

RealityX

Release 4.0

Differences Between Releases 3.1 and 4.0

All trademarks including but not limited to brand names, logos and product names referred to in this document are trademarks or registered trademarks of Northgate Information Solutions UK Limited (Northgate) or where appropriate a third party.

This document is protected by laws in England and other countries. Unauthorised use, transmission, reproduction, distribution or storage in any form or by any means in whole or in part is prohibited unless expressly authorised in writing by Northgate. In the event of any such violations or attempted violations of this notice, Northgate reserves all rights it has in contract and in law, including without limitation, the right to terminate the contract without notice.

© Copyright Northgate Information Solutions UK Limited, 1996.

Document No. UM70006362A
January 1996

Northgate Information Solutions UK Limited
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

Chapter 1 About this Manual

Purpose of this Manual	1-2
Contents	1-3
References	1-4
RealityX Reference Manuals	1-4
Manuals for 7.0/Pre-7.0 Upgrade	1-4
Conventions	1-5
User Comments	1-7

Chapter 2 Partition Database

Introduction	2-2
Comparative Benefits	2-2
Differences in Administrative Procedures	2-3
Making a Database	2-3
realfstab File	2-3
Saving and Restoring a Database	2-4
Database Error Checking	2-4
Monitoring Disk Space	2-4
realdbck utility	2-5

Chapter 3 Shadow Database

Chapter 4 Indexing

Introduction to Indexing	4-2
TCL Verbs for Managing Indexing	4-2
Indexing Functions	4-3
Defining an Index	4-3
Creating an Index	4-3
Verifying an Index	4-3
Deleting an Index	4-3
Changing an Index Definition	4-3
Indexes in the Database File Structure	4-4

Management and Maintenance of Indexes	4-5
Maintenance of Indexes	4-5
File Statistics for Indexes.....	4-5
Effect of Creating Multiple Indexes	4-5
Accessing an Index	4-6
Using SELECT-INDEX	4-6
Using DATA/BASIC	4-6
Implicit Use of Indexes by ENGLISH.....	4-7
Rules for Implicit Use of an Index.....	4-7
Examples of Implicit Use of Indexes	4-7

Chapter 5 Spooler

List of New Features.....	5-2
Design Changes	5-4
Compatibility	5-4
Structural Changes.....	5-5
New Commands	5-6
Changed Commands.....	5-6
New Options	5-7
Obsolete Commands.....	5-7
Job File Feature.....	5-8
Job File Determines References	5-8
Using ENGLISH on a Job file	5-8
File Utilities	5-8
Miscellaneous Differences	5-9
Security.....	5-9
Spooling to a Remote System	5-9
Network Printing with Printer Sharing	5-9
Printer Independence	5-9
Prologues and Epilogues.....	5-9

Chapter 6 TCL Commands

List of New Commands	6-2
List of Changed Commands.....	6-4
List of Obsolete Commands.....	6-6

Descriptions of New TCL Commands.....	6-7
%FULL	6-7
AND-ITEMS	6-7
AND-LISTS	6-7
BUILD.DESPOOLERS	6-7
CLEAR-OPTION	6-7
CONVERT.OBJECT	6-7
CREATE-INDEX	6-7
DEBUG.....	6-7
DECAT	6-7
DEFINE-INDEX.....	6-7
DELETE-INDEX.....	6-8
DESPOOLER.DETAIL	6-8
DISCTOTAPE	6-8
DSM	6-8
DSM	6-8
FQM	6-8
GO.....	6-8
LIST-SPREAD.....	6-8
LOAD-ALPHA.....	6-8
LOAD-BNF	6-8
MAN	6-8
NSELECT	6-8
OR-ITEMS.....	6-8
OR-LISTS.....	6-9
PCSM	6-9
PDM	6-9
POVF.....	6-9
RESET-ALPHA	6-9
RESET.DESPOOLER.....	6-9
SELECT-INDEX Verb	6-9
SET-ALPHA	6-9
SET-OPTION	6-9
SHOW-MODULI.....	6-9
SORT-SPREAD	6-9
SP-COPY	6-9

SP-DESCRIPTION	6-9
SP-DESPOOLERS	6-10
SP-DETAIL	6-10
SP-FIX	6-10
SP-FORMQUEUE	6-10
SPM	6-10
START-DESPOOLER	6-10
STOP-DESPOOLER	6-10
TAPETODISC	6-10
UPGRADE.BASIC.OBJECT	6-10
VERIFY-INDEX	6-10
XOR-ITEMS	6-10
XOR-LISTS	6-10
XSELECT	6-10
Description of Differences in Changed TCL Commands	6-11
. (Period) TCL Stacker	6-11
ASSIGN	6-11
BASIC	6-11
BLIST	6-11
BLOCK-PRINT	6-11
BLOCK-TERM	6-12
BVERIFY	6-12
CATALOG	6-13
CREATE-ACCOUNT	6-13
CREATE-FILE	6-14
DATA.PRODUCTS	6-14
DELETE-ACCOUNT	6-14
EVFU-SETUP	6-14
Changing FILE-SAVE PROC to Save Paper	6-14
FIND	6-15
LISTFILES	6-15
LOGTO	6-15
MOVE-FILE	6-16
M-A-S	6-16
MSG	6-16
NEW-SORT-LIST	6-17

PORT-DESPOOL.....	6-18
PORTOUT.....	6-18
PRINTRONIX.....	6-18
REFORMAT.....	6-18
RENAME-FILE.....	6-18
RUN.....	6-18
RUNOFF.....	6-18
SAVE.....	6-19
SAVE-LIST.....	6-19
SEL-RESTORE.....	6-19
SORT-LIST.....	6-19
SP-ALIGN.....	6-20
SP-ASSIGN.....	6-20
SP-CLEAR.....	6-20
SP-CLOSE.....	6-20
SP-COPIES.....	6-20
SP-CREATE.....	6-20
SP-DELETE.....	6-20
SP-DEVICE.....	6-20
SP-EDIT.....	6-20
SP-FQDELETE.....	6-20
SP-JOBS.....	6-20
SP-KILL.....	6-21
SP-LOOK.....	6-21
SP-MOVEQ.....	6-21
SP-OPTS.....	6-21
SP-RESUME.....	6-21
SP-STATUS.....	6-21
SP-STOP.....	6-21
SP-SUSPEND.....	6-21
SP-TRANSLATE.....	6-21
SP-SWITCH.....	6-21
SSM.....	6-21
START-PRINTER.....	6-22
STOP-PRINTER.....	6-23
STAT.....	6-23

T-ATT	6-23
T-DET	6-23
T-DUMP/T-LOAD	6-23
T-STATUS	6-23
USER.....	6-23

Chapter 7 ENGLISH and List Processing Commands

Masks Used in Value Limiters	7-2
Date Transformation.....	7-3
Displaying Date Conversions.....	7-3
New D Codes Available	7-3
Tfile Code - Translation	7-4
New and Modified ENGLISH and List Processing Verbs	7-5
Connectives.....	7-7
SORT Connectives	7-7
Grouping Connectives	7-7
Mask Character Operations	7-8
Other Modified Verbs.....	7-9
EDELETE Verb.....	7-9
R Option for ENGLISH	7-9
T-DUMP/T-LOAD Verbs.....	7-9
Messages	7-10
Message Formats.....	7-10
ENGLISH Reports	7-10

Chapter 8 PROC

PQ and PQN PROC Changes.....	8-2
File Buffers	8-2
Registers.....	8-2
[] Command	8-2
FB Command	8-2
GOSUB Command	8-2
IF Command.....	8-2
MV Command.....	8-3
ENGLISH Conversions in MV syntax	8-4
T Command.....	8-4

New PQN PROC Function 8-5
 SYSTEM() Function 8-5

Chapter 9 Networking

NETDEVS File 9-2

Chapter 10 DATA/BASIC

New Statements 10-2
 POSITION Statement 10-2
 READNEXT Statement 10-2
 VARVALSET Statement 10-3
 Modified Statements 10-4
 CONNECT Statement 10-4
 FOR Statement 10-4
 INCLUDE Statement 10-4
 LOOP Statement 10-5
 PAGE Statement 10-5
 PERFORM Statement 10-5
 PERFORM SELECTS 10-6
 PRECISION Statement 10-6
 READNEXT Statement 10-6
 RECWAIT Statement 10-7
 RQM/SLEEP Statement 10-7
 SELECT Statement 10-8
 SUBROUTINE Statement 10-8
 TRANSTART Statement 10-8
 New Intrinsic Functions 10-9
 FMT Function 10-9
 FOLD Function 10-9
 PTR Function 10-9
 ROUND Function 10-10
 TRUNC Statement 10-11
 Variable Checking Functions 10-11
 VARTYPE Function 10-11
 VARVAL Function 10-12
 VARVALTYPE Function 10-12

Modified Intrinsic Functions	10-13
CHAR function	10-13
DECRYPT and ENCRYPT Functions	10-13
SEQ Function	10-13
SPOOLER Function.....	10-13
SYSTEM(57) Function.....	10-14
SYSTEM(58) Function.....	10-14
SYSTEM(60) Function.....	10-15
SYSTEM(61) Function.....	10-15
New, Modified and Deleted Debugger Commands	10-16
Debug Displays.....	10-16
A Command.....	10-16
D Command	10-16
L Command	10-16
V Command.....	10-16
R Command	10-16
Using an Alternative Compiler	10-17
Using the BASIC Verb with the C option	10-18
Shared BASIC Object.....	10-20
Share Size	10-20
Share Modulo	10-20
Other DATA/BASIC Changes.....	10-21
Variables.....	10-21
Numeric Variables	10-21
Catalogued Programs.....	10-21
Indexing	10-21
Date Conversions	10-21
Mask Character Operations.....	10-22
Upgrade Utilities	10-22
Reserved Words.....	10-23
NEW TCL Commands.....	10-23
Modified TCL Commands.....	10-23
Obsolete TCL Commands	10-23

Chapter 11 Miscellaneous Changes

Checking for Despooler Errors.....	11-2
STAT-FILE	11-4
Management and File Reallocation.....	11-4
Statistics for Indexes	11-4
Macros in DICT STAT-FILE	11-4
On-line Documentation	11-5
RealityX On-line Help	11-5
RealityX On-line Help for Windows	11-5
Miscellaneous Changes	11-6
Lower case TCL Commands	11-6
File References and Q-pointers	11-6
Renaming Accounts	11-6
Options Processor.....	11-6
System Messages Formats.....	11-6
Database Configuration Files.....	11-6
reality -C Option	11-7

Appendix A PROC SYSTEM() Function Numbers

PROC SYSTEM() Function Number.....	A-2
------------------------------------	-----

Appendix B Printer Independent Functions

Introduction to Printer Independence	B-2
Printer Independent Function Classes	B-3
Example of a Prologue.....	B-3
Example of Using PTR in a Program	B-4

Appendix C TDM and Terminal Definitions

TDM and Terminal Definitions	C-2
------------------------------------	-----

Index

List of Figures

Figure 4-1. Index Structures in a Database.....	4-4
---	-----

List of Tables

Table B-1. Class 1: Device Control	B-5
Table B-2. Class 2: Page Format	B-7
Table B-3. Class 3: Line Format.....	B-8
Table B-4. Class 4: Tabulations	B-9
Table B-5. Class 5: Horizontal Control	B-10
Table B-6. Class 6: Line Spacing	B-11
Table B-7. Class 7: Character Pitch	B-13
Table B-8. Class 8: Form Layout.....	B-14
Table B-9. Class 9: Special Effects	B-15
Table B-10. Class 10: Typeface	B-17
Table B-11. Class 11: Character Set.....	B-18
Table B-12. Class 12: Font Selection.....	B-19
Table B-13. Class 13: Colour	B-20

Chapter 1

About this Manual

This chapter describes the purpose and contents of this manual, lists other manuals that may be useful, and the conventions used in text.

Purpose of this Manual

This manual summarises the differences seen by users upgrading from RealityX Release 3.1 to Release 4.0.

Differences described include two completely new features:

- Partition Database. An alternative to a File Store Database. in order to support large databases (>4 Gbyte).
- Shadow Database. An optional resilience feature which can be installed on a Transaction Logging database.

Other differences include:

- A substantial re-design of the Spooler.
- New, modified and obsolete TCL commands.
- Various changes to DATA/BASIC, ENGLISH, PROC, transaction handling and networking.

This manual provides information for users upgrading to RealityX Release 4.0 from Release 3.1, or from Series 18/19 (REALITY) to RealityX, in conjunction with the RealityX Differences Supplement which documents the differences between REALITY 7.0 and RealityX 3.1.

A list of associated references manuals is provided in this chapter.

Contents

Chapter 1, About this Manual, describes the purpose and contents of this manual and gives references to further information sources.

Chapter 2, Partition Database, describes the new Partition Database feature, listing the performance improvements obtained and describing the differences in administrative procedures.

Chapter 3, Shadow Database, specifies a new database resilience feature.

Chapter 4, Indexing, describes the new indexing features.

Chapter 5, Spooler, gives an overview of the new Release 4.0 Spooler.

Chapter 6, TCL Commands, describes new and modified TCL commands, and lists commands which have become obsolete with appropriate alternatives.

Chapter 7, ENGLISH and List Processing Commands, describes the changes made to ENGLISH elements and commands and to the list processing commands.

Chapter 8, PROC, describes changes made to file buffers, registers and PROC statements, as well as introduces the new SYSTEM() function.

Chapter 9, Networking, summarises the changes made to networking.

Chapter 10, DATA/BASIC, describes new and modified statements, intrinsic functions and other changes.

Chapter 11, Miscellaneous Changes, describes changes to disk configuration and the minimum and maximum size numbers used on the REALITY system.

Appendix A, SYSTEM() Function Numbers.

Appendix B, Printer Independent Functions.

Appendix C, TDM and Terminal Definitions.

References

RealityX Reference Manuals

Reality X Reference Manual Volume 1: General

RealityX Reference Manual Volume 2: Operation

RealityX Reference Manual Volume 3: Administration

ENGLISH Reference Manual (Vol. 4)

PROC Reference Manual (Vol. 5)

EDITOR Reference Manual (Vol. 6)

Screen Editor Reference Manual (Vol. 6)

DATA/BASIC Reference Manual (Vol. 7)

General Utilities and Printing (Vol. 8)

Manuals for 7.0/Pre-7.0 Upgrade

If you are upgrading from a Series 18/19 (REALITY) system, the following manuals may be helpful:

User's Guide to the REALITY Migration Utilities (UM30002112A)

RealityX Differences Supplement (UM70006051B). Describes differences between REALITY 7.0 and RealityX 3.1.

Conventions

The following conventions are used in this manual:

Text	Bold text in this typeface shows input to be typed at the terminal.
Text	Text in this typeface shows text that is output to the screen.
Bold text	Bold text in syntax descriptions represents characters typed exactly as shown. For example WHO
<i>text</i>	Characters or words in italics indicate parameters which must be supplied by the user. For example in LIST <i>file-specifier</i> the parameter <i>file-specifier</i> is italicized to indicate that you must specify an actual file.
<i>Document Title</i>	Italic text also indicates titles of documents referenced
{Braces}	Braces enclose options and optional parameters. For example in BLIST { DICT } <i>file-specifier</i> <i>item-id</i> {(options)} <ul style="list-style-type: none"> • <i>file-specifier</i> and <i>item-id</i> must be supplied • one or more single-letter options can be included, as defined for the command; these must be preceded by an open parenthesis, can be given in any order, and are not separated by spaces. Any number of options can be used except where specified in text.
[param param]	Parameters shown separated by vertical lines within square brackets in syntax descriptions indicate that at least one of these parameters must be selected. For instance, [THEN <i>statements</i> ELSE <i>statements</i>] indicates that either a THEN clause or an ELSE clause must be included (or both).

...	In syntax descriptions, indicates that the parameters preceding can be repeated as many times as necessary.
<i>file-specifier</i>	The parameter <i>file-specifier</i> represents the following: <i>{file-modifier} filename{,data-section-name}</i> <ul style="list-style-type: none">• <i>file-modifier</i> can be DICT to indicate the dictionary section of the file. With appropriate ENGLISH verbs, it can also be ONLY, TAPE, or WITHIN.• <i>data-section-name</i> can be used to specify a data section other than the default. In this case, DICT must be omitted.
SMALL CAPITALS	Small capitals are used for the names of keys such as RETURN.
CTRL+X	Two (or more) key names joined by a plus sign (+) indicate a combination of keys, where the first key(s) must be held down while the second (or last) is pressed. For example, CTRL+X indicates that the CTRL key must be held down while the X key is pressed. Enter To enter means to type text then press RETURN. For instance, 'Enter the WHO command' means type WHO, then press RETURN In general, the RETURN key (shown as ENTER or ↵ on some keyboards) must be used to complete all terminal input unless otherwise specified.
Press	Press single key or key combination, but do not press RETURN afterwards.
X'nn'	This denotes a hexadecimal value.

User Comments

A Comment Sheet is included at the front of this manual. If you find any errors or have any suggestions for improvements in the manual please complete and return the form. If it has already been used then send your comments to the Technical Publications Manager at the address on the title page, or email techpubs@northgate-is.com.

Chapter 2

Partition Database

This chapter describes the new Partition Database feature, listing the performance improvements obtained and describing the differences in administrative procedures.

Introduction

RealityX Release 3.1 constructs a database using a UNIX file system. Each RealityX file equates to a normal UNIX file, and each account is represented by a UNIX directory.

RealityX Release 4.0 also supports the above method for constructing a database. However, due to the limitation on file and database size in a UNIX file system (4 Gbyte on Series X), this method cannot be used to support very large databases (>4 Gbytes).

To enable very large databases, RealityX Release 4.0 supports a second method of database construction which uses one or more UNIX partitions per database without a UNIX file system being mounted to organise the RealityX files. A database constructed in this way is called a **Partition Database**. This is a software key enabled feature.

Both types of database, UNIX File Store Database and Partition Database, can coexist on the same machine. File Store Databases can still be used for small and medium sized databases. A Partition Database is used when a very large amount of data needs to be stored.

The type of database is determined when the database is constructed. Once made, the underlying type is transparent to general users and to applications. It does, however, have implications for administration and performance.

In addition to increasing the potential size of a database, a Partition Database also provides a number of other improvements in performance. These are listed below.

Comparative Benefits

Partition database allows databases of up to 16 Gbytes to be built, whereas the limit on file store databases is 4 Gbyte. Also, unlimited numbers of files can be opened, whereas file store databases have a fixed limit on the numbers of files open. Access to different parts of a file is allowed simultaneously, whereas on file store databases an update locks the whole file. Partition database supports a striping algorithm which distributes the database across multiple partitions, allowing the load to be shared by multiple disks.

However, a partition database does require a fixed disk partition to be assigned to it. File store databases can be located at any convenient point in a file system. Also, backups of partition databases cannot be done via the **cpio** utility.

Differences in Administrative Procedures

Normal user operation on a Partition Database is the same as a Release 3.1 File Store Database. Differences from Release 3.1 are mainly in the area of database administration.

Making a Database

In Release 4.0, the **mkdbase** utility looks in the file **/etc/realstab** to determine which type of database is required (Partition or File Store) and constructs the database accordingly.

realstab File

To create a partition database, you must create an entry in the file **/etc/realstab**. However, you must be superuser to edit this file.

The file **/etc/realstab** contains a list of Partition Databases on the system. Each entry comprises the name of a Partition Database and a list of partitions assigned to it. For example:

```
#comment
database1      (partition1  partition2  partition3)  #comment
                (partition4  partition5  partition6)
                + partition7
```

CAUTION

Ensure that no two database entries in **/etc/realstab** reference the same partition.

The rules for formatting **/etc/realstab** are as follows:

- A hashed (#) line is ignored.
- Whitespace lines are ignored.
- A database name may be an absolute path (e.g. **/user0/dbase1**) or a local ROUTE-FILE entry.
- Database names are left justified.
- Partitions are absolute paths to block special devices (e.g. **/dev/dsk/m328_c0d1s2**).
- Partitions are separated by whitespace.
- A partition is assigned to the last listed database.

Saving and Restoring a Database

RealityX Save and Restore procedures, executed at TCL, must be used to save and restore a Partition Database. The **cpio** backup utility cannot be used for this purpose. If you are currently using **cpio** to backup your RealityX File Store Database, you will need to change your procedures if you want to upgrade to a Partition Database.

