refresh – for example, you delete a prompt – any notes related to the deleted prompt are removed but the data is saved in Workday. If you add the prompt again later, Worksheets shows the notes again in their corresponding rows.
- If you type a note into the 2nd row of a column adjacent to the live data area and press Enter, Worksheets automatically adds a note column into the live data area without doing a live data refresh. (Skip a column if you don't want to add the column as a note column.)

Filtering

If you have an active filter in live data and you refresh the data, Worksheets automatically reapplies the filter criteria to the live data area.

Sorting

You can sort live data areas and entry areas in a workbook. Optionally, your sort can include contiguous columns in the workbook that are outside the live data area or entry area. By default, if your workbook sheet contains live data and you select Advanced Sort on the Data menu, Worksheets selects only the live data area for the sort. If you want to include additional columns in the sort, highlight the entire range that you want to sort before selecting Advanced Sort.

When you refresh live data in a workbook, Worksheets preserves the sort order that you set in the Order By option in the data wizard, but doesn't preserve standard sorting within the workbook (using Data > Sort).

Formatting

In the Data Wizard Select Columns page, you can use the Column Options menu to set the data format for your live data columns (if your live data is from an Advanced report). If you set the formatting here,

Worksheets preserves the data format when the live data is refreshed. Data formatting isn't available for Note columns.

We recommend that you not change any font/style settings (such as bold text) in the workbook live data. When you refresh the live data, Worksheets resets font/style formatting. If you want to change your live data's appearance, you can use the ARRAYAREA function on a different sheet to insert the data from the live data area, and format those rows and columns as desired.

Group Column Headings

If your original report definition contains Group Column Headings, those values become column headings in your workbook. If you want to change the column headings later, you can do so using the SELECT formula to copy the data into a different sheet and assign new column names.

Here's an example formula that copies data from Sheet1 into the current sheet and assigns the new column names ID and Name:

=SELECT("SELECT `~Employee~ ID` `ID`, `Employee Name` `Name` from ? ",ARRAYAREA(Sheet1!A1))

Although this example is related to Group Column Headings, you can use it to change any live data column headings.

Concept: Global Prompts in Live Data

In a workbook with multiple live data areas that are based on an advanced, composite, or matrix report, you don't need to manage each set of live data prompts separately. You can select a set of prompts to be global prompts, and Worksheets then applies those prompt values to all live data areas in all sheets (reports) in the workbook when the live data refreshes.

In Worksheets, a source report is a designated report within a workbook that serves as the main point of reference for global prompts. This setup allows you to manage shared global prompts across multiple reports more efficiently.

Worksheets supports a subset of prompt field types as global prompts: Boolean, Date, Date Time, Numeric (integers only), Text, Multi-Instance, or Single Instance. When mapping a prompt to a global prompt, only prompts of the same type are available. Keep in mind that when you're mapping a prompt to a global, all report prompts of the same type are available for selection; be careful that the prompt you select makes sense in the context of the report. For example, the Company prompt and the Worker prompt have the same data type (Text), but you would not want to map a Company prompt as the global prompt for a Worker.

We strongly recommend creating a table name for the live data area that you're using as the global prompt source. The table displays as a prompt prefix when mapping.

To assign global prompts for a new live data area and map those prompts to other areas in the workbook:

1. Assign a source report: In the Data Wizard Select Options page, select the Set as Source Report option. You can select only one source report for a workbook.
2. Map other live data area prompt values to the global prompts, using one of these methods:
    - In the Data Wizard Select Prompt Values page, select Map to Existing Prompt, then select the defined global prompt.
    - Click the Global Prompts icon in the workbook toolbar.

For existing live data areas, select to edit the report prompts by clicking the Edit link in the live details panel. Then follow the steps above.

In the live data details panel, you see a Source identifier if the live data area is being used as the global prompt source report. Additionally, if you are in the Select Prompt Values wizard page for the live data source report, a Source identifier displays below any prompts that are currently assigned as mapped

prompts in other live data areas. Keep in mind that any changes you make to a source global prompt will impact other live data areas.

Undo is supported only when you make the change using the Data Wizard. You can't undo any changes made using the global prompts panel.

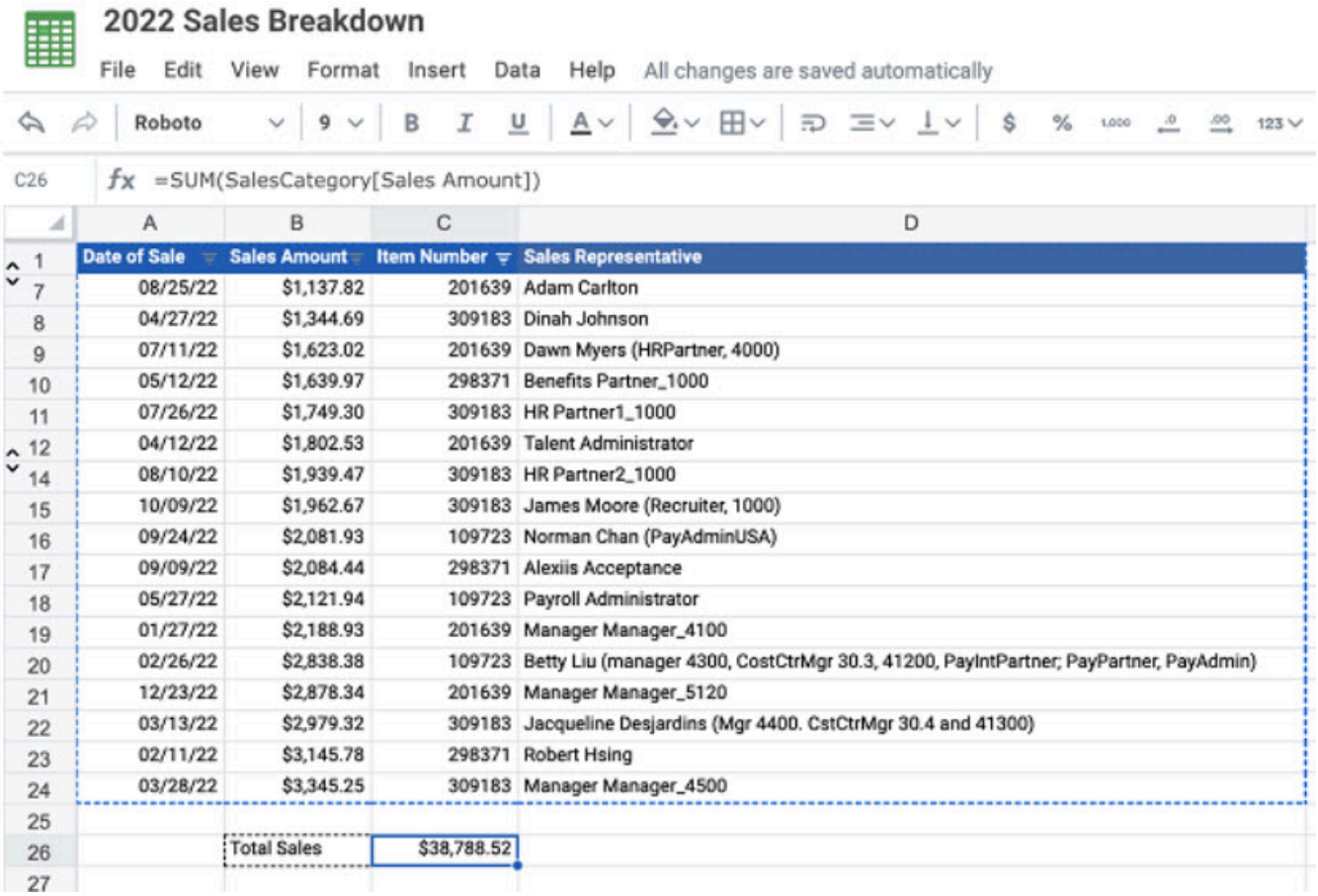
Concept: Structured References in Workbooks

Live data areas in Worksheets workbooks are similar to Excel tables. When you add a live data area to a workbook based on a Workday advanced report, Worksheets creates a table that contains the live data. By default, Worksheets assigns a live data table name using the word Report and an incrementing digit. Example: Report1.

Optionally, you can create your own name for the live data table, either at the time you run the data wizard or later. Worksheets supports table names only for advanced reports. If you do rename the live data table, remember that the table name must be unique in the workbook and also can't be the same as any defined names or pivot table names.

After running the data wizard, you can use a special syntax incorporating the table name and column name to specify dynamic references in formulas instead of using literal cell ranges. This combination of a table and column name is a *structured reference*. Worksheets updates the range of a structured reference automatically as needed when the live data is refreshed.

Example: Instead of the formula =SUM(B1:B24) you can use =SUM(SalesCategory[Sales Amount]) where SalesCategory is the table name and Sales Amount is the column name. The SalesCategory table name isn't displayed in the live data area; however, if you were to display the live data panel, it would display there.



	A	B	C	D
1	Date of Sale	Sales Amount	Item Number	Sales Representative
7	08/25/22	$1,137.82	201639	Adam Carlton
8	04/27/22	$1,344.69	309183	Dinah Johnson
9	07/11/22	$1,623.02	201639	Dawn Myers (HRPartner, 4000)
10	05/12/22	$1,639.97	298371	Benefits Partner_1000
11	07/26/22	$1,749.30	309183	HR Partner1_1000
12	04/12/22	$1,802.53	201639	Talent Administrator
14	08/10/22	$1,939.47	309183	HR Partner2_1000
15	10/09/22	$1,962.67	309183	James Moore (Recruiter, 1000)
16	09/24/22	$2,081.93	109723	Norman Chan (PayAdminUSA)
17	09/09/22	$2,084.44	298371	Alexiis Acceptance
18	05/27/22	$2,121.94	109723	Payroll Administrator
19	01/27/22	$2,188.93	201639	Manager Manager_4100
20	02/26/22	$2,838.38	109723	Betty Liu (manager 4300, CostCtrMgr 30.3, 41200, PayIntPartner; PayPartner, PayAdmin)
21	12/23/22	$2,878.34	201639	Manager Manager_5120
22	03/13/22	$2,979.32	309183	Jacqueline Desjardins (Mgr 4400. CstCtrMgr 30.4 and 41300)
23	02/11/22	$3,145.78	298371	Robert Hsing
24	03/28/22	$3,345.25	309183	Manager Manager_4500
25				
26		Total Sales	$38,788.52	
27				

Keep these considerations in mind:

- For live data areas that were created before Workday 23R1, you need to edit the live data area to add a table name. You can do this from the live data panel by clicking Edit Live Data or by clicking any of the Edit links in the panel.
- If you change a table name or column name later, Worksheets updates existing structured references *in the workbook*; however, if you have an external reference that contains a table or column name, Worksheets doesn't automatically update the consumer workbook. You need to update the reference manually.
- Worksheets doesn't support:
    - Using structured references (table and column names) to define conditional formatting rules. We do support applying conditional formatting rules to data that was populated using a structured reference.
    - Uploading files to Worksheets that were originally created in Excel and that contain structured references.
    - Using structured references in a NOTIFYIF formula.

Structured Reference Syntax Auto-Fill

When you are typing a formula in the formula bar or in a cell, you don't need to remember the syntax for the structured reference. Depending on what you type into the formula, or what you select in the live data table, Worksheets generates and displays the appropriate syntax.

Worksheets generates live data structured reference syntax when you select:

- An individual cell in the live data table.
- Column headings. Select one or more heading rows.
- Columns of live data. You can select either the heading along with the data, or select only the data.
- The entire table. Select the entire live data area including headings.

Additionally, if you type a table name and then type a left bracket in the formula, the column names and special identifiers display for selection.

Structured Reference Syntax Guidelines

- Table names don't need to be inside quotes or square brackets.
- Column names don’t need to be inside quotes, but you do need to surround them with square brackets.
- Use the pound symbol (#) and indicators described below to refer to a special subset of a cell range.
- When including a formula column in a live data area, the formula doesn't need to include the table name, but it does need to include the at symbol (@) preceding any column names.
- Remember to press the unconstrained array keyboard shortcut when submitting a formula: Ctrl+Alt +Enter or Ctrl+Alt+Shift+Enter (Windows) or Command+Option+Enter (Mac).

Special Identifiers for Structured References

Specifier	Example	Refers To
#All	[#All]	The entire table, including column headers and data.
#Data	[#Data]	Only the data rows, no header.
#Headers	[#Headers]	Only the header row, no data.
#This Row or @		Only the cells in the same row as the formula. You can't combine these specifiers with any other item specifier. This must be used in a live data formula column before every header.

Specifier	Example	Refers To
		Example: =SUM([@Current Salary],[@Proposed Bonus])

Syntax Examples

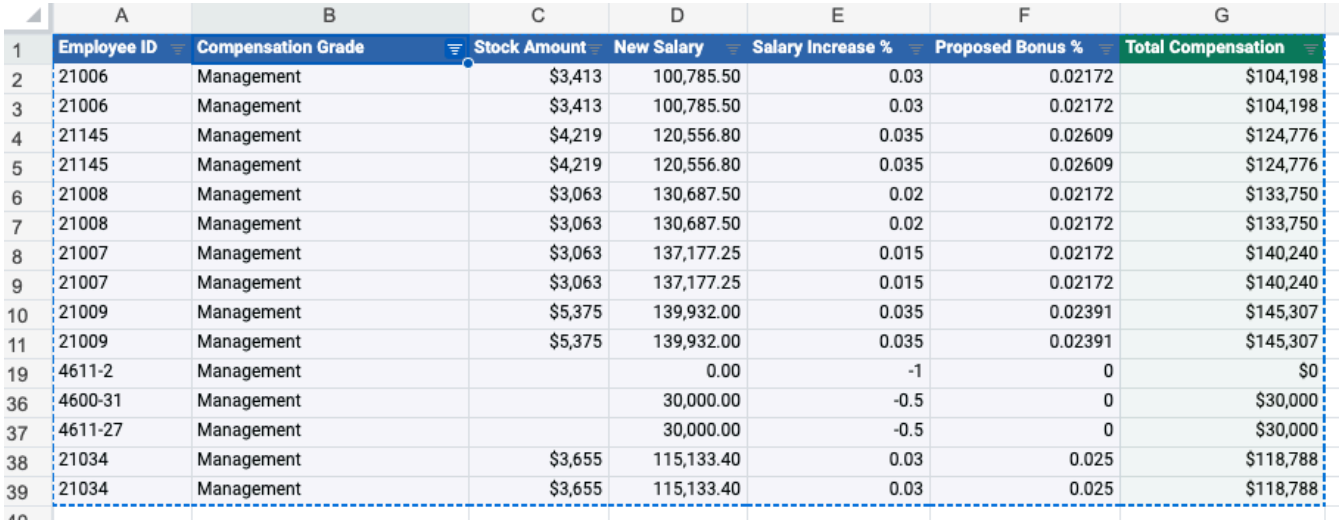
Basic example:

=SUM(CompDetails[New Salary],CompDetails[Stock Amount])

Notes:

- The formula adds the values of the two columns together and displays the results in a new column. The table name is CompDetails.
- The column names are New Salary and Stock Amount (enclosed in brackets).

For the table of examples below, we added live data to a workbook and named the table CompDetails. Here's an image showing the column names.



	A	B	C	D	E	F	G
1	Employee ID	Compensation Grade	Stock Amount	New Salary	Salary Increase %	Proposed Bonus %	Total Compensation
2	21006	Management	$3,413	100,785.50	0.03	0.02172	$104,198
3	21006	Management	$3,413	100,785.50	0.03	0.02172	$104,198
4	21145	Management	$4,219	120,556.80	0.035	0.02609	$124,776
5	21145	Management	$4,219	120,556.80	0.035	0.02609	$124,776
6	21008	Management	$3,063	130,687.50	0.02	0.02172	$133,750
7	21008	Management	$3,063	130,687.50	0.02	0.02172	$133,750
8	21007	Management	$3,063	137,177.25	0.015	0.02172	$140,240
9	21007	Management	$3,063	137,177.25	0.015	0.02172	$140,240
10	21009	Management	$5,375	139,932.00	0.035	0.02391	$145,307
11	21009	Management	$5,375	139,932.00	0.035	0.02391	$145,307
19	4611-2	Management		0.00	-1	0	$0
36	4600-31	Management		30,000.00	-0.5	0	$30,000
37	4611-27	Management		30,000.00	-0.5	0	$30,000
38	21034	Management	$3,655	115,133.40	0.03	0.025	$118,788
39	21034	Management	$3,655	115,133.40	0.03	0.025	$118,788

Structured Reference	Content Included	Implied Cell Range
CompDetails or CompDetails[#Data]	All data in the table, without column headers.	A2:G39
=CompDetails[[#All],[Proposed Bonus %]]	All cells in the Proposed Bonus % column	F1:F39
=CompDetails[[#Headers],[Salary Increase %]]	The header of the Salary Increase % column	E1
=CompDetails[[#All],[Compensation Grade]:[Salary Increase %]]	All cells between the Compensation Grade and Salary Increase % columns, inclusive, with headers	B1:E39
=CompDetails[[Compensation Grade]:[New Salary]]	The data between the Compensation Grade and New Salary columns, inclusive, no headers	B2:D39
=CompDetails[[#Headers],[Employee ID]:[New Salary]]	The headers of the columns between Employee ID and New Salary	A1:D1

Structured Reference	Content Included	Implied Cell Range
=CompDetails[[#Headers],[#Data], [Compensation Grade]]	The header and data of the Compensation Grade column	B1:B39
=CompDetails[[#This Row], [New Salary]] or =CompDetails[@New Salary]	The cell at the intersection of the current row and the New Salary column	D5 (if the current row is 5)

Comparing Structured References and Defined Names

Defined names are similar to structured references in some ways, but there are notable differences. This table summarizes the key differences:

Defined Names	Structured References
You can create defined names for any workbook range.	Structured references refer only to data in a live data area, for advanced report data only.
A defined name specifies a fixed range of cells across columns and rows, or you can use the Formula field to define a name for a formula, constant, or non-contiguous set of cell ranges.	A structured reference combines a table name and column names to specify a dynamic range of cells in one or more live data columns.
If the defined name is in a live data area, the range of cells remains the same even if the data changes during a refresh.	The data referred to by the reference is dynamic and is updated automatically after a live data refresh.
A defined names panel ("right pane") shows all defined names so you can view and manage them.	Structured references aren't standalone entities - they're managed as part of the live data area - so they don't need a separate panel.
You can use defined names in Slides presentations.	Structured references aren't visible or usable in Slides presentations.
If you use a defined name in a Slides presentation linked table and later you change its name or range, you need to refresh the linked data in the presentation to see the updated data. If you use a defined name value in a Slides presentation linked value, and then you edit a defined name's range so that the linked value is no longer included, you need to remove and re-add the linked value.	You can change a table name or column name. If you do so, Worksheets updates existing structured references in the workbook; however, if you have an external reference (consumer workbook) that contains the previous table or column name, Worksheets doesn't automatically update it. For more information about external references, see Concept: Using External References to Refer to Data in Other Workbooks on page 17.

Concept: Creating Reports to Insert into Workbooks

You can add data from one or more Workday reports into a workbook. Adhere to these requirements and best practices for any reports that you want to add data from.

The Workday report must:

- Be an advanced, matrix, or composite report. Worksheets doesn't support inserting other report types, such as Simple, Standard, or Xpresso.

- Be enabled for Worksheets. Select Enable for Worksheets in the Advanced tab of the custom report. Enabling a report causes it to display in the list of available reports, when you start the Data Wizard in a workbook.
- Be enabled as a web service (required only for advanced reports). Select Enable as Web Service in the Advanced tab of the custom report.
- Include fields from the primary business object and any necessary related business objects. If you use related business objects, your report must contain group column headings.
- Use these field types: Boolean, Numeric (integers only), Text, Date, Date Time, Multi-Instance, or Single Instance. Worksheets doesn't support other types (Currency, DateTimeZone, Time).
- Not contain duplicate column labels (Column Heading Override) or XML aliases.
- Not contain fields that use the Do Not Prompt at Runtime prompt option.

For optimal performance as the report runs and Worksheets inserts the data into the workbook:

- Use indexed calculated fields whenever possible.
- Select report data sources (RDSs) that are efficient with data:
    - Use indexed RDSs whenever they're available.
    - Use RDSs that provide the smallest set of data that meets your needs. Example: To report on compensation-related transactions, use employee compensation events instead of all business process transactions.
    - Use RDSs with the prompts you need built in. Example: To report on workers by organization, use the Workers by Organization RDS instead of using the All Workers RDS and then using filters to select organizations.
    - Limit your use of calculated fields.
    - Limit your use of filter conditions.
    - When using more than 1 filter, first use the 1 that reduces the most instances.

Keep in mind that your report's efficiency affects how Worksheets performs when adding your report to a workbook. To ensure that a report is efficient, you can test the report by creating multiple versions of it using different RDSs and options and then comparing performance data using the View Report Performance Logs report.

Considerations for Workday Reports and the Worksheets Data Wizard

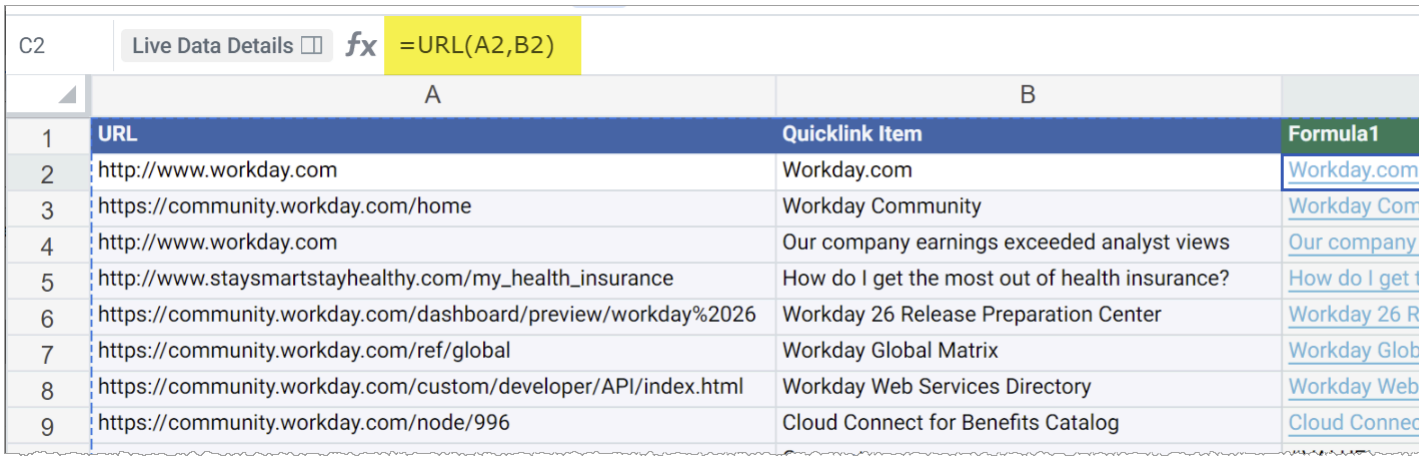
Report Type	Notes
Advanced	Advanced reports that Workday runs for Worksheets have a timeout value of 30 minutes. If the report times out before it can finish calculating, then the Data Wizard can't insert the data into the workbook, and you see an #ERROR indicator along with a message that it wasn't possible to do a SELECT. If you see an error and you think you might have an overly large report, we recommend that you use the Limit number of rows inserted option in the Data Wizard to test whether a report timeout issue exists. This limitation doesn't apply to matrix or composite reports.
Matrix and Composite	After you complete the Data Wizard, Worksheets runs the report in the background before inserting the data into the workbook; this allows you to continue working on other activities while the report runs. For advanced reports, you need to wait while the report runs; a spinner icon lets you know that it's being processed.
Composite	By default the associated workbooks display only 1 level of the outline hierarchy. To include all levels, in the report definition Advanced tab, select the Enable Export Expansion Hierarchy to Excel option.

The Data Wizard doesn't support:

- Analytic indicators.
- Including Quicklinks in live data. Quicklinks from the report definition will display in the workbook but the link isn't functional. To enable clicking in a workbook cell to open Quicklink URLs:

  1. In the Workday report definition, for the Quicklink Assignment business object, include the fields URL and Quicklink Item.
  2. In the workbook live data, add a formula column and use the URL or HYPERLINK function to display a clickable link. (To add a formula column, right-click in the header cell of a live data column and select Insert Formula Column.)

  Example: With the URL field in column A and Quicklink Item field in column B of your workbook live data, adding the formula `=URL(A2,B2)` in cell C2 provides clickable links in that column.



	A	B	
1	URL	Quicklink Item	Formula1
2	http://www.workday.com	Workday.com	Workday.com
3	https://community.workday.com/home	Workday Community	Workday Com
4	http://www.workday.com	Our company earnings exceeded analyst views	Our company
5	http://www.staysmartstayhealthy.com/my_health_insurance	How do I get the most out of health insurance?	How do I get t
6	https://community.workday.com/dashboard/preview/workday%2026	Workday 26 Release Preparation Center	Workday 26 R
7	https://community.workday.com/ref/global	Workday Global Matrix	Workday Glob
8	https://community.workday.com/custom/developer/API/index.html	Workday Web Services Directory	Workday Web
9	https://community.workday.com/node/996	Cloud Connect for Benefits Catalog	Cloud Connec

Concept: Editing Live Data Prompt Values

If you have adequate permissions on the workbook, you can change 1 or more of the Workday report prompt values for an individual live data area and then refresh the data. Additionally, you can select whether or not to use the default prompt value from the report definition when refreshing.

To edit report prompt values before refreshing, click Edit Prompts in the live data information panel, then select new prompt values and click Refresh Now. If you change prompt values but later you change your mind, click Reset All to Defaults to go back to the default prompt values from the report, or click Cancel to keep the original prompt values.

Keep in mind that workbook owners can prevent non-owners from refreshing live data, either in the Data Wizard or by navigating to File > Settings and selecting Only owner can refresh live data. When selected, other users don't see the live data refresh selections in the Data menu or the live data panel.

For any workbooks containing live data that you created before Workday added the Edit Prompts feature: If you didn't enter any prompt values in the report that you added data from – meaning, all prompts were optional and you left them blank – then the Edit Prompts button doesn't display in the live data panel. Run the Data Wizard again to cause the Edit Prompts button to display.

If changes to the report formula cause rows or columns to be added to the inserted data in the workbook when you refresh the live data, Workday overwrites any other data existing in those rows or columns.

Concept: Editing Live Data Columns

If you have adequate permissions on the workbook, you can add, remove, and reorder columns in workbook live data areas. You can also change the display names of columns in the live data area and specify data formats for columns.

Live data column editing applies only to live data from advanced reports. The columns in a matrix/composite report are established by the results of a field in a report definition when the report runs, so it's not possible to edit columns for live data that's from matrix or composite reports.

The most direct way to edit the live data columns is to click the Edit link in the Columns section of the live data details panel. Alternatively, you can click any of the Edit links in the panel, or the Edit Live Data button at the bottom of the panel.

You edit the live data columns using the same dialogs that you see when you run the full data wizard. You can add or remove columns, change column display names, add data formats, add or remove note columns or formula columns.

If changes were made to the underlying Workday report definition after you originally added the live data, you'll see those changes reflected in the data wizard. You can also move to the other pages of the data wizard to do additional adjustments as needed.

When you're finished with your changes, click Save and Refresh to re-run the report and view the updated live data in your workbook.

When you refresh the live data, if changes to the report cause rows or columns to be added to the inserted live data in the workbook, Workday overwrites any other data existing in those rows or columns.

Considerations to keep in mind when editing live data columns:

- If you have static cell references in formulas that refer to live data columns, you need to update them manually; otherwise, the formulas will stop working. To prevent this issue, you can use structured references to refer to columns of live data.
- Pivot tables containing live data rely on the report column names from the Workday report. If you change a report column name in the original report, and that data is used as a field in a pivot table, you need to replace the pivot table field associated with the original name with the field that's based on the new report column name. But don't worry, changing the *display name* of a column in the data wizard doesn't affect the original column name in the report.
- If you use conditional formatting in a live data area, and you edit any columns where you previously applied it, you'll need to manually update those ranges.
- If you use notes columns in your live data, we strongly recommend making a new version of the workbook before making any changes related to notes columns, because:
    - If you remove a notes column, all existing note content in that column will be permanently deleted when you save and refresh the live data.
    - If you remove or change a notes column key, all existing note content in all notes columns will be permanently deleted when you save and refresh the live data.

Concept: Refreshing Live Data in a Workbook

After adding report data to a workbook as live data, in the future you can refresh the workbook based on the report, if you are the workbook owner or the owner hasn't restricted live data refresh permission.

During a live data refresh, the workbook isn't available to you or anyone you shared the workbook with.

When Worksheets refreshes live data, it:

- Uses any numerical formatting that you defined in the original report definition.
- Recalculates any volatile functions, and formulas affected by changed workbook data.
- Preserves the sort order that you set in the Order By option in the data wizard, but doesn't preserve standard sorting within the workbook (using Data > Sort).
- Resets any formatting settings, such as bold text and background colors.

If the original report definition changed after you inserted the live data into a workbook, and you later refresh the data, Worksheets obtains the valid prompts from the report definition and displays only that data.

When you refresh the live data in a workbook, Worksheets updates data even if that data is in a protected range. You can't prevent a refresh from updating existing workbook content. Example: If you update a plan entry area in Workday Planning, or if a scheduled live data refresh runs, Worksheets updates any necessary data even if you protected the plan entry area or live data range.

This table describes what happens when a live data refresh occurs, whether it starts manually or automatically.

Refresh Type	Method	Notes
Manual	Select Refresh Now in the live data panel.	• Refreshes a single live data range. • For any prompts where you selected the Use Report Default option, Worksheets uses the default values from the report definition when refreshing the data; otherwise, the refresh uses the current prompt values displayed in the Live Data panel.
Manual	Select Edit Prompts in the live data panel.	• Refreshes a single live data range. • Enables you to edit report prompt values or select to use the default values from the report definition, if applicable. • Adding or removing selected report columns isn't supported.
Manual	Select Data > Refresh All Live Data.	• Refreshes all live data ranges in the workbook. • For any prompts where you selected the Use Report Default option, Worksheets uses the default values from the report definition when refreshing the data; otherwise, the refresh uses the current prompt values displayed in the Live Data panel.
Manual	Add a formula column or a note column in the live data by right-clicking the header of a live data column.	• Refreshes a single live data range. • Uses the current prompt values. • If you type a note or formula into the 2nd row of a column adjacent to the live data area and press Enter, Worksheets automatically adds a note or formula column into the live data area without re-running the Workday report.
Automatic	Select Data > Schedule Live Data Refresh. You must be the workbook owner.	• Refreshes all live data ranges in the workbook. • For any prompts where you selected the Use Report Default option, Worksheets uses the default values from the report definition when refreshing the data; otherwise, the refresh uses the current prompt values displayed in the Live Data panel.

Concept: Single- and Multi-Instance Field Values in Workbooks

This topic describes workbooks with multi-instance field values that were inserted after December 2019. Refreshing live data in older workbooks doesn't add multiple instance values; only the first value would be returned. To include values for multi-instance fields in your workbook, run the Data Wizard to insert the report data into a new area of the workbook and select the Enable Multi-Instance Values option.

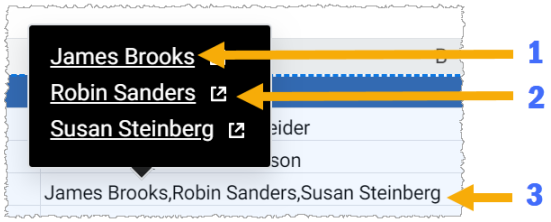
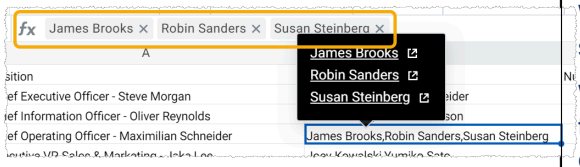
Multi-instance values display only for live data areas in workbooks - not for workbooks that are created after you click the Export to Worksheets button from a Workday report.

Enabling Multi-Instance Field Values in the Data Wizard

In the Options dialog of the Data Wizard, the Enable Multi-Instance Values check box causes all values of all multi-instance fields to be displayed in the resulting workbook. By default, the check box isn't selected; this preserves backward compatibility with existing workbooks.

Visual Cues for Instance Field Values in Workbooks

To interact with the values, hover over a cell containing single-instance or multi-instance field values.

Instance Value Visual Cues	Notes
	When you insert instances as live data, you see these characteristics: 1. Click the instance link to view its details, in the same browser tab. 2. Click the icon to open the instance details in a new browser tab. 3. The cell shows the values in a comma-separated list. If you don't have access to an instance's details, you see an error when you click the details link.
	When you insert instances as values, you see the same characteristics as above. When you select a cell containing instance values, they display in the formula bar as tags with an x icon, enabling you to delete individual values from the cell if needed. You can also delete all values from within the cell using the Delete key. If you want to type into one of these cells, first you must remove all instance values; text and instance values can't coexist in a cell. Entry area instance values have the same appearance and behavior as instances that Worksheets inserted as values.

Workbook Actions and Instance Value Behaviors

Keep these behaviors in mind when your workbook contains instance values:

Action/Object	Notes
References in formulas	When comparing a single instance value to a string that visually appears to be the same, the formula evaluates them as equal.
Using the formula editor	The formula editor isn't available when working with multi-instance field values in workbooks.
Filtering	Each multi-instance value set displays as a selectable filter.
Sorting	Worksheets sequences the sort based on the concatenated set of values in the cell, as if it were a string.
Pivot tables	Worksheets treats each unique set of instance values as a separate value.
Charts	Worksheets treats each unique set of instance values as a separate value.
Links to documents such as PDFs from an instance	If you click the link of an instance that has a document preview page, but you don't have permission to view the preview, you see the error message *The task submitted is not authorized*. If you click the link of an instance that doesn't have a preview page, you see the message *No task runner for message*.
Download to Excel as XLS, XLSX	Worksheets converts multi-instance values into a string in the form of JSON, maintaining the multi-instance values and types if you reupload the workbook.
Download to CSV	Instance values are in a comma-separated list.
Download to Excel as values	Worksheets converts instance values to strings with a line break between each value.
Download as PDF/Print	Worksheets converts instance values to strings with a line break between each value.

Example Comparisons of Single Instance, Multi-Instance, and String Values

This table shows when comparisons evaluate as the same or different, based on the type of data you're comparing.

Comparison	Result	Example
Multi-instance values in a different sequence	Not equal	MULTIINST(INSTANCE("ID1","Denver"), INSTANCE("ID2", "Chicago")) Compared to: MULTIINST(INSTANCE("ID2","Chicago"), INSTANCE("ID1", "Denver"))
Multi-instance with 2 IDs and values, compared to string with values in a different sequence.	Not equal	=MULTIINST(INSTANCE("ID1","Denver"), INSTANCE("ID2", "Chicago")) Compared to: The string "Chicago,Denver"
Multi-instance with 2 IDs and values, compared to a single instance with	Not equal	=MULTIINST(INSTANCE("ID1","Denver"), INSTANCE("ID2", "Chicago")) Compared to:

Comparison	Result	Example
the value in a different sequence.		The single instance ("ID1", "Chicago,Denver")
Multi-instance values with different IDs but the same descriptors.	Equal	=MULTIINST(INSTANCE("ID1","Chicago")) Compared to =MULTIINST(INSTANCE("ID2","Chicago"))
Multi-instance with a single value, compared to an equivalent string.	Equal	=MULTIINST(INSTANCE("ID1","Chicago")) Compared to: The string "Chicago"
Multi-instance with a single value, compared to a single instance value; IDs can differ.	Equal	=MULTIINST(INSTANCE("ID1","Chicago")) Compared to: The single instance ("ID4", "Chicago")
Multi-instance with 2 IDs and values, compared to an equivalent string.	Equal	=MULTIINST(INSTANCE("ID1","Denver"), INSTANCE("ID2", "Chicago")) Compared to: The string "Denver,Chicago"
Multi-instance with 2 IDs and values, compared to a single instance that has the same descriptor.	Equal	=MULTIINST(INSTANCE("ID1","Denver"), INSTANCE("ID2", "Chicago")) Compared to: The single instance ("ID1", "Denver,Chicago")

Worksheets-Unique Formulas That Manipulate Instance Field Values

This table summarizes functions that enable you to work with single-instance and multi-instance values. For details, see the Function Reference in the User Guide (Help > Function Reference).

Function	Description
INSTANCE	Returns a single-instance value.
INSTANCE.DESCRIPTOR	Returns the descriptor (string value) of the instance.
INSTANCE.ID	Returns the ID of the instance.
MULTIINST	Creates a multi-instance value from a comma-separated list of single-instance values.
MI.COUNT	Returns the number of instances inside a multi-instance value.
MI.INDEX	Returns the instance at a specific index in a multi-instance value.

Add Live Data from Workday Reports to Workbooks

Prerequisites

- Security: *Worksheets* security domain in the System functional area.

The Workday report must:

- Be an advanced, matrix, or composite report. Worksheets doesn't support inserting other report types, such as Simple, Standard, or Xpresso.
- Be enabled for Worksheets. Select Enable for Worksheets in the Advanced tab of the custom report. Enabling a report causes it to display in the list of available reports, when you start the Data Wizard.
- Be enabled as a web service (required only for advanced reports). Select Enable as Web Service in the Advanced tab of the custom report.
- Include fields from the primary business object and any necessary related business objects. If you use related business objects, your report must contain group column headings.
- Use these field types: Boolean, Date, Numeric (integers only), Text, Multi-Instance, or Single Instance. Worksheets doesn't support other types (Currency, Date Time, DateTimeZone, Time).
- Not contain duplicate column labels (Column Heading Override) or XML aliases.
- Not contain fields that use the Do Not Prompt at Runtime prompt option.

Context

The Data Wizard enables you to define the data that you want to add to a workbook, based on an existing Workday report, and to select whether to keep the workbook in sync with the original report data.

You can add data from one or more Workday reports into a workbook. We recommend using a new sheet for each report that you're adding data from. When you select a report to use for the live data, and the report contains prompts, you select the values that you want to use for those prompts when the report runs to refresh the live data.

In a workbook with multiple live data areas that are based on an advanced report, you don't need to manage each set of live data prompts separately. You can select a set of prompts to be global prompts, and Worksheets then applies those prompt values to all live data areas in all sheets (reports) in the workbook when the live data refreshes.

Worksheets supports these data field types in reports: Boolean, Date, Numeric (integers only), Text, Multi-Instance, and Single Instance.

Start the Data Wizard

1. Select the starting cell for the data. Worksheets starts adding data in the selected cell and adds data to the right and down from that cell.
2. Click the Add Live Data button, which is next to the Share button. Then follow along as the Data Wizard guides you through the workflow of creating a live data area from a report.

Select Report

1. Select a report to use as the data source for the live data. By default, only reports that are enabled for Worksheets display in the list of reports.
2. (Optional) Select the Include reports not enabled for Worksheets option to view all reports that you have access to.

   Note: If you select a report that's not enabled for Worksheets, the process of adding the live data to the workbook might fail. The Include reports not enabled for Worksheets option adheres to existing report sharing security. Worksheets displays only reports that you can run elsewhere in the system.

Select Prompt Values

Select values for the report prompts, if applicable:

- (Optional) Select Use Report Default to use the default prompt value from the report when initially getting the report data, and when refreshing the live data.
- (Optional) Select Determine Value Dynamically for applicable date prompts, then select a dynamic value, to determine the prompt value based on a context, such as the current date.

- (Optional) If you previously assigned a report as a source report, you can select the Map to Existing Prompt selection, then select the defined global prompt. Available prompts show the table name and the prompt name.
- (Optional) If you change or add prompt values but later you change your mind, select Reset All to Defaults to go back to the default prompt values.

Select Columns

1. Select the columns to include in the workbook using one of these methods:
    - Drag them into the preview area.
    - Click the + next to the column name.
    - Search for a column by name.
    - Click Select All.
2. (Optional) Remove report columns by clicking the X icon, or reorder columns by dragging them.
3. (Optional) Rename report columns to use names that are more meaningful for your live data area, by clicking the Column Options (three dot) menu at the right side of the column. Then type a name into the Display Name field. The original name from the Workday report displays in a smaller font below the new name. The name must be unique to the live data area (report).
4. (Optional) Set a format for the data by clicking the Column Options (three dot) menu at the right side of the column. In the Formatting drop-down menu, the available formats match the selections in the Format > Number menu. Worksheets retains the formatting when the related live data refreshes.
5. (Optional, for Advanced reports only) Click Add Note Column to add a column into the live data for notes that you want to manually enter later. Use a note column when you want to keep your notes aligned with the correct rows of data even after a live data refresh. After adding a Note column:
    - You can drag the note column to a different location in the sheet.
    - You must select a report column in the Key field. The column you select must contain unique values, such as Employee IDs. The value must not change over time, and the value must exist in *only one row* of the live data. Worksheets uses the key to match note rows to live data rows. Make sure your key is unique; if it isn't, notes will be lost.
    - If you add at least one note column now in the Data Wizard, you can add more note columns in the live data after completing the wizard.
6. (Optional, for Advanced reports only) Click Add Formula Column to add a column into the live data for a scalar formula that you want to manually enter later. This formula will calculate only for the rows of data in the live data area. You can:
    - Drag the formula column to a different location in the sheet.
    - Add more than one formula column.
7. (Optional) To sort the data according to a particular column in the report, select an Order By column and a sequence (A-Z or Z-A). If you use Order By to sort the data, the sort overrides the sort settings from the Workday report definition. When you use the data wizard's sorting option, Worksheets preserves the selected sort order when you refresh the data. Standard sorting within the workbook (using Data > Sort) isn't preserved when you refresh.

Select Options (Optional)

1. Insert report data as either live data or static values. If you added a note column or formula column in a previous step, the static values option isn't available.
2. Type a Table Name for the live data table to be created. You can use this table name in combination with column names to create references to the live data; these are called structured references. The table name can be between 1 and 255 characters long. Table names must be unique across the workbook, and the name must not be the same as any pivot table names. If you don't enter a table

name, Worksheets automatically creates a value based on the underlying Workday report name. Additionally:

- If the report name contains non-valid characters (characters other than letter, digit, underscore, or period), those characters are replaced with underscores in the generated table name.
- If the generated table name would conflict with a workbook address (such as a cell address of A1 or a sheet name/address), Worksheets appends an underscore to the generated name.
- For the first instance of a live data table, Worksheets uses the report name as the table name, without adding a number at the end. If you add two or more live data areas into the workbook based on the same report, Worksheets adds a unique number at the end of each automatically generated table name.
- If the report name is longer than 255 characters, the generated table name is truncated and a unique number is added.

3. Select Limit number of rows inserted if you want to preview the results before generating the live data area with all the report data.
4. Clear the Highlight live data area check box if you don't want Worksheets to automatically highlight the live data area content.
5. Restrict live data editing and refresh to the owner only.
6. Enable multi-instance values to cause all values of multi-instance fields to be displayed in the resulting workbook.
7. (Optional) To assign the prompts from this report as global prompts, select the Set as Source Report option. You can select only one source report for a workbook. If you select this report as the source report, then you need to map other live data area prompt values to the global prompts. For the other live data areas, in the Data Wizard Select Prompt Values page, select the Map to Existing Prompt selection, then select the desired global prompt.
8. Click Save & Refresh to run the associated Workday report and display the live data in the workbook.

Schedule Automatic Live Data Refreshes

Context

In a workbook containing live data from one or more Workday reports, the workbook owner can create a schedule to refresh all the live data in the workbook and recalculate the workbook. For example, you can schedule refreshes to occur during the night so you have access to the most recent data in the morning. Workday automatically runs any associated Workday reports and refreshes the workbook based on changes that occurred in the report data.

When the schedule runs:

- Worksheets refreshes live data; it also recalculates any volatile functions, and formulas affected by changed workbook data.
- The refreshed live data is available to any users you shared the workbook with.
- Worksheets assigns you as the user who runs the refresh. This information displays in the live data panel.
- If the Workday report associated with the live data refresh times out 3 times consecutively, Worksheets automatically cancels the refresh and pauses the schedule. If Worksheets pauses a schedule, the workbook owner receives a Workday notification that includes a link to the affected workbook. After you fix the problem that caused the report failure, you need to manually resume the schedule by clicking Resume in the live data panel in the workbook.

Keep these considerations in mind:

- Prompt values for the report fields vary depending on your selection in the Data Wizard:
  - Use Report Default: Uses the default value from the report definition.
  - Map to Global Prompt: Uses the global prompt value you selected from the source report.
  - Determine Value Dynamically (for applicable date prompts): Uses the selected dynamic value, such as `Today` or `Last Day of Last Month`.
- You can create 1 schedule per workbook. If a schedule already exists, you can edit it or delete it.
- If you copy a workbook, the schedule no longer runs; you must create a new schedule in the copied workbook.
- The Report Writer Schedule a Report task doesn't support sending the report output to a workbook.
- If the workbook is in manual calculation mode, the schedule doesn't update any formulas.
- Worksheets doesn't preserve the schedule for:
  - Copied workbooks.
  - Removed workbooks (that are in the trash). If you restore the workbook, Worksheets restores the schedule.
  - Workbooks that are merged into another workbook.
  - Workbooks that change ownership.
  - Workbooks from tenants that you refresh from tenants with a different name. You need to create new schedules for the workbooks. Example: You refresh a tenant named *company_name_preview* from a tenant named *company_name*.

Steps

1. In the workbook, select Data > Schedule Live Data Refresh.

   If a schedule already exists for a workbook, the menu option is Data > Edit Scheduled Refresh. Select this option to edit or delete an existing schedule.
2. Select either a one-time refresh, or define the refresh frequency by selecting options.
   - The fields in the dialog differ depending on the refresh frequency you choose.
   - The schedule must start on a future day; you can't start a schedule during the current day.
   - You can set an expiration date for a schedule up to 1 year in the future.
   - You can run a scheduled refresh 1 time per day.
3. (Optional) Prevent Workday notifications by clearing the Refresh Notifications check box.

Example: Use Worksheets with the What's New in Workday Report

This example illustrates how to use Worksheets with the *What's New in Workday* report. To do so, you want to create a workbook that includes:

- Live data.
- Note columns for each analyst and their assessments.
- Data validations.
- Formula columns that extract a list of features that affect your training materials.
- Adoption items.

Context

You lead a team of 3 to manage the implementation of a Workday feature release. With each release, you and your team assess the What's New in Workday report to determine which features you:

- Must implement because they're automatically available.
- Can ignore.
- Want to consider for future adoption and the setup effort needed.
- Need to research to determine what changes to make to your employee training manual.

- Want to create adoption items for.

The steps use GMS tenant names Logan McNeil (you), Betty Liu, Steve Morgan, and Teresa Serrano. Substitute your team names for these names in your tenant.

Prerequisites

Configure Worksheets and Drive, and give access to the feature planning analysts.

Security for you (not needed for the other users):

- *Custom/Standard Report Copy* domain in the Tenant Non-Configurable functional area.
- *Set Up: Tenant Setup - General* domain in the System functional area.

Steps

1. Copy a standard report to create a custom report.
    a) Access the Copy Standard Report to Custom Report task.
    b) Select *What's New in Workday* from the Standard Report Name prompt.
    c) Click OK.
    d) Enter *Release Planning with WN Report and Worksheets* on the Name field.
    e) Click OK. (An alert displays on the page; you can ignore it.)
    f) On the Columns tab, click + to add a new column.
    g) Select Workday ID from the Field prompt.
    h) On the Advanced tab, select these check boxes:
        - Enable As Web Service
        - Enable for Worksheets
    i) Click OK.
2. Create a new workbook in Drive:
    a) Select Drive in the Workday main menu.
    b) Select New > Workbook.
    c) Use Workday [Release #] Release Planning as the name.
    d) Click Create.
3. In the workbook, add live data using the report you created:
    a) Click Add Live Data.
    b) In the Select Report dialog, select the Release Planning with WN Report and Worksheets report.
    c) Click Next.
    d) In the Select Prompt Values dialog, select:

Option	Description
Enabled Functional Areas	Select the check box
Workday Releases	[Next release]

    e) Click Next.
    f) In the Select Columns dialog:
        - Click Select All to add all of the report columns to the workbook.
        - Click Add Note Column 2 times to add 2 note columns.
        - In the Notes Key field, select *Workday ID*.
    g) Click Next.
    h) In the Select Options dialog, select Enable Multi-Instance Values so that all enabled functional areas display in the workbook.
    i) Click Add. The Workday report runs and inserts the data into the workbook.

4. Rename the Sheet1 tab and note column headings:
    a) On the Sheet1 tab, click the arrow on the right side of the tab and then click Rename.
    b) Rename the sheet as *Feature Data*.
    c) Click OK.
    d) Double-click in each of the note column headings to rename them:

From...	To...
Note1	Analyst
Note2	Assessment

5. Set up data validation values so you can create drop-down lists for assigning analysts to items, and for selecting an assessment result:
    a) Click + at the bottom left of the workbook to create a new sheet. Worksheets names the sheet Sheet1.
    b) On the Sheet1 tab, click the arrow on the right side of the tab and then click Rename.
    c) Rename the sheet as *Data Validation Values*.
    d) Click OK.
    e) Enter these values:

	A	B
1	*Betty*	*Implement*
2	*Steve*	*Investigate*
3	*Teresa*	*No action*

    f) In the Feature Data sheet, select the first cell in the Analyst column.
    g) Click in the cell below the heading and go to Data > Validation.
    h) Enter these values:

Field	Value
Cell Range to Validate	Enter a range that includes the entire column. Example: *N:N*.
List of Values from Formula	Enter `='Data Validation Values'!A:A`.

    i) Click OK.
    j) In the Assessment column, click in the cell below the heading and go to Data > Validation.
    k) In Cell Range to Validate, edit the range to include the entire column. Example: Change the value from O2:O2 to `O:O`.
    l) In List of Values from Formula, enter `='Data Validation Values'!B:B`.
    m) Click OK.
6. In the Analyst column, click in each cell and select an analyst from the drop-down list. (We recommend placing values in at least 20 of the cells, so that later steps in the example will display data.)

7. Share the workbook with your analysts, giving Can Edit access:
    a) Click the Share icon at the top right of the workbook.
    b) In the Share with Individuals section, enter:
        - *Betty Liu*
        - *Steve Morgan*
        - *Teresa Serrano*
    c) Type this comment in the message area: *Analysts, please select an assessment value in each of the workbook rows assigned to you.*
    d) Click Share. The analysts receive a notification in their Inbox that contains a link to the workbook.
8. The analysts can open the shared workbook, study the What's New Items assigned to them, and select a value in the Assessment column for each item. We recommend placing values in at least 20 of the cells, so that later steps in the example will display data.
9. In a new sheet, extract information about What's New items that you're implementing, and that affect your training materials. Your education team can use this information to update their employee training materials:
    a) Click + at the bottom left of the workbook to create a new sheet.
    b) Name the sheet `TrainingUpdates`.
    c) Paste this formula into cell A1 of the sheet, replacing the final argument A1:O200 with the range of data in the Feature Data sheet: `=SELECT("SELECT `Feature`, `What's New Item`,`Functional Area(s)`, `Feature Description` from ? WHERE `Training & Testing Impact` = 'This feature may require additional testing and may impact your training materials.' AND `Assessment` = 'Implement'",FeatureData!A1:O200)`
    d) Press Ctrl+Alt+Enter (Windows) or Command+Option+Enter (Mac) to submit the formula.
10. Add adoption items to your adoption dashboard:
    a) Go to the *Feature Data* sheet.
    b) In the What's New Item column, hover the cursor over an item until you see a link.
    c) Move the cursor to the far right of the link and click the outward arrow icon; this opens the What's New Item in a new browser tab.
    d) From the related actions menu of the View What's New Item, select What's New Item > Create Adoption Item. The Create Adoption Item task page displays, with fields from the workbook filled in automatically.
    e) Fill in any additional fields for the adoption item that you want.
    f) Click OK.
    g) Click Done.
    h) Repeat this step for other What's New items.
11. Refresh the live data so that the newly added adoption items display in the workbook:
    a) In the live data panel, click Refresh Now. (If you previously closed the panel to see more of the workbook data, you can display it again from View > Panels > Live Data.)
    b) Click Confirm. The Workday report runs, and the workbook displays any new data; the adoption items you added display in the Adoption Item(s) column.

