

Reality

V14.1

Transferring a Database

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, 2010.

Document No. UM70006960AR3
December 2010

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

Introduction 4
Procedure on the Source System 5
Source Database on Series 18/19 (Release 7.2) 5
Source Database on RealityX 3.1C, 4.1x or 5.0x, or Reality 8.0x or 8.1x 5
Procedure on the Destination System 7
SYS-UPDATE Details 8
Data/Basic Conversion 8
Changing Lower- to Upper-Case Dates 9

Introduction

When upgrading an older system running Reality, or migrating Reality data to a new platform, it is often necessary, or preferable, to create new Reality databases and transfer the data from the old system via a FILE-SAVE tape. This technique has the advantage that the data is defragmented and, if file reallocation parameters have been defined, files are resized at the same time.

The main steps in transferring a database are:

- If the source is a Series 18/19, carry out a FILE-SAVE and VERIFY-SAVE to secure the database, then apply a patch as appropriate to the destination.
- Secure any local customisation.
- Carry out a FILE-SAVE and VERIFY-SAVE on the updated source database.
- Customise the Reality configuration on the destination host.
- Create a new database on the destination host.
- Restore the last FILE-SAVE from the source system to the new database.
- Run SYS-UPDATE on the new database to apply necessary updates to it.
- If you have old applications that require it, configure the database to use the old upper-case date format
- Carry out a FILE-SAVE and VERIFY-SAVE on the new database to secure it.
- Start transaction processing on the new database if required, and enable logons.

The detailed steps depend on the source and destination systems, as described in the remainder of this document.

Procedure on the Source System

Source Database on Series 18/19 (Release 7.2)

1. Carry out a **FILE-SAVE** and a **VERIFY-SAVE** of the database.

2. Apply custom patch 008 (if transferring to UNIX) or 10 (if transferring to Windows) to make multi-reel file saves compatible with the destination database.

3. Save to elsewhere in the database any system file items that you have customised. Customised files often include SYSPROG-PL, PROCLIB, BP, SYSBP, SYSBP.MSGS, SYSPL, SYS.BASLIB, BASIC.COMPILERS and NEWAC. It is advisable to save the whole contents of these files if in doubt about what has been customised within them.

   Excluding Accounts/Files from Backup4. **FILE-SAVE** and **VERIFY-SAVE** the database.

   Note: This file save will be restored to a new database on the destination host, so you must use a compatible format. Windows does not support ½ inch tape drives.

5. Use the **POVF** verb to display the total size of the database. Note this size to ensure that the database created on the destination system is sufficiently large.

Source Database on RealityX 3.1C, 4.1x or 5.0x, or Reality 8.0x or 8.1x

1. Save any customised database configuration files.

   - On RealityX Release 3.1C, these are located in the UNIX directory DatabasePath/config (where DatabasePath is the location of the database).

   - On later UNIX releases, they are located in the directory DatabasePath/configs/config.

   - On Windows systems, they are located in the Windows folder DatabasePath\configs\config.

2. Save any PLId to port maps from:

   `$REALROOT/files/devices` (UNIX)
   or
   drive`:\Realman\RealityVersion\files\devices` (Windows).

3. Save to elsewhere in the database any system file items that you have customised. Customised files often include SYSPROG-PL, PROCLIB, BP, SYSBP, SYSBP.MSGS, SYSPL, SYS.BASLIB, BASIC.COMPILERS and NEWAC. It is advisable to save the whole contents of these files if in doubt about what has been customised within them.

4. **FILE-SAVE** and **VERIFY-SAVE**.

Note: This file save will be restored to a new database on the destination host, so you must use a compatible format. Windows does not support ½ inch tape drives.

5. Use the **POVF** verb to display the total size of the database. Note this size to ensure that the database created on the destination system is sufficiently large.

6. On a UNIX host, note the numbers of any RPQs that are installed.

7. If Transaction Handling is enabled, use the command **plog -h** to display details of the raw log configuration. Take a note of the raw log size and the buffer size.

8. On a UNIX host, save any printer interface scripts from `$REALROOT/files` to a safe location.

Procedure on the Destination System

1. Integrate any saved PLId to port maps into the devices file:

   /usr/Realman/*RealityVersion*/files/devices (UNIX)

   or

   Drive:\Realman*RealityVersion*\files\devices (Windows)

2. On UNIX, copy any saved printer interface scripts into: $REALROOT/files/interfaces.

3. On UNIX, check that any RPQs that were installed on the old system are available on the new system. If not, contact Northgate support.

4. Create a new database using the **mkdbase** command - see *Set-up After Installation* in the appropriate installation guide. The new database must be large enough to hold the data transferred, as noted after saving the source database.