Database Error Checking

The **realdbck** utility can be used to check a Partition Database for consistency and fix some errors caused by a database failure. For example, if a system crashes, or connected processes terminate prematurely in mid- update. The usage of **realdbck** is described in the next section.

Monitoring Disk Space

Two TCL commands are provided to enable SYSMAN/SYSPROG users to monitor disk space on a partition database. They are.

%FULL
POVF

Refer to Chapter 6 for more details.

realdbck utility

Purpose

To check the consistency of a Partition Database and attempt to fix errors found.

CAUTION

realdbck will validate and repair the physical structure of a database, but it will not report GFE's and restore data loss. To check for GFE's you must execute the ALL-FILE-STATS command. To ensure complete database integrity, it is recommended that you remake the database, restore the last FILE-SAVE and roll back clean logs.

Syntax

realdbck [*options*] [*database*]

Options

- a** Allows **realdbck** to be run on an active database without logging users off. It is recommended that you use the **-n** option, as well.
- c** Checks only that the database is marked as having been shutdown cleanly. Does not do a full check.
- l *num*** Sets limits on the number of errors. Abandons the check/repair if the limit is reached.
- n** Assumes the answer is "no" to any question. This ensures no modifications are made to the database.
- s** Marks the database as having been shutdown cleanly (For database daemon use only).
- S** Marks the database as dirty and does not carry out the check sequence (For database daemon use only).
- v** Verbose mode. If not selected, messages to terminal will refer only to errors.
- y** Assumes the answer is "yes" to any "Are you sure?" type question.
- z** Discards the existing free space table before starting the check.

The **-S** and **-s** options are used by the database daemon only at start and end of a session. Both fail silently if the database is not a Partition Database.

Parameters

database is the absolute UNIX path name of the database to be checked, or its **ROUTE-FILE** entry identifier. If you do not specify *database*, **realdbck** checks the default database in **REALDBASE**.

Comments

realdbck logs off all database users before starting its check, unless you use the **-a** option.

realdbck with the **-c** option is executed by the database daemon each time a database is started up. If the check fails, then the utility must be run again to clean up the database. The degree of checking and salvaging is user selectable using the options described above.

Checking

realdbck maintains a small structure describing each block in the database, all initially null. The status of each block is established and recorded. Duplicate references and lost blocks can be located. Approximately 8 Kilobyte of work space per Gigabyte of database is required.

realdbck walks through all database files, beginning with the first group of the SYSTEM file. For each group it checks off each block used for in-group space. Assuming there are no GFEs, it checks the:

- D-pointer, recursing down that section.
- Out-of-group item with less than 127 contiguous block, checking off referenced blocks without reading them.
- Fragmented out-of-group item, checking off referenced blocks by reading only the tails of each fragment.
- In-group item.

Repair

realdbck cleans up inconsistent data blocks and transfers them to the free block list. Its main purpose is the quick recovery of the free space table, black-holed files and out-of-group items. Black Holes are unreferenced blocks, or groups of blocks.

If your database is badly damaged it is recommended that you rebuild it using **mkdbase** with the **-r** option, then restore the last backup tape. **realdbck** is not equipped to repair a badly damaged database.

Types of Errors

The following types of errors can be found by **realdbck**:

- **Group Format Errors (GFEs).** These can be caused if a multi-block operation is interrupted. A GFE would black-hole any branches from that group.
- **Black Holes.** These are unreferenced blocks, or groups of blocks. If the blocks are marked as being type free or unknown, the hole can be cleaned up and added to the free list.
- **Duplicate References.** These may arise from an aspect of delayed-write in which the blocks are not synchronised to disk in correct order. For example, when an out-of-group item is deleted and its space assigned to a new item, if the older in-group reference was not synchronised, but the new one was, the item body would be double-referenced. Similarly a block might be on the free list as well as in a file.
- **Incorrect Linkage of blocks.** These occur in the same way as Duplicate References.

Chapter 3

Shadow Database

Shadow Database is an optional software facility which provides enhanced database resilience in the event of a system or disk failure. The level of resilience provided falls between that provided by Transaction Logging and that provided by FailSafe software. Shadow database is supported on a stand-alone system and provides a method of recovering the database without having to perform a full database restore, thus minimising downtime. Transaction Logging is an intrinsic part of Shadow Database operation and is therefore a pre-requisite for its implementation. A description of Shadow Database and how to administer it, is provided in the *RealityX Resilience Reference Manual (UM70006467A)*.

Chapter 4

Indexing

This chapter describes the new indexing feature, which is available as an option in RealityX Release 4.0.

Introduction to Indexing

Indexing is a RealityX feature which enables the creation and use of indexes in order to increase speed of access to file data. RealityX processors, including ENGLISH and DATA/BASIC, can use indexes. Also TCL, using the SELECT-INDEX verb, provides an active list, usable by TCL-II and ENGLISH.

Note: Indexing is a purchasable feature enabled by a software key. If it is not enabled on your system, see your Northgate Support representative.

An index is a data structure containing the item-ids of all items in the associated data section that match specified selection criteria, ordered according to specified sort criteria. The setting up of an index for a data section is carried out in two stages:

1. You define the index.
2. You create the index.

Once set up, the index is automatically maintained by any process that modifies the data in the associated data section.

TCL Verbs for Managing Indexing

The following TCL verbs are provided to set up, remove, verify and select from an index:

DEFINE-INDEX

CREATE-INDEX

DELETE-INDEX

VERIFY-INDEX

SELECT-INDEX

Their use is described in the sections following. Refer to Chapter 6 for detailed descriptions of these commands.

Indexing Functions

Defining an Index To define an index you must use the DEFINE-INDEX verb. This creates an index definition item in the associated file dictionary with the index name as its item-id.

Index definition item The index definition item contains both the source and a compiled form of the selection and sort criteria. These are specified via the DEFINE-INDEX command exactly as they would be in an ENGLISH command. The referenced data definitions are compiled as defined when DEFINE-INDEX was executed. Refer to Chapter 4 in *RealityX Reference Manual Volume 1* for a description of the index definition item structure.

Note that the definition item only defines the index structure, it does not actually create the index.

If you create an index definition and then change a data definition item, the index definition is not changed. It still reflects the data definition item at the time the index definition was created.

You should avoid referencing data definition items that return data such as the current date or a file translation, where the data in the translate file might change, as these can produce an inconsistent index.

Creating an Index To create an index, you must use the CREATE-INDEX verb and supply the index name (index definition item-id) as a parameter. This creates an index section with an entry for each item in the specified data section that matches the selection criteria. The entries are ordered in the index according to their sort 'key value', developed from the sort criteria. Each key value is made up of the values of the attributes sorted on, if any, with the item-id as the least significant portion. If sort criteria are not specified, then the item-id is the key value.

Verifying an Index To verify that an index is created correctly for a particular data section, you use the VERIFY-INDEX verb. VERIFY-INDEX reads each item in the index and checks that the data in the item generates a key value with the same value as that stored in the index.

Deleting an Index To remove an index section you must use the DELETE-INDEX verb. To remove the index definition as well, you must use the ENGLISH verb EDELETE.

Changing an Index To change an index definition, you must first delete the index section using DELETE-INDEX and the index definition item using EDELETE. Then, if required, change any data definition items, redefine the index using DEFINE-INDEX and recreate the index section using CREATE-INDEX.

Indexes in the Database File Structure

The index definition item created by DEFINE-INDEX resides in the file dictionary of the data file with which it is associated. The index section itself is closely coupled to the data section to which it relates. The D-pointer for the data section, also points to its index section(s). The construction and maintenance of an index section is transparent to the user in normal operation.

The index structures associated with a file dictionary are illustrated below in Figure 5-1.

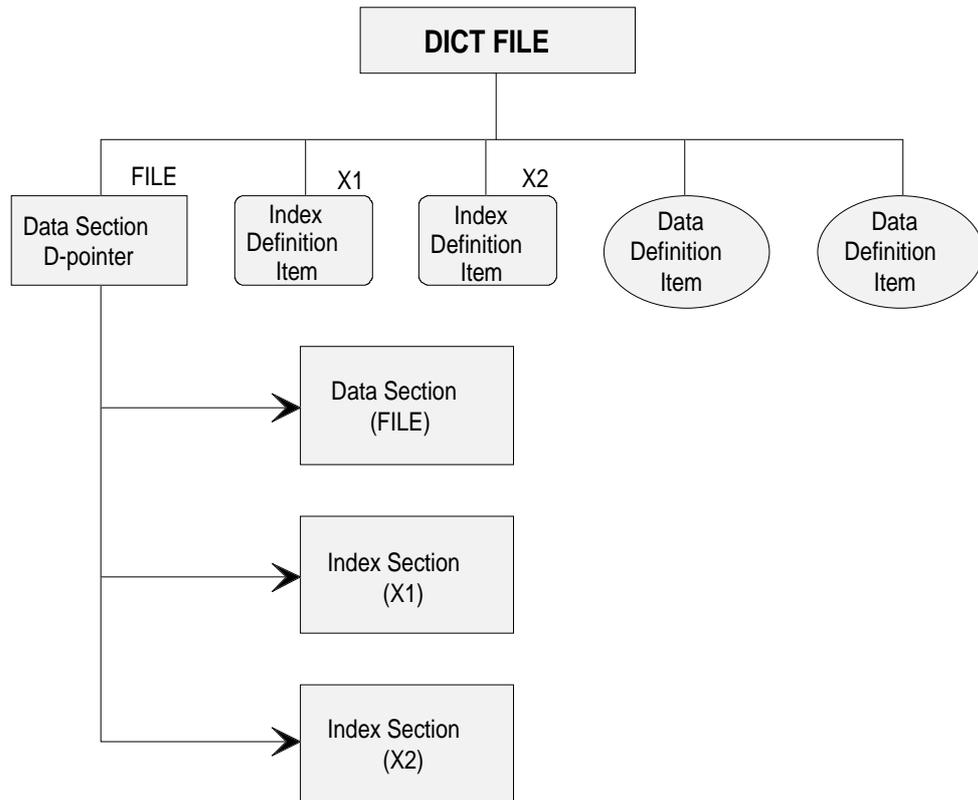


Figure 4-1. Index Structures in a Database

Management and Maintenance of Indexes

The following points should be noted when using indexes:

- Only data sections may be indexed.
- Index sections may not be opened, read or modified explicitly.
- Indexes are transaction logged if the associated file is logged.
- Indexes are file-saved and file-restored if the associated file is saved and restored.
- Indexes should be created only when a file is not being updated, so as to prevent invalid data being compiled in the index.

Maintenance of Indexes

Once an index has been created, it is maintained automatically by RealityX. Each time RealityX opens a data section, it checks the D-pointer for associated indexes, and whenever an item is updated all index entries are checked. It calculates the old and new key values, and if the key value has changed, it deletes the old index entry and adds the new one. However, if the item no longer matches the selection criteria, or if it has been deleted, its entry is removed altogether from the index.

File Statistics for Indexes

File statistics listings generated by routine database backup procedures show index sections after the associated data section, separated by colons (:). These statistics allow you to see what index sections exist and how much space they take up.

The LISTFILES verb, which lists all files defined via a D-pointer from the current account (or a named dictionary), also lists index sections.

Effect of Creating Multiple Indexes

Any number of indexes may be associated with a data section. However, the number of indexes to a file should be kept to a minimum because:

- Each index definition has to be checked when a file item is updated.
- Each index takes up disk space dependent on:
 - The number of items in the index
 - The average size of item-ids
 - The sort criteria used and the size of values in the attributes sorted on.

Accessing an Index

An index can be explicitly accessed by TCL or DATA/BASIC. Both processors can reference any number of entries or range of entries across an index.

Using SELECT-INDEX

Using the SELECT-INDEX verb, you can generate an active item list from an index. The verb allows you to specify individual key values, or ranges of key values. Refer to Chapter 6 for a description of SELECT-INDEX.

Using DATA/BASIC

To access an index from DATA/BASIC, you attach a variable to an index with the SELECT statement. You can then read the next or previous item-id relative to a pointer via the READNEXT or the READPREV statements. Both statements give you the option of retrieving the key value for each item-id. You can use the POSITION statement to move the pointer within the index. For descriptions of these statements refer to the *DATA/BASIC Reference Manual*.

Implicit Use of Indexes by ENGLISH

Where a data section has associated indexes, ENGLISH may use one of the indexes to access the data section. This increases the speed of execution by the ENGLISH processor. In order for ENGLISH to be able to use an index implicitly, the selection and sort criteria of the index must correspond to the ENGLISH statement's selection or sort criteria according to the following rules.

Rules for Implicit Use of an Index

1. The index must contain all items in the file (its definition does not include selection criteria).
2. The selection in the ENGLISH statement is on the item-id: an index sorted primarily by item-id is then used.
3. The selection in the statement is on a specific attribute: an index sorted primarily by that attribute is then used.
4. The selection in the statement must be based on a left-justified complete or partial string.
5. If selection criteria are ANDed, either for the item-id or for the Data Definition Item, the use of the index is based on the left-most ANDed value.
6. The index is not used if there is ORed selection.
7. If there is no selection but the sort criteria matches an index definition exactly, the index is used.

Examples of Implicit Use of Indexes

In the following examples ENGLISH uses a suitable index if one is available. Note that item-id selection always precedes and is implicitly ANDed with attribute selection criteria.

```
LIST FILE = '12']'
```

Uses an index that has no selection criteria, so that it contains all items in the file, which is also sorted by item-id. Rules 1, 2 and 4 are obeyed.

```
LIST FILE = '[12]'
```

Does not use an index as the selection criterion in the statement is not left-justified. Rule 4 is obeyed.

```
LIST FILE = '12]' AND = '[X'
```

Uses an index as in the first example. Rules 1, 2, 4 and 5 are obeyed.

```
LIST FILE = '12]' = '3]'
```

Does not use an index as the item selection criteria in the statement are implicitly OR'd. Rule 6 is obeyed.

```
LIST FILE WITH DDI = "12]"
```

Uses an index that contains all file items, sorted primarily by attribute DDI. Rules 1 and 3 are obeyed.

```
LIST FILE = '12]' WITH DDI = "[ABC]"
```

Uses a complete index sorted primarily by item-id. Rules 1, 2, 4, and 5 are obeyed.

```
LIST FILE WITH DDI = "12]" AND = "[X"
```

Uses a complete index based primarily on attribute DDI. Rules 1, 3, 4, and 5 are obeyed.

```
LIST FILE WITH DDI = "12]" = "3"
```

Does not use an index as there is an implicit OR in the statement. Rule 6 is obeyed.

```
LIST FILE WITH DDI = "12]" AND WITH YYY = "ABC"
```

Uses a complete index sorted primarily on DDI. Rules 1, 3 and 5 are obeyed.

```
LIST FILE WITH DDI = "12]" WITH YYY = "ABC"
```

Does not use an index because the selection criteria are implicitly OR'd. Rule 6 is obeyed.

```
SORT FILE
```

Uses a complete index sorted primarily by item-id. Rules 1 and 7 are obeyed.

```
SORT FILE BY DDI
```

Uses a complete index sorted primarily on DDI. Rules 1 and 7 are obeyed.

Chapter 5

Spooler

This chapter gives a brief overview of how the 4.0 Spooler differs from the Release 3.1 version. For a detailed description of the 4.0 Spooler, refer to *RealityX Reference Manual Volume 2: Operation*.

List of New Features

The 4.0 Spooler supports the following new features:

- Spooler data is held in an ordinary RealityX files, designated job files, enabling it to be managed using standard RealityX file handling methods.
- A job file can be specified as any file in a user's account or the system default file, SPOOL.JOBS.
- A job name can be assigned to a user process. The assigned name is then prefixed to all print job identities for that process.
- A print job comprises two RealityX items; one in the PRINT.JOB.CONTROL file, containing job control information, and the other in a job file, containing the print data. Both items have the same item-id.
- Formqueues and despooler processes are defined and maintained separately.
- A despooler is defined by an item in the DESPOOLER.CONTROL file.
- A formqueue is a RealityX item containing a list of print jobs as a multi-valued attribute, and a set of characteristics which are assigned to the queued jobs at print time.
- Multiple despoolers can be assigned to one formqueue.
- Multiple formqueues can be serviced by one or more despoolers.
- Two types of formqueues can be defined, public and private. A public formqueue is an item in the FORM.QUEUES system file. A private formqueue is an item in any user-specified RealityX file which can be protected from access by other users using standard update and retrieval lock codes.
- Printer independence facilities are provided.
- Prologues and epilogues, which are instructions to be performed before or at the end of a print job, can be defined for a particular formqueue. A prologue can be defined for a particular despooler.
- Printer command strings (PCS) can be created, and appended to a print job as a prologue or epilogue. The PCS source items are held in the PCS file. The compiled PCS output is held in the PCS.OUTPUT file.

- Remote spooling across a network is supported.
- Network despooling with printer sharing is also supported.
- Transaction Logging can be applied to Spooler data.
- Network printers can be shared.
- Despooler monitor utility DSPMON

Design Changes

The 4.0 Spooler records all its data in conventionally-structured RealityX files. This gives you a simpler, more general method for managing print data. The print data, however, is still protected against direct update by a RealityX editor or other updating processor.

This is a significant difference from the 3.1 Spooler which stores print data in special structures that are different from the standard files in the system. This means that in Release 3.1 you can not manipulate Spooler data as you can other files.

Another important new feature of the 4.0 Spooler is that it separates the despooling process and the formqueue, whereas in Release 3.1 there is no real distinction between the formqueue and the output device or despooler. The formqueue and the despooler appear to be synonymous.

In Release 4.0, the formqueue is an item containing a list of queued job ids and a set of characteristics that are assigned to the queued jobs at print time. A print job can be moved from one formqueue to another, in which case it takes on the characteristics of its new queue at print time. The despooler is defined by a separate definition item.

At some point a despooler must be assigned to a formqueue so that jobs can be output, but this can be done after the job has been created.

Furthermore, several despoolers can be assigned to a formqueue and several formqueues can be assigned to a given despooler. If a print job is put into a formqueue with several despoolers, the job will be output by the first despooler that can take that job out of the queue.

On the other hand, if a job is put into a queue that is one of several assigned to a despooler, the job could be intermingled with jobs from other queues.

Compatibility

As much as possible, the Spooler has been kept compatible with previous REALITY/RealityX releases. However, some functionality has been made obsolete by this release.

The REALITY and RealityX user interfaces with the Spooler are compatible.

Structural Changes

The 4.0 Spooler uses conventionally structured RealityX files to store Spooler definitions and print data. These comprise:

- SPOOL.JOBS file (the system default job file) or some other user-specified job file which holds print data for print jobs.
- PRINT.JOB.CONTROL file which contains the control information for a print job.
- FORM.QUEUES file which contains public formqueue definitions and Q-pointers to private formqueue definitions. Private formqueue definition items can reside in a file in user's account. The system still has a default queue named STANDARD.
- DESPOOLER.CONTROL file which contains the despooler definition items.
- PCS.OUTPUT file which contains strings that can be specified as a formqueue prologue or epilogue or a despooler prologue.
- PCS file which contains DATA/BASIC source code for specialised programs. When executed the programs in PCS produce the string items in file PCS.OUTPUT.
- PRINTER file which contains items that provide printer independence characteristics.

All of these files are in SYSFILES.

The data structures are compatible between REALITY and RealityX.

Because the formqueue and print job data are stored in RealityX files, the data can be handled by the system's saving and restoring processes and by Transaction Logging. However, you will need extra care when performing day-to-day file management functions. For example, clearing file PRINT.JOB.CONTROL deletes the job control data only. The system will not update the associated job data files. You have to provide this capability.

TCL Commands

Note: For a detailed description of all TCL commands supported by the Release 4.0 Spooler, refer to *RealityX Reference Manual Volume 2: Operation*.

New Commands

The following Spooler commands are new in Release 4.0. See also the topic *Description of New TCL Commands* in Chapter 6.

BUILD.DESPOOLERS	FQM	SP-FIX
DESPOOLER.DETAIL	SP-COPY	SP-FORMQUEUE
DSPMON	SP-DESCRIPTION	SPM
PCSM	SP-DESPOOLERS	START-DESPOOLER
RESET.DESPOOLER	SP-DETAIL	STOP-DESPOOLER

Changed Commands

The following Spooler commands have been changed for Release 4.0. See also the topic *Description of Differences in Changed TCL Commands* in Chapter 6.