12. Create a pivot table to see a summary of the analyst data:
    a) In the *Feature Data* sheet, select Insert > Pivot Table.
    b) In the Create Pivot Table dialog, leave the fields as is and click Create.
    A basic pivot table displays in a new Sheet1 and the Pivot Table panel displays.
    c) In the Pivot Table panel:
        - From Available Fields, find Analyst and drag it to the Columns area.
        - From Available Fields, find Assessment and drag it to the Rows area.
        - Click Update. The table updates to display a summary that looks similar to this image:

	A	B	C	D	E
1	*COUNTA of Workday ID*	*Analyst*			
2	*Assessment*	**Betty**	**Steve**	**Teresa**	**(blank)**
3	Implement	2	42	15	0
4	Investigate	1	45	5	0
5	No Action	23	0	25	0
6	(blank)	0	4	142	593
7					
8					

Next Steps

Optionally, you can create a formula column in the workbook to create working Community Post links in the live data area so that the links open the associated What's New posts instead of an instance page. (Worksheets doesn't support opening external links or attachments from a workbook.) To do this:

1. In the workbook, create a formula column to the right of the Community Post column. Right-click the Community Post header and select Insert Formula Column > Insert Right.
2. Click Confirm to do the live data refresh and create the formula column.
3. Double-click the Formula1 header and rename the header to `Community Post (Corrected)`.
4. In the first cell of the formula column, type the formula `=IFERROR(HYPERLINK(F1),"None")`, replacing the F column identifier with the Community Post column letter in your live data.
5. Press Enter to submit the formula. The IFERROR function causes cells without a Community Post link to display the text None and the HYPERLINK function takes the content of the non-working cell and creates a working link.

Refresh the live data at any time to bring in any corrected or newly added What's New items.

Create charts to track progress of analysis.

Share with others outside the team, giving them comment or view access, so they can see the assessments but can't edit the data.

Workbook Templates

Concept: Workbook Templates

Workbook templates enable you to deliver robust data modeling and analysis that maintains the Workday security model. You can define a standardized layout and data model in a workbook, convert the workbook to a template, and distribute that template at scale. Although the most popular reason to use templates

is to preserve live data security, templates can also be extremely valuable for workbooks that contain complex formulas.

When you distribute a template to a recipient, and they run the template to create a workbook, the resulting workbook displays Workday data based on the recipient's security permissions.

Workbook templates:

- Retain workbook functionality.
- Display Workday report data based on the template recipient's report access and data access when they run the template.
- Display in Drive like any other item type. A template icon looks like a workbook icon, but it has a T at the bottom right.

The general template workflow steps are to:

1. (Author/Owner) Convert the workbook to a template.
2. (Author/Owner) Distribute the template to individuals or groups, or turn on link distribution and give others a link.
3. (Recipient) Run the template to create a workbook.

After converting a workbook to a template, you can edit the template layout, data model, and so on if needed. Before editing, you must prevent users from running it while it's being edited; to do this, you make it unavailable to run by switching a toggle. When you're done editing, you make it available again so that your template recipients can re-run the template to see your updates.

Considerations for Using Workbook Templates

- When you convert a workbook to a template, Worksheets doesn't preserve the original workbook.
- When you convert a workbook to a template, Worksheets clears any existing scheduled live data refreshes. (For any templates that you created before functionality changed in February 2024, existing scheduled refreshes will no longer run.)
- You can't schedule a live data refresh for a template.
- If a live data refresh is currently running in a template, you can't select the Available to Run option.
- When the template recipient runs a template containing live data to create a workbook, the Workday report runs with the default prompt values from the report definition wherever applicable.
- Worksheets doesn't preserve live data note column information in converted templates, even if the template recipient has access to the particular data item; this allows the recipient to enter notes that are most relevant to their specific data instead of the original workbook owner's data.
- If the template author protected areas in the workbook before converting it to a template, ownership of those areas transfers to template recipients.
- After you distribute a template, any previous workbook schedules are not preserved in resulting workbooks; you need to create a new schedule.
- Workday administrators can permanently delete workbooks that are used as the source for templates, and can also permanently delete templates, regardless of whether they have been distributed.
- Templates aren't supported for active entry areas; the Convert to Template action isn't selectable.
- We recommend not linking from workbook templates to slides presentations. A template isn't intended for use as the direct source of data. Link to workbooks generated from templates instead.

Create and Distribute Workbook Templates

Prerequisites

For group distribution, the administrator must enable the security group for sharing in Drive.

Context

Any Worksheets user can create and distribute templates.

Steps

1. Convert the workbook to a template:
    a) In the workbook, select File > Convert to Template.
    b) Select the Got it! check box, then click Convert.
2. (Optional) Distribute the template:

To individuals	Type user names in the Distribute to Individuals area of the Distribute dialog.
To groups	Type security group names in the Distribute to Groups area of the Distribute dialog. Worksheets supports template distribution only for unconstrained security groups.
To unspecified people with a link	Click the Link Distribution Off toggle to turn link-based distribution on. Copy the link and share it with your recipients.

    Distribution is similar to sharing, except that recipients run the template to create their own workbook, and the permission levels edit, comment, and view aren't applicable.
3. (Optional) Add a description for the template by clicking Template Settings at the top right of the template.

Next Steps

You can distribute a template immediately after converting it as described above, or:

- From Drive by selecting the template and then clicking Distribute Template.
- From the template by navigating to File > Manage Distribution or by clicking Manage Distribution to the right of the Available to Run toggle.

After you distribute a template, the recipient runs the template to create a workbook. From Drive, the recipient selects to open the template, and a dialog guides them to click Run Template to create a workbook. If the template uses live data, the associated Workday report runs, formulas calculate, and the workbook is created based on the recipient's level of access to Workday data. If the recipient has limited or no access to the report, an error occurs for those reports areas and subsequent formulas.

You can edit a template just as you would a workbook, but you need to prevent users from running the template while you're editing. To do this, click the toggle at the top of the template so it displays Not Available to Run. In this state, you can edit the template, and users that you distributed the template to can't run it. Any changes that you make to the template don't automatically propagate to any shared templates. When you're done editing, click the toggle to display Available to Run. Users that you previously distributed the template to must run the template again to see your updates.

Just as you can share workbooks, you can share templates, giving edit, comment, or view permission. But only the author of the template can control the Available to Run setting that allows editing to occur; editors rely on the author to make it unavailable to run, so that they can edit it.

Reference: Template Actions Available Based on Permissions

These tables summarize the primary actions available for templates based on the state of the template and the user's permission level.

Actions in bold text indicate that the available actions are different based on whether the template is available to run or not.

The Can Run permission level indicates that the template was distributed to the person, but not shared with them.

Distributed and Available to Run

Action	Can Run	Can View	Can Comment	Can Edit	Owner
Run template	X	X	X	X	X
View template		X	X	X	X
Comment on template			X	X	X
Edit template					
Rename template				X	X
Copy template	X	Note 1	Note 1	X	X
Export template		Note 1	Note 1	X	X
Share template				Note 2	X

Note 1: Allowed only if the workbook owner selected the Commenters and viewers can copy, download, and print option when sharing.

Note 2: Allowed only if the workbook owner selected the Editors can share option for the workbook when sharing it.

Not Available to Run (can be edited)

Action	Can Run	Can View	Can Comment	Can Edit	Owner
Run template					
View template	X	X	X	X	X
Comment on template			X	X	X
Edit template				X	X
Rename template				X	X
Copy template	X	Note 1	Note 1	X	X
Export template		Note 1	Note 1	X	X
Share template				Note 2	X

Note 1: Allowed only if the workbook owner selected the Commenters and viewers can copy, download, and print option when sharing.

Note 2: Allowed only if the workbook owner selected the Editors can share option for the workbook when sharing it.

Data Analysis in Workbooks

Concept: Data Analysis with Worksheets Functions

Data analysis is an iterative process of collecting, cleaning, and shaping your data, followed by aggregation and analysis. Worksheets helps to make your workflow more efficient by enabling you to work with live Workday data, and to use Worksheets-unique functions that improve the analysis workflow.

As you do each of these steps, remember to keep your original data as a backup by creating new workbooks for the manipulated data.

Collect

Compile your data, gathering it from resources both outside and within Workday. Make sure that your raw data is complete. Here's a summary of actions that you might do during data collection:

- Select +New > Upload from Drive to create workbooks from data that exists outside Workday.
- Select Data > Add Live Data to add live data from Workday reports into a workbook. The Data Wizard helps you identify the subset of data that you want to insert to the workbook.
- Keep your data current by creating a schedule to refresh the live data.
- Use the ARRAYAREA function to copy data from one unconstrained array to another, creating an organized set of raw data to manipulate in the Clean step. ARRAYAREA returns the containing range of the array formula, based on the cell address you specify. The array can originate either in a Workday report or an array formula.

Clean

Reduce your data to only the information you need by removing duplication, trimming empty workbook values, and more.

Keep these Worksheets-unique functions in mind:

Function	Notes
DISTINCTROWS	Combines a set of ranges into a single range while removing any rows that are duplicates. DISTINCTROWS evaluates text and instance values as not distinct from each other. When the supplied range contains both an instance value and a text string that are the same, the function returns 1 row, the instance value.
REMOVECOLUMNS	Removes one or more columns from the referenced area. The function removes the number_of_columns, starting at and including start_column.
REMOVEROWS	Removes one or more rows from the referenced area. The function removes the number_of_rows following *start_row*, starting at and including *start_row*. Use REMOVEROWS without specifying any rows in order to remove the first (heading) row.
TRIMCOLUMNS TRIMROWS	Removes trailing blank columns and rows from a range, when the data is a result of an unconstrained array formula.
TRUNCATEMATRIX	Removes rows, columns, or both, from a matrix.
UNIQUE	Returns a matrix whose rows are unique according to the specified keys. The function returns only unique rows, based on the values in the specified columns. This function is similar to DISTINCTROWS(), but UNIQUE() takes a single range and a set of columns.

Shape

Arrange, convert, and organize your data to create consistency:

- Standardize your columns and create new ones if needed.
- Give each column a unique and descriptive header.
- Format each column consistently.
- Double-check for duplicate or missing rows.
- Make sure the wording and formatting for text data is consistent.
- Make sure that no cells are blank.
- Convert data values to the same unit where needed.

When shaping data, there are 2 main types of data manipulation:

- Value-specific: The action you want to take depends on the value in the cell.
- Data arrangement: The manipulation is value-agnostic and you're moving cells, columns, and rows to different locations.

For value-specific manipulation, these Worksheets-unique functions can be helpful:

Function	Notes
CONVERT	Converts a number from 1 unit of measurement to another.
DATESBETWEEN	Returns an array of dates that starts and ends on a particular date, with a step interval between each date.
DATESFROM	Returns an array of dates that starts on a particular date and continues for the number of dates you specify, with a step interval between each date.
IN	Determines whether a value or list of values that you specify is in another list of values. If so, returns True; otherwise, returns False.
MATCHCOMPOSITE	A common use case for MATCHCOMPOSITE is to consolidate column data from 2 sheets into 1, where you have data on a sheet, along with notes about that same data in a column on a different sheet. MATCHCOMPOSITE copies values in one or more columns from the location to the right of a source array, and returns values to the right of a destination array. You use a composite key of columns to match copied data to the correct rows in the destination.
MATCHEXACT	Looks up an exact match for the value in the sorted list (one-dimensional array) you specify, and returns the position of the value. You can use this function to match logical values, numeric values, or text strings. This function is similar to MATCH, but MATCHEXACT: • Always searches for an exact match; it returns an #N/A error if it doesn't find the value. • Doesn't allow wildcard characters such as * or ?. • Uses a binary search for better performance.
MHLOOKUP	We recommend this function as a replacement for HLOOKUP. MHLOOKUP performs a horizontal (row) lookup on a table and returns all matches. MHLOOKUP is similar to HLOOKUP, but: • HLOOKUP scans only the top row for matches; in MHLOOKUP you specify the row to search. • HLOOKUP stops after finding 1 match, and returns a single cell value; MHLOOKUP scans the entire lookup row for matches. Each match in that row returns a new column in the output. If you specify 4 return_row_index values, then each resulting column will have 4 rows.

Function	Notes
MVLOOKUP	We recommend this function as a replacement for VLOOKUP, especially when you're working with live data. MVLOOKUP performs a vertical (column) lookup on a table and returns all matches. MVLOOKUP is similar to VLOOKUP, but: • VLOOKUP scans only the left column for matches; in MVLOOKUP you specify the column to search. • VLOOKUP stops after finding 1 match, and returns a single cell value; MVLOOKUP scans the entire lookup column for matches. Each match in that column returns a new row in the output. If you specify 4 return_column_index values, then each resulting row will have 4 columns.
REGEXFIND	Returns the position of the first character of a substring that matches the regular expression pattern. The position value is zero-based.
REGEXPARSE	Extracts parts of a string by matching to a pattern.
SETUNITS	Converts a number from its current unit of measurement to another. This function is similar to CONVERT(), but in CONVERT() you must specify both the original and the new unit values.
SELECT	Valuable for value-based manipulation as well as data arrangement. The SELECT function is similar to an SQL SELECT statement. The basic format is: SELECT column1, [column2], ... FROM table where column is the data to return and table is the data source to select from. In the FROM clause you can specify, for example: • Defined name. • Column, row, or area range. • Parameter that you specify as an argument.

For data arrangement, these Worksheets-unique functions can be helpful:

Function	Notes
WD.ARRANGECOLUMNS WD.ARRANGEROWS	Creates a new range from an existing range, with the columns or rows ordered according to the specified indexes. With WD.ARRANGECOLUMNS you can add an empty column by including a null index value. Example: =WD.ARRANGECOLUMNS([range],1,2,3,,4) inserts a blank column between the 3rd index value and the 4th index value.
CORRELATE	Creates a new matrix by combining rows from the ranges you specify. This function is similar to a database join.
FLATTEN	Returns an expanded range of data based on the hierarchical data that you specify. Typically you use this function to expand an organization's manager and employee information so that it displays all levels of the hierarchy.
JOIN	Performs an inner left join on 2 ranges.
MERGECOLUMNS	Merges columns by placing them side by side into a new range.
MERGEROWS	Merges rows by placing them 1 below the other into a new range.
MINUS	Returns all rows from a first range that don't appear in any of the other supplied ranges.
SORT SORT2 SORT3	Sorts an existing matrix and returns a new matrix. SORT accepts 1 sort direction and sorts all columns you specified based on that direction. SORT2 accepts pairs of parameters, which you use to specify the referenced column and the sort direction for

Function	Notes
	that column. SORT3 assumes that the first row of the array to be sorted is a header and returns that row at the top of the results.
VALUEAT	Returns the value at the intersection of a column header and row label.
SELECT	Valuable for value-based manipulation as well as data arrangement. The SELECT function is similar to an SQL SELECT statement. The basic format is: SELECT column1, [column2], ... FROM table where column is the data to return and table is the data source to select from. In the FROM clause you can specify, for example: • Defined name. • Column, row, or area range. • Parameter that you specify as an argument.

Analyze

Now you're ready to aggregate and analyze the data you've prepared.

The most commonly used analysis tool is the pivot table, which enables you to summarize and analyze large amounts of data. The Pivot Table Wizard and details panel enable you to create and edit pivot tables interactively. You can also create charts to visually demonstrate data relationships.

Keep these Worksheets-unique data analysis functions in mind:

Function	Notes
CAPPEDVALUES	Typically used for 401(k) deductions, ESPP deductions, or tax payments that have a regular value per period, but drop to zero (0) when the payment reaches the cap. Returns an array of values over a set of periods from an array of values that you provide, over the same periods, limited by a provided cap over the whole duration.
FORECAST.WD.SEASONAL	Returns a predicted sequence of values using patterns in the historical linear and nonlinear data that you specify.
GROUPBY	GROUPBY is a powerful function that can often replace COUNTIF(S), AVERAGEIF(S), and SUMIF(S). GROUPBY aggregates data, and orders the results based on the order that you specify. The grouping is based on a key that you can predefine in the table, or you can define it using columns in the workbook. The result looks similar to a sorted pivot table. This function is often useful as part of headcount planning.
SELECT	The SELECT function is similar to an SQL SELECT statement. The basic format is: SELECT column1, [column2], ... FROM table where column is the data to return and table is the data source to select from. In the FROM clause you can specify, for example: • Defined name. • Column, row, or area range. • Parameter that you specify as an argument.

Other Notable Worksheets-Unique Functions

These functions aren't specific to data analysis but are very useful in all steps:

Function	Notes
NOTIFYIF NOTIFYIFS	Sends notifications if a condition is met. You can send a notification to a user whether or not they have access to the workbook.

Function	Notes
ONCE	Calculates a formula exactly 1 time. Worksheets never re-evaluates the formula even if you request re-calculation using Data > Recalculate; however, you can manually resubmit the formula. Example: Use this function when the volatile function NOW() places a timestamp in a workbook, and this timestamp must never change.

Concept: Public, Private, and Shared Filters in Workbooks

You can use filters to modify the visibility of rows in your data so that it is easier to analyze. When you share a workbook with other users, you allow them access to all of the data in that workbook, and filtering the data doesn't prevent them from seeing it. Filters should never be used as a security mechanism to limit or hide content from other workbook users.

Worksheets supports three types of filters:

Filter Type	Description	Workbook Permissions	Banner and Border Color
Public	Ensures all users in a workbook see the same filtered data.	Owner, Editor	Orange
Private	Customizes how you see the data without affecting other users' filter settings. When you apply a private filter, the filter uses the entire dataset. Example: If you apply a private filter to a range of data that already has a public filter, the private filter is applied to the full set of data and not just the filtered rows from the public filter. You can save private filters you create in a workbook so that you can easily reapply them.	Owner, Editor, Commenter, Viewer	Teal
Shared	Enables collaboration on filtered data with anyone who has access to the workbook. When you share a filter with others, they can modify the filter values of the shared filter or copy it and make their own version as a private filter. If you click Unshare to stop sharing a filter, it reverts to a private filter and users no longer have access to it. Filters shared by users who no longer have access to a workbook, can continue to be used by others who do have access to the workbook. Users can delete the filters they have shared, but only workbook owners can delete filters shared by other users.	Owner, Editor, Commenter, Viewer	Purple

Creating, Saving, and Sharing Filters

Select the workbook data you want to filter. Then, from the Filter menu, choose the type of filter and click Start Filter. Use the drop-down arrows on a column of data to select values and apply the filter. A filter icon displays at the top of each column with an active filter. To remove the filter and display all the data again, click Stop Filter.

Save or share a private filter from the Related Actions menu in the filter banner. You can view all your private and shared filters, and any filters shared with you, from the Filter menu or the drop-down list in the filter banner. Each private filter you save or share applies only to a single sheet in the workbook.

Using Filters

You can use filters on either static or live data in workbooks (or a range that contains both), as well as in entry areas, which are Worksheets integrations with other products. For advanced report live data, you can't filter on a subset of the data because Worksheets applies filters to the entire table. For data from matrix or composite reports, Worksheets applies filters to all rows within the selected range. If you apply a filter to live data and then refresh the data, Worksheets automatically recalculates the worksheet and reapplies the filter criteria to the live data area. For large worksheets delays may occur.

Note: SUBTOTAL or AGGREGATE functions in formulas don't respond if you apply a private or shared filter. Example: If a formula contains the SUBTOTAL function that looks at A1:A100, and you apply a filter that results in only 5 of those rows being returned, the results of the SUBTOTAL function apply for all 100 rows and not just the 5 filtered ones.

Any action you take with the data when using a filter, such as editing, formatting, or sorting, affects the data in all filters. You can edit data and then use the Reapply Filter button in the banner to exclude any rows that no longer meet the filter criteria.

Public, private, and shared filters can't be used with images, charts, and pivot tables, or when displaying workbooks on mobile devices.

Concept: Automatic Subtotaling and Grouping

Worksheets can automatically calculate subtotals for sets of related data in your workbook, and calculate a grand total. You can also manually create groupings (also called outlines) of data.

Keep these considerations in mind:

- The data must have column headings, including a field (heading) that identifies the group.
- The data must be sorted according to those groups. Example: if you want to subtotal by month, all rows for the individual months must be contiguous.
- Subtotaling looks visually similar in the user interface to Workday composite reports.
- Subtotaling isn't supported in entry areas such as plan entry areas or project entry areas.

Steps

1. Click in a cell containing a value, which you want to subtotal by, then select Data > Subtotal.
2. In the Subtotal dialog, complete these fields:

At each change in	Select the column heading for the data you want to group by.
Use function	Select SUM for a subtotal, or one of the other formulas.
Add subtotal to	Select one or more columns to place the resulting subtotals in.

Results

Worksheets inserts a SUBTOTAL formula into the appropriate cells based on your selections in the Subtotals dialog, placing a subtotal row between each group, and a grand total row at the bottom of the data set.

Additionally, to the left of the data, you see numbered Group buttons indicating the levels of grouped data. You can expand or collapse the data details by clicking the numbered buttons or by clicking the + or – buttons that display vertically alongside the data.

Example

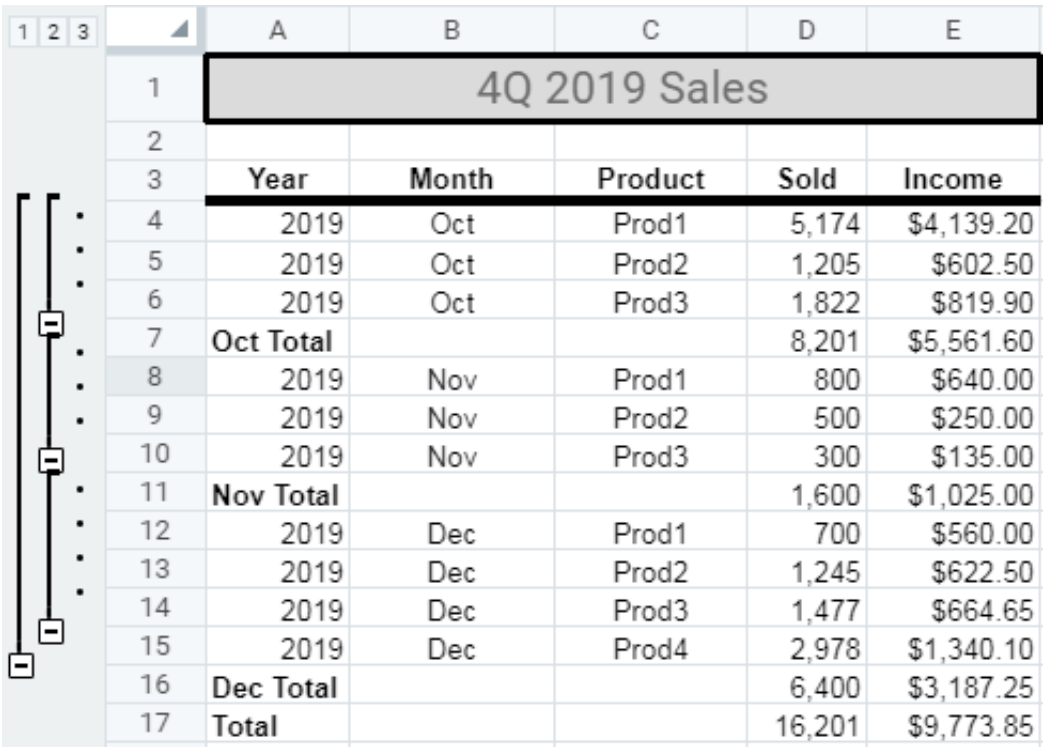
This is a data set that we can add subtotals to:

	A	B	C	D	E
1	4Q 2019 Sales				
2					
3	**Year**	**Month**	**Product**	**Sold**	**Income**
4	2019	Oct	Prod1	5,174	$4,139.20
5	2019	Oct	Prod2	1,205	$602.50
6	2019	Oct	Prod3	1,822	$819.90
7	2019	Nov	Prod1	800	$640.00
8	2019	Nov	Prod2	500	$250.00
9	2019	Nov	Prod3	300	$135.00
10	2019	Dec	Prod1	700	$560.00
11	2019	Dec	Prod2	1,245	$622.50
12	2019	Dec	Prod3	1,477	$664.65
13	2019	Dec	Prod4	2,978	$1,340.10

In the Subtotals dialog, we select these values:

At each change in	Month
Use function	SUM
Add subtotal to	Sold and Income

The resulting workbook looks like this:



	A	B	C	D	E
1	4Q 2019 Sales				
2					
3	Year	Month	Product	Sold	Income
4	2019	Oct	Prod1	5,174	$4,139.20
5	2019	Oct	Prod2	1,205	$602.50
6	2019	Oct	Prod3	1,822	$819.90
7	Oct Total			8,201	$5,561.60
8	2019	Nov	Prod1	800	$640.00
9	2019	Nov	Prod2	500	$250.00
10	2019	Nov	Prod3	300	$135.00
11	Nov Total			1,600	$1,025.00
12	2019	Dec	Prod1	700	$560.00
13	2019	Dec	Prod2	1,245	$622.50
14	2019	Dec	Prod3	1,477	$664.65
15	2019	Dec	Prod4	2,978	$1,340.10
16	Dec Total			6,400	$3,187.25
17	Total			16,201	$9,773.85

Removing Subtotals

If you want to remove the subtotals, select a cell in the subtotaled data set, select Data > Subtotal, and click Remove Subtotal.

Grouping

You can manually group (outline) related data in your workbook without adding subtotals. To do so, select the rows or columns that you want to group, then right-click and select Group. After grouping, you can select to Ungroup a single group, or select Ungroup All to remove associated groupings.

Create and Edit Pivot Tables in Workbooks

Context

Pivot tables enable you to summarize and analyze the significance of large amounts of data. The Pivot Table Wizard and panel enable you to create and edit pivot tables interactively.

Notes:

- The maximum number of data points that Worksheets can generate in a pivot table is 1 million (1,000,000).
- Pivot tables containing live data rely on the report column names from the Workday report. If you change a report column name in the report, and that data is used as a field in a pivot table, you need to replace the pivot table field associated with the original name with the field that's based on the new report column name.
- Pivot tables don't support calculations when the data range contains multiple currency units. Example: If you have a pivot table that includes salaries, and some salaries are in USD while others are in CAD, you can't do a SUM, AVERAGE, or other calculation on the data; if you hover over the error cell you see *A unit conversion issue exists*. The only valid function in this situation is COUNTA.
- You can't use the formula editor for pivot table formulas.

- If you change the name of a pivot table, remember that the name must be unique in the workbook and also can't be the same as any defined names or live data table names.
- If you change the name of a pivot table, and the pivot table is being used in a Slides presentation, the new name won't display in Slides until you refresh the data in the Linked Data panel.
- Pivot tables require that all source data reside on the same workbook sheet. The sheet can include both live data and static data.

Steps

1. From the sheet containing the data that you want a pivot table for, select the range to include and click Insert > Pivot Table. You can move the popup around the sheet if needed to see your data.
2. (Optional) In the Source Data field, edit the range to use when creating the table.
3. (Optional) For pivot tables where *all* the source data is in an array (live data): Select Auto-update as unconstrained array data adds or removes rows to automatically update the pivot table if an array size change causes rows to be added or deleted.

   Note that the pivot table doesn't automatically update if *columns* are added or removed from the Workday report.
4. Select whether to create a pivot table on a new sheet or an existing sheet.

   If you choose Existing sheet, type the starting cell for the pivot table or click the Select icon and select the cell.
5. Click OK. The pivot table displays and the pivot table panel opens.
6. Configure the table as desired using the following fields and options. When you're finished, click Update to make your changes take effect.

Field	Notes
Columns Rows	From the fields area, place one or more fields into the Rows or Columns area to include them in the pivot table. You can click the + icon and select an area, or drag the field. If you have lots of fields, you might want to use Search to find it quickly. Click the Filter icon to sort the displayed values or filter the data. You can either sort by the column or row, or by any of the value fields. Click the Menu Options (three dots) icon to type a custom Display Name in the header for the field, or to show subtotals.
Values	Click the Menu Options (three dots) icon to specify a Display Name, a Summarize by function, a Show data as option, and a Formatting option. Summarize by functions: • AVERAGE • COUNT: Returns the number of specified values, counting only numbers and string representations of numbers. • COUNTA: Returns the number of specified values, counting all data types. • COUNTNZ: Counts non-zero numbers. If a value is a string representation of a number, Worksheets converts it to a number. • DISTINCT: Counts distinct (not duplicate) values. • FIRST • LAST • MIN

Field	Notes
	• MAX • STDEV: Estimates the standard deviation. • SUM • VARIANCE: Estimates the variance. In the Formatting drop-down menu, the available formats match the selections in the Format > Number menu. Worksheets retains the formatting when the related live data refreshes.
Filter	Place one or more fields here to show only a subset of the data for those fields or to sort the data. Public, private, and shared filters aren't supported for pivot tables.

Next Steps

You can change the source data range for the table by clicking the Source Data icon.

Anyone with view access or higher for the workbook can view detailed information about a pivot table value. Select a single pivot value, right-click, and select Show Details. If you have edit or owner permission, you can optionally create a new sheet for the data by clicking Create Sheet.

If you're the owner or an editor for the workbook, you can quickly generate the formula that produces a specific pivot table value. This formula enables you to get related data from a pivot table without directly referencing cells, so the reference is preserved if you change the pivot table in the future. Select a single pivot value, right-click, and select Copy Pivot Data Formula. The resulting formula uses the GETPIVOTDATA function.

You can change settings for the entire table by clicking the Settings (gear) icon:

Field	Notes
Pivot Name	Pivot table names can contain letters, numbers, and underscores, and they must be between 1 and 255 characters long. Pivot table names must be unique across the workbook, and the name must not be the same as a live data area table name.
Table Style	Select a predefined color and highlighting pattern from the drop-down menu.
Source Data	Display only. To change the source data range, click the Source Data icon.
Grand Totals	Select whether to display row or column grand totals.
Repeat Labels	Select whether to repeat row or column labels.
Format	Select whether to display custom text for error cells or for empty cells.

Worksheets Function Reference

Reference: Worksheets-Unique Functions

Worksheets functions help you organize and manage your data to see trends and obtain insights.

The functions listed in this table distinguish Worksheets from most other spreadsheet products by providing additional value for data analysis.

Functions in bold text are designed for use as unconstrained array formulas. Submit unconstrained array formulas with the special keyboard shortcut Ctrl+Alt+Enter (Windows) or Command+Option+Enter (Mac) so Worksheets can use the entire range it needs when returning the results.

As a best practice, use absolute references in Worksheets-unique functions wherever possible to minimize the use of system resources. To make a cell reference absolute, add a dollar sign ($) in front of the column letter and row number in the formula. Examples: $Q$45, $T$67, $A$1:$A$10, etc.

The Workday Administrator Guide describes the Worksheets-unique functions. The Worksheets User Guide describes all the functions in Worksheets.

Function Type	Function
Date	• CONVERTTZ • DATESBETWEEN • DATESFROM • DATETIME • NOWTZ • WD.DATEDIF
Engineering	• DIMENSIONS • IMCOTH • IMTANH • SETUNITS • UNITS
Information	• ISBOOLEAN
Logical	• IFEMPTY
Lookup	• ARRAYAREA • IN • MATCHEXACT • MHLOOKUP • MVLOOKUP • WD.MVLOOKUP • WD.VLOOKUP
Math	• CBRT • E • EXPM1 • HYPOT • LOG1P • RANDCONST • RINT • WD.MAXIFS • WD.MINIFS • WD.SUMIF • WD.SUMIFS
Matrix	• MIDENTITY • TRUNCATEMATRIX
Miscellaneous	• CAPPEDVALUES • CLUSTER.KMEANS

Function Type	Function
	• CLUSTER.KMEANS.CENTROIDS • CORRELATE • DISTINCTROWS • DISTINCTROWS2 • EMAIL • GROUPBY • INSTANCE • INSTANCE.DESCRIPTOR • INSTANCE.ID • JOIN • MATCHCOMPOSITE • MERGECOLUMNS • MERGEROWS • MI.COUNT • MI.INDEX • MINUS • MULTIINST • NOTIFYIF • NOTIFYIFS • REMOVECOLUMNS • REMOVEROWS • SELECT • SORT • SORT2 • SORT3 • TRIMCOLUMNS • TRIMROWS • UNIQUE • UNIQUE2 • URL • URLTEXT • WD.ARRANGECOLUMNS • WD.ARRANGEROWS • WD.LIVEDATA • WD.SLICE
Operator	• COMPARE • SUBTRACT
Statistical	• WD.AVERAGEIF • WD.AVERAGEIFS • WD.COUNTIF • WD.COUNTIFS • FORECAST.WD.SEASONAL • TDISTRT
Table	• FLATTEN • VALUEAT
Text	• REGEXFIND • REGEXPARSE

Reference: Worksheets Functions by Category

This table lists the Worksheets functions, sorted according to category. When you view a function in the Functions Library panel, you can identify its category by the icon to the left of the name.

The Workday Administrator Guide describes the Worksheets-unique functions. The Worksheets User Guide describes all the functions in Worksheets.

Function Type	Function
Array	• CHOOSECOLS • CHOOSEROWS • DROP • EXPAND • FILTER • HSTACK • MS.SORT • MS.UNIQUE • MS.UNIQUE2 • RANDARRAY • SEQUENCE • SORTBY • TAKE • TOROW • TOCOL • VSTACK • WRAPROWS • WRAPCOLS • XLOOKUP • XMATCH
Date	• CONVERTTZ • DATE • DATESBETWEEN • DATESFROM • DATETIME • DATEVALUE • DAY • DAYNAME • DAYS • DAYS360 • DUR2DAYS • DUR2HOURS • DUR2MILLISECONDS • DUR2MINUTES • DUR2SECONDS • DUR2WEEKS • DURATIONVALUE • EDATE • EOMONTH • HOUR • ISOWEEKNUM

Function Type	Function
	• MINUTE • MONTH • MONTHNAME • NETWORKDAYS • NETWORKDAYS.INTL • NOW • NOWTZ • SECOND • TIME • TIMEVALUE • TODAY • WD.DATEDIF • WEEKDAY • WEEKNUM • WORKDAY • WORKDAY.INTL • YEAR • YEARFRAC
Engineering	• BESSELI • BESSELJ • BESSELK • BESSELY • BIN2DEC • BIN2HEX • BIN2OCT • BITAND • BITLSHIFT • BITOR • BITRSHIFT • BITXOR • COMPLEX • CONVERT • DEC2BIN • DEC2HEX • DEC2OCT • DELTA • DIMENSIONS • ERF • ERF.PRECISE • ERFC • ERFC.PRECISE • GESTEP • HEX2BIN • HEX2DEC • HEX2OCT • IMABS • IMAGINARY • IMARGUMENT

Function Type	Function
	• IMCONJUGATE • IMCOS • IMCOSH • IMCOT • IMCOTH • IMCSC • IMCSCH • IMDIV • IMEXP • IMLN • IMLOG10 • IMLOG2 • IMPOWER • IMPRODUCT • IMREAL • IMSEC • IMSECH • IMSIN • IMSINH • IMSQRT • IMSUB • IMSUM • IMTAN • IMTANH • OCT2BIN • OCT2DEC • OCT2HEX • SETUNITS • UNITS
Financial	• ACCRINT • ACCRINTM • AMORDEGRC • AMORLINC • BONDDURATION • BONDMDURATION • COUPDAYBS • COUPDAYS • COUPDAYSNC • COUPNCD • COUPNUM • COUPPCD • CUMIPMT • CUMPRINC • DB • DDB • DISC • DOLLARDE • DOLLARFR

Function Type	Function
	• DURATION • EFFECT • FV • FVSCHEDULE • INTRATE • IPMT • IRR • ISPMT • MDURATION • MIRR • NOMINAL • NPER • NPV • ODDFYIELD • ODDLPRICE • ODDLYIELD • PMT • PPMT • PRICE • PRICEDISC • PRICEMAT • PV • RATE • RECEIVED • RRI • SLN • SYD • TBILLEQ • TBILLPRICE • TBILLYIELD • VDB • XIRR • XNPV • YIELD • YIELDDISC • YIELDMAT
Information	• ERROR.TYPE • INFO • ISBLANK • ISBOOLEAN • ISERR • ISERROR • ISEVEN • ISFORMULA • ISLOGICAL • ISNA • ISNONTEXT • ISNUMBER

Function Type	Function
	• ISODD • ISREF • ISTEXT • N • NA • TYPE
List	• WD.LIST.GET • WD.LIST.LIST • WD.LIST.SIZE
Logical	• AND • FALSE • IF • IFEMPTY • IFERROR • IFNA • NOT • OR • SWITCH • TRUE • XOR
Lookup	• ADDRESS • AREAS • ARRAYAREA • CHOOSE • COLUMN • COLUMNS • FORMULATEXT • HLOOKUP • IN • INDEX • INDIRECT • LOOKUP • MATCH • MATCHEXACT • MHLOOKUP • MVLOOKUP • OFFSET • ROW • ROWS • VLOOKUP • WD.MVLOOKUP • WD.VLOOKUP
Math	• ABS • ACOS • ACOSH • ACOT • ARABIC

Function Type	Function
	• ASIN • ATAN • ATAN2 • ATANH • BASE • BASETONUM • CBRT • CEILING • CEILING.MATH • CEILING.PRECISE • COMBIN • COMBINA • COS • COSH • COT • COTH • CSC • CSCH • DECIMAL • DEGREES • E • EVEN • EXP • EXPM1 • EXPONENT • FACT • FACTDOUBLE • FLOOR • FLOOR.MATH • FLOOR.PRECISE • GCD • HYPOT • INT • ISO.CEILING • ISOCEILING • LCM • LN • LOG • LOG10 • LOG1P • MAXIFS • MDETERM • MINIFS • MOD • MROUND • MULTINOMIAL • ODD • PERCENTOF

Function Type	Function
	• PI • POWER • PRODUCT • QUOTIENT • RADIANS • RAND • RANDBETWEEN • RANDCONST • RINT • ROMAN • ROUND • ROUNDDOWN • ROUNDUP • SEC • SECH • SERIESSUM • SIGN • SIN • SINH • SQRT • SQRTPI • SUBTOTAL • SUM • SUMIF • SUMIFS • SUMPRODUCT • SUMSQ • SUMX2MY2 • SUMX2PY2 • SUMXMY2 • TAN • TANH • TRUNC • WD.AVERAGEIFS • WD.MAXIF • WD.MAXIFS • WD.MINIF • WD.MINIFS • WD.SUMIF • WD.SUMIFS
Matrix	• MIDENTITY • MMULT • MUNIT • TRANSPOSE • TRUNCATEMATRIX
Miscellaneous	• BYCOL • BYROW • CAPPEDVALUES

Function Type	Function
	• CELL • CLUSTER.KMEANS • CLUSTER.KMEANS.CENTROIDS • CORRELATE • DISTINCTROWS • DISTINCTROWS2 • EMAIL • GROUPBY • HYPERLINK • INSTANCE • INSTANCE.DESCRIPTOR • INSTANCE.ID • ISOMITTED • JOIN • LAMBDA • LET • MAKEARRAY • MAP • MATCHCOMPOSITE • MERGECOLUMNS • MERGEROWS • MI.COUNT • MI.INDEX • MINUS • MS.GROUPBY • MULTIINST • NOTIFYIF • NOTIFYIFS • PIVOTBY • REDUCE • REMOVECOLUMNS • REMOVEROWS • SCAN • SELECT • SORT • SORT2 • SORT3 • TRIMCOLUMNS • TRIMRANGE • TRIMROWS • UNIQUE • UNIQUE2 • URL • URLTEXT • WD.ARRANGECOLUMNS • WD.ARRANGEROWS • WD.EXCHANGE • WD.GROUPBY

Function Type	Function
	• WD.LIVEDATA • WD.PIVOTBY • WD.SLICE
Operator	• ADD • COMPARE • DIVIDE • EQ • GT • GTE • LT • LTE • MULTIPLY • NE • POW • SUBTRACT • UMINUS • UNARY_PERCENT • UPLUS
Statistical	• AGGREGATE • AVEDEV • AVERAGE • AVERAGEA • AVERAGEIF • AVERAGEIFS • BINOM.DIST • BINOM.INV • BINOMDIST • BINOMINV • CHIDIST • CHISQ.DIST • CHISQ.DIST.RT • CHISQDIST • CHISQDISTRT • CONFIDENCE • CONFIDENCE.NORM • CONFIDENCE.T • CORREL • COUNT • COUNTA • COUNTBLANK • COUNTIF • COUNTIFS • COVARIANCE.P • COVARIANCE.S • COVARIANCEP • COVARIANCES • CRITBINOM • DEVSQ

Function Type	Function
	• EXPON.DIST • EXPONDIST • FISHER • FISHERINV • FORECAST • FORECAST.WD.SEASONAL • FREQUENCY • GAMMALN • GAMMALN.PRECISE • GAUSS • GEOMEAN • GROWTH • HARMEAN • HYPGEOM.DIST • HYPGEOMDIST • INTERCEPT • KURT • LARGE • LINEST • LOGEST • LOGINV • LOGNORM.DIST • LOGNORM.INV • LOGNORMDIST • LOGNORMINV • MAX • MAXA • MEDIAN • MIN • MINA • MODE • MODE.MULT • MODE.SNGL • NEGBINOM.DIST • NEGBINOMDIST • NORM.DIST • NORM.INV • NORM.S.DIST • NORM.S.INV • NORMDIST • NORMINV • NORMSDIST • NORMSINV • PEARSON • PERCENTILE • PERCENTILE.EXC • PERCENTILE.INC • PERCENTRANK

Function Type	Function
	• PERCENTRANK.INC • PERMUT • PERMUTATIONA • PHI • POISSON • POISSON.DIST • QUARTILE • QUARTILE.EXC • QUARTILE.INC • RANK • RANK.AVG • RANK.EQ • RANKAVG • RANKEQ • RSQ • SLOPE • SMALL • STANDARDIZE • STDEV • STDEV.P • STDEV.S • STDEVP • STDEVS • T.DIST • T.DIST.RT • T.INV • TDIST • TDISTRT • TINV • TRIMMEAN • VAR • VAR.P • VARA • VARP • VARPA • WD.AVERAGEIF • WD.COUNTIF • WD.COUNTIFS • WEIBULL • WEIBULL.DIST • WEIBULLDIST • Z.TEST • ZTEST
Text	• ARRAYTOTEXT • ASC • CHAR • CLEAN • CODE

Function Type	Function
	• CONCAT • CONCATENATE • ENCODEURL • EXACT • FIND • FIXED • LEFT • LEN • LOWER • MID • MS.REGEXEXTRACT • MS.REGEXREPLACE • MS.REGEXTEST • PROPER • REGEXEXTRACT • REGEXFIND • REGEXMATCH • REGEXPARSE • REGEXREPLACE • REGEXTEST • REPLACE • REPT • RIGHT • SEARCH • SPLIT • SUBSTITUTE • T • TEXT • TEXTAFTER • TEXTBEFORE • TEXTJOIN • TEXTSPLIT • TRIM • UNICHAR • UNICODE • UPPER • VALUE • VALUETOTEXT • WD.HALFTOFULL • WD.FULLTOHALF
Table	• FLATTEN • VALUEAT

Reference: Formula Errors

Error	Notes
#DIV/0!	The formula is trying to divide by zero. Example: You might have a formula where a cell value/result is unexpectedly zero or blank, and the formula is trying to divide by that number.
#ERROR!	A rare error. Most error messages are fairly specific, but #ERROR might merely specify that a syntax error exists, or that you can't submit the formula for an unknown reason.
#FIELD!	A data type error exists.
#GETTING_DATA!	Not a true error, but it can display temporarily in workbook cells when large or complex calculations are in progress, or when a live data refresh is running. The message disappears when the calculations are complete or the live data refresh is complete.
#N/A!	A value isn't available to a function or formula. Examples: • A lookup function can't find a key value. • An array formula has an argument that's a different size (cell range) from another argument. • A function has 1 or more missing arguments.
#NAME!	You referenced a defined name or a function that doesn't exist.
#NULL!	You might have: • Used an incorrect range or reference separator such as a space (separating individual cell references in a formula with a space instead of a comma). • Used a space (the intersect operator) to do a calculation on intersecting ranges but there was no intersection.
#NUM!	A formula has invalid numeric data for the type of operation. Example: Pivot tables don't support calculations when the data range contains multiple currency units. If you have a pivot table that includes salaries, and some salaries are in USD while others are in CAD, you can't do a SUM, AVERAGE, or other calculation on the data; if you hover over the error cell you see *A unit conversion issue exists*. The only valid function in this situation is COUNTA. To help troubleshoot a #NUM error that might be a units problem, you can view the units settings by selecting View > Show Units.
#REF!	A reference isn't valid.
#SPILL!	The most frequent cause of this error is that array formula results can't be placed into the workbook because the results would overlap existing data. When in an entry area, #SPILL occurs when the formula attempts to place data across the boundary of the entry area. It's common to "clear" a cell by typing a space character followed by Enter, Tab, or an arrow key. This doesn't really clear the cell; it leaves a single space character in it, and can cause formulas not to work correctly. Example: Non-empty calls can cause spill errors when calculating array formulas, or can cause unexpected results with COUNTBLANK or other issues. There isn't a visual indicator for cells with spaces in them, so when in doubt, manually clear the cells..

Error	Notes
#VALUE!	You used the wrong type of operand or function argument. Example: You see #VALUE if the formula finds spaces, characters, or text where it's expecting a number.
#######	Not actually an error; your column isn't wide enough to display a value.

Tips for Formula Errors

Remember that you must submit array formulas with the appropriate keyboard shortcut. See Concept: Array Formulas in Workbooks for information about constrained arrays, unconstrained arrays, and more.

If you see an error related to circular references, you might want to review Concept: Circular References in the Worksheets User Guide.

Take note of the workbook setting for automatic or manual recalculation in File > Settings.

Try to simplify complex formulas:

- If the formula contains lots of nested functions, separate out one or more of them, placing their results in a separate range that you refer to in the rest of the original formula.
- If the formula has external references that appear not to resolve, try merging the workbooks so that the reference is merely to another sheet instead of to a workbook; or, put all the data on 1 sheet.

Displaying User-Friendly Text Instead of Error Codes

Sometimes you use formulas that do calculations based on cell values, and if one of those values is unexpected you end up with an error such as a #DIV/0 error. If you want to plan for the possibility of an error and give the workbook user a more friendly message, you can use:

- The IFERROR function or the ISERROR function.
- The IFEMPTY formula for unconstrained array formulas where the result is an empty array.

Example: We have a workbook that shows salespeople's average bonuses for a quarter. To get a bonus, a salesperson must sell more than $5,000 in goods in a particular month. Because Matt didn't get any bonuses, the Avg Bonus cell would contain a #DIV/0 error. To prevent this unfriendly error from displaying, we can nest our AVERAGEIF function in an IFERROR. This formula is in F2:

=IFERROR(AVERAGEIF(B2:D2,">5000",B2:D2),"No bonuses")

	A	B	C	D	E	F
1	Name	Jan Sales	Feb Sales	Mar Sales	# Bonuses	Avg Bonus
2	Matt Bond	$1500	$3275	$2900	Zero	No bonuses

The ISERROR function is similar to IFERROR, but it returns a TRUE or FALSE result based on whether an error exists.

Reference: Worksheets Rounding Functions

Worksheets supports the same rounding functions as other popular spreadsheet applications. These tables can help you decide when to use each one.

Rounding a Number to an Integer

Function	Rounds...	For Positive Numbers	For Negative Numbers
EVEN	Away from zero to next even number	Result becomes more positive	Result becomes more negative
ODD	Away from zero to next odd number	Result becomes more positive	Result becomes more negative
INT	Down to next integer below	Result becomes more negative	Result becomes more negative

Rounding a Number Using Decimal Places

Function	Rounds...	For Positive Numbers	For Negative Numbers
ROUND	Up or down to the closest value, based on the number of decimal places you specify	Result becomes more positive or negative based on the closest value	Result becomes more positive or negative based on the closest value
ROUNDUP	Away from zero, based on the number of decimal places you specify	Result becomes more positive	Result becomes more negative
ROUNDDOWN and TRUNC	Toward zero, based on the number of decimal places you specify	Result becomes more negative	Result becomes more positive

Rounding a Number Using a Multiple of Significance (MoS)

Function	Rounds...	For Positive Numbers with a Positive MoS	For Positive Numbers with a Negative MoS	For Negative Numbers with a Positive MoS	For Negative Numbers with a Negative MoS
CEILING	Away from zero, based on the MoS you specify	Result becomes more positive	Not applicable (#NUM error)	Result becomes more negative	Result becomes less negative
CEILING.PRECISE and ISO.CEILING/ ISOCEILING	Up, based on the MoS you specify	Result becomes more positive	Result becomes more positive	Result becomes less negative	Result becomes less negative
CEILING.MATH	Up, based on the MoS you specify	Result becomes more positive	Result becomes more positive	Result becomes less negative (or use mode to reverse it)	Result becomes less negative (or use mode to reverse it)
FLOOR	Towards zero, based on the MoS you specify	Result becomes less positive	Not applicable (#NUM error)	Result becomes less negative	Result becomes more negative

Function	Rounds...	For Positive Numbers with a Positive MoS	For Positive Numbers with a Negative MoS	For Negative Numbers with a Positive MoS	For Negative Numbers with a Negative MoS
FLOOR.PRECISE	Down, based on the MoS you specify	Result becomes less positive	Result becomes less positive	Result becomes more negative	Result becomes more negative
FLOOR.MATH	Down, based on the MoS you specify	Result becomes less positive	Result becomes less positive	Result becomes more negative (or use mode to reverse it)	Result becomes more negative (or use mode to reverse it)
MROUND	Up or down, based on the MoS you specify	Result becomes more positive or negative, based on the closest multiple	Not applicable (#NUM error)	Result becomes more positive or negative, based on the closest multiple	Not applicable (#NUM error)

Reference: Measurement Units in Worksheets Functions

These tables list the primary units of measurement that Worksheets functions support. Use the abbreviations in formulas.

For more information, see the Java library `javax.measure.unit.Dimension`.

Abbreviations are case-sensitive.

Make sure that all unit types in a formula are compatible. Example: Don't try to convert a distance unit to a time unit.

