Reality

V14.0

MultiValue Compatibility

Document No. UM70007043A
March 2008

Peoplebuilding 2
Peoplebuilding Estate
Maylands Avenue
Hemel Hempstead
Herts
HP2 4NW

Tel: +44 (0)1442 232424
Fax: +44 (0)1442 256454

www.northgate-is.com

Contents

Introduction 4
Main Features 4
Reality Documentation 4
Feedback 4
Conventions 4
Reality Features that help you Migrate 6
Pseudo-floppy 6
MIGRATE.ACCOUNT 6
Operating Environments 6
DataBasic Compiler Options 6
MultiValue Compatible Features 7
Environment Options 8
Predefined Environments 11
DataBasic Features that Depend on the Compatibility Mode 12
Summary 14
Options for MultiValue Compatibility 16
General 16
English 16
Proc 16
DataBasic 17
Reality Features with Additional Functionality 18
General 18
English 18
DataBasic 18

Introduction

Reality was the first commercial release of the MultiValue PICK Database in the early 1970s. Initially, other versions that came onto the market were comparable in features and functions, but over time changes have occurred and a migration process is now required to transfer to Reality from the other versions.

A significant area of on-going Reality development concerns improving compatibility with other MultiValue platforms and the migration process. This document describes features that can help you migrate to Reality from other MultiValue platforms. For details of the migration process, see the document MultiValue Migration.

The latest software, patches, migration information and on-line documentation are available at *www.northgate-is.com/reality* – in the heading click the [Products] tab.

Main Features

All MultiValue systems share main features that are very similar, but these are sometimes given different names. The following are the names used in Reality:

- Command processor – TCL (Terminal Control Language).
- Query language – English.
- Programming language – DataBasic.
- Batch processing language – Proc.

Reality Documentation

Always refer to the on-line documentation for complete details of features and implementation of Reality Systems. This is available on the Reality Installation CD and as a download. When run refer to the link "*How to use the On-line Documentation"*, which is on the front page.

You will find Migration information for the current release in the on-line documentation under ⓘ<MultiValue compatibility>.

General FAQs can be found on the Reality Website, as given above. Click the [Support] tab and then refer under FAQs.

Feedback

This is an evolving document and Northgate would appreciate any feedback, either directly to a representative or via an email to reality@northgate-is.com. Please indicate clearly that this is "MultiValue Compatibility Feedback".

Conventions

The following conventions are used in this documentation:

`Text`	Text shown in this typeface is used to show text output on the screen.

Text — Characters or words in this italic font indicate parameters which must be supplied by the user. For example in

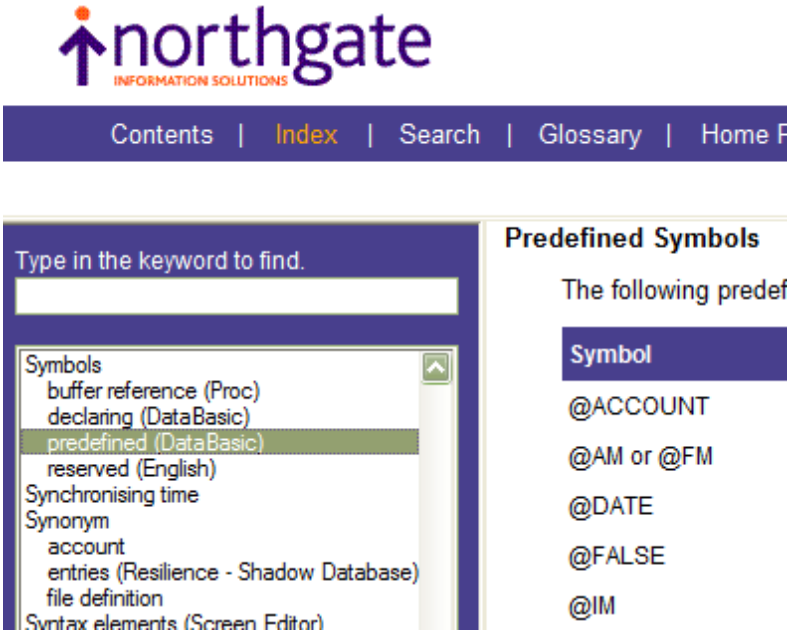
LIST *file-specifier*

the parameter *file-specifier* is italicised to indicate that you must specify an actual file defined on your system.