PORT-DESPOOL	SP-DEVICE	SP-RESUME
SP-ALIGN	SP-EDIT	SP-STATUS
SP-ASSIGN	SP-FQDELETE	SP-STOP
SP-CLEAR	SP-JOBS	SP-SUSPEND
SP-CLOSE	SP-KILL	SP-SWITCH
SP-COPIES	SP-LOOK	SP-TRANSLATE
SP-CREATE	SP-MOVEQ	START-PRINTER
SP-DELETE	SP-OPTS	STOP-PRINTER

New Options

There are several new options to accommodate this version's approach. The following are used by several commands:

- F option to specify a job filename explicitly.
- J option to specify a job-id.
- U option to reference all jobs regardless of job file (this can be used in `SYSMAN` or `SYSPROG` accounts only).

**Obsolete
Commands**

The following commands are now obsolete:

- `CLEAR-SP-LOCK`
- `FIX-SP-ERRORS`
- `SP-FORM`
- `SP-TYPE`
- `:SP-NEWTAB`
- `START-NET-PTR`

Job File Feature

Job File Determines References

In RealityX Release 3.1, job-related Spooler commands, such as SP-JOBS or SP-MOVEQ, can reference all print jobs in the Spooler.

In Release 4.0, job-related Spooler command can only reference print jobs in one job file at a time, designated the "specified job file", unless the U option is used.

At logon, the specified job file is the system default job file SPOOL.JOBS, unless a different default is specified in a formqueue's definition item.

The job file can be specified in the SP-ASSIGN command statement. This overrides the default job file and remains the specified job file unless it is overridden by a job-related command with the F option.

Some job-related commands allow you to specify the F option to name a job file for reference. This overrides any previously assigned job file for the duration of the command execution.

In the SYSMAN and SYSPROG accounts, some job-related commands can be executed with the U (universal) option which enables them to reference all existing jobs.

The SP-JOBS command also has an action code which allows you to change the specified job file for the duration of the SP-JOBS execution.

Using ENGLISH on a Job file

You can now use ENGLISH commands to reference print jobs. For example, you can enter the following command:

```
:SORT SPOOL.JOBS WITH SEL.PF.STATUS = "QUEUED" AND WITH PF.QUEUE = "LASER"
```

The Data Definition Items for use with ENGLISH commands are defined in DICT SPOOL.JOBS.

File Utilities

You can use commands to perform file operations on Spooler files. For example, you can enter the following sequence:

```
:SELECT SPOOL.JOBS WITH STATUS = "Finished"  
>SP-DELETE JOB-FILE 1 4 7 (F
```

Miscellaneous Changes

Security

To ensure the security of print data, a print job is flagged as a Spooler file which protects it from being changed by any of the RealityX processors. Only the print job control data can be updated.

Password protection of print jobs has been removed. Instead, print jobs, which are now conventionally structured Reality files, can be protected using the standard retrieval/update lock mechanism.

Spooling to a Remote System

The 4.0 Spooler allows a process to spool directly to a job file in another database. This can be on the same system or on another system across a network.

Network Printing with Printer Sharing

In addition to remote spooling, the RealityX 4.0 Spooler also supports remote despooling to a shared printer on another host. Sharing of the same printer between different databases /hosts is enabled using a Network Printing Utility (**npu**) to drive the printer connection.

Printer Independence

The Spooler now accommodates printer independence. Printer independence characteristics are held in the PRINTER file.

Prologues and Epilogues

Printer instructions to be performed before or after each job or when the despooler first starts can be set up using SPM. These are held in PCS.OUTPUT.

Chapter 6

TCL Commands

This chapter describes the differences in the TCL command set supported by RealityX Release 4.0, compared to Release 3.1. It describes new commands in Release 4.0, and 3.1 commands which have been changed in Release 4.0. It also lists Release 3.1 commands which are now obsolete for Release 4.0.

For a detailed description of Release 4.0 TCL commands discussed in this chapter, refer to the *RealityX 4.0 Reference Manuals*, listed in Chapter 1 under *References*.

Note that TCL commands can be entered in lower-case or upper case letters. However, since upper-case is the traditional method, the user documentation shows them in upper-case format.

List of New Commands

The following is a list of new TCL commands introduced in Release 4.0. Brief descriptions of commands are provided in this chapter. For a detailed description, refer to the appropriate manual in the RealityX 4.0 reference manual set.

%FULL	LOAD-ALPHA
AND-ITEMS	LOAD-BNF
AND-LISTS	MAN
BUILD.DESPOOLERS	NSELECT
CLEAR-OPTION	OR-ITEMS
CONVERT.OBJECT	OR-LISTS
CREATE-INDEX	PCSM
DEBUG	PDM
DECAT	POVF
DEFINE-INDEX	RESET-ALPHA
DELETE-INDEX	RESET.DESPOOLER
DESPOOLER.DETAIL	SELECT-INDEX
DISCTOTAPE	SET-ALPHA
DSM	SET-OPTION
DSPMON	SHOW-MODULI
FQM	SORT-SPREAD
GO	SP-COPY
LIST-SPREAD	SP-DESCRIPTION

SP-DESPOOLERS

SP-DETAIL

SP-FORMQUEUE

SPM

START-DESPOOLER

STOP-DESPOOLER

TAPETODISC

UPGRADE.BASIC.OBJECT

VERIFY-INDEX

XOR-ITEMS

XOR-LISTS

XSELECT

List of Changed Commands

The following is a list of TCL commands which have changed functionality in Release 4.0 from Release 3.1. A description of the differences is given later in this chapter under the topic *Description of Differences in Modified Commands*.

.(Period)	PORTOUT
TCL Stacker	PRINTRONIX
ASSIGN	REFORMAT
BASIC	RENAME-FILE
BLIST	RUN
BLOCK-PRINT	RUNOFF
BLOCK-TERM	SAVE
BVERIFY	SAVE-LIST
CATALOG	SEL-RESTORE
CREATE-FILE	SORT-LIST
DB	SP-ALIGN
FILE-SAVE	SP-ASSIGN
FIND	SP-CLEAR
LISTFILES	SP-CLOSE
LOGTO	SP-COPIES
MSG	SP-CREATE
NEW-SORT-LIST	SP-DELETE
PORT-DESPOOL	SP-DEVICE

SP-EDIT	SP-TRANSLATE
SP-FQDELETE	SSM
SP-JOBS	START-PRINTER
SP-KILL	STAT
SP-LOOK	STOP-PRINTER
SP-MOVEQ	T-ATT
SP-OPTS	T-DET
SP-RESUME	T-DUMP
SP-STATUS	T-LOAD
SP-STOP	T-STATUS
SP-SUSPEND	USER
SP-SWITCH	

List of Obsolete Commands

The following are Release 3.1 TCL commands which are no longer supported by Release 4.0.

CLEAR-SP-LOCK

EBASIC

FIX-SP-ERRORS

LOADBNF (Replaced by LOAD-BNF)

PRINT-HEADER

:SP-NEWTAB

SP-FORM

SP-TYPE

START-NET-PTR

STOREPF

Descriptions of New TCL Commands

- %FULL** Reports the percentage of disk space allocated to a Partition Database that is in use, or the percentage of the partition containing a Filestore Database that is currently in use. It is restricted to SYSMAN and SYSPROG accounts. See *RealityX Reference Manual Vol. 3* for details.
- AND-ITEMS** Returns a list comprising all attribute values contained in each of two or more items, or a specified number of times altogether, that is, the intersection of the values. See *RealityX Reference Manual Vol. 4* for details.
- AND-LISTS** Returns a list comprising all attribute values contained in each of two or more lists, or a specified number of times altogether, that is, the intersection of the lists. See *RealityX Reference Manual Vol. 4* for details.
- BUILD.
DESPOOLERS** Creates despooler records in the DESPOOLER.CONTROL file for specified ports and tapes TAPE1 to TAPE16. See *RealityX Reference Manual Vol. 2* for details.
- CLEAR-OPTION** Resets one or all process options previously set by the SET-OPTION command. See *RealityX Reference Manual Vol. 1* for details.
- CONVERT.
OBJECT** Used in the upgrade procedure from SYS-UPDATE to convert pre-4.0 intermediate object code (\$/£-item) to the 4.0 format. It should not be run from TCL. See *RealityX Reference Manual Vol. 7* for details.
- CREATE-INDEX** Creates an index to a particular data section, based on a specified index definition. See *RealityX Reference Manual Vol. 1* for details.
- DEBUG** Breaks to the DATA/BASIC symbolic debugger. This command replaces the D option of the RUN command and the D option used to run cataloged programs. See *RealityX Reference Manual Vol. 7* for details.
- DECAT** Used in the upgrade procedure from SYS-UPDATE and should not be run from TCL. It produces intermediate object code (\$/£-item) from a cataloged DATA/BASIC program, which usually resides in the POINTER-FILE, but may also reside in the program file. See *RealityX Reference Manual Vol. 7* for details.
- DEFINE-INDEX** Creates an index definition item in the dictionary of a data file to be indexed which defines the name of the index and the selection and sort criteria to be used to create it. See *RealityX Reference Manual Vol. 1* for details.

DELETE-INDEX	Deletes a particular index section previously generated by the CREATE-INDEX verb. See <i>RealityX Reference Manual Vol. 1</i> for details.
DESPOOLER .DETAIL	Lists information about all despoolers on a database. See <i>RealityX Reference Manual Vol. 2</i> for details.
DISCTOTAPE	Writes data items created on disk by the TAPETODISC command back to tape so as create an identical binary image of the original tape. See <i>RealityX Reference Manual Vol. 2</i> for details.
DSM	Displays the DESPOOLER.CONTROL FILE MAINTENANCE screen to create, change and delete despooler definition items. See <i>RealityX Reference Manual Vol. 2</i> for details.
DSPMON	Used to manage the Despooler Monitor process. See <i>RealityX Reference Manual Vol.2</i> for details.
FQM	Displays the FORM.QUEUES FILE MAINTENANCE screen to create, change and delete formqueue definition items. See <i>RealityX Reference Manual Vol. 2</i> for details.
GO	Enables access to a PROC from TCL without the PROC name being in the current account MD. See <i>RealityX Reference Manual Vol. 1</i> for details.
LIST-SPREAD	Generates an ENGLISH listing in the same way as the LIST verb. However, instead of putting spaces between columns, it inserts a tab character and it does not supply a paged output. The syntax is the same as for the LIST verb. A SORT-SPREAD verb is also supported. See <i>RealityX Reference Manual Vol. 4</i> for details.
LOAD-ALPHA	Loads an item generated via DENAT menu selection 'Make Alphabet'. See <i>RealityX Reference Manual Vol. 1</i> for details.
LOAD-BNF	Loads the DATA/BASIC compiler BNFS. This command replaces the STOREPF command. See <i>RealityX Reference Manual Vol. 7</i> for details.
MAN	Displays on-line documentation. See <i>RealityX Reference Manual Vol. 1</i> for details.
NSELECT	Compares the values in an active list against the item-ids of a specified file and returns a list of all values that are not item-ids in the file. See <i>RealityX Reference Manual Vol. 4</i> for details.
OR-ITEMS	Returns a list comprising one copy of any value contained in any of two or more items, that is, the union of the values. See <i>RealityX Reference Manual Vol. 4</i> for details..

OR-LISTS	Returns a list comprising one copy of any value contained in any of two or more lists, that is, the union of the lists. See <i>RealityX Reference Manual Vol. 4</i> for details.
PCSM	Displays the Print Control String Maintenance screen to create, change and delete printer command strings in file PCS.OUTPUT. See <i>RealityX Reference Manual Vol. 2</i> for details.
PDM	Used to create, modify or delete printer definition items in the PRINTER file. See Appendix B or, for full details, the <i>Printer Definition Maintenance</i> manual.
POVF	Reports on free disk space available within current partition (Filestore Database) or current database (Partition Database). See <i>RealityX Reference Manual Vol. 3</i> for details.
RESET-ALPHA	Resets the complete alphabet structure to the standard ASCII alphabetic character set. See <i>RealityX Reference Manual Vol. 1</i> for details.
RESET. DESPOOLER	Resets the despooler system. See <i>RealityX Reference Manual Vol. 2</i> for details.
SELECT-INDEX	Generates a list of all selected item-ids an index, sorted in ascending sequence. This list can then be supplied to an immediately-following ENGLISH or TCL-II verb, including SAVE-LIST, in the usual way. See <i>RealityX Reference Manual Vol. 1</i> for details.
SET-ALPHA	Sets a specified range of hexadecimal values to be alphabetic for the executing process. See <i>RealityX Reference Manual Vol. 1</i> for details.
SET-OPTION	Sets one of a number of special options for the duration of the current session of the executing process. See <i>RealityX Reference Manual Vol. 1</i> for details.
SHOW-MODULI	Displays a series of prime numbers that can be used as modulus, from one to greater than 16 million. Each number differs from the previous number by at least 10%. See <i>RealityX Reference Manual Vol. 1</i> for details.
SORT-SPREAD	Generates an ENGLISH listing in the same way as the SORT verb, except that tab characters are inserted between columns and output is not paged. The syntax is the same as for the SORT verb. See <i>RealityX Reference Manual Vol. 4</i> for details.
SP-COPY	Copies a job's print data from one job file to another. See <i>RealityX Reference Manual Vol. 2</i> for details.
SP- DESCRIPTION	Sets the default description for all jobs created by a process. See <i>RealityX Reference Manual Vol. 2</i> for details.

SP-DESPOOLERS	Lists all despoolers on a database with their status and associated form queues. See <i>RealityX Reference Manual Vol. 2</i> for details.
SP-DETAIL	Displays the control data for jobs in the specified job file. See <i>RealityX Reference Manual Vol. 2</i> for details.
SP-FIX	Cleans up print jobs which have been left in an inconsistent state as a result of a program crashing. See <i>RealityX Reference Manual Vol. 2</i> for details.
SP-FORMQUEUE	Lists all the print jobs in a specified form queue in order of priority. See <i>RealityX Reference Manual Vol. 2</i> for details.
SPM	Used to create, change or delete form queue definition items, despooler definition items and printer command strings. See <i>RealityX Reference Manual Vol. 3</i> for details.
START-DESPOOLER	Starts a despooler. It is synonymous with START-PRINTER, retained for compatibility. See <i>RealityX Reference Manual Vol. 2</i> for details.
STOP-DESPOOLER	Stops a despooler. It is synonymous with STOP-PRINTER, retained for compatibility. See <i>RealityX Reference Manual Vol. 2</i> for details.
TAPETODISC	Reads data from tape to disk, creating a database item for each tape block. Any type of data can be written, including binary data. See <i>RealityX Reference Manual Vol. 2</i> for details.
UPGRADE.BASIC.OBJECT	Searches the POINTER-FILE for cataloged programs and converts to the latest format. See <i>RealityX Reference Manual Vol. 7</i> for details.
VERIFY-INDEX	Verifies an index with respect to the specified data , according to the specified index definition See <i>RealityX Reference Manual Vol. 1</i> for details.
XOR-ITEMS	Returns a list comprising all values contained only once in all the items referenced. A value occurring more than once in one item will not be included in the result. See <i>RealityX Reference Manual Vol. 4</i> for details.
XOR-LISTS	Returns a list comprising all values contained only once in all the lists referenced. A value occurring more than once in one list will not be included in the result. See <i>RealityX Reference Manual Vol. 4</i> for details.
XSELECT	Compares the values in an active list against the item-ids of a specified file and returns a list of item-ids in the file that are not in the list. See <i>RealityX Reference Manual Vol. 4</i> for details.

Description of Differences in Changed TCL Commands

The following differences from Release 3.1 are supported by Release 4.0 TCL commands.

. (Period) TCL Stacker

The command **.Xn** at TCL executes the *n*th command in the stack and moves that command to the top of the stack.

ASSIGN

The ASSIGN verb now supports the following options for 1/2 inch tape units:

L Leave tapes loaded, until all tapes have been written once. If the tape system attempts to wrap around and write on the first tape, then all units are unloaded. Thereafter, each reel is unloaded on completion of rewind.

M Logs all tape errors including recovered errors

Multiple tape units can be attached to a channel. The syntax of ASSIGN contains the parameter unit-list which is the number of tape(s) to be used, separated by commas.

BASIC

Options supported by the BASIC verb have changed, as follows:

B and I No longer available.

C New option that prompts for a compiler name (replaces the LOADBNF PROC). Refer to the topic *Using an Alternative Compiler* in Chapter 10.

M No longer outputs a statement map of the program.

R New option that creates an executable image of the program in the file in which it is compiled (£*item-id* or \$\$*item-id*). If you catalog this program with option R and then recompile it with option R, you do not have to re-CATALOG the program.

Also, there is no longer a maximum size for a DATA/BASIC object code item.

BLIST

The BLIST verb has been modified to support the extensions to DATA/BASIC syntax described previously and to inhibit indenting of END, STOP and RETURN statements.

BLOCK-PRINT

BLOCK-PRINT has a new M option that uses the MINI-CONVERT data section of the BLOCK-CONVERT file to print smaller characters.

To output the smaller characters, first create the following Q-pointer and then execute BLOCK-PRINT with the M option:

```
MINI-CONVERT
001   Q
002   SYSFILES
003   BLOCK-CONVERT , MINI-CONVERT
```

BLOCK-TERM

BLOCK-TERM has a new M option that uses the MINI-CONVERT data section of the BLOCK-CONVERT file to output smaller characters. Before you use the M option, create the Q-pointer described under BLOCK-PRINT above.

BVERIFY

The BVERIFY verb now supports both the existing format and the standard TCLII format: if the first parameter is a valid file specification, then the TCLII format is assumed. Its format is one of the following:

BVERIFY *program.name* {*account.name*} {(options)}

BVERIFY *file.specification* *item.list* {(options)}

Where:

item.list One or more item-ids separated by spaces, '*' for the whole file. Can be omitted if supplied by an immediately-preceding select-type command.

file.specification and *item-list* may refer to:

1. Catalog entries in an MD
2. Binary executable items in POINTER-FILE
3. \$\$ items in the specified file.

In the first case, the process identifies the catalog entry type, finds the file in which the desired object is loaded and then generates the name of the loaded object and reads it. In the second case, the process reads the named item from POINTER-FILE, and in the third, from the specified file. In each case the item must be named as it is stored in the named file.

Display serious errors only: suppress 'successful verification' messages and errors caused by specifying items that are not MD catalog items or loaded object code.

F Display an error message if unable to locate the object (\$) item

N Suppress the EOP wait

P Output to the printer

Q Quick verify - verify the checksum only (generally sufficient).

BVERIFY retrieves the binary image specified either on the command line or in the MD item. If the Q option is not specified, then the processor attempts to locate and open the file in which the object (\$) item resides, and reads that item. If found, then a byte-for-byte comparison is made between a load module image of the \$-item and the loaded image. If the item cannot be found, or if the Q was specified, then a checksum is calculated for the loaded image and compared to the one stored with the loaded item when it was loaded.

CATALOG

The CATALOG verb now takes option **R** to redirect the executable code item to the file from which it is cataloged. The new item in that file has item-id **££source-item-id** (or **\$\$source-item-id**).

Verb items created in the MD by CATALOG now include additional attributes as follows:

005	Null
006	File in which executable code is contained (POINTER-FILE or file from which cataloged)
007	Item-id of executable code item (££source-item-id or \$\$source-item-id) or, if in POINTER-FILE, in format <i>account*C*item</i> , where <i>account</i> is the account from which the program was cataloged and <i>item</i> is the item-id of the source item)

This has the effect of allowing the executable code to be in any file and with any name.

Notes:

1. The DELETE-CATALOG verb does not delete the ££ item: it just deletes the MD pointer and any POINTER-FILE item referenced by the pointer.
2. The system still processes correctly earlier cataloged pointers with:

005 account.name item.name

After a program is cataloged, the system no longer displays the size of the object code in frames.

CREATE-ACCOUNT

When creating a remote account, 'option 2. REMOTE SYSTEM NAME' will not exit simply by entering a **RETURN**. You must enter **END** to exit the process without creating a remote account.