Weight and Mass

Abbreviation	Name
g	Gram
u	U (Atomic Mass Unit)
grain	Grain
uk_cwt	Imperial hundredweight
lcwt	Imperial hundredweight
hweight	Imperial hundredweight
stone	Stone
ton	Ton
uk_ton	Imperial ton
LTON	Imperial ton
brton	Imperial ton

Distance

Abbreviation	Name
m	Meter
mi	Statute mile
Nmi	Nautical mile
in	Inch
ft	Foot
yd	Yard
ang	Angstrom
ly	Light year
Picapt	Picapt (point: 1/72 inch)
Pica	Pica (1/72 inch)
pica	pica (1/6 inch)
survey_mi	US survey mile (statute mile)

Time

Abbreviation	Name
yr	Year
day	Day
d	Day
hr	Hour
mn	Minute
min	Minute
sec	Second
s	Second

Energy

Abbreviation	Name
J	Joule
c	Thermodynamic calorie
cal	IT calorie
eV	Electron volt
HPh	Horsepower-hour
hh	Horsepower-hour

Temperature

Abbreviation	Name
C	Degree Celsius
cel	Degree Celsius
F	Degree Fahrenheit
fah	Degree Fahrenheit
K	Kelvin
kel	Kelvin

Volume (Liquid Measure)

Abbreviation	Name
tsp	Teaspoon
tbs	Tablespoon
oz	Ounce
cup	Cup
pt	Pint
us_pt	US pint
uk_pt	UK pint
qt	Quart
uk_qt	Imperial quart
gal	Gallon
uk_gal	Imperial gallon
l	Liter
L	Liter
lt	Liter
barrel	US oil barrel
bushel	US bushel
ft^3	Cubic feet
in^3	Cubic inch
m^3	Cubic meter
mi^3	Cubic mile
yd^3	Cubic yard
Pica^3	Cubic Pica
MTON	Measurement ton (freight ton)

Area

Abbreviation	Name
uk_acre	International acre
us_acre	US survey/statute acre
ang^2	Square angstrom
ft^2	Square feet
in^2	Square inches
m^2	Square meters
mi^2	Square miles
Picapt^2	Square Pica
Pica^2	Square Pica
yd^2	Square yards

Information

Abbreviation	Name
bit	Bit
byte	Byte

Speed

Abbreviation	Name
m/h	Meters per hour
m/hr	Meters per hour
m/s	Meters per second
m/sec	Meters per second
mph	Miles per hour

Persons

Abbreviation	Name
person	Person
pers	Person
emp	Employee
hc	Headcount
head	Head

Currency

Abbreviation	Name
AFA	Afghani
AFN	Afghani
ALK	Albanian Old Lek
DZD	Algerian Dinar
ADF	Andorran Franc
ADP	Andorran Peseta
AOR	Angolan Kwanza Readjustado
ARS	Argentine Peso
AMD	Armenian Dram
AWG	Aruban Florin
AUD	Australian Dollar
ATS	Austrian Schilling
AZM	Azerbaijani Manat
AZN	Azerbaijanian Manat
BSD	Bahamian Dollar
BHD	Bahraini Dinar
THB	Baht
PAB	Balboa
BBD	Barbados Dollar
BYR	Belarussian Ruble
BEF	Belgian Franc
BZD	Belize Dollar
BMD	Bermudian Dollar
VEF	Bolivar
BOB	Boliviano
BRL	Brazilian Real
BND	Brunei Dollar
BGN	Bulgarian Lev
BIF	Burundi Franc
CVE	Cabo Verde Escudo
CAD	Canadian Dollar
KYD	Cayman Islands Dollar
XOF	CFA Franc BCEAO

Abbreviation	Name
XAF	CFA Franc BEAC
XPF	CFP Franc
CLP	Chilean Peso
COP	Colombian Peso
KMF	Comoro Franc
CDF	Congolese Franc
BAM	Convertible Mark
NIO	Cordoba Oro
CRC	Costa Rican Colon
HRK	Croatian Kuna
CUP	Cuban Peso
CYP	Cyprus Pound
CZK	Czech Koruna
GMD	Dalasi
DKK	Danish Krone
MKD	Denar
DJF	Djibouti Franc
STD	Dobra
DOP	Dominican Peso
VND	Dong
NLG	Dutch Guilder
XCD	East Caribbean Dollar
ECS	Ecuador Sucre
ECV	Ecuador Unidad de Valor Constante
TJR	Tajikistani ruble
EGP	Egyptian Pound
SVC	El Salvador Colon
ETB	Ethiopian Birr
EUR	Euro
XEU	European Currency Unit
FKP	Falkland Islands Pound
FJD	Fiji Dollar
FIM	Finnish Markka
HUF	Forint

Abbreviation	Name
FRF	French Franc
DEM	German Mark
GHC	Ghana Cedi
GHS	Ghana Cedi
GIP	Gibraltar Pound
XFO	Gold Franc
HTG	Gourde
GRD	Greek Drachma
PYG	Guarani
GNF	Guinea Franc
GWP	Guinea-Bissau Peso
GYD	Guyana Dollar
HKD	Hong Kong Dollar
UAH	Hryvnia
ISK	Iceland Krona
INR	Indian Rupee
IRR	Iranian Rial
IQD	Iraqi Dinar
IEP	Irish Pound
ITL	Italian Lira
JMD	Jamaican Dollar
JOD	Jordanian Dinar
KES	Kenyan Shilling
PGK	Kina
LAK	Kip
EEK	Kroon
KWD	Kuwaiti Dinar
MWK	Kwacha
AOA	Kwanza
MMK	Kyat
GEL	Lari
LVL	Latvian Lats
LBP	Lebanese Pound
ALL	Lek

Abbreviation	Name
HNL	Lempira
SLL	Leone
LRD	Liberian Dollar
LYD	Libyan Dinar
SZL	Lilangeni
LTL	Lithuanian Litas
LSL	Loti
LUF	Luxembourgian Franc
MGA	Malagasy Ariary
MGF	Malagasy Franc
MYR	Malaysian Ringgit
MTL	Maltese Lira
TMM	Manat
MUR	Mauritius Rupee
MXN	Mexican Peso
MXV	Mexican Unidad de Inversion (UDI)
MDL	Moldovan Leu
MCF	Monegasque Franc
MAD	Moroccan Dirham
MZM	Mozambique Metical
MZN	Mozambique Metical
BOV	Mvdol
NGN	Naira
ERN	Nakfa
NAD	Namibia Dollar
NPR	Nepalese Rupee
ANG	Netherlands Antillean Guilder
ILS	New Israeli Sheqel
RON	New Romanian Leu
TWD	New Taiwan Dollar
NZD	New Zealand Dollar
BTN	Ngultrum
KPW	North Korean Won
NOK	Norwegian Krone

Abbreviation	Name
PEN	Nuevo Sol
OMR	Omani Rial
MRO	Ouguiya
TOP	Pa'anga
PKR	Pakistan Rupee
MOP	Pataca
UYU	Peso Uruguayo
PHP	Philippine Peso
PTE	Portugese Escudo
GBP	Pound Sterling
BWP	Pula
QAR	Qatari Rial
GTQ	Quetzal
ZAR	Rand
KHR	Riel
ROL	Romanian Leu
MVR	Rufiyaa
IDR	Rupiah
RUB	Russian Ruble
RWF	Rwanda Franc
SHP	Saint Helena Pound
SML	San Marinese Lira
SAR	Saudi Riyal
CSD	Serbian Dinar
RSD	Serbian Dinar
SCR	Seychelles Rupee
SGD	Singapore Dollar
SKK	Slovak Koruna
SIT	Slovenian Tolar
SBD	Solomon Islands Dollar
KGS	Som
SOS	Somali Shilling
TJS	Somoni
SSP	South Sudanese Pound

Abbreviation	Name
ESP	Spanish Peseta
LKR	Sri Lanka Rupee
SDD	Sudanese Dinar
SDG	Sudanese Pound
SRD	Surinam Dollar
SRG	Suriname Guilder
SEK	Swedish Krona
CHF	Swiss Franc
SYP	Syrian Pound
BDT	Taka
WST	Tala
TZS	Tanzanian Shilling
KZT	Tenge
TTD	Trinidad and Tobago Dollar
MNT	Tugrik
TND	Tunisian Dinar
TRL	Turkish Lira
TRY	Turkish Lira
TMT	Turkmenistan New Manat
AED	UAE Dirham
UGX	Uganda Shilling
COU	Unidad de Valor Real
CLF	Unidades de fomento
UYI	Uruguay Peso en Unidades Indexadas
USD	US Dollar
UZS	Uzbekistan Sum
VAL	Vatican Coins
VUV	Vatu
VEB	Venezuelan Bolívar
CHE	WIR Euro
CHW	WIR Franc
KRW	Won
YER	Yemeni Rial
JPY	Yen

Abbreviation	Name
CNY	Yuan Renminbi
YUM	Yugoslav Dinar
ZMK	Zambian Kwacha
ZMW	Zambian Kwacha
ZWD	Zimbabwe Dollar
ZWL	Zimbabwe Dollar

All Worksheets Functions

ABS

Description

Returns the absolute value of a number. The absolute value of a number is the number without its sign.

Syntax

`ABS(value)`

- value: The real number to get the absolute value for.

Example

Formula	Result
=ABS(12)	12
=ABS(-5)	5
=ABS(20-25)	5

ACCRINT

Description

Returns the accrued interest for a security that pays periodic interest.

Syntax

```
ACCRINT(issue, first_interest, settlement, rate, [ par], frequency, [ basis],
[ calc_method])
```

- issue: The security's issue date.
- first_interest: The security's first interest date.
- settlement: The security's settlement date. The security settlement date is the date after the issue date when the security is traded to the buyer.
- rate: The security's annual coupon rate.
- par: The security's par value (principal to pay). If you omit par, ACCRINT uses $1,000.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.

- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.
- calc_method: The logical value that specifies how to calculate the total accrued interest when the settlement date is later than the first_interest date. A value of TRUE (1) returns the total accrued interest from issue to settlement. A value of FALSE (0) returns the accrued interest from first_interest to settlement. If you do not enter the argument, it defaults to TRUE.

ACCRINTM

Description

Returns the accrued interest for a security that pays at maturity.

Syntax

```
ACCRINTM(issue, settlement, rate, par, [ basis])
```

- issue: The security's issue date.
- settlement: The security's settlement date.
- rate: The security's annual coupon rate.
- par: The security's par value (principal to pay at maturity).
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

ACOS

Description

Returns the inverse cosine of the value that you specify, in radians.

Syntax

```
ACOS(value)
```

- value: The value between -1 and 1 to calculate the inverse cosine for.

ACOSH

Description

Returns the inverse hyperbolic cosine of the specified number.

Syntax

```
ACOSH(number)
```

- number: The value of 1 or greater to calculate the inverse hyperbolic cosine for.

ACOT

Description

Returns the inverse cotangent of the specified number.

Syntax

`ACOT(number)`

- number: The number to calculate the inverse cotangent for.

ADD

Description

Returns the sum of two numbers or lists of numbers.

Syntax

`ADD(number1, number2)`

- number1: The number, or list of numbers.
- number2: The number, or list of numbers.

Example

Formula	Result
=ADD(A1,A2) Where cell A1 contains `42` and A2 contains `43.3` .	85.3
=ADD({1, 2, 3, 4},{5, 6, 7, 8}) Note: Remember to press Ctrl+Enter because this is an array formula.	`6 8 10 12`

ADDRESS

Description

Returns a cell reference (address) as text.

Syntax

`ADDRESS(row_num, column_num, [ abs_num], [ a1_ref_style_flag], [ sheet_text])`

- row_num: The row number for the cell.
- column_num: The column number for the cell.
- abs_num: Specifies whether to use absolute or relative referencing. 1 = absolute; 2 = absolute row and relative column; 3 = absolute column and relative row; 4 = relative. The default is 1.
- a1_ref_style_flag: The cell reference style. TRUE or Empty = A1 style; this is the only style that Worksheets supports.
- sheet_text: The sheet name. If not specified, the reference is for the current sheet.

Example

Formula	Result
=ADDRESS(1,1)	$A$1
=ADDRESS(2,4,4,,"MySheet")	MySheet!D2

AGGREGATE

Description

Runs the aggregation evaluation that you specify. Optionally, you can ignore certain types of values. This function is intended for use on columns of data (vertical ranges), not rows of data.

Syntax

`AGGREGATE(function_num, [ options], ref1, [ ref2])`

- function_num: A number from 1 to 19 that specifies the aggregation function.

Number	Function
1	AVERAGE
2	COUNT
3	COUNTA
4	MAX
5	MIN
6	PRODUCT
7	STDEV.S
8	STDEV.P
9	SUM
11	VAR.P
12	MEDIAN
13	MODE.SNGL
14	LARGE
15	SMALL
16	PERCENTILE.INC
17	QUARTILE.INC
18	PERCENTILE.EXC
19	QUARTILE.EXC

- options: A number from 0-7 that specifies which values in the range to ignore. If you omit the argument, the function uses 0.

Number	Action
0	Ignore nested SUBTOTAL and AGGREGATE functions
1	Ignore hidden rows, nested SUBTOTAL and AGGREGATE functions
2	Ignore error values, nested SUBTOTAL and AGGREGATE functions
3	Ignore hidden rows, error values, nested SUBTOTAL and AGGREGATE functions
4	Ignore nothing

Number	Action
5	Ignore hidden rows
6	Ignore error values
7	Ignore hidden rows and error values

- ref1: The first argument; it can be a number, array, an array formula, or a reference to a range of cells.
- ref2: One or more additional arguments. These functions require a ref2 argument: LARGE, SMALL, PERCENTILE.INC, QUARTILE.INC, PERCENTILE.EXC, QUARTILE.EXC.

Notes

If the array argument includes a calculation, the function won't ignore hidden rows, nested subtotals, or nested aggregates.

AMORDEGRC

Description

Calculates the prorated linear asset depreciation for the French accounting system. The calculation is for each accounting period, based on the life of the asset. A depreciation coefficient is applied.

Syntax

```
AMORDEGRC(cost, date_purchased, first_period, salvage, period, rate, [ basis])
```

- cost: The cost of the asset.
- date_purchased: The purchase date of the asset.
- first_period: The date when the first period ends.
- salvage: The value of the asset at the end of its lifetime.
- period: The number of the period to calculate the depreciation for.
- rate: The rate of depreciation for the asset.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

AMORLINC

Description

Calculates the prorated linear depreciation or amortization value. This function is intended for use with the French accounting system.

Syntax

```
AMORLINC(cost, date_purchased, first_period, salvage, period, rate, [ basis])
```

- cost: The cost of the asset.
- date_purchased: The purchase date of the asset.
- first_period: The date when the first period ends.
- salvage: The value of the asset at the end of its lifetime.
- period: The number of the period to calculate the depreciation for.
- rate: The rate of depreciation for the asset.

- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

AND

Description

Tests the specified conditions. Returns True if all conditions are True; otherwise, returns False. Example: =AND(A5<B12,C4>D22).

Syntax

`AND(value1, value2,...)`

- value1: A condition to test.
- value2: A condition to test.

Example

Formula	Result
=AND(A1<A2,A3>A4) where A1 contains 5, A2 contains 10, A3 contains 15, and A4 contains 20.	FALSE

ARABIC

Description

Returns an arabic number for the specified Roman numeral string.

Syntax

`ARABIC(text)`

- text: The roman numeral to convert.

AREAS

Description

Returns the number of areas in the specified reference or list of references.

Syntax

`AREAS(reference)`

- reference: The reference (cell) or list of references (range of cells).

Example

Formula	Result
=AREAS((A1:B2,C3:D5))	2
=AREAS(1:1)	The number of the maximum column that has ever been used on the current sheet.

Related Functions

ARRAYAREA

GROUPBY

ARRAYAREA

Description

Returns the containing range of the array formula, based on the single cell address that you specify. The array can originate either in a Workday report or an array formula. This function helps you create formulas that operate on arrays where the cell range is unknown or changes over time.

Syntax

ARRAYAREA(`ref_cell`)

- ref_cell: The cell reference. Make sure you enter only a single cell address and not a range.

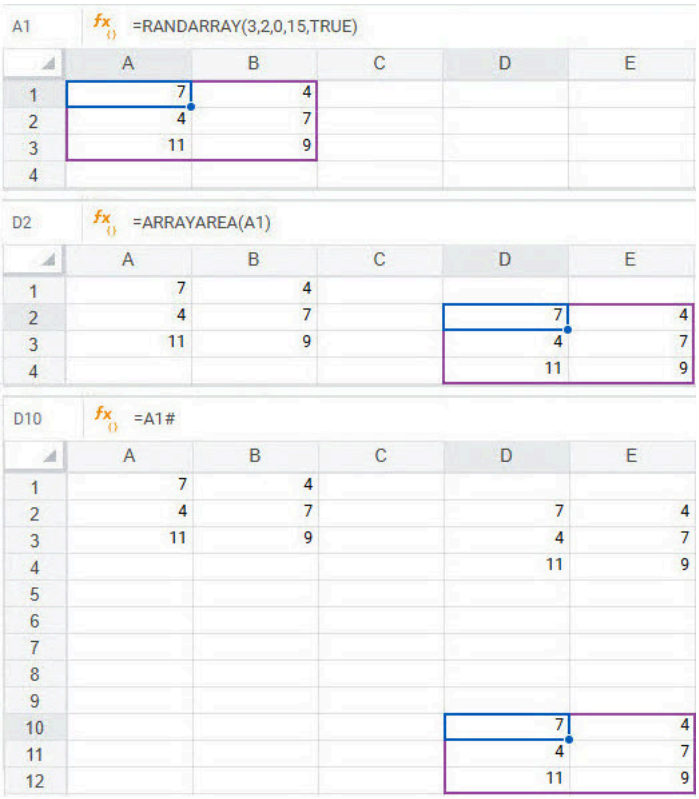
Example

The ARRAYAREA formula is dependent on the ref_cell being an array formula.

In this example, A1 contains the formula =RANDARRAY(3,2,0,15,TRUE) and A1:B3 is the array formula result set.

To return the results of the RANDARRAY formula, you can use the ARRAYAREA formula and reference A1, as in =ARRAYAREA(A1).

You can also get the same results as =ARRAYAREA(A1) by simply using the cell reference A1 with #, as in =A1#.



ARRAYTOTEXT

Description

Converts an array of values into a single text string.

This function is useful for displaying array contents in a readable text format or converting array outputs for text-based operations.

Syntax

```
ARRAYTOTEXT(array, [format])
```

- array: The array to return as text.
- format: The format of the returned data. Options include:
  - 0 (default): Concise format that is easy to read.
  - 1: Strict format that includes escape characters and row delimiters.

ASC

Description

ASC is provided for compatibility with Excel. It returns the input value. If you want to convert Unicode full-width or half-width text, use WD.FULLTOHALF or WD.HALFTOFULL.

Syntax

```
ASC(text)
```

- text: The text to return.

ASIN

Description

Returns the inverse sine of the specified value.

Syntax

```
ASIN(value)
```

- value: The value to calculate the inverse sine for.

ATAN

Description

Returns an angle, in radians, showing the inverse tangent of the specified value.

Syntax

```
ATAN(value)
```

- value: The value to calculate the inverse tangent for.

ATAN2

Description

Returns an angle, in radians, showing the inverse tangent calculated from x and y coordinates.

Syntax

```
ATAN2(x, y)
```

- x: The x coordinate of the point to calculate the inverse tangent for.
- y: The y coordinate of the point to calculate the inverse tangent for.

ATANH

Description

Returns an angle, in radians, showing the inverse hyperbolic tangent of the specified value.

Syntax

```
ATANH(value)
```

- value: The value between -1 and 1 to calculate the inverse hyperbolic tangent for.

AVEDEV

Description

Calculates the average of the absolute deviations of the values in one or more lists.

Syntax

```
AVEDEV(number1, [ number2], ...)
```

- number1: The number, or list of numbers.
- number2: The number, or list of numbers.

AVERAGE

Description

Returns the average of a list of numbers.

Syntax

```
AVERAGE(number1, [ number2], ...)
```

- number1: The number, or list of numbers.
- number2: The number, or list of numbers.

AVERAGEA

Description

Returns the average value from a list of values. Similar to AVERAGE(), but in AVERAGEA() you can include text and logical values. A text value or logical FALSE value is treated as a 0; a logical TRUE is counted as a 1.

Syntax

```
AVERAGEA(value1, [ value2], ...)
```

- value1: The value, or list of values.
- value2: The value, or list of values.

AVERAGEIF

Description

Returns the average of the values that meet the specified condition (criterion).

Syntax

```
AVERAGEIF(range, criteria, [ average_range])
```

- range: The values, or the range of cells, to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.
- average_range: The values, or the range of cells, to be averaged if the range meets the specified condition. If not specified, the range is averaged.

AVERAGEIFS

Description

Returns the average of the values that meet multiple conditions (criteria).

Syntax

```
AVERAGEIFS(average_range, range, criteria, [ range2, criteria2], ...)
```

- average_range: The values, or the range of cells, to be averaged if the specified ranges meet the specified conditions.
- range: The values, or the range of cells, to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.
- range2: The values, or the range of cells, to be evaluated according to the condition.
- criteria2: The condition to use to evaluate the values.

BASE

Description

Converts an integer to the specified base. Returns a string (text).

Syntax

```
BASE(value, base, [ min_length])
```

- value: The number to convert.
- base: The base for conversion. Supports values between 2 and 36.
- min_length: The minimum length of the resulting string. Leading zeros are added to the result if it is shorter than the minimum length.

Example

Formula	Result
=BASE(15,2)	1111
=BASE(15,2,8)	00001111
=BASE(123456,16)	1E240

Notes

- This function does the same action as BASETONUM().

BASETONUM

Description

Converts an integer to the specified base. Returns a string (text).

Syntax

```
BASETONUM(value, base, [ min_length])
```

- value: The number to convert.
- base: The base for conversion. Supports values between 2 and 36.
- min_length: The minimum length of the resulting string. Leading zeros are added to the result if it is shorter than the minimum length.

Example

Formula	Result
=BASETONUM(15,2)	1111
=BASETONUM(15,2,8)	00001111
=BASETONUM(123456,16)	1E240

Notes

- This function does the same action as BASE().

BESSELI

Description

Returns the Bessel function of integer order In(x).

Syntax

```
BESSELI(x, n)
```

- x: The value where the function is to be evaluated.
- n: The order of the function. If the specified value is a decimal number, it is truncated to an integer.

BESSELJ

Description

Returns the Bessel function of integer order Jn(x).

Syntax

`BESSELJ(x, n)`

- x: The value where the function is to be evaluated.
- n: The order of the function. If the specified value is a decimal number, it is truncated to an integer.

BESSELK

Description

Returns the modified Bessel function of integer order Kn(x).

Syntax

`BESSELK(x, n)`

- x: The value where the function is to be evaluated.
- n: The order of the function. If the specified value is a decimal number, it is truncated to an integer.

BESSELY

Description

Returns the Bessel function of integer order Yn(x).

Syntax

`BESSELY(x, n)`

- x: The value where the function is to be evaluated.
- n: The order of the function. If the specified value is a decimal number, it is truncated to an integer.

BIN2DEC

Description

Returns the decimal equivalent of the binary number.

Syntax

`BIN2DEC(number)`

- number: The number to convert.

BIN2HEX

Description

Returns the hexadecimal equivalent of the binary number.

Syntax

```
BIN2HEX(number, [ places])
```

- number: The number to convert.
- places: The number of digits to return, up to a limit of 10 digits. If the result is shorter than the specified number of digits, it is left-padded with zeroes.

BIN2OCT

Description

Returns the octal equivalent of the binary number, to the specified number of decimal places.

Syntax

```
BIN2OCT(number, [ places])
```

- number: The number to convert.
- places: The number of digits to return, up to a limit of 10 digits. If the result is shorter than the specified number of digits, it is left-padded with zeroes.

BINOM.DIST

Description

Returns the binomial distribution based on the number of trial successes, probability, and probability type.

Syntax

```
BINOM.DIST(number_s, trials, probability_s, cumulative)
```

- number_s: The number of successes to calculate the probability for.
- trials: The number of trials. If you enter a non-integer, it is truncated.
- probability_s: The probability of success for a trial.
- cumulative: The type of distribution to use. FALSE = probability mass function; TRUE = cumulative distribution function.

BINOM.INV

Description

Returns the smallest value where the cumulative binomial distribution is greater than or equal to the criterion value.

Syntax

```
BINOM.INV(trials, probability_s, alpha)
```

- trials: The number of (Bernoulli) trials. If you enter a non-integer, it is truncated.
- probability_s: The probability of success for a trial.
- alpha: The number between 0 and 1 for the criterion.

Example

Formula	Result
=BINOM.INV(6, 0.5, 0.75)	4

Notes

- This function does the same action as BINOMINV() and CRITBINOM().
- If any argument is nonnumeric, BINOM.INV() returns #VALUE! (invalid type).
- If trials is not an integer, it is truncated.
- If trials is less than zero, BINOM.INV() returns #NUM! (invalid number).
- If probability_s is less than zero or greater than 1, BINOM.INV() returns #NUM! (invalid number).
- If alpha is less than zero or greater than 1, BINOM.INV() returns #NUM! (invalid number).

BINOMDIST

Description

Returns the binomial distribution based on the number of trial successes, probability, and probability type.

Syntax

```
BINOMDIST(number_s, trials, probability_s, cumulative)
```

- number_s: The number of successes to calculate the probability for.
- trials: The number of trials. If you enter a non-integer, it is truncated.
- probability_s: The probability of success for a trial.
- cumulative: The type of distribution to use. FALSE = probability mass function; TRUE = cumulative distribution function.

BINOMINV

Description

Returns the smallest value where the cumulative binomial distribution is greater than or equal to the criterion value.

Syntax

```
BINOMINV(trials, probability_s, alpha)
```

- trials: The number of (Bernoulli) trials. If you enter a non-integer, it is truncated.
- probability_s: The probability of success for a trial.
- alpha: The number between 0 and 1 for the criterion.

Example

Formula	Result
=BINOMINV(6, 0.5, 0.75)	4

Notes

- This function does the same action as BINOM.INV() and CRITBINOM().
- If any argument is nonnumeric, BINOMINV() returns #VALUE! (invalid type).
- If trials is not an integer, it is truncated.
- If trials trials is less than zero, BINOMINV() returns #NUM! (invalid number).
- If probability_s is less than zero or greater than 1, BINOMINV() returns #NUM! (invalid number).
- If alpha is less than zero or greater than 1, BINOMINV() returns #NUM! (invalid number).

BITAND

Description

Performs a bitwise AND operation on the specified integer values.

Syntax

```
BITAND(number1, number2)
```

- number1: A positive integer.
- number2: A positive integer.

BITLSHIFT

Description

Returns the specified value after shifting left by the specified number of bit positions.

Syntax

```
BITLSHIFT(number, shift_amount)
```

- number: The number to do the left shift operation on. The value must be an integer.
- shift_amount: The number of bits to shift left. The value must be an integer.

BITOR

Description

Performs a bitwise OR operation on the specified integer values.

Syntax

```
BITOR(number1, number2)
```

- number1: A positive integer.
- number2: A positive integer.

BITRSHIFT

Description

Returns the specified value after shifting right by the specified number of bit positions.

Syntax

```
BITRSHIFT(number, shift_amount)
```

- number: The number to do the right shift operation on. The value must be an integer.
- shift_amount: The number of bits to shift right. The value must be an integer.

BITXOR

Description

Performs a bitwise exclusive OR operation on the specified integer values.

Syntax

```
BITXOR(number1, number2)
```

- number1: A positive integer.
- number2: A positive integer.

BONDDURATION

Description

Returns the duration of a fixed interest security in years. Maturity is based on the Macaulay duration. For the modified duration, use MDURATION() or BONDMDURATION().

Syntax

```
BONDDURATION(settlement, maturity, coupon, yield, frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- coupon: The security's annual coupon rate.
- yield: The security's annual yield.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

BONDMDURATION

Description

Returns the modified duration of a fixed interest security. For the Macaulay duration, use DURATION() or BONDDURATION().

Syntax

```
BONDMDURATION(settlement, maturity, coupon, yield, frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- coupon: The security's annual coupon rate.
- yield: The security's annual yield.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

BYCOL

Description

Applies a LAMBDA function to each column of an array and returns an array of the results.

This function is useful for performing column-wise calculations and aggregations on your data.

Syntax

```
BYCOL(array, lambda)
```

- array: The array to be separated by column.
- lambda: A LAMBDA that takes a column as a single parameter and calculates one result.
    - column: A column from the array.

BYROW

Description

Applies a LAMBDA function to each row of an array and returns an array of the results.

This function is useful for performing row-wise operations, such as calculating sums, averages, or custom metrics for each row.

Syntax

```
BYROW(array, lambda)
```

- array: The array to be separated by row.
- lambda: A LAMBDA that takes a row as a single parameter and calculates one result.
    - row: A row from the array.

CAPPEDVALUES

Description

Typically used for 401(k) deductions, ESPP deductions, or tax payments that have a regular value per period, but drop to zero (0) when the payment reaches the cap. Returns an array of values over a set of periods from an array of values that you provide, over the same periods, limited by a provided cap over the whole duration. The function returns the full values for all periods until it reaches the one where the cap you specified is reached; for that period it returns a partial value, and if there are any periods left, the function returns 0 for those.

Syntax

```
CAPPEDVALUES(values, cap, [ periods])
```

- values: The values array. The array must be either 1 row or 1 column.
- cap: The cap value.
- periods: The number of periods, such as quarters or months, in the array. If you omit the periods argument, the function uses the number of values that you specified to determine the number of periods.

Example

This example shows how to populate the 401(k) withholding values in rows 6 & 8, based on the values in rows 2, 5, and 7. The function uses salaries and 401(k) withholding amounts for Peter and Aileen, based on a 401(k) cap of $18,000 and a 401(k) contribution percentage of 17%.

The formula in row 6 (for Peter) is =CAPPEDVALUES(F2*C5:N5,$B$2). The values argument value provides the information that the formula uses to determine the number of periods.

The formula in row 8 (for Aileen) is =CAPPEDVALUES(F2*C7,$B$2,12).

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1															

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
2	401K cap	$18,000.00			Contribution %										
3															
4			Jan	Feb	Mar	Apr	May	June	July	Aug	Sept	Oct	Nov	Dec	Total
5	Peter	Salary	11,054.00	11,054.00	11,054.00	11,054.00	11,054.00	11,054.00	11,054.00	11,054.00	11,054.00	11,054.00	11,054.00	11,054.00	$132,648
6	Peter	401K	1879.18	1879.18	1879.18	1879.18	1879.18	1879.18	1879.18	1879.18	1879.18	1087.38	0	0	$18,000
7	Aileen	Salary	$12,487.00	$12,487.00	$12,487.00	$12,487.00	$12,487.00	$12,487.00	$12,487.00	$12,487.00	$12,487.00	$12,487.00	$12,487.00	$12,487.00	$149,844
8	Aileen	401K	2122.79	2122.79	2122.79	2122.79	2122.79	2122.79	2122.79	2122.79	1017.63	0	0	0	$18,000

CBRT

Description

Returns the cube root of a number.

Syntax

`CBRT(value)`

- value: The value to compute the cube root for.

Example

Formula	Result
=CBRT(8)	2

CEILING

Description

Rounds a number up to the nearest integer, based on the significance and mode that you specify.

Syntax

`CEILING(number, [ significance], [ mode])`

- number: The number to round up.
- significance: The multiple of significance to round the number up to. If you omit the argument, the function uses the value 1, which causes the function to round the number up to the next larger integer.
- mode: If you specify -1, the function rounds negative numbers down, becoming more negative. If you specify any other number, the function rounds negative numbers up, toward zero.

Example

Formula	Result
=CEILING(12.34)	13
=CEILING(12.34,0.1)	12.4

Formula	Result
=CEILING(12.34,0.5)	12.5
=CEILING(12.34,10)	20
=CEILING(-12.34,-1)	-13
=CEILING(-12.34,-0.1)	-12.4

Notes

- number and significance must have the same sign.
- This function does the same action as CEILING.MATH().

CEILING.MATH

Description

Rounds a number up to the nearest integer, based on the significance and mode that you specify.

Syntax

```
CEILING.MATH(value, [ significance], [ mode])
```

- value: The number to round up.
- significance: The multiple of significance to round the number up to. If you omit the argument, the function uses the value 1 for positive numbers and -1 for negative numbers.
- mode: Applicable only to negative numbers. If you specify -1, the function rounds negative numbers down, becoming more negative. If you omit the argument or specify any other number, the function rounds negative numbers up, toward zero.

Example

Formula	Result
=CEILING.MATH(12.34)	13
=CEILING.MATH(12.34,0.1)	12.4
=CEILING.MATH(12.34,0.5)	12.5
=CEILING.MATH(12.34,10)	20
=CEILING.MATH(-12.34)	-12
=CEILING.MATH(-12.34,-1)	-12
=CEILING.MATH(-12.34,-0.1)	-12.3

CEILING.PRECISE

Description

Rounds a number up to a multiple of significance. The function doesn't consider the number's sign. The function rounds positive numbers up, and also rounds negative numbers up (toward zero, becoming less negative).

Syntax

```
CEILING.PRECISE(value, [ significance])
```

- value: The number to round up.
- significance: The multiple of significance to round the number up to. If you omit the argument, the function uses the value 1, which causes the function to round the number up to the next larger integer.

Example

Formula	Result
=CEILING.PRECISE(12.34)	13
=CEILING.PRECISE(12.34,1)	13
=CEILING.PRECISE(-12.34,1)	-12
=CEILING.PRECISE(12.34,0.1)	12.4

Notes

- This function does the same action as ISO.CEILING() and ISOCEILING().

CELL

Description

Returns information about the address, column, row, type, width, or height of a cell, or general workbook information. If you omit the referenceargument, the function is volatile. We recommend always specifying the referenceargument.

Syntax

```
CELL(info_type, [ reference])
```

- info_type: The type of cell information you want to obtain. Possible values are:
  - address: The reference of the first cell in reference, as text.
  - col: The column number of the cell in the reference.
  - row: The row number of the cell in the reference.
  - type: A text value corresponding to the type of data in the cell. The function returns b for blank if the cell is empty, l for label if the cell contains a text constant, or v for value if the cell isn't in either of the other categories.
  - width: The column width of the cell. The function returns an array with 2 items: first the column width of the cell, rounded off to an integer; second a Boolean value of TRUE if the column width is the default or FALSE if the width was set by the user. Each unit of width is equal to the width of one character in the default font size.
  - height: The row height of the cell. The function returns an array with 2 items: first the row height of the cell, rounded off to an integer; second a Boolean value of TRUE if the height is the default or FALSE if the height was set by the user. Each unit of height is equal to the height of one character in the default font size.
  - filename: The name of the workbook.
  - wd.sheet.count: The count of sheets in the workbook.
  - wd.sheet.name: The name of the sheet containing the formula.
  - wd.sheet.names: The names of all sheets in the workbook.
  - wd.workbook.id: The workbook ID.
  - wd.workbook.name: The workbook name.
- reference: The cell that you want information about. If you omit the argument, the function returns the information specified in the info_type argument for the last cell that was changed; in Worksheets,

a last-changed cell is one that was changed by setting values, setting formulas, clearing values, or clearing formulas. If you specify a range as the reference argument, the function returns the information for only the upper left cell of the range.

CHAR

Description

Returns the character value for the specified character set number. The function uses UTF-16 encoding, which allows for 2-byte characters.

Syntax

`CHAR(number)`

- number: The character set number representing a character.

Example

Formula	Result
=CHAR(A1) Where cell A1 contains `101` .	e

CHIDIST

Description

Returns the (left-tail) chi square distribution. Depending on the cumulative argument, calculates either the probability density function value or the cumulative distribution function value.

Syntax

`CHIDIST(x, deg_freedom, cumulative)`

- x: The value to use when evaluating the distribution.
- deg_freedom: The number of degrees of freedom.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

Example

Formula	Result
=CHIDIST(0.5, 1, TRUE)	0.520499878
=CHIDIST(2, 3, FALSE)	0.207553749

Notes

- This function does the same action as CHISQ.DIST() and CHISQDIST().

CHISQ.DIST

Description

Returns the (left-tail) chi square distribution. Depending on the cumulative argument, calculates either the probability density function value or the cumulative distribution function value.

Syntax

```
CHISQ.DIST(x, deg_freedom, cumulative)
```

- x: The value to use when evaluating the distribution.
- deg_freedom: The number of degrees of freedom.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

Example

Formula	Result
=CHISQ.DIST(0.5, 1, TRUE)	0.520499878
=CHISQ.DIST(2, 3, FALSE)	0.207553749

Notes

- This function does the same action as CHISQDIST() and CHIDIST().

CHISQ.DIST.RT

Description

Returns the right-tail probability of the chi-square distribution.

Syntax

```
CHISQ.DIST.RT(x, deg_freedom)
```

- x: The value to use when evaluating the distribution.
- deg_freedom: The number of degrees of freedom.

Example

Formula	Result
=CHISQ.DIST.RT(18.307, 10)	0.050000589

Notes

- This function does the same action as CHISQDISTRT().

CHISQDIST

Description

Returns the (left-tail) chi square distribution. Depending on the cumulative argument, calculates either the probability density function value or the cumulative distribution function value.

Syntax

```
CHISQDIST(x, deg_freedom, cumulative)
```

- x: The value to use when evaluating the distribution.
- deg_freedom: The number of degrees of freedom.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

Example

Formula	Result
=CHISQDIST(0.5, 1, TRUE)	0.520499878
=CHISQDIST(2, 3, FALSE)	0.207553749

Notes

- This function does the same action as CHISQ.DIST() and CHIDIST().

CHISQDISTRT

Description

Returns the right-tail probability of the chi-square distribution.

Syntax

```
CHISQDISTRT(x, deg_freedom)
```

- x: The value to use when evaluating the distribution.
- deg_freedom: The number of degrees of freedom.

Example

Formula	Result
=CHISQDISTRT(18.307, 10)	0.050000589

Notes

- This function does the same action as CHISQ.DIST.RT().

CHOOSE

Description

Returns a value from a list of values, based on a specified index number.

Syntax

```
CHOOSE(index_num, value1, ...)
```

- index_num: The index of the item to return. If index_num = 1, the function returns the first value. You can use a cell reference as the index value.
- value1: A list of values, or references to cells containing values. The values can be formula expressions.

Example

Formula	Result
=CHOOSE(A1,B1,B2,B3,B4) where A1 contains 3 and B3 contains 30.	30

CHOOSECOLS

Description

Returns the specified columns from an array.

Syntax

`CHOOSECOLS(array,col_num1,[col_num2],…)`

- array: The array containing the columns to be returned.
- col_num1: The first column number to be returned.
- col_num2: One or more additional column numbers to be returned.

CHOOSEROWS

Description

Returns the specified rows from an array.

Syntax

`CHOOSEROWS(array,row_num1,[row_num2],…)`

- array: The array containing the rows to be returned.
- row_num1: The first row number to be returned.
- row_num2: One or more additional row numbers to be returned.

CLEAN

Description

Removes all non-printable characters from the specified text string and returns the result. Non-printable characters are the ASCII numeric codes from 0 to 31. CLEAN is similar to TRIM, but TRIM also removes the space character: ASCII numeric code 32.

Syntax

`CLEAN(text)`

- text: The text to remove characters from.

CLUSTER.KMEANS

Description

Clustering is a data mining technique for grouping a set of objects into smaller groups, where each group's members are similar to the other members in some aspect. Clustering is used in machine learning, for example, where the goal is to find meaningful structures or to explain processes. You can cluster quantitatively using numbers, or qualitatively using categories. CLUSTER.KMEANS is a quantitative function that enables you to cluster a range of data in a workbook, by computing the distances between

points, and grouping centers. The function groups each data point with the cluster having the nearest center.

Syntax

```
CLUSTER.KMEANS(range1, k, [ maxIterations], [ distanceMeasure], [
emptyStrategy])
```

- range1: The range of cells to analyze.
- k: The number of clusters to make.
- maxIterations: The maximum number of times to re-run the algorithm. If you don't specify a value, the limit is 100.
- distanceMeasure: The algorithm used to find the distance between points. Possible values are CanberraDistance, ChebyshevDistance, EarthMoversDistance, EuclideanDistance, or ManhattanDistance. If you don't specify a value, the function uses EuclideanDistance.
- emptyStrategy: The strategy to use if the function finds empty clusters while running the algorithm iterations. Possible values are ERROR, FARTHEST_POINT, LARGEST_POINTS_NUMBER, or LARGEST_VARIANCE. If you don't specify a value, the function uses LARGEST_VARIANCE. If you specify ERROR, the function returns #ERROR in the cell with a description. Example: CLUSTER.KMEANS: Empty cluster in k-means.

CLUSTER.KMEANS.CENTROIDS

Description

Clustering is a data mining technique for grouping a set of objects into smaller groups, where each group's members are similar to the other members in some aspect. Clustering is used in machine learning, for example, where the goal is to find meaningful structures or to explain processes. You can cluster quantitatively using numbers, or qualitatively using categories. CLUSTER.KMEANS.CENTROIDS is a quantitative function that enables you to cluster a range of data in a workbook, by locating the center point of each group and computing the distances between points and group centers. The function groups each data point with the cluster having the nearest center.

Syntax

```
CLUSTER.KMEANS.CENTROIDS(range1, k, [ maxIterations], [ distanceMeasure], [
emptyStrategy])
```

- range1: The range of cells to analyze.
- k: The number of clusters to make.
- maxIterations: The maximum number of times to re-run the algorithm. If you don't specify a value, the limit is 100.
- distanceMeasure: The algorithm used to find the distance between points. Possible values are CanberraDistance, ChebyshevDistance, EarthMoversDistance, EuclideanDistance, or ManhattanDistance. If you don't specify a value, the function uses EuclideanDistance.
- emptyStrategy: The strategy to use if the function finds empty clusters while running the algorithm iterations. Possible values are ERROR, FARTHEST_POINT, LARGEST_POINTS_NUMBER, or LARGEST_VARIANCE. If you don't specify a value, the function uses LARGEST_VARIANCE. If you specify ERROR, the function returns #ERROR in the cell with a description. Example: CLUSTER.KMEANS: Empty cluster in k-means.

CODE

Description

Returns the decimal UTF-16 code of the first character in the specified text.

Syntax

`CODE(text)`

- text: The text to evaluate.

Example

Formula	Result
=CODE(A1) Where cell A1 contains `Workday` .	87

COLUMN

Description

Returns the column number of the reference that you specify.

Syntax

`COLUMN([ reference])`

- reference: The reference to return the column number for. If not specified, COLUMN() returns the column number of the cell where you place the function.

Example

Formula	Result
=COLUMN() where the current cell is B2.	2
=COLUMN(J2:P2)	10

COLUMNS

Description

Returns the number of columns in a range that you specify.

Syntax

`COLUMNS(array)`

- array: The range to return the number of columns for.

Example

Formula	Result
=COLUMNS(A1:J5)	10
=COLUMNS(1:1)	The number of the maximum column that has ever been used on the current sheet.

COMBIN

Description

Returns an integer representing the number of distinct combinations (subset count = n) for a specified number of objects (k).

Syntax

`COMBIN(n, k)`

- n: The number of objects.
- k: The number of selected objects.

COMBINA

Description

Returns an integer representing the number of combinations for a specified number of objects (k), where repetition is allowed.

Syntax

`COMBINA(n, k)`

- n: The number of objects.
- k: The number of selected objects.

COMPARE

Description

Compares two matrices, returning an integer representing the comparative relationship of two values. If the values match, the function returns null. Optionally, you can set a difference percentage so that if two values are within a certain percentage of each other, the function considers them to be the same.

Syntax

`COMPARE(value1, value2, [ mode], [ percentage])`

- value1: The first value to compare.
- value2: The second value to compare.
- mode: Specifies the result to return. If `difference`, then return the difference. (If value2 is greater than value1, the return value is positive; otherwise it's negative.) If `changed`, then the function returns value2. The default is `changed`.
- percentage: Specifies a percentage allowance of difference where the function should consider the two values to be the same. The two values are equal if they are within value1*percentage of each other.

Example

Formula	Result
=COMPARE(B2,B3,"changed",0.01) where B2 contains 15 and B3 contains 18.	18

Related Functions

ARRAYAREA

MINUS

COMPLEX

Description

Returns a complex number based on the specified real number and imaginary number.

Syntax

```
COMPLEX(real_num, i_num, [ suffix])
```

- real_num: The real coefficient.
- i_num: The imaginary coefficient.
- suffix: Either a lowercase i or j in quotes to specify the suffix to use.

CONCAT

Description

Concatenates the string arguments, in order, into 1 string.

Syntax

```
CONCAT(text, ...)
```

- text: The strings to concatenate.

Example

Formula	Result
=CONCAT(A1," ",B1) Where cell A1 contains John and cell B1 contains Davis .	John Davis

Notes

- This function does the same action as CONCATENATE().

CONCATENATE

Description

Concatenates the string arguments, in order, into 1 string.

Syntax

```
CONCATENATE(text, ...)
```

- text: The strings to concatenate.

Example

Formula	Result
=CONCAT(A1," ",B1) Where cell A1 contains John and cell B1 contains Davis .	John Davis

Notes

- This function does the same action as CONCAT().

CONFIDENCE

Description

Returns the confidence interval for a population mean.

Syntax

```
CONFIDENCE(alpha, stdev, size)
```

- alpha: The significance level.
- stdev: The standard deviation for the population.
- size: The sample size of the population.

CONFIDENCE.NORM

Description

Returns the confidence interval for a population mean, using a normal distribution.

Syntax

```
CONFIDENCE.NORM(alpha, stdev, size)
```

- alpha: The significance level.
- stdev: The standard deviation for the population.
- size: The sample size of the population.

CONFIDENCE.T

Description

Returns the confidence interval for a population mean, using a student t-distribution.

Syntax

```
CONFIDENCE.T(alpha, stdev, size)
```

- alpha: The significance level.
- stdev: The standard deviation for the population.
- size: The sample size of the population.

CONVERT

Description

Converts a number from 1 unit of measurement to another.

Syntax

`CONVERT(number, from_unit, to_unit)`

- number: The number to convert.
- from_unit: The current units value for the number. The value is case-sensitive.
- to_unit: The units to convert the number to. The value is case-sensitive.

Notes

- Valid units values are from the base library javax.measure.unit.Dimension.
- This function is similar to SETUNITS(), but SETUNITS() expects the number to have an existing units value.

CONVERTTZ

Description

Converts a datetime value from one time zone to another.

Syntax

`CONVERTTZ(range, from_time_zone_id, to_time_zone_id)`

- range: The cell, or range of cells, to return the converted time zone for.
- from_time_zone_id: The time zone specifier to convert from. Examples: GMT, US/Central.
- to_time_zone_id: The time zone specifier to convert to.

Example

Formula	Result
=CONVERTTZ(A8,"America/Montreal","US/Pacific") where A8 contains 10/4/2016 5:37 PM	10/4/2016 2:37 PM

Notes

- Time zone IDs are based on the Java TimeZone utility. A complete list of time zones is available at http://joda-time.sourceforge.net/timezones.html. Examples:
  - America/New_York
  - Asia/Hong_Kong
  - Australia/Melbourne
  - Canada/Eastern
  - Europe/London
  - GMT
  - MET
  - US/Mountain
  - UTC

CORREL

Description

Returns the correlation coefficient, based on an array of independent variables and an array of dependent variables.

Syntax

```
CORREL(array1, array2)
```

- array1: The array of independent variables.
- array2: The array of dependent variables.

CORRELATE

Description

Creates a new matrix by combining rows from the ranges you specify. This function is similar to a database join.

Syntax

```
CORRELATE(primary_table, key_column, ignore_case, [ join_range], [
join_key_column], ...)
```

- primary_table: The matrix whose rows are used as the origin. The function combines rows from this range with rows from the other ranges if their key values match.
- key_column: The column number in the primary-range whose value is used as the primary key value for comparison with other ranges.
- ignore_case: If the type of the key column is text, then if ignore_case is TRUE, the function ignores case when comparing with other key values. If FALSE, the function considers case in the comparisons.
- join_range: The first range to combine with a row from the primary range if the key column values are equal.
- join_key_column: The column number in the first join range whose value is used as the key value for comparison with the primary key.

Example

This is the original spreadsheet, with uncombined data in two different sections:

	A	B	C	D
25	Salesperson	Customer	Product	Revenue
26	Jeffrey	Aberdeen Asset Management	HCM	$1,000,000
27	Jeffrey	Admiral Group	FIN	$500,000
28	Lei	Babcock International	HCM	$506,000
29	Lei	Barclays	LER	$1,500,000
30	Cian	Capita	LER	$1,895,000
31	Cian	Centrica	PAY	$2,004,560

	G	H	I
25	Salesperson	Manager	Region
26	Jeffrey	Peter	EMEA
27	Lei	Fredrik	US
28	Cian	John	UK

The formula =CORRELATE(A25:D31,1,TRUE,G25:I28,1), placed in cell G37, combines the employee information with the manager and region information.

	G	H	I	J	K	L
37	Salesperson	Customer	Product	Revenue	Manager	Region
38	Jeffrey	Aberdeen Asset Management	HCM	$1,000,000	Peter	EMEA
39	Jeffrey	Admiral Group	FIN	$500,000	Peter	EMEA
40	Lei	Babcock International	HCM	$506,000	Fredrik	US
41	Lei	Barclays	LER	$1,500,000	Fredrik	US
42	Cian	Capita	LER	$1,895,000	John	UK
43	Cian	Centrica	PAY	$2,004,560	John	UK

Notes

- The CORRELATE() function requires at least one pair of join_range and join_key_column values. To use other ranges, add each one with an argument specifying which column within that range has the role of the key for that range.
- The function works like this:
    1. The function examines each row in primary_table.
    2. For each range you specified, the function examines each row in the range. The function compares the value you specified as the primary key (based on the primary key column index) from the primary_table with the value you specified as the key within the row in the range currently being examined.
    3. If their values match, then the function merges all the values from the row in that range (except the key column) onto the end of the row from the primary table.
- A row from the primary table appears in the output only if there was at least one correlated row from at least one of the other match ranges you specified. If you want all rows from the primary to appear, regardless, use MERGEROWS. If you want a true database inner join, use JOIN.
- When comparing a single instance value to a string that visually appears to be the same, the function evaluates them as equal. When comparing a single value in a multi-instance field to a string that visually appears to be the same, the formula evaluates them as *not* equal.
- This function is intended for use in array formulas.

Related Functions

JOIN

COS

Description

Returns the cosine of the specified value.

Syntax

`COS(value)`

- value: The value to calculate the cosine for.

Related Functions

CBRT

HYPOT

COSH

Description

Returns the hyperbolic cosine of the specified value.

Syntax

`COSH(value)`

- value: The value to calculate the hyperbolic cosine for.

COT

Description

Returns the cotangent of the specified angle.

Syntax

`COT(number)`

- number: The angle to calculate the cotangent for.

COTH

Description

Returns the hyperbolic cotangent of the specified value.

Syntax

`COTH(value)`

- value: The value to calculate the hyperbolic cotangent for.

COUNT

Description

Returns the number of values provided. Only numbers and string representations of numbers are counted. To count other data types, use COUNTA.

Syntax

```
COUNT(value1, [ value2], ...)
```

- value1: The value or list of values.
- value2: The value or list of values.

COUNTA

Description

Returns the number of specified values. All data types are counted. To count only numbers and string representations of numbers, use COUNT.

Syntax

```
COUNTA(value1, [ value2], ...)
```

- value1: The value or list of values.
- value2: The value or list of values.

COUNTBLANK

Description

Returns the number of blank cells in the specified range of cells.

Syntax

```
COUNTBLANK(value)
```

- value: A range of cells.

COUNTIF

Description

Returns the number of specified values that meet the condition (criterion). You can use the * wildcard to match string values.

Syntax

```
COUNTIF(range, criteria)
```

- range: The values, or a range of cells, to be counted if the condition is met.
- criteria: The condition to use to evaluate the values.

COUNTIFS

Description

Returns the number of times that the cells in the specified ranges meet multiple conditions (criteria). You can use the * wildcard to match string values.

Syntax

```
COUNTIFS(range, criteria, [ range2, criteria2], ...)
```

- range: The value, or list of values, to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.

- range2: The value, or list of values, to be evaluated according to the condition.
- criteria2: The condition to use to evaluate the values.

COUPDAYBS

Description

Returns the number of days from the start of a coupon period to the settlement date.

Syntax

```
COUPDAYBS(settlement, maturity, frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

COUPDAYS

Description

Returns the number of days in a coupon period that contains the settlement date.

Syntax

```
COUPDAYS(settlement, maturity, frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

COUPDAYSNC

Description

Returns the number of days between a settlement date and the next coupon date.

Syntax

```
COUPDAYSNC(settlement, maturity, frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

COUPNCD

Description

Returns the next coupon date after the settlement date.

Syntax

```
COUPNCD(settlement, maturity, frequency, [ basis])
```

- settlement: The securitys settlement date.
- maturity: The securitys maturity date.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

COUPNUM

Description

Returns the number of coupons to be paid, between the settlement date and the maturity date.

Syntax

```
COUPNUM(settlement, maturity, frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

COVARIANCE.P

Description

Returns the population covariance: the average of the products of deviations for each data point pair in two data sets.

Syntax

```
COVARIANCE.P(array1, array2)
```

- array1: The first cell range of integers.
- array2: The second cell range of integers.

Example

Formula	Result
=COVARIANCEP({3;2;4;5;6}, {9;7;12;15;17})	5.2

Notes

- This function does the same action as COVARIANCEP().

COVARIANCE.S

Description

Returns the sample covariance: the average of the products of deviations for each data point pair in two data sets.

Syntax

`COVARIANCE.S(array1, array2)`

- array1: The first cell range of integers.
- array2: The second cell range of integers.

Example

Formula	Result
COVARIANCE.S({2,4,8}, {5,11,12})	9.666666667

Notes

- This function does the same action as COVARIANCES().

COVARIANCEP

Description

Returns the population covariance: the average of the products of deviations for each data point pair in two data sets.

Syntax

`COVARIANCEP(array1, array2)`

- array1: The first cell range of integers.
- array2: The second cell range of integers.

Example

Formula	Result
=COVARIANCEP({3;2;4;5;6}, {9;7;12;15;17})	5.2

Notes

- This function does the same action as COVARIANCE.P().

COVARIANCES

Description

Returns the sample covariance: the average of the products of deviations for each data point pair in two data sets.

Syntax

`COVARIANCES(array1, array2)`

- array1: The first cell range of integers.
- array2: The second cell range of integers.

Example

Formula	Result
COVARIANCES({2,4,8}, {5,11,12})	9.666666667

Notes

- This function does the same action as COVARIANCE.S().

CRITBINOM

Description

Returns the smallest value where the cumulative binomial distribution is greater than or equal to the criterion value.

Syntax

```
CRITBINOM(trials, probability_s, alpha)
```

- trials: The number of trials. If you enter a non-integer, it is truncated.
- probability_s: The number between 0 and 1 for the probability of success in one trial.
- alpha: The number between 0 and 1 for the criterion.

Example

Formula	Result
=CRITBINOM(6, 0.5, 0.75)	4

Notes

- This function does the same action as BINOM.INV() and BINOMINV().
- If any argument is nonnumeric, CRITBINOM() returns #VALUE! (invalid type).
- If trials is not an integer, it is truncated.
- If trials is less than zero, CRITBINOM() returns #NUM! (invalid number).
- If probability_s is less than zero or greater than 1, CRITBINOM() returns #NUM! (invalid number).
- If alpha is less than zero or greater than 1, CRITBINOM() returns #NUM! (invalid number).