Document Title — Italic text in this font indicates topic titles and other referenced documents. References shown in *blue* or *purple* are hyperlinks – if clicked, they will take you to the referenced section (purple links are those that you have already visited).

SMALL CAPITALS — Small capitals are used for the names of keys such as RETURN.

ⓘ<> — This symbol preceding text enclosed in chevrons is used for references to the index in the Reality on-line user documentation (select the Index link in the documentation banner). Enter the text between the chevrons into the field at the top of the Index tool and then double click or press RETURN to display the topic.



Reality Features that help you Migrate

This section summarises those features of Reality that have been introduced to aid migration from other MultiValue systems.

Pseudo-floppy

The format used for Reality tape images is different to the pseudo-floppy (.vtf) format used by other MultiValue systems. Reality provides commands that allow you to transfer data between Reality and other MultiValue systems by converting Reality tape images into MultiValue pseudo-floppy images and vice versa.

See ⓘ<FLOPPYTOTAPE command>, ⓘ<TAPETOFLOPPY command>.

MIGRATE.ACCOUNT

ⓘ<MIGRATE.ACCOUNT> converts accounts saved from other MultiValue systems to run on Reality. This must be run after an account to be converted has been loaded (using ⓘ<ACCOUNT-RESTORE> with the appropriate options), but before logging on to the account for the first time.

Operating Environments

An ⓘ<Operating Environment> is a combination of settings that changes the way in which Reality functions. One use of these is to simplify migration from a different type of MultiValue system. Several predefined Environments are provided, which you can use either directly or as templates for creating your own Environments.

You create environments using the ⓘ<SSM> command. They can then be associated with user profiles so that each user is given a suitable profile at log on or applied at TCL when needed with the ⓘ<SET-ENVIRONMENT command>. You can also set and clear the individual environment options with the ⓘ<SET-OPTION> and ⓘ<CLEAR-OPTION> commands respectively.

DataBasic Compiler Options

You will need to recompile your applications in order to run them on Reality. To simplify migration, Reality provides a simple way of changing the way in which your applications are compiled.

AutoInclude

Statements that you need in every code module can be placed in an auto-include item. When your programs are compiled, the contents of this item will be automatically included at the beginning of your code.

The auto-include item can be placed in your source file, to include its contents at the beginning of every code module in the same data section, or in the default data section of the file BASIC-COMPILERS (SYSFILES account); in this case, its contents will be included in every DataBasic program in your database after any local automatically included code. For details, see ⓘ<Automatically including common code>.

$OPTIONS

MultiValue Modes

The $OPTIONS statement changes the functionality of the compiler and of certain DataBasic program elements for compatibility with the MultiValue environment for which your program was originally written. You place a $OPTIONS statement specifying the MultiValue mode to use at the start of each source module– currently mvEnterprise, mvBase and D3 modes are available. The differences between the different modes are detailed in the section *DataBasic Features that Depend on the Compatibility Mode*.

DataBasic Compiler

By default, the Reality compiler generates a platform-independent deliverable object in the data section of the source file. The CATALOG command must be used to generate an executable item, which is placed in the global pointer file.

The $OPTIONS setting **EXEC.OBJ** allows you to change this so that the compiler works in the same way as other MultiValue systems. An executable item is generated in the dictionary of the source file and CATALOG simply creates a command entry in master dictionary of the account. Refer to ⓘ<$OPTIONS statement> for details.

MultiValue Compatible Features

In addition to those listed above, Reality now includes additional functionality that is compatible with other MultiValue systems. For details, refer to *Reality Features with Additional Functionality* and *Options for MultiValue Compatibility*.

Environment Options

The table below lists the options that can be set when defining an ⓘ<operating environment>.