- CREATE-FILE** To use the I option, you must have SYS2 privileges.
- The name of the file you are creating can have up to 98 characters.
- DATA.
PRODUCTS** The C option has been added to set up a Data Products printer in Centronics mode.
- DB** The following options are now supported:
- B** Updates source with logical indents (via BLIST options **U**, **S**)
 - D** Runs program via DEBUG command.
 - S** Enters Screen Editor to edit program.
 - T *command*** Executes the specified TCL command then returns to @ prompt.
- DB inserts version number comment line as the first line of new items, and updates existing items that have that comment with a new version number. It deletes the comment line if it is not required (for example, if the program is to be called as an external subroutine and SUBROUTINE must be the first statement). A description of this feature was omitted from 3.1 documentation.
- DELETE-ACCOUNT** You cannot delete an account to which a user is logged on or to which an active or passive (deferred close) file reference exists.
- EVFU-SETUP** The C option has been added to set up EVFU codes in Centronics mode.
- Changing FILE-SAVE PROC to Save Paper** The FILE-SAVE PROC in its standard form lists the file STAT-FILE, one line per account. This is generated by making a call to the LIST-\$STAT-FILE*ACCOUNT-SUMMARY. This uses the N.ACCOUNT-SUMMARY macro which generates the one line per account output.
- An alternative call to the LIST-\$STAT-FILE PROC, normally commented out in the FILE-SAVE PROC, can be made. This lists the file STAT-FILE with one line for each file on the database. You can create a PROC to generate this alternative listing by copying the standard FILE-SAVE PROC to your own PROC library and editing the copy.

The following example shows the statements in the FILE-SAVE PROC involved, although the line numbers may change depending on the release.

```

058   ONOW PRODUCING FILE-STATISTICS REPORT
059 *   [SYSPROG-PL LIST-$STAT-FILE* ]
060 *   FOR A LISTING OF ALL THE FILES, BY ACCOUNT
061 *
062   [SYSPROG-PL LIST-$STAT-FILE*ACCOUNT-SUMMARY]
063 *   IF YOU WANT TO SAVE PAPER BY GETTING ONE LINE PER ACCOUNT
064 *

```

To print a listing of all files on the database, remove the asterisk from line 59 and insert an asterisk in line 62.

FIND

If the system finds an item, you are given the option to edit that item. A sample dialogue is shown below:

```

:FIND
Item-id of lost item:  TEST
Selecting all file definitions in the master dictionary

7 Items selected
Checking in DICT BP
Checking in BP
There is a TEST on BP
Carry on/Edit?  (Y/N/E)  E
Top
.

```

LISTFILES

The LISTFILES verb lists index sections immediately after the data sections to which they apply, indented by one further space.

LOGTO

LOGTO has a new C option. If you use the C option and log to an account, the system will bypass that account's LOGON PROC if the account's D-pointer does not have an R in attribute 9. This option can be used when LOGTO is executed from TCL or PROC.

MOVE-FILE

Now also used to rename accounts using syntax:

MOVE-FILE SYSTEM *old.account.name*
TO : **SYSTEM** *new.account.name*

M-A-S

The M-A-S command supports three different formats which are prompted for at the start of the procedure as follows:

Please select a format:

- A) Single File
- B) Multifile
- C) Oldsave
- H) Help

Format?

Enter 'H' to display a description of Options A, B and C.

MSG

Variants of MSG are supplied in NEWAC and standard system accounts, but are not automatically distributed to user accounts. You can select the version you prefer by copying it to your MD. Syntax is the same as for MSG. The variants are, as follows:

Note: BELL indicates that a bell sounds when the message is received.

MSG0 displays the message on status line 25 in the format:

BELL *user: message*

MSG2 displays the message at the recipients cursor position in the format:

BELL *system-time system-date* **FROM** *port-number user-id*
message

MSG3 displays the message at the recipients cursor position in the format:

message

MSG4 allows you to save a list of recipient's port numbers. When you enter more than one addressee the software displays the numbers of all on-line ports and prompts:

Save port list (Y/N)

If you enter **Y**, the software prompts you with:

File and Item ID:

Enter the name of the file and item-id in which you wish to save the list of recipient port numbers.

MSG4 displays the message on status line 25 in the format:

BELL FROM *port-number*: *message*

MSG6 displays the message at the recipients cursor position in the format:

BELL *message*

NEW-SORT-LIST The NEW-SORT-LIST verb sorts values in one or more items previously saved by NEW-SAVE-LIST. It now has options and allows an item-list:

NEW-SORT-LIST *file.specification* {*item-list*} {*options*}

CAUTION

NEW-SORT-LIST sorts the attribute values within the listed items and does not sort the item-ids

- item-list* can be one or more list item-ids or an * to specify all items in *filename*.
- n* See option M.
- C** Counts number of identical attribute values, appending to the value the count, if greater than one, preceded by a subvalue mark. Only active with option U.
- M** Discards unique entries in list. If *n* not specified, keeps one copy of entries contained more than once. If *n* specified, keeps one copy of entries contained *n* or more times. Gives the intersection of merged lists.
- N** Inverts the M option so as to keep only those values that occur once in a list. Relevant only with the M option. If the list was created by merging two or more lists, this option gives the symmetric difference.

R Sorts the list in a right-justified ascending order.

U Removes multiple cases of a value. If the list was created by merging two or more lists, this option gives the union of lists.

If no options are specified, the list is sorted in a left-justified ascending order.

PORT-DESPOOL The PORT-DESPOOL command now takes a numeric option:

PORT-DESPOOL $\{(n)\}$

The command assigns translation table n to the despooler while it remains active.

PORTOUT This is a synonym for PORT-DESPOOL and takes the same parameters and options.

PRINTRONIX The C option has been added to set up a Printronix printer in Centronics mode.

REFORMAT This command will not output the generated list to TAPE as stated in the Release 3.1 *ENGLISH Reference Manual*.

RENAME-FILE Not now used to rename accounts. Refer to the topic *MOVE-FILE*.

RUN D and G options are no longer available. To invoke the DATA/BASIC debugger, use the new DEBUG command.

RUNOFF The 'hilite' feature prints a character in the margin against selected lines of text.

The following command turns on the hilite capability, using the current hilite character or the default hilite character, which is '|':

.HILITE ON

The following command turns on the hilite capability and assigns hilite character c , where c is a character greater than space, less than a system delimiter, and not 'o' or 'O':

.HILITE c

The following command turns the hilite feature off:

.HILITE OFF

The command TP is now a synonym for TEST PAGE.

SAVE The SAVE verb now writes the current file number, amount of data saved, reel number, and current account to the status line on terminals which support status line addressing.

This information is not output if option **I** is used.

SAVE-LIST The SAVE-LIST command is used after a GET-LIST as follows:

GET-LIST *list.name*

>**SAVE-LIST** *new.list.name*

saves the copy of *list.name* as the new list *new.list.name*.

'FRAMES USED' are not reported in 4.0.

SEL-RESTORE SEL-RESTORE now takes option **S**, which suppresses 'exists on file' messages (as for COPY).

SORT-LIST The SORT-LIST verb sorts a list previously saved in file POINTER-FILE. It now has options:

SORT-LIST {*list-name* {*account-name*}} {(options)}

n See option **M**.

C Counts number of identical attribute values, appending to the value the count, if greater than one, preceded by a subvalue mark. Only active with option **U**.

M Discards unique entries in list. If *n* not specified, keeps one copy of entries contained more than once. If *n* specified, keeps one copy of entries contained *n* or more times. Gives the intersection of merged lists.

N Inverts the **M** option so as to keep only those values that occur once in a list. Relevant only with the **M** option. If the list was created by merging two or more lists, this option gives the symmetric difference.

R Sorts the list in a right-justified ascending order.

U Removes multiple cases of a value. If the list was created by merging two or more lists, this option gives the union of lists.

SP-ALIGN	Allows you to specify either a form queue or a despooler.
SP-ASSIGN	<p>The syntax has change and the following new option is supported:</p> <p>B Banner override. Prevents a report banner from being printed even if the formqueue has banner printing selected.</p> <p>The options C, P and U are no longer supported.</p>
SP-CLEAR	Only clears print jobs in a particular job file. The syntax is changed to allow a job file to be specified using the F option. A U option is also supported which enables jobs to be cleared irrespective of job file.
SP-CLOSE	Accepts report numbers 0 through 127.
SP-COPIES	Can be executed for all 'Queued', 'Hold' and 'Finished' print jobs in a SYS type formqueue. The restriction in Release 3.1 is because system jobs are passed to the UNIX Spooler. This no longer applies Release 4.0.
SP-CREATE	Has a changed syntax.
SP-DELETE	Will not delete a job that is being printed.
SP-DEVICE	Has a changed syntax.
SP-EDIT	The job to be edited must be in a particular job file. The syntax is changed to allow a job file to be specified using the F option. Another new option supported is the I option which enables a print job to be copied to a regular file item.
SP-FQDELETE	Can delete the default form queue.
SP-JOBS	<p>Provides a modified display and menu. Changes in the menu include:</p> <ul style="list-style-type: none">• Action Codes 1 and 2 are changed to 'Switch all Jobs' and 'Switch One Job', respectively• Action Code 15 is now 'FQ/System/Jobs'• Three completely new actions are specified. <p>Action Code 16 'Detail Display' Action Code 17 'Reset Selection' Action Code 18 'Change Job File'</p>

SP-KILL	Allows you to kill a print job by specifying its form queue, job name, or despooler.
SP-LOOK	Displays job name and job file, but not open job numbers.
SP-MOVEQ	Will not move a job that is being printed.
SP-OPTS	B option suppresses printing of banner.
SP-RESUME	Resumes a suspended form queue, job, or despooler.
SP-STATUS	Has a slightly different display.
SP-STOP	Has a changed syntax which allows printing to be stopped by specifying the job id, queue name or despooler name.
SP-SUSPEND	Suspends a form queue, job, or despooler.
SP-TRANSLATE	Has a changed syntax. The <i>despooler-name</i> must be specified instead of the <i>form-queue</i> . Also it can be used to change the table number for a system despooler.
SP-SWITCH	Will not move a job that is being printed.
SSM	RealityX 4.0 SSM is due for substantial re-development which will be documented in a later version of the manual. However, the following differences based on REALITY Release 7.2 should be noted.
SSM Option 1	If you enter a '?' at the File Key > prompt on the NETWORK FILE MAINTENANCE screen, it creates a new item-id. It does not list the NETWORK file as on Release 3.1
SSM Option 2	The USERS FILE MAINTENANCE screen contains three new options, 20, 21 and 22, as follows: 20 Access Locks Allows you to specify keys that permit the user to access accounts and files that are protected with retrieval locks. If these keys are not specified, the system continues to use an account's keys to determine file access. The prompt is File Access Keys (separated by ','); A=Add; D=Delete)

Enter a key or multiple keys separated by commas. To add a key, enter A and enter the key in response to the prompt. To delete a key, enter D and enter the key in response to the prompt.

21 Update Locks

Allows you to specify keys that permit the user to update accounts and files that are protected with locks. If these keys are not specified, the system continues to use an account's keys to determine file update. The prompt is:

```
File Update Keys (separated by ','); A=Add;
D=Delete)
```

Enter a key or multiple keys separated by commas. To add a key, enter A and enter the key in response to the prompt. To delete a key, enter D and enter the key in response to the prompt.

22 Verb file path

Allows you to specify a file to contain additional verb definitions. After looking in the account's MD, the TCL processor looks in the file specified here for a verb definition entered at TCL

The syntax is:

```
{/account{/}{filename}}
```

If the file name is omitted, the default is the current account's MD.

SSM Option 3

The SECURITY FILE MAINTENANCE screen has a change to option 18 and a new option 19. as follows:

18 RPL Debugger

Allows you to specify whether the user is allowed entry into the RPL Debugger. If you enter Y, the command prompts:

```
Enter RPL debug priv. level (0,1 or 2)
```

Enter the level number, 0, 1, or 2, where 0 is the lowest, most restrictive level.

**START-
PRINTER**

A synonym of START-DESPOOLER. Has the same syntax as START-DESPOOLER where the despooler name is specified instead of formqueue name.

- STOP-PRINTER** A synonym of STOP-DESPOOLER. Has the same syntax as STOP-DESPOOLER where the despooler name is specified instead of formqueue name.
- STAT** The TOTAL report uses a comma to separate thousands.
- T-ATT** Multiple tape units can be attached to the default channel. The syntax of T-ATT contains the parameter unit-list which is the number of tape(s) to be used, separated by commas.
- T-DET** Now has a U option which detaches a tape unit from any line (except the Spooler's line) if its not in use and logs off the user. SYS privileges are required.
- T-DUMP
/T-LOAD** Now handles binary data. T-LOAD takes the S option to suppress 'exists on file' messages.
- T-STATUS** The T-STATUS verb now takes options **Dn** and **En**. T-STATUS with option **Dn** causes deletion of the specified unit from the normal T-STATUS display. With option **En**, the specified unit is re-enabled within the display.
- USER** Now displays the actual size of the :
- Input String (HS) workspace
 - Output String (OS) workspace
 - History String (HS) workspace
 - Temporary String (TS) workspace
 - BASIC workspace
- The IS, OS and HS workspace form the major part of the scratch space used by a process. Their purpose is as follows:
- The IS workpaces is used to process input strings, particularly from the EDITOR and ENGLISH processors.
 - The OS workpaces is used to process output strings, particularly to the EDITOR and ENGLISH processors.
 - The HS workspace is used to pass messages to the Wrapup Processor at the conclusion of a TCL statement.

HS, IS and OS workspaces are each initially 6K (6144 Bytes). They are extended as necessary up to a system limit. By default the system limit is 16M.

The TS workspace is used by system user exits for input and output conversion.

The BASIC workspace is used by DATA/BASIC.

Chapter 7

ENGLISH and List Processing Commands

This chapter describes changes relevant to ENGLISH, including:

- Masks used in value limiters
- Data transformation.
- The Tfile code - translation
- LIST processing verbs
- List and Sort verbs with spreadsheet output
- Sort and grouping connectives
- The EDELETE verb.
- Mask Character operations.
- R option for ENGLISH verbs,
- T-DUMP and T-LOAD verbs.
- System and ENGLISH messages

This chapter also describes new and modified list processing commands.

Masks Used in Value Limiters

The use of the right ignore bracket ']' in masks has changed in Release 4.0.

In Release 3.1, you can only use a righthand bracket (]) at the end of a mask as a right ignore character.

In Release 4.0 you can use a righthand bracket (]) to specify an indefinite number of wild card characters between two strings in the middle of a mask, as well as being used as a right ignore character at the end.

When inserted in the middle of a mask between two or more strings, it specifies the selection of two or more strings with any number of intervening characters. This allows selection based on *string1* followed by *string2*, and so on.

A general form of the new syntax is shown below.

Mask Syntax

```
"[{string1[[ ^...]] ]{string2 }...{ ]}"
```

string1 is the first string **searched for**.

] between *string1* and *string2* indicates that any number of characters can come between *string1* and *string2*.

Use the] as the last character in the mask to ignore any characters, or none, that follow.

string2 is the next string searched for and must be preceded with a right ignore character.

^ is a wild card character. '^' specifies one character that must be present but can have any value. (Example: "Joh^ns^n" would select "Johanson"). One or more '^' can be used, instead of ']' if you want to specify a set number of characters.

Example

```
LIST ROOMS WITH GUEST = "GA]AG]ER]"
```

Finds the guest named GALLAGHER.

Date Transformation

Displaying Date Conversions

Date conversions that display alphabetic characters now (in Release 4.0) show an initial capital followed by lower case letters. See examples below:

This is different from Release 3.1 where date conversions display dates in uppercase only.

<u>Code</u>	<u>Internal Value</u>	<u>Result</u>
D	2705	28 May 1975
D0	2705	28 May
DMA	2705	May
DWA	2705	Wednesday

However the earlier Release 3.1 uppercase format can be restored for the duration of a particular logon session by executing the SET-OPTION command with the UCASEDATES option. See Chapter 6 for details.

New D Codes Available

DAW Returns the administrative week number, which ends on Sunday. For example

<u>Date</u>	<u>Value Returned</u>
31 March 1992	14
31 December 1995	52

DAM Returns the administrative month number, which is based on the first two months of a quarter having four weeks and the third month having five weeks. For example:

<u>Date</u>	<u>Value Returned</u>
31 March 1992	4
31 December 1995	12

DAY Returns the administrative year, which is based on each week number starting on Sunday. For example:

<u>Date</u>	<u>Value Returned</u>
31 March 1992	1992
31 December 1995	1995

Tfile Code - Translation

The syntax of Tfile code has changed, as follows:

Syntax **T**[* | **DICT**] *file-specifier*; *c*{*n*}{:*w*}{;*i-amc*}{;*o-amc*}{;*b-amc*}

Syntax Elements Two syntax elements are different, *n* and *w*.

n can be:

a value mark count: returns just that value from the attribute referenced. If omitted, all values from the attribute are referenced.

an asterisk: uses the same value mark count in the target file as the primary file.

9999: returns the size of the target item.

w is the character that will delimit multivalued and subvalues.

Comments If *n* is not used, all values (and subvalues) are returned with a blank rather than a value mark (or subvalue mark).

New and Modified ENGLISH and List Processing Verbs

Release 4.0 supports two new ENGLISH verbs to generate listings for a spreadsheet environment, a number of new list processing verbs and some changes to existing list processing verbs. Refer to Chapter 6 for details.

Lists can also be located in normal files as well as in POINTER-FILE.

New ENGLISH Verbs

The following two ENGLISH verbs are provided to enable the interpretation of an ENGLISH listing by spreadsheet software, such as that provided by UNIPLEX.

LIST-SPREAD Generates an ENGLISH listing in the same way as the LIST verb. However, instead of putting spaces between columns, it inserts a tab character and it does not supply a paged output. The syntax is the same as for the LIST verb.

SORT-SPREAD Generates a sorted ENGLISH listing in the same way as the SORT verb. However, instead of putting spaces between columns, it inserts a tab character and it does not supply a paged output. The syntax is the same as for the SORT verb.

The output generated by LIST-SPREAD and SORT-SPREAD is intended for output to a spreadsheet environment. It is not meant for output to the screen in the RealityX environment.

New List Processing Verbs

Three new verbs provide the capability to perform logical operations on lists:

AND-LISTS Returns the intersection of two or more lists.

OR-LISTS Returns the union of two or more lists.

XOR-LISTS Returns the logical difference of two or more lists.

Three new verbs provide the capability to perform logical operations on lists residing in items:

AND-ITEMS Returns the intersection of two or more lists.

OR-ITEMS Returns the union of two or more lists.

XOR-ITEMS Returns the logical difference of two or more lists.

Two verbs compare an active list against the item-ids in a file:

NSELECT Returns the item-ids that are in the list but not in the file.

XSELECT Returns the item-ids that are in the file but not in the list.

Modified List Processing Verbs

The following verbs have been modified:

SAVE-LIST

NEW-SORT-LIST

SORT-LIST

Refer to Chapter 6.

Location of Lists

Lists are saved either in POINTER-FILE using the SAVE-LIST verb, with item-id format *account.name*L*list.name*, or in normal items in any specified file using NEW-SAVE-LIST.

Retrieval of lists is via GET-LIST or NEW-GET-LIST respectively. EDIT-LIST works only on POINTER-FILE lists.

List Headings

ENGLISH lists which used to leave just a heading on the last page no longer do so.

Connectives

Release 4.0 provides the following new connectives:

SORT Connectives

BY-EXP-SUB which specifies ascending sort by subvalue for following data definition item.

BY-EXP-SUB-DSD
 which specifies descending sort by value for following data definition item.

Grouping Connectives

MIN which specifies that the minimum value for the data definition item following is to be returned at control break time.

MAX which specifies that the maximum value for the data definition item following is to be returned at control break time.

Mask Character Operations

MC (mask character) codes now retain delimiters within the string being converted, except for MCP and its variants which convert delimiters as defined in existing documentation.

MCA now extracts characters in standard, primary added and secondary added alphabets (if any).

MC0 extracts characters in standard alphabet only.

MC1 extracts characters in primary added alphabet only.

MC2 extracts characters in secondary added alphabet only.

Refer to the description of LOAD-ALPHA in this chapter more details on alternative alphabets.

Other Modified Verbs

EDELETE Verb Can be used to delete a D-pointer. However D-pointers should not normally be deleted. Use DELETE-FILE.

R Option for ENGLISH ENGLISH verbs now take the 'R' option to suppress 'Item not on file' messages.

