NUTIL

Navan Utilities for IBM i

Version 7 Release 7

January, 2024

http://www.navan.co.uk

NUTIL - Navan Utilities for the iSeries

TABLE OF CONTENTS

COPYRIGHT

VERSION 7 RELEASE 7, MODIFICATION 0

This edition applies to Version 7 Release 7, Modification level 0 of the NUTIL Navan Utilities Program Product.

Unless otherwise stated, all pages in this edition are to Version 7, Release 7 and refer to the iSeries Operating System, Version 7.3 or greater.

NUTIL Version 7 Release 7 Modification level 0 is certified to be operationally compliant with IBM Power Systems servers running the IBM i Operating System; V7R4

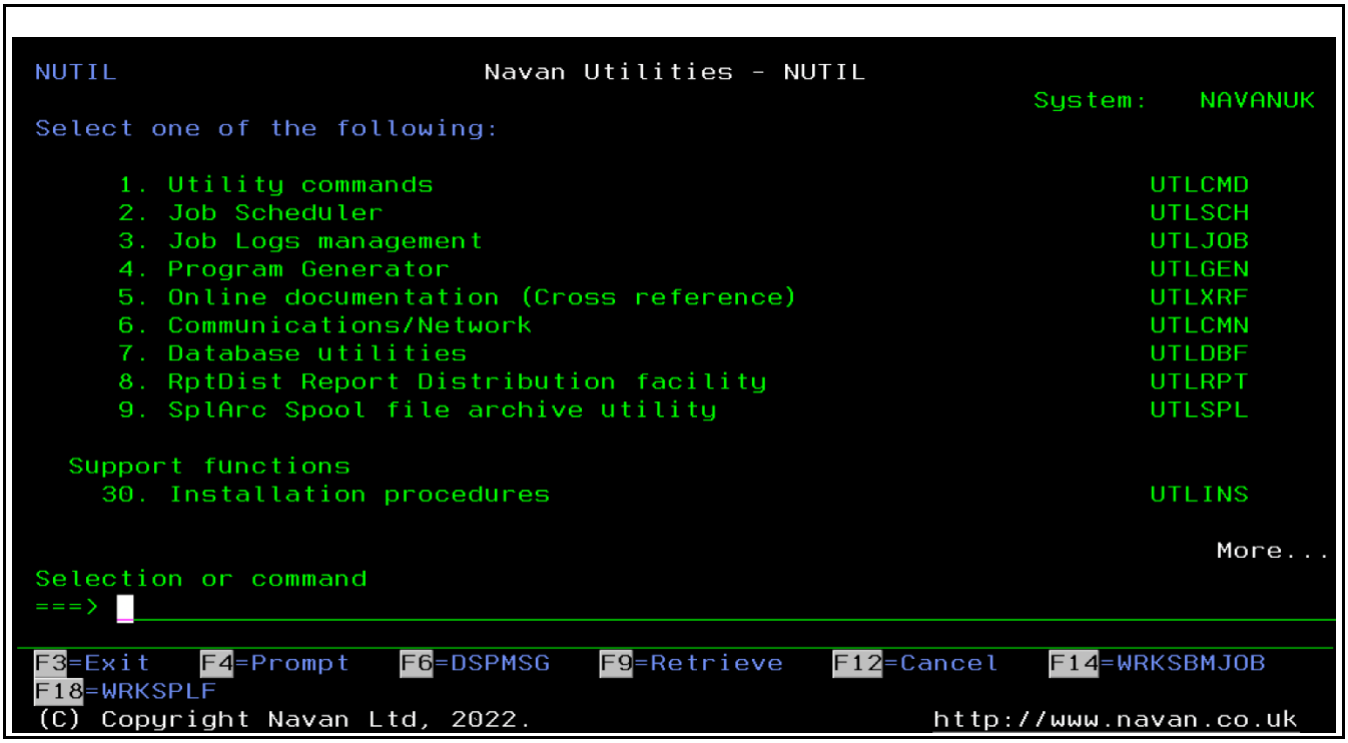
Any references in this manual to Navan Utilities or NUTIL refer to standard Navan Utility products.

Information on these products can be obtained from your nearest Navan agent.

The NUTIL Main Menu

All major functions within NUTIL can be accessed via the NUTIL main menu.

```
===> GO NUTIL/NUTIL
```



NUTIL Installation Procedures

General

Please read this section of the document thoroughly before commencing the installation of NUTIL.

All steps in this installation guide must be performed using the System Security Officer user profile, or by using a user profile assigned *ALLOBJ special authority.

Certain programs in NUTIL adopt authority when in use. This authority adoption must be retained if the system is to operate as intended.

Installing NUTIL

The distribution media supplied contains a complete (replacement) NUTIL library and can be installed automatically. