R&R Report Writer *Technical Bulletin*

Using the PREVIOUS Function to Suppress Repeats

Product: ARPEGGIO™
R&R Report Writer® for Windows®
Version: All

Host: N.A.
NIC: N.A.
Interface: N.A.
Oper Sys: Microsoft® Windows® NT®

Summary

This technical bulletin presents an example of using the PREVIOUS() function to suppress repeating data in fields.

When to Use the PREVIOUS Function

The report excerpted in Figure 1 shows airline ticket information for a list of passengers. The report uses a master ticket table to look up records in a related passenger table and a related routings table.

Passenger	Ticket No	Fare	Routing
Allen, Joan	014-37898	216.00	YYZ - YUL
			YUL - YYZ
	014-37899	384.00	YYZ - YYC
			YYC - YYZ
Arseneault, H	014-37910	446.00	YYC - YOW
			YOW - YYC
Bannen, Colin	014-37955	201.50	YVR - YYC

Page 2

Passenger	Ticket No	Fare	Routing
Bannen, Colin	014-37955	201.50	YYC - YVR
Cohen, Jill	014-37999	216.00	YYZ - YUL
			YUL - YYZ

Figure I. Sample Report with Repeating Data

As you can see, the report is grouped by passenger name, ticket number, and fare. Marking the "Print Once" setting for all group fields will keep data from printing more than once for each group, unless the records for a group are split across the page, as they are for passenger

Bannen. In this case, R&R Report Writer reprints the data in all the group fields at the top of the second page.

While this repeated data usually clarifies a report by making it easier to follow groups from page to page, in some cases the repeated fare data makes it look as if passenger Bannen purchased two tickets. You have to look at the ticket number carefully to see that this is a single group that has two records, one for routing YVR-YYC and one for routing YYC-YVR.

How can you suppress the repeating fare data even when a group continues on another page? The solution is to use the PREVlOUS() and IIF() functions as described in the following paragraphs to suppress repeating data. The PREVIOUS() function returns the previous value placed in a field, while the IIF() function can be used to test whether this PREVIOUS value is the same as the field's current value. If the values are the same, the IIF function returns an empty string (effectively suppressing printing of a duplicate value). If the values are different, IIF returns the field's current value. Note that this technique can be used to suppress data even in fields that are not group fields.

How the PREVlOUS() Function Works

In order to use the PREVIOUS() function properly, you must understand exactly how the function works. In the Xbase version of R&R Report Writer, PREVIOUS() returns the value of the specified field when it was last evaluated by R&R Report Writer — that is, when R&R Report Writer last placed a value in the field by reading the table containing the field and/or performing an R&R Report Writer calculation.

R&R Report Writer evaluates every table, calculated, and simple total field for each composite record. That is, R&R Report Writer reads the table(s) and places values in these fields each time it builds a composite record. In this case, the PREVIOUS value of any of these fields will be the field's value in the previous composite record.

For example, take a report that uses a master orders table to look up a customer address in a related customer table. Each time R&R REPORT WRITER reads an orders record, it must also read a customer record to find an address for the order. This means that fields from both files are evaluated each time a composite record is built. As a result, the PREVIOUS value of any field in the report will be the value of the field in the previous composite record.

NOTE: Fields that total other fields are evaluated only when the field being totaled resets. These fields are therefore an exception to the general rule stated here. Their PREVIOUS value will not always be the value as of the previous composite record.

Using PREVIOUS() to Suppress Fare Data

Given the way the PREVIOUS function works, the first step in using this function to suppress repeating fare data is to determine how often the FARE field is evaluated. The FARE value does not change for each record, but for each ticket. As a result, the value of PREVIOUS(FARE) will be the fare for the previous ticket, rather than the previous composite record.

In order to suppress any repetition of the fare value within a single ticket group, we must create a calculated field that says, in effect, if the ticket number for this composite record is the same as the ticket number for the previous composite record, don't print the fare value for the ticket (since it will have already been printed). But if the ticket number for this composite record is different from the ticket number for the previous composite record, print the fare value.

Before we can create this calculated field, we must first create an intermediate calculated field on which the PREVIOUS function can operate. This field will contain the same value as the ticket number field, but it will be evaluated for each composite record rather than for each group.

We can create a calculated ticket number field evaluated for every record by using TICKETNO as an argument to a function that causes R&R Report Writer to perform a calculation for each composite record. For example, the RECNO() function used without a file name argument causes R&R Report Writer to return the current composite record number. Any field using this function must therefore be evaluated for each composite record.

Let's create a calculated field called PERCOMP with the expression given below. This field will return the same value as that contained in the TICKETNO field, but the calculated field will be evaluated for each composite record rather than for each ticket group:

```
IIF( RECNO(), TICKETNO, TICKETNO)
```

Translated, this expression says that if the current composite record number is not equal to zero, return the value of TICKETNO. Otherwise, return the value of TICKETNO. The trick to the expression is that it will simply return the value of the TICKETNO field for each composite record, no matter what the value of RECNO.

After creating the PERCOMP field, you can create a second calculated field with the following expression:

```
IIF (PERCOMP <>PREVIOUS (PERCOMP), STR (FARE),"")
```

Using the PREV and IIF functions, this field tests to see if the ticket number value (i.e. the PERCOMP value) in the current composite record is the same as that in the previous composite record. If so, the expression returns an empty string. If not, the expression returns the FARE value from the current composite record. Note that the STR function has been used in this

expression because the two result arguments of an IIF expression must have the same data type. For date fields, you could use the DTOC function in the same way; that is,

```
IIF (PERCOMP <> PREVIOUS (PERCOMP), DTOC (<date field>),"")
```

When this calculated field replaces the FARE field on the report layout, the fare value will print once (and once only) for each ticket group, even without making the calculated field a group field.

You could use the same strategy to suppress the repeated printing of the passenger name and/or ticket number.