**T-DUMP/
T-LOAD Verbs** Now handles binary data. T-LOAD takes the S option to suppress 'exists on file' messages.

Messages

Message Formats ENGLISH message format has changed in 4.0 to lower case letters with initial capitals. Hyphens have been removed and message item numbers shown in front of some messages.

ENGLISH Reports

The following minor differences are observed in 4.0 ENGLISH reports.

- The message at the bottom of a report, for example, '24 ITEMS LISTED' (Release 3.1), is in lower case with initial capital in 4.0 with a space at the beginning, for example ' 24 Items listed'

SORTS and LISTS greater than 999 now have a comma as a thousands separator in the ' Items listed' message.

Chapter 8

PROC

This chapter describes changes relevant to PQ and PQN PROC and describes the PQN PROC SYSTEM() function.

PQ and PQN PROC Changes

File Buffers

There are now 20 file buffers (1-20) rather than 9.

File buffers 19 and 20 are used for System PROCs; therefore, it is recommended that you restrict your use to buffers 1-18.

Registers

There are now 10 select registers rather than 5.

[] Command

You can specify a label in an external subroutine where you want execution to begin.

The syntax is:

[file-spec {item-id}] {label}

label is a specific label where you want execution to begin.

If *label* is not specified, execution begins at the first line of the PROC. Note that a space must precede the label.

FB Command

The FB command stores the item in a scratch buffer rather than OS space. To reference this scratch buffer, use:

&0.a

where *a* is the attribute number. Note that this scratch buffer can be used without the FB command. For backwards compatibility, the older syntax, **&.a**, will continue to be supported.

GOSUB Command

To transfer control within a PROC to a subroutine with a specified label, you can enter:

GOSUB *label*

or

[] *label*

Note: A space must precede the label.

IF Command

The following IF statement syntax will check for the existence or non-existence of an active SELECT list:

IF {#} *S command*

IF # S *command* tests for the absence of a SELECT list.

IF S *command* tests for the presence of a SELECT list.

where *command* is a valid PROC command.

MV Command

The syntax of the MV statement is:

MV *destination source*{,*source ...*}{**source...*}{,*{*n*}}{,[_]}

destination is a direct or indirect reference to a buffer or select register where you want the data copied to. If the destination is a select register, certain restrictions on syntax apply.

source is the data you want to copy. It can be:

- a direct or indirect reference to a buffer or select register.
- a direct or indirect reference to a buffer followed by ";InputConversion;" or ":OutputConversion:". See the topic "ENGLISH Conversions in MV syntax" for details.
- a string of zero or more characters enclosed in double quotes, or single quotes if *destination* is not a select register. Entering an uneven number of quotes is no longer allowed; a syntax error will result.
- a single character expressed in one of two ways:

Xx where *x* is a hexadecimal number in the range 00 - FF. Thus, FD is a value mark.

In where *n* is a decimal number in the range 0 to 255. Thus, I253 is a value mark.

This feature is not available if *destination* is a select register.

,* copies all source parameters starting with the specified parameter. The destination buffer is truncated after the last parameter is copied unless the ',*' clause is followed by a ',' or '*' and other source parameters.

,* <i>n</i>	copies <i>n</i> source parameters following the specified parameter; in other words, <i>n</i> +1 parameters are copied. The destination buffer is overlaid element by element until the source parameters run out or the count <i>n</i> runs out. The destination buffer is not truncated.
* <i>source</i>	concatenates the source values into the specified destination buffer or buffer element when an asterisk is used in place of a comma in separating source parameters.
,_	should be the last source parameter in the source list, and should only be used when it is desired that the destination buffer be truncated following the last parameter copied. Note that the ',_ ' is not required when the last operand in the source field is ',' as this automatically truncates the destination buffer upon completion.

ENGLISH Conversions in MV syntax

The MV syntax now supports ENGLISH conversion of parameters to destinations other than select registers for direct or indirect references in the source operand fields. The format is similar to the use of conversions with IBH:

```
reference;InputConversion;  
reference:OutputConversion:
```

For example

```
MV %3 %3;T1,3;
```

extracts the first three characters of the contents of %3 and replaces the original %3. An advantage of using the MV command for such conversions is that the buffer size limitations placed on these conversions with the IH or IBH commands are not present here, as MV uses an extensible buffer. The conversion buffer is limited to the size of your OS workspace area.

Note: Not all conversions may be used within PROC. (Some ENGLISH conversions have such wide element usage that they may effect the way PROC runs.) Conversions which cause a problem when running PROC should not be used.

Information on ENGLISH conversions can be found in the RealityX 3.1 *ENGLISH Reference Manual*.

T Command

The *text* element can be enclosed in single or double quotes.

New PQN PROC Function

SYSTEM() Function

PQN PROC now supports most of the DATA/BASIC SYSTEM() functions.

The syntax is:

```
@SYS{TEM}(n)
@SYS{TEM}(reference)
@SYS{TEM}(indirect reference)
```

n is a numeric (from 0 to 60) that specifies a particular function. The numbers you can use are listed in Appendix A.

The SYSTEM() function can be used with the following PROC commands:

MV	(using <i>destination</i> and <i>source</i> elements only; <i>destination</i> cannot be a select register)
T	
IH	(if not a valid function, element is treated as a string)
IBH	(if not a valid function, element is treated as a string)
H	(if not a valid function, element is treated as a string)
IF	(used as a conditional test)

A few functions (such as 15 and 22) return multiple fields separated by attribute marks. Use the MV command to access these functions. For example, the following command places the alphabetic option in %2, the first numeric option in %3, and the second numeric option in %4:

```
MV %2 @SYSTEM(15) , *2
```

The following example returns the system name:

```
MV %3 52"
T "SYSTEM NAME: " , @SYSTEM(%3)
```

The following example returns 'PORT*portnumber':

```
MV %2 "PORT*" * @SYSTEM(18)
```

Chapter 9

Networking

This chapter summarises changes relevant to RealityX release 4.0 networking. For details on RealityX Release 3.1 networking, refer to the *RealityX Reference Manual Volume 3: Administration*.

NETDEVS File

RealityX release 4.0 uses one additional system file called NETDEVS, located in SYSFILES. This contains device definition items for devices such as local or remote printers. The format of a device definition item in NETDEVS is:

The NETDEVS item must be of the form:

Item-id	Device name
1	ROUTE-FILE entry
2	NPU
3	NETDSP

This enables the **npu** which enables Printer Sharing to drive the network device.

For backward compatibility network printing can be configured without the **npu**, using a NETDEVS entry of the form:

Item-id	Device name
1	ROUTE-FILE entry
2	null
3	NETDSP

However, it is recommended that you configure the network printer with the **npu** (that is enter NPU in attribute 2 of NETDEVS item), as this enables printer sharing. The **npu** supports both shared and non-shared printing.

The NETDEVS item structure is as follows:

Item-id	is the device name specified in the DESPOOLER.CONTROL record for the PTR network despooler. See Chapter 6.
Attribute 1	contains the ROUTE-FILE entry for the remote system/printer to which the printer is attached.
Attribute 2	must contain 'NPU' to execute the Network Printing Utility and null if the npu is not to be used.
Attribute 3	contains NETDSP to specify a network despooler.

Chapter 10

DATA/BASIC

This chapter describes the changes to DATA/BASIC. These include new and modified statements, new intrinsic functions, and an enhanced debugger.

New Statements

POSITION Statement

Positions the read pointer of an index.

POSITION *index.var* = [*location-value* | **END**] {**SETTING** *setting.var*} [**THEN** *statements* | **ELSE** *statements*]

This positions the pointer to the first index entry equal to location value, or to the first value greater if there is no match, on a left-to-right matching basis. The END form sets the pointer to the end of the index.

index-var is a value to compare against the index's sort criteria.

END sets the pointer to the end of the index.

setting-var is the name of the variable to which the error code or zero is assigned when the else clause is taken.

statement(s) is either a **THEN** or an **ELSE** clause (or both). A statement must be included.

For example:

```
POSITION ACCPAYINDEX = MONTHCURR
```

sets the pointer to the index in variable ACCPAYINDEX to the entry that is equal to or greater than the value in MONTHCURR.

READPREV Statement

Reads the previous sequential item-id from an index. The syntax, which is similar to that of the READNEXT statement is:

READPREV *variable* {,*vmc-var*} **FROM** *index-var*]{**RETURNING** *key-var*} {**SETTING** *setting-var*} [**THEN** *statement(s)* | **ELSE** *statement(s)*]

READNEXT is described later in this chapter. For a detailed description of READPREV, refer to the *DATA/BASIC Reference Manual*.

**VARVALSET
Statement**

Changes the value stored by a variable. The syntax is:

VARVALSET *variable.name* **TO** *expression* [**THEN** *statements* / **ELSE** *statements*]

where:

variable.name is the compile time name of the variable supplied as a character string at run time.

expression is any value.

Note: The variable will not be updated if it does not exist, or it is not otherwise legal to update.

See also the description of the VARTYPE, VARVAL and VARVALTYPE functions.

Modified Statements

CONNECT Statement

The remote-system syntax is:

```
{system}^account{,account-passwd}^server{,server-passwd}{^Q}
```

server-passwd is required if the server requires a password. The *account-passwd* is required if the account requires a password, unless the account is the server's default account, which means that *account-passwd* is not required.

system is an item in **/etc/ROUTE-FILE**.

FOR Statement

```
FOR variable = value1 TO value2 {STEP value3}  
{[WHILE|UNTIL] expression {DO}} ...  
{statement{;statement} ...  
{[WHILE|UNTIL] expression} {DO}} ...  
{statement{;statement} ...  
NEXT variable
```

You can now use one or more WHILE and UNTIL clauses inside a FOR loop. The DO is optional. For example:

```
FOR X = 1 TO TBLSIZE  
    WHILE TBL1(X) NE ""  
        TBL2(X) = TBL1(X)  
    UNTIL TBL1(X) = 0  
NEXT X
```

This program copies TBL1 to TBL2 until a null or zero element is found.

INCLUDE Statement

INCLUDE statements can be nested to a maximum of 16 levels. Each level can contain any number of include statements.

LOOP Statement The LOOP statement now supports the following syntax:

```

LOOP { VARYING counter.var = start.expr
        { STEP step.expr } }
    { statement { ; statement } ...
    { [WHILE|UNTIL] limit-exp { DO } }
    { statement { ; statement } ...
    { [WHILE|UNTIL] limit-exp { DO } }
    .
    .
    .
REPEAT

```

The new VARYING clause provides a running counter without specification of a termination value.

You can have multiple WHILE and UNTIL clauses within the loop, and DO is optional.

PAGE Statement **PAGE** { *expression* }

expression is a valid DATA/BASIC expression used to reset the page number in a heading or footing.

For example:

```

A = 10
HEADING "MONTHLY REPORT 'L' PAGE 'PP' "
.
.
.
PAGE A

```

The current output device advances to top of form and prints the heading:

```

MONTHLY
REPORT
PAGE 10

```

PERFORM Statement

The PERFORM options listed below can be used in any order:

PASSLIST	SETTING
RTNLIST	PASSDATA
CAPTURING	RTNDATA

**PERFORM
SELECTs**

A PERFORM statement without a RTNLIST clause that executes a SELECT or other list-generating command creates a 'pending' item list consisting of the selected items.

The pending list is supplied to the next TCL command in a PERFORM statement without a PASSLIST clause.

Any PERFORM with or without a PASSLIST clause empties the 'pending' list. However, a PERFORM that executes a SELECT and lacks a RTNLIST clause creates a new 'pending' list as described above.

**PRECISION
Statement**

A program can have multiple PRECISION statements. A PRECISION statement remains in force until another PRECISION statement is encountered or the program ends.

The maximum precision is 99; if no PRECISION statement is used, precision defaults to 4.

Precision is used to govern the number of decimal places generated by operations that could generate more decimal places than either of the operands have. If a literal is specified, its precision will be used until a result is truncated. The operations that will truncate the result, and possibly intermediate values, to precision digits are:

Multiply
Divide
SQRT
Runtime conversion of a string to a number

The following functions currently operate at a maximum precision of 5:

SIN	PWR
COS	EXP
TAN	LN

When adding or subtracting numbers, the resulting number inherits the precision of the operand with the greatest precision.

**READNEXT
Statement**

The READNEXT syntax has been changed to enable the reading of the next item-id from an index. It now includes a variable *index-var* and a RETURNING clause. The complete syntax is as follows:

READNEXT *variable* {*,vmc-var*} {**FROM** [*select-var* | *list-var*/*index-var*]}{**RETURNING** *key-var*}} {**SETTING** *setting-var*} [**THEN** *statement(s)* | **ELSE** *statement(s)*]

index-var is the name of a variable that contains an index. The index is assigned to it by the SELECT statement. If *index-var* is to be used, the FROM clause must be included.

key-var is the name of a variable that receives the key value associated with the item-id in the index variable. The key value comprises the attributes specified as sort criteria when the index was defined.

READNEXT returns the next sequential item-id from the index. The key value which determines the location in the index of the item-id can optionally be returned via the RETURNING clause. This consists of the values, in that particular item, of those attributes named as sort criteria when defining the index. Each value is separated from the next by an attribute mark. Values are returned in the order specified in the index definition.

RECWAIT Statement

The RECWAIT statement now has an optional TIMEOUT by which you can specify a time-out value:

RECWAIT *data* **FROM** *session*{*,reference*} {**USING** *function*{*,qualifier*}} {**TIMEOUT** *minutes*} {**SETTING** *error*} [**THEN** *statements* | **ELSE** *statements*]

minutes is an expression evaluating to a time-out value in minutes. The ELSE clause is executed if data is not received within this time and error 4225 is returned. If omitted, it waits indefinitely.

RQM/SLEEP Statement

Supports the facility to specify a *wake-val* of less than a second.

For example:

```
SLEEP 0.5
```

causes the program to sleep for a minimum of half a second.

Decimal fractions greater than one are rounded down to an integer.

For example:

```
SLEEP 2.75
```

causes the program to sleep for a minimum of two seconds.

SELECT Statement

Indexes are accessed from DATA/BASIC by the SELECT statement, which selects an index for access via POSITION, READNEXT and READPREV statements.: The syntax is as follows:

```
SELECT file-var,index-name TO index-var {SETTING setting-var}
```

where

file-var is the name of the file variable to which an indexed file has been opened

index-name is the name of the index

index-var is the variable that receives the list derived from the index.

This data structure is then stored in *index.var*, which is referenced in the statements that follow. Note that *index-var*, like *select-var* cannot be used in the PASSLIST clause of a perform.

SUBROUTINE Statement

A calling program and corresponding subroutines do not need to have the same precision. In fact, when a value is passed back to the source program, the value retains the precision used in the subroutine.

TRANSTART Statement

```
TRANSTART {transaction-information} {SETTING setting-var} [THEN statement(s) | ELSE statement(s)]
```

setting-var is the name of a variable that is set to a value corresponding to a file I/O error code.

The TRANSTART statement now accepts an optional SETTING clause, which will return the error message should the TRANSTART fail (i.e., should it take the ELSE clause because transactions have been disabled or the process is already inside a transaction).

This will allow the applications programmer to maintain control of the program instead of hanging, waiting for the system manager to allow transactions again.

The SETTING clause is only valid if the ELSE clause is executed.

New Intrinsic Functions

FMT Function

The FMT (format) function enables you to apply a mask character conversion to a variable or input string. The syntax is:

FMT(*x,y*)

x is the variable or input string.

y is the mask character conversion.

For example, in the program

```
A = "towne"  
B = FMT(A, 'MCU')  
PRINT B
```

the B variable would be TOWNE, because MCU changes lowercase characters to uppercase.

FOLD Function

The FOLD function places attribute marks in a string in place of spaces. The spaces are specified as being no more than a specified number of characters apart. The syntax is:

FOLD(*string,fold-width*)

where:

string is any text containing spaces

fold-width is the maximum number of characters in each fold.

This function replicates the T justification code in ENGLISH.

PTR Function

This PTR function generates a printer control string that is not unique to any type of printer. The printer despooler interprets the string according to the printer independence item-id that has been assigned via the SPM (Spooler Maintenance) program on the SYSPROG account. Thus, one program may output bold, underlined text to a print job, which could be printed correctly on any of your printers. The PTR syntax is:

PTR(*function-class,function-subclass*{,*param,param,...*})

function-class is a numeric that specifies a function class defined by the PDM program.

function-subclass

is a numeric that specifies a function subclass defined by the PDM program.

param

can be one or more parameters required by a particular function class and subclass.

For example,

```
PTR ( 4 , 1 , 5 , 20 )
```

invokes the Set Horizontal Tabs function, which is class 4, subclass 1. This example includes two parameters, 5 and 20.

For information on Printer Independent Functions see Appendix B or, for full details, the *Printer Definition Maintenance* manual.

ROUND Function

The ROUND function rounds a numeric to the specified number of decimal places. If the number of decimal places exceeds the current precision, ROUND will affect only numbers which have not already been shortened by the PRECISION statement. The syntax is:

ROUND(x,y)

x is the numeric variable or string.

y is the number of decimal places that will be displayed after the rounding takes place.

For example, in the program

```
A = 0.05  
B = ROUND(A,1)  
PRINT B
```

the B variable would be 0.1.

TRUNC Function The TRUNC function truncates a numeric variable or string to a specified number of decimal places. The syntax is:

x is the numeric variable or string.

y is the number of decimal places that will be retained after truncation.

For example, the program

```
A = 125.3456
B = TRUNC(A,2)
PRINT B
```

prints 125.34.

If the number of decimal places exceed the current precision, TRUNC will affect only those numbers which have not already been shortened as a result of operations determined by the PRECISION statement.

Variable Checking Functions Release 4.0 supports a set of three new functions to inspect the nature and state of a named variable in a program. These are described below.

Note: The VARVALSET statement, described previously in this chapter, is also provided to change the value of the variable.

VARTYPE Function Returns the type of variable, as determined at compilation, as a string. The syntax is:

VARTYPE(*variable.name*)

VARTYPE returns the following string:

type .ordinal {row {column}}

where:

type may be:

- V Simple variable
- D Dimensioned variable, with *row* and *column*
- U Undefined variable

variable.set.ordinal is the group of variables to which the variable belongs, where:

- 0 specifies a local variable.
- 1 specifies global common.
- 2 specifies labelled common block ordinal.

VARVAL Function Returns the current value of the named variable. The syntax is:

VARVAL(*variable.name*)

The value of an unassigned variable is indicated by the literal <unassigned>

The value of a file variable is indicated by *file.variable file name string*.

VARVALTYPE Function Returns the data type of the current value of a variable.

VARVALTYPE(*variable.name*)

VARVALTYPE returns a type character, as follows:

- 00 Cleared, unassigned variable
- 01 Scaled binary number
- 02 Short string
- 04 File variable
- 81 String number
- 82 Indirect string
- U Undefined

Modified Intrinsic Functions

CHAR Function This function now has the syntax:

CHAR(*char-value*{,*size*})

This new form allows the specification of character size (*size*), which can be from one to four bytes. The form CHAR(*char-value*) assumes one byte.

The new syntax can be used with character sets that require more than eight bits to represent a character.

DECRYPT and ENCRYPT Functions

The DECRYPT and ENCRYPT functions have a new method (*method.idx*) for decryption and encryption, which is method 3:

3 A one-for-one exclusive OR between the string in *exp1* and an infinite garbage string generated through a Fibonacci algorithm. This method uses the entire key (*exp2*).

In addition methods 0 and 2 use only the last character of expression *exp2*; method 1 does not use *exp2*, but it must be present in the syntax even as a null string ("").

SEQ Function This function now has the syntax:

SEQ(*expression*{,*size*})

This new form allows the specification of character size (*size*), which can be from one to four bytes. The form SEQ(*expression*) assumes one byte.

The new syntax can be used with character sets that require more than eight bits to represent a character.

SPOOLER Function

When you specify an *sp-function* of 3, the information returned now includes the job file name. The returned string has the following multivalued format:

BASIC print file #]Form Q name]Options]Copies]Job File Name

**SYSTEM(57)
Function**

This new function gives a snapshot of overflow handler activity. Four attributes are returned in a dynamic array. They are:

<u>Attribute</u>	<u>Description</u>
1	Number of calls to obtain space from the overflow table.
2	Number of frames requested from the overflow table.
3	Number of calls to the release handler to put frames back in the table.
4	Number of frames released back into the overflow table.

