R&R Report Writer *Technical Bulletin*

Rounding Numbers

Product: R&R Report Writer **Oper Sys:** DOS®
Version: All

Summary

If you create reports that include non-integer values (that is, numbers with decimal places), you probably understand about rounding. But do you understand the difference between using the /Field Format command and the ROUND() function to specify the number of decimal places you want? Do you know when you should use the ROUND() function? Do you know how the ROUND() function works? For those of you who want your reports to be numerically accurate, we offer the following explanation of R&R Report Writer's two methods of rounding numbers: rounding values and rounding display formats.

Rounding Values

To round a numeric value, you use the ROUND() function, which returns a rounded value that is different from the value to which it is applied. For example, you can round the value of the NUMBER field to two decimal places by using the expression ROUND(NUMBER,2). The result of this calculation will be a value different from the value of the NUMBER field. When the value of NUMBER is 5.778, for example, the value of this expression will be 5.78. This rounded value can be displayed in any format.

While many values returned by the ROUND function are what you would expect if you rounded a number by hand (for example, the example given above), some values may not be what you expect. For example, ROUND(.5115,3) returns .511. The reason for this behavior is that, like the ROUND function in any of the database languages, ROUND does not round the ASCII representation of a number, but the number's internal representation. This internal representation is a binary (that is, base 2) floating point number that is the product of the floating point package used by the language in which an application is written (R&R Report Writer uses Microsoft C's floating point emulator package).

So why does .5115 round to .511? Because, like many numbers with decimal fractions, this number cannot be represented exactly as a binary fraction (since many base 10 numbers cannot be converted exactly to base 2 numbers). When R&R Report Writer uses Microsoft C floating point routines to store a binary value for .5115, this value, which is actually a very long

sequence of ones and zeros, is .511499...[1], which is slightly less than .5115. As a result, when R&R Report Writer rounds this internal value to three decimal places, it looks at the fourth digit (4) and returns .511 instead of .512. You will see equivalent rounding behavior in applications that use the same floating point package as R&R Report Writer. In FoxBASE+® and Clipper 5, for example, ROUND(.5115,3) also returns 0.511.

Rounding Display Formats

To round the display format of any numeric value (regardless of whether you have used the ROUND() function), you use the /Field Format command to select any format except "General" and specify a number of decimal places from 0 to 15. The /Field Format command does not actually change the value of the field. Instead, it changes the way R&R will display and print the field value. For example, you can round the display format of the NUMBER field to two decimal places by applying a "Fixed" field format and specifying 2 as the number of decimal places. The result of this formatting is that values with more than two decimal places will display and print in rounded form, *although the actual values will not be rounded*. When the value of the formatted NUMBER field is 5.778, for example, the field will display and print as 5.78, while its value for the purposes of any calculation or query remains 5.778.

When Should You Use Round()?

So which method of rounding should you use? For reports in which numeric accuracy is important, you should not rely on field format to round numbers, since formatting doesn't round values. A field's format affects only the way in which the field's data displays. If you care about the actual value of a field, as you would when the field is used by other calculations, you should use the ROUND() function to return a rounded value that provides the degree of precision you require.

Take the example of an invoice report that calculates an order total by: 0

1. Multiplying a list price times a discount percent.
2. Multiplying the resulting discount price times a sales tax percent.
3. Adding these two calculated values.

As the example in Figure 1 illustrates, these calculations can result in non-integer values that must be rounded in order to express them in dollars and cents. How you round these values (given to six decimal places in the example) is crucial to the appearance of your report.

---

[1] The stored binary value actually corresponds to a fifty-three digit number whose first twenty digits are .51149999999999999547.

	FIELD/EXPRESSION	VALUE
List Price	LIST	176.990000
Discount Percent	DISCNT	.430000
Discounted Price	DISPRICE = LIST-(LIST*DISCNT)	100.884300
Tax Percent	TAXRATE	.050000
Tax Amount	TAX = DISPRICE * TAXRATE	5.044215
Total	TOTAL = DISPRICE + TAX	105.928515

Figure 1. Invoice Fields.

Unless you want to print invoices with values in fractional cents, you must round these values in your report. If you round these values by using /Field Format to format fields with a two-decimal currency format, you will get display formats like those illustrated in Figure 2.

Discounted Price	$100.88
Tax Amount	$5.04

TOTAL	$105.93

Figure 2. Invoice Total with /Field Format Rounding.

As you can see, the rounding effect of the /Field Format command makes the total appear inaccurate. The fields containing the discounted price and the tax amount round down to $100.88 and $5.04 respectively. However, the total field, which is the sum of the fields' actual rather than displayed values, rounds up to $105.93.

How can you avoid an irate call from a customer who claims you or your software can't add? Use the ROUND() function to round the discounted price and tax amount values that contribute to the total before you calculate the total. Figure 3 illustrates the revised expressions for these calculated fields and Figure 4 shows the resulting invoice values.

	FIELD/EXPRESSION	VALUE
List Price	LIST	176.99
Discount	DISCNT	.43
Disc. Price	**DISPRICE=ROUND(LIST-(LIST*DISCNT),2)**	100.88
Tax Percentage	TAXRATE	.05
Tax Amount	**TAX=ROUND(DISPRICE*TAXRATE,2)**	5.04
Total	TOTAL = DISPRICE + TAX	105.92

Figure 3. Invoice Fields with ROUND().

Discounted Price	$100.88
Tax Amount	$5.04

TOTAL	$105.92

Figure 4. Invoice Total using ROUND().

Although the discounted price and tax amount printed in Figure 4 are the same as those printed in Figure 2, the *values* in these fields are different. Due to the use of the ROUND() function, the discounted price and tax amount printed in Figure 4 are exactly the same as the values of the fields. When R&R Report Writer adds these two rounded values together, it gets $105.92 instead of $105.93 (which is the sum of the unrounded values).