CSC

Description

Returns the cosecant of the specified angle.

Syntax

```
CSC(number)
```

- number: The angle to calculate the cosecant for.

CSCH

Description

Returns the hyperbolic cosecant of the specified angle.

Syntax

```
CSCH(number)
```

- number: The angle to calculate the hyperbolic cosecant for.

CUMIPMT

Description

Returns the cumulative interest paid between the two specified periods.

Syntax

```
CUMIPMT(rate, nper, pv, start_period, end_period, type)
```

- rate: The interest rate.
- nper: The total number of payment periods.
- pv: The present value.
- start_period: The first period.
- end_period: The last period.
- type: The payment timing. 0 = the payment occurs at the end of the period; 1 = the payment occurs at the beginning of the period.

CUMPRINC

Description

Returns the cumulative payment on the principal between the two specified periods.

Syntax

```
CUMPRINC(rate, nper, pv, start_period, end_period, type)
```

- rate: The interest rate.
- nper: The total number of payment periods.
- pv: The present value.
- start_period: The first period.
- end_period: The last period.
- type: The payment timing. 0 = the payment occurs at the end of the period; 1 = the payment occurs at the beginning of the period.

DATE

Description

Returns the date based on the specified year, month, and day.

Syntax

```
DATE(year, month, day)
```

- year: The integer representing the year.
- month: The integer representing the month.
- day: The integer representing the day.

Example

Formula	Result
=DATE(2016,7,15)	"7/15/2016"

Notes

- If the result is a numeric value instead of a date, check the formatting for the cell to make sure it's set to Date .

Related Functions

DATESBETWEEN

DATEDIF

Description

Returns the number of days, months, or years between two dates, based on the specified unit. We provide DATEDIF for compatibility but it calculates incorrect results in some situations. We strongly recommend using WD.DATEDIF for more accurate results.

Syntax

```
DATEDIF(start_date, end_date, unit)
```

- start_date: The starting date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- end_date: The ending date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- unit: The 1- or 2-character code specifying the information to include in the result. The value is case-sensitive.

Code	Returns
Y	The number of complete years in the period.
M	The number of complete months in the period.
D	The number of days in the period.
MD	The difference between the days in start_date and end_date. The function ignores the months and years of the dates.
YM	The difference between the months in start_date and end_date. The function ignores the days and years of the dates.
YD	The difference between the days of start_date and end_date. The function ignores the years of the dates.

Example

Formula	Result
=DATEDIF("01-Jan-2015", "10-Jan-2015", "D")	9
=DATEDIF("14-Jul-2016", "01-Dec-2016", "YM")	5

DATESBETWEEN

Description

Returns an array of dates that starts and ends on a particular date, with a step interval between each date.

Syntax

```
DATESBETWEEN(start_date, end_date, [ step])
```

- start_date: The starting date of the range.
- end_date: The ending date of the range.
- step: The number of days to step between each date. The default is 1.

Example

Formula	Result
=DATESBETWEEN("01-Jan-2015", "10-Jan-2015", 7)	{ "1/1/2015"; "1/8/2015" }
=DATESBETWEEN("14-Jul-2016", "01-Jul-2016", 2)	{ "7/14/2016"; "7/12/2016"; "7/10/2016"; "7/8/2016"; "7/6/2016"; "7/4/2016"; "7/2/2016" }

Notes

- This function returns a column of dates starting at the start_date and ending either before or at the end_date.
- The step parameter must be a positive integer when present.
- Since the step parameter may be such that the end_date is never reached exactly, the final date Workday returns in the range is the greatest date that is less than the end_date, but where adding *step* days to that date would make the new date greater than the end_date.
- This function is intended for use in array formulas.

DATESFROM

Description

Returns an array of dates that starts on a particular date and continues for the number of dates you specify, with a step interval between each date.

Syntax

```
DATESFROM(start_date, num_dates, [ step])
```

- start_date: The starting date of the range.
- num_dates: The number of dates to be returned.
- step: The number of days to step between each date. The default is 1.

Example

Formula	Result
=DATESFROM("01-Jan-2015", 4, 2)	{ "1/1/2015"; "1/3/2015"; "1/5/2015"; "1/7/2015" }
=DATESFROM("14-Jul-2016", 3, -1)	{ "7/14/2016"; "7/13/2016"; "7/12/2016" }

Notes

- This function returns a column of num_dates dates.
- This function is intended for use in array formulas.

DATETIME

Description

Returns the date and time based on the year, month, day, hour, minute, and second that you specify.

Syntax

```
DATETIME(year, month, day, hour, minute, second)
```

- year: The integer representing the year.
- month: The integer representing the month.
- day: The integer representing the day.
- hour: The integer representing the hour.
- minute: The integer representing the minutes.
- second: The integer representing the seconds.

Example

Formula	Result
=DATETIME(2016,7,15,8,12,12)	"7/15/2016 8:12 AM"

Notes

- If you specify an argument that is outside the 24-hour clock, or you specify a negative number in an argument, Workday adjusts the resulting time to be valid. Example: =DATETIME(2016,7,15,-44,12,12) returns the value 7/13/2016 4:12 AM.
- If the result is a numeric value instead of a date, check the formatting for the cell to make sure it's set to Date .

DATEVALUE

Description

Returns a numerical (serial) date value based on the specified text.

Syntax

```
DATEVALUE(date_text)
```

- date_text: The text to convert to a numerical date value. Example: =DATEVALUE("06/06/16").

Example

Formula	Result
=DATEVALUE("12/1/2018")	43435
=DATEVALUE("Dec 1, 2018")	43435
=DATEVALUE("1-Dec-2018")	43435
=DATEVALUE("Dec 1")	43435

Related Functions

DATESBETWEEN

DAY

Description

Returns an integer from 0 to 31 representing the day portion of the date that you specify.

Syntax

`DAY(serial_number)`

- serial_number: The date.

Example

Formula	Result
=DAY("7/14/2016")	14
=DAY(DATE(2016,12,31))	31

DAYNAME

Description

Returns the name of the day, based on the day number that you specify.

Syntax

`DAYNAME(day_num)`

- day_num: The number of the day of the week, where 1 = Sunday, 2 = Monday, and so on.

Example

Formula	Result
=DAYNAME(1)	Sunday

DAYS

Description

Returns an integer representing the number of days between two specified dates.

Syntax

```
DAYS(end_date, [ start_date])
```

- end_date: The ending date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- start_date: The starting date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.

Example

Formula	Result
=DAYS("01-Jan-2015", "10-Jan-2015")	-9
=DAYS("01-Dec-2016","14-Jul-2016")	140

DAYS360

Description

Returns an integer representing the number of days between two specified dates, based on a 360-day year.

Syntax

```
DAYS360(start_date, end_date, [ method])
```

- start_date: The starting date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- end_date: The ending date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- method: The method to use in the count. False or omitted = US (NASD) method; True = European method.
    - In the US method, if start_date is last day of a month, Workday sets it to the 30th of that month. If end_date is last day of month, then if start_date is the last day of month, Workday sets end_date to the 1st of the following month; otherwise, Workday sets end_date to the 30th of that month.
    - In the European method, if start_date is the last day of the month, Workday sets it to the 30th of that month. If end_date is the last day of the month, Workday sets it to the 30th of that month.

Example

Formula	Result
=DAYS360("01-Jan-2016", "1-Jan-2017")	360
=DAYS360("01-Dec-2016","14-Jul-2016")	-137

DB

Description

Returns the depreciation allowance of an asset, based on the fixed declining balance method.

Syntax

```
DB(cost, salvage, life, period, [ month])
```

- cost: The initial cost of the asset.
- salvage: The value of the asset after depreciation.
- life: The number of periods to depreciate the asset over.
- period: The number of the period to calculate the depreciation allowance for.
- month: The number of months of the year to use in calculating the first period of depreciation.

DDB

Description

Returns the depreciation of an asset, based on the double declining balance method or a different specified method.

Syntax

```
DDB(cost, salvage, life, period, [ factor])
```

- cost: The initial cost of the asset.
- salvage: The value of the asset after depreciation.
- life: The number of periods to depreciate the asset over.
- period: The number of the period to calculate the depreciation for.
- factor: The rate of decline for the balance. The default is 2.

DEC2BIN

Description

Returns the binary equivalent of the number.

Syntax

```
DEC2BIN(number, [ places])
```

- number: The number to convert.
- places: The number of digits to return, up to a limit of 10 digits. If the result is shorter than the specified number of digits, it is left-padded with zeroes.

DEC2HEX

Description

Returns the hexadecimal equivalent of the number.

Syntax

```
DEC2HEX(number, [ places])
```

- number: The number to convert.
- places: The number of digits to return, up to a limit of 10 digits. If the result is shorter than the specified number of digits, it is left-padded with zeroes.

DEC2OCT

Description

Returns the octal equivalent of the number.

Syntax

```
DEC2OCT(number, [ places])
```

- number: The number to convert.
- places: The number of digits to return, up to a limit of 10 digits. If the result is shorter than the specified number of digits, it is left-padded with zeroes.

DECIMAL

Description

Converts the text representation of a number in the specified base to a decimal (base 10) value.

Syntax

```
DECIMAL(value, radix)
```

- value: The text value to convert.
- radix: The base of the value to convert.

DEGREES

Description

Converts radians to degrees for the specified angle.

Syntax

```
DEGREES(value)
```

- value: The angle to convert.

DELTA

Description

If value1 and value2 are equal, returns 1; otherwise, returns 0.

Syntax

```
DELTA(value1, value2)
```

- value1: The first number to compare.
- value2: The second number to compare.

DEVSQ

Description

Returns the sum of squares of deviations. If an array or reference contains text, logical values, or empty cells, the function ignores their values, but it includes values of zero in cells.

Syntax

```
DEVSQ(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

DIMENSIONS

Description

Returns the dimension of a numeric value. If there are no dimensions, the function returns an empty string. If the argument is not numeric, the function returns a #VALUE error.

A dimension is a property that can be measured, such as time (T), length (L), or mass (M). A unit is a value of a dimension; for example, seconds (sec) or meters (m).

Syntax

```
DIMENSIONS(value)
```

- value: The value to return the dimension for.

Example

Formula	Result
=DIMENSIONS(50) where 50 currently has the "m" (meters) unit	L

Notes

- Valid dimension values are from the base library javax.measure.unit.Dimension.
- This function is similar to UNITS(), but UNITS() returns the unit value instead of the dimension value.

DISC

Description

Returns the discount rate for a security.

Syntax

```
DISC(settlement, maturity, par, redemption, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- par: The security's price.
- redemption: The security's redemption value.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

DISTINCTROWS

Description

Combines a set of ranges into a single range while removing any rows that are duplicates. DISTINCTROWS evaluates text and instance values as not distinct from each other. When the supplied range contains both an instance value and a text string that are the same, the function returns 1 row, the instance value. DISTINCTROWS is case-sensitive.

Syntax

```
DISTINCTROWS(range1, [ range2], ...)
```

- range1: The first range of rows.
- range2: The second range of rows.

Example

This is the original spreadsheet:

	A	B	C	D	E
1	Heading1	Heading2		Heading3	Heading4
2	A	1		B	2
3	B	2		2	B
4	C	3		C	5
5	D	4		D	4

The result of `=DISTINCTROWS(A2:B5,D2:E5)` where the formula is in cell A7, is:

	A	B	C	D	E
7	A	1			
8	B	2			
9	C	3			
10	D	4			
11	2	B			
13	C	5			

Notes

- This function takes any number of ranges. Typically each range has a similar row structure, but this is not required.
- The function adds a row to the final result if a row containing the exact same values in the exact same order is not already present in the result. This enables you to combine rows from different ranges and remove duplicate rows. The function considers all values in a row when determining if a row is a duplicate or not.
- This function is intended for use in array formulas.

Related Functions

UNIQUE

DISTINCTROWS2

Description

Combines a set of ranges into a single range while removing any rows that are duplicates. DISTINCTROWS2 evaluates text and instance values as not distinct from each other. When the supplied range contains both an instance value and a text string that are the same, the function returns 1 row, the instance value. DISTINCTROWS2 is case-insensitive.

Syntax

```
DISTINCTROWS2(range1, [ range2], ...)
```

- range1: The first range of rows.
- range2: The second range of rows.

Example

This is the original spreadsheet:

	A	B	C	D	E
1	Heading1	Heading2		Heading3	Heading4
2	A	1		B	2
3	B	2		2	B
4	C	3		C	5
5	D	4		D	4

The result of `=DISTINCTROWS2(A2:B5,D2:E5)` where the formula is in cell A6, is:

	A	B	C	D	E
7	A	1			
8	B	2			
9	C	3			
10	D	4			
11	2	B			
13	C	5			

Notes

- This function takes any number of ranges. Typically each range has a similar row structure, but this is not required.
- The function adds a row to the final result if a row containing the exact same values in the exact same order is not already present in the result. This enables you to combine rows from different ranges and remove duplicate rows. The function considers all values in a row when determining if a row is a duplicate or not.
- This function is intended for use in array formulas.

Related Functions

UNIQUE2

DIVIDE

Description

Returns the result of a division operation.

Syntax

`DIVIDE(numerator, denominator)`

- numerator: The number to be divided (dividend).
- denominator: The number to divide the numerator by.

Example

Formula	Result
=DIVIDE(A1,A2) Where cell A1 contains `42` and A2 contains `20` .	2.1

Notes

- If you want to obtain only the integer portion of the result, use QUOTIENT.

DOLLARDE

Description

Converts a dollar amount from fractional format into decimal format.

Syntax

`DOLLARDE(fractional_dollar, fraction)`

- fractional_dollar: The fractional dollar amount to be converted to decimal.
- fraction: The divisor.

DOLLARFR

Description

Converts a dollar amount from decimal format into fractional format.

Syntax

`DOLLARFR(decimal_dollar, fraction)`

- decimal_dollar: The decimal dollar amount to be converted to fractional.
- fraction: The divisor.

DROP

Description

Returns a subset of the specified array, excluding the specified number of rows or columns from the start or end of the array.

Syntax

```
DROP(array, rows, [columns])
```

- array: The array to drop rows or columns from.
- rows: The number of rows to drop. If you specify a negative value, the function drops rows from the end of the array.
- columns: The number of columns to drop. If you specify a negative value, the function drops columns from the end of the array.

DUR2DAYS

Description

Returns the number of days, based on the specified duration.

Syntax

```
DUR2DAYS(duration)
```

- duration: The time value to convert. Examples: 1w2d3h4m5s6ms or 11:2:33:05:05.600.

Example

Formula	Result
=DUR2DAYS("1w2d3h4m5s6ms")	9.127835718
=DUR2DAYS("11:2:33:05:05.600")	80.378537037

DUR2HOURS

Description

Returns the number of hours, based on the specified duration.

Syntax

```
DUR2HOURS(duration)
```

- duration: The time value to convert. Examples: 1w2d3h4m5s6ms or 11:2:33:05:05.600.

Example

Formula	Result
=DUR2HOURS("1w2d3h4m5s6ms")	219.068057222
=DUR2HOURS("11:2:33:05:05.600")	1929.084888889

DUR2MILLISECONDS

Description

Returns the number of milliseconds, based on the specified duration.

Syntax

```
DUR2MILLISECONDS(duration)
```

- duration: The time value to convert. Examples: 1w2d3h4m5s6ms or 11:2:33:05:05.600.

Example

Formula	Result
=DUR2MILLISECONDS("1w2d3h4m5s6ms")	788645006
=DUR2MILLISECONDS("11:2:33:05:05.600")	6944705600

DUR2MINUTES

Description

Returns the number of minutes, based on the specified duration.

Syntax

`DUR2MINUTES(duration)`

- duration: The time value to convert. Examples: 1w2d3h4m5s6ms or 11:2:33:05:05.600.

Example

Formula	Result
=DUR2MINUTES("1w2d3h4m5s6ms")	13144.083433333
=DUR2MINUTES("11:2:33:05:05.600")	115745.093333333

DUR2SECONDS

Description

Returns the number of seconds, based on the specified duration.

Syntax

`DUR2SECONDS(duration)`

- duration: The time value to convert. Examples: 1w2d3h4m5s6ms or 11:2:33:05:05.600.

Example

Formula	Result
=DUR2SECONDS("1w2d3h4m5s6ms")	788645.006
=DUR2SECONDS("11:2:33:05:05.600")	6944705.6

DUR2WEEKS

Description

Returns the number of weeks, based on the specified duration.

Syntax

`DUR2WEEKS(duration)`

- duration: The time value to convert. Examples: 1w2d3h4m5s6ms or 11:2:33:05:05.600.

Example

Formula	Result
=DUR2WEEKS("1w2d3h4m5s6ms")	1.303976531
=DUR2WEEKS("11:2:33:05:05.600")	11.482648148

DURATION

Description

Returns a number representing the duration of a fixed interest security in years. Maturity is based on the Macaulay duration. For the modified duration, use MDURATION() or BONDMDURATION().

Syntax

```
DURATION(settlement, maturity, coupon, yield, frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- coupon: The security's annual coupon rate.
- yield: The security's annual yield.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

DURATIONVALUE

Description

Returns the duration value, based on the specified time value arguments. Each argument is optional but at least one must be provided. Include a comma for any omitted argument.

Syntax

```
DURATIONVALUE([ weeks], [ days], [ hours], [ minutes], [ seconds], [
milliseconds])
```

- weeks: The number of weeks.
- days: The number of days.
- hours: The number of hours.
- minutes: The number of minutes.
- seconds: The number of seconds.
- milliseconds: The number of milliseconds.

Example

Formula	Result
=DURATIONVALUE(1,2,3,4,5,6)	1w 2d 3h 4m 5s 6ms
=DURATIONVALUE(,,5)	5h

Formula	Result
=DURATIONVALUE(0,2,-24)	1d

E

Description

Returns the value of the base of natural logarithms (approximately 2.71828182845904523536).

Syntax

`E()`

Example

Formula	Result
=E()	2.718281828

EDATE

Description

Returns a date that is the specified number of months after or before the start date.

Syntax

`EDATE(start_date, months)`

- start_date: The date to add or subtract months from. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- months: The number of months to add or subtract from the start_date.

Example

Formula	Result
=EDATE("15-Dec-2016",5)	5/15/2017
=EDATE("15-Dec-2016",-5)	7/15/2016

EFFECT

Description

Returns the effective (net) annual interest rate based on the specified nominal rate.

Syntax

`EFFECT(nominal_rate, npery)`

- nominal_rate: The nominal interest rate.
- npery: The number of compounding periods for each year.

EMAIL

Description

Converts an email address into a clickable `mailto` link. Optionally, you can insert email details such as email content and recipients.

Syntax

`EMAIL(email, [ text], [ cc], [ bcc], [ subject], [ body])`

- email: The email address.
- text: The clickable text to be displayed in the cell. If you don't include this argument, the function uses the email address to generate the clickable text.
- cc: The copy list of recipients for the email. To separate multiple email addresses, use a character that your email client supports.
- bcc: The blind copy list of recipients for the email. To separate multiple email addresses, use a character that your email client supports.
- subject: The subject of the email.
- body: The body of the email.

Example

Formula	Result
=EMAIL("william.davis@workday.com", "Will Davis")	Will Davis (as a clickable link)

ENCODEURL

Description

Returns the URL-encoded string for the specified text.

Syntax

`ENCODEURL(text)`

- text: The text to encode.

Example

Formula	Result
=ENCODEURL(A1) Where cell A1 contains `<Workday>` .	%3CWorkday%3E
=ENCODEURL(A1) Where cell A1 contains `Directeur régional des ventes` .	Directeur+r%C3%A9gional+des+ventes

EOMONTH

Description

Returns the date of the last day of the month (ending in the evening at 11:59:59.999 PM or 23:59:59.999) using the specified start_date and number of months to add or subtract.

Syntax

`EOMONTH(start_date, months)`

- start_date: The date to determine the last day of the month for. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- months: The number of months to add or subtract from the start_date before determining the last day of the month.

Example

Formula	Result
=EOMONTH("15-Dec-2016",5)	5/31/2017
=EOMONTH("15-Dec-2016",-5)	7/31/2016

EQ

Description

If number1 and number2 are equal, returns TRUE; otherwise, returns FALSE.

Syntax

`EQ(number1, number2)`

- number1: The first number to compare.
- number2: The second number to compare.

Example

Formula	Result
=EQ(A1,A2) Where cell A1 contains 4 and A2 contains 4 .	TRUE

ERF

Description

Returns the error function, based on the specified lower and upper limits for the integral.

Syntax

`ERF(lower_bound, [ upper_bound])`

- lower_bound: The lower limit.
- upper_bound: The upper limit. If empty, the function returns a result based on the lower limit and 0.

ERF.PRECISE

Description

Returns the error function, based on the specified lower and upper limits for the integral.

Syntax

`ERF.PRECISE(lower_bound, [ upper_bound])`

- lower_bound: The lower limit.
- upper_bound: The upper limit. If empty, the function returns a result based on the lower limit and 0.

ERFC

Description

Returns the complementary error function, based on the specified lower limit for the integral and infinity.

Syntax

`ERFC(x)`

- x: The lower limit.

ERFC.PRECISE

Description

Returns the complementary error function, based on the specified lower limit for the integral and 0.

Syntax

`ERFC.PRECISE(x)`

- x: The lower limit.

ERROR.TYPE

Description

Returns an integer representing the error type of the specified value. 1 = #NULL!; 2 = #DIV/0!; 3 = #VALUE!; 4 = #REF!; 5 = #NAME?; 6 = #NUM!; 7 = #N/A; 8 = #GETTING_DATA; #N/A = Anything else.

Syntax

`ERROR.TYPE(error_value)`

- error_value: The value to return the error type for.

Example

Formula	Result
=ERROR.TYPE(12/0)	2
=ERROR.TYPE({10,11,12})	#N/A

Notes

- The #GETTING_DATA state occurs during a live data refresh and isn't an actual error.

EVEN

Description

Rounds a number to the nearest even integer. Rounding is away from zero; the function rounds positive numbers up and negative numbers down.

Syntax

`EVEN(number)`

- number: The number to round.

Example

Formula	Result
=EVEN(12.34)	14
=EVEN(-12.34)	-14

EXACT

Description

Determines if two text strings are exactly equal. If so, returns TRUE; otherwise, returns FALSE. The function is case-sensitive, and it considers spaces.

Syntax

`EXACT(text1, text2)`

- text1: The first text to compare.
- text2: The second text to compare.

Example

Formula	Result
=EXACT(A1,A2) Where cell A1 contains `6000:salaries and wages` and A2 contains `6000:Salaries and Wages` .	FALSE

Notes

- When comparing a single instance value to a string that visually appears to be the same, the function evaluates them as equal. When comparing a single value in a multi-instance field to a string that visually appears to be the same, the formula evaluates them as *not* equal.

EXP

Description

Returns e raised to the specified power.

Syntax

`EXP(value)`

- value: The exponent to raise e to.

Related Functions

EXPM1

EXPAND

Description

Expands an array to the specified dimensions. Optionally you can specify a value to place in the added array cells.

Syntax

`EXPAND(array, [rows], [columns], [pad_with])`

- array: The array to expand.
- rows: The number of rows in the expanded array. If you don't specify the argument, the function doesn't expand the rows.
- columns: The number of columns in the expanded array. If you don't specify the argument, the function doesn't expand the rows.
- pad_with: The value to pad the result with. If you don't specify the argument, the default is #N/A.

EXPM1

Description

Returns e raised to the power you specify, minus 1.

Syntax

`EXPM1(value)`

- value: The exponent to raise e to.

Example

Formula	Result
=EXPM1(2)	6.389056099

EXPON.DIST

Description

Returns the exponential distribution, using either the cumulative distribution function or the probability density function.

Syntax

```
EXPON.DIST(value, lambda, cumulative)
```

- value: The input value.
- lambda: The lambda value.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

EXPONDIST

Description

Returns the exponential distribution, using either the cumulative distribution function or the probability density function.

Syntax

```
EXPONDIST(value, lambda, cumulative)
```

- value: The input value.
- lambda: The lambda value.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

EXPONENT

Description

Returns the exponent of the internal double representation of the number.

Syntax

```
EXPONENT(value)
```

- value: The number to return the exponent for.

FACT

Description

Returns the factorial of the number, where 0 <= number <= 170.

Syntax

```
FACT(number)
```

- number: The number to calculate the factorial for.

FACTDOUBLE

Description

Returns the double factorial of the number, where -1 <= number <= 301.

Syntax

```
FACTDOUBLE(number)
```

- number: The number to calculate the double factorial for.

FALSE

Description

Returns the logical value False.

Syntax

`FALSE()`

Example

Formula	Result
=FALSE()	FALSE

FILTER

Description

Returns a filtered range of data based on criteria that you specify.

Syntax

`FILTER(array, include, [ if_empty])`

- array: The value or expression to evaluate.
- include: A Boolean array with the same height or width as array, that identifies the data to return.
- if_empty: The value to return if the expression results in an empty array. The result must be both empty and an array; otherwise, the function returns the #VALUE error.

Example

This is the original spreadsheet:

	A	B	C
1	Name	Grade	90
2	Joe	85	
3	Alice	92	
4	Annri	88	
5	Navin	97	
6	Carlos	89	
7	Ren	91	

The result of `=FILTER(A1:B7,B1:B7>C1)`, where the formula is placed in cell F1, submitted using the unconstrained array keyboard shortcut, is:

	F	G
1	Name	Grade
2	Alice	92
3	Navin	97

	F	G
4	Ren	91

FIND

Description

Searches the specified text string for a sub-string. Returns the starting position of the find_text in the within_text. The search is case-sensitive. For case-insensitive search, use SEARCH.

Syntax

```
FIND(find_text, within_text, [ start_num])
```

- find_text: The text to find in the string.
- within_text: The text to search.
- start_num: The starting position for the search.

Example

Formula	Result
=FIND("Sales",A1) Where cell A1 contains `Regional Sales Manager` .	10

FIXED

Description

Rounds a number to the specified number of decimal places and formats the result as text.

Syntax

```
FIXED(number, [ decimals], [ no_commas])
```

- number: The number to format.
- decimals: The number of decimal places to use for the result.
- no_commas: Specifies whether to omit the thousands separator character. FALSE or Empty = include separators; TRUE = omit separators. The default is TRUE.

Example

Formula	Result
=FIXED(A1) Where cell A1 contains `$2451.00` .	2451.000
=FIXED(A1,4,FALSE) Where cell A1 contains `1234567` .	1,234,567.0000

FISHERINV

Description

Returns the inverse of the Fisher transformation.

Syntax

`FISHERINV(number)`

- number: The number to transform.

FISHER

Description

Returns the Fisher transformation.

Syntax

`FISHER(number)`

- number: The number to transform.

FLATTEN

Description

Returns an expanded range of data based on the hierarchical data that you specify. Typically you use this function to expand an organization's manager and employee information so that it displays all levels of the hierarchy.

Syntax

`FLATTEN(source, primary_column, detail_column, data_columns, [ root_values])`

- source: The range of data to flatten.
- primary_column: The index value in the source for the higher-level item that you want to flatten by.
- detail_column: The index value in the source for the lower-level item that you want to flatten by.
- data_columns: The column indexes to select into the output.
- root_values: Values that identify the topmost items in the hierarchy. The default is null (indicating the highest point in the hierarchy).

FLOOR

Description

Rounds a number down to the nearest integer, based on the specified significance.

Syntax

`FLOOR(number, significance)`

- number: The number to round.
- significance: The multiple of significance to round the number down to. The default is 1, which causes the number to be rounded down to the next smaller integer.

Example

Formula	Result
=FLOOR(12.34,1)	12
=FLOOR(12.34,10)	10
=FLOOR(12.34,2)	12
=FLOOR(-12.34,2)	-14
=FLOOR(-12.34,-1)	-12

FLOOR.MATH

Description

Rounds a number down to the nearest integer, based on the significance and mode that you specify.

Syntax

```
FLOOR.MATH(value, [ significance], [ mode])
```

- value: The number to round.
- significance: The multiple of significance to round the number down to. If you omit the argument, the function uses the value 1, which causes the function to round the number down to the next smaller integer.
- mode: If you specify any number other than zero, the function rounds negative numbers up (toward zero).

Example

Formula	Result
=FLOOR.MATH(12.34,1)	13
=FLOOR.MATH(12.34,10)	20
=FLOOR.MATH(-12.34,1)	-12
=FLOOR.MATH(-12.34,1)	-13
=FLOOR.MATH(-12.34,-1,1)	-13

FLOOR.PRECISE

Description

Rounds a number down to the nearest integer, based on the multiple of significance that you specify. The function rounds positive numbers down (toward zero) and also rounds negative numbers down (away from zero, becoming more negative).

Syntax

```
FLOOR.PRECISE(value, [ significance])
```

- value: The number to round.
- significance: The multiple of significance to round the number down to. If you omit the argument, the function uses the value 1, causing the function to round down to the next smaller integer.

Example

Formula	Result
=FLOOR.PRECISE(12.34,1)	12
=FLOOR.PRECISE(12.34,10)	10
=FLOOR.PRECISE(-12.34,0)	-13
=FLOOR.PRECISE(-12.34,2)	-14
=FLOOR.PRECISE(-12.34,-1,1)	-13

FORECAST

Description

Returns a predicted value based on the specified known values, using linear regression.

Syntax

`FORECAST(x, known_ys, known_xs)`

- x: The number to predict a value for.
- known_ys: The array of y (dependent) values.
- known_xs: The array of x (independent) values.

Related Functions

FORECAST.WD.SEASONAL

FORECAST.WD.SEASONAL

Description

Returns a predicted sequence of values using patterns in the historical linear and non-linear data you specify.

Syntax

`FORECAST.WD.SEASONAL(values, num_forecast)`

- values: An array of historical values to predict from.
- num_forecast: The number of values to predict.

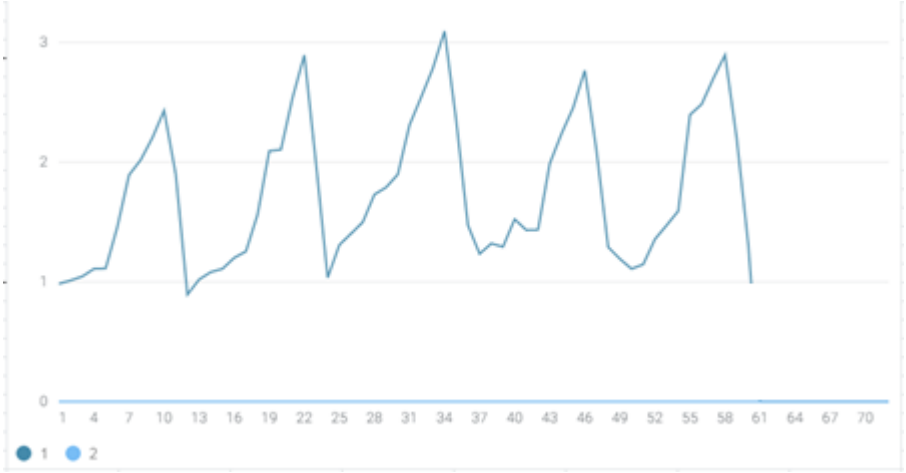
Example

An analyst wants to predict revenue values for the 12 months of 2017 based on historical revenue values from 2012 through 2016. The values for 2012-2016 are in cells C2 through C61. This table excerpt shows the values from 2012:

	A	B	C
1	Year	Month	Actual Total Rev per Month (millions)
2	2012	Jan	0.984
3	2012	Feb	1.012
4	2012	Mar	1.045

5	2012	Apr	1.109
6	2012	May	1.112
7	2012	Jun	1.458
8	2012	Jul	1.892
9	2012	Aug	2.018
10	2012	Sep	2.203
11	2012	Oct	2.43
12	2012	Nov	1.897
13	2012	Dec	0.894

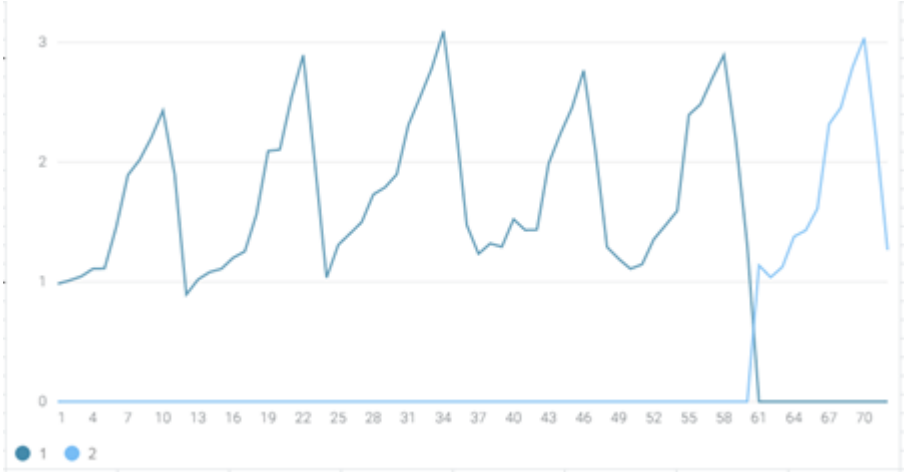
This graph shows the values as a dark blue line:



This formula adds 12 predicted values to the workbook:

```
=FORECAST.WD.SEASONAL(C2:C61, 12)
```

The light blue line in this graph shows the forecast values:



FORMULATEXT

Description

Returns a formula as text.

Syntax

`FORMULATEXT(reference)`

- reference: The reference to a cell that contains the formula.

Example

Formula	Result
=FORMULATEXT(!'Sheet 1'!A1)	The formula contained in Sheet 1 in cell A1 .

FREQUENCY

Description

Returns the number of values in the data_array that are in the specified bin_array range.

Syntax

`FREQUENCY(data_array, bin_array)`

- data_array: The list of values to group into ranges and to count according to those ranges.
- bin_array: The list of values that specifies the ranges to group the data_array by.

FV

Description

Returns the future value of an investment based on periodic payments and a constant interest rate.

Syntax

`FV(rate, nper, pmt, [ pv], [ type])`

- rate: The interest rate for each period.
- nper: The number of periods over the life of the investment.
- pmt: The payment made for each period.
- pv: The present value of the investment.
- type: The payment timing. 0 or Empty = the payment occurs at the end of the period; 1 = the payment occurs at the beginning of the period.

FVSCHEDULE

Description

Returns the future value of an investment based on a variable interest rate.

Syntax

`FVSCHEDULE(principal, schedule)`

- principal: The present value of the investment.

- schedule: The array of values specifying the schedule of interest rates to apply to the principal.

GAMMALN

Description

Returns the natural logarithm of the gamma function value.

Syntax

`GAMMALN(value)`

- value: The input value.

GAMMALN.PRECISE

Description

Returns the natural logarithm of the gamma function value.

Syntax

`GAMMALN.PRECISE(value)`

- value: The input value.

GAUSS

Description

Returns the probability of being in the set (mean, mean + N-sigma). Negative probabilities are interpreted as being leftward of the mean.

Syntax

`GAUSS(value)`

- value: The input value.

GCD

Description

Returns the greatest common divisor of the specified number values.

Syntax

`GCD(number1, [ number2], ...)`

- number1: The number or list of numbers.
- number2: The number or list of numbers.

Example

Formula	Result
=GCD(8, 12)	4

Notes

- Worksheets ignores string values.

GEOMEAN

Description

Returns the geometric mean of a specified list of values.

Syntax

```
GEOMEAN(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

GESTEP

Description

If the number is greater than or equal to the specified step, returns TRUE; otherwise, returns FALSE.

Syntax

```
GESTEP(number, [ step])
```

- number: The number to evaluate.
- step: The step size to compare the number to. The default is 0.

GETPIVOTDATA

Description

Returns related data from a pivot table without directly referencing cells. Using this function instead of a cell reference preserves the data if the pivot table changes in the future. You can include multiple fields and items.

Syntax

```
GETPIVOTDATA(data_field, pivot_table, [ field1, item1, field2, item2], ...)
```

- data_field: The pivot summary value field name or display label.
- pivot_table: A pivot table cell or range reference.
- field1: A row or column pivot field name or display label.
- item1: The corresponding row or column pivot field value.

GROUPBY

Description

GROUPBY is a powerful function that can often replace COUNTIF(S), AVERAGEIF(S), and SUMIF(S). GROUPBY aggregates data, and orders the results based on the order that you specify. The grouping is based on a key that you can predefine in the table, or you can define it using columns in the workbook. The result looks similar to a sorted pivot table. This function is often useful as part of headcount planning.

When you group an array of cells that represent some instance values as strings and others as instances (variables from a Workday report), GROUPBY bases the grouping on the instance name, not the instance ID. GROUPBY groups identical names together, even when the instance IDs are different.

Syntax

```
GROUPBY(join_by_matrix, join_by_matrix_column_indexes, group_by_matrix,
group_by_matrix_column_indexes, group_by_matrix_aggregate_column_indexes,
[group_by_matrix_aggregate_types])
```

- join_by_matrix: The 1-dimensional array of unique identifiers (keys) that identifies the rows to include when grouping.
- join_by_matrix_column_indexes: The starting position of the keys in the join_by_matrix, or the set of columns that comprise the matching key (the column values to concatenate to create the primary key).
- group_by_matrix: The array of cells to group.
- group_by_matrix_column_indexes: The position of the column that contains the keys in the group_by_matrix.
- group_by_matrix_aggregate_column_indexes: The positions of the columns that contain values to group.
- group_by_matrix_aggregate_types: The operation to perform.

Argument Value	Operation
0	SUM (default)
1	COUNT of non-null values
2	COUNT of non-zero numbers
3	MIN number
4	MAX number
5	AVG number

Example

A workbook has 2 sheets named Working and Organization. This is the Working sheet:

	A	B	C	D	E	F	G	H	I
1	Supervisory Org	Cost Center	Company	Location	Name	Jan (HC) 2017	Feb (HC) 2017	Mar (HC) 2017	Organization KEY
2	Finance	CC-1 Finance	GMS	Pleasanton	Tim	1	1	1	FinanceCC-1 FinanceGMSPleasanton
3	Product Management	CC-2 Product	GMS	Boulder	Josh	0	0	1	Product ManagementCC-2 ProductGMSBoulder
4	Development	CC-3 Development	GMS-CA	Toronto	Brian	1	1	1	DevelopmentCC-3 DevelopmentGMS-CAToronto
5	Product Management	CC-2 Product	GMS	Boulder	Scott	0	1	1	Product ManagementCC-2 ProductGMSBoulder
6	Development	CC-3 Development	GMS-CA	Toronto	Lenny	1	1	1	DevelopmentCC-3 DevelopmentGMS-CAToronto

	A	B	C	D	E	F	G	H	I
7	Sales	CC-4 Sales	GMS-UK	London	Andy	0	1	1	SalesCC-4 SalesGMS-UKLondon
8	Finance	CC-1 Finance	GMS	Pleasanton	Ratna	1	1	1	FinanceCC-1 FinanceGMSPleasanton
9	Sales	CC-4 Sales	GMS-UK	Toronto	Aidan	0	0	1	SalesCC-4 SalesGMS-UKToronto
10	Sales	CC-4 Sales	GMS-UK	London	Katie	1	1	1	SalesCC-4 SalesGMS-UKLondon
11					Total	5	7	9	

The GROUPBY array formula is in cells E2:G6 of the sheet Organization:

=GROUPBY('Organization'!H2:H6,1,'Working'!A2:I10,9,{6, 7, 8})

Alternatively, instead of generating a key separately as shown in column I above and H below, you can specify the columns that you want GROUPBY to use as the key:

=GROUPBY('Organization'!A2:G6,{1,2,3,4},'Working'!A2:H10,{1,2,3,4},{6, 7, 8})

The formula returns these results in sheet Organization:

	A	B	C	D	E	F	G	H
1	Supervisory Org	Cost Center	Company	Location	Jan (HC) 2017	Feb (HC) 2017	Mar (HC) 2017	Organization KEY
2	Finance	CC-1 Finance	GMS	Pleasanton	2	2	2	FinanceCC-1 FinanceGMSPleasanton
3	Product Management	CC-2 Product	GMS	Boulder	0	1	2	Product ManagementCC-2 ProductGMSBoulder
4	Development	CC-3 Development	GMS-CA	Toronto	2	2	2	DevelopmentCC-3 DevelopmentGMS-CAToronto
5	Sales	CC-4 Sales	GMS-UK	London	1	2	2	SalesCC-4 SalesGMS-UKLondon
6	Sales	CC-4 Sales	GMS-UK	Toronto	0	0	1	SalesCC-4 SalesGMS-UKToronto
7					5	7	9	

Note that row 7 in the sheet Organization is a simple calculation of the total headcount number; it's outside the plan entry area.

Notes

- When doing an operation on date values, GROUPBY converts the date to a serial number before calculating and does not restore the display format to a date. To show a date format, select Format > Number > Date .

Related Functions

ARRAYAREA

GROWTH

Description

Calculates the exponential growth curve based on the specified y values and one or more sets of x values, and extends the curve to calculate more y values for a specified set of new x values.

Syntax

```
GROWTH(known_ys, [ known_xs], [ new_xs], [ const])
```

- known_ys: The known y values.
- known_xs: The known x values.
- new_xs: New x values to calculate y values based on.
- const: Specifies whether to force the b constant to equal 1. TRUE or Empty = calculate normally; FALSE = set b to 1.

GT

Description

Compares two numbers. If number1 is greater than number2, returns TRUE; otherwise, returns FALSE.

Syntax

```
GT(number1, number2)
```

- number1: The first number to compare.
- number2: The second number to compare.

Example

Formula	Result
=GT(A1,A2) Where cell A1 contains 2 and A2 contains `-2` .	TRUE

GTE

Description

Compares two numbers. If number1 is greater than or equal to number2, returns TRUE; otherwise, returns FALSE.

Syntax

```
GTE(number1, number2)
```

- number1: The first number to compare.
- number2: The second number to compare.

Example

Formula	Result
=GTE(A1,A2) Where cell A1 contains 2 and A2 contains -2 .	TRUE

HARMEAN

Description

Returns the harmonic mean of a specified list of values.

Syntax

```
HARMEAN(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

HEX2BIN

Description

Returns the binary equivalent of the hexadecimal number, to the specified number of decimal places.

Syntax

```
HEX2BIN(number, [ places])
```

- number: The hexadecimal number.
- places: The number of digits to return, up to a limit of 10 digits. If the result is shorter than the specified number of digits, it is left-padded with zeroes.

HEX2DEC

Description

Returns the decimal equivalent of the hexadecimal number, to the specified number of decimal places.

Syntax

```
HEX2DEC(number)
```

- number: The hexadecimal number.

HEX2OCT

Description

Returns the octal equivalent of the hexadecimal number, to the specified number of decimal places.

Syntax

```
HEX2OCT(number, [ places])
```

- number: The hexadecimal number.
- places: The number of digits to return, up to a limit of 10 digits. If the result is shorter than the specified number of digits, it is left-padded with zeroes.

HOUR

Description

Returns an integer from 0 to 23 representing only the hour (12:00 AM to 11:00 PM, respectively) specified in the time.

Syntax

```
HOUR(date)
```

- date: The time to get the hour value from. You can enter the time as a serial number, text, a cell reference, or a value returned from another formula.

Example

Formula	Result
=HOUR("16:15:14")	16
=HOUR("5:35 PM")	17
=HOUR("11/1/2017 6:31 PM")	8

HLOOKUP

Description

Finds a value in the top row of the specified data array, and returns the value for the corresponding cell in the same column in a different row.

Syntax

```
HLOOKUP(lookup_value, table_array, row_index_num, [ range_lookup])
```

- lookup_value: The value to find in the first row.
- table_array: The array or table to search.
- row_index_num: The row to return the corresponding value from.
- range_lookup: Specifies whether to require an exact match. FALSE = return an error if an exact match is not found; TRUE or Empty = if the function doesn't find an exact match, use the closest match below the lookup_value as a match.

Related Functions

ARRAYAREA

GROUPBY

MATCHEXACT

MHLOOKUP

HSTACK

Description

Appends the specified arrays into a single array as a sequential set of columns.

Syntax

`HSTACK(array1,[array2],...)`

- range1: The array to append (stack).
- range2: The array to append (stack).

HYPERLINK

Description

Converts a URL into a clickable link on a sheet.

Syntax

`HYPERLINK(url, [ label]`

- url: The URL. For linking within a workbook, precede the path with a pound sign (#).
- label: The clickable text to display in the cell. If you don't include this argument, the function uses the url argument as the clickable text. You can use a cell reference to contain the label.

Example

Formula	Result
=HYPERLINK("http://www.workday.com","Workday")	Workday (as a clickable link)
=HYPERLINK("#"&REMOVEROWS(INFO("wd.sheet.names"),1),"Create")	Creates a list of clickable links to the sheets of the workbook, excluding the current sheet.
=IFERROR(HYPERLINK(F1),"None")	Cell F1 displays the text None if an error occurs in the HYPERLINK function.

Notes

- This function is similar to the URL function; however, to maintain compatibility with Excel, HYPERLINK doesn't have the optional tooltip argument.
- Keep in mind that if you use the right-click (context) menu in a Microsoft® Excel® workbook to create a hyperlink, and then you upload the workbook to Worksheets, it isn't converted into a HYPERLINK function; it becomes a link-formatted cell containing a URL.

Related Functions

URL

HYPGEOM.DIST

Description

Returns the hypgeomdist distribution probability (cumulative or density) for the specified value.

Syntax

```
HYPGEOM.DIST(sample_s, number_sample, population_s, number_pop, [ cumulative])
```

- sample_s: The number of successes in the sample.
- number_sample: The size of the sample.
- population_s: The number of successes in the population.
- number_pop: The size of the population.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

Example

For the table:

	A
1	1
2	4
3	8
4	20

The formulas and results are:

Formula	Result
HYPGEOM.DIST(A1,A2,A3,A4,TRUE)	0.465428277
HYPGEOM.DIST(A1,A2,A3,A4,FALSE)	0.363261094

Notes

- This function does the same action as HYPGEOMDIST().
- If an impossible scenario occurs (for example, there are more successes than trials), Worksheets returns an *invalid number* error.

HYPGEOMDIST

Description

Returns the hypgeomdist distribution probability (cumulative or density) for the specified value.

Syntax

```
HYPGEOMDIST(sample_s, number_sample, population_s, number_pop, [ cumulative])
```

- sample_s: The number of successes in the sample.
- number_sample: The size of the sample.
- population_s: The number of successes in the population.
- number_pop: The size of the population.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

Example

For the table:

	A
1	1
2	4
3	8
4	20

The formulas and results are:

Formula	Result
HYPGEOMDIST(A1,A2,A3,A4,TRUE)	0.465428277
HYPGEOMDIST(A1,A2,A3,A4,FALSE)	0.363261094

Notes

- This function does the same action as HYPGEOM.DIST().
- If an impossible scenario occurs (for example, there are more successes than trials), Worksheets returns an *invalid number* error.

HYPOT

Description

Calculates the hypotenuse of a right triangle using the Pythagorean theorem.

Syntax

```
HYPOT(x, y)
```

- x: The length of a non-hypotenuse side of the triangle.
- y: The length of the second non-hypotenuse side of the triangle.

Example

Formula	Result
=HYPOT(3,4)	5

IF

Description

If the condition is True, returns the then value; if False, returns the else value. You can nest up to 7 IF functions in a formula.

Syntax

```
IF(condition, [ then], [ else])
```

- condition: The condition to evaluate.
- then: The value to return if the condition is True.
- else: The value to return if the condition is False.

Example

Formula	Result
=IF(A1=0, A1, -A1) where A1 contains 10	-10

IFEMPTY

Description

If the specified value or expression results in an empty array, the function returns the value_if_empty value; otherwise, it returns the value. This function is intended for use with unconstrained array formulas.

Syntax

```
IFEMPTY(value, value_if_empty)
```

- value: The value or expression to evaluate.
- value_if_empty: The value to return if the initial value or expression results in an empty array; it must be both empty and an array. Otherwise, the function returns value.

IFERROR

Description

If the specified value or expression results in an error, returns the value_if_error value; otherwise, returns the value.

Syntax

```
IFERROR(value, value_if_error)
```

- value: The value or expression to evaluate.
- value_if_error: The value to return if the initial value or expression results in an error.

Example

Formula	Result
=IFERROR(12/0, 0)	0
=IFERROR(1*5,0)	5
=IFERROR(HYPERLINK(F1),"None")	Cell F1 displays the text None if an error occurs in the HYPERLINK function.

IFNA

Description

If the specified value or expression results in an #N/A error, returns the value_if_error value or run the alternative formula; otherwise, returns the value.

Syntax

```
IFNA(value, value_if_error)
```

- value: The value or expression to evaluate.

- value_if_error: The value to return, or a formula to run, if the initial value or expression results in an #N/A error. If you want to run a formula, don't include the equals sign (=) at the beginning.

Example

Formula	Result
=IFNA(1*5, "error")	5
=IFNA(A1, "error") where A1 contains #N/A	error

IMABS

Description

Returns the absolute value of the specified complex number.

Syntax

`IMABS(inumber)`

- inumber: The complex number.

IMAGINARY

Description

Returns the imaginary coefficient of the specified complex number.

Syntax

`IMAGINARY(inumber)`

- inumber: The complex number.

IMARGUMENT

Description

Returns the argument θ of the specified complex number.

Syntax

`IMARGUMENT(inumber)`

- inumber: The complex number.

IMCONJUGATE

Description

Returns the complex conjugate of the specified complex number.

Syntax

`IMCONJUGATE(inumber)`

- inumber: The complex number.

IMCOS

Description

Returns the cosine of the specified complex number.

Syntax

`IMCOS(inumber)`

- inumber: The complex number.

IMCOSH

Description

Returns the hyperbolic cosine of the specified complex number.

Syntax

`IMCOSH(inumber)`

- inumber: The complex number.

IMCOT

Description

Returns the cotangent of the specified complex number.

Syntax

`IMCOT(inumber)`

- inumber: The complex number.

IMCOTH

Description

Returns the hyperbolic cotangent of a complex number.

Syntax

`IMCOTH(inumber)`

- inumber: The complex number. Specify the number as text in either x+yi or x+yj format, where the x component or y component is optional.

Example

Formula	Result
=IMCOTH("3 + 4i")	0.9992669278059017-0.0049011823943044056i
=IMCOTH(3)	1.004969823313689

IMCSC

Description

Returns the cosecant of the specified complex number.

Syntax

`IMCSC(inumber)`

- inumber: The complex number.

IMCSCH

Description

Returns the hyperbolic cosecant of the specified complex number.

Syntax

`IMCSCH(inumber)`

- inumber: The complex number.

IMDIV

Description

Returns the quotient of the two specified complex numbers.

Syntax

`IMDIV(inumber1, inumber2)`

- inumber1: The number to be divided (dividend).
- inumber2: The number to divide the dividend by (divisor).

IMEXP

Description

Returns the exponential of the specified complex number.

Syntax

`IMEXP(inumber)`

- inumber: The complex number.

IMLN

Description

Returns the natural logarithm of the specified complex number.

Syntax

`IMLN(inumber)`

- inumber: The complex number.

IMLOG10

Description

Returns the base-10 logarithm of the specified complex number.

Syntax

`IMLOG10(inumber)`

- inumber: The complex number.

IMLOG2

Description

Returns the base-2 logarithm of the specified complex number.

Syntax

`IMLOG2(inumber)`

- inumber: The complex number.

IMPOWER

Description

Returns the complex number raised to the specified power.

Syntax

`IMPOWER(inumber, power)`

- inumber: The complex number.
- power: The exponent.

IMPRODUCT

Description

Returns the product of the specified complex numbers.

Syntax

`IMPRODUCT(inumber, [ inumber2], ...)`

- inumber1: A complex number or list of complex numbers.
- inumber2: A complex number or list of complex numbers.

IMREAL

Description

Returns the real coefficient of the specified complex number.

Syntax

`IMREAL(inumber)`

- inumber: The complex number.

IMSEC

Description

Returns the secant of the specified complex number.

Syntax

`IMSEC(inumber)`

- inumber: The complex number.

IMSECH

Description

Returns the hyperbolic secant of the specified complex number.

Syntax

`IMSECH(inumber)`

- inumber: The complex number.

IMSIN

Description

Returns the sine of the specified complex number.

Syntax

`IMSIN(inumber)`

- inumber: The complex number.

IMSINH

Description

Returns the hyperbolic sine of the specified complex number.

Syntax

`IMSINH(inumber)`

- inumber: The complex number.

IMSQRT

Description

Returns the square root of the specified complex number.

Syntax

`IMSQRT(inumber)`

- inumber: The complex number.

IMSUB

Description

Subtracts inumber2 from inumber1.

Syntax

```
IMSUB(inumber1, inumber2)
```

- inumber1: The complex number to be subtracted from.
- inumber2: The complex number to subtract.

IMSUM

Description

Returns the sum of one or more complex numbers, or lists of complex numbers.

Syntax

```
IMSUM(inumber1, [ inumber2], ...)
```

- inumber1: The complex number or list of complex numbers.
- inumber2: The complex number or list of complex numbers.

IMTAN

Description

Returns the tangent of the specified complex number.

Syntax

```
IMTAN(inumber)
```

- inumber: The complex number.

IMTANH

Description

Returns the hyperbolic tangent of a complex number.

Syntax

```
IMTANH(inumber)
```

- inumber: The complex number. Specify the number as text in either x+yi or x+yj format, where the x component or y component is optional.

Example

Formula	Result
=IMTANH("3 + 4i")	1.000709536067233+0.004908258067495992i

Formula	Result
=IMTANH(3)	0.9950547536867306

IN

Description

Determines whether a value or list of values that you specify is in another list of values. If so, returns True; otherwise, returns False.

Syntax

```
IN(source, target)
```

- source: The number or list of numbers.
- target: The array of target values.

Example

Formula	Result
=IN(5,{1,2,3,4,5})	TRUE
=IN({"a","b"},{"d","c","b","a"})	TRUE

INDEX

Description

Returns a value, or a reference to a value, based on the specified row and column of a range of cells. You can use INDEX() for arrays (finding a cell reference in a single range) and ranges (finding a reference from ranges that have more than one area).

Syntax

```
INDEX(region, row_number, [ column_number], [ area_number])
```

- region: The array or range of cells.
- row_number: The row number for the specified array or range.
- column_number: The column number of the array or range.
- area_number: Specifies the number of the area to be used, if your original range contained more than one range. Areas are numbered in the order you specified them.

Example

The examples are based on this workbook:

	A	B	C	D	E
1	Item	2014	2015	2016	2017
2	100	6452	6557	6772	6457
3	200	3458	3568	4668	3455
4	300	6791	6552	5382	6235
5	400	3524	6592	3562	3899

	A	B	C	D	E
6	500	6458	3112	2855	3298
7	600	7211	3760	2468	6412
8	700	1369	2399	3465	1928
9	800	9271	8364	7561	2649
10	900	1135	1999	2023	3841

The result of `=INDEX(B1:B6,5)` is `3524` .

The result of `=INDEX(B1:C6,5,2)` is `6592` .

The result of `=INDEX(B1:C6,5,0)` is `3524 6592` .

The result of `=INDEX((C1:D5,B6:C7,B9:D10),4,2,1)` is `5382` .

The result of `=INDEX((C1:D5,B6:C7,B9:D10),2,0,3)` is `1135 1999 2023` .

INDIRECT

Description

Returns a cell reference corresponding to the specified text string. This function is volatile; it generates a new value any time any cell in the workbook changes.

Syntax

`INDIRECT(ref_text, [ a1_ref_style_flag])`

- ref_text: The text describing the cell reference.
- a1_ref_style_flag: The cell reference style. TRUE or Empty = A1 style; this is the only style that Worksheets supports.

Example

The examples are based on this spreadsheet:

	A	B	C
1	4	6	12
2	8	10	18
3	14	22	26

Formula	Result
=INDIRECT("C2")	18
=GCD(INDIRECT("A1:"&ADDRESS(1,3)))	2

Notes

- The function doesn't run as a volatile function if the Worksheets calculation mode is set to Manual in File > Settings .
- If the reference text attempts to reference a cell on a different sheet, Worksheets returns an *invalid cell reference* error.

INFO

Description

Returns the requested information about the Workday environment. Possible values are: memused, recalc (recalculation mode), release (build version), system, username (the most recent user to run a function), usertimezoneshort, and usertimezonelong.

Syntax

```
INFO(type_text)
```

- type_text: The type of information to return. Possible values are:
  - memused: Memory usage.
  - recalc: Recalculation mode: Automatic or Manual.
  - release: Build version of Worksheets.
  - system: System identifier string such as Sheets Engine Server.
  - username: The Workday user name. Note that this argument gives only one result: the most recent user to run the function.
  - usertimezoneshort: Time zone abbreviation.
  - usertimezonelong: Time zone name.
  - wd.sheet.count: The count of sheets in the workbook.
  - wd.sheet.name: The name of the sheet containing the formula.
  - wd.sheet.names: The names of all sheets in the workbook.
  - wd.workbook.id: The workbook ID.
  - wd.workbook.name: The workbook name.

Example

Formula	Result
=INFO("usertimezonelong")	Mountain Daylight Time

INSTANCE

Description

Returns a single-instance value.

Syntax

```
INSTANCE(id, descriptor)
```

- id: The Workday ID string value. Example: "371ae106294a76f2702".
- descriptor: The label string. Example: "Boulder".

INSTANCE.DESCRIPTOR

Description

For a single-instance value, the function returns the descriptor (string) of the instance.

Syntax

```
INSTANCE.DESCRIPTOR(instance)
```

- instance: A reference to the instance. Example: =INSTANCE.DESCRIPTOR(A1).

INSTANCE.ID

Description

For a single-instance value, the function returns the ID of the instance.

Syntax

```
INSTANCE.ID(instance)
```

- instance: A reference to the instance. Example: =INSTANCE.ID(A1).

INT

Description

Returns the integer closest to, and below, the number.

Syntax

```
INT(value)
```

- value: The number to return the closest lower integer to.

Example

Formula	Result
=INT(12.34)	12
=INT(-12.34)	-13

Related Functions

RINT

INTERCEPT

Description

Returns the intercept of the linear regression line, based on the specified x values and y values.

Syntax

```
INTERCEPT(known_ys, known_xs)
```

- known_ys: The array of y values.
- known_xs: The array of x values.

INTRATE

Description

Returns the interest rate for a fully invested security.

Syntax

```
INTRATE(settlement, maturity, investment, redemption, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- investment: The security's initial investment amount.
- redemption: The amount to receive at the security's maturity date.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

IPMT

Description

Returns the interest payment during the specified period, based on a constant interest rate and regular payments.