Note: Those marked with an asterisk (*) cannot be set with ⓘ<SET-OPTION> or cleared with ⓘ<CLEAR-OPTION>.

$<.FORMAT	Changes the position of the currency symbol when this is combined with the **E** credit indicator in ML and MR conversion codes (see ⓘ<Conversion codes,ML/MR>. The default is to place the currency symbol inside the chevrons that enclose negative values (that is, after the "<" chevron). When this option is set, the currency symbol will be placed before the "<" chevron. Also applies to DataBasic ⓘ<format strings>.
2DIGIT.DATE	Changes the format of the output of the English DD, DM and DAM conversion codes to include a leading zero, if appropriate, in the day and month. See ⓘ<D,conversion code>.
ALT.MT	Allows decimal points to be used as separators in times passed to the ⓘ<MT> input conversion code.
ALTFCORR	In the English ⓘ<F conversion code>, reverses the order of the operands for Div, Rem, Sub, and Cat, so that these operations work as with FS code, while leaving the [,], < and > to operate as with F code.
ALTHEADING	Subtly changes the relationship of heading and page number, particularly in DataBasic. The most noticeable change is that a null heading maintains pagination, but outputs no line, and that after a heading is initially defined, re-execution of the HEADING statement in DataBasic will not cause a heading to be created until the end of the current page. Also, the page number is right-adjusted in a field of four blanks, unless the PN option is used, in which case it will be left-adjusted. SYSTEM(4), SYSTEM(5) and SYSTEM(6) continue to function as though a heading is in effect with this null heading.
BSELECT.NULL	Causes null attributes to be included in the list produced by the ⓘ<BSELECT> and ⓘ<BSSELECT> commands. This makes it possible to iterate through such a list with, for instance, the DataBasic READNEXT statement without having to specifically process the null items.
CATALOG.COMP	Causes the ⓘ<CATALOG> command to check that any MD entry that will be overwritten is a DataBasic program definition item, but not whether it references the file and item being cataloged. If this option is not set, CATALOG will not proceed if the MD contains any entry with the same name as the program being cataloged, unless it is a DataBasic program definition item that specifically references the program being cataloged; if set, it allows any

DataBasic program definition item to be overwritten. No other types of MD entry will be overwritten.

DB.DEBUG — Causes any DataBasic program initiated by the user to enter the DataBasic symbolic debugger on executing DEBUG statements within the program. This is similar to starting the program with the ⓘ<DEBUG command>, but can be used to debug programs called from PERFORM statements and from Procs.

Programs initiated by the user are those that are started directly or indirectly from TCL. They include those initiated by a Proc or other program that was itself started directly or indirectly from TCL.

DEL.FILE.EXEC — Causes the ⓘ<DELETE-CATALOG> and ⓘ<DECATALOG> commands to delete executable items from the local dictionary or data section by default. If not set, executable items from the local dictionary or data section are not deleted by default.

DOT/Stacker * — Sets the mode of operation for the TCL Stacker Recall Command (see ⓘ<Recalling TCL commands>). The following modes are available:

1 REALITY – Do not emulate another MultiValue system (default).
2 D3
3 MVBASE
4 MVENTERPRISE

EB.DEBUG — Causes any DataBasic program run from External Basic to enter the DataBasic symbolic debugger on executing DEBUG statements within the program. Programs run from External Basic include ⓘ<file triggersⓘ> and ⓘ<RealWeb> subroutines.

EMBEDDED.OPTIONS — Allows command options to be entered in the middle of a TCL command statement.

ENGDIVIDE — Causes division operations in English A and F conversion codes to return 0 when dividing by 0 (normally returns dividend).

EXEC.BASIC.OBJ — Changes the types, names and locations of the items generated by the DataBasic compiler. If set, an executable (platform-specific) item with the same name as the source item is generated in the dictionary of the file. No deliverable (platform-independent) item is generated unless the DEL.OBJ option is also selected (in a ⓘ<$OPTIONS> statement), or the BASIC command's **(R** option is used.