These attributes are initialized at full restore time and they roll over to zero periodically, in between restores (rollover occurs at X'FFFFFFFF'). This function may be called more than once to see what activity is taking place in terms of calls to get or release overflow space.

**SYSTEM(58)
Function**

This new function returns the Spooler assignment table data for the currently running port. This information is similar to SP-LOOK. There is one entry for each assigned report number. Each entry is separated by an attribute mark (CHAR(254)). The format of each entry is:

<u>Value #</u>	<u>Subvalue#</u>	<u>Description</u>
1		Report number
2		Job file name string
	1	Job file name (D-pointer or Q-pointer)
	2	System name storing job file
	3	Account name storing job file
	4	Job file DICT name (D-pointer)
	5	Job file data section name (D-pointer)
3		Form Queue file name
4		Current job name
5		Assigned option flags
6		Assigned copies

Note that subvalues 1 and 4 will be the same name if subvalue 1 is the name of a D-pointer rather than a Q-pointer.

**SYSTEM(60)
Function**

This function returns the TCL input statement without the verb, redundant spaces, or options and with spaces replaced by attribute marks. Use this function in place of `OCONV(",'U20E0')`.

**SYSTEM(61)
Function**

This function returns the name of the physical account (D-pointer) onto which the process is logged.

New, Modified and Deleted Debugger Commands

Debug Displays When displaying variables, undefined variables are shown as having value "<<unassigned>>" and null variables as having the value "<<null>>".

A Command The A command is a new command synonym for the \$ command.

D Command The D command displays the Break and Trace tables.

L Command The L command now has the following syntax:

L*	Lists all lines of the program
L<i>n</i>	Lists line <i>n</i>
L<i>n,m</i>	Lists <i>m</i> lines starting from line <i>n</i>
L<i>n-m</i>	Lists lines <i>n</i> to <i>m</i>

V Command The V command sets the number of source code lines to be displayed automatically at every entry into the debugger. The syntax is:

V*m*{*n*}, where,

m shows *m* lines from the current line forward.

,*n* shows *n* lines prior to the current line.

To turn off this feature, use V0.

R Command Not supported by RealityX. Documented by mistake for Release 3.1.

Using an Alternative Compiler

In order to create program object items that will run on an earlier system, you need to use a different compiler from that provided by the default version of the BASIC verb. These alternative compilers which are held in the file BASIC-COMPILERS create object items that will run on their target release, and on any equivalent or earlier release. However, they only recognise and compile DATA/BASIC elements that are part of the subset defined for the target release.

Note: On RealityX 3.1 loaded compilers are stored in POINTER-FILE.

On RealityX Release 3.1, alternative compilers are made available by creating a new verb to execute the chosen compiler. This is done via the LOADBNF verb.

On RealityX Release 4.0, the procedure has been changed to make the use of multiple compilers easier. To use compilers other than the default, or to change the default, logon to the SYSPROG account and do the following:

1. To select a compiler, first list the items in BASIC-COMPILERS,OPTOM and in BASIC-COMPILERS (default data section). BASIC-COMPILERS,OPTOM contains object items for the available compilers. Their item-ids indicate the main release for which they are suitable. BASIC-COMPILERS contains loaded compilers together with synonym items (showing Q in attribute 1) whose item-ids indicate other releases for which the associated OPTOM item (shown in attribute 2) is suitable.
2. If the compiler you need is not already loaded (the corresponding OPTOM item is not present in the default section of BASIC-COMPILERS), load it via the LOAD-BNF verb as follows:

```
LOAD-BNF BASIC-COMPILERS,OPTOM compiler
```

```
DESTINATION FILE: BASIC-COMPILERS
```

where *compiler* is the item-id of the OPTOM item. For example, BASIC*SEQUEL*1.2 is suitable for compiling programs to run on systems with REALITY release 1.2/4.3 (in fact, this compiler should also be used to compile for any earlier release).

3. Once the required compiler is loaded, you can run it as follows:
 - For frequent use, create a new verb to run it. To do this, copy the verb BASIC to a new MD item with a suitable name (such as OLD-BASIC). Modify attribute 1 of this item to change the 'tag' (the character B) to some other single character (not G, O, or Q). For example, change attribute 1 from PB to PX. Now create an item in BASIC-COMPILERS with the new tag as its item-id, Q as attribute 1, and the item-id of the loaded compiler item in attribute 2. For example:

```
X
001 Q
002 BASIC*SEQUEL*1.2
```

You can now use the new verb to compile using the selected compiler.

- For infrequent use, use the BASIC verb with option C. See below for a description of syntax and prompts.

Note: If you are using PROCs or programs to drive multiple compilers, you should also use BASIC with option C so that you can specify exactly which compiler will be used.

To change the default compiler for the database or system, edit item BASIC*DEFAULT in BASIC-COMPILERS to replace the contents of attribute 2 with the item-id of the loaded compiler item required. The standard BASIC verb will then call that compiler.

Using the BASIC Verb with the C option

To use an alternative DATA/BASIC compiler, enter the BASIC verb with the C option as shown below:

BASIC *file-name prog-name* (C

file-name is the name of the file that contains the program you want to compile.

prog-name is the name of the program you want to compile.

(C prompts for the compiler name.

At the prompt

Compiler name :

enter the name of the compiler you want to use. The names of the compilers are listed in the BASIC-COMPILERS file. For example, at the Compiler name prompt, enter:

BASIC*SEQUEL*1.2	for SEQUEL 1.2 and REALITY 4.2 systems
BASIC*ROS*7.0	for systems on Release 7.0
BASIC*SPIRIT*2.2	for SPIRIT 2.2 systems

Shared BASIC Object

In RealityX Release 3.1, by default, cataloged DATA/BASIC programs and subroutines are read from POINTER-FILE into private memory. Each RealityX process must read a program or subroutine into its own private workspace in order to execute it. Hence the system memory requirement for multiple users to execute the same code is of the order of the amount of code multiplied by the number of users.

In RealityX Release 4.0, cataloged DATA/BASIC programs and subroutines can be shared. When starting a program or subroutine, RealityX first looks in shared memory to see if it is there already. If not, it reads the cataloged item from POINTER-FILE and copies it into shared memory. This enhances performance by reducing the system memory requirement for multiple users executing the same code.

ShareSize

By default cataloged DATA/BASIC programs are not shared. Sharing is enabled by configuring ShareSize = N, where N is in Kbytes. To calculate the required size, use ENGLISH to find the total size of the cataloged programs for the application. For example, if all programs were cataloged from account APP, use:

```
LIST POINTER-FILE WITH PF-TYPE "C" AND WITH PC-ACCT "APP" TOTAL PF-  
SIZE DET-SUPP
```

ShareModulo

Shared cataloged programs are held in a hashed table, with default modulo 101. To use a different modulo, set ShareModulo - M, where M is the modulo to use. A larger modulo may be required if an application has a large number of programs.

Other DATA/BASIC Changes

- Variables** In Release 4.0 you can have a maximum of 64,000 variables in a program/subroutine, whereas in Release 3.1 the maximum is 3,200.
- Numeric Variables** In Release 4.0 each numeric variable has its own scale associated with it. This allows the precision to be changed dynamically. In Release 3.1, all numeric variables are stored at the same scale.
- Catalogued Programs** After a program is cataloged, the system no longer displays the size of the object code in frames.
- It is strongly recommended that you catalog all DATA/BASIC programs.
- Indexing** See Chapter 10 for a discussion of indexing and how it can be used in the following statements:
- ```
SELECT
READNEXT
READPREV
POSITION
```
- Note also that ENGLISH statements executed in a PERFORM statement will use a suitable index if the criteria specified in the topic *Implicit Use of Indexes* in Chapter 7 are satisfied.
- Date Conversions** In Release 4.0, all date conversions that display alphabetic characters now show an initial capital followed by lower case letters. Examples of 4.0 conversions are shown below:
- In Release 3.1 date conversions display dates in uppercase only.
- | <u>Code</u> | <u>Internal Value</u> | <u>Result</u> |
|-------------|-----------------------|---------------|
| D           | 2705                  | 28 May 1975   |
| D0          | 2705                  | 28 May        |
| DMA         | 2705                  | May           |
| DWA         | 2705                  | Wednesday     |

The following date conversions are now available:

**DAM** Returns the administrative month number, which is based on the first two months of each quarter having four weeks and the third month having five weeks. For example:

| <u>Date</u>      | <u>Value Returned</u> |
|------------------|-----------------------|
| 31 March 1992    | 4                     |
| 31 December 1995 | 12                    |

**DAW** Returns the administrative week number, which ends on Sunday. For example:

| <u>Date</u>      | <u>Value Returned</u> |
|------------------|-----------------------|
| 31 March 1992    | 14                    |
| 31 December 1995 | 53                    |

**DAY** Returns the administrative year, which is based on each week number starting on Sunday. For example:

| <u>Date</u>      | <u>Value Returned</u> |
|------------------|-----------------------|
| 31 March 1992    | 1992                  |
| 31 December 1995 | 1995                  |

## **Mask Character Operations**

MC (mask character) codes, used via DATA/BASIC functions ICONV function and OCONV function (and ENGLISH dictionary attribute definition items), now retain any delimiters within the string being converted, except that MCP and its variants convert delimiters as defined in existing documentation.

## **Upgrade Utilities**

The DATA/BASIC compiler on 4.0 has been changed so that intermediate object code (\$-items) can be more easily transported and used within different host systems and hardware platforms.

Within the RealityX runtime environment, loaded object code (the code after using CATALOG) has also been changed.

Because of these changes, applications running on Release 3.1 have object code that it incompatible with Release 4.0. In general, this incompatibility is handled by the automatic upgrade procedure.

**Reserved Words**      TIMEOUT is now a reserved word.

**NEW TCL  
Commands**              The following two commands are new in Release 4.0:

DEBUG  
LOAD-BNF

For detailed information on these commands, see Chapter 6.

**Modified TCL  
Commands**              The following commands have been modified for Release 4.0:

BASIC  
BLIST  
BVERIFY  
CATALOG  
RUN

For detailed information on these commands, see Chapter 6.

**Obsolete TCL  
Commands**              The following commands have been made obsolescent:

EBASIC  
LOADBNF  
PRINT-HEADER  
STOREPF (Replaced by LOAD-BNF)

# Chapter 11

## Miscellaneous Changes

This chapter describes several miscellaneous changes as follows:

- How to check file SYSTEM-LOG for despooler errors.
- File specifications.
- Renaming accounts.
- Options.
- STAT-FILE Management.
- Statistics for indexes.
- Macros in DICT STAT-FILE.
- On-line documentation
- Changes to the format of system messages.
- The minimum and maximum size numbers you can use on your system.
- Additional commands in System Debugger

## Checking for Despooler Errors

**Overview** If errors occur during the execution of a despooler, the Spooler records these errors in the SYSTEM-LOG file.

To display information about despooler errors, you can display an ENGLISH list of the SYSTEM-LOG file using the DSP.ERRORS or DSP.DETAIL macros.

**Syntax** **LIST SYSTEM-LOG DSP.ERRORS {DSP.DETAIL} {(options)}**

**Command Class** ENGLISH

**Syntax Elements** DSP.ERRORS is a macro that generates a general report of despooler errors  
 DSP.DETAIL is a macro that generates a detailed report of despooler errors.

**Options**  
**N** No page. Suppress automatic paging.  
**P** Printer. Send output to printer assigned to your port.

| <b>General Report Headings</b> | Heading   | Description                                            |
|--------------------------------|-----------|--------------------------------------------------------|
|                                | DATE      | Date the error was recorded.                           |
|                                | Time      | Time the error was recorded.                           |
|                                | Err       | Error number.                                          |
|                                | Dsplr iid | Despooler item-id.                                     |
|                                | Port#     | Number of port that requested output.                  |
|                                | Que iid   | Name of the form queue the despooler was working with. |
|                                | Job iid   | Job-id of the job the despooler was outputting.        |
|                                | Str 1     | Yes if there is a string; otherwise, No.               |
|                                | Str 2     | Yes if there is a string; otherwise, No.               |

---

**Detailed Report Headings**

The detailed report is listed vertically because of the large number and sizes of the fields. In addition to the general report headings, the detailed report uses the following headings:

| Heading  | Description                                                                   |
|----------|-------------------------------------------------------------------------------|
| Dsp sts  | Despooler status.                                                             |
| Que sts  | Form queue status.                                                            |
| Job sts  | Job status.                                                                   |
| RtnStk   | Contents of the system return stack.                                          |
| String 1 | If a string is displayed, it is diagnostic information for support personnel. |
| String 2 | If a string is displayed, it is diagnostic information for support personnel. |

For an explanation of the values in the status columns refer Appendix B

## STAT-FILE

### Management and File Reallocation

The dictionary of STAT-FILE file now contains some macros useful for managing large machines. This depends on a list of modulus which are prime numbers at 10% intervals from 1 to 16,000,000. The data definition items in the dictionary of STAT-FILE which recommend modulus for files, REC.MODULO, as well as the SET-ALLOCATION program, depend on this list.

### Statistics for Indexes

STAT-FILE listings (including file statistics reports produced by standard backup utilities) show the name of each index section (see Chapter 7) after the name of the associated data section. This is preceded by a colon (:) and indented to the same column as the data section name. This allows you to see what indexes exist and how much space they occupy. The file number of each index section immediately succeeds that of the data section or preceding index section.

### Macros in DICT STAT-FILE

The dictionary of STAT-FILE now contains:

**ACCOUNT.SUMMARY**  
**N.ACCOUNT.SUMMARY**  
**W.ACCOUNT.SUMMARY**

which give summary data about accounts;

**FILES.WITH.GFES**

which lists files containing GFES; and:

**W.FILES.NZ.SEP**  
**W.UNDERSIZED.FILES**  
**W.VERY.UNDERSIZED.FILES**  
**W.OVERSIZED.FILES**  
**W.VERY.OVERSIZED.FILES**

which list files with the specified characteristic.

Macros whose names start with 'W' produce 136 column output; macros whose name starts with 'N' produce 80 column output. The administrator of the system is advised to modify these and other macros as required for a particular installation.

**Note:** System-delivered macros are overwritten on upgrade by the same or a modified version of the object. Therefore, you should ensure that you preserve any modifications you make as appropriate.

---

## On-line Documentation

On-line documentation is now available in the RealityX environment.

Two versions are available:

- RealityX On-line Help
- RealityX On-line Help for Windows

### RealityX On-line Help

The RealityX on-line Help can be 'called' from TCL or EDITOR.

**Note:** The information displayed currently relates to REALITY Release 7.x and has not yet been updated for Reality Release 4.0

#### Accessing from TCL

At TCL the documentation is referenced by means of the following command:

**MAN** {*element*}

where *element* is the name of a command, statement, conversion code or section number. If *element* is not specified, the system displays the main menu of documentation topics. See Chapter 6 for further information.

#### Access from EDITOR

In the EDITOR, it is referenced by the command:

**??** {*element*}

where *element* is the name of a command, statement, conversion code or section number. If *element* is not specified, the system displays the main menu of documentation topics. See Chapter 6 for further information.

When the documentation is called from EDITOR, it exits back to the EDITOR. **?? ??** provides an on-line description of the **??** command.

### RealityX On-line Help for Windows

RealityX On-line Help for Windows provides, in the form of a standard Windows help file, the information documented in the RealityX Quick Reference Guides. This means that all TCL, ENGLISH, EDITOR, Screen Editor, PROC and DATA/BASIC information in the QRGs can be accessed on-line using RealLink for Windows.

## Other Miscellaneous Changes

### Lower case TCL Commands

TCL commands can now be entered in lower-case or upper case letters.

### File References and Q-pointers

In any file-name referencing command or statement, files can be specified in the form:

`{DICT} /account-name/file-name{,data-section-name}`

Account-name and file-name can be a D-pointer or a Q-pointer. Account-name can be a Q-pointer in the master dictionary to a D- or R-pointer, and file-name can be a Q-pointer to a D-pointer. This enables you to access accounts and files on another database, local or remote, without needing to create a Q-pointer.

This capability exists in an R-pointer's target system. Hence, long chains or loops can exhaust the virtual ports on a database..

### Renaming Accounts

Accounts should now be renamed using the command:

`MOVE-FILE SYSTEM old-account-name`

`TO:SYSTEM new.account.name`

instead of using the `RENAME-FILE` command.

### Options Processor

Alphabetic options may be upper or lower case. Only standard ASCII alphabetic characters, A-Z and a-z, are accepted as alphabetic options.

### System Messages Formats

The system messages have been changed to upper and lower case letters and end with a full stop. If you have programs or PROCs that expect messages to be in upper case only, with no full stop, they will have to be changed.

Also a number of messages have been changed to include the message item number in parentheses at the beginning of the message. For example, "[202] Not on File".

### Database Configuration File

The `config` file for a Release 4.0 database resides in the `configs` directory, together with other optional configuration files (`tmp` and `spool`). The `configs` directory is located in the database directory. This differs from Release 3.1 where the `config` file resides in the database directory.

**reality -C Option** Suppresses the running of the LOGON PROC unless there is an R in attribute 9 of the account definition item.

# Appendix A

## PROC SYSTEM() Function Numbers

This appendix lists the functions of each of the SYSTEM() elements performed in PROC.

**PROC SYSTEM() Function Number**

| <b>SYSTEM()<br/>Element #</b> | <b><u>Information Returned</u></b>                                                                                                                                                                                  |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0                             | SYSERR system error status.                                                                                                                                                                                         |
| 1                             | 1 if print output is specified; otherwise 0.                                                                                                                                                                        |
| 2                             | Page width, as determined by the TERM setting.                                                                                                                                                                      |
| 3                             | Page length, as determined by the TERM setting.                                                                                                                                                                     |
| 4                             | Lines remaining on current page.                                                                                                                                                                                    |
| 5                             | Current page number.                                                                                                                                                                                                |
| 6                             | Current line count on a page.                                                                                                                                                                                       |
| 7                             | Terminal type as defined by TERM.                                                                                                                                                                                   |
| 8                             | Zero.                                                                                                                                                                                                               |
| 9                             | Current CPU millisecond count.                                                                                                                                                                                      |
| 10                            | 1 if the stack is on via STON statement; otherwise 0.                                                                                                                                                               |
| 11                            | 1 if an externally generated select-list is active; otherwise 0.                                                                                                                                                    |
| 12                            | System time in 1/10 seconds.                                                                                                                                                                                        |
| 13                            | Zero.                                                                                                                                                                                                               |
| 14                            | Typeahead count.                                                                                                                                                                                                    |
| 15                            | TCL options as three sub-elements, delimited by attribute marks:<br>001 alphabetic options, as a string of sorted letters<br>002 first numeric option (D4 interface)<br>003 second numeric option (D5 interface)    |
| 16                            | Context level.                                                                                                                                                                                                      |
| 17                            | Unused.                                                                                                                                                                                                             |
| 18                            | Port number.                                                                                                                                                                                                        |
| 19                            | Account name.                                                                                                                                                                                                       |
| 20                            | 1 if program is cataloged (thus, always returns 0 for PROC).                                                                                                                                                        |
| 21                            | Type of visual terminal characteristics supported by TERM termtype setting:<br>0 Invalid<br>1 Video characteristics not supported<br>2 Video character requires a CRT position<br>3 CRT position is not required    |
| 22                            | System configuration information, returned as multiple sub-elements delimited by attribute marks, and with the order and content defined by the DATA/BASIC Reference Manual's description of the SYSTEM() function. |
| 23                            | 0 if the BREAK key is enabled; otherwise non-zero.                                                                                                                                                                  |
| 24                            | 1 if echoing is enabled; otherwise 0.                                                                                                                                                                               |
| 25                            | 1 if process is phantom.                                                                                                                                                                                            |

**SYSTEM()**

| <b><u>Element #</u></b> | <b><u>Information Returned</u></b>                                                                                             |
|-------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| 26                      | Current PRMPC (prompt) character.                                                                                              |
| 27                      | 1 if PROC is active.                                                                                                           |
| 28                      | System privilege level (0, 1 or 2).                                                                                            |
| 29                      | MS.FRM.SZ (number of bytes of data space in a system frame; 1024 for 3.1 and 4.0).                                             |
| 30                      | 1 if pagination is in effect; otherwise 0.                                                                                     |
| 31                      | Reserved.                                                                                                                      |
| 32                      | Reserved.                                                                                                                      |
| 33                      | Reserved.                                                                                                                      |
| 34                      | Reserved.                                                                                                                      |
| 35                      | Number of denationalization language currently in use.                                                                         |
| 36                      | Number of denationalization collation table.                                                                                   |
| 37                      | Thousands separator.                                                                                                           |
| 38                      | Decimal separator.                                                                                                             |
| 39                      | Money sign.                                                                                                                    |
| 40                      | Name of the program currently executing.                                                                                       |
| 41                      | Release number of the OS.                                                                                                      |
| 42                      | Zero.                                                                                                                          |
| 43                      | Number of the port which has locked an item via a READU or F-UREAD statement.                                                  |
| 44                      | System environment type:<br>0 Series 6000<br>1 Series 18/19<br>3 RealityX                                                      |
| 45                      | 1 if the last item was binary (PROC cannot yet read binary items).                                                             |
| 46                      | 1 if the last item read was a D-pointer; otherwise 0.                                                                          |
| 47                      | 1 if currently inside a transaction.                                                                                           |
| 48                      | The CCI (Consistent Circuit Identifier); a value of -1 is returned for TIPH process.                                           |
| 49                      | The PLID (Physical Location Identifier); null is returned for TIPH process.                                                    |
| 50                      | User-id.                                                                                                                       |
| 51                      | Software user-id (which can be multi-valued).                                                                                  |
| 52                      | System name.                                                                                                                   |
| 53                      | Process activity snapshot (this is a dynamic array and is unusable in PROC, generally, but could be moved into a file buffer). |
| 54                      | Null for PROC only.                                                                                                            |
| 55                      | System time in milliseconds.                                                                                                   |
| 56                      | Zero.                                                                                                                          |

| <b>SYSTEM()<br/>Element #</b> | <b>Information Returned</b>                                                                                                                                                                           |
|-------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 57                            | Dynamic array containing overflow activity snapshot.                                                                                                                                                  |
| 58                            | Spooler assign information.                                                                                                                                                                           |
| 59                            | Unused                                                                                                                                                                                                |
| 60                            | Returns same information as U20E0. The TCL statement with the verb removed is returned as a series of sub-elements delimited by attribute marks. Redundant blanks and the options string are removed. |

# Appendix B

## Printer Independent Functions

This appendix introduces the printer independence feature and provides a listing of Classes 1 to 13.