Syntax

```
IPMT(rate, per, nper, pv, [ fv], [ type])
```

- rate: The interest rate for each period.
- per: The period to calculate the payment for.
- nper: The total number of periods for the loan or investment.
- pv: The present value.
- fv: The future value.
- type: The payment timing. 0 or Empty = the payment occurs at the end of the period; 1 = the payment occurs at the beginning of the period.

IRR

Description

Returns the internal rate of return for a specified series of periodic cash flows.

Syntax

```
IRR(values, [ guess])
```

- values: The array or reference to a range of cells containing the cash flow information.
- guess: The estimated rate of return. The default is 0.1 (10%).

ISBLANK

Description

If the specified value contains an empty string or a null value, returns TRUE; otherwise, returns FALSE.

Syntax

```
ISBLANK(value)
```

- value: The value to evaluate.

Example

Formula	Result
=ISBLANK(123)	FALSE
=ISBLANK(A1) where A1 is empty	TRUE

ISBOOLEAN

Description

Returns True if the value you specify is a boolean (logical) value; otherwise, returns False.

Syntax

`ISBOOLEAN(value)`

- value: The value to evaluate.

Example

Formula	Result
=ISBOOLEAN(FALSE)	TRUE
=ISBOOLEAN("phrase")	FALSE

Notes

- This function does the same action as ISLOGICAL().

ISERR

Description

If the specified value or expression results in an error, except for the #N/A error, returns TRUE; otherwise, returns FALSE.

Syntax

`ISERR(value)`

- value: The value or expression to evaluate.

Example

Formula	Result
=ISERR(123)	FALSE
=ISERR("phrase")	FALSE
=ISERR(12/0)	TRUE
=ISERR(A1) where A1 contains #N/A	FALSE

ISERROR

Description

If the specified value or expression results in an error, returns TRUE; otherwise, returns FALSE.

Syntax

`ISERROR(value)`

- value: The value or expression to evaluate.

Example

Formula	Result
=ISERROR(123)	FALSE
=ISERROR("phrase")	FALSE
=ISERROR(12/0)	TRUE
=ISERROR(A1) where A1 contains #N/A	TRUE

ISEVEN

Description

If the specified number is an even number, returns TRUE; otherwise, returns FALSE.

Syntax

`ISEVEN(number)`

- number: The number to evaluate.

Example

Formula	Result
=ISEVEN(4)	TRUE
=ISEVEN(50/10)	FALSE

ISFORMULA

Description

If the specified value is a formula, returns TRUE; otherwise, returns FALSE.

Syntax

`ISFORMULA(value)`

- value: The value to evaluate.

Example

Formula	Result
=ISFORMULA(A1) where A1 contains =5*5	TRUE
=ISFORMULA(A1) where A1 contains NOW()	TRUE
=ISFORMULA("phrase")	FALSE

ISLOGICAL

Description

If the specified value is a logical value, returns TRUE; otherwise, returns FALSE.

Syntax

`ISLOGICAL(value)`

- value: The value to evaluate.

Example

Formula	Result
=ISLOGICAL(FALSE)	TRUE
=ISLOGICAL("phrase")	FALSE

Notes

- This function does the same action as ISBOOLEAN().

ISNA

Description

If the specified value or expression results in an #N/A error, returns TRUE; otherwise, returns FALSE.

Syntax

`ISNA(value)`

- value: The value or expression to evaluate.

Example

Formula	Result
=ISNA(123)	FALSE
=ISNA("phrase")	FALSE
=ISNA(12/0)	FALSE
=ISNA(A1) where A1 contains #N/A	TRUE

ISNONTEXT

Description

If the specified value is not a text value, returns TRUE; otherwise, returns FALSE.

Syntax

`ISNONTEXT(value)`

- value: The value to evaluate.

Example

Formula	Result
=ISNONTEXT(4)	TRUE
=ISNONTEXT("phrase")	FALSE

ISNUMBER

Description

If the specified value is a number, returns TRUE; otherwise, returns FALSE.

Syntax

`ISNUMBER(value)`

- value: The value to evaluate.

Example

Formula	Result
=ISNUMBER(10)	TRUE
=ISNUMBER("text")	FALSE

ISO.CEILING

Description

Rounds a number up to the nearest integer, based on the multiple of significance that you specify. The function doesn't consider the number's sign. The function rounds positive numbers up, and also rounds negative numbers up (toward zero, becoming less negative).

Syntax

`ISO.CEILING(value, [ multiple])`

- value: The number to round up.
- multiple: The multiple of significance to round the number up to. If you omit the argument, the function uses the value 1, which causes the function to round the number up to the next larger integer.

Example

Formula	Result
=ISO.CEILING(12.34)	13
=ISO.CEILING(12.34,1)	13
=ISO.CEILING(-12.34,1)	-12
=ISO.CEILING(12.34,0.1)	12.4

Notes

- This function does the same action as ISOCEILING() and CEILING.PRECISE().

ISOCEILING

Description

Rounds a number up to the nearest integer, based on the specified significance. The function doesn't consider the number's sign. The function rounds positive numbers up, and also rounds negative numbers up (toward zero, becoming less negative).

Syntax

```
ISOCEILING(value, [ multiple])
```

- value: The number to round up.
- multiple: The multiple of significance to round the number up to. If you omit the argument, the function uses the value 1, which causes the function to round the number up to the next larger integer.

Example

Formula	Result
=ISOCEILING(12.34)	13
=ISOCEILING(12.34,1)	13
=ISOCEILING(-12.34,1)	-12
=ISOCEILING(12.34,0.1)	12.4

Notes

- This function does the same action as ISO.CEILING() and CEILING.PRECISE.

ISODD

Description

If the specified number is an odd number, returns TRUE; otherwise, returns FALSE.

Syntax

```
ISODD(number)
```

- number: The number to evaluate.

Example

Formula	Result
=ISODD(4)	FALSE
=ISODD(50/10)	TRUE

ISOMITTED

Description

Checks if a LAMBDA parameter has been omitted when the LAMBDA function is called.

This function is useful for building robust custom functions that can handle optional arguments.

Syntax

```
ISOMITTED(argument)
```

- argument: The value you want to test, such as a LAMBDA parameter.

ISOWEEKNUM

Description

Returns an integer from 1 to 53 representing the ISO week number of the specified date.

Each week of the International Organization for Standardization (ISO) date system starts on a Monday; week number 1 is the first week of the year that contains a Thursday. For more information, see Wikipedia ISO Week Date .

Syntax

```
ISOWEEKNUM(date)
```

- date: The date to get the week value from. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.

Example

Formula	Result
=ISOWEEKNUM("6-May-2016")	18
=ISOWEEKNUM("6-Dec-2016")	49

ISPMT

Description

Returns the interest payment during the specified period, based on a constant interest rate and regular payments.

Syntax

```
ISPMT(rate, per, nper, pv)
```

- rate: The interest rate for each period.
- per: The period to calculate the payment for.

- nper: The total number of periods for the loan or investment.
- pv: The present value.

ISREF

Description

If the specified value is a reference, returns TRUE; otherwise, returns FALSE.

Syntax

`ISREF(value)`

- value: The value to evaluate.

Example

Formula	Result
=ISREF(A1)	TRUE
=ISREF(A1+A2)	FALSE

ISTEXT

Description

If the specified value is text, returns TRUE; otherwise, returns FALSE.

Syntax

`ISTEXT(value)`

- value: The value to evaluate.

Example

Formula	Result
=ISTEXT(4)	FALSE
=ISTEXT("phrase")	TRUE

JOIN

Description

Performs an inner left join on two ranges.

Syntax

`JOIN(primary_table, key_column, join_range, join_key_column, [ has_headers], [ ignore_case])`

- primary_table: The first table to join. This range can be a matrix, range, or result of a data set transform operation.
- key_column: The column number in the primary_table expression that acts as the primary key for the join operation. The function numbers columns starting from 1.

- join_range: The second table to join. This range can be a matrix, range, or result of a data set transform operation.
- join_key_column: The column number in the join_range expression that acts as the key for the join operation. The function numbers columns starting from 1.
- has_headers: Not used.
- ignore_case: If TRUE, the function ignores the case of the text when comparing with other key values. If FALSE, the function considers case. The default is FALSE.

Example

This is the original workbook. The data to join is in two different sections:

	A	B	C	D
25	Salesperson	Customer	Product	Revenue
26	Jeffrey	Aberdeen Asset Management	HCM	$1,000,000
27	Jeffrey	Admiral Group	FIN	$500,000
28	Lei	Babcock International	HCM	$506,000
29	Lei	Barclays	LER	$1,500,000
30	Cian	Capita	LER	$1,895,000
31	Cian	Centrica	PAY	$2,004,560

	G	H	I
25	Salesperson	Manager	Region
26	Jeffrey	Peter	EMEA
27	Lei	Fredrik	US
28	Cian	John	UK

The formula =JOIN(A25:D31,1,G25:I28,1,,TRUE), placed in cell L4, combines the employee information with the manager and region information. The Salesperson, Manager, and Region are on the left because that data range has fewer rows.

	L	M	N	O	P	Q
4	Jeffrey	Peter	EMEA	Aberdeen Management	HCM	$1,000,000
5	Jeffrey	Peter	EMEA	Admiral Group	FIN	$500,000
6	Lei	Fredrik	US	Babcock International	HCM	$506,000
7	Lei	Fredrik	US	Barclays	LER	$1,500,000
8	Cian	John	UK	Capita	LER	$1,895,000
9	Cian	John	UK	Centrica	PAY	$2,004,560

Notes

- This function does an inner left join. The function considers the range with fewer rows to be the left table.
- For each row in the left table, the function checks to see if there is a matching row in the right table by comparing the key values in the key columns you specified. If a match exists, the function creates a new row by joining the row from each of the tables with the key value only appearing one time. A row appears in the output only if the join condition is satisfied. It is also possible for a row from the left table to appear multiple times if multiple rows in the right table have a matching key value.
- If you want all rows from the primary_table to always be present in the output, use the MERGEROWS function instead.
- When comparing a single instance value to a string that visually appears to be the same, the function evaluates them as equal. When comparing a single value in a multi-instance field to a string that visually appears to be the same, the formula evaluates them as *not* equal.
- This function is intended for use in array formulas.

Related Functions

CORRELATE

KURT

Description

Returns the kurtosis of a list of numbers.

Syntax

```
KURT(number1, [ number2], ...)
```

- number1: The list of numbers. The list must contain at least 4 values.
- number2: The list of numbers. The list must contain at least 4 values.

Example

Formula	Result
=KURT(1,7,6,4,9,2,3,8)	-1.596

LAMBDA

Description

Allows you to create custom, reusable functions, which can then be called by user-friendly names.

This function is useful for building functions for complex calculations and repetitive tasks thus eliminating the need to copy and paste formulas.

Syntax

```
LAMBDA(parameter1_or_calculation, [parameter2_or_calculation …])
```

- parameter_or_calculation: A value that you want to pass or the formula you want to execute.

LARGE

Description

Returns the kth largest value from the specified list of values and value of k.

Syntax

`LARGE(array, k)`

- array: The array of numbers.
- k: The index.

LCM

Description

Returns the least common multiple of the specified number values.

Syntax

`LCM(number1, [ number2], ...)`

- number1: The number or list of numbers.
- number2: The number or list of numbers.

Example

Formula	Result
=LCM(1, 4)	4
=LCM(3, 6, 27)	54

Notes

- The function ignores string values.

LEFT

Description

Returns the leftmost characters from a string, based on the specified text and number of characters.

Syntax

`LEFT(text, [ num_chars])`

- text: The text to return characters from.
- num_chars: The number of characters to return. The default is 1.

Example

Formula	Result
=LEFT(A1,8) Where cell A1 contains `Regional Sales Manager` .	Regional

LEN

Description

Returns the number of characters in the specified text.

Syntax

`LEN(text)`

- text: The text to evaluate.

Example

Formula	Result
=LEN(A1) Where cell A1 contains `Regional Sales Manager` .	22

LET

Description

The LET function is intended for use in larger workbooks where you want to minimize calculation time. LET allows you to create and use variables in your formulas, reducing the number of times a part of a formula is calculated..

Syntax

```
LET(name1, name_value1, calculation_or_name2, [ name_value2,
calculation_or_name3, ...])
```

- name1: The first variable name.
- name_value1: The value of the first variable. The value can be a cell, a range, or the result of a function or formula. Arrays are allowed.
- calculation_or_name2: The calculation to perform using the variable, or, a second variable name.
- name_value2: The value of the second variable. The value can be a cell, a range, or the result of a function or formula. Arrays are allowed.
- calculation_or_name3: The calculation to perform using the variable, or, a third variable name.
- name_value3: The value of the third variable. The value can be a cell, a range, or the result of a function or formula. Arrays are allowed. You can have more than 3 variables.

LINEST

Description

Returns information on the line that best fits the specified x values and y values.

Syntax

`LINEST(known_ys, [ known_xs], [ const], [ stats])`

- known_ys: The array of x values.
- known_xs: The array of y values.
- const: Specifies how to handle the constant b. FALSE or Empty = the value is zero; TRUE = treat it normally.

- stats: Specifies whether to return more regression information about the best fitting line. FALSE or Empty = do not return more information; TRUE = do return more information.

LN

Description

Returns the natural logarithm of the specified value.

Syntax

```
LN(value)
```

- value: The value to calculate the natural logarithm for.

Related Functions

E

LOG

Description

Returns the logarithm of the number to the specified base.

Syntax

```
LOG(number, [ base])
```

- number: The number to calculate the logarithm for.
- base: The base for the result. The default is 10 (base 10).

Related Functions

E

LOG1P

LOG10

Description

Returns the base-10 logarithm of the number.

Syntax

```
LOG10(value)
```

- value: The number to calculate the logarithm for.

Related Functions

LOG1P

LOG1P

Description

Returns the natural logarithm of the sum of 1 plus the argument.

Syntax

`LOG1P(value)`

- value: The value to use for the calculation.

Example

Formula	Result
=LOG1P(5)	1.791759469

LOGEST

Description

Returns the logarithmic regression, based on the specified x values and y values.

Syntax

`LOGEST(known_ys, [ known_xs], [ const], [ stats])`

- known_ys: The array of x values.
- known_xs: The array of y values.
- const: Specifies how to handle the constant b. FALSE or Empty = b is 1; TRUE = treat it normally.
- stats: Specifies whether to return more regression information. FALSE or Empty = do not return more information; TRUE = do return more information.

LOGINV

Description

Returns the inverse of the log-normal cumulative distribution function of x, where ln(x) is normally distributed with parameters mean and standard deviation.

Syntax

`LOGINV(probability, mean, standard_dev)`

- probability: The probability associated with the log-normal distribution.
- mean: The mean of ln(x).
- standard_dev: The standard deviation of ln(x).

Example

Formula	Result
=LOGINV(0.039084, 3.5, 1.2)	4.000025219

Notes

- This function does the same action as LOGNORM.INV() and LOGNORMINV().

LOGNORM.DIST

Description

Returns the log-normal distribution probability (cumulative or density) for the specified value.

Syntax

```
LOGNORM.DIST(x, mean, standard_dev, [ cumulative], ...)
```

- x: The value to calculate the log-normal distribution probability for.
- mean: The mean of the distribution.
- standard_dev: The standard deviation of the distribution.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

LOGNORM.INV

Description

Returns the inverse of the log-normal cumulative distribution function of x, where ln(x) is normally distributed with parameters mean and standard deviation.

Syntax

```
LOGNORM.INV(probability, mean, standard_dev)
```

- probability: The probability associated with the log-normal distribution.
- mean: The mean of ln(x).
- standard_dev: The standard deviation of ln(x).

Example

Formula	Result
=LOGNORM.INV(0.039084, 3.5, 1.2)	4.000025219

Notes

- This function does the same action as LOGNORMINV() and LOGINV().

LOGNORMDIST

Description

Returns the log-normal distribution probability (cumulative or density) for the specified value.

Syntax

```
LOGNORMDIST(x, mean, standard_dev, [ cumulative], ...)
```

- x: The value to calculate the log-normal distribution probability for.
- mean: The mean of the distribution.
- standard_dev: The standard deviation of the distribution.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

LOGNORMINV

Description

Returns the inverse of the log-normal cumulative distribution function of x, where ln(x) is normally distributed with parameters mean and standard deviation.

Syntax

`LOGNORMINV(probability, mean, standard_dev)`

- probability: The probability associated with the log-normal distribution.
- mean: The mean of ln(x).
- standard_dev: The standard deviation of ln(x).

Example

Formula	Result
=LOGNORMINV(0.039084, 3.5, 1.2)	4.000025219

Notes

- This function does the same action as LOGNORM.INV() and LOGINV().

LOOKUP

Description

This function looks for a value in a 1-dimensional list and returns the value from the same position in a second list.

Syntax

`LOOKUP(lookup_value, lookup_vector, [ result_vector])`

- lookup_value: The value to find.
- lookup_vector: The list to search in.
- result_vector: The list to return a value from. If not specified, the function uses the lookup_vector.

Example

The examples are based on this workbook:

	A	B	C
1		# Sold - Upper Limit	Level
2		199	Too Low
3		299	Bronze
4		399	Silver
5		1000	Gold
6	Geoff	388	

The result of `=LOOKUP(B6,B2:B5,C2:C5)` is `Bronze` .

Notes

- You must sort the lookup_vector in ascending order.
- If the LOOKUP function can't find the lookup_value, it returns the largest value in lookup_vector that is less than or equal to lookup_value.
- Use VLOOKUP or HLOOKUP for lookup operations on arrays.

Related Functions

ARRAYAREA

GROUPBY

MATCHEXACT

MVLOOKUP

LOWER

Description

Converts the specified text to lowercase.

Syntax

`LOWER(text)`

- text: The text to convert.

Example

Formula	Result
=LOWER(A1) Where cell A1 contains `Regional Sales Manager` .	regional sales manager

LT

Description

If number1 is less than number2, returns TRUE; otherwise, returns FALSE.

Syntax

`LT(number1, number2)`

- number1: The first number to compare.
- number2: The second number to compare.

Example

Formula	Result
=LT(A1,A2) Where cell A1 contains 2 and A2 contains `-2` .	FALSE

LTE

Description

Compares two numbers. If number1 is less than or equal to number2, returns TRUE; otherwise, returns FALSE.

Syntax

`LTE(number1, number2)`

- number1: The first number to compare.
- number2: The second number to compare.

Example

Formula	Result
=LT(A1,A2) Where cell A1 contains 2 and A2 contains 2 .	TRUE

MAKEARRAY

Description

Creates a calculated array of a specified size of rows and columns by applying a LAMBDA function.

This function allows for dynamic array creation based on complex calculations or patterns.

Syntax

`MAKEARRAY(rows, cols, lambda)`

- rows: The number of rows in the array. Must be greater than zero.
- cols: The number of columns in the array. Must be greater than zero.
- lambda: A LAMBDA that is called to create the array. There are two parameters:
  - row: The row index of the array.
  - col: The column index of the array.

MAP

Description

Applies a LAMBDA function to each value in an array(s) to create a new array of results.

This function is useful for transforming each element of an array based on a custom calculation.

Syntax

`MAP(array1, lambda_or_arrayN, …)`

- array1: The array to be mapped.
- lambda_or_arrayN: A LAMBDA which must be the last argument and which must have either a parameter for each array passed, or another array to be mapped.

MATCH

Description

Looks up a value in the specified list (one-dimensional array), and returns the position of the value. You can match numeric values, logical values, or text strings. With text strings you can match patterns: use an asterisk (*) to match any number of any characters; use a question mark (?) to match any single character.

Syntax

`MATCH(lookup_value, lookup_array, [ match_type])`

- lookup_value: The value to find.
- lookup_array: The array to search.
- match_type: Specifies the criteria to use when searching.
    - 1 or Empty = find the largest value less than or equal to lookup_value. lookup_array data must be in ascending order. The default is 1.
    - 0 = Find an exact match to lookup_value. With text strings you can use the asterisk (*) and question mark (?) characters to match patterns.
    - -1 = find the smallest value greater than or equal to lookup_value. lookup_array data must be in descending order.

Example

The examples are based on this workbook:

	A	B	C	D	E	F	G	H
1	Supervisory Org	Cost Center	Company	Location	Jan (HC) 2017	Feb (HC) 2017	Mar (HC) 2017	PEA KEY
2	Finance	CC-1 Finance	GMS	Pleasanton	2	2	2	FinanceCC-1 FinanceGMSPleasanton
3	Product Management	CC-2 Product	GMS	Boulder	0	1	2	Product ManagementCC-2 ProductGMSBoulder
4	Development	CC-3 Development	GMS-CA	Toronto	2	2	2	DevelopmentCC-3 DevelopmentGMS-CAToronto
5	Sales	CC-4 Sales	GMS-UK	London	1	2	3	SalesCC-4 SalesGMS-UKLondon
6	Sales	CC-4 Sales	GMS-UK	Toronto	0	0	1	SalesCC-4 SalesGMS-UKToronto
7					5	7	9	

The result of `=MATCH("CC-3 Development",B2:B6)` is `3` .

The result of `=MATCH("*Sales*",B2:B6,0)` is `4` .

You can combine the MATCH and INDEX functions to return the matched value instead of the position of the value.

For the formula:

=INDEX(G2:G6,MATCH("*Sales*",B2:B6,0))

First, MATCH searches the range B2:B6 for the string "*Sales*" (using a wildcard) and returns the position of the value, which is `4` . Then INDEX uses `4` as its second argument and returns the value in position 4 of G2:G6, which is `2` .

Notes

- If there are multiple matches, the function returns the position of the first one.
- Wildcard examples when finding text patterns:
  - Search for `*00` to find values that end with two zeros.
  - Search for `1?5` to find all three-digit values that begin with 1 and end with 5.
  - If the value to find contains a * or ?, precede it with a tilde (~). Example: search for `~*N/A~*` to find the text `*N/A*` .
- We recommend using the MATCHEXACT function instead of MATCH where possible, for better performance.

Related Functions

MATCHEXACT

MHLOOKUP

MVLOOKUP

MATCHCOMPOSITE

Description

A common use case for MATCHCOMPOSITE is to consolidate column data from different sheets. Example: If there are 2 sheets, where you have data on a sheet, along with notes about that same data in a column on a different sheet, you can use MATCHCOMPOSITE to consolidate the column data from those 2 sheets into 1 sheet. MATCHCOMPOSITE copies values in one or more columns from the location to the right of a source array, and returns values to the right of a destination array. You use a composite key of columns to match copied data to the correct rows in the destination.

Syntax

```
MATCHCOMPOSITE(destination_matrix, destination_column_indexes, source_matrix,
source_column_indexes, return_column_indexes, [ ifNA])
```

- destination_matrix: The area of data in the destination.
- destination_column_indexes: The position of the columns that make up the composite key in the destination.
- source_matrix: The array of the data in the source. You can specify only a cell range, not a column range.
- source_column_indexes: The position of the columns that make up the composite key in the source.
- return_column_indexes: The position of the columns in the source that you want the formula to return.
- ifNA: The default value to return if the function doesn't find a match. The default is an empty string.

Related Functions

MATCH

GROUPBY

MATCHEXACT

Description

Looks up an exact match for the value in the sorted list (one-dimensional array) you specify, and returns the position of the value. You can use this function to match logical values, numeric values, or text strings. This function is similar to MATCH, but MATCHEXACT:

- Always searches for an exact match; it returns an #N/A error if it doesn't find the value.
- Doesn't allow wildcard characters such as * or ?.
- Uses a binary search for better performance.

Syntax

`MATCHEXACT(lookup_value, lookup_array, [ match_type])`

- lookup_value: The value to find.
- lookup_array: The array to search.
- match_type: The criteria to use when searching. Possible values are:
  - 0 or greater = do a binary search for the specified lookup_value. lookup_array data must be sorted in ascending order. If you omit the argument, the function uses this as the argument value.
  - Less than 0 = do a binary search for the specified lookup_value. lookup_array data must be sorted in descending order.

Example

The examples are based on this workbook:

	A	B	C	D	E	F	G	H
1	Supervisory Org	Cost Center	Company	Location	Jan (HC) 2017	Feb (HC) 2017	Mar (HC) 2017	Organization KEY
2	Finance	CC-1 Finance	GMS	Pleasanton	2	2	2	FinanceCC-1 FinanceGMSPleasanton
3	Product Management	CC-2 Product	GMS	Boulder	0	1	2	Product ManagementCC-2 ProductGMSBoulder
4	Development	CC-3 Development	GMS-CA	Toronto	2	2	2	DevelopmentCC-3 DevelopmentGMS-CAToronto
5	Sales	CC-4 Sales	GMS-UK	London	1	2	2	SalesCC-4 SalesGMS-UKLondon
6	Sales	CC-4 Sales	GMS-UK	Toronto	0	0	1	SalesCC-4 SalesGMS-UKToronto
7					5	7	9	

For the formula `=MATCHEXACT("CC-3 Development",B2:B6)` , the result is `3` .

You can also return the matched value instead of its position by combining the MATCHEXACT and INDEX functions.

For the formula `=INDEX(G2:G6,MATCHEXACT("CC-3 Development",B2:B6,1))` :

First, MATCHEXACT searches the array B2:B6 for the string "CC-3 Development" and returns the position of the value, which is `3` . Then, INDEX uses `3` as its second argument and returns the value in position 3 of G2:G6, which is `2` .

Notes

- If multiple matches exist, the function returns the position of the first match.

Related Functions

MATCH

MHLOOKUP

MVLOOKUP

MAX

Description

Returns the maximum value from a list of values.

Syntax

```
MAX(number1, number2,...)
```

- number1: A value to test.
- number2: A value to test.

MAXA

Description

Returns the maximum value from a list of values. Similar to MAX(), but in MAXA() you can include text and logical values. A text value or logical FALSE value is treated as a 0; a logical TRUE is counted as a 1.

Syntax

```
MAXA(number1, [ number2], ...)
```

- number1: The list of number or date values.
- number2: The list of number or date values.

MAXIFS

Description

Returns the maximum value from a range of values, according to one or more criteria.

Syntax

```
MAXIFS(max_range, range, criteria, [ range2], [ criteria2], ...)
```

- max_range: The array of numeric values that you want to return the maximum value from, if the criteria are met.
- range: The values, or the range of cells, to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.
- range2: The second set of values, or the range of cells, to be evaluated according to the condition.
- criteria2: A second condition to use to evaluate the values.

Notes

- Each condition can be one of these:
  - A numeric value (integer, decimal, date, time, or logical value), such as 5, 12/5/2019, or TRUE.
  - A text string, such as "Name" or "November".
  - An expression, such as ">9" or "<>0".
  - In text-based criteria, you can use the ? wildcard to match any single character or the * wildcard to match any sequence of characters.
  - Whenever you use a text string or an expression as a condition, surround it with quotes.
  - The function isn't case-sensitive. Example: When evaluating the values in criteria_range against the criteria, the text strings "MONTH" and "month" is considered a match.

MDETERM

Description

Returns the determinant of an array.

Syntax

```
MDETERM(array)
```

- array: The array of numeric values.

MDURATION

Description

Returns the modified duration of a fixed interest security. For the Macaulay duration, use DURATION() or BONDDURATION().

Syntax

```
MDURATION(settlement, maturity, coupon, yield, frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- coupon: The security's annual coupon rate.
- yield: The security's annual yield.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

MEDIAN

Description

Returns the median of the specified numbers.

Syntax

```
MEDIAN(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

MERGECOLUMNS

Description

Merges columns by placing them side by side into a new range.

Syntax

```
MERGECOLUMNS(range1, [ range2], ...)
```

- range1: The first range.
- range2: The second range.

Example

Formula	Result
=MERGECOLUMNS({1,2;3,4}, {10,20,30;40,50,60})	1 2 10 20 30 3 4 40 50 60

Notes

- This function creates a new range by combining all the range arguments. It starts with the first range and then adds the second range on its right side, aligning the top of the new range with the resulting range. This continues for all subsequent ranges. In the resulting range, the number of rows equals the number of rows in the largest range. If a range does not have as many rows as the maximum range, those additional rows are filled with null values. The number of columns in the resulting range equals the sum of the columns in all of the ranges. Duplicate columns are not removed.
- This function is intended for use in array formulas.

Related Functions

REMOVECOLUMNS

MERGEROWS

Description

Merges rows by placing them one below the other into a new range.

Syntax

```
MERGEROWS(range1, [ range2], ...)
```

- range1: The first range.
- range2: The second range.

Example

Formula	Result
=MERGEROWS({1,2;3,4},{10,20,30;40,50,60})	1 2 3 4 10 20 30 40 50 60

Notes

- This function creates a new range by combining all the range arguments; it starts with the first range and then adds the second range underneath, left-aligning the new range with the resulting range. This continues for all subsequent ranges. In the resulting range, the number of columns equals the

widest range in the input. If a range does not have as many columns as the widest range, those additional columns are filled with null values. The number of rows in the resulting range equals the sum of the rows in all of the ranges. Duplicate rows are not removed.
- This function is intended for use in array formulas.

Related Functions

REMOVEROWS

MHLOOKUP

Description

We recommend this function as a replacement for HLOOKUP. MHLOOKUP performs a horizontal (row) lookup on a table and returns all matches. MHLOOKUP is similar to HLOOKUP, but:

- HLOOKUP scans only the top row for matches; in MHLOOKUP you specify the row to search.
- HLOOKUP stops after finding 1 match, and returns a single cell value; MHLOOKUP scans the entire lookup row for matches. Each match in that row results in a new column in the result. If you specify 4 return_row_index values, then each resulting column will have 4 rows.

Syntax

`MHLOOKUP(lookup_value, table_array, lookup_row_index, return_row_index, ...)`

- lookup_value: The value to match.
- table_array: The array or table to search.
- lookup_row_index: The row in the table to search (1-based).
- return_row_index: The row number to return results from. You can list any number of row numbers.

Example

The example is based on this workbook:

	A	B	C	D	E	F	G	H
1	Salesperson	Dixon	Dixon	Kelly	Kelly	Payne	Payne	Payne
2	Customer	O'Reilly, Auer, & Lind	Runolfsson and Steuber	Kuphal Group	Ernser Inc	Williamson Group	Ratke-Sanford	Leuschke and Sons
3	Product	Qosolex	Saoplus	Qosolex	Voltflarn	Singlflix	Qosolex	Qosolex
4	Revenue	$8,232,000.00	$4,853,000.00	$2,358,000.00	$2,064,000.00	$6,974,000.00	$8,433,000.00	$8,181,000.00

The result of `MHLOOKUP("Qosolex",A1:H4,3,1,2,4)` is:

Dixon	Kelly	Payne	Payne
O'Reilly, Auer, & Lind	Kuphal Group	Ratke-Sanford	Leuschke and Sons
$8,232,000.00	$2,358,000.00	$8,433,000.00	$8,181,000.00

Notes

- MHLOOKUP() scans the columns in the lookup_row_index row for lookup_value, then gathers values from the rows listed in return_row_index. Then it places each value into a row in the resulting matrix.
- This function is intended for use in array formulas.

Related Functions

MATCHEXACT

MVLOOKUP

MI.CONTAINS

Description

If the lookup value is equal to any of the instance descriptions in the instances value, the function returns TRUE; otherwise, it returns FALSE. The text must be an exact match. The function doesn't support wild cards.

Syntax

```
MI.CONTAINS(lookup_value, within_value)
```

- lookup_value: The value to find.
- within_value: The instances value to search.

MI.COUNT

Description

Returns the number of single instances inside a multi-instance value.

Syntax

```
MI.COUNT(instances)
```

- instances: The instances value.

MI.INDEX

Description

Returns the instance at a specific index in a multi-instance value.

Syntax

```
MI.INDEX(instances, [ index])
```

- instances: The instances value. This value can be an array.
- index: The index of the instance.

MID

Description

Returns the middle characters from a string, based on the specified text and number of characters.

Syntax

```
MID(text, start_num, num_chars)
```

- text: The text to return characters from.
- start_num: The position of the first character to return.
- num_chars: The number of characters to return.

Example

Formula	Result
=MID(A1,10,5) Where cell A1 contains `Regional Sales Manager` .	Sales

MIDENTITY

Description

Returns a mathematical identity matrix.

Syntax

`MIDENTITY(value)`

- value: The dimension of the resulting matrix.

Example

Formula	Result
=MIDENTITY(3)	`1 0 0 0 1 0 0 0 1`

Notes

- This function returns a mathematical identity matrix. In an identity matrix, all the values in the matrix are zero except for values in the diagonal (from upper left to lower right), which are set to one.
- This function is intended for use in array formulas.

MIN

Description

Returns the minimum value from a list of values.

Syntax

`MIN(number1, number2,...)`

- number1: A value to test.
- number2: A value to test.

MINA

Description

Returns the minimum value from a list of values. Similar to MIN(), but in MINA() you can include text and logical values. A text value or logical FALSE value is treated as a 0; a logical TRUE is counted as a 1.

Syntax

`MINA(number1, [ number2], ...)`

- number1: The number, or list of numbers.
- number2: The number, or list of numbers.

MINIFS

Description

Returns the minimum value from a range of values, according to one or more criteria.

Syntax

```
MINIFS(min_range, range, criteria, [ range2], [ criteria2], ...)
```

- min_range: The array of numeric values that you want to return the minimum value from, if the criteria are met.
- range: The values, or the range of cells, to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.
- range2: The second set of values, or the range of cells, to be evaluated according to the condition.
- criteria2: A second condition to use to evaluate the values.

Notes

- Each condition can be one of these:
  - A numeric value (integer, decimal, date, time, or logical value), such as 5, 12/5/2019, or TRUE.
  - A text string, such as "Name" or "November".
  - An expression, such as ">9" or "<>0".
  - In text-based criteria, you can use the ? wildcard to match any single character or the * wildcard to match any sequence of characters.
  - Whenever you use a text string or an expression as a condition, surround it with quotes.
  - The function isn't case-sensitive. Example: When evaluating the values in criteria_range against the criteria, the text strings "MONTH" and "month" is considered a match.

MINUS

Description

Returns all rows from a first range that do not appear in any of the other supplied ranges.

Syntax

```
MINUS(range1, [ range2], ...)
```

- range1: The range to subtract matching rows from.
- range2: The range to subtract from the base range. You can subtract any number of ranges from the start range.

Example

This is the original spreadsheet:

	A	B	C
1	Cost Center	Q1	Q2
2	6010:Benefits Expenses	6200:Marketing	6010:Benefits Expenses
3	6300:Office & Administrative	4000:Revenue	6100:Facilities Taxes

	A	B	C
4	6100:Facilities Taxes	5000:Cost of Sales	6300:Office & Administrative
5	6400:Legal & Service Fees	6870:Talent Acquisition	6500:Information Technology
6	6800:Travel & Entertainment	6000:Salaries and Wages	6700:Depreciation
7	6870:Talent Acquisition	6400:Legal & Service Fees	6300:Office & Administrative

The formula =MINUS(A2:A7,B2:B7,C2:C7), placed in cell A8, returns any values in the first range that are not present in subsequent ranges (cost centers that were not budgeted for in Q1 or Q2).

	A
8	6800:Travel & Entertainment

Notes

- The result of this function is the subset of rows from range1 that do not appear in any of the other supplied range arguments. A row appears in range1 if all values in the row from a subsequent range are identical (and in the same order) to values in range1.
- This function is intended for use in array formulas.

Related Functions

COMPARE

MINUTE

Description

Returns an integer from 0 to 59 representing only the minutes specified in the time.

Syntax

```
MINUTE(date)
```

- date: The time to get the minute value from.

Example

Formula	Result
=MINUTE("8/14/2016 05:30 PM")	30

MIRR

Description

Returns the modified internal rate of return for a specified series of periodic cash flows.

Syntax

```
MIRR(values, finance_rate, reinvest_rate)
```

- values: The array or reference to a range of cells containing the cash flow information.
- finance_rate: The interest rate for the money used in the cash flows.
- reinvest_rate: The interest rate for the reinvested cash flows.

Notes

- MIRR() results show all digits of precision. Microsoft® Excel® rounds results to nearest hundredth (percent).

MMULT

Description

Returns the matrix product of two specified arrays. The number of columns in array1 must be the same as the number of rows in array2.

Syntax

```
MMULT(array1, array2)
```

- array1: The first array of numeric values.
- array2: The second array of numeric values.

Example

The examples are based on this workbook:

	A	B	C
1	2	4	6
2	3	6	9
3	5	10	15

The result of `=MMULT(A1:A3,A3:C3)` is shown in this workbook:

10	20	30
15	30	45
25	50	75

The result of `=MMULT(A1:B3,A1:C2)` is shown in this workbook:

16	32	48
24	48	72
40	80	120

MOD

Description

Returns the modulo, which is the remainder when dividing the number by the divisor.

Syntax

```
MOD(dividend, divisor)
```

- dividend: The number to be divided.
- divisor: The number to divide the dividend by (divisor).

Example

Formula	Result
=MOD(9,3)	0
=MOD(9,2)	1

MODE

Description

Returns the most frequently occurring number in the specified values.

Syntax

```
MODE(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

MODE.MULT

Description

Returns an array of the modes in a list of specified numbers.

Syntax

```
MODE.MULT(number1, [ number2], ...)
```

- number1: The list of numbers.
- number2: The list of numbers.

MODE.SNGL

Description

Returns the most frequently occurring number in the specified values.

Syntax

```
MODE.SNGL(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

MONTH

Description

Returns an integer from 1 to 12 representing the month portion of a specified date.

Syntax

```
MONTH(date)
```

- date: The date to get the month value from. Workday recommends entering a serial date or using a cell reference to specify the date instead of entering the date, to ensure reliable results.

Example

Formula	Result
=MONTH("08/15/2016")	8

MONTHNAME

Description

Returns the name of the month for the specified number value.

Syntax

`MONTHNAME(month_num)`

- month_num: The month number. 1 = January, 2 = February, and so on.

Example

Formula	Result
=MONTHNAME("8")	August

MROUND

Description

Returns the rounded value of the number to the multiple that you specify. If the remainder of dividing the number by the multiple is greater than or equal to half of the multiple, MROUND rounds away from 0 (up if positive; down if negative).

Syntax

`MROUND(value, multiple)`

- value: The number to round.
- multiple: The multiple to round the number to.

Example

Formula	Result
=MROUND(12.34,1)	12
=MROUND(12.34,10)	10
=MROUND(12.34,2)	12
=MROUND(-12.34,-2)	-12
=MROUND(-12.34,-1)	-12

Notes

- value and multiple arguments must have the same sign.

MS.GROUPBY

Description

Summarizes data by grouping, aggregating, sorting, and filtering based on specified fields. This function is the same as WD.GROUPBY but *without* a style parameter.

It is useful for creating summary tables and performing aggregations similar to a pivot table but directly within a formula.

Syntax

```
MS.GROUPBY(row_fields, values, function, [field_headers], [total_depth],
[sort_order], [filter_array], [field_relationship])
```

- row_fields: The range of values to group.
- values: The values to aggregate.
- function: The calculation to run when aggregating values.
- field_headers: A number that specifies whether the data has headers and whether you want to include them in the results. Options include:
    - Omitted value (default): Headers are not shown.
    - 0: No headers.
    - 1: Yes, but don't show headers.
    - 2: No headers, but generate them.
    - 3: Yes, and show headers.
- total_depth: A number that specifies whether to show totals and subtotals. For subtotals, row_fields must have at least 2 columns. Options include:
    - Omitted value (default): Grand totals and, where possible, subtotals.
    - 0: No totals.
    - 1: Grand totals at the bottom.
    - 2: Grand totals and subtotals at the bottom.
    - -1: Grand totals at the top.
    - -2: Grand totals and subtotals at the top.
- sort_order: A number that indicates how rows should be sorted. Numbers correspond to columns in row_fields, followed by columns in values. A positive number sorts rows in ascending order. A negative number sorts rows in descending order.
- filter_array: A column-oriented array of booleans that indicates which rows should be included or excluded from the output.
- field_relationship: A number that specifies the relationship between columns when multiple columns are provided as row_fields and affects the sorting of the results. Options include:
    - 0: Hierarchy (default) - Sorts row_fields from left to right where the sorting of later columns takes into account the hierarchy of earlier columns.
    - 1: Table - Sorts each column independently.

MS.REGEXEXTRACT

Description

Extracts text from a string that matches a specified regular expression pattern.

This function is useful for parsing and extracting specific pieces of information from complex text data.

Syntax

```
MS.REGEXEXTRACT(text, pattern, [return_mode], [case_sensitivity])
```

- text: The text or cell reference containing the text you want to extract strings from.
- pattern: The regular expression that defines the pattern of text you want to extract.
- return_mode: A number that specifies what strings you want to extract. Options include:
    - 0 (default): Returns the first string that matches the pattern.
    - 1: Returns all strings that match the pattern as an array.
    - 2: Returns capturing groups from the first match as an array. Capturing groups are parts of a regex pattern enclosed in parentheses that allow you to return separate parts of a single match individually.
- case_sensitivity: A number that determines whether the match is case-sensitive. Options include:
    - 0 (default): Case-sensitive.
    - 1: Case-insensitive.

MS.REGEXREPLACE

Description

Replaces text within a string with new text based on a regular expression pattern.

This function is useful for cleaning, reformatting, or anonymizing text data based on patterns.

Syntax

```
REGEXREPLACE(text, pattern, replacement, [occurrence], [case_sensitivity])
```

- text: The text or cell reference containing the text you want to replace strings within.
- pattern: The regular expression that defines the pattern of text you want to replace.
- replacement: The text that will be inserted into the string to replace the matched pattern.
- occurrence: The instance of the pattern you want to replace. The default is 0 which replaces all instances.
- case_sensitivity: A number that determines whether the match is case-sensitive. Options include:
    - 0 (default): Case-sensitive.
    - 1: Case-insensitive.

MS.REGEXTEST

Description

Checks if text within a string matches a specified regular expression pattern and returns TRUE if there is a match and FALSE if there is not. This function is the same as REGEXTEST.

This function is useful for validating data against specific formatting rules or identifying strings that contain certain patterns.

Syntax

```
MS.REGEXTEST(text, pattern, [case_sensitivity])
```

- text: The text or cell reference containing the text you want to match against.
- pattern: The regular expression that defines the pattern of text you want to match.
- case_sensitivity: A number that determines whether the match is case-sensitive. Options include:
    - 0 (default): Case-sensitive.
    - 1: Case-insensitive.

MS.SORT

Description

Sorts an existing array and returns a new array. MS.SORT accepts one sort direction and sorts all columns you specified based on that direction. This function is intended for use with unconstrained arrays. To submit the formula, use the unconstrained keyboard shortcut Ctrl+Alt+Enter (Windows) or Command +Option+Enter (Mac).

Syntax

```
MS.SORT(array, [ sort_index], [ sort_order], [ by_col])
```

- array: The array to sort.
- sort_index: A number that specifies the column to sort by. The default is to sort by the first (left-most) column.
- sort_order: A number that specifies the sort order. Specify 1 to sort in ascending order or -1 to sort in descending order. The default is 1 (ascending).
- by_col: A logical value that specifies the desired sort direction. Specify FALSE to sort by row or TRUE to sort by column. The default is FALSE (by row).

Notes

To sort by more than one column, with a different sort direction for each column, use SORT2.

Related Functions

ARRAYAREA

SORT

SORT2

MS.UNIQUE

Description

Returns the unique values in a specified array. This function is intended for use with unconstrained arrays. To submit the formula, use the unconstrained keyboard shortcut Ctrl+Alt+Enter (Windows) or Command +Option+Enter (Mac). MS.UNIQUE is case-sensitive.

Syntax

```
MS.UNIQUE(array, [ by_col], [ exactly_once])
```

- array: The array to extract unique row or column values from.
- by_col: A logical value that specifies how to do the comparison. Specify TRUE to compare columns against each other and return the unique columns. Specify FALSE to compare rows against each other and return the unique rows. The default is FALSE.
- exactly_once: A logical value that specifies whether to return only rows or columns that occur exactly one time. Specify TRUE to return distinct rows or columns that occur exactly one time. Specify FALSE to return all distinct rows or columns. The default is FALSE.

Related Functions

UNIQUE

DISTINCTROWS

MS.UNIQUE2

Description

Returns the unique values in a specified array. This function is intended for use with unconstrained arrays. To submit the formula, use the unconstrained keyboard shortcut Ctrl+Alt+Enter (Windows) or Command +Option+Enter (Mac). MS.UNIQUE2 is case-insensitive.

Syntax

```
MS.UNIQUE2(array, [ by_col], [ exactly_once])
```

- array: The array to extract unique row or column values from.
- by_col: A logical value that specifies how to do the comparison. Specify TRUE to compare columns against each other and return the unique columns. Specify FALSE to compare rows against each other and return the unique rows. The default is FALSE.
- exactly_once: A logical value that specifies whether to return only rows or columns that occur exactly one time. Specify TRUE to return distinct rows or columns that occur exactly one time. Specify FALSE to return all distinct rows or columns. The default is FALSE.

Related Functions

UNIQUE2

DISTINCTROWS2

MULTIINST

Description

Create a multi-instance value from a comma-separated list of single instance values.

Syntax

```
MULTIINST(instance, ...)
```

- instance: The list of single instance values.

Example

Where cells A4, A5, and A6 contain single instance values, the formula:

=MULTIINST(A4,A5,A6)

creates a multi-instance value in the cell, containing the instances in cells A4-A6.

MULTINOMIAL

Description

Returns the multinomial coefficient of the specified values.

Syntax

```
MULTINOMIAL(number1, [ number2], ...)
```

- number1: The value, or list of values.
- number2: The value, or list of values.

MULTIPLY

Description

Multiplies the specified values. This function does not support arrays; if you want to multiply lists of numbers, use PRODUCT.

Syntax

```
MULTIPLY(number1, number2)
```

- number1: The value, or list of values.
- number2: The value, or list of values.

Example

Formula	Result
=MULTIPLY(A1,A2) Where cell A1 contains `4` and A2 contains `20.7` .	82.8

MUNIT

Description

Returns the unit matrix for the specified size.

Syntax

```
MUNIT(value)
```

- value: The integer value for the dimension of the matrix.

MVLOOKUP

Description

We recommend this function as a replacement for VLOOKUP, especially when you're working with live data. MVLOOKUP performs a vertical (column) lookup on a table and returns all matches. MVLOOKUP is similar to VLOOKUP, but:

- VLOOKUP scans only the left column for matches; in MVLOOKUP you specify the column to search.
- VLOOKUP stops after finding 1 match, and returns a single cell value; MVLOOKUP scans the entire lookup column for matches. Each match in that column results in a new row in the output. If you specify 4 return_column_index values, then each resulting row will have 4 columns.