FATAL.WARNINGS — Causes all warning messages generated by DataBasic programs to be treated as fatal errors. Breaks to the DataBasic debugger to allow determination of error and possible recovery. Similar to starting the program with the F option.

INHIB.MLMR	Causes a null value generated by the English conversion processor to be passed to an ML or MR conversion as a null string. If not set, a null value is passed as the number 0.
INHIBIT!SYS	Disables the ⓘ<! command>, so that TCL commands can be given names beginning with !. The SYS command is unaffected.
KEEPLIST	Specifies that an active list is to be kept active when a non-existent TCL command is entered.
LITERAL.MASK	If set, causes a ⓘ<format mask> containing no fill characters to return the mask text. If not set, the fill data is returned.
MCT.SQUOTE	The ⓘ<MCT> conversion code normally treats the first alphanumeric character following a space, double quotation mark, left parenthesis or hyphen as the start of a word. If you set this environment option, a word starts with the first alphanumeric character following any non-alphanumeric character other than a single quote.
MFILL.FORMAT	In a format mask, normally only the first fill character (#, * or %) encountered is used; any subsequent occurrences of either of the other fill characters are treated as literals. If set, this option causes all of these characters to be treated as fill characters wherever they occur in the mask.
MVPROCSELECT	Specifies that active select lists in Proc should be handled as on other MultiValue systems. For details, see *Proc, Select Lists* in the section *Options for MultiValue Compatibility*.
OLD.CREATEFILE	Makes the modulo parameters mandatory in the ⓘ<CREATE-FILE> command. With this option selected, you cannot create an automatically sized data section by simply omitting the modulo, but must specify the **A** option.
PROCX	Causes X in Proc to return from a subroutine, rather than exit.
RPLDIVIDE	If set, causes the original value to be returned by RPL if the divisor is zero; if not set, zero is returned.
RPLTERM	If set, enables terminal independence in RPL (RPQ feature R205).
SPASSIGN	Causes ⓘ<SP-ASSIGN> to close open print jobs only when no parameters are specified.
TCLDELIMITER	Allows '\' to be equivalent to ' " ' at TCL.
UCASEDATES	Causes the names or abbreviations of names of months produced by output conversion to be all uppercase, rather than just initial capitals.
UCASEDAYS	Causes the names of days produced by output conversion to be all uppercase.
UCASEMSGS	Causes system messages to be displayed in all uppercase letters, rather than just initial uppercase.

Predefined Environments

The table below lists the predefined MultiValue Environments that are currently available and the settings defined for each.

Option	D3	mvBase	mvEnterprise
$<.FORMAT	N	N	N
2DIGIT.DATE	N	N	Y
ALT.MT	N	N	Y
ALTFCORR	N	N	Y
ALTHEADING	N	N	N
BSELECT.NULL	N	N	Y
CATALOG.COMP	N	N	Y
DB.DEBUG	N	N	N
DEL.FILE.EXEC	N	N	Y
DOT/Stacker	D3	MVBASE	MVENTERPRISE
EB.DEBUG	N	N	N
EMBEDDED.OPTIONS	N	N	N
ENGDIVIDE	N	N	N
EXEC.BASIC.OBJ	N	N	N
FATAL.WARNINGS	N	N	N
INHIB.MLMR	N	N	Y
INHIBIT!SYS	N	N	N
KEEPLIST	N	N	N
LITERAL.MASK	N	N	N
MCT.SQUOTE	N	N	Y
MFILL.FORMAT	N	N	N
MVPROCSELECT	N	N	N
OLD.CREATEFILE	N	N	N
PROCX	N	N	N
RPLDIVIDE	N	N	N
RPLTERM	N	N	N
SPASSIGN	N	N	N
TCLDELIMITER	N	N	N
UCASEDATES	N	N	Y
UCASEDAYS	N	N	Y
UCASEMSGS	N	N	N

DataBasic Features that Depend on the Compatibility Mode

This section lists the DataBasic features that are controlled by the ⓘ<MultiValue Compatibility,$OPTIONS statement> compatibility mode – see *MultiValue Modes*.