5. Restore any customised database configuration by integrating the contents of your saved configuration files with those in the directory/folder:

   DatabasePath/configs/config (UNIX)
   or
   DatabasePath\configs\config (Windows)

6. Log on to the SYSMAN account. Then load the FILE-SAVE tape and ensure that it is online.

7. Enter the following sequence of commands to restore your system/database:

   ```
   ASSIGN =TAPE n          n = tape unit No.
   T-REW
   T-FWD
   T-RDLBL 1
   T-FWD
   ACCOUNT-RESTORE * (O
   ```

8. If upgrading from a version earlier than Reality V9.1, load the system tools. Enter the following sequence of commands:

   ```
   T-DEVICE n $REALROOT/files/upgfile.rti      n = tape unit No.
   ASSIGN = TAPE n
   T-REW
   INSTALL
   ```

   Follow the prompts to install the upgrade bootstrap. Then enter

   ```
   CLEAR-ASSIGN
   ```

9. Run **SYS-UPDATE**, responding to the prompts as described in *SYS-UPDATE Details* on page 8.

10. Integrate any customised system file items that were saved from the source system/database.

11. Carry out **FILE-SAVE** and **VERIFY-SAVE**.

12. Start Transaction Processing, if required.

13. Enable logons (use **ENABLE-LOGONS** from TCL, or the **unlockdbase** host command).

SYS-UPDATE Details

When you run SYS-UPDATE, you are prompted to enter the release of Reality from which you are transferring.

The following prompt is then displayed:

```
Restore from a different machine type? (Y/N) :
```

At this prompt enter **Y** if you are transferring from a host with a different binary format; otherwise, enter **N**. Reality hosts have the following binary formats:

Byte normal: Solaris, AIX.

Byte reversed: Windows, Linux.

For example, if transferring data from Solaris to AIX, enter **N**; if transferring from AIX to Linux, enter **Y**; and so on.

DataBasic Conversion

During SYS-UPDATE, cataloged DataBasic programs in POINTER-FILE are upgraded if necessary. Accounts BASIC.CONVERSION and UPGRADE.ACCOUNT are populated by this process.

If the transfer is from a release prior to RealityX 4.0 or ROS 7.2, the DataBasic object code is converted. If the SYS-UPDATE process cannot determine the byte order of the source system the following prompt is displayed:

```
Unable to determine BYTE ORDER of item to be upgraded.
Please enter - 'R' if from a Byte Reversed, ie RX, NT or DGUX system.
          or - 'N' if from a Byte Normal system.
'S' will suppress this message and log item(s) with error code 16.
```

If you see this prompt, enter a response based on the table of binary formats shown above.

Note: If you do not enter a response, or enter **S**, you must run the following from the SYSMAN account after the upgrade:

```
:RUN SYSBP UPGRADE.BASIC.OBJECT
```

and enter whether the system is Byte Normal or Reversed at that time.

The format of basic object code changed on release 4.x. All cataloged Basic programs are checked and converted to the 4.x format if possible. The converter attempts to find the original object item (**$***item-id*) that was used to produce the cataloged image

and verify that the compilation date and times match. If successful, the converter converts the original object item to 4.x format and recatalogs it. If the correct object cannot be found, the pointer file item is 'decataloged', converted and recataloged.

An account 'BASIC.CONVERSION' is created to maintain a record of the conversion process and allow any problems to be investigated. In the account are the following three files:

CONVERTER.LOG

An entry is made in this file for each basic program item found in pointer file that needed conversion. The error codes in attribute 3 are decoded through a data section 'XLATE', but fall into three groups. Any codes between 1 and 9 are represent successful conversions with informational messages. Codes 10 through 20 represent failures to convert with associated messages. Any other codes correspond to errors produced by the CONVERT.OBJECT or CATALOG verbs.

PRIOR.PF

A copy of the pointer file item is placed in this file prior to conversion.

PRIOR.OBJ

A copy of the $ item is placed here prior to conversion, but with the same item-id as the pointer file item.

The command

```
LIST CONVERTER.LOG PROBLEMS
```

lists any items that failed to convert. The main reason for conversion failure is prior corruption of the pointer file item that has gone unnoticed because the item is no longer used. After resolving any errors the account should be saved to tape and may be deleted from the system.

Changing Lower- to Upper-Case Dates

From RealityX 5.0 onwards, Reality has used mixed-case month names. If you have older applications that rely on having all upper case month names for date verification, you will have to force dates into upper case. There are three ways of doing this:

- For each user that needs uppercase dates, set the UCASEDATES and UCASEDAYS options in their operating environment.

- Add the following commands to the LOGON program for each account that needs uppercase dates:

```
SET-OPTION UCASEDATES
SET-OPTION UCASEDAYS
```

- Configure Reality to use only upper-case dates:

  1. Logto the DENAT account on the database. The following menu is displayed:

```
                    MAIN MENU


        1. Language table maintenance.
        2. Error message file maintenance.
        3. Character tables maintenance.
        4. List file names.
        5. Make alphabet
        6. Exit to TCL.
        7. Logoff.

            Enter selection:
```

2. Selection option 6 (Exit to TCL).

3. Enter:

```
:ED ENGLISH CLASS0
Top
.MU87-98
.FI
'CLASS0' filed in file 'ENGLISH'.
```

4. Then enter:

```
:LOAD-LANG ENGLISH CLASS0
Language number or name:0
[9100] 'L0' loaded.
: OFF
```

The next time a logon or process is performed the dates will all be in uppercase.