Syntax

```
MVLOOKUP(lookup_value, table_array, lookup_column_index, order,
return_column_index, ...)
```

- lookup_value: The value to match.
- table_array: The array or table to search.
- lookup_column_index: The column in the table to search (1-based).

- order: The order for the search, based on the sort order of table_array.
    - 0: The default linear search, for unsorted data.
    - 1: Binary search, for data sorted in ascending order.
    - -1: Binary search, for data sorted in descending order.
- return_column_index: The column number to return results from. You can list any number of column numbers.

Example

The example is based on this workbook:

	A	B	C	D
1	Salesperson	Customer	Product	Revenue
2	Dixon	O'Reilly, Auer, & Lind	Qosolex	$8,232,000.00
3	Dixon	Runolfsson and Steuber	Saoplus	$4,853,000.00
4	Kelly	Kuphal Group	Qosolex	$2,358,000.00
5	Kelly	Ernser Inc	Voltflarn	$2,064,000.00
6	Payne	Williamson Group	Singlflix	$6,974,000.00
7	Payne	Ratke-Sanford	Qosolex	$8,433,000.00
8	Payne	Leuschke and Sons	Qosolex	$8,181,000.00
9	Wu	Dach-Halvorson	Singlflix	$2,361,000.00
10	Wu	Wisoky LLC	Bextain	$5,752,000.00
11	Wu	Stiedemann Grp	Saoplus	$3,987,000.00

The result of:

`=MVLOOKUP(A6,A2:D11,1,0,2,3,4)`

or alternatively, `=MVLOOKUP("Payne",A2:D11,1,0,2,3,4)`

is:

Williamson Group	Singlflix	$6,974,000.00
Ratke-Sanford	Qosolex	$8,433,000.00
Leuschke and Sons	Qosolex	$8,181,000.00

Notes

- MVLOOKUP() scans the rows in column lookup_column_index for lookup_value. The function then gathers values in the columns listed in return_column_index. Then it places each value into a column in the resulting matrix.
- This function is intended for use in array formulas.

Related Functions

MATCHEXACT

MHLOOKUP

N

Description

Returns a numeric value for the specified value, using these rules: dates are converted to serial numbers, the logical value True is converted to the value 1, numeric values remain the same, error values return the same error, and all other values (such as text or the logical value True) are coverted to the value 0.

Syntax

`N(type_text)`

- type_text: The value to convert to a number.

Example

Formula	Result
=N(A1) where A1 contains 12/31/2017	43100
=N(A1) where A1 contains 12-Dec-2017 13:55 PM	0
=N(10)	10

NA

Description

Returns the #N/A error.

Syntax

`NA()`

Example

Formula	Result
=NA()	#N/A

NE

Description

Compares two numbers. If number1 is not equal to number2, returns TRUE; otherwise, returns FALSE.

Syntax

`NE(number1, number2)`

- number1: The first number to compare.
- number2: The second number to compare.

Example

Formula	Result
=NE(A1,A2) Where cell A1 contains 4 and A2 contains 4 .	FALSE

NEGBINOM.DIST

Description

Returns the negative binomial distribution. Similar to BINOMDIST() and BINOM.DIST() but in NEGBINOM.DIST() the number of trials is variable and the number of successes is fixed.

Syntax

```
NEGBINOM.DIST(number_f, number_s, probability_s, [ cumulative])
```

- number_f: The number of failures.
- number_s: The required number of successes.
- probability_s: The probability of number_f failures before the required number of successes.
- cumulative: The probability type. FALSE = probability mass function; TRUE = cumulative distribution.

NEGBINOMDIST

Description

Returns the negative binomial distribution. Similar to BINOMDIST() and BINOM.DIST() but in NEGBINOM.DIST() the number of trials is variable and the number of successes is fixed.

Syntax

```
NEGBINOMDIST(number_f, number_s, probability_s, [ cumulative])
```

- number_f: The number of failures.
- number_s: The required number of successes.
- probability_s: The probability of success.
- cumulative: The type of distribution to use. FALSE = probability mass function; TRUE = cumulative distribution function.

NETWORKDAYS

Description

Returns the number of work days between two specified dates. The start_date and end_date are included in the result.

Syntax

```
NETWORKDAYS(start_date, end_date, [ holidays])
```

- start_date: The starting date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- end_date: The ending date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- holidays: The list of dates to exclude from the work day count.

Example

Formula	Result
=NETWORKDAYS("11/23/2017","12/31/2017")	30
=NETWORKDAYS("11/23/2017","12/31/2017", {"11/27/2016", "12/1/2016", "12/25/2016"})	27

NETWORKDAYS.INTL

Description

Returns the number of work days between two specified dates. The start_date and end_date are included in the result. Similar to NETWORKDAYS(), but in NETWORKDAYS.INTL() you can specify which days of the week are weekend days.

Syntax

```
NETWORKDAYS.INTL(start_date, end_date, [ weekend], [ holidays])
```

- start_date: The starting date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- end_date: The ending date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- weekend: The code specifying which days of the week are weekend days. The code can be numerical, as described in the table below, or a string, where a 0 represents a weekday and a 1 represents a weekend day. Example: "0000111" specifies that Friday, Saturday, and Sunday are weekend days.
- holidays: The list of dates to exclude from the work day count.

Code	Description
1	Saturday and Sunday
2	Sunday and Monday
3	Monday and Tuesday
4	Tuesday and Wednesday
5	Wednesday and Thursday
6	Thursday and Friday
7	Friday and Saturday
11	Sunday only
12	Monday only
13	Tuesday only
14	Wednesday only
15	Thursday only
16	Friday only
17	Saturday only

Example

Formula	Result
=NETWORKDAYS.INTL("11/20/2017","12/31/2017")	30
=NETWORKDAYS.INTL("11/20/2017","12/31/2017",1,{"11/23/2017", "11/24/2017", "12/25/2017"})	27
=NETWORKDAYS.INTL("11/20/2017","12/31/2017","0000100",{"11/23/2017", "11/24/2017", "12/25/2017"})	30

NOMINAL

Description

Returns the nominal interest rate based on the specified effective interest rate and the number of compounding periods for each year.

Syntax

```
NOMINAL(effect_rate, npery)
```

- effect_rate: The effective (net) interest rate.
- npery: The number of compounding periods for each year.

NORM.DIST

Description

Returns the normal distribution probability for the specified value.

Syntax

```
NORM.DIST(x, mean, standard_dev, cumulative)
```

- x: The number that you want the distribution for.
- mean: The arithmetic mean of the distribution.
- standard_dev: The standard deviation of the distribution.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

NORM.INV

Description

Returns the inverse normal distribution probability for the specified value.

Syntax

```
NORM.INV(probability, mean, standard_dev)
```

- probability: The value to calculate the inverse normal distribution probability for.
- mean: The mean of the distribution.
- standard_dev: The standard deviation of the distribution.

NORM.S.DIST

Description

Returns the standard normal distribution function for the specified value.

Syntax

```
NORM.S.DIST(x, [ cumulative])
```

- x: The value to calculate the standard normal distribution probability for.
- cumulative: The type of distribution to use. FALSE = probability mass function; TRUE = cumulative distribution function.

NORM.S.INV

Description

Returns the inverse standard distribution function for the specified value.

Syntax

```
NORM.S.INV(probability)
```

- probability: The value to calculate the inverse standard distribution probability for.

NORMDIST

Description

Returns the normal distribution probability for the specified value. This functions is the same as NORM.DIST.

Syntax

```
NORMDIST(x, mean, standard_dev, cumulative)
```

- x: The number that you want the distribution for.
- mean: The arithmetic mean of the distribution.
- standard_dev: The standard deviation of the distribution.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

NORMINV

Description

Returns the inverse normal distribution probability for the specified value.

Syntax

```
NORMINV(probability, mean, standard_dev)
```

- probability: The value to calculate the inverse normal distribution probability for.
- mean: The mean of the distribution.
- standard_dev: The standard deviation of the distribution.

NORMSDIST

Description

Returns the standard normal distribution function for the specified value.

Syntax

`NORMSDIST(x, [ cumulative])`

- x: The value to calculate the standard normal distribution probability for.
- cumulative: The type of distribution to use. FALSE = probability mass function; TRUE = cumulative distribution function.

NORMSINV

Description

Returns the inverse standard distribution function for the specified value.

Syntax

`NORMSINV(probability)`

- probability: The value to calculate the inverse standard distribution probability for.

NOT

Description

Returns the opposite logical value of the logical value that you specify.

Syntax

`NOT(value)`

- value: The value to return the opposite value for. Numeric values other than zero are treated as True (1); 0 is treated as False.

Example

Formula	Result
=NOT(0)	TRUE
=NOT(1*5)	FALSE

NOTIFYIF

Description

Sends up to 1,000 notifications if a value in the specified criterion changes to meet a condition. You can send a notification to a user whether or not they have access to the workbook. If the function exceeds the limit, notifications stop and the #ERROR indicator displays in the cell containing the function. When you hover the cursor over the cell, a message describes the error.

Syntax

```
NOTIFYIF(range, criteria, [ user_names], [ subject], [ message], [
send_on_each])
```

- range: The range to evaluate the criteria for.
- criteria: The string expression to evaluate against the range. The format is the same as functions such as SUMIF() and COUNTIF().
- user_names: The array or list of users to send the notification to. If you don't include this argument, the function uses the user_name of the current user (the user composing the function).
- subject: The text string to use as the subject of the Workday notification. If you include a subject, Workday adds a colon character at the end of the string.
- message: The text to use as the body of the notification. These HTML tags are supported: <br>, <b>, <i>, and <u>.
- send_on_each: If TRUE, the function sends a notification on each value in the range that matches the criterion (TRUE). If FALSE, the function sends a notification only if all the values in the range meet the criterion. The default is TRUE.

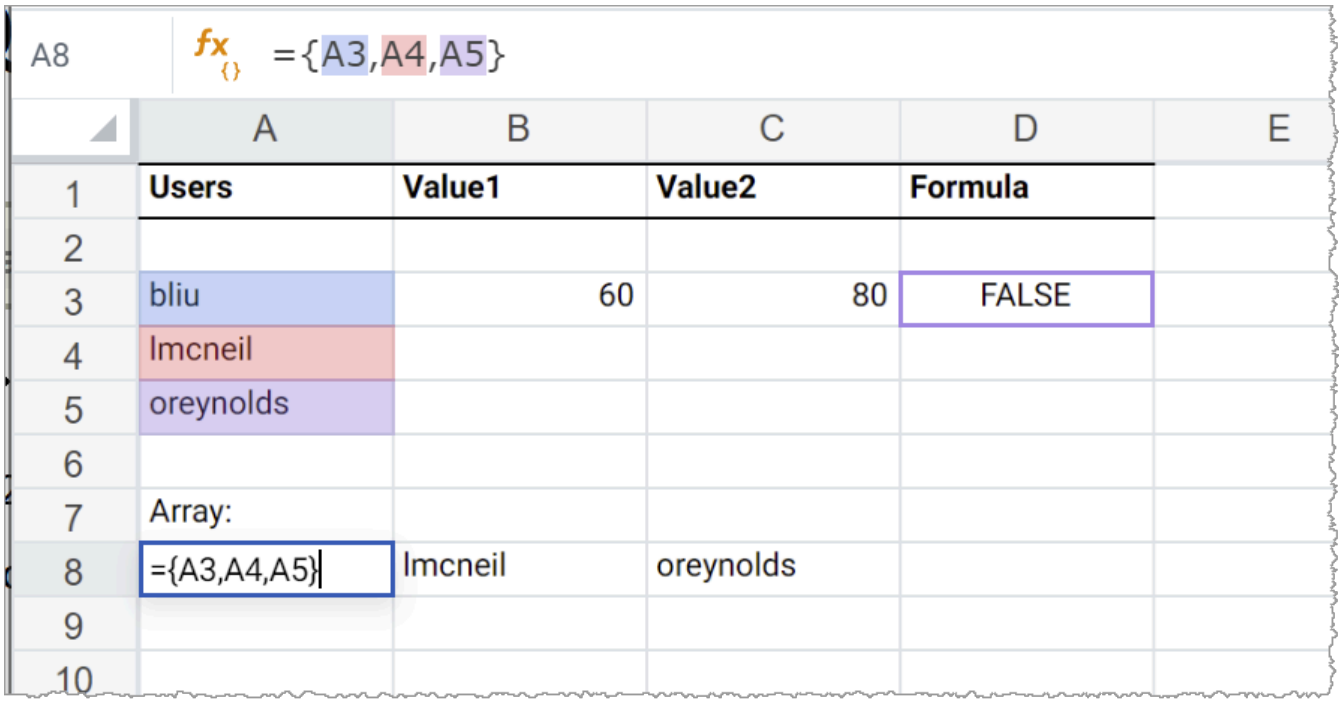
Example

Formula	Result
=NOTIFYIF(A1:A3,"=won","dave.smith","Contest results","you won!")	Send this notification to dave.smith: "Contest results: You won!" if a cell in the range A1:A3 equals won .
=NOTIFYIF(E8,"<"&E7,"tserrano","Forecast Drop",CONCAT(A8, " of $",E8," has fallen below target of $", E7))	Send a notification to tserrano if the value in E8 is less than E7. This example uses data in the sample workbook on Community. Example notification: "Forecast Drop: EMEA Forecast of $3214321 has fallen below target of $3530400"
=NOTIFYIF(A1,">"&0.5,"tserrano","Sales Goal",TEXT(A1,"0%")&" of your employees met their sales goal.")	Send a notification to tserrano if the value in A1 is greater than 0.5. The TEXT function causes the A1 value to be formatted as an integer percentage. Example notification: "Sales Goal: 61% of your employees met their sales goal.")

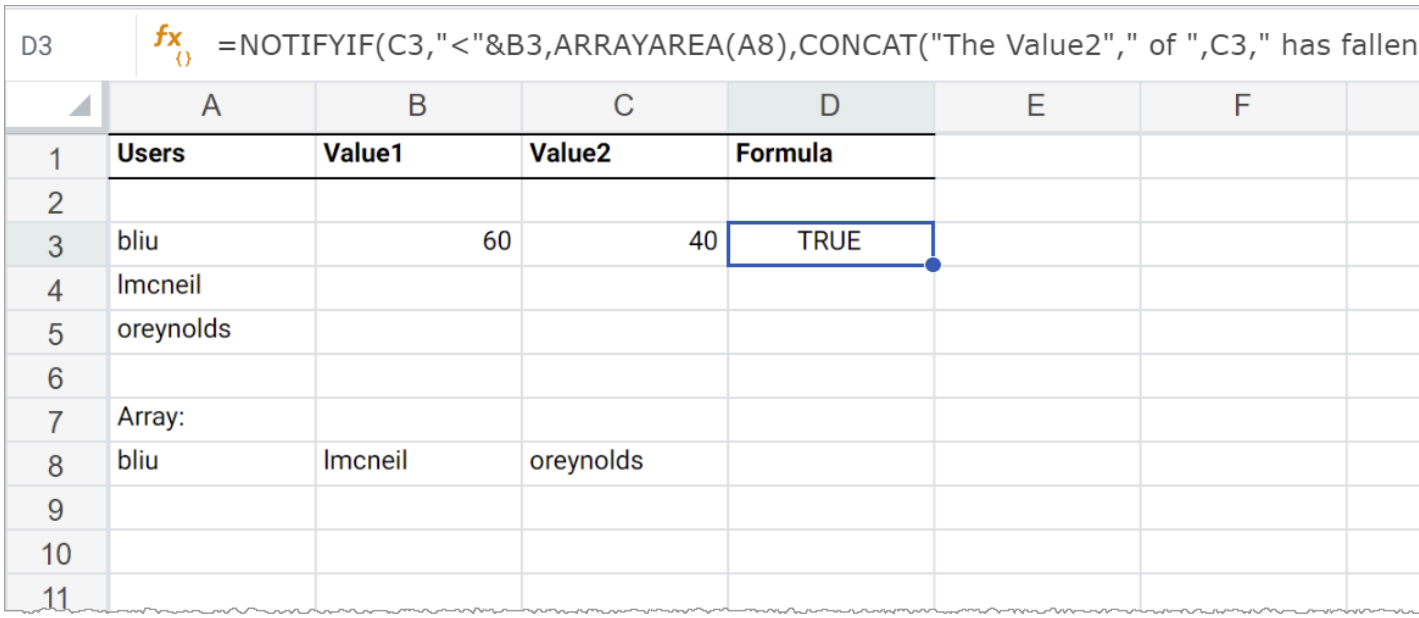
Example Notifications Using an Array of Values

Here's an example of how to use NOTIFYIF with an array of values.

The formula in cell A8 puts the different users listed in A3, A4 and A5 into an array:

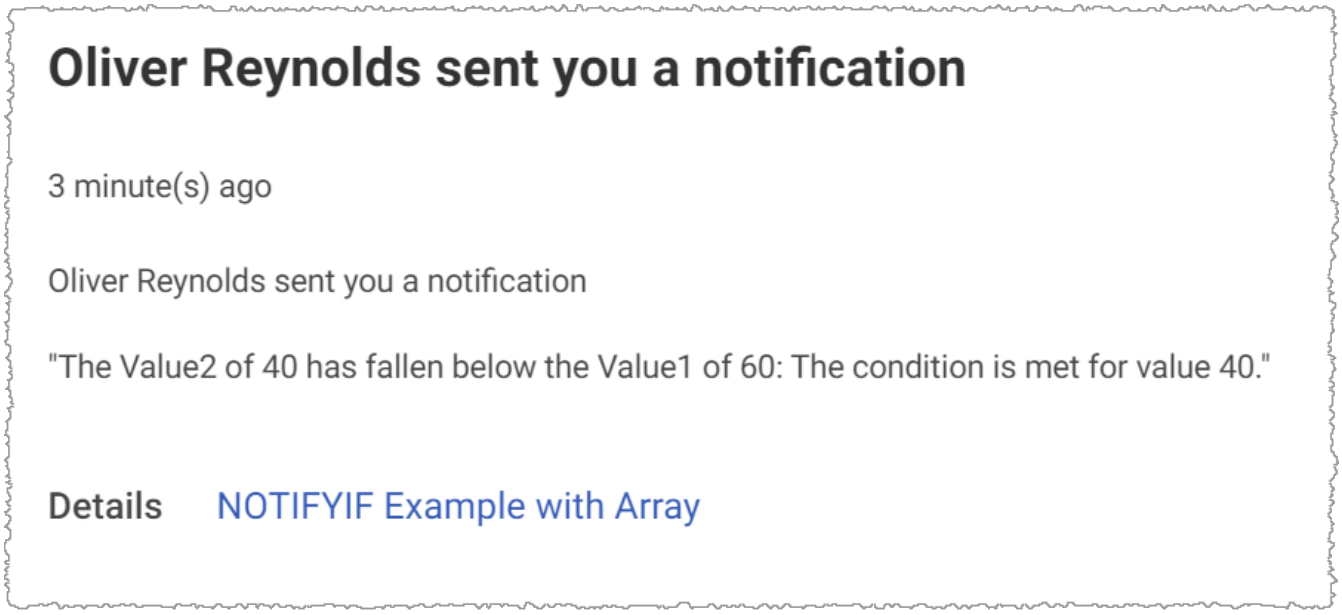


When including the reference cell in the NOTIFYIF formula, use ARRAYAREA(A8):

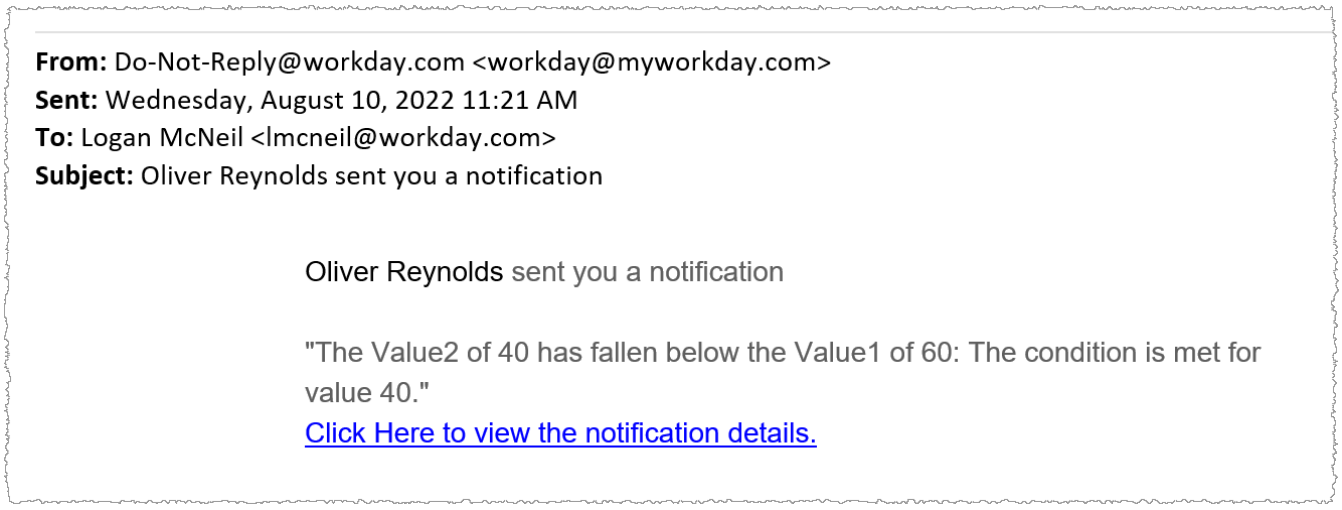


Remember to submit the formula using the array keyboard shortcut: Ctrl+Alt+Enter (Windows) or Command+Option+Enter (Mac).

The My Tasks notification for the formula looks similar to this, if Oliver Reynolds submitted the formula:



The email notification looks similar to this:



Notes

- The function sends a notification only if a value *changes* to meet a condition. If, for example, a live data refresh occurs and a condition that was previously TRUE (notification was sent) remains TRUE when the schedule runs, no notification is sent. However, a notification is sent if you recalculate the workbook because this is the equivalent of re-submitting the formula from the cell.
- The user_names format in your tenant might vary from the examples.
- This function evaluates a criterion expression against all the values in the range you specified. If send_on_each is true, then the function sends a notification for each criterion match in the range. If send_on_each is false, the function sends a notification only if all values in the match meet the criterion. The form of the criterion is the same as for SUMIF() and COUNTIF().
- The function generates a Workday notification. If emails are enabled in your environment, an email is also sent, but the email subject is auto-generated and doesn't match the subject argument of the notification.
- Workday identifies the sender of the notification as the user who placed the formula into the cell.
- This function returns the value TRUE if it sent a notification or FALSE if it didn't.
- Workday doesn't support using structured references with NOTIFYIF.

Related Functions

NOTIFYIFS

NOTIFYIFS

Description

Sends up to 1,000 notifications if values in the specified criteria change to meet multiple conditions. You can send a notification to a user whether or not they have access to the workbook. If the function exceeds the limit, notifications stop and the #ERROR indicator displays in the cell containing the function. When you hover the cursor over the cell, a message describes the error.

Syntax

```
NOTIFYIFS(range1, criteria1, [ user_names], [ subject], [ message], [
send_on_each], [ range2, criteria2], ...)
```

- range1: A range to evaluate the criteria for.
- criteria1: The string expression to evaluate against the range. The format is the same as functions such as SUMIF() and COUNTIF().
- user_names: The array or list of users to send the notification to. If you don't include this argument, the function uses the user_name of the current user (the user composing the function).
- subject: The text string to use as the subject of the Workday notification. If you include a subject, Workday adds a colon character at the end of the string.
- message: The text to use as the body of the notification. These HTML tags are supported: <br>, <b>, <i>, and <u>.
- send_on_each: If TRUE, a notification is sent on each value in the range that matches the criterion (TRUE). If FALSE, the notification is sent only if all the values in the range meet the criterion. The default is TRUE.
- range2: A range to evaluate the criteria for.
- criteria2: The string expression to evaluate against the range.

Example

A1:B3 contains:

```
1 2 1 2 1 2
```

Formula	Result
=NOTIFYIFS(A1:A3,1,,,,false,B1:B3,2)	TRUE

Notes

- The function sends a notification only if a value *changes* to meet a condition. If, for example, a live data refresh occurs and a condition that was previously TRUE (notification was sent) remains TRUE when the schedule runs, no notification is sent. However, a notification is sent if you recalculate the workbook because this is the equivalent of re-submitting the formula from the cell.
- This function evaluates a criterion expression against all the values in the range you specified. If send_on_each is true, then the function sends a notification for each criterion match in the range. If send_on_each is false, the function sends a notification only if all values in the match meet the criterion. The form of the criterion is the same as in SUMIFS() and COUNTIFS().
- The difference between NOTIFYIFS() and NOTIFYIF() is that in NOTIFYIFS(), the condition for sending a notification is met only if every criterion is met against its corresponding range. You can specify any number of range/criterion pairs; you must specify them as matching pairs. In addition, the shape of each range must match the shape of the first range (they must all have the same

number of rows and columns). Example: If we have three range/criterion pairs, then a condition is met if the criterion for each range is met for a specific cell in the range. If send_on_each is false, then the condition must be met for every cell in every range in order for the function to send a notification.
- The function generates a Workday notification. If emails are enabled in your environment, an email is also sent, but the email subject is auto-generated and doesn't match the subject argument of the notification.
- Workday identifies the sender of the notification as the user who placed the formula into the cell.
- The function returns the value TRUE if it sent a notification or FALSE if it didn't.
- Workday doesn't support using structured references with NOTIFYIFS.

Related Functions

NOTIFYIF

NOW

Description

Returns the current date and time as a time zone independent (UTC) value. This function is volatile; it generates a new value any time any cell in the workbook changes.

Syntax

```
NOW()
```

Example

Formula	Result
=NOW()	10/4/2016 5:37 PM (current time)

Notes

- The function doesn't run as a volatile function if the Worksheets calculation mode is set to Manual in File > Settings .
- The NOW() function returns the user's current time. The workbook time, shown when you select the Info icon on the Worksheets main page, shows the time based on the browser time (web) or the device time (mobile).

NOWTZ

Description

Returns the current date and time based on the time zone you specify. NOWTZ() generates a new date and time whenever any cell on the sheet changes. This function is volatile; it generates a new value any time any cell in the workbook changes.

Syntax

```
NOWTZ([ time_zone_id])
```

- time_zone_id: The time zone specifier, such as GMT, US/Central, and so on.

Example

Formula	Result
=NOWTZ("US/Pacific")	10/4/2016 5:37 PM

Notes

- The function doesn't run as a volatile function if the Worksheets calculation mode is set to Manual in File > Settings .
- Time zone IDs are based on the Java TimeZone utility. A complete list of time zones is available at http://joda-time.sourceforge.net/timezones.html. Examples:
  - America/New_York
  - Asia/Hong_Kong
  - Australia/Melbourne
  - Canada/Eastern
  - Europe/London
  - GMT
  - MET
  - US/Mountain
  - UTC

NPER

Description

Returns the number of periods required to pay off a loan, based on the specified periodic payment and interest rate.

Syntax

```
NPER(rate, pmt, pv, [ fv], [ type])
```

- rate: The interest rate for each period.
- pmt: The amount of the payment.
- pv: The present value.
- fv: The future value.
- type: The payment timing. 0 or Empty = the payment occurs at the end of the period; 1 = the payment occurs at the beginning of the period.

NPV

Description

Returns the net present value of an investment.

Syntax

```
NPV(rate, values, ...)
```

- rate: The discount rate for a single period.
- values: The numeric values specifying the series of regular payments and income. Negative values are payments; positive values are income.

OCT2BIN

Description

Returns the binary equivalent of the octal number, to the specified number of places.

Syntax

```
OCT2BIN(number, [ places])
```

- number: The octal number to convert.
- places: The number of digits to return, up to a limit of 10 digits. If the result is shorter than the specified number of digits, it is left-padded with zeroes.

OCT2DEC

Description

Returns the decimal equivalent of the octal number.

Syntax

```
OCT2DEC(number)
```

- number: The octal number to convert.

OCT2HEX

Description

Returns the hexadecimal equivalent of the octal number, to the specified number of places.

Syntax

```
OCT2HEX(number, [ places])
```

- number: The octal number to convert.
- places: The number of digits to return, up to a limit of 10 digits. If the result is shorter than the specified number of digits, it is left-padded with zeroes.

ODD

Description

Rounds a number to the nearest odd integer. Rounding is away from zero; the function rounds positive numbers up and negative numbers down.

Syntax

```
ODD(number)
```

- number: The number to round.

Example

Formula	Result
=ODD(12.34)	13

Formula	Result
=ODD(-12.34)	-13

ODDFYIELD

Description

Returns the yield per $100 face value, for a security that has a short or long first period.

Syntax

```
ODDFYIELD(settlement, maturity, issue, first_coupon, rate, price, redemption,
frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- issue: The security's issue date.
- first_coupon: The security's first coupon date.
- rate: The security's annual coupon rate.
- price: The security's price.
- redemption: The security's redemption value.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

ODDLPRICE

Description

Returns the price per $100 face value, for a security that has a short or long last period.

Syntax

```
ODDLPRICE(settlement, maturity, last_interest, rate, yield, redemption,
frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- last_interest: The security's last coupon date.
- rate: The security's annual coupon rate.
- yield: The security's annual yield.
- redemption: The security's redemption value.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

ODDLYIELD

Description

Returns the yield per $100 face value, for a security that has a short or long last period.

Syntax

```
ODDLYIELD(settlement, maturity, last_interest, rate, price, redemption,
frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- last_interest: The security's last coupon date.
- rate: The security's annual coupon rate.
- price: The security's price.
- redemption: The security's redemption value.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

OFFSET

Description

Returns a range of cells that is offset by a specified number of rows and columns from the original reference. This function is volatile; it generates a new value any time any cell in the workbook changes.

Syntax

```
OFFSET(reference, rows, cols, [ height], [ width])
```

- reference: The range of cells to return the offset for.
- rows: The number of rows to offset the range by.
- cols: The number of columns to offset the range by.
- height: The height of the range to return.
- width: The width of the range to return.

Notes

- The function doesn't run as a volatile function if the Worksheets calculation mode is set to Manual in File > Settings .
- If you don't include the height and width arguments, the function returns the same height and width as the reference.
- If you use OFFSET as a stand-alone formula and the result contains a range instead of a single cell, you must enter the function as an array formula.

OR

Description

Tests the specified conditions and returns True if any of the conditions are true; otherwise, returns False.

Syntax

```
OR(value1, value2,...)
```

- value1: A condition to test.
- value2: A condition to test.

Example

Formula	Result
=OR(1 > 0,5 < 10)	TRUE
=OR(1 > 2,5 > 10)	FALSE

PEARSON

Description

returns the Pearson correlation coefficient of the two specified data sets.

Syntax

`PEARSON(array1, array2)`

- array1: The array of independent values.
- array2: The array of dependent values.

PERCENTILE

Description

Returns the kth percentile for the specified list of values and value of k.

Syntax

`PERCENTILE(array, k)`

- array: The list of values to calculate the kth percentile for.
- k: The percentile value, where 0 <= k <= 1.

PERCENTILE.EXC

Description

Returns the kth percentile for the specified list of values and value of k.

Syntax

`PERCENTILE.EXC(array1, k)`

- array1: The list of values to calculate the kth percentile for. Similar to PERCENTILE() and PERCENTILE.INC(), but k must be in the range 0 to 1, exclusive.
- k: The percentile value, where 0 < k < 1.

PERCENTILE.INC

Description

Returns the kth percentile for the specified list of values and value of k.

Syntax

`PERCENTILE.INC(array, k)`

- array: The list of values to calculate the kth percentile for.

- k: The percentile value, where 0 <= k <= 1.

PERCENTOF

Description

Sums the values in the subset and divides it by all the values.

This function is useful for determining ratios and calculating percentage contributions within grouped data.

Syntax

```
PERCENTOF(data_subset,data_all)
```

- data_subset: The value that is in the data subset.
- data_all: The values that make up the entire set.

PERCENTRANK

Description

Returns the percentage rank of a value in a specified list of values.

Syntax

```
PERCENTRANK(number1, value, [ significance])
```

- number1: The list of values containing the value to rank.
- value: The value to calculate the percentage rank for.
- significance: The number of significant digits to round the result to.

PERCENTRANK.INC

Description

Returns the percentage rank of a value in a specified list of values.

Syntax

```
PERCENTRANK.INC(number1, value, [ significance])
```

- number1: The list of values containing the value to rank.
- value: The value to calculate the percentage rank for.
- significance: The number of significant digits to round the result to.

PERMUT

Description

Returns the number of permutations of the specified number of objects from a set. Similar to PERMUTATIONA() but PERMUT() does not allow repetitions.

Syntax

```
PERMUT(number, number_chosen)
```

- number: The number of objects in the set.
- number_chosen: The number of objects in each permutation.

PERMUTATIONA

Description

Returns the number of permutations of the specified number of objects from a set, with repetitions. Similar to PERMUT() but PERMUTATIONA() allows repetitions.

Syntax

`PERMUTATIONA(number, number_chosen)`

- number: The number of objects in the set.
- number_chosen: The number of objects in each permutation.

PHI

Description

Returns the values of the density function for a standard normal distribution.

Syntax

`PHI(x)`

- x: The number to calculate the density of the standard normal distribution for.

Example

Formula	Result
=PHI(1)	0.241970725
=PHI(4)	0.00013383

Notes

- The result of PHI(1.234E+201) or PHI(-1.234E+201) is zero. In Microsoft® Excel® the result is an *invalid number* error.

PI

Description

Returns the value of pi (approximately 3.141592654).

Syntax

`PI()`

PIVOTBY

Description

Summarizes data by grouping, aggregating, sorting, and filtering based on specified row and column fields. This function is the same as WD.PIVOTBY but *without* a style parameter.

This function is useful for dynamic cross-tabulation and advanced data summarization directly within a formula.

Syntax

```
PIVOTBY(row_fields, col_fields, values, function, [field_headers],
[row_total_depth], [row_sort_order], [col_total_depth], [col_sort_order],
[filter_array], [relative_to])
```

- row_fields: The values to use when grouping rows.
- col_fields: The values to use when grouping columns.
- values: The values to aggregate.
- function: The calculation to run when aggregating.
- field_headers: A number that specifies whether the data has headers and whether you want to include them in the results. Options include:
    - Omitted value (default): Headers are not shown.
    - 0: No headers.
    - 1: Yes, but don't show headers.
    - 2: No headers, but generate them.
    - 3: Yes, and show headers.
- row_total_depth: A number that specifies whether to show totals and subtotals. For subtotals, row_fields must have at least 2 columns. Options include:
    - Omitted value (default): Grand totals and, where possible, subtotals.
    - 0: No totals.
    - 1: Grand totals at the bottom.
    - 2: Grand totals and subtotals at the bottom.
    - -1: Grand totals at the top.
    - -2: Grand totals and subtotals at the top.
- row_sort_order: A number that indicates how rows should be sorted. Numbers correspond to columns in row_fields, followed by columns in values. A positive number sorts rows in ascending order. A negative number sorts rows in descending order.
- col_total_depth: A number that specifies whether to show totals and subtotals. For subtotals, col_fields must have at least 2 columns. Options include:
    - Omitted value (default): Grand totals and, where possible, subtotals.
    - 0: No totals.
    - 1: Grand totals on the right.
    - 2: Grand totals and subtotals on the right.
    - -1: Grand totals on the left.
    - -2: Grand totals and subtotals on the left.
- col_sort_order: A number that indicates how columns should be sorted. Numbers correspond to columns in col_fields, followed by columns in values. A positive number sorts rows in ascending order. A negative number sorts rows in descending order.
- filter_array: A column-oriented array of booleans that indicates which rows should be included or excluded from the output.
- relative_to: A number that indicates where to find values for the second argument of an aggregation function. This is typically used when PERCENTOF is supplied to function. Options include:
    - 0 (default): Column totals.
    - 1: Row totals.
    - 2: Grand total.
    - 3: Parent column total.
    - 4: Parent row total.

PMT

Description

Returns the payment required to pay off a loan or investment, based on a constant interest rate and for a specified period.

Syntax

```
PMT(rate, nper, pv, [ fv], [ type])
```

- rate: The interest rate for each period.
- nper: The total number of periods for the loan or investment.
- pv: The present value.
- fv: The future value.
- type: The payment timing. 0 or Empty = the payment occurs at the end of the period; 1 = the payment occurs at the beginning of the period.

POISSON

Description

Returns the poisson distribution probability for the specified value.

Syntax

```
POISSON(x, mean, cumulative)
```

- x: The number of events to calculate the probability for.
- mean: The mean of the distribution.
- cumulative: The type of distribution to use. FALSE = probability mass function; TRUE = cumulative distribution function.

Example

Formula	Result
=POISSON(15,25,TRUE)	0.022293
=POISSON(15,25,FALSE)	0.009891

Notes

- This function does the same action as POISSON.DIST().
- Worksheets supports only integer values of x.

POISSON.DIST

Description

Returns the poisson distribution probability for the specified value.

Syntax

```
POISSON.DIST(x, mean, cumulative)
```

- x: The number of events to calculate the probability for.
- mean: The mean of the distribution.

- cumulative: The type of distribution to use. FALSE = probability mass function; TRUE = cumulative distribution function.

Example

Formula	Result
=POISSON.DIST(15,25,TRUE)	0.022293
=POISSON.DIST(15,25,FALSE)	0.009891

Notes

- This function is a synonym of POISSON().
- Worksheets supports only integer values of x.

POW

Description

Returns a number raised to the specified power.

Syntax

`POW(number, power)`

- number: The number to be raised to a power.
- power: The exponent.

Example

Formula	Result
=POW(A1,A2) Where cell A1 contains 2 and A2 contains 3 .	8
=POW(A1,A2) Where cell A1 contains 2 and A2 contains -2 .	.25

Notes

- This function does the same action as POWER.

POWER

Description

Returns a number raised to the specified power.

Syntax

`POWER(number, power)`

- number: The number to be raised to a power.
- power: The exponent.

Example

Formula	Result
=POWER(A1,A2) Where cell A1 contains 2 and A2 contains 3 .	8
=POWER(A1,A2) Where cell A1 contains 2 and A2 contains -2 .	.25

Notes

- This function does the same action as POW.

PPMT

Description

Returns the payment on the principal for a specific period, based on a constant interest rate and constant periodic payments.

Syntax

```
PPMT(rate, per, nper, pv, [ fv], [ type])
```

- rate: The interest rate for each period.
- per: The period to calculate the payment for.
- nper: The total number of periods for the loan or investment.
- pv: The present value.
- fv: The future value.
- type: The payment timing. 0 or Empty = the payment occurs at the end of the period; 1 = the payment occurs at the beginning of the period.

PRICE

Description

Returns the price per $100 face value, for a security paying periodic interest.

Syntax

```
PRICE(settlement, maturity, rate, yield, redemption, frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- rate: The security's annual coupon rate.
- yield: The security's annual yield.
- redemption: The security's redemption value.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

PRICEDISC

Description

Returns the price per $100 face value, for a discounted security paying periodic interest.

Syntax

```
PRICEDISC(settlement, maturity, discount, redemption, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- discount: The security's discount rate.
- redemption: The security's redemption value.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

PRICEMAT

Description

Returns the price per $100 face value, for a security paying interest at maturity.

Syntax

```
PRICEMAT(settlement, maturity, issue, rate, yield, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- issue: The security's issue date.
- rate: The security's interest rate as of the issue date.
- yield: The security's annual yield.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

PRODUCT

Description

Returns the product of a list of values.

Syntax

```
PRODUCT(args, ...)
```

- args: The values or range of cells to multiply.

Example

Formula	Result
=PRODUCT(A1,A2) Where cell A1 contains `4` and A2 contains `20.7` .	82.8
=PRODUCT(A1:A3,B1:B3)	288

Formula	Result
Where cell A1:A3 contains `1 2 3` and B1:B3 contains `2 4 6`.	

PROPER

Description

Capitalizes the first letter of each word in the specified text.

Syntax

`PROPER(text)`

- text: The text to convert.

Example

Formula	Result
=PROPER(A1) Where cell A1 contains `6000:salaries and wages`.	6000:Salaries And Wages

PV

Description

Returns the present value of an investment based on future payments and a constant interest rate.

Syntax

`PV(rate, nper, pmt, [ fv], [ type])`

- rate: The interest rate for each period.
- nper: The number of periods over the life of the investment.
- pmt: The payment made for each period.
- fv: The future value of the investment after nper payments.
- type: The payment timing. 0 or Empty = the payment occurs at the end of the period; 1 = the payment occurs at the beginning of the period.

QUARTILE

Description

Returns the quartile of the specified list of values. The specified values must be numeric; if a unit is included as part of a number, the function ignores it.

Syntax

`QUARTILE(array, quart)`

- array: The list of values to calculate the quartile for.
- quart: The quartile to calculate. 1 = 25th percentile; 2 = 50th percentile; 3 = 75th percentile; 4 = 100th percentile (the function returns the maximum value). Specifying 0 causes the function to return the minimum value.

Example

Formula	Result
=QUARTILE({55,91,11},2)	55

QUARTILE.EXC

Description

Returns the quartile of the specified list of values. Similar to QUARTILE() or QUARTILE.INC() but QUARTILE.EXC() does not calculate the 0th or 4th quartile. The specified values must be numeric; if a unit is included as part of a number, the function ignores it.

Syntax

```
QUARTILE.EXC(array, quart)
```

- array: The list of values to calculate the quartile for.
- quart: The quartile to calculate. 1 = 25th percentile; 2 = 50th percentile; 3 = 75th percentile.

Example

Formula	Result
=QUARTILE.EXC({55,91,11},2)	55

QUARTILE.INC

Description

Returns the quartile of the specified list of values. The specified values must be numeric; if a unit is included as part of a number, the function ignores it.

Syntax

```
QUARTILE.INC(array, quart)
```

- array: The list of values to calculate the quartile for.
- quart: The quartile to calculate. 1 = 25th percentile; 2 = 50th percentile; 3 = 75th percentile; 4 = 100th percentile. Specifying 0 returns 0 as the result.

Example

Formula	Result
=QUARTILE.INC({55,91,11},2)	55

QUOTIENT

Description

Returns the integer portion of a division operation.

Syntax

```
QUOTIENT(numerator, denominator)
```

- numerator: The number to be divided (dividend).
- denominator: The number to divide the numerator by (divisor).

Example

Formula	Result
=QUOTIENT(A1,A2) Where cell A1 contains `42` and A2 contains `20` .	2

Notes

- If you want to obtain the full result rather than only the integer portion, use DIVIDE.

RADIANS

Description

Converts degrees to radians for the specified angle.

Syntax

`RADIANS(value)`

- value: The angle to convert.

RAND

Description

Returns a random number between 0 and 1 by default, or between different limits that you specify. This function is volatile; it generates a new random number any time any cell changes in the workbook. To generate a random number only when you cause the formula to run, such as by selecting Data > Recalculate , use RANDCONST().

Syntax

`RAND()`

Example

Formula	Result
=RAND()	varies (between 0 and 1)
=50*RAND()	varies (between 0 and 50)
=RAND()*(50-10)+10	varies (between 10 and 50)

Notes

- The function doesn't run as a volatile function if the Worksheets calculation mode is set to Manual in File > Settings .
- If you need one-time-only calculation, for any formula, enclose it in the ONCE function.

Related Functions

ONCE

RANDCONST

RANDARRAY

Description

Returns an array of random numbers. This function is intended for use with unconstrained arrays. To submit the formula, use the unconstrained keyboard shortcut Ctrl+Alt+Enter (Windows) or Command +Option+Enter (Mac).

Syntax

```
RANDARRAY([ rows], [ columns], [ min], [ max],[ whole_number])
```

- rows: The number of rows to return. If you don't specify a value, the function returns a single decimal value between 0 and 1.
- columns: The number of columns to return. If you don't specify a value, the function returns a single decimal value between 0 and 1.
- min: The lowest possible random number to return. This number is inclusive. (The value might be returned as a result.) If you omit the argument, the function uses 0 as the value.
- max: The highest possible random number to return. This number is inclusive. (The value might be returned as a result.) If you omit the argument, the function uses 1 as the value.
- whole_number: A logical value that specifies whether to return whole numbers or decimal values. The default is FALSE (decimal value).

Related Functions

RAND

RANDCONST

SEQUENCE

RANDBETWEEN

Description

Returns a random integer between the minimum and maximum values that you specify. This function is volatile; it generates a new random number any time any cell changes in the workbook. To generate a random number only when you cause the formula to run, such as by selecting Data > Recalculate , use RANDCONST().

Syntax

```
RANDBETWEEN(min, max)
```

- min: The minimum value.
- max: The maximum value.

Example

Formula	Result
=RANDBETWEEN(1,5)	varies (between 1 and 5)

Notes

- The function doesn't run as a volatile function if the Worksheets calculation mode is set to Manual in File > Settings .
- If you need one-time-only calculation, for any formula, enclose it in the ONCE function.

Related Functions

ONCE

RANDCONST

RANDCONST

Description

Returns a random floating point number. This function is similar to RAND() except that RANDCONST() generates a random floating point number only when you make the function run (for example, you select Data > Recalculate).

Syntax

```
RANDCONST([ low_bound], [ high_bound])
```

- low_bound: The low bound on the generated random number. This number is inclusive (the value might be returned as a result). If you omit the argument, the function uses 0 as the value.
- high_bound: The high bound on the generated random number. This number is inclusive (the value might be returned as a result). If you omit the argument, the function uses 1 as the value.

Example

Formula	Result
=RANDCONST()	varies (between 0 and 1)
=50*RANDCONST()	varies (between 0 and 50)
=RANDCONST()*(50-10)+10	varies (between 10 and 50)

Notes

- If you need one-time-only calculation, for any formula, enclose it in the ONCE function.
- This non-volatile function is recalculated only when:
  - A scheduled live data refresh runs.
  - You select Data > Recalculate .
  - You select Data > Recalculate All

Related Functions

ONCE

RANK

Description

Returns the rank of a number in a list of numbers. Similar to RANKAVG() but if there are duplicate instances of the number to be ranked, RANK() returns the lower rank.

Syntax

```
RANK(number, ref, [ order])
```

- number: The number to determine the rank for.
- ref: The list of numbers to use in determining the rank.
- order: The sort order. 0 (the default) = descending; 1 = ascending.

RANK.AVG

Description

Returns the rank of a number in a list of numbers. Similar to RANK.EQ() but if there are duplicate instances of the number to be ranked, RANK.AVG() returns the average rank.

Syntax

```
RANK.AVG(number, ref, [ order])
```

- number: The number to determine the rank for.
- ref: The list of numbers to use in determining the rank.
- order: The sort order. 0 (the default) = descending; 1 = ascending.

RANK.EQ

Description

Returns the rank of a number in a list of numbers. Similar to RANK.AVG() but if there are duplicate instances of the number to be ranked, RANK.EQ() returns the lower rank.

Syntax

```
RANK.EQ(number, ref, [ order])
```

- number: The number to determine the rank for.
- ref: The list of numbers to use in determining the rank.
- order: The sort order. 0 (the default) = descending; 1 = ascending.

RANKAVG

Description

Returns the rank of a number in a list of numbers. Similar to RANKEQ() but if there are duplicate instances of the number to be ranked, RANKAVG() returns the average rank.

Syntax

```
RANKAVG(number, ref, [ order])
```

- number: The number to determine the rank for.
- ref: The list of numbers to use in determining the rank.
- order: The sort order. 0 (the default) = descending; 1 = ascending.

RANKEQ

Description

Returns the rank of a number in a list of numbers. Similar to RANKAVG() but if there are duplicate instances of the number to be ranked, RANKEQ() returns the lower rank.

Syntax

```
RANKEQ(number, ref, [ order])
```

- number: The number to determine the rank for.
- ref: The list of numbers to use in determining the rank.
- order: The sort order. 0 (the default) = descending; 1 = ascending.

RATE

Description

Returns the interest rate for an investment or loan for the specified period.

Syntax

```
RATE(nper, pmt, pv, [ fv], [ type], [ guess])
```

- nper: The total number of periods for the loan or investment.
- pmt: The payment amount for each period.
- pv: The present value.
- fv: The future value.
- type: The payment timing. 0 = the payment occurs at the end of the period; 1 = the payment occurs at the beginning of the period.
- guess: The estimated rate. The default is 0.1 (10%).

RECEIVED

Description

Returns the amount received at maturity, for a fully invested security.