Predefined Symbols

In mvEnterprise mode, @DAY, @MONTH and @YEAR symbols are available in addition to the standard Reality ones (see ⓘ<Symbols,predefined>). These return the current day of the week, and the current month and year.

Pattern Matching

In mvEnterprise mode, multiple patterns used with the ⓘ<MATCH> operator can be separated by any of the Reality system delimiters – attribute mark (@AM), value mark (@VM) or subvalue mark (@SM or @SVM).

In any other mode, multiple patterns must be separated by value marks.

Format Strings

- In mvEnterprise mode, if a scaling factor is specified when formatting numeric data, the current precision is subtracted from this before being applied and the **M** modifier has no effect. If the result is negative, a scaling factor of 0 is used.
- In other MultiValue modes, if a scaling factor is specified, the current precision is subtracted from this before being applied. The **M** modifier has no effect.
- Otherwise, if a scaling factor is specified, both the precision and scaling are applied without modification. If the **M** modifier is used and no scaling factor is included, the *precision* parameter is also used as the scaling factor (scaling is performed in the same way as in ⓘ<English>).

Strings as Logical Expressions

In mvEnterprise mode, if a non-numeric string is used as a logical (Boolean) expression, it evaluates to true. In any other mode, the result is false and a run-time error is generated.

@ Function

Support for ⓘ<@ function> extended cursor addressing codes depends on the selected compatibility mode. Full support is only available in Reality mode.

CLEARINPUT Statement

In mvEnterprise mode, the CLEARINPUT statement can be used to clear the typeahead buffer (the same as ⓘ<INPUTCLEAR>).

DCOUNT Function

In mvEnterprise mode, if the ⓘ<DCOUNT> function is passed a string as a delimiter, the whole string is used as a delimiter. In any other mode, just the first character is used.

DELETESEQ Statement

In mvEnterprise mode, the specified host file is deleted – the file need not be open. In any other mode, the file must have first been opened with the ⓘ<OPENSEQ> statement.

FORMLIST Statement

Creates a select list from a dynamic array. The resulting list can be read sequentially with the ⓘ<READNEXT> statement and can be used in the PASSLIST clause of ⓘ<PERFORM> or EXECUTE. Only available in mvEnterprise mode.

IN and INPUT Statements

In all MultiValue modes, if the *time* parameter is set to 0, the statement returns after one tenth of a second. In Reality mode, if *time* is 0 or negative, the result is no time-out.

In addition, in mvEnterprise mode, the IN statement echoes the characters entered to the screen. In any other mode, no characters are echoed.

LOCATE Statement

In mvEnterprise mode, setting the ⓘ<LOCATE> statement's *sequence* parameter to "AR" or "DR" specifies:

- That the elements in the dynamic array have been sorted in right-aligned ASCII order (ascending or descending as appropriate), rather than in numeric order.
- That value and subvalue marks will be treated as printable rather than control characters.

MOD Function

The MOD function divides one number by another. In mvEnterprise mode, the remainder is returned, while in any other mode, the modulus is returned.

ON GOSUB and ON GOTO Statements

In all MultiValue modes, if *expression* evaluates to a value less than 1 or greater than the number of statement labels, control is transferred to the statement following the ⓘ<ON GOSUB> or ⓘ<ON GOTO> statement.

In Reality mode:

- If *expression* evaluates to less than one, a message is displayed and control is transferred to the first statement label.
- If *expression* evaluates to a value greater than the number of statement labels in the list, a message is displayed and control is transferred to the last statement label.

OPENSEQ Statement

In mvEnterprise mode, Reality items cannot be opened for sequential access. The path and filename of the host file must be separated by a comma.

In all other modes, both host files and Reality items can be opened. When opening a host file, the filename must be part of the path.

READLIST Statement

In mvEnterprise mode, READLIST creates a dynamic array from a select-list.

Otherwise, a list is read from ⓘ<POINTER-FILE> and assigned to a variable.

REMOVE Statement

In mvEnterprise mode, if the ⓘ<REMOVE statement> is called with the *setting-var* variable set to -1, the Remove Pointer is reset to the beginning of the array. In any other mode, any previous value in *setting-var* is ignored.

