R&R Report Writer *Technical Bulletin*

# Focus on the IIF Function

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

## Summary

This technical bulletin discusses the immediate if (IIF) function. The IIF function plays an important role in many report applications. First we will review the IIF syntax, and then construct some useful expressions with IIF.

## Syntax

The proper syntax for IIF is: IIF(<expL>,<exp1>,<exp2>). The way to read this is "if the logical expression is true, use the first expression, otherwise use the second expression." For example, IIF(1=1,"Equal","Not Equal") will always return the word Equal.

## Constructing an IIF Expression

There are two things to note when constructing an IIF expression. First, unlike dBASE®, you may test whether a character field contains something by including just the field name. In dBASE you would have to include a complete logical expression.

For example, in dBASE you must use IIF(STRING<>"",... to test whether the field STRING contains anything. In R&R Report Writer you may simply use IIF(STRING,.... The difference is minor, but you will find R&R Report Writer more convenient. Of course, the expression IIF(STRING<>"",... can be used in R&R Report Writer.

The second thing to note about IIF is that the two return expressions, <exp1> and <exp2>, must be the same data type. In the previous example they were both *character* data type. The same is true in dBASE.

While this may appear to be restrictive, it really isn't because R&R Report Writer includes functions that convert values from any data type to any other data type. An example will help illustrate this point.



© 1999 Liveware Publishing Inc.

Suppose a database contains a logical field, PAID, which indicates whether a balance has been paid. If PAID is true, you want to print the amount paid, which is stored in a *numeric* field named AMOUNT. If PAID is false, you want to print the *character* string "Not Paid".

The following expression produces this result by using the STR function to convert a numeric amount to a character string.

```
IIF(PAID, STR(AMOUNT), "Not Paid")
```

Using STR is the easiest way to convert a numeric value to a character string. Alternatively, you could use the TRANSFORM function with the advantage of having many of the same numeric format options that are available with the /Field Format command.

# Conditional Field Placement

Many accounting applications store a negative number in a database field to indicate a credit and a positive number in the same field to indicate a debit (or vice versa). Reports must then print credits and debits in different columns as in this example.

**Transaction Journal**

| Transaction | Date | Credit | Debit |
|---|---|---|---|
| Cash receipt | 08/30/86 | 320,000.00 | |
| Employee bonuses | 12/25/86 | | 11,000.00 |

To print a report that lists credits and debits in separate columns you need to create two calculated fields, one named Credit and the other named Debit. Use the /Field Calculate **C**reate command to define them as follows.

```
Credit = IIF(AMOUNT<0, AMOUNT, 0)
Debit = IIF(AMOUNT>0, AMOUNT, 0)
```

You can then place these fields in the appropriate position on the report format. The AMOUNT field itself will not appear anywhere on the report.

You will also want to format these fields to print blank instead of zero. To do this, use the /Field Format command. After you select a format, another menu will appear requesting you to select either Show-Zero or Blank-Zero. Select Blank-Zero.

The resulting report format should look similar to the following screen.

```
Field: Credit()                          Line:  1 Col:  41       READY
Type / for command menu.  Press F1 for help, F10 to insert field
IIF(AMOUNT<0, AMOUNT, 0)

Header                    Transaction·Journal
Header
Header  Transaction          Date        Credit        Debit
Header
Body    XXXXXXXXXXXXXXXXXX   mm/dd/yy    #,###,###.##  #,###,###.##
```

# Creating Group Fields

A wine and spirits distributing company has products organized in categories such as Vodka, Gin, Rum, Whiskey, Wine, Sherry, Champagne, Soda, and Tonic. The Data Processing department receives a request for a report that groups products by category within three product groups: Spirits, Wine, and Other. How can this be done?

There are two problems that need to be addressed. First, the database does not contain information about which categories belong to which product groups. The second problem, which doesn't arise until you solve the first problem, is the words *Spirits, Wine,* and *Other* do not sort alphabetically in this order (*O* comes first.)

One way to solve both problems is to restructure the database, adding a field for product group code. This code would be keyed to a new product group description file so that 1=Spirits, 2=Wine, and 3=Other.

The disadvantage to this approach is the large amount of programming required to modify existing programs and to create a new product group file maintenance program. The company decides this approach is unacceptable.

# Creating Fields within R&R Report Writer

Another way to produce the same report without modifying the database is to create the fields for product group code and product group description within R&R Report Writer. The advantage of using R&R Report Writer is that you can calculate the needed fields without modifying your database structure.

Let's assume the existing database includes a field named CATEGORY that contains one of the following names: Vodka, Gin, Rum, Whiskey, Wine, Sherry, Champagne, Soda, and Tonic. You will create two new fields, Prod_Group and Group_Name.

Prod_Group will contain the number 1, 2, or 3, and will be used to sort our new product groups in the desired order. Group_Name will contain the name Spirits, Wines or Other, and will appear in the product group header on the report.

Select the /Field Calculate Create command and enter the name Prod_Group. Then enter the following expression.

```
IIF(TRIM(CATEGORY)$"VodkaGinRumWhiskey",1,
 IIF(TRIM(CATEGORY)$"WineSherryChampagne",2,
  IIF(TRIM(CATEGORY)$"SodaTonic",3,
   4)))
```

This expression assigns a 1 to all of the spirits, a 2 to all wines, and a 3 to the other items. Note that we have included a fourth group. This provides an error checking mechanism in case a record in the database has a blank, misspelled, or unknown category.

This error checking technique is not required, but it's generally good to include. If you are confident that your data entry program will trap all of these errors, you can omit the error check using the following expression.

```
IIF(TRIM(CATEGORY)$"VodkaGinRumWhiskey",1,
 IIF(TRIM(CATEGORY)$"WineSherryChampagne",2,
  3))
```

Next, select the /Field Calculate Create command and enter Group_Name. Then enter the following expression.

```
IIF(Prod_Group=1,"Spirits", IIF(Prod_Group=2,"Wines",
      IIF(Prod_Group=3,"Other", "Error - category unknown")))
```

To omit the error check, use this expression instead.

```
IIF(Prod_Group=1,"Spirits",IIF(Prod_Group=2,"Wines", "Other"))
```

Note that this expression references the other calculated field, Prod_Group. R&R allows you to create calculated fields that depend on other calculated fields and automatically evaluates them in the correct order. This technique simplifies the expression for Group_Name.

Once these fields have been created, use the /Sort-Group Sort-Fields command to specify Prod_Group as the level one sort field and CATEGORY as the level two sort field.

Next, use the /Line Delete command to remove any body lines. Then use the Create Header 1 command to create a level one header line to print the product group. Use the Create Footer 2 command to create a level two footer line for the category name and any category totals.

The final step is to use the /Field Insert command to include the Group_Name field in the level one header and the CATEGORY field in the level two footer. The resulting report could look similar to the following example.

**Sales Report**

| Group/Category | Quantity |
|---|---|
| Spirits --Level One Header Line | |
| Gin Level | 1058 |
| Rum Two | 3827 |
| Vodka Footer | 2052 |
| Whiskey Lines | 5442 |
| Wines | |
| Champagne | 247 |
| Sherry | 43 |
| Wine | 1532 |
| Other | |
| Soda | 384 |
| Tonic | 281 |

All trademarks are the property of their respective owners. The information contained in this technical bulletin is subject to change without notice. Liveware Publishing Inc. provides this information “as is” without warranty of any kind, either expressed or implied, but not limited to the implied warranty of merchantability and fitness for a particular purpose. Liveware Publishing may improve or change the product at any time without further notice; this document does not represent a commitment on the part of Liveware Publishing. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of the licensing agreement.