Syntax

```
RECEIVED(settlement, maturity, investment, discount, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- investment: The amount invested.
- discount: The discount rate.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

REDUCE

Description

Reduces an array to an accumulated value by applying a LAMBDA to each value and returning the value in the accumulator.

This function is useful for aggregations and final calculations, such as finding a total or maximum value from an array.

Syntax

```
REDUCE(initial_value, array, lambda)
```

- initial_value: Sets the starting value for the accumulator.

- array: The array to be reduced.
- lambda: The LAMBDA that defines the operation to reduce the array. There are three parameters:
  - accumulator: The value returned as the final result.
  - value: The current value from the array.
  - body: The calculation applied to each element in the array.

REGEXEXTRACT

Description

Returns a substring extracted from another string, based on the specified regular expression.

Syntax

`REGEXEXTRACT(text, regex, [ ignore_case])`

- text: The text to search.
- regex: The regular expression.
- ignore_case: If TRUE, the function ignores case when searching. If FALSE, the function considers case. The default is FALSE.

Example

Formula	Result
=REGEXEXTRACT(A1, "\D{3}\d+") Where cell A1 contains `Find the DPT231 cost center value in a comment.` .	DPT231
=REGEXEXTRACT(A1, "\(([A-Za-z]+)\)") Where cell A1 contains `Check total (monthly)` .	(monthly)
=REGEXEXTRACT(A1,"[0-9]+") Where cell A1 contains `6000:Salaries and Wages` .	6000
=REGEXEXTRACT(A1,"[0-9]*\.[0-9]+[0-9]+") Where cell A1 contains `Total for 2017 is $2451.00` .	2451.00

Related Functions

REGEXFIND

REGEXPARSE

REGEXFIND

Description

Returns the position of the first character of a substring that matches the regular expression pattern. The position value is zero-based. If the function doesn't find a match, it returns the #N/A error.

Syntax

`REGEXFIND(text, regex, [ ignore_case])`

- text: The text to search.
- regex: The regular expression.
- ignore_case: If TRUE, the function ignores the case of the text. If FALSE, the function considers case. The default is FALSE.

Example

Formula	Result
=REGEXFIND(A1,"el+") Where cell A1 contains `hello world` .	1

Notes

- Workday uses Java's regular expression parser; the expression must conform to Java's regex rules (http://docs.oracle.com/javase/8/docs/api/java/util/regex/Pattern.html) .

REGEXMATCH

Description

Evaluates whether a substring exists in a string, based on the specified regular expression. If the function finds the substring, it returns TRUE; otherwise, it returns FALSE.

Syntax

`REGEXMATCH(text, regex, [ ignore_case])`

- text: The text to search.
- regex: The regular expression.
- ignore_case: If TRUE, the function ignores case when searching. If FALSE, the function considers case. The default is FALSE.

Example

Formula	Result	Notes
=REGEXMATCH("hello world", "el+")	TRUE	Does the text contain the string "el"?
=REGEXMATCH("12-OCT-2015", "\d\d-[A-Z]{3}-\d{4}")	TRUE	Is the date formatted correctly?
=REGEXMATCH("7", "^\d+$")	TRUE	Is the integer positive?
=REGEXMATCH("-7", "^-\d+$")	TRUE	Is the integer negative?

Related Functions

REGEXFIND

REGEXPARSE

REGEXPARSE

Description

Extracts parts of a string by matching to a pattern. The regex pattern must match the *entire* input value.

Syntax

`REGEXPARSE(text, regex, [ ignore_case])`

- text: The text to parse.
- regex: The regular expression.
- ignore_case: If TRUE, the function ignores case when searching. If FALSE, the function considers case. The default is FALSE.

Example

Formula	Result
=REGEXPARSE("[Los Angeles]","\[(.*)\]")	Los Angeles
=REGEXPARSE("12-OCT-2015","(\d\d)-([A-Z]{3})-(\d{2,4})")	`12-OCT-2015 12 OCT 2015`
=INDEX(REGEXPARSE("aa;bbbbbb24;cccc;","(.*?);(.*?);.*"),3) Extracts the nth part of a delimited string	bbbbbb24

Notes

- This function returns an array of values where each value is part of the string where it found a pattern match. This function is useful if you have a pattern and you want to pull out the parts of the text that matched the pattern.
- This function is intended for use in array formulas.
- Workday uses Java's regular expression parser; the expression must conform to Java's regex rules (http://docs.oracle.com/javase/8/docs/api/java/util/regex/Pattern.html) .

REGEXREPLACE

Description

Replaces parts of a string by matching to a pattern.

Syntax

`REGEXREPLACE(text, regex, replacement, [ ignore_case])`

- text: The text to search.
- regex: The regular expression.
- replacement: The text to insert.
- ignore_case: If TRUE, the function ignores case when searching. If FALSE, the function considers case. The default is FALSE.

Example

Formula	Result
=REGEXREPLACE(A1,"^0$"," ") Where cell A1 contains `0` .	(Blank in cell)
=REGEXREPLACE(A1,"\n"," ") Where cell A1 contains: Text with a line break	Text with a line break
=REGEXREPLACE(A1, "\(([A-Za-z]+)\)", "(yearly)") Where cell A1 contains `Check total (monthly)` .	Check total (yearly)
=REGEXREPLACE(A1,"[0-9]+","5250") Where cell A1 contains `6000:Salaries and Wages` .	5250:salaries and wages
=REGEXREPLACE(A1,"[0-9]*\.[0-9]+[0-9]+","2500.00") Where cell A1 contains `Total for 2017 is $2451.00` .	Total for 2017 is $2500.00

Related Functions

REGEXFIND

REGEXPARSE

REGEXTEST

Description

Checks if text within a string matches a specified regular expression pattern and returns TRUE if there is a match and FALSE if there is not. This function works the same as MS.REGEXTEST.

This function is useful for validating data against specific formatting rules or identifying strings that contain certain patterns.

Syntax

```
REGEXTEST(text, regex, [ignore_case])
```

- text: The text or cell reference containing the text you want to match against.
- regex: The regular expression that defines the pattern of text you want to match.
- ignore_case: If TRUE, the case of the text is ignored. If FALSE (default), the case is considered.

REMOVECOLUMNS

Description

Removes one or more columns from the sheet. The function removes the number_of_columns, starting at and including start_column.

Syntax

`REMOVECOLUMNS([range], [number_of_columns], [start_column])`

- range: The matrix to remove columns from.
- number_of_columns: The number of columns to remove. Use a value greater than 0 to remove start_column and subsequent columns. Use a value less than 0 to remove start_column and previous columns.
- start_column: The starting column to remove. You can set start_column to -1 to start removing columns from the right side of the matrix instead of the left.

Example

The examples are based on this workbook:

	A	B	C	D	E
1	Item	2014	2015	2016	2017
2	100	6452	6557	6772	6457
3	200	3458	3568	4668	3455
4	300	6791	6552	5382	6235
5	400	3524	6592	3562	3899
6	500	6458	3112	2855	3298

The result of `=REMOVECOLUMNS(A1:E6,3,2)` is shown in this table:

Item	2017
100	6457
200	3455
300	6235
400	3899
500	3298

The result of `=REMOVECOLUMNS(A1:E6,-4,-1)` is shown in this table:

Item
100
200
300
400
500

Notes

- In the results, formatting from the original cells is not preserved.
- This function is intended for use in array formulas.

Related Functions

MERGECOLUMNS

REMOVEROWS

Description

Removes one or more rows from the referenced area. The function removes the number_of_rows following start_row, starting at and including start_row. Use REMOVEROWS without specifying any rows in order to remove the first (heading) row.

Syntax

`REMOVEROWS([range], [number_of_rows], [start_row])`

- range: The matrix to remove the rows from.
- number_of_rows: The number of rows to remove. Use a value greater than 0 to remove start_row and subsequent rows. Use a value less than 0 to remove start_row and previous rows.
- start_row: The starting row to remove. You can set start_row to -1 to start removing rows from the bottom of the matrix instead of the top.

Example

The examples are based on this workbook:

	A	B	C	D	E
1	Item	2014	2015	2016	2017
2	100	6452	6557	6772	6457
3	200	3458	3568	4668	3455
4	300	6791	6552	5382	6235
5	400	3524	6592	3562	3899
6	500	6458	3112	2855	3298

The result of `=REMOVEROWS(A1:E6,3,2)` is shown in this table:

	A	B	C	D	E
1	Item	2014	2015	2016	2017
2	400	3524	6592	3562	3899
3	500	6458	3112	2855	3298

The result of `=REMOVEROWS(A1:E6,-4,-1)` is shown in this table:

	A	B	C	D	E
1	Item	2014	2015	2016	2017
2	100	6452	6557	6772	6457

The result of `=REMOVEROWS(A1:E6)` is shown in this table (headings are removed):

	A	B	C	D	E
1	100	6452	6557	6772	6457
2	200	3458	3568	4668	3455
3	300	6791	6552	5382	6235
4	400	3524	6592	3562	3899
5	500	6458	3112	2855	3298

Notes

- In the results, formatting from the original cells is not preserved.
- This function is intended for use in array formulas.

Related Functions

ARRAYAREA

MERGEROWS

REPLACE

Description

Replaces a number of characters of old_text with new_text, from the start_num position.

Syntax

```
REPLACE(old_text, start_num, num_chars, new_text)
```

- old_text: The text containing the characters to replace.
- start_num: The position where you want to start replacing text. The value must be 0 or greater.
- num_chars: The number of characters to replace.
- new_text: The text to insert.

Example

Formula	Result
=REPLACE(A1,10,5,"Production") Where cell A1 contains `Regional Sales Manager` . Replace text starting with the 10th character (the letter S in Sales), and replace 5 characters (replace the word Sales).	Regional Production Manager

REPT

Description

Returns a string where the specified text is repeated number_times times.

Syntax

```
REPT(text, number_times)
```

- text: The text to repeat.
- number_times: The number of times to repeat the specified text.

Example

Formula	Result
=REPT(A1,2) Where cell A1 contains `Always` .	AlwaysAlways
=A1 & REPT("-", 8) & C1 Where cell A1 contains `Regional` and C1 contains `Manager` .	Regional--------Manager

RIGHT

Description

Returns the rightmost characters from a string, based on the specified text and number of characters.

Syntax

```
RIGHT(text, [ num_chars])
```

- text: The text to return characters from.
- num_chars: The number of characters to return. The default is 1.

Example

Formula	Result
=RIGHT(A1,7) Where cell A1 contains `Regional Sales Manager` .	Manager

RINT

Description

Returns the integer closest to the number. If two integers are equally close, the result is the integer that is even.

Syntax

```
RINT(value)
```

- value: The number to round.

Example

Formula	Result
=RINT(12.34)	12
=RINT(-12.34)	-12
=RINT(PI())	3

ROMAN

Description

Returns the Roman numeral text string for the specified number.

Syntax

`ROMAN(number, [ form])`

- number: The integer to convert. The value must be between 0 and 3999.
- form: The form of the resulting Roman numeral. Valid values are from 0 to 4, where 0 is the Classic form and 4 is the most simplified form.

ROUND

Description

Rounds the number up or down to the accuracy that you specify, based on the sign and the value of the num_digits argument.

Syntax

`ROUND(number, num_digits)`

- number: The number to round.
- num_digits: The number of decimal places to rount to and the direction to round. Specify 0 to round to the closest integer. A positive num_digits specifies the number of digits to the right of the decimal point. A negative num_digits specifies the number of digits to the left of the decimal point.

Example

Formula	Result
=ROUND(12.34,0)	12
=ROUND(-12.34,0)	-12
=ROUND(12.34,-1)	10
=ROUND(-12.34,-1)	-10

ROUNDDOWN

Description

Rounds to the nearest number toward zero (down if positive; up if negative), to the number of decimal places that you specify.

Syntax

`ROUNDDOWN(number, num_digits)`

- number: The number to round.
- num_digits: The number of decimal places to round to. Specify 0 to round down to the closest integer. A positive num_digits specifies the number of digits to the right of the decimal point. A negative num_digits specifies the number of digits to the left of the decimal point.

Example

Formula	Result
=ROUNDDOWN(12.34,0)	12
=ROUNDDOWN(-12.34,0)	-12
=ROUNDDOWN(12.34,-1)	10
=ROUNDDOWN(-12.34,-1)	-10

ROUNDUP

Description

Rounds number to the nearest number away from zero (up if positive; down if negative), to the number of decimal places that you specify.

Syntax

```
ROUNDUP(number, num_digits)
```

- number: The number to round.
- num_digits: The number of decimal places to round to. Specify 0 to round up to the closest integer. A positive num_digits specifies the number of digits to the right of the decimal point. A negative num_digits specifies the number of digits to the left of the decimal point.

Example

Formula	Result
=ROUNDUP(12.34,0)	13
=ROUNDUP(-12.34,0)	-13
=ROUNDUP(12.34,-1)	20
=ROUNDUP(-12.34,-1)	-20

ROW

Description

Returns the first row number in the specified cell reference.

Syntax

```
ROW([ reference])
```

- reference: The reference to return the row number for. If not specified, ROW() returns the row number of the current cell.

ROWS

Description

Returns the number of rows in the specified range.

Syntax

```
ROWS(array)
```

- array: The array, or reference to a range of cells, to return the number of rows for.

Notes

- When you use ROWS() using a column argument such as B:B, the result is the last row in the sheet, in any column, that has ever contained a value.

RRI

Description

Returns the interest rate required to meet an investment goal, based on the specified present value, future value, and duration.

Syntax

```
RRI(nper, pv, fv)
```

- nper: The total number of periods for the investment.
- pv: The present value.
- fv: The future value.

RSQ

Description

Returns the square of the Pearson product-moment correlation coefficient for the specified values.

Syntax

```
RSQ(array1, array2)
```

- array1: The first array of numeric values.
- array2: The second array of numeric values.

SCAN

Description

Applies a LAMBDA function to each value in an array and returns an array of values with each intermediate value.

This formula is useful for creating running totals, cumulative sums, or other calculations that show the results after each step.

Syntax

```
SCAN(initial_value, array, lambda)
```

- initial_value: The starting value for the accumulator.
- array: The array of values to be scanned.
- lambda: The LAMBDA that defines the operation to reduce the array. There are three parameters:
  - accumulator: The value returned as the final result.
  - value: The current value from the array.
  - body: The calculation applied to each element in the array.

SEARCH

Description

Searches the specified text string for a sub-string. Returns the position of the find_text in the within_text. The search is not case-sensitive. For case-sensitive search, use FIND.

Syntax

`SEARCH(find_text, within_text, [ start_num])`

- find_text: The text to find in the string.
- within_text: The text to search.
- start_num: The starting position for the search.

Example

Formula	Result
=SEARCH("sales",A1) Where cell A1 contains `Regional Sales Manager` .	10

SEC

Description

Returns the secant of the specified angle.

Syntax

`SEC(value)`

- value: The angle to calculate the secant for.

SECH

Description

Returns the hyperbolic secant of the specified angle.

Syntax

`SECH(value)`

- value: The angle to calculate the hyperbolic secant for.

SECOND

Description

Returns an integer from 0 to 59 representing only the seconds specified in the time.

Syntax

`SECOND(date)`

- date: The time to get the seconds value from.

Example

Formula	Result
=SECOND("8/14/2016 05:30 PM")	0
=SECOND("12/31/2016 05:30:59")	59

SELECT

Description

Selects data from a workbook range, a defined name, or a Workday report. The SELECT function is similar to an SQL SELECT statement. The primary rules for writing a SELECT statement are: Data that you reference in a select statement must have a header name. Surround the select statement with double quotes. Surround strings with single quotes. If you reference a column heading or table name that contains spaces or special characters, enclose it in backticks. Use question marks for parameters. If the SELECT formula doesn't finish calculating within 30 minutes, it returns an #ERROR. SELECT can process up to 5 million rows, but the number of rows processed might not exactly match the number of returnable rows due to the aggregation that SELECT performs. To see working examples of the SELECT function, find the "Worksheets Function Example Workbooks" page on Community or paste this link into your browser: https://community.workday.com/node/408872.

Keep in mind that the SELECT function is not intended as a replacement for the Data Wizard.

Syntax

`SELECT(select_statement, [ parameter], ...)`

- select_statement: The select statement. The statement is similar to an SQL select statement. Enclose the select statement in double quotes. The basic format is: `"SELECT column1, [column2], ... FROM table"` where `column` is the data to return and `table` is the data source to select from.

  You can embed functions in the SELECT statement such as COUNT, AVG, SUM, LIMIT, and more. The complete list of functions that SELECT supports is below.

  When doing an operation on date values in your SELECT formulas, keep in mind that Worksheets works differently from SQL syntax. Worksheets calculations treat dates as serial numbers. Example: The date March 20, 2023 is represented as the serial number 45005. Also, SQL automatically converts string dates for calculations, but Worksheets doesn't.

  If the string *from* or *where* exists in a column name, you must surround it with quotes because from and where are reserved words in SQL.

  In the FROM clause of the statement, you can use values such as a:

  - Workday report.
  - Parameter that you specify as an argument.
- parameter: A statement parameter.

Examples

We'll use this table to show some simple examples:

header1	header2	header3
1	a	red
2	b	orange

header1	header2	header3
3	c	orange
4	d	red
5	e	red

The formula:

=SELECT("SELECT header1,header3 FROM ? WHERE header3 = ?",A1:C6,"red")

has this result:

header1	header3
1	red
4	red
5	red

Automatic recalculation and the parameter argument. We recommend using the parameter argument of the function to reference data; Worksheets considers these references to be data dependencies, so the SELECT formula runs automatically when you change the data in the referenced cells. If you use the SELECT statement to refer to data instead, the formula doesn't re-run automatically if the data changes, and even selecting Data > Recalculate doesn't cause the formula to run again.

In the following formula, we use the parameter argument to reference the data. If the data changes in the range A1:C6, Worksheets automatically recalculates and updates the result:

=SELECT("SELECT header1,header3 FROM ?",A1:C6)

Defined name. In this example, an area was given the defined name all_employees:

- =SELECT("SELECT employeeName FROM ?", all_employees)

Column, row, or range. You can refer to a column, row, or area in the same sheet, a different sheet, or in another workbook. If you don't specify a sheet, the range must be on the current sheet.

- =SELECT("SELECT employeeName FROM ?",$A:$G)
- =SELECT("SELECT employeeName FROM ?','All Employees Sheet'!$A:$G)
- =SELECT("SELECT employeeName FROM ?",All_Employees_Sheet!$A:$G)
- =SELECT("SELECT employeeName FROM ?", !'Other Workbook Name'!'All Employees Sheet'!A:G)

More Examples and Results

Formula	Result
SELECT("SELECT employeeID, employeeName FROM ?", AllEmployeesSheet!$A:$G)	Returns the employeeID and employeeName from columns A through G of the sheet named AllEmployeesSheet.
SELECT("SELECT COUNT(employeeLastName) FROM ?", AllEmployeesSheet)	Returns the number of rows in the employeeLastName column in the sheet named AllEmployeesSheet.
SELECT("SELECT employeeFirstName, employeeLastName, employeeRating FROM ? ORDER BY employeeRating DESC LIMIT 3", AllEmployeesSheet!$A:$G)	Return the first name, last name, and rating of the 3 highest rated employees.

Workday report. Query data from a Workday report.

- =SELECT("SELECT employeeName FROM WorkdayReports.AllEmployees") where the report name is AllEmployees.
- =SELECT("SELECT employeeName FROM WorkdayReports.`All Employees`") where the report name is All Employees.
- =SELECT("SELECT * from WorkdayReports.`All Employees`") selects all rows from the report.

Notes

- We recommend using the parameter section of the function to reference data; Worksheets considers these references to be data dependencies, so the SELECT formula runs whenever you select Data > Recalculate. The data you refer to in the SELECT statement portion of the formula isn't considered dependent data, so selecting Data > Recalculate doesn't cause the formula to run again even if the underlying data changed.
- This function is intended for use in array formulas.

SELECT Functions: Constants

TRUE	CURRENT_DATE
FALSE	CURRENT_TIME
NULL	CURRENT_TIMESTAMP

SELECT Functions: Operators and Logical

UNION	JOIN	INTERSECT	INNER JOIN
UNION ALL	LEFT JOIN	EXCEPT	BETWEEN
NOT BETWEEN	IN	*	<=
|| (string concatenation)	NOT IN	/	<>
% (modulo)	() (parentheses)	+	!=
<< (bitshift left)	IF	- (minus)	>=
>> (bitshift right)	+ (unary positive)	- (unary negative)	~ (bit inversion)
EXISTS	! (logical not)	|	LIKE
NOT EXISTS	NOT	<	NOT LIKE
IS NULL	AND	>	CASE [1]
IS NOT NULL	OR	=	&
IFNULL	ISNULL	POW	"

[1] Including WHEN THEN ELSE END keywords

SELECT Functions: Scalar, Substring

DAYOFMONTH	YEAR	LOWER	UPPER
HOUROFDAY	COALESCE	LTRIM	DATE Y M D
MINUTE	NULLIF	RTRIM	DATETIME Y M D H M S
MONTH	ROUND	SUBSTRING	INSTANCE (ID and description)
SECOND	LENGTH	TRIM	DATEDIF

REPLICATE	STUFF	LOCATE	INSTR

SELECT Functions: Math, Financial, Statistical

ABS	COUNT	LOG10	Q (Quarter)
AVG	FLOOR	MAX	SQRT
CEIL	LOG	MIN	SUM
LIMIT	ASC	DESC	

SEQUENCE

Description

Creates an array of sequential values, such as a numbers or a series of dates. This function can be helpful when creating dynamic models. Example: You are creating an ad hoc forecasting model and you want to create a dynamic sequence of dates based on time and not based on the live data in the workbook.

Syntax

```
SEQUENCE(rows, [ columns], [ start], [ step])
```

- rows: An integer for the number of rows to return.
- columns: An integer for the number of columns to return.
- start: The starting number for the sequence.
- step: The number to increase each value by. Example: When start is for date values, a step value of 7 increments a date series in weekly (7 day) increments.

Example

To download working examples of the SEQUENCE function in a workbook, see https://collaborate.workday.com/t5/-/-/m-p/910596 .

SERIESSUM

Description

Returns the sum of a power series.

Syntax

```
SERIESSUM(value, initial_power, step_size, coefficients)
```

- value: The value for the power series.
- initial_power: The initial power to raise the value to.
- step_size: The step size to increase the initial power by, for each successive power of the input value.
- coefficients: The set of coefficients to multiply each successive power of the input value by.

SETUNITS

Description

Converts a number from its current unit of measurement to another. This function is similar to CONVERT(), but in CONVERT() you must specify both the original and the new units value.

Syntax

```
SETUNITS(value, set_unit)
```

- value: The number to convert.
- set_unit: The units to return the value in. The set_unit argument is case-sensitive.

Example

Formula	Result
=SETUNITS(825,"cup") where 825 currently has the "tsp" unit	17.185
=SETUNITS(98.6,"cel") where 98.6 currently has the "F" unit	37

Notes

- Valid units values are from the base library javax.measure.unit.Dimension.

SIGN

Description

Returns -1 if the number is less than 0; otherwise, returns 1.

Syntax

```
SIGN(value)
```

- value: The number to evaluate.

SIN

Description

Returns the sine of the specified value.

Syntax

```
SIN(value)
```

- value: The value to calculate the sine for.

Related Functions

CBRT

HYPOT

SINH

Description

Returns the hyperbolic sine of the specified value.

Syntax

```
SINH(value)
```

- value: The value to calculate the hyperbolic sine for.

SLN

Description

Returns the depreciation amount for an asset for the specified period, based on the straight line depreciation method.

Syntax

```
SLN(cost, salvage, life)
```

- cost: The initial cost of the asset.
- salvage: The value of the asset after depreciation.
- life: The number of periods that the asset will depreciate over.

SLOPE

Description

Returns the slope of the linear regression line, based on the specified x and y values.

Syntax

```
SLOPE(known_ys, known_xs)
```

- known_ys: The known y values.
- known_xs: The known x values.

SMALL

Description

Returns the kth smallest value from the specified list of values and value of k.

Syntax

```
SMALL(array, k)
```

- array: The array of numbers.
- k: The index.

SORT

Description

Sorts an existing matrix and returns a new matrix. SORT() accepts one sort direction and sorts all columns you specified based on that direction.

Syntax

```
SORT(matrix, [ ascending], [ sort_column], ...)
```

- matrix: The matrix to sort.

- ascending: If TRUE, the values are sorted in ascending order; if FALSE, the values are sorted in descending order. The default is TRUE.
- sort_column: One or more column names or column positions to use as the columns to sort on. If you don't specify any columns, then the first (left-most) column is used.

Example

This is the original spreadsheet:

	A	B	C	D	E
1	Name	Grade			
2	Joe	85			
3	Alice	92			
4	Joe	90			

The result of `=SORT(A1:B4,true,"Name","Grade")` is:

Name	Grade
Alice	92
Joe	85
Joe	90

Notes

- To specify a different sort direction for each column being sorted, use SORT2().
- This function sorts the rows in the source matrix and returns a new matrix reflecting the sorted rows. The order of the rows is determined by the values in the sort_column and the direction you specified (ascending or descending).
- Specify sort columns either by name (as text strings without any surrounding square brackets) or by position (where the first column is in position 1). You can mix these values (specifying some by name and others by position).
- When you specify more than one sort column, the function breaks ties by sorting the values in each subsequently specified column. Example: If the first column has three values of the string `"hello"`, and you specified a second column, the function sorts the values in that column for the three rows that have `"hello"` in the first column. This determines the final order of those three columns.
- This function is intended for use in array formulas.

Related Functions

ARRAYAREA

SORT2

SORT2

Description

Sorts an existing matrix and returns a new matrix. SORT2 accepts pairs of parameters, specifying the column to sort by and the sort direction for that column. When ties occur during the sort, arguments for columns after the first column you specified are used.

Syntax

`SORT2(matrix, [ sort_column_1, ascending_1], ...)`

- matrix: The matrix to sort.
- sort_column_1: The column name or position of the value in the top priority sorting column. If you specify a sort_column, you must also specify a corresponding ascending boolean.
- ascending_1: If TRUE, the function sorts the values in the associated column in ascending order; if FALSE, the function sorts the values in the associated column in descending order.

Example

This is the original spreadsheet:

	A	B
1	Name	Grade
2	Joe	85
3	Alice	92
4	Joe	90
5	Luis	92
6	Xibin	85
7	Alice	79
8	Joe	82
9	Luis	95
10	Xibin	88
11	Alice	97

The result of `=SORT2(A1:B10,"Name",,"Grade",FALSE)` is:

Name	Grade
Xibin	88
Xibin	85
Luis	95
Luis	92
Joe	90
Joe	85
Joe	82
Alice	97
Alice	92
Alice	79

Notes

- You can specify up to 8 column/direction pairs.
- To sort all the columns you specified in only one direction, you can use SORT.

- This function sorts the rows in the source matrix and returns a new matrix reflecting the sorted rows. The order of the rows is determined by the value in sort_column_n and the direction you specified (ascending or descending).
- Specify sort columns either by name (as text strings without any surrounding square brackets) or by position (where the first column is in position 1). You can mix these values (specifying some by name and others by position). If you use the column header name (string) the function preserves the column headers, but if you use the column indexes (numbers), the function sorts the headers as data.
- When you specify more than one sort column, the function breaks ties by sorting the values in each subsequently specified column. Example: If the first column has three values of the string `"hello"`, and you specified a second column, the function sorts the values in that column for the three rows that have `"hello"` in the first column. This determines the final order of those three columns.
- This function is intended for use in array formulas.

Related Functions

ARRAYAREA

SORT

SORT3

SORT3

Description

Sorts an existing matrix and returns a new matrix. SORT3 assumes that the first row is a header, and returns the header as the first row in the results. SORT3 accepts pairs of parameters, specifying the column to sort by and the sort direction for that column. When ties occur during the sort, arguments for columns after the first column you specified are used.

Syntax

```
SORT3(matrix, [ sort_column_1, ascending_1], ...)
```

- matrix: The matrix to sort. SORT3 assumes that the first row is a header, and returns the header as the first row in the results.
- sort_column_1: The column name or position of the value in the top priority sorting column.
- ascending_1: If TRUE, the function sorts the values in the associated column in ascending order; if FALSE, the function sorts the values in the associated column in descending order.
- sort_column_2: The column name or position of the value in the 2nd priority sorting column.
- ascending_2: If TRUE, the function sorts the values in the associated column in ascending order; if FALSE, the function sorts the values in the associated column in descending order.
- sort_column_3: The column name or position of the value in the 3rd priority sorting column.
- ascending_3: If TRUE, the function sorts the values in the associated column in ascending order; if FALSE, the function sorts the values in the associated column in descending order.

Example

This is the original spreadsheet:

	A	B
1	Name	Grade
2	Joe	85
3	Alice	92

	A	B
4	Joe	90
5	Luis	92
6	Xibin	85
7	Alice	79
8	Joe	82
9	Luis	95
10	Xibin	88
11	Alice	97

The result of `=SORT3(A1:B10,1,,2,FALSE)` is:

Name	Grade
Xibin	88
Xibin	85
Luis	95
Luis	92
Joe	90
Joe	85
Joe	82
Alice	97
Alice	92
Alice	79

Notes

- You can specify up to 8 column/direction pairs. The function ignores empty sort columns.
- Empty parameter pairs are ignored. For strict parameter pairs you can use SORT2.
- To sort all the columns you specified in only one direction, you can use SORT().
- This function sorts the rows in the source matrix and returns a new matrix reflecting the sorted rows. The order of the rows is determined by the value in sort_column_n and the direction you specified (ascending or descending).
- Specify sort columns either by name (as text strings without any surrounding square brackets) or by position (where the first column is in position 1). You can mix these values (specifying some by name and others by position). With either method, the function preserves the header and returns it as the first row of results.
- When you specify more than one sort column, the function breaks ties by sorting the values in each subsequently specified column. Example: If the first column has three values of the string `"hello"`, and you specified a second column, the function sorts the values in that column for the three rows that have `"hello"` in the first column. This determines the final order of those three columns.
- This function is intended for use in array formulas.

Related Functions

ARRAYAREA

SORT

SORT2

SORTBY

Description

Sorts the specified array or range based on the values in a corresponding array. You can specify pairs of by_array and sort_order arguments for a primary sort, secondary sort, and so on. This function is intended for use with unconstrained arrays. To submit the formula, use the unconstrained keyboard shortcut Ctrl+Alt +Enter (Windows) or Command+Option+Enter (Mac).

Syntax

```
SORTBY(array, by_array1, [ sort_order1], [ by_array2, sort_order2], ...)
```

- array: The array to sort.
- by_array1: The array to sort on.
- sort_order1: A number that specifies the sort order. Specify 1 to sort in ascending order or -1 to sort in descending order. The default is 1 (ascending).
- by_array2: A second priority array to sort on.
- sort_order2: A number that specifies the sort order for the secondary sort. Specify 1 to sort in ascending order or -1 to sort in descending order. The default is 1 (ascending).

Related Functions

ARRAYAREA

SORT

SORT2

SPLIT

Description

Splits a string into multiple strings. This function is useful when the string has boundaries that you can match using a regular expression.

Syntax

```
SPLIT(text, regex)
```

- text: The text to split.
- regex: The regular expression, specified as text, to use to find the location of the split.

Example

Formula	Result
=SPLIT("a,b,c,d", ",")	`a b c d`

Notes

- The SPLIT() function is useful when you want to break a string into pieces and the string has boundaries that you can match using a regular expression. Example: In the string "a,b,c,d", you can split on the "," character to produce 4 separate strings: "a", "b", "c", and "d".
- This function is intended for use in array formulas.

SQRT

Description

Returns the square root of the (positive) number.

Syntax

`SQRT(value)`

- value: The number to calculate the square root for.

Related Functions

CBRT

HYPOT

SQRTPI

Description

Returns the square root of the (positive) number, multiplied by pi.

Syntax

`SQRTPI(value)`

- value: The number to calculate the square root for and to multiply by pi.

STANDARDIZE

Description

Returns the normalized value of a distribution based on the specified mean and standard deviation.

Syntax

`STANDARDIZE(x, mean, standard_dev)`

- x: The value to normalize.
- mean: The arithmetic mean of the distribution.
- standard_dev: The standard deviation of the distribution.

STDEV

Description

Estimates the standard deviation of the argument values, which represent a sample of the population.

Syntax

`STDEV(numbers, ...)`

- numbers: The list of numbers for the population sample.

STDEV.P

Description

Estimates the standard deviation of the argument values, which represent the entire population. Supports up to 255 arguments.

Syntax

```
STDEV.P(args, ...)
```

- args: The list of numbers for the population.

STDEV.S

Description

Estimates the standard deviation of the argument values, which represent a sample of the population. Supports up to 255 arguments.

Syntax

```
STDEV.S(numbers, ...)
```

- numbers: The list of numbers for the population sample.

STDEVP

Description

Estimates the standard deviation of the argument values, which represent the entire population. Supports up to 255 arguments.

Syntax

```
STDEVP(args, ...)
```

- args: The list of numbers for the population.

STDEVS

Description

Estimates the standard deviation of the argument values, which represent a sample of the population. Supports up to 255 arguments.

Syntax

```
STDEVS(numbers, ...)
```

- numbers: The reference to cells, or list of references, to do the calculation on.

SUBSTITUTE

Description

Replaces either one or all occurrences of old_text with new_text. The function is case-sensitive.

Syntax

`SUBSTITUTE(text, old_text, new_text, [ instance_num])`

- text: The text to replace characters in.
- old_text: The text to replace.
- new_text: The text to insert.
- instance_num: The occurrence of old_text to replace with new_text. If you don't specify an instance_num, the function replaces all occurrences.

Example

Formula	Result
=SUBSTITUTE(A1,"Sales","Marketing") Where cell A1 contains `Regional Sales Manager` .	Regional Marketing Manager

SUBTOTAL

Description

Returns a subtotal based on the specified type of calculation and the cells to use. The function doesn't include values from hidden rows (but it does from hidden columns), depending on the aggregation specifier. The function doesn't include values from rows that have been filtered out. A value that's the result of, or includes, a SUBTOTAL() function isn't included in the new SUBTOTAL(). If you specify multiple ranges, then the function includes all values from the ranges in the final aggregation.

Syntax

`SUBTOTAL(function_num, ref1, [ ref2], ...)`

- function_num: The number representing the function to perform.
- ref1: A reference to cells containing values for the calculation.
- ref2: A reference to cells containing values for the calculation.

SUBTRACT

Description

Subtracts number2 from number1. Although this function is available to you, we recommend that you simply write `number2 - number1` when you want to subtract two numbers.

Syntax

`SUBTRACT(number1, number2)`

- number1: The number to be subtracted from.
- number2: The number to subtract.

Example

Formula	Result
=SUBTRACT(10,3)	7

Notes

- Don't confuse this function with MINUS(). MINUS() performs set-based subtraction.

SUM

Description

Adds the numbers specified as arguments.

Syntax

```
SUM(args, ...)
```

- args: The list of numbers.

SUMIF

Description

Adds the numbers specified as arguments, if a condition is met.

Syntax

```
SUMIF(range, criteria, [ sum_range])
```

- range: The range of values, or cells, to test.
- criteria: The condition to test each value against.
- sum_range: The range of values, or cells, to add together. If not specified, the values in the range argument are summed.

SUMIFS

Description

Adds the numbers specified in the sum_range, if a set of criteria are met.

Syntax

```
SUMIFS(sum_range, range, criteria, [ range2, criteria2], ...)
```

- sum_range: The range of values, or values in cells, to sum if all criteria are met.
- range: The range of values, or cells, to test.
- criteria: The condition to test each value against.
- range2: The range of values, or cells, to test.
- criteria2: The condition to test each value against.

SUMPRODUCT

Description

Returns the sum of the products of the specified array elements. The arrays must have the same dimensions.

Syntax

```
SUMPRODUCT(array, ...)
```

- array: The array or list of arrays.

SUMSQ

Description

Returns the sum of the squares of the specified set of values.

Syntax

```
SUMSQ(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

Related Functions

CBRT

HYPOT

SUMX2MY2

Description

Returns the sum of the differences of the squares of two specified arrays.

Syntax

```
SUMX2MY2(array1, array2)
```

- array1: The array of values, or cells.
- array2: The array of values, or cells.

SUMX2PY2

Description

Returns the sum of the sum of the squares of two specified arrays.

Syntax

```
SUMX2PY2(array1, array2)
```

- array1: The array of values, or cells.
- array2: The array of values, or cells.

SUMXMY2

Description

Returns the sum of the squares of the differences between two specified arrays.

Syntax

```
SUMXMY2(array1, array2)
```

- array1: The array of values, or cells.
- array2: The array of values, or cells.

SWITCH

Description

Evaluates an expression against the match values. You can include multiple sets of match values and return values. The function returns a result corresponding to the first matched argument. You can specify a default value to return if the function doesn't find a match; otherwise, the function returns the #N/A error.

Syntax

```
SWITCH(expression, value1, result1, [ value2, result2, ...], [ default])
```

- expression: The value, such as a number, date, or text, to evaluate against the match values.
- value1: Value to compare to the expression.
- result1: Value to return if the match value matches the expression.
- value2: Value to compare for a possible match.
- result2: Value to return if the match value matches the expression.
- default: Value to return if the function doesn't find a match.

SYD

Description

Returns the depreciation amount for an asset for the specified period, based on the sum of the years' digits method.

Syntax

```
SYD(cost, salvage, life, per)
```

- cost: The initial cost of the asset.
- salvage: The value of the asset after depreciation.
- life: The number of periods that the asset will depreciate over.
- per: The number of the period to calculate the depreciation for.

T

Description

Determines if the specified value is text. If the value is text, the function returns the specified text; otherwise, it returns an empty text string.

Syntax

```
T(text)
```

- text: The value to evaluate.

Example

Formula	Result
=T(A1) Where cell A1 contains `Regional Sales Manager` .	TRUE
=T(A1)	(empty)

Formula	Result
Where cell A1 contains `$2,451.00` .	

T.DIST

Description

Returns the t-distribution at a specified value. Similar to TDIST() but T.DIST() enables you to select the type of distribution instead of the number of tails.

Syntax

```
T.DIST(x, deg_freedom, cumulative)
```

- x: The input value.
- deg_freedom: The number of degrees of freedom.
- cumulative: The type of distribution to use. FALSE = probability density function; TRUE = cumulative distribution function.

T.DIST.RT

Description

Returns the right-tailed t-distribution.

Syntax

```
T.DIST.RT(x, deg_freedom)
```

- x: The input value.
- deg_freedom: The number of degrees of freedom.

Example

Formula	Result
=T.DIST.RT(18.307,10)	0.000000003

Notes

- This function does the same action as TDISTRT().

T.INV

Description

Returns the inverse of the left-tailed t-distribution.

Syntax

```
T.INV(probability, deg_freedom)
```

- probability: The number between 0 and 1 for the probability.
- deg_freedom: The number of degrees of freedom.

TAKE

Description

Returns a subset of the specified array, from the start or end of an array, based on the number of rows or columns that you specify.

Syntax

```
TAKE(array, rows,[columns])
```

- array: The array to take rows or columns from.
- rows: The number of rows to take. If you specify a negative value, the function takes from the end of the array.
- columns: The number of columns to take. If you specify a negative value, the function takes from the end of the array.

TAN

Description

Returns the tangent of the specified angle.

Syntax

```
TAN(value)
```

- value: The angle to calculate the tangent for.

Related Functions

CBRT

HYPOT

TANH

Description

Returns the hyperbolic tangent of the specified value.

Syntax

```
TANH(value)
```

- value: The value to calculate the hyperbolic tangent for.

TBILLEQ

Description

Returns the bond-equivalent yield for a treasury bill.

Syntax

```
TBILLEQ(settlement, maturity, discount)
```

- settlement: The treasury bills settlement date.
- maturity: The treasury bills maturity date.
- discount: The treasury bills discount rate.

TBILLPRICE

Description

Returns the price per $100 face value, for a treasury bill.

Syntax

```
TBILLPRICE(settlement, maturity, discount)
```

- settlement: The treasury bills settlement date.
- maturity: The treasury bills maturity date.
- discount: The treasury bills discount rate.

TBILLYIELD

Description

Returns the yield per $100 face value, for a treasury bill.

Syntax

```
TBILLYIELD(settlement, maturity, price)
```

- settlement: The treasury bills settlement date.
- maturity: The treasury bills maturity date.
- price: The treasury bills price.

TDIST

Description

Returns the t-distribution at a specified value. Similar to T.DIST() but TDIST() enables you to select the number of tails for the distribution instead of the type of distribution.

Syntax

```
TDIST(x, deg_freedom, tails)
```

- x: The input value.
- deg_freedom: The number of degrees of freedom.
- tails: The number of tails for the distribution. 1 = one-tailed; 2 = two-tailed.

TDISTRT

Description

Returns the right-tailed t-distribution.

Syntax

```
TDISTRT(x, deg_freedom)
```

- x: The input value.
- deg_freedom: The number of degrees of freedom.

Example

Formula	Result
=TDISTRT(18.307,10)	0.000000003

Notes

- This function does the same action as T.DIST.RT().

TEXT

Description

Converts the specified number into text, based on the specified format. Example: Worksheets normally stores dates using their serial numbers for efficiency in calculations; you can use the TEXT function to reformat the date as a user-readable value.

Syntax

`TEXT(value, format_text)`

- value: The number to convert.
- format_text: The format to use when converting the number.

Format options:

0	Display a digit.
#	Display a digit if it increases number accuracy. Don't display a digit if it is a leading zero or a zero at the end of a decimal.
.	Specifies the position of the decimal.
d	Day of month or day of week: • d = 1 or 2 digit representation (e.g. 1, 12) • dd = 2 digit representation (such as 01, 12) • ddd = day of week abbreviation (such as Mon, Tue) • dddd = full day of week (such as Monday, Tuesday)
m	Month (when part of a date): • m = 1 or 2 digit representation (such as 1, 12) • mm = 2 digit representation (such as 01, 12) • mmm = month name abbreviation (such as Jan, Dec) • mmmm = full month name (such as January, December)
y	Year: • yy = 2 digit representation (such as 17, 09) • yyyy = 4 digit representation (such as 2017)
h	Hour: • h = 1 or 2 digit representation (such as 1, 20) • hh = 2 digit representation (such as 01, 20)
m	Minute (when part of a time): • m = 1 or 2 digit representation (such as 1, 55) • mm = 2 digit representation (such as 01, 55)

s	Second: • s = 1 or two 2 digit representation (such as 1, 45) • ss = 2 digit representation (such as 01, 45)
AM/PM	Represent time using a 12-hour clock, followed by "AM" or "PM".

Example

Formula	Result
=TEXT(A1, "mm/dd/yyyy") Where cell A1 contains `43210` .	04/20/2018
=TEXT(A1, "dd/mm/yyyy") Where cell A1 contains `12/25/2017` .	25/12/2017
=TEXT(A1, "0.0%") Where cell A1 contains `123.45678` .	12345.7%

TEXTAFTER

Description

Returns text that occurs after the specified delimiting characters.

Syntax

```
TEXTAFTER(text, delimiter, [instance_num], [match_mode], [match_end],
[if_not_found])
```

- text: The text to search. Worksheets doesn't support using wildcard characters in the search.
- delimiter: The text that marks the point after which you want to extract. The function returns the text that's located after this text.
- instance_num: The delimiter instance after which you want to extract the text. If you don't specify the argument, the default is 1. If you specify a negative number, the search starts from the end.
- match_mode: Determines whether the text search is case-sensitive. The default is case-sensitive. Optional. Enter one of the following: 0 = case sensitive. 1 = case insensitive.
- match_end: Treats the end of text as a delimiter. By default, the text is an exact match. Optional. Enter the following: 0 = Don't match the delimiter against the end of the text. 1 = Match the delimiter against the end of the text.
- if_not_found: The value to return if the function doesn't find a match. By default, the function returns #N/A.

Example

The examples are based on this table. The example functions will return a worker's last name using the space between the first and last names as the delimiter, and the last 4 digits of their Social Security number using the hyphen as the delimiter.

	A	B	C	D	E
1	Full Name	First Name	Last Name	SS#	Last 4
2	Logan McNeil			232-45-7667	
3	Steve Morgan			611-33-5576	

	A	B	C	D	E
4	Oliver Reynolds			242-56-9755	
5	Teresa Serrano			432-67-1442	

Formula	Result
=TEXTAFTER(A2," ")	`McNeil`
=TEXTAFTER(D2,"-",-1)	`7667`

TEXTBEFORE

Description

Returns the text that occurs before the specified delimiting characters.

Syntax

```
TEXTBEFORE(text, delimiter, [instance_num], [match_mode], [match_end],
[if_not_found])
```

- text: The text to search. Worksheets doesn't support using wildcard characters in the search.
- delimiter: The text that marks the point before which you want to extract. The function returns the text that's located before (in front of) this text.
- instance_num: The delimiter instance after which you want to extract the text. If you don't specify the argument, the default is 1. If you specify a negative number, the search starts from the end.
- match_mode: Determines whether the text search is case-sensitive. The default is case-sensitive. Optional. Enter one of the following: 0 = case sensitive. 1 = case insensitive.
- match_end: Treats the end of text as a delimiter. By default, the text is an exact match. Optional. Enter the following: 0 = Don't match the delimiter against the end of the text. 1 = Match the delimiter against the end of the text.
- if_not_found: The value to return if the function doesn't find a match. By default,the function returns #N/A.

Example

The example is based on this table. The example function will return a worker's first name, using the space between the first and last names as the delimiter.

	A	B	C	D	E
1	Full Name	First Name	Last Name	SS#	Last 4
2	Logan McNeil			232-45-7667	
3	Steve Morgan			611-33-5576	
4	Oliver Reynolds			242-56-9755	
5	Teresa Serrano			432-67-1442	

Formula	Result
=TEXTBEFORE(A2," ")	`Logan`

TEXTJOIN

Description

Joins a series of text strings into a single text string. You specify a delimiter to add between the text items. TEXTJOIN is similar to the CONCAT function, but TEXTJOIN accepts a delimiter, whereas CONCAT can't.

Syntax

```
TEXTJOIN(delimiter, ignore_empty, text,[ text2],...)
```

- delimiter: A delimiter to be inserted between each text string.
- ignore_empty: A logical value that specifies whether to skip or consider empty cells.
- text: A text string or array of text strings.
- text2: An additional text string or array of text strings.

Notes

- If you want to join text containing dates, keep in mind that Worksheets stores dates and times as serial numbers; if you enter a date or a time directly into the TEXTJOIN function, the number displays in the resulting text string instead of the formatted date or time. To display the value as a date, use the TEXT function to convert the date or time into a text string first.

TEXTSPLIT

Description

Splits text strings into rows or columns, by specifying delimiter text that indicates where to split the text.

Syntax

```
TEXTSPLIT(text, col_delimiter, [row_delimiter], [ignore_empty], [match_mode],
[pad_with])
```

- text: The text that you want to split.
- col_delimiter: The text that marks the point where to spill the text across columns.
- row_delimiter: The text that marks the point where to spill the text down rows.
- ignore_empty: Specify TRUE to ignore empty values. Specify FALSE to create an empty cell when two delimiters are consecutive. If you don't specify an argument, the default is TRUE.
- match_mode: Determines whether the search is case-sensitive. The default is case-sensitive. Enter one of the following: 0 = Case sensitive. 1 = Case insensitive.
- pad_with: The value to pad the result with. If you don't specify the argument, the default is #N/A.

TIME

Description

Returns the current time based on the specified hour, minute, and second.

Syntax

```
TIME(hour, minute, second)
```

- hour: The integer representing the hour.
- minute: The integer representing the minutes.
- second: The integer representing the seconds.

Example

Formula	Result
=TIME(23,23,59)	11:23 PM
=TIME(1,15,99)	1:16 AM

TIMEVALUE

Description

Returns a numerical time value based on the specified text.

Syntax

`TIMEVALUE(time_text)`

- time_text: The text to convert to a numerical time value. Example: =TIMEVALUE("2:23 pm").

Example

Formula	Result
=TIMEVALUE("12/31/2016 00:01:01")	0.000706019
=TIMEVALUE("11:59 pm")	0.999305556
=TIMEVALUE("23:59:59")	0.999988426

TINV

Description

Returns the inverse of the left-tailed t-distribution.

Syntax

`TINV(probability, deg_freedom)`

- probability: The number between 0 and 1 for the probability.
- deg_freedom: The number of degrees of freedom.

TOCOL

Description

Returns the specified array as a single column.

Syntax

`TOCOL(array, [ignore], [scan_by_column])`

- array: The array or reference to return as a column.
- ignore: Specifies whether to ignore certain types of values. By default, the function doesn't ignore any values. Possible values are: 0 = Keep all values. 1 = Ignore blanks. 2 = Ignore errors. 3 = Ignore blanks and errors.
- scan_by_row: Scan the array by row. By default, the function scans the array by column. Scanning determines whether the values are ordered by row or by column.

TODAY

Description

Returns the current date. This function is volatile - it generates a new value any time any cell in the workbook changes - but it doesn't recalculate automatically based on the workbook date. If the function doesn't return the value you expect, check the workbook time zone setting in File > About. The workbook inherits the time zone setting of the workbook creator. If needed, you can change the time zone value by navigating to File > Settings.

Syntax

```
TODAY()
```

Example

Formula	Result
=TODAY()	12/21/2017

TOROW

Description

Returns the specified array as a single row.

Syntax

```
TOROW(array, [ignore], [scan_by_row])
```

- array: The array or reference to return as a row.
- ignore: Specifies whether to ignore certain types of values. By default, the function doesn't ignore any values. Possible values are: 0 = Keep all values. 1 = Ignore blanks. 2 = Ignore errors. 3 = Ignore blanks and errors.
- scan_by_column: Scan the array by column. By default, the function scans the array by row. Scanning determines whether the values are ordered by row or by column.

TRANSPOSE

Description

Returns a transposed reference to a range, where a horizontal range of cells is copied to a vertical range, or a vertical range of cells is copied to a horizontal range.

Syntax

```
TRANSPOSE(array)
```

- array: The range of cells to transpose.

TRIM

Description

Removes non-printable characters and extra spaces from text, including leading and trailing spaces. The function replaces internal multiple spaces with a single space. Non-printable characters are the ASCII numeric codes from 0 to 31. CLEAN is similar to TRIM, but TRIM also removes the space character: ASCII numeric code 32.

Syntax

```
TRIM(text)
```

- text: The text to be trimmed.

TRIMCOLUMNS

Description

Removes trailing blank columns from a range, when the data is a result of an unconstrained array formula.

Syntax

```
TRIMCOLUMNS(range)
```

- range: The range to trim.

Example

When you use an array formula such as =SampleWorkerData!3:5, which contains an unconstrained row reference, the formula returns a blank in each cell next to the cells containing data in rows 3, 4, and 5.

The formula =TRIMCOLUMNS(SampleWorkerData!3:5) removes the trailing columns containing blanks in the cells.

Notes

- This function is intended for use in array formulas.

TRIMMEAN

Description

Returns the trimmed mean of a specified list of values. The function discards the specified percentage of values: half the values from each end of the list.

Syntax

```
TRIMMEAN(array, percent)
```

- array: The list of values.
- percent: The percentage of values to be discarded before calculating the mean.

TRIMRANGE

Description

Removes empty rows and columns from the edges of a range of data or array.

This function is useful for adjusting the range size of dynamic datasets.

Syntax

```
TRIMRANGE(range,[trim_rows],[trim_columns])
```

- range: The range or array to be trimmed.

- trim_rows: A number that determines how rows should be trimmed. Options include:
  - 0: None.
  - 1: Trims leading blank rows.
  - 2: Trims trailing blank rows.
  - 3 (default): Trims both leading and trailing blank rows.
- trim_columns: A number that determines how columns should be trimmed. Options include:
  - 0: None.
  - 1: Trims leading blank columns.
  - 2: Trims trailing blank columns.
  - 3 (default): Trims both leading and trailing blank columns.

Trim References

A trim reference operator can be used to achieve the same functionality as TRIMRANGE by adding a period before and/or after the colon in the reference.

Table 1: Trim Reference Operators

Type	Example	Equivalent TRIMRANGE	Description
Trim All (.:.)	A1.:.E10	TRIMRANGE(A1:E10,3,3)	Trim leading and trailing blanks.
Trim Trailing (:.)	A1:.E10	TRIMRANGE(A1:E10,2,2)	Trim trailing blanks.
Trim Leading (.:)	A1.:Z10	TRIMRANGE(A1:E10,1,1)	Trim leading blanks.

This pattern can also be applied to full-column or full-row references (eg. A:.A). For example: The formula =SUM(A:.A) will sum only the values in column A, ignoring blank rows below.

TRIMROWS

Description

Removes trailing blank rows from a range, when the data is a result of an unconstrained array formula.

Syntax

`TRIMROWS(range)`

- range: The range to trim.

Example

When you use an array formula such as =SampleWorkerData!A:C, which contains an unconstrained column reference, the formula returns blanks in each cell below the cells containing data in columns A, B, and C.

The formula =TRIMROWS(SampleWorkerData!A:C) removes trailing rows containing blanks in the cells.

Notes

- This function is intended for use in array formulas.

TRUE

Description

Returns the logical value True.

Syntax

```
TRUE( )
```

Example

Formula	Result
=TRUE()	TRUE

TRUNC

Description

Converts a float to a number based on the decimal places that you specify.

Syntax

```
TRUNC(value, [ num_digits])
```

- value: The number to truncate.
- num_digits: The number of decimal places to truncate to. Specify 0 to truncate to the closest integer. A positive num_digits specifies the number of digits to the right of the decimal point. A negative num_digits specifies the number of digits to the left of the decimal point. If you omit the argument, the function uses the value 0, which causes the function to truncate to the closest integer.

Example

Formula	Result
=TRUNC(12.34)	12
=TRUNC(-12.34)	-12
=TRUNC(12.34,-1)	10
=TRUNC(-12.34,-1)	-10

TRUNCATEMATRIX

Description

Removes rows, columns, or both, from a matrix.

Syntax

```
TRUNCATEMATRIX(matrix, rows, [ columns])
```

- matrix: The source matrix to truncate.
- rows: The number of rows in the output matrix. If you specify zero, the output matrix will have the same number of rows as the input matrix.
- columns: The number of columns in the output matrix. If you specify zero or you don't specify a value, the output matrix will have the same number of columns as the input matrix.

Example

Formula	Result
=TRUNCATEMATRIX({1,2,3;4,5,6;7,8,9}, 2, 2)	`1 2 4 5`

Notes

- This function returns a matrix that is a result of removing rows and/or columns from the input matrix, essentially cropping the input matrix to a new set of dimensions.
- This function is intended for use in array formulas.

TYPE

Description

Returns an integer representing the data type of the specified value. 1 = Number, 2 = Text; 4 = Logical; 16 = Error; 64 = Array.

Syntax

`TYPE(value)`

- value: The value to return the data type for.

Example

Formula	Result
=TYPE("phrase")	2
=TYPE(TRUE)	4
=TYPE({A, B, C})	64

UMINUS

Description

Returns the number with its sign reversed.

Syntax

`UMINUS(number)`

- number: The number to reverse the sign for.

Example

Formula	Result
=UMINUS(A1) Where cell A1 contains 2 .	-2

UNARY_PERCENT

Description

Returns the specified number divided by 100. The appearance of the returned result depends on the Format setting for the cell.

Syntax

`UNARY_PERCENT(number)`

- number: The number to divide by 100. A null value returns 0. A string or date value is converted to a number and divided by 100.

Example

Formula	Result
=UNARY_PERCENT(A1) Where cell A1 contains `1.1123` .	0.01123
=UNARY_PERCENT(A1) Where cell A1 contains `32.255%` .	0.0032255

UNICHAR

Description

Returns the unicode character for the specified number.

Syntax

`UNICHAR(number)`

- number: The number that you want the Unicode character for.

Example

Formula	Result
=UNICHAR(A1) Where cell A1 contains `84` .	T
=UNICHAR(A1) Where cell A1 contains `25000` .	憨

UNICODE

Description

Returns the numeric value (code point) for the first character in the specified text string.

Syntax

`UNICODE(text)`

- text: The text that you want the Unicode value for; the function returns the first character's value.

Example

Formula	Result
=UNICODE(A1) Where cell A1 contains `T` .	84

UNIQUE

Description

Returns a matrix whose rows are unique according to the specified keys. The function returns only unique rows, based on the values in the specified columns. This function is similar to DISTINCTROWS(), but UNIQUE() takes a single range and a set of columns. UNIQUE is case-sensitive.

Syntax

```
UNIQUE(range, col1, [ col2], ...)
```

- range: The matrix to extract unique rows from.
- col1: The name of a column to use as a key in determining whether a row is unique.
- col2: The name of a column to use as a key in determining whether a row is unique.

Example

This is the original spreadsheet:

	A	B	C	D	E
1	Name	Grade			
2	Joe	85			
3	Alice	92			
4	Joe	90			

The result of =UNIQUE(A1:B4,"Name") is:

Name	Grade
Joe	85
Alice	92

Notes

- UNIQUE() is similar to DISTINCTROWS(). DISTINCTROWS() uses all the values in a row to determine whether the row is different from other rows, but UNIQUE() uses only the values in the specified columns. UNIQUE() filters the range so that it returns only unique rows based on the values in the specified columns. The first row it considers unique is the one it returns; it doesn't return subsequent duplicate rows.
- This function is intended for use in array formulas.