SELECT Statement

In all MultiValue modes, if an active select list is present the SELECT statement ignores any file variable or dynamic array specified and makes the active select list available. If you specify an index, however, the list is created from that index.

In Reality mode, you must use the SELECTE statement to make the active select list available.

Default SELECT List

In all MultiValue modes, the default SELECT list is global. This means that it is possible to use a ⓘ<PERFORM> or *EXECUTE* statement to generate a new default SELECT list. The ⓘ<READNEXT> statement has access to this global list; when the current list is exhausted, a PERFORM or EXECUTE statement can make a new list available.

In Reality mode, the default SELECT list is local to the current context; a PERFORM statement cannot make a new list available.

SUM Function

In mvEnterprise mode, the ⓘ<SUM> function returns 0 if passed an empty dynamic array. In any other mode it returns an empty string.

SYSTEM Function

The elements available in the ⓘ<SYSTEM function> depends on the MultiValue mode. For unsupported elements, a runtime error occurs and the function returns 0.

Note: The *ASSIGN* and *SYSTEM* statements can only change the values of the settable elements that are available in the selected MultiValue mode.

User Exits

In all MultiValue modes, if a user exit is called with the ICONV or OCONV function, a runtime error occurs and 0 is returned.

Summary

The following table lists the DataBasic options that are set in the different $OPTIONS modes. If nothing is shown for a particular option, the feature operates in the same way as in Reality mode; in some cases, this means that the feature is not available.

Option	D3	mvBase	mvEnterprise
Predefined symbols			✓
Pattern matching			✓
Format strings	✓	✓	✓
Strings as logical expressions			✓
@ function	✓	✓	✓
CLEARINPUT statement			✓

Option	D3	mvBase	mvEnterprise
DCOUNT function			✓
DELETESEQ statement			✓
FORMLIST statement			✓
IN and INPUT statements	✓	✓	✓
LOCATE statement			✓
MOD function			✓
ON GOSUB and ON GOTO statements	✓	✓	✓
OPENSEQ statement			✓
READLIST statement			✓
REMOVE statement			✓
SELECT statement	✓	✓	✓
Default SELECT list	✓	✓	✓
SUM function			✓
SYSTEM function	✓	✓	✓
User exits	✓	✓	✓

Options for MultiValue Compatibility

General

SP-ASSIGN

By default, the Reality ⓘ<SP-ASSIGN> command will close any open print jobs. This behaviour can be changed by setting the **SPASSIGN** environment option, so that open print jobs will only be closed if SP-ASSIGN is called with no parameters.

English

Division using A and F Conversion Codes

The division (/) operations provided by the A and F conversion codes normally return the dividend if division by zero is attempted. If required, this can be be changed to return zero by setting the **ENGDIVIDE** option in your operating environment.

Date Format

The DD, DM and DAM conversion codes normally return the day or month (as appropriate) without any leading zero. If required, this can be changed to include a leading zero if appropriate, by setting the **2DIGIT.DATE** option in your operating environment.

Changing Case

The MCT conversion code normally treats the first alphanumeric character following a space, double quotation mark, left parenthesis or hyphen as the start of a word. If you set the **MCT.SQUOTE** option in your operating environment, however, a word starts with the first alphanumeric character following any non-alphanumeric character other than a single quote.

Proc

Select Lists

The **MVPROCSELECT** environment option changes the way in which active select lists are handled in Proc:

- Without MVPROCSELECT set, the SELECT command is placed in the primary output buffer, and the command that uses the resulting list must be placed in the secondary output buffer. The P command will then execute both commands. For example:

```
HSELECT MD = "A]"
STON
HCOUNT MD<
P
```

- With MVPROCSELECT set, the SELECT command can be executed with the P command without first placing the second command in the secondary output buffer. The select list then remains active until used by a second command placed in the primary output buffer. For example:

```
HSELECT MD = "A]"
P
HCOUNT MD
P
```

The second form is compatible with other MultiValue systems.