## Introduction to Printer Independence

Because each printer manufacturer chooses its own method for specifying printer control strings for performing special functions such as ejecting a sheet, changing fonts, and so on, it is difficult for an operating system producer to provide tables for controlling all printer types.

The REALITY System provides a method by which a despooler can take specially coded strings from the job print data and convert them to control data that is meaningful to a given printer.

Using the PTR function of DATA/BASIC, you can specify a printer function, such as setting line spacing to six lines per inch, that is independent of the printer used. The DATA/BASIC compiler encodes the parameters of the PTR function into a string of characters called a Printer Independent Function Sequence (PIFS). When your print data is despoiled, the system translates the PIFS into a command sequence for a particular printer.

In the PTR function you specify a function class and a function subclass. The eject-page function, for example, is in class 1, subclass 10. A DATA/BASIC statement that specifies page eject would be:

```
PRINT PTR(1,10)
```

The system compiles this into a coded string of characters that is printer independent (PIFS). When the despooler encounters the PIFS, it converts the PIFS into the printer dependent commands required for its printer to perform the requested function.

If the printer function requires parameters, you also specify them in the PTR function. The Set n CPI function (class 7, subclass 5) sets the character pitch to a specified value. The statement to set the character pitch to 12 is:

```
PRINT PTR(7,5,12)
```

PIFS strings are intended mainly for prologues and epilogues, which you would create using option 3 of the SPM command (explained in the manual *Using the Spooler*). However, you can embed a PIFS in print data created by a DATA/BASIC program.

The remainder of this appendix presents the classes and subclasses of printer independent functions and their parameters that you can use with the PTR function.

---

## Printer Independent Function Classes

The printer independent functions for use in the PTR function are divided into thirteen classes. Each class has several subclasses. The remainder of this appendix presents a table for each class with its subclasses. Each table has the subclass numbers, the function name, a description of the function, and the format to use with the PTR function.

The classes and their reference tables are as follows:

| Class | Function           | Table Number |
|-------|--------------------|--------------|
| 1     | Device Control     | B-1          |
| 2     | Page Format        | B-2          |
| 3     | Line Format        | B-3          |
| 4     | Tabulations        | B-4          |
| 5     | Horizontal Control | B-5          |
| 6     | Line Spacing       | B-6          |
| 7     | Character Pitch    | B-7          |
| 8     | Form Layout        | B-8          |
| 9     | Special Effects    | B-9          |
| 10    | Typeface           | B-10         |
| 11    | Character Set      | B-11         |
| 12    | Font Selection     | B-12         |
| 13    | Colour             | B-13         |

### Example of a Prologue

To create a prologue for a form queue you would invoke the SPM command and enter option 3 to produce a print control string, which is a series of PIFS strings.

For this example say that each time the form queue is used to output a job to the printer all the margins are to be cleared, the left margin is to be set to 10, tabulation stops are to be set at positions 20, 35, and 50, and the alternate tray is to be selected.

The information to be gleaned from the class tables is the following:

| Function                        | Class | Subclass | PTR Format        |
|---------------------------------|-------|----------|-------------------|
| Clear All Margins               | 2     | 8        | PTR(2,8)          |
| Set Left Margin                 | 2     | 4        | PTR(2,4,p)        |
| Set Horizontal Tabulation Stops | 4     | 1        | PTR(4,1,p{,p}...) |
| Alternate Sheet Feeder          |       |          | PTR(1,8)          |

**Example of Using PTR in a Program**

The following DATA/BASIC statements could be the prologue:

```

PRINTER ON
PRINT PTR(2,8):PTR(2,4,10):PTR(4,1,20,35,50):PTR(1,8):
END

```

The PRINT statement ends with a colon so that a carriage return-line feed is not output as part of the sequence.

Say that you wish to print text that includes footnote numbers with numbers in the text as superscripted condensed characters. The class tables provide the following:

| Function                  | Class | Subclass | PTR Format |
|---------------------------|-------|----------|------------|
| Superscript On            | 9     | 19       | PTR(9,19)  |
| Super/Subscript Off       | 9     | 21       | PTR(9,21)  |
| Set Condensed Printing    | 7     | 10       | PTR(7,10)  |
| Cancel Condensed Printing | 7     | 11       | PTR(7,11)  |

The DATA/BASIC statements to achieve this could be:

```

FIRSTTEXT="some text"
FOOTNOTE= FOOTNOTE + 1
NEXTTEXT= "some more text:"
PRINT FIRSTTEXT:
PRINT PTR(9,19):PTR(7,10):FOOTNOTE:PTR(9,21):PTR(7,11):
PRINT NEXTTEXT

```

**Table B-1. Class 1: Device Control**

| Sub Class | Function                       | Description                                                                                            | PTR Format |
|-----------|--------------------------------|--------------------------------------------------------------------------------------------------------|------------|
| 1         | Disable Printer Independence   | For the remainder of the present job send any PIFS to the printer as is, that is, without translation. | PTR(1,1)   |
| 2         | Reset                          | Restore the printer to an initial setting. (Not all printers clear buffers with this command.)         | PTR(1,2)   |
| 3         | Initialize                     | Set the printer to an initial state.                                                                   | PTR(1,3)   |
| 4         | Queued Initialize              | When the current print job is complete, set the printer to an initial state.                           | PTR(1,4)   |
| 5         | Self Test                      | Execute the self test routine in the printer.                                                          | PTR(1,5)   |
| 6         | Clear Buffer                   | print or delete all data in the output buffer.                                                         | PTR(1,6)   |
| 7         | Main Sheet Feeder              | Select the primary, or default, paper tray.                                                            | PTR(1,7)   |
| 8         | Alternate Sheet Feeder         | Select the alternate, or secondary, paper tray.                                                        | PTR(1,8)   |
| 9         | Paper Tray n                   | Select paper tray n.                                                                                   | PTR(1,9n)  |
| 10        | Eject Sheet                    | Eject a sheet of paper.                                                                                | PTR(1,10)  |
| 11        | Eject and Load Sheet           | Eject the current sheet and reload from the active tray.                                               | PTR(1,11)  |
| 12        | MSB as is                      | Accept the most significant bit of each data byte as it is.                                            | PTR(1,12)  |
| 13        | MSB = 1                        | Force the most significant bit of each data byte to 1.                                                 | PTR(1,13)  |
| 14        | MSB = 0                        | Force the most significant bit of each data byte to 0.                                                 | PTR(1,14)  |
| 15        | Enable Remote On-line Control  | Enable on-line control at the printer.                                                                 | ptr(1,15)  |
| 16        | Disable Remote On-line Control | Disable on-line control at the printer                                                                 | PTR(1,16)  |

|    |                                 |                                                                                                     |           |
|----|---------------------------------|-----------------------------------------------------------------------------------------------------|-----------|
| 17 | Enable Print Control Sequences  | Enable the printer to print the control sequences it receives. This is a diagnostic function.       | PTR(1,17) |
| 18 | Disable Print Control Sequences | Disable the printer from printing the control sequences it receives. This is a diagnostic function. | PTR(1,18) |
| 19 | Time                            | Print the current system time in the format HH:MM:SS.                                               | PTR(1,19) |
| 20 | Date                            | Print the current system date in the format DD MMM YYYY.                                            | PTR(1,20) |

**Table B-2. Class 2: Page Format**

| Sub Class | Function                    | Description                                                               | PTR Format |
|-----------|-----------------------------|---------------------------------------------------------------------------|------------|
| 1         | Set Top Margin              | Set the top margin to line position p.                                    | PTR(2,1p)  |
| 2         | Set Bottom Margin           | Set the bottom margin to line position p.                                 | PTR(2,2p)  |
| 3         | Set Right Margin            | Set the right margin to character position p.                             | PTR(2,3p)  |
| 4         | Set Left Margin             | Set the left margin to character position p.                              | PTR(2,4p)  |
| 5         | Set Text Length             | Define the number (n) of lines from the top to bottom margins.            | PTR(2,5n)  |
| 6         | Clear Vertical Margins      | Reset the vertical margins to their default settings.                     | PTR(2,6)   |
| 7         | Clear Horizontal Margins    | Reset the horizontal margins to their default settings.                   | PTR(2,7)   |
| 8         | Clear All Margins           | Reset both the vertical and horizontal margins to their default settings. | PTR(2,8)   |
| 9         | Set Top of Form             | Set the current printer position at the top of form.                      | PTR(2,9)   |
| 10        | Set Skip Over Perforation   | Set the number (n) of line feeds to allow for perforations.               | PTR(2,10n) |
| 11        | Clear Skip Over Perforation | Reset the skip-over-perforation value to zero.                            | PTR(2,11)  |

**Table B-3. Class 3: Line Format**

| Sub Class | Function             | Description                                                          | PTR Format |
|-----------|----------------------|----------------------------------------------------------------------|------------|
| 1         | Auto-Justify Off     | Disable the automatic left- and right- justification of print lines. | PTR(3,1)   |
| 2         | Left-Justify On      | Enable left-justification of print lines.                            | PTR(3,2)   |
| 3         | Right-Justify On     | Enable right-justification of print lines.                           | PTR(3,3)   |
| 4         | Auto-Centre On       | Enable the automatic centring of print lines.                        | PTR(3,4)   |
| 5         | Full Auto-Justify On | Enable the automatic left- and right-justification of print lines.   | PTR(3,5)   |
| 6         | Auto-Linefeed On     | Enable automatic advancement of paper after a carriage return.       | PTR(3,6)   |
| 7         | Auto-Linefeed Off    | Disable automatic advancement of paper after a carriage return.      | PTR(3,7)   |
| 8         | Auto CR On           | Enable automatic return of print head after a line feed.             | PTR(3,8)   |
| 9         | Auto CR Off          | Disable automatic return of print head after a line feed.            | PTR(3,9)   |

**Table B-4. Class 4: Tabulations**

| Sub Class | Function                                        | Description                                                                                                                       | PTR Format                |
|-----------|-------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|---------------------------|
| 1         | Set Horizontal Tabulations Stops                | Set horizontal tabulations stops at character positions $p$ .                                                                     | PTR(4,1, $p\{,p\}...$ )   |
| 2         | Clear All Horizontal Tabulations Stops          | Clear all horizontal tabulation stops. On some printers this may reassert default tabulation stops.                               | PTR(4,2)                  |
| 3         | Set Vertical Tabulation Stops                   | Set vertical tabulation stops at line position $p$ .                                                                              | PTR(4,3, $p\{,p\}...$ )   |
| 4         | Set Vertical Tabulation Channel                 | Set vertical tabulation stops on channel $c$ at line positions $p$ . On some printers this may reassert default tabulation stops. | PTR(4,4, $c,p\{,p\}...$ ) |
| 5         | Select Vertical Channel (VFU)                   | Select channel $c$ for subsequent use.                                                                                            | PTR(4,5, $c$ )            |
| 6         | Clear All Vertical Tabulation Stops             | Clear all vertical tabulation stops.                                                                                              | PTR(4,6)                  |
| 7         | Clear Vertical Channel's Tabulation Stops (VFU) | Clear all tabulation stops on channel $c$ .                                                                                       | PTR(4,7, $c$ )            |
| 8         | EVFU Load                                       | Send an EVFU loading sequence to the printer. Each $c$ is a channel number, and the numbers must be in line-number sequence.      | PTR(4,8, $c\{,c\}...$ )   |

**Table B-5. Class 5: Horizontal Control**

| Sub Class | Function                                 | Description                                                                                                                | PTR Format     |
|-----------|------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|----------------|
| 1         | Bi-directional Printing On               | Enable printing in both directions.                                                                                        | PTR(5,1)       |
| 2         | Bi-directional Printing Off              | Disable printing in both directions, allowing unidirectional printing only.                                                | PTR(5,2)       |
| 3         | Move Horizontal, Relative, in Columns    | Change the position of the cursor/head on the current line to column displacement $d$ relative to the current position.    | PTR(5,3, $d$ ) |
| 4         | Move Horizontal, to Absolute Column      | Change the position of the cursor/head on the current line to column displacement $d$ from the left margin.                | PTR(5,4, $d$ ) |
| 5         | Move Horizontal, Relative, in Dots       | Change the position of the cursor/head on the current line to dot displacement $d$ relative to the current position.       | PTR(5,5, $d$ ) |
| 6         | Move Horizontal, Absolute, in Dots       | Change the position of the cursor/head on the current line to dot displacement $d$ from the left margin.                   | PTR(5,6, $d$ ) |
| 7         | Move Horizontal, Relative, in Decipoints | Change the position of the cursor/head on the current line to decipoint displacement $d$ relative to the current position. | PTR(5,7, $d$ ) |
| 8         | Move Horizontal, Absolute, in Decipoints | Change the position of the cursor/head on the current line to decipoint displacement $d$ from the left margin.             | PTR(5,8, $d$ ) |
| 9         | Horizontal Motion Index                  | Set the Horizontal Motion Index (HMI), which is distance $d$ between characters or columns.                                | PTR(5,9, $d$ ) |
| 10        | Backspace                                | Move the cursor/head over the most recently printed character.                                                             | PTR(5,10)      |
| 11        | Carriage Return                          | Move the cursor/head to the current left margin.                                                                           | PTR(5,11)      |
| 12        | Horizontal Tabulation                    | Move the cursor/head to the next horizontal tabulation stop.                                                               | PTR(5,12)      |

**Table B-6. Class 6: Line Spacing**

| Sub Class | Function                      | Description                                                                                    | PTR Format            |
|-----------|-------------------------------|------------------------------------------------------------------------------------------------|-----------------------|
| 1         | 6 LPI                         | Set line spacing to six lines per inch (8/48" per line).                                       | PTR(6,1)              |
| 2         | 8 LPI                         | Set line spacing to eight lines per inch (6/48" per line).                                     | PTR(6,2)              |
| 3         | 10 LPI                        | Set line spacing to ten lines per inch (7/72" per line).                                       | PTR(6,3)              |
| 4         | Store a Line Spacing Value    | Store line spacing fraction values as numerator ( <i>n</i> ) and denominator ( <i>d</i> ).     | PTR(6,4, <i>n,d</i> ) |
| 5         | Set Stored Line Spacing Value | Set line spacing to the previously stored line spacing values.                                 | PTR(6,5)              |
| 6         | Set $n/360$ " Line Spacing    | Change the paper feed amount to $n/360$ " per line.                                            | PTR(6,6, <i>n</i> )   |
| 7         | Set $n/180$ " Line Spacing    | Change the paper feed amount to $n/180$ " per line.                                            | PTR(6,7, <i>n</i> )   |
| 8         | Set $n/60$ " Line Spacing     | Change the paper feed amount to $n.60$ " per line.                                             | PTR(6,8, <i>n</i> )   |
| 9         | Set $n/48$ " Line Spacing     | Change the paper feed amount to $n/48$ " per line.                                             | PTR(6,9, <i>n</i> )   |
| 10        | Vertical Motion Index         | Set the Vertical Motion Index (VMI), which is the distance ( <i>d</i> ) between lines or rows. | PTR(6,10, <i>d</i> )  |
| 11        | 8 Point Character Height      | Set the character height to 8 points.                                                          | PTR(6,11)             |
| 12        | 10 Point Character Height     | Set the character height to 10 points.                                                         | PTR(6,12)             |
| 13        | 12 Point Character Height     | Set the character height to 12 points.                                                         | PTR(6,13)             |
| 14        | 18 Point Character Height     | Set the character height to 18 points.                                                         | PTR(6,14)             |
| 15        | Form Feed                     | Advance the paper until the top of page is at the cursor/head.                                 | PTR(6,15)             |

|    |                     |                                                                                           |           |
|----|---------------------|-------------------------------------------------------------------------------------------|-----------|
| 16 | Line-feed           | Advance the paper until the cursor/head is at the same character position one line down.  | PTR(6,16) |
| 17 | Half Line-feed      | Advance paper until the cursor/head is at the same character position one-half line down. | PTR(6,17) |
| 18 | Reverse Line-feed   | Reverse move the paper until cursor/head is at the same character position one line up.   | PTR(6,18) |
| 19 | Vertical Tabulation | Advance the paper until the cursor/head is at the next vertical tabulation stop.          | PTR(6,19) |

**Table B-7. Class 7: Character Pitch**

| Sub Class | Function                  | Description                                                                   | PTR Format     |
|-----------|---------------------------|-------------------------------------------------------------------------------|----------------|
| 1         | Set 8 CPI                 | Set character pitch to fixed at eight characters per inch.                    | PTR(7,1)       |
| 2         | Set 10 CPI (pics)         | Set character pitch to fixed at ten characters per inch.                      | PTR(7,2)       |
| 3         | Set 12 CPI (elite)        | Set character pitch to fixed at twelve characters per inch.                   | PTR(7,3)       |
| 4         | Set 15 CPI (micron0       | Set character pitch at fifteen characters per inch.                           | PTR(7,4)       |
| 5         | Set $n$ CPI               | Set the character pitch to $n$ characters per inch.                           | PTR(7,5, $n$ ) |
| 6         | Character Pitch Offset    | Define additional increment $i$ to be added to a character's width.           | PTR(7,6, $i$ ) |
| 7         | (reserved)                |                                                                               |                |
| 8         | Proportional Printing On  | Enable proportional spacing during printing.                                  | PTR(7,8)       |
| 9         | Proportional Printing Off | Disable proportional spacing during printing.                                 | PTR(7,9)       |
| 10        | Set Condensed Printing    | Enable condensed, or compressed, printing, which reduces the character pitch. | PTR(7,10)      |
| 11        | Cancel Condensed Printing | Disable condensed, or compressed, printing.                                   | PTR(7,11)      |