Related Functions

DISTINCTROWS

UNIQUE2

Description

Returns a matrix whose rows are unique according to the specified keys. The function returns only unique rows, based on the values in the specified columns. This function is similar to DISTINCTROWS2(), but UNIQUE2() takes a single range and a set of columns. UNIQUE2 is case-insensitive.

Syntax

```
UNIQUE2(range, col1, [ col2], ...)
```

- range: The matrix to extract unique rows from.
- col1: The name of a column to use as a key in determining whether a row is unique.
- col2: The name of a column to use as a key in determining whether a row is unique.

Example

This is the original spreadsheet:

	A	B	C	D	E
1	Name	Grade			
2	Joe	85			
3	Alice	92			
4	Joe	90			

The result of =UNIQUE2(A1:B4,"Name") is:

Name	Grade
Joe	85
Alice	92

Notes

- UNIQUE2() is similar to DISTINCTROWS2(). DISTINCTROWS2() uses all the values in a row to determine whether the row is different from other rows, but UNIQUE2() uses only the values in the specified columns. UNIQUE2() filters the range so that it returns only unique rows based on the values in the specified columns. The first row it considers unique is the one it returns; it doesn't return subsequent duplicate rows.
- This function is intended for use in array formulas.

Related Functions

DISTINCTROWS2

UNITS

Description

Returns the units part of a number as a string. If the number does not have a units value, the function returns an empty string.

Syntax

```
UNITS(value)
```

- value: The number to return the unit of measurement for.

Example

Formula	Result
=UNITS(A1)	cup

Formula	Result
where the value in A1 currently has the "cup" unit)	

UPLUS

Description

Returns the specified number without changing it.

Syntax

`UPLUS(number)`

- number: The number to return.

Example

Formula	Result
=UPLUS(A1) Where cell A1 contains `-2` .	-2

UPPER

Description

Converts the specified text to uppercase.

Syntax

`UPPER(text)`

- text: The text to convert.

Example

Formula	Result
=UPPER("Regional Sales Manager")	REGIONAL SALES MANAGER
=UPPER(A1) Where cell A1 contains `Regional Sales Manager` .	REGIONAL SALES MANAGER

URL

Description

Converts a URL into a clickable link on a sheet.

Syntax

`URL(url, [ label], [tooltip])`

- url: The URL. For linking within a workbook, precede the path with a pound sign (#).
- label: The clickable text to display in the cell. If you don't include this argument, the function uses the url argument as the clickable text. You can use a cell reference to contain the label.

- tooltip: The text to display when the user hovers the cursor over the cell. If you don't include this argument, the function uses the label argument as the clickable text. You can use a cell reference to contain the tooltip.

Example

Formula	Result
=URL("http://www.workday.com","Workday")	Workday (as a clickable link)
=URL("http://www.workday.com","Workday","Open link in new browser window")	Workday (as a clickable link), with a tooltip of "Open link in new browser window"
=URL("#"&REMOVEROWS(INFO("wd.sheet.names"),1))	Creates a clickable list of links to the sheets of the workbook; this essentially provides a Table of Contents of the workbook tabs.

Notes

- This function is similar to the HYPERLINK function; however, to maintain compatibility with Excel, HYPERLINK doesn't have the optional tooltip argument.
- Keep in mind that if you use the right-click (context) menu in a Microsoft® Excel® workbook to create a hyperlink, and then you upload the workbook to Worksheets, it isn't converted into a URL function; it becomes a link-formatted cell containing a URL.

Related Functions

HYPERLINK

URLTEXT

Description

Returns, as a string, the anchor text component of a URL.

Syntax

`URLTEXT(url)`

- url: The URL, typically the output of the URL function.

Example

Formula	Result
=URLTEXT(URL("http://www.workday.com","Workday"))	Workday (as plain text)

VALUE

Description

Converts the specified text into a numeric value. The value to convert must be in a recognized date, time, or number format.

Syntax

`VALUE(text)`

- text: The text to convert.

Example

Formula	Result
=VALUE(A1) Where cell A1 contains `$2,451.00` .	2451
=VALUE(A1) Where cell A1 contains `Regional Sales Manager` .	#VALUE

VALUEAT

Description

Returns the value at the intersection of a column header and row label.

Syntax

`VALUEAT(table, column_name, row_name)`

- table: The table.
- column_name: The column name.
- row_name: The row name.

Example

Results are based on this spreadsheet:

	A	B
1	Name	Grade
2	Joe	85
3	Alice	92
4	Luis	90

Formula	Result
=VALUEAT(A1:B4,"Grade","Alice")	92

Related Functions

INDEX

MATCHEXACT

VALUETOTEXT

Description

Converts numbers, dates, and other values to text.

This function is useful for ensuring consistency in text-based operations or formatting numbers and dates as text.

Syntax

```
VALUETOTEXT(value, [format])
```

- value: The value to return as text.
- format: The format of the returned data. Options include:
  - 0 (default): Concise format that is easy to read.
  - 1: Strict format that includes escape characters and row delimiters.

VAR

Description

Returns the variance based on a sample.

Syntax

```
VAR(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

VAR.P

Description

Returns the variance based on the total population.

Syntax

```
VAR.P(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

VARA

Description

Returns the variance based on a sample. Similar to VAR(), but in VARA() you can include text and logical values. A text value or logical FALSE value is treated as a 0; a logical TRUE is counted as a 1.

Syntax

```
VARA(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

VARP

Description

Returns the variance based on the total population.

Syntax

```
VARP(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

VARPA

Description

Returns the variance based on the total population. Similar to VARP(), but in VARPA() you can include text and logical values. A text value or logical FALSE value is treated as a 0; a logical TRUE is counted as a 1.

Syntax

```
VARPA(number1, [ number2], ...)
```

- number1: The number or list of numbers.
- number2: The number or list of numbers.

VDB

Description

Returns the depreciation of an asset, based on the double declining balance method or a different specified method, for a specified period.

Syntax

```
VDB(cost, salvage, life, start_period, end_period, [ factor], [ no_switch])
```

- cost: The initial cost of the asset.
- salvage: The value of the asset after depreciation.
- life: The number of periods to depreciate the asset over.
- start_period: The starting period to calculate the depreciation for.
- end_period: The ending period to calculate the depreciation for.
- factor: The rate of decline for the balance. The default is 2.
- no_switch: Specifies whether to switch to straight line depreciation when depreciation is more than the declining balance result. 0 or Empty = switch; 1 = do not switch.

VLOOKUP

Description

Finds a value in the leftmost column of the specified array, and returns the value for the corresponding cell in the same row in a different column.

Syntax

```
VLOOKUP(lookup_value, table_array, col_index_num, [ range_lookup])
```

- lookup_value: The value to find in the first column.
- table_array: The array or table to search.
- col_index_num: The column to return the corresponding value from.
- range_lookup: Specifies whether to require an exact match. FALSE = return an error if the function doesn't find an exact match; TRUE or Empty = if the function doesn't find an exact match, use the closest match below the lookup_value as a match.

Related Functions

ARRAYAREA

GROUPBY

MATCHEXACT

MVLOOKUP

VSTACK

Description

Appends the specified arrays into a single array as a sequential set of rows.

Syntax

`VSTACK(array1,[array2],...)`

- range1: The array to append (stack).
- range2: The array to append (stack).

WD.ARRANGECOLUMNS

Description

Creates a new range from an existing range, with the columns ordered according to the specified indexes. You can add an empty column by including a null index value. Example: `=WD.ARRANGECOLUMNS([range],1,2,3,,4)` inserts a blank column between the 3rd index value and 4th index value.

Syntax

`WD.ARRANGECOLUMNS(range, column_indexes, ...)`

- range: The range to create the new range from.
- column_indexes: The column indexes. The indexes must be in the range of 1 to the number of columns in the range.

Example

The example is based on this range:

	A	B
1	4	3
2	8	9
3	14	11

The formula =WD.ARRANGECOLUMNS(A1:B3,1,1,2,2) in cell A5 has these results:

	A	B	C	D
5	4	4	3	3
6	8	8	9	9
7	14	14	11	11

Notes

- The index values can repeat, if desired, and can be in any order.
- This function is intended for use in array formulas.

WD.ARRANGEROWS

Description

Creates a new range from an existing range, with the rows ordered according to the specified indexes. You can add an empty row by including a null index value. Example: `=WD.ARRANGEROWS([range],1,2,3,,4)` inserts a blank row between the 3rd index value and 4th index value.

Syntax

```
WD.ARRANGEROWS(range, row_indexes, ...)
```

- range: The range to create the new range from.
- row_indexes: The row indexes. The indexes must be in the range of 1 to the number of rows in the range.

Example

The example is based on this range:

	A	B
1	4	3
2	8	9
3	14	11

The formula =WD.ARRANGEROWS(A1:B3,1,1,2,2,3,3) submitted in cell A5 has these results:

	A	B
5	4	3
6	4	3
7	8	9
8	8	9
9	14	11
10	14	11

Notes

- The index values can repeat, if desired, and can be in any order.

- This function is intended for use in array formulas.

WD.AVERAGEIF

Description

Returns the average of the values that meet the specified condition (criterion). This function is the same as AVERAGEIF but WD.AVERAGEIF creates an index (b-tree) for the specified range or array. This function provides better performance for repeated use on the same range/array. Null and error values are ignored.

Syntax

```
WD.AVERAGEIF(range, criteria, [ average_range])
```

- range: The values, or the range/array of cells, to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.
- average_range: The values, or the range of cells, to be averaged if the range meets the specified condition. If not specified, the range is averaged.

WD.AVERAGEIFS

Description

Returns the average of the values that meet multiple conditions (criteria). This function is the same as AVERAGEIFS but WD.AVERAGEIFS creates an index (b-tree) for the specified ranges/arrays. This function provides better performance for repeated use on the same range/array. Null and error values are ignored.

Syntax

```
WD.AVERAGEIFS(average_range, range, criteria, [ range2, criteria2], ...)
```

- average_range: The values, or the range of cells, to be averaged if the specified ranges meet the specified conditions.
- range: The values, or the range/array of cells, to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.
- range2: The values, or the range of cells, to be evaluated according to the condition.
- criteria2: The condition to use to evaluate the values.

WD.COUNTIF

Description

Returns the number of specified values that meet the condition (criterion). You can use the * wildcard to match string values. This function is the same as COUNTIF but WD.COUNTIF creates an index (b-tree) for the specified range or array, providing better performance for repeated use on the same range/array. Null and error values are ignored.

Syntax

```
WD.COUNTIF(range, criteria)
```

- range: The range or array of values to be counted if the condition is met.
- criteria: The condition to use to evaluate the values.

WD.COUNTIFS

Description

Returns the number of times that the cells in the specified ranges or arrays meet multiple conditions (criteria). You can use the * wildcard to match string values. This function is the same as COUNTIFS but WD.COUNTIFS creates an index (b-tree) for the specified ranges or arrays, providing better performance for repeated use on the same range/array. Null and error values are ignored.

Syntax

```
WD.COUNTIFS(range, criteria, [ range2, criteria2], ...)
```

- range: The range/array of values to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.
- range2: The value, or list of values, to be evaluated according to the condition.
- criteria2: The condition to use to evaluate the values.

WD.DATEDIF

Description

Returns the amount of time between two dates, based on the specified period and its corresponding unit. We recommend using WD.DATEDIF instead of DATEDIF; we provide DATEDIF for compatibility but it calculates incorrect results in some situations. Note that Worksheets doesn't support submitting non-English values for the arguments.

Syntax

```
WD.DATEDIF(start_date, end_date, period, unit)
```

- start_date: The starting date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- end_date: The ending date for the count. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- period: The number of whole periods between start_date and end_date. If the period is a grouping of values, such as YMD, the function calculates all the whole periods as segments; you use the unit argument to specify which number (segment) to return. Example: If the start_date is January 1,

2010, and the end_date is February 3, 2020, then the function calculates that Year=10, Month=1 and Day=2. If the unit=Month, the function returns 1.

  - Year (Y)
  - Month (M)
  - Week (W)
  - Day (D)
  - Hour (H)
  - Minute (M)
  - Second (Sec or S)
  - Milli (Mil or MS)
  - YearMonthWeekDayTime (YMWDT): Years, months, weeks, days, hours, minutes, seconds, and milliseconds.
  - YearMonthDayTime (YMDT): Years, months, days, hours, minutes, seconds, and milliseconds
  - YearMonthDay (YMD): Years, months, and days
  - YearWeekDayTime (YMDT): Years, weeks, days, hours, minutes, seconds, and milliseconds
  - YearWeekDay (YWD): Years, weeks, and days
  - YearDayTime (YDT): Years, days, hours, minutes, seconds, and milliseconds
  - YearDay (YD): Years and days
  - DayTime (DT): Days, hours, minutes, seconds, and milliseconds
  - Time (T): Hours, minutes, seconds, and milliseconds
- unit: The number to return. If the period is a single value, such as Day, the unit must match the period. If the period is a grouped value, such as YMD, the unit must match one of the segments in the period.
  - Year (Y)
  - Month (M)
  - Week (W)
  - Day (D)
  - Hour (H)
  - Minute (Min)
  - Second (S or Sec)
  - Milli (or MS or Mil)

Example

The examples are based on this data:

	A	B	C	D
1	Start Date	End Date	Period	Unit
2	07/02/2018 05:17:33.636 PM	03/31/2020 12:00:00.000 AM	Year	Year
3			Month	Month
4			Week	Week
5			Day	Day
6			YMD	M

Here are some example formulas and results:

Formula	Result
=WD.DATEDIF(A2,B2,C2,D2)	1
=WD.DATEDIF(A2,B2,C3,D3)	20
=WD.DATEDIF(A2,B2,C4,D4)	91
=WD.DATEDIF(A2,B2,C5,D5)	637
=WD.DATEDIF(A2,B2,C6,D6)	8

In a more complex example, you can specify several unit values using a range of cells to calculate the same number of results.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Date 1	Date 2	Period	Unit									
2	7/2/2018	3/31/2020	YMD	Y	M	D							
3				1	8	28	Formula in D3:F3 is =WD.DATEDIF(A2,B2,C2,D2:F2)						

Place the formula =WD.DATEDIF(A2,B2,C2,D2:F2) in cell D3 and submit it as an unconstrained array formula using Ctrl+Alt+Enter (Window) or Option+Alt+Enter (Mac).

WD.EXCHANGE

Description

Converts a value to a value with a currency unit, according to the exchange rate you specify. The function multiplies the numeric value of value by the numeric value of exchange_rate_value, setting the units to the unit value of exchange_rate_value if specified, or to the unit value of exchange_rate_unit if present.

Syntax

```
WD.EXCHANGE(value, exchange_rate_value,[exchange_rate_unit])
```

- value: The value to convert. The value can be a simple number or a number with a currency unit. The function ignores the currency unit portion of the argument if you included it.
- exchange_rate_value: The exchange rate value. The value can be a simple number or a number with a currency unit. If you specify a simple number, you must also specify the argument exchange_rate_unit. If you specify a number with a currency unit, the function uses the specified unit unless you specified the optional exchange_rate_unit argument.
- exchange_rate_unit: The exchange rate currency unit as a string or as the units from a value with a currency unit. The function ignores the value portion of the argument if you included it.

WD.FULLTOHALF

Description

Converts full-width characters to half-width characters.

Syntax

```
WD.FULLTOHALF(text)
```

- text: The text to convert.

Examples

=WD.FULLTOHALF(" エクセル") returns " ｴｸｾﾙ"

=WD.HALFTOFULL("ｴｸｾﾙ") returns "エクセル"

WD.GROUPBY

Description

Summarizes data by grouping, aggregating, sorting, and filtering based on specified fields. This function is the same as MS.GROUPBY but *with* a style parameter.

It is useful for creating summary tables and performing aggregations similar to a pivot table but directly within a formula.

Syntax

```
WD.GROUPBY(row_fields, values, function, [field_headers], [total_depth],
[sort_order], [filter_array], [field_relationship], [style])
```

- row_fields: The range of values to group.
- values: The values to aggregate.
- function: The calculation to run when aggregating values.
- field_headers: A number that specifies whether the data has headers and whether you want to include them in the results. Options include:
    - Omitted value (default): Headers are not shown.
    - 0: No headers.
    - 1: Yes, but don't show headers.
    - 2: No headers, but generate them.
    - 3: Yes, and show headers.
- total_depth: A number that specifies whether to show totals and subtotals. For subtotals, row_fields must have at least 2 columns. Options include:
    - Omitted value (default): Grand totals and, where possible, subtotals.
    - 0: No totals.
    - 1: Grand totals at the bottom.
    - 2: Grand totals and subtotals at the bottom.
    - -1: Grand totals at the top.
    - -2: Grand totals and subtotals at the top.
- sort_order: A number that indicates how rows should be sorted. Numbers correspond to columns in row_fields, followed by columns in values. A positive number sorts rows in ascending order. A negative number sorts rows in descending order.
- filter_array: A column-oriented array of booleans that indicates which rows should be included or excluded from the output.
- field_relationship: A number that specifies the relationship between columns when multiple columns are provided as row_fields and affects the sorting of the results. Options include:
    - 0: Hierarchy (default) - Sorts row_fields from left to right where the sorting of later columns takes into account the hierarchy of earlier columns.
    - 1: Table - Sorts each column independently.
- style: The way the results display. See the Table Style drop-down list in the Pivot Settings dialog for a list of style options.

WD.HALFTOFULL

Description

Converts half-width characters to full-width characters.

Syntax

```
WD.HALFTOFULL(text)
```

- text: The text to convert.

Examples

=WD.FULLTOHALF(" エクセル") returns " ｴｸｾﾙ"

=WD.HALFTOFULL("ｴｸｾﾙ") returns "エクセル"

WD.LIST.GET

Description

Returns a single element from a list, based on the specified index.

Syntax

`WD.LIST.GET(list, index)`

- list: The list to return the element from.
- index: The position of the element in the list.

Example

The formula =list.get({10,9,8,7},2) returns the value: 9

The example below uses the values in this table:

	A	B	C
1	jan	feb	mar
2	apr	may	jun
3	jul	aug	sept

The formula =LIST.GET(A2:C2,2) returns the result: may.

Related Functions

WD.LIST.LIST

WD.LIST.SIZE

WD.LIST.LIST

Description

Creates a list based on the specified values.

Syntax

`WD.LIST.LIST(value)`

- value: The values to place in the list.

Example

The formula =list.list({10,9,8,7}) returns a 4 item list.

The example below uses the values in this table:

	A	B	C
1	jan	feb	mar
2	apr	may	jun
3	jul	aug	sept

The formula =LIST.LIST(A1:C1) returns the values jan, feb, and mar across 3 cells.

Related Functions

WD.LIST.GET

WD.LIST.SIZE

WD.LIST.SIZE

Description

Returns the number of elements in the specified list.

Syntax

`WD.LIST.SIZE(list)`

- list: The list to return the size from.

Example

The formula =list.size({10,9,8,7}) returns the value: 4.

The example below uses the values in this table:

	A	B	C
1	jan	feb	mar
2	apr	may	jun
3	jul	aug	sept

The formula =LIST.SIZE(A2:C2) returns the result: 3.

Related Functions

WD.LIST.GET

WD.LIST.LIST

WD.LIVEDATA

Description

Returns live data (report) metadata based on the reference and option that you specify. Remember to submit the formula as an unconstrained formula using Ctrl+Alt+Enter (Windows) or Command+Option +Enter (Mac).

Syntax

```
(reference, option)
```

- reference: The anchor cell (top left cell) of the live data range.
- option: The metadata to return. Valid options are 0-4 , 100-104, and 10-29. If there are no results for the option, the function returns an error. Example: If you select option 3 (report prompts used) and there are no prompts in the report, the function returns the #N/A error. For detailed descriptions of the options, see the Function Reference in the User Guide (Help > Function Reference).

Function Options and Descriptions

Option	Description	Notes
100	Report metadata (all). With header.	All metadata, as described in the rest of this table.
101	Report column metadata: Name, Alias, Type. With header.	Type values: STRING, STRINGS, STRING_RICH, NUMERIC, NUMERIC_UNITS, CURRENCY, DATE, DATES, DATE_TIME, TIME, BOOLEAN, BOOLEANS, URL, URLS, INSTANCE, INSTANCES
102	Columns included: Name, Alias, Type, Editable, Key. With header.	Type values: COLUMN, EXPRESSION, FORMULA, NOTE Editable values: TRUE/FALSE Key values: TRUE/FALSE
103	Report prompts used: Name, Alias, Type. With header.	Type values: STRING, STRINGS, STRING_RICH, NUMERIC, NUMERIC_UNITS, CURRENCY, DATE, DATES, DATE_TIME, TIME, BOOLEAN, BOOLEANS, URL, URLS, INSTANCE, INSTANCES
104	Report prompts with settings: Name, Alias, Type, Value, PromptType. With header.	Type values: STRING, STRINGS, STRING_RICH, NUMERIC, NUMERIC_UNITS, CURRENCY, DATE, DATES, DATE_TIME, TIME, BOOLEAN, BOOLEANS, URL, URLS, INSTANCE, INSTANCES Prompt Type values: AS_SPECIFIED, DO_NOT_USE, USE_DEFAULT

Option	Description	Notes
0	Report metadata (all). No header.	All metadata, as described in the rest of this table.
1	Report column metadata: Name, Alias, Type. No header.	Type values: STRING, STRINGS, STRING_RICH, NUMERIC, NUMERIC_UNITS, CURRENCY, DATE, DATES, DATE_TIME, TIME, BOOLEAN, BOOLEANS, URL, URLS, INSTANCE, INSTANCES
2	Columns included: Name, Alias, Type, Editable, Key. No header.	Type values: COLUMN, EXPRESSION, FORMULA, NOTE Editable values: TRUE/FALSE Key values: TRUE/FALSE
3	Report prompts used: Name, Alias, Type. No header.	Type values: STRING, STRINGS, STRING_RICH, NUMERIC, NUMERIC_UNITS, CURRENCY, DATE, DATES, DATE_TIME, TIME, BOOLEAN, BOOLEANS, URL, URLS, INSTANCE, INSTANCES
4	Report prompts with settings: Name, Alias, Type, Value, PromptType. No header.	Type values: STRING, STRINGS, STRING_RICH, NUMERIC, NUMERIC_UNITS, CURRENCY, DATE, DATES, DATE_TIME, TIME, BOOLEAN, BOOLEANS, URL, URLS, INSTANCE, INSTANCES Prompt Type values: AS_SPECIFIED, DO_NOT_USE, USE_DEFAULT
10	Workbook name	
11	Sheet name	
12	Cell	Root cell of live data
13	Report name	
14	Report alias	Workday ID
15	Report description	
16	Report type	ADVANCED_REPORT, MATRIX_REPORT, COMPOSITE_REPORT
17	Report async status	TRUE, FALSE
18	Report limit	
19	Report key column count	
20	Report multi-instance enabled	TRUE, FALSE

Option	Description	Notes
21	Report highlighted status	TRUE, FALSE
22	Table range name	
23	Formula user	
24	Last run user	
25	Last run date	The date displays as a serial number; you can use Format > Number > Date to display the result using date formatting.
26	Last run time	In milliseconds
27	Last run row count	
28	Next run user	Applicable if a refresh schedule exists
29	Next run date	Applicable if a refresh schedule exists
30	Schedule start date	
31	Schedule end date	
32	Schedule type	
33	Schedule repetition interval	
34	Schedule data in cron expression format	

Examples

=TRANSPOSE((A1,100)) returns all metadata and includes a header. The TRANSPOSE function puts the headers in a column for improved readability.

WorkbookName	Workbook Example
SheetName	Sheet1
Cell	A1
ReportName	Workday Report Example
ReportAlias	[Workday ID]
ReportDescription	
ReportType	ADVANCED_REPORT
ReportAsync	FALSE
ReportLimit	0
ReportKeyColumnCount	1
ReportMultiInstanceEnabled	TRUE
ReportHighlightEnabled	TRUE
ReportTableName	
FormulaUser	Logan McNeil

LastRunUser	Logan McNeil
LastRunDate	05/05/2021 11:09:17.928 PM
LastRunTime	426
LastRunRowCount	34
NextRunUser	Logan McNeil
NextRunDate	06/01/2021 7:00:00.000 AM

=(A1,101) returns column data information and includes a header.

Name	Alias	Type
Worker Name	[XML Alias]	INSTANCE
Dependents	[XML Alias]	INSTANCES

=(A1,102) returns report column details and includes a header.

Name	Alias	Type	Editable	Key
Employee ID	[XML Alias]	COLUMN	FALSE	TRUE
Worker Name	[XML Alias]	COLUMN	FALSE	FALSE
Dependents	[XML Alias]	COLUMN	FALSE	FALSE
Note1	[XML Alias]	NOTE	TRUE	FALSE
Formula1	[XML Alias]	FORMULA	TRUE	FALSE

=(A1,103) returns report prompt information and includes a header.

Name	Alias	Type
Effective Date	[XML Alias]	DATE
Cost Centers	[XML Alias]	INSTANCES

=(A1,104) returns report prompt details and includes a header.

Name	Alias	Type	Value	PromptType
Effective Date	[XML Alias]	DATE	4/15/2021	USE_DEFAULT
Cost Centers	[XML Alias]	INSTANCES	Value	DO_NOT_USE

Notes

- This function is intended for use in array formulas. You submit array formulas with special keyboard shortcuts. We recommend that you use the unconstrained keyboard shortcut Ctrl+Alt+Enter (Windows) or Command+Option+Enter (Mac) so Worksheets can use all the cells it needs for the results.

WD.MAXIF

Description

Returns the maximum value from a range or array of values, according to a criterion. This function creates an index (b-tree) for the specified range or array, providing better performance for repeated use on the same range/array. Null and error values are ignored.

Syntax

```
WD.MAXIF(range, criteria, max_range)
```

- range: The values, or the range/array of cells, to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.
- max_range: The array of numeric values that you want to return the maximum value from, if the criteria are met.

Notes

- The condition can be one of these:
  - A numeric value (integer, decimal, date, time, or logical value), such as 5, 12/5/2019, or TRUE.
  - A text string, such as "Name" or "November".
  - An expression, such as ">9" or "<>0".
- The function isn't case-sensitive. Example: When evaluating the values in range against the criteria, the text strings "MONTH" and "month" are considered a match.
- Whenever you use a text string or an expression as a condition, surround it with quotes.
- In text-based criteria, you can use the ? wildcard to match any single character or the * wildcard to match any sequence of characters.

WD.MAXIFS

Description

Returns the maximum value from a range of values, according to one or more criteria. This function is the same as MAXIFS but WD.MAXIFS creates an index (b-tree) for the specified ranges/arrays. This function provides better performance for repeated use on the same ranges/arrays. Null and error values are ignored.

Syntax

```
WD.MAXIFS(max_range, range, criteria, [ range2], [ criteria2], ...)
```

- max_range: The array of numeric values that you want to return the maximum value from, if the criteria are met.
- range: The range or array of values to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.
- range2: The second set of values, or the range of cells, to be evaluated according to the condition.
- criteria2: A second condition to use to evaluate the values.

Notes

- The condition can be one of these:
  - A numeric value (integer, decimal, date, time, or logical value), such as 5, 12/5/2019, or TRUE.
  - A text string, such as "Name" or "November".
  - An expression, such as ">9" or "<>0".
- The function isn't case-sensitive. Example: When evaluating the values in range against the criteria, the text strings "MONTH" and "month" are considered a match.
- Whenever you use a text string or an expression as a condition, surround it with quotes.
- In text-based criteria, you can use the ? wildcard to match any single character or the * wildcard to match any sequence of characters.

WD.MINIF

Description

Returns the minimum value from a range or array of values, according to a criterion. This function creates an index (b-tree) for the specified range or array, providing better performance for repeated use on the same range/array. Null and error values are ignored.

Syntax

```
WD.MINIF(min_range, range, criteria)
```

- min_range: The array of numeric values that you want to return the minimum value from, if the criteria are met.
- range: The range or array of values to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.

Notes

- The condition can be one of these:
  - A numeric value (integer, decimal, date, time, or logical value), such as 5, 12/5/2019, or TRUE.
  - A text string, such as "Name" or "November".
  - An expression, such as ">9" or "<>0".
- The function isn't case-sensitive. Example: When evaluating the values in range against the criteria, the text strings "MONTH" and "month" are considered a match.
- Whenever you use a text string or an expression as a condition, surround it with quotes.
- In text-based criteria, you can use the ? wildcard to match any single character or the * wildcard to match any sequence of characters.

WD.MINIFS

Description

Returns the minimum value from a range or array of values, according to one or more criteria. This function is the same as MINIFS but WD.MINIFS creates an index (b-tree) for the specified ranges or arrays. This function provides better performance for repeated use on the same range/array. Null and error values are ignored.

Syntax

```
WD.MINIFS(min_range, range, criteria, [ range2], [ criteria2], ...)
```

- min_range: The array of numeric values that you want to return the minimum value from, if the criteria are met.
- range: The values, or the range/array of cells, to be evaluated according to the condition.
- criteria: The condition to use to evaluate the values.
- range2: The second set of values, or the range of cells, to be evaluated according to the condition.
- criteria2: A second condition to use to evaluate the values.

Notes

- The condition can be one of these:
  - A numeric value (integer, decimal, date, time, or logical value), such as 5, 12/5/2019, or TRUE.
  - A text string, such as "Name" or "November".
  - An expression, such as ">9" or "<>0".
- The function isn't case-sensitive. Example: When evaluating the values in range against the criteria, the text strings "MONTH" and "month" are considered a match.
- Whenever you use a text string or an expression as a condition, surround it with quotes.
- In text-based criteria, you can use the ? wildcard to match any single character or the * wildcard to match any sequence of characters.

WD.MVLOOKUP

Description

This function is the same as MVLOOKUP but WD.MVLOOKUP creates an index (b-tree) for the specified range or array, and it ignores the order argument. This function provides better performance for repeated use on the same range/array. Null and error values are ignored. We recommend this function as a replacement for VLOOKUP, especially when you're working with live data. WD.MVLOOKUP performs a vertical (column) lookup on a table and returns all matches. WD.MVLOOKUP is similar to VLOOKUP, but:

- VLOOKUP scans only the left column for matches; in WD.MVLOOKUP you specify the column to search.
- VLOOKUP stops after finding 1 match, and returns a single cell value; WD.MVLOOKUP scans the entire lookup column for matches. Each match in that column results in a new row in the output. If you specify 4 return_column_index values, then each resulting row will have 4 columns.

Syntax

```
WD.MVLOOKUP(lookup_value, table_array, lookup_column_index, order,
return_column_index, ...)
```

- lookup_value: The value to match.
- table_array: The array or table to search.
- lookup_column_index: The column in the table to search (1-based).
- order: Not used.
- return_column_index: The column number to return results from. You can list any number of column numbers.

Example

The example is based on this workbook:

	A	B	C	D
1	Salesperson	Customer	Product	Revenue

	A	B	C	D
2	Dixon	O'Reilly, Auer, & Lind	Qosolex	$8,232,000.00
3	Dixon	Runolfsson and Steuber	Saoplus	$4,853,000.00
4	Kelly	Kuphal Group	Qosolex	$2,358,000.00
5	Kelly	Ernser Inc	Voltflarn	$2,064,000.00
6	Payne	Williamson Group	Singlflix	$6,974,000.00
7	Payne	Ratke-Sanford	Qosolex	$8,433,000.00
8	Payne	Leuschke and Sons	Qosolex	$8,181,000.00
9	Wu	Dach-Halvorson	Singlflix	$2,361,000.00
10	Wu	Wisoky LLC	Bextain	$5,752,000.00
11	Wu	Stiedemann Grp	Saoplus	$3,987,000.00

The result of:

`=WD.MVLOOKUP(A6,$A$2:$D$11,1,0,2,3,4)`

or alternatively, `=WD.MVLOOKUP("Payne",$A$2:$D$11,1,0,2,3,4)`

is:

Williamson Group	Singlflix	$6,974,000.00
Ratke-Sanford	Qosolex	$8,433,000.00
Leuschke and Sons	Qosolex	$8,181,000.00

Notes

- WD.MVLOOKUP() scans the rows in column lookup_column_index for lookup_value. The function then gathers values in the columns listed in return_column_index. Then it places each value into a column in the resulting matrix.
- This function is intended for use in array formulas.

Related Functions

MATCHEXACT

MHLOOKUP

WD.PIVOTBY

Description

Summarizes data by grouping, aggregating, sorting, and filtering based on specified row and column fields. This function is the same as PIVOTBY but *with* a style parameter.

This function is useful for dynamic cross-tabulation and advanced data summarization directly within a formula.

Syntax

```
WD.PIVOTBY(row_fields, col_fields, values, function, [field_headers],
[row_total_depth], [row_sort_order], [col_total_depth], [col_sort_order],
[filter_array], [relative_to], [style])
```

- row_fields: The values to use when grouping rows.
- col_fields: The values to use when grouping columns.
- values: The values to aggregate.
- function: The calculation to run when aggregating.
- field_headers: A number that specifies whether the data has headers and whether you want to include them in the results. Options include:
    - Omitted value (default): Headers are not shown.
    - 0: No headers.
    - 1: Yes, but don't show headers.
    - 2: No headers, but generate them.
    - 3: Yes, and show headers.
- row_total_depth: A number that specifies whether to show totals and subtotals. For subtotals, row_fields must have at least 2 columns. Options include:
    - Omitted value (default): Grand totals and, where possible, subtotals.
    - 0: No totals.
    - 1: Grand totals at the bottom.
    - 2: Grand totals and subtotals at the bottom.
    - -1: Grand totals at the top.
    - -2: Grand totals and subtotals at the top.
- row_sort_order: A number that indicates how rows should be sorted. Numbers correspond to columns in row_fields, followed by columns in values. A positive number sorts rows in ascending order. A negative number sorts rows in descending order.
- col_total_depth: A number that specifies whether to show totals and subtotals. For subtotals, col_fields must have at least 2 columns. Options include:
    - Omitted value (default): Grand totals and, where possible, subtotals.
    - 0: No totals.
    - 1: Grand totals on the right.
    - 2: Grand totals and subtotals on the right.
    - -1: Grand totals on the left.
    - -2: Grand totals and subtotals on the left.
- col_sort_order: A number that indicates how columns should be sorted. Numbers correspond to columns in col_fields, followed by columns in values. A positive number sorts rows in ascending order. A negative number sorts rows in descending order.
- filter_array: A column-oriented array of booleans that indicates which rows should be included or excluded from the output.
- relative_to: A number that indicates where to find values for the 2nd argument of an aggregation function. This is typically used when PERCENTOF is supplied to function. Options include:
    - 0 (default): Column totals.
    - 1: Row totals.
    - 2: Grand total.
    - 3: Parent column total.
    - 4: Parent row total.
- style: The way the results display. See the Table Style drop-down list in the Pivot Settings dialog for a list of style options.

WD.SLICE

Description

This function enables you to extract and return a segment of a specified array; it returns the extracted (sliced) array. Optionally you can specify indexes. Specify positive indexes to increment from the beginning of the array, or specify negative indexes to increment from the end of the array (reverse the array). The indexes are 1-based. The index value 0 isn't allowed.

Syntax

```
WD.SLICE(array, [ row_start], [ row_end], [ row_step], [ column_start], [
column_end], [ column_step])
```

- array: The array to slice.
- row_start: The starting row index. If omitted, the default value is 1.
- row_end: The ending row index. If omitted, the default is the current number of rows in the specified range.
- row_step: The row step index. If omitted, the default value is 1.
- column_start: The starting column index. If omitted, the default value is 1.
- column_end: The ending column index. If omitted, the default is the current number of columns in the specified range.
- column_step: The column step index. If omitted, the default value is 1.

WD.SUMIF

Description

Adds the numbers specified as arguments, if a condition is met. This function is the same as SUMIF but WD.SUMIF creates an index (b-tree) for the specified range or array. This function provides better performance for repeated use on the same range/array. Null and error values are ignored.

Syntax

```
WD.SUMIF(range, criteria, [ sum_range])
```

- range: The range/array of values to test.
- criteria: The condition to test each value against.
- sum_range: The range of values, or cells, to add together. If not specified, the values in the range argument are summed.

WD.SUMIFS

Description

Adds the numbers specified in the sum_range, if a set of criteria are met. This function is the same as SUMIFS but WD.SUMIFS creates an index (b-tree) for the specified range or array. This function provides better performance for repeated use on the same range/array. Null and error values are ignored.

Syntax

```
WD.SUMIFS(sum_range, range, criteria, [ range2, criteria2], ...)
```

- sum_range: The range of values, or values in cells, to sum if all criteria are met.
- range: The range or array of values to test.
- criteria: The condition to test each value against.
- range2: The range of values, or cells, to test.

- criteria2: The condition to test each value against.

WD.VLOOKUP

Description

Finds a value in the leftmost column of the specified array, and returns the value for the corresponding cell in the same row in a different column. This function is the same as VLOOKUP but WD.VLOOKUP creates an index (b-tree) for the specified ranges/arrays, and it ignores the range_lookup argument. This function provides better performance for repeated use on the same range/array. Null and error values are ignored.

Syntax

```
WD.VLOOKUP(lookup_value, table_array, col_index_num, [ range_lookup])
```

- lookup_value: The value to find in the first column.
- table_array: The array or table to search.
- col_index_num: The column to return the corresponding value from.
- range_lookup: Not used.

Example

The example is based on this workbook:

	A	B	C	D
1	Salesperson	Customer	Product	Revenue
2	Dixon	O'Reilly, Auer, & Lind	Qosolex	$8,232,000.00
3	Dixon	Runolfsson and Steuber	Saoplus	$4,853,000.00
4	Kelly	Kuphal Group	Qosolex	$2,358,000.00
5	Kelly	Ernser Inc	Voltflarn	$2,064,000.00
6	Payne	Williamson Group	Singlflix	$6,974,000.00
7	Payne	Ratke-Sanford	Qosolex	$8,433,000.00
8	Payne	Leuschke and Sons	Qosolex	$8,181,000.00
9	Wu	Dach-Halvorson	Singlflix	$2,361,000.00
10	Wu	Wisoky LLC	Bextain	$5,752,000.00
11	Wu	Stiedemann Grp	Saoplus	$3,987,000.00

The result of `=WD.VLOOKUP(A6,$A$2:$D$11,4)` is `$6,974,000.00` .

Related Functions

ARRAYAREA

GROUPBY

MATCHEXACT

WD.MVLOOKUP

WEEKDAY

Description

Returns an integer representing the day of the week.

Syntax

```
WEEKDAY(date, [ return_type])
```

- date: The serial date to get the day value from.
- return_type: The code specifying the integer to assign to each day of the week. The default return_type is 1. (1 = Sunday, 2 = Monday, and so on.)
    - 1: 1 = Sunday, 2 = Monday, and so on.
    - 2: 1 = Monday, 2 = Tuesday, and so on.
    - 3: 0 = Monday, 1 = Tuesday, and so on.
    - 11: 1 = Monday, 2 = Tuesday, and so on.
    - 12: 1 = Tuesday, 2 = Wednesday, and so on.
    - 13: 1 = Wednesday, 2 = Thursday, and so on.
    - 14: 1 = Thursday, 2 = Friday, and so on.
    - 15: 1 = Friday, 2 = Saturday, and so on.
    - 16: 1 = Saturday, 2 = Sunday, and so on.
    - 17: 1 = Sunday, 2 = Monday, and so on.

Example

Formula	Result
=WEEKDAY("12/31/2016 5:37 PM",1)	7
=WEEKDAY("12/31/2016",3)	5

Notes

- To format the result as a weekday name instead of a number, select Format > Number > Custom and enter the formatting string ddd .

WEEKNUM

Description

Returns an integer from 1 to 53 representing the week number of the specified date.

Syntax

```
WEEKNUM(serial_date, [ return_type])
```

- serial_date: The serial number representing the date.

- return_type: The code specifying the week numbering system to use and the first day of the week. The default return_type is 1. In week numbering system 1, the week of January 1st is Week 1; in week numbering system 2, the week containing the first Thursday of the year is Week 1.
    - 1: Numbering system 1 and the week runs from Sunday to Saturday.
    - 2: Numbering system 1 and the week runs from Monday to Sunday.
    - 11: Numbering system 1 and the week runs from Monday to Sunday.
    - 12: Numbering system 1 and the week runs from Tuesday to Monday.
    - 13: Numbering system 1 and the week runs from Wednesday to Tuesday.
    - 14: Numbering system 1 and the week runs from Thursday to Wednesday.
    - 15: Numbering system 1 and the week runs from Friday to Thursday.
    - 16: Numbering system 1 and the week runs from Saturday to Friday.
    - 17: Numbering system 1 and the week runs from Sunday to Monday.
    - 21: Numbering system 2 and the week runs from Monday to Sunday.

Example

Formula	Result
=WEEKNUM("10/4/2016 5:37 PM")	41
=WEEKNUM("10/4/2016",21)	40

WEIBULL

Description

Returns either the weibull density function or the weibull cumulative distribution function for the specified arguments.

Syntax

```
WEIBULL(x, alpha, beta, cumulative)
```

- x: The input value.
- alpha: The alpha parameter.
- beta: The beta parameter.
- cumulative: The type of distribution to use. FALSE = Weibull probability density function; TRUE = Weibull cumulative distribution function.

WEIBULL.DIST

Description

Returns either the weibull density function or the weibull cumulative distribution function for the specified arguments.

Syntax

```
WEIBULL.DIST(x, alpha, beta, cumulative)
```

- x: The input value.
- alpha: The alpha parameter.
- beta: The beta parameter.
- cumulative: The type of distribution to use. FALSE = Weibull probability density function; TRUE = Weibull cumulative distribution function.

Example

Formula	Result
=WEIBULL.DIST(129335,54,151456,TRUE)	0.000198223

Notes

- This function does the same action as WEIBULLDIST().
- In a formula with a null entry such as:

  =WEIBULL.DIST(,54,151456,TRUE)

  Worksheets evaluates this formula as having an invalid data type. Microsoft® Excel® evaluates it as if it were sent a zero instead of the null value.

WEIBULLDIST

Description

Returns either the weibull density function or the weibull cumulative distribution function for the specified arguments.

Syntax

```
WEIBULLDIST(x, alpha, beta, cumulative)
```

- x: The input value.
- alpha: The alpha parameter.
- beta: The beta parameter.
- cumulative: The type of distribution to use. FALSE = Weibull probability density function; TRUE = Weibull cumulative distribution function.

Example

Formula	Result
=WEIBULLDIST(129335,54,151456,TRUE)	0.000198223

Notes

- This function does the same action as WEIBULL.DIST().
- In a formula with a null entry such as:

  =WEIBULLDIST(,54,151456,TRUE)

  Worksheets evaluates this formula as having an invalid data type. Microsoft® Excel® evaluates it as if it were sent a zero instead of the null value.

WORKDAY

Description

Returns a number representing the date that is the specified number of work days after the specified start_date.

Syntax

```
WORKDAY(start_date, days, [ holidays])
```

- start_date: The date to start counting from. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- days: The number of work days to add to the start_date.
- holidays: The list of dates to exclude from the work day count. Workday recommends using a cell reference to specify the dates instead of entering them as text, to ensure reliable results.

Example

Formula	Result
=WORKDAY("11/20/2017",30)	1/1/2018
=WORKDAY("11/20/2017",27,{"11/23/2017", "11/24/2017", "12/25/2017"})	1/1/2018

WORKDAY.INTL

Description

Returns a number representing the date that is the specified number of work days after the specified start date. Similar to WORKDAY(), but in WORKDAY.INTL() you can specify which days of the week are weekend days.

Syntax

```
WORKDAY.INTL(start_date, days, [ weekend], [ holidays])
```

- start_date: The date to start counting from. Workday recommends using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.
- days: The number of work days to add to the start_date.
- weekend: The code specifying which days of the week are weekend days. The code can be numerical, as described in the table below, or a string, where a 0 represents a weekday and a 1 represents a weekend day. Example: "0000111" specifies that Friday, Saturday, and Sunday are weekend days.

Code	Description
1	Saturday and Sunday
2	Sunday and Monday
3	Monday and Tuesday
4	Tuesday and Wednesday
5	Wednesday and Thursday
6	Thursday and Friday
7	Friday and Saturday
11	Sunday only
12	Monday only
13	Tuesday only
14	Wednesday only
15	Thursday only
16	Friday only

Code	Description
17	Saturday only

- holidays: The list of dates to exclude from the work day count. Workday recommends using a cell reference to specify the dates instead of entering them as text, to ensure reliable results.

Example

Formula	Result
=WORKDAY.INTL("11/20/2017",30)	1/1/2018
=WORKDAY.INTL("11/20/2017",27,1, {"11/23/2017", "11/24/2017", "12/25/2017"})	1/1/2018
=WORKDAY.INTL("11/20/2017",30,"0000100", {"11/23/2017", "11/24/2017", "12/25/2017"})	12/27/2017

WRAPCOLS

Description

Converts a single row or column into multiple columns, in a new 2-dimensional array, using the maximum number of values for each column that you specify.

Syntax

```
WRAPCOLS(vector, wrap_count, [pad_with])
```

- vector: The vector or reference to wrap.
- wrap_count: The maximum number of values for each column.
- pad_with: The value to pad the column cells with. The default is #N/A.

WRAPROWS

Description

Wraps the specified row or column of values by rows, into a new 2-dimensional array, using the maximum number of values for each row that you specify.

Syntax

```
WRAPROWS(vector, wrap_count, [pad_with])
```

- vector: The vector or reference to wrap.
- wrap_count: The maximum number of values for each row.
- pad_with: The value to pad the result with. If you don't specify the argument, the default is #N/A.
- criteria: The condition to use to evaluate the values.

XIRR

Description

Returns the internal rate of return for a specified series of cash flows that is not necessarily periodic.

Syntax

```
XIRR(values, dates, [ guess])
```

- values: The array or reference to a range of cells containing the cash flow information.
- dates: The dates corresponding to the cash flows.
- guess: The estimated rate of return. The default is 0.1 (10%).

XLOOKUP

Description

Searches an array or range, and returns one or more matching values. XLOOKUP can do exact matching or approximate matching, vertical or horizontal lookups, and wildcard searches. This function is intended for use with unconstrained arrays. To submit the formula; use the unconstrained keyboard shortcut Ctrl+Alt +Enter (Windows) or Command+Option+Enter (Mac). For syntax details, see the Function Reference in the User Guide.

Syntax

```
XLOOKUP(lookup_value, lookup_array, return_array, [ if_not_found], [
match_mode], [ search_mode])
```

- lookup_value: The lookup value.
- lookup_array: The array to search.
- return_array: The array or range to return.
- if_not_found: Text to return, or a formula to run, if the function doesn't find a valid match. If you omit this argument, the function returns an #N/A error.
- match_mode: A number that specifies the type of match.
    - 0: Exact match. If not found, return #N/A. This is the default.
    - -1: Exact match. If not found, return the next smaller item.
    - 1: Exact match. If not found, return the next larger item.
    - 2: A wildcard match. In text-based criteria, you can use the ? wildcard to match any single character or the * wildcard to match any sequence of characters. If the value to find contains a * or ?, precede it with a tilde (~).
- search_mode: A number that specifies the search mode to use.
    - 1: Do a search starting with the first item. This is the default.
    - -1: Do a reverse search starting with the last item.
    - 2: Do a binary search that relies on lookup_array being sorted in ascending order. If it isn't sorted, the results won't be valid.
    - -2: Do a binary search that relies on lookup_array being sorted in descending order. If it isn't sorted, the results won't be valid.

Notes

- XLOOKUP isn't case-sensitive by default (using a match_mode of 0). You can use XLOOKUP for a case-sensitive exact match search only if the array is sorted.

Related Functions

VLOOKUP

HLOOKUP

MHLOOKUP

MVLOOKUP

XMATCH

Description

Searches for a specified item in the list (one-dimensional array) that you specify, and returns the item's index (position). XMATCH does an exact match search by default. You can use this function to match logical values, numeric values, or text strings. This function is intended for use with unconstrained arrays. To submit the formula, use the unconstrained keyboard shortcut Ctrl+Alt+Enter (Windows) or Command +Option+Enter (Mac). For syntax details, see the Function Reference in the User Guide.

Syntax

```
XMATCH(lookup_value, lookup_array, [ match_mode], [ search_mode])
```

- lookup_value: The lookup value.
- lookup_array: The array to search.
- match_mode: A number that specifies the type of match.
    - 0: Exact match. If not found, return #N/A. This is the default.
    - -1: Exact match. If not found, return the next smaller item.
    - 1: Exact match. If not found, return the next larger item.
    - 2: A wildcard match. In text-based criteria, you can use the ? wildcard to match any single character or the * wildcard to match any sequence of characters. If the value to find contains a * or ?, precede it with a tilde (~).
- search_mode: A number that specifies the search mode to use.
    - 1: Do a search starting with the first item. This is the default.
    - -1: Do a reverse search starting with the last item.
    - 2: Do a binary search that relies on lookup_array being sorted in ascending order. If it isn't sorted, the results won't be valid.
    - -2: Do a binary search that relies on lookup_array being sorted in descending order. If it isn't sorted, the results won't be valid.

Notes

- XMATCH isn't case-sensitive by default (using a match_mode of 0). You can use XMATCH for a case-sensitive exact match search only if the array is sorted.

Related Functions

MATCH

MATCHEXACT

XNPV

Description

Returns the net present value based on the specified discount rate, payment/income amounts, and payment dates.

Syntax

```
XNPV(rate, values, dates)
```

- rate: The discount rate.
- values: The array of numeric values representing the payments and income.
- dates: The array of dates that correspond to the payment and income values.

XOR

Description

Returns the logical Exclusive Or result for the specified conditions.

Syntax

`XOR(value1, [ value2], ...)`

- value1: One or more conditions to evaluate.
- value2: One or more conditions to evaluate.

Example

Formula	Result
=XOR(2>1,5<10)	FALSE
=XOR(1<2,5>10)	TRUE
=XOR(1>2,5>10)	FALSE

YEAR

Description

Returns an integer from 1900-9999 representing only the year specified in the date.

Syntax

`YEAR(date)`

- date: The date to get the year value from. Workday recommends entering a serial date or using a cell reference to specify the date instead of entering the date as text, to ensure reliable results.

Example

Formula	Result
=YEAR("12/31/2016 5:37 PM")	2016
=YEAR("31-Dec-2016")	2016

YEARFRAC

Description

Returns the number of days between two specified dates, shown as a fraction of the year.

Syntax

`YEARFRAC(start_date, end_date, [ basis])`

- start_date: The start date.
- end_date: The end date.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

Example

Formula	Result
=YEARFRAC("8/14/2016 05:30 PM","12/31/2016")	0.380555556
=YEARFRAC("8/14/2016","12/31/2016",1)	0.379781421

YIELD

Description

Returns the yield for a security paying periodic interest.

Syntax

```
YIELD(settlement, maturity, rate, price, redemption, frequency, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- rate: The security's annual coupon rate.
- price: The security's price per $100 face value.
- redemption: The security's redemption value per $100 face value.
- frequency: The number of coupon payments per year. 1 = annual; 2 = semiannual; 4 = quarterly.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

YIELDDISC

Description

Returns the annual yield for a discounted security.

Syntax

```
YIELDDISC(settlement, maturity, par, redemption, [ basis])
```

- settlement: The security's settlement date.
- maturity: The security's maturity date.
- par: The security's price per $100 face value.
- redemption: The security's redemption value.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

YIELDMAT

Description

Returns the annual yield for a security paying interest at maturity.

Syntax

```
YIELDMAT(settlement, maturity, issue, rate, price, [ basis])
```

- settlement: The security's settlement date.

- maturity: The security's maturity date.
- issue: The security's issue date.
- rate: The security's interest rate when issued.
- price: The security's price per $100 face value.
- basis: The financial day count to use, shown here using x/y where x = days per month and y = days per year. 0 or Empty = US NASD 30/360; 1 = actual/actual; 2 = actual/360; 3 = actual/365; 4 = European 30/360.

Z.TEST

Description

Returns the one-tailed probability value of a Z-test.

Syntax

```
Z.TEST(number1, x, [ sigma])
```

- number1: The set of numbers to test x against.
- x: The number to test.
- sigma: The standard deviation of the population.

ZTEST

Description

Returns the one-tailed probability value of a Z-test.

Syntax

```
ZTEST(number1, x, [ sigma])
```

- number1: The set of numbers to test x against.
- x: The number to test.
- sigma: The standard deviation of the population.