Item Locks

A Reality database can be configured to handle item locks in a similar way to some other MultiValue systems (see *DataBasic – Item locks* for more details).

DataBasic

The following statements and functions have been provided for compatibility with other MultiValue systems. You do not have to specify a MultiValue mode.

ASSIGNED Function

Returns 1 if the specified variable is currently assigned a value; otherwise, returns 0 (zero). This is the opposite of the UNASSIGNED function.

CREATE statement

Creates a host file or Reality item that has been opened for sequential access using OPENSEQ with the **EXISTING** keyword.

EXECUTE statement

Executes a TCL command from DataBasic. Data (including lists) can be passed between the command and the program. Similar to PERFORM.

STATUS function

Returns an error code which is equivalent to that which would have been returned via the **SETTING** clause in the previous file operation. Provided for compatibility with MultiValue systems that do not include a **SETTING** clause in their file access statements.

SWAP function

Replaces a substring with a new substring.

Reality Features with Additional Functionality

General

ACCOUNT-RESTORE

Additional ⓘ<ACCOUNT-RESTORE> options allow you to restore a jBASE file save tape onto Reality and restore a tape image that has been converted from an mvEnterprise Save tape using ⓘ<FLOPPYTOTAPE>. See ⓘ<MultiValue compatibility,ACCOUNT-RESTORE> for details.

VERIFY-SAVE

An additional option allows you to check that a tape image that has been converted from an mvEnterprise Save tape using FLOPPYTOTAPE can be restored onto Reality. See ⓘ<MultiValue compatibility,VERIFY-SAVE> for details.

Indexes

An index can now be created directly from a dictionary definition item – see ⓘ<CREATE-INDEX command> for details.

English

B Conversion Code

This is provided for compatibility with the Raining Data mvBase and mvEnterprise systems. It provides the same functionality as the Reality ⓘ<CALL conversion code>; that is, it allows dictionary subroutines written for these systems to be called from File and Data Definition items on Reality. See ⓘ<MultiValue compatibility,B conversion code> for details.

DataBasic

Statement Labels

Reality accepts the following:

- A label can have one or more spaces between the label identifier and the terminating colon (if any – can be omitted for numeric labels).
- A label's terminating colon can be followed by a semicolon. This allows a comment to be included on the same line as the label.
- A label can have the same name as a literal, symbol or variable.

Substring Extraction

Reality will accept an alternative syntax for substring extraction that is used on many other MultiValue systems. If the *length* parameter is omitted, a positive *start#* parameter is treated as shorthand for a negative *start#* parameter, combined with a *length* parameter with the same value. For example, the expression VAR[3] produces the same result as VAR[-3,3]; that is, it extracts the last three characters of VAR.

Item Locks

DataBasic and Proc allow you to take multiple locks on the same item (this will normally occur if the item is processed by a number different subroutines, each of which locks the item). On Reality, each of these locks must be separately released, while some other MultiValue systems (for example, Power 95) require only a single release operation.

If required, a Reality database can be configured to work in a similar way to these MultiValue systems (see ⓘ<MultiValue compatibility,item locks>).

EQUATE Statement

You can use the EQUATE statement to assign an identifier to represent a single statement or intrinsic function, the result of calling the @ function and predefined symbols. For details, see ⓘ<EQUATE statement> and ⓘ<MultiValue compatibility,EQUATE statement>.

FIELD Function

Reality accepts an optional fourth parameter to the ⓘ<FIELD function>. If supplied, this parameter must contain the number of fields to return (default 1).

GROUP Function

When calling the ⓘ<GROUP function>, the fourth parameter is optional. If this parameter is omitted, GROUP returns just a single field.

MATPARSE Statement

Reality will accept an alternative syntax for the ⓘ<MATPARSE statement>, where a comma introduces the USING clause instead of the USING keyword. This is used on many other MultiValue systems.

READNEXT/READPREV Statements

The READNEXT and READPREV statements will accept an alternative syntax that is used on other MultiValue systems.

DECATALOG Command

DECATALOG is provided as a synonym for the Reality DELETE-CATALOG command.