**Table B-8. Class 8: Form Layout**

| Sub Class | Function                     | Description                                      | PTR Format |
|-----------|------------------------------|--------------------------------------------------|------------|
| 1         | Set Form Length = n Lines    | Define paper length in number of lines n.        | PTR(8,1,n) |
| 2         | Set Form Length in Inches    | Define paper length in number of inches n.       | PTR(8,2,n) |
| 3         | Set Form for Letter Size     | Define paper as letter size (8.5" X 11").        | PTR(8,3)   |
| 4         | Set Form for Legal Size      | Define paper as legal size (8.5" X 14").         | PTR(8,4)   |
| 5         | Set form for A4 Size         | Define paper as A4 size 210mm X 297mm).          | PTR(8,5)   |
| 6         | Set Form for Specified Size  | Define paper as size s.                          | PTR(8,6,s) |
| 7         | Select Portrait Orientation  | Set the printer to print across the page width.  | PTR(8,7)   |
| 8         | Select Landscape Orientation | Set the printer to print across the page length. | PTR(8,8)   |

**Table B-9. Class 9: Special Effects**

| Sub Class | Function             | Description                                                                           | PTR Format |
|-----------|----------------------|---------------------------------------------------------------------------------------|------------|
| 1         | Print Double Wide    | Enable printing characters twice their specified width.                               | PTR(9,1)   |
| 2         | Cancel Double Wide   | Disable printing double wide characters.                                              | PTR(9,2)   |
| 3         | Printing Double High | Enable printing characters twice their specified height.                              | PTR(9,3)   |
| 4         | Cancel Double High   | Disable printing characters twice their specified height.                             | PTR(9,4)   |
| 5         | Print Emphasized     | Enable printing each character twice, the second time a little offset from the first. | PTR(9,5)   |
| 6         | Cancel Emphasized    | Disable printing emphasized characters.                                               | PTR(9,6)   |
| 7         | Shadow On            | Enable printing double-strike with offset.                                            | PTR(9,7)   |
| 8         | Shadow Off           | Disable printing double-strike with offset.                                           | PTR(9,8)   |
| 9         | Outline On           | Enable printing outline characters.                                                   | PTR(9,9)   |
| 10        | Outline Off          | Disable printing outline characters.                                                  | PTR(9,10)  |
| 11        | Underline On         | Enable printing underline.                                                            | PTR(9,11)  |
| 12        | Underline Off        | Disable printing underline.                                                           | PTR(9,12)  |
| 13        | Bold On              | Enable printing bold characters.                                                      | PTR(9,14)  |
| 14        | Bold Off             | Disable printing bold characters.                                                     | PTR(9,14)  |
| 15        | Strike-through On    | Enable printing the strike-through character.                                         | PTR(9,15)  |
| 16        | Strike-through Off   | Disable printing the strike-through character.                                        | PTR(9,16)  |
| 17        | Overscore On         | Enable printing overscore.                                                            | PTR(9,17)  |
| 18        | Overscore Off        | Disable printing overscore.                                                           | PTR(9,18)  |

|    |                        |                                                                                       |           |
|----|------------------------|---------------------------------------------------------------------------------------|-----------|
| 19 | Superscript On         | Enable printing characters a half line above the current line.                        | PTR(9,19) |
| 20 | Subscript On           | Enable printing characters a half line below the current line.                        | PTR(9,20) |
| 21 | Super/Subscript On     | Disable superscript and subscript printing.                                           | PTR(9,21) |
| 22 | Italic Off             | Disable printing italic style.                                                        | PTR(9,22) |
| 23 | Italic On              | Enable printing italic style.                                                         | PTR(9,23) |
| 24 | Select Draft Quality   | Enable printing in rough draft quality (usually faster than higher quality printing). | PTR(9,24) |
| 25 | Select Highest Quality | Enable printing in the printer's highest quality.                                     | PTR(9,25) |

**Table B-10. Class 10: Typeface**

| Sub Class | Function                  | Description                            | PTR Format           |
|-----------|---------------------------|----------------------------------------|----------------------|
| 1         | Select Default Typeface   | Select the printer's default typeface. | PTR(10,1)            |
| 2         | Select Roman Typeface     | Select Roman typeface.                 | PTR(10,2)            |
| 3         | Select Gothic Typeface    | Select Gothic typeface.                | PTR(10,3)            |
| 4         | Select Courier Typeface   | Select Courier typeface.               | PTR(10,4)            |
| 5         | Select Prestige Typeface  | Select Prestige typeface.              | PTR(10,5)            |
| 6         | Select Helvetica Typeface | Select Helvetica typeface.             | PTR(10,6)            |
| 7         | Select Script Typeface    | Select Script typeface.                | PTR(10,7)            |
| 8         | Select Typeface <i>n</i>  | Select typeface <i>n</i> .             | PTR(10,8, <i>n</i> ) |

**Table B-11. Class 11: Character Set**

| Sub Class | Function                     | Description                                                                            | PTR Format  |
|-----------|------------------------------|----------------------------------------------------------------------------------------|-------------|
| 1         | Character Set 1              | Select the primary, or default, character set.                                         | PTR(11,1)   |
| 2         | Character Set 2              | Select the secondary, or alternate, character set.                                     | PTR(11,2)   |
| 3         | Graphics 1                   | Select the primary, or default, graphics set.                                          | PTR(11,3)   |
| 4         | Graphics 2                   | Select the secondary, or alternate, graphics set.                                      | PTR(11,4)   |
| 5         | Select International Set     | Select international character set n.                                                  | PTR(11,5,n) |
| 6         | Select Character Set n       | Select character set n.                                                                | PTR(11,6,n) |
| 7         | Select Extended Characters   | Select, in the current set, characters represented by numbers in the range 128 to 255. | PTR(11,7)   |
| 8         | Deselect Extended Characters | Disable the use of extended characters.                                                | PTR(11,8)   |
| 9         | Select User Set              | Select the character set downloaded, or defined, by the user.                          | PTR(11,9)   |
| 10        | Deselect User Set            | Deselect the user-loaded character set and re-enable the normal, or default, set.      | PTR(11,10)  |
| 11        | Remap User's Set             | Remap user-loaded characters from positions 0-127 to positions 128-255.                | PTR(11,11)  |
| 12        | Enable Control Codes         | Enable treating character codes 128-159 as control codes.                              | PTR(11,12)  |
| 13        | Print Control Codes          | Enable treating character codes 128-159 as printable characters.                       | PTR(11,13)  |

**Table B-12. Class 12: Font Selection**

| Sub Class | Function                | Description                                   | PTR Format |
|-----------|-------------------------|-----------------------------------------------|------------|
| 1         | Select Primary Font Set | Select the primary, or default, font set.     | PTR(12,1)  |
| 2         | Select Font Set 2       | Select the secondary, or alternate, font set. | PTR(12,2)  |
| 3         | Select Font Set 3       | Select font set 3.                            | PTR(12,3)  |
| 4         | Select Font Set 4       | Select font set 4.                            | PTR(12,4)  |
| 5         | Select Font Set 5       | Select font set 5.                            | PTR(12,5)  |
| 6         | Select Font Set 6       | Select font set 6.                            | PTR(12,6)  |
| 7         | Select Font Set 7       | Select font set 7.                            | PTR(12,7)  |
| 8         | Select Font Set 8       | Select font set 8.                            | PTR(12,8)  |
| 9         | Select Sub-font Set 1   | Select sub font 1 from a set of fonts.        | PTR(12,9)  |
| 10        | Select Sub-font Set 2   | Select sub-font 2 from a set of fonts.        | PTR(12,10) |
| 11        | Select Sub-font Set 3   | Select sub-font 3 from a set of fonts.        | PTR(12,11) |
| 12        | Select Sub-font Set 4   | Select sub-font 4 from a set of fonts.        | PTR(12,12) |
| 13        | Select Sub-font Set 5   | Select sub-font 5 from a set of fonts.        | PTR(12,13) |
| 14        | Select Sub-font Set 6   | Select sub-font 6 from a set of fonts.        | PTR(12,14) |
| 15        | Select Sub-font Set 7   | Select sub-font 7 from a set of fonts.        | PTR(12,15) |
| 16        | Select Sub-font Set 8   | Select sub-font 8 from a set of fonts.        | PTR(12,16) |

**Table B-13. Class 13: Colour**

| Sub Class | Function | Description                     | PTR Format |
|-----------|----------|---------------------------------|------------|
| 1         | Black    | Select printing colour black.   | PTR(13,1)  |
| 2         | Magenta  | Select printing colour magenta. | PTR(13,2)  |
| 3         | Cyan     | Select printing colour cyan.    | PTR(13,3)  |
| 4         | Violet   | Select printing colour violet.  | PTR(13,4)  |
| 5         | Yellow   | Select printing colour yellow.  | PTR(13,5)  |
| 6         | Red      | Select printing colour red.     | PTR(13,6)  |
| 7         | Green    | Select printing colour green.   | PTR(13,7)  |

# Appendix C

## TDM and Terminal Definitions

This appendix discusses the changes in terminal definitions in release 4.0.

## TDM and Terminal Definitions

There is a new format for terminal definitions. Terminal definitions from previous releases are automatically converted to the new format.

The Cursor Definition File CURSOR-DEFS is now used for all terminal types, rather than having some of the definitions as internal data and some on file. Also, the file representation of TDM definitions has changed so that the shared item in POINTER-FILE is no longer used.

The source items in CURSOR-DEFS that define terminal types are generated by TDM, which also 'compiles' the definition into a form convenient for runtime use.

Each item now contains the values for the maximum row and column, form-feed delay, line-feed delay, terminal line skip and alternate backspace. When a port logs on or changes term-type, an item is read from CURSOR-DEFS and used to build a terminal definition table in the port's workspace. This means that changing a terminal definition does not affect already logged-on users.

However, a user logging on or changing term type with the TERM verb acquires all of the above characteristics from the TDM definition item, unless they are overridden by explicit specifications in the TERM verb. Any or all of these characteristics may be modified for the particular port by successive uses of TERM.

**Miscellaneous**

%FULL command 6-2, 6-7  
 .command 6-11

**A**

ACCOUNT.SUMMARY macro 11-4  
 AND-ITEMS command 6-2, 6-7  
 AND-LISTS command 6-2, 6-7  
 ASSIGN command 6-4, 6-11

**B**

BASIC command 6-4, 6-11  
   C option 10-18  
 BLIST command 6-4, 6-11  
 BLOCK-PRINT command 6-4, 6-11  
 BLOCK-TERM command 6-4, 6-12  
 BUILD.DESPOOLERS command 6-2, 6-7  
 BVERIFY command 6-4, 6-12  
 BY-EXP-SUB connective 7-7  
 BY-EXP-SUB-DSD connective 7-7

**C**

CATALOG command 6-4, 6-13  
 Cataloged programs, DATA/BASIC  
   Sharing 10-20  
 Catalogued programs, DATA/BASIC 10-21  
 CHAR Function 10-13  
 CLEAR-OPTION command 6-2, 6-7  
 CLEAR-SP-LOCK command 5-7, 6-6  
 compilers, DATA/BASIC 10-18  
**config** parameters  
   ShareModulo 10-20  
   ShareSize 10-20  
 CONNECT statement 10-4  
 Connectives 7-7  
 Conversions, date  
   DATA/BASIC 10-21  
 CONVERT.OBJECT command 6-2, 6-7  
 CREATE-ACCOUNT command 6-13

CREATE-FILE command 6-4, 6-14  
 CREATE-INDEX command 6-2, 6-7  
 CURSOR-DEFS file C-2

**D**

D command, DATA/BASIC Debug 10-16  
 DATA.PRODUCTS command 6-14  
 DATA/BASIC 10-1 to 10-23  
 DATA/BASIC compilers  
   alternatives 10-17  
 DATA/BASIC object sharing 10-20  
 Database  
   partition type defined 2-2  
 Date conversions  
   display changed 7-3  
 Date conversions, DATA/BASIC 10-21  
 DB command 6-4  
 DEBUG command 6-2, 6-7  
 DECAT command 6-2, 6-7  
 DECRYPT function 10-13  
 DEFINE-INDEX command 6-2, 6-7  
 DELETE-ACCOUNT command 6-14  
 DELETE-INDEX command 4-3, 6-2, 6-8  
 Despooler errors 11-2  
 DESPOOLER.DETAIL command 6-2, 6-8  
 DISCTOTAPE command 6-2, 6-8  
 Disk space  
   monitoring 2-4  
 Disk striping 2-2  
 Dot processor 6-11  
 DSM command 6-8  
 DSP.ERRORS macro 11-2  
 DSPMON command 6-2, 6-8

**E**

EBASIC command 6-6, 10-23  
 EDELETE command 4-3, 7-9  
 ENCRYPT function 10-13  
 ENGLISH 7-1 to 7-10

## Errors

despooler 11-2

EVFU-SETUP command 6-14

**F**

FILES.WITH.GFEs macro 11-4

FILE-SAVE command 6-4

FIND command 6-4, 6-15

FMT function 10-9

FOLD function 10-9

FOR statement 10-4

FQM command 6-2, 6-8

Function, PTR B-3

**G**

GET-LIST command 6-19

GFE's 2-7

GO command 6-2, 6-8

Group format errors – *see* GFE's

Grouping connectives 7-7

**H**

Hilite capability 6-18

**I**

ICONV function 10-22

INCLUDE statement 10-4

Independence, printer B-2, B-3

Index

statistics 11-4

Indexing 4-1, 4-2

**L**

L command, DATA/BASIC Debug 10-16

List processing verbs 7-5

LIST SYSTEM-LOG

DSP.ERRORS 11-2

LISTFILES command 6-4, 6-15

LIST-SPREAD command 6-8, 7-5

LOAD-ALPHA command 6-2, 6-8

LOAD-BNF command 6-2, 6-8

LOADBNF command 10-17, 10-23

Location of lists 7-6

LOGTO command 6-4, 6-15

LOOP statement 10-5

**M**

Macros in DICT STAT-FILE 11-4

MAN command 6-2, 6-8

M-A-S command 6-16

Masks 7-2

MAX connective 7-7

MC (mask character) code 10-22

MIN connective 7-7

MOVE-FILE command 6-16

MSG command 6-4, 6-16

Multiple-Account-Save – *see* M-A-S command**N**

N.ACCOUNT.SUMMARY macro 11-4

Networking 9-1 to 9-2

NEW-SORT-LIST command 6-4, 6-17

NSELECT command 6-2, 6-8

**O**

OCONV function 10-22

On-line documentation

overview 11-5

RealLink for Windows 11-5

Options processor 11-6

OR-ITEMS command 6-2, 6-8

OR-LISTS command 6-2, 6-9

**P**

PAGE statement 10-5

Partition database 2-1 to 2-7

creating 2-3

defined 2-2  
 differences from filestore type 2-3  
 disk striping 2-2  
 errors 2-7  
 purpose 2-2  
 PCSM command 6-2, 6-9  
 PDM command 6-2, 6-9  
 PERFORM statement 10-5, 10-6  
 PIFS B-2  
 POINTER-FILE file C-2  
 PORT-DESPOOL command 6-4, 6-18  
 PORT-OUT command 6-18  
 PORTOUT command 6-4  
 POVf command 6-2, 6-9  
 PRECISION statement 10-6  
 Printer independence B-2, B-3  
 Printer Independence Function Sequence  
   PIFS B-2  
 PRINT-HEADER command 6-6, 10-23  
 PROC 8-1 to 8-5  
 PROC SYSTEM() function numbers A-2, C-2  
 PTR function 10-9, B-2, B-3

## R

READNEXT statement 10-2, 10-7  
**realdbck** utility 2-5  
 RECWAIT statement 10-7  
 REFORMAT command 6-4, 6-18  
 Remote spooling 5-9  
 RENAME-FILE command 6-4, 6-18, 11-6  
 Renaming accounts 11-6  
 Reserved words, DATA/BASIC 10-23  
 RESET.DESPOOLER command 6-2, 6-9  
 RESET-ALPHA command 6-2, 6-9  
 ROUND function 10-10  
 Routing files 9-2  
 RQM statement 10-7  
 RUN command 6-4, 6-18  
 RUNOFF command 6-4, 6-18

## S

SAVE command 6-4, 6-19  
 SAVE-LIST command 6-4, 6-19  
 SELECT command 10-6  
 SELECT statement 10-8  
 SELECT-INDEX command 6-2, 6-9  
 SEL-RESTORE command 6-4, 6-19  
 SEQ function 10-13  
 SET-ALLOCATION program 11-4  
 SET-ALPHA command 6-2, 6-9  
 SET-OPTION command 6-2, 6-9  
 Shadow database 3-1  
 Shared DATA/BASIC object 10-20  
 SharedSize parameter 10-20  
 ShareModulo parameter 10-20  
 SHOW-MODULI command 6-2, 6-9  
 SLEEP statement 10-7  
 Sort connectives 7-7  
 SORT-LIST command 6-4, 6-19  
 SORT-SPREAD command 6-2, 6-9, 7-5  
 SP-ALIGN command 6-4, 6-20  
 SP-ASSIGN command 6-4, 6-20  
 SP-CLEAR command 6-4  
 SP-CLOSE command 6-4, 6-20  
 SP-COPIES command 6-4  
 SP-COPY command 6-2, 6-9  
 SP-CREATE command 6-4, 6-20  
 SP-DELETE command 6-4, 6-20  
 SP-DESCRIPTION command 6-2, 6-9  
 SP-DESPOOLERS 6-10  
 SP-DESPOOLERS command 6-3  
 SP-DETAIL command 6-3, 6-10  
 SP-DEVICE command 6-4, 6-20  
 SP-EDIT command 6-5  
 SP-FIX command 6-10  
 SP-FORM command 5-7, 6-6  
 SP-FORMQUEUE command 6-3, 6-10  
 SP-FQDELETE command 6-5, 6-20  
 SP-JOBS command 6-5  
 SP-KILL command 6-5, 6-21  
 SP-LOOK command 6-5, 6-21  
 SPM command 6-3, 6-10

SP-MOVEQ command 6-5, 6-21  
SP-NEWTAB 6-6  
SP-NEWTAB command 5-7  
Spooler 5-1 to 5-9  
Spooler commands  
    Obsolete 5-7  
SPOOLER function 10-13  
Spooling to a remote system 5-9  
SP-OPTS command 6-5, 6-21  
SP-RESUME command 6-5, 6-21  
SP-STATUS command 6-5, 6-21  
SP-STOP command 6-5  
SP-SUSPEND command 6-5, 6-21  
SP-SWITCH command 6-5, 6-21  
SP-TRANSLATE command 6-5  
SP-TYPE command 5-7, 6-6  
SSM command 6-5, 6-21  
START-DESPOOLER command 6-3, 6-10  
START-NET-PTR 6-6  
START-PRINTER command 6-5  
STAT command 6-5, 6-23  
STOP-DESPOOLER command 6-3, 6-10  
STOP-PRINTER command 6-5  
STOREPF 6-6  
STOREPF command 10-23  
SUBROUTINE statement 10-8  
SYSTEM() function numbers, PROC A-2, C-2  
SYSTEM(57) function 10-14  
SYSTEM(58) function 10-14  
SYSTEM(60) function 10-15  
SYSTEM(61) function 10-15

## T

TAPETO DISC command 6-10  
TAPETODISC command 6-3  
T-ATT command 6-5, 6-23  
TCL stacker 6-4, 6-11  
T-DET command 6-5, 6-23  
TDM C-2  
T-DUMP command 6-5, 6-23  
TERM command C-2  
Terminal definitions C-2

Tfile code 7-4  
TIMEOUT, DATA/BASIC 10-23  
T-LOAD command 6-5, 6-23  
Translation code 7-4  
TRANSTART statement 10-8  
TRUNC function 10-11  
T-STATUS command 6-5, 6-23

## U

Upgrade utilities, DATA/BASIC 10-22  
UPGRADE.BASIC.OBJECT command 6-3, 6-10  
USER command 6-5, 6-23

## V

V command, DATA/BASIC Debug 10-16  
Value limiters  
    masks 7-2  
Variables, DATA/BASIC 10-21  
VARTYPE function 10-11  
VARVAL function 10-12  
VARVALSET statement 10-3  
VARVALTYPE function 10-12  
VERIFY-INDEX command 6-3, 6-10

## W

W.ACCOUNT.SUMMARY macro 11-4  
W.OVERSIZED.FILES macro 11-4  
W.UNDERSIZED.FILES macro 11-4  
W.VERY.OVERSIZED.FILES macro 11-4  
W.VERY.UNDERSIZED.FILES macro 11-4

## X

XOR-ITEMS command 6-3, 6-10  
XOR-LISTS command 6-3, 6-10  
XSELECT command 6-3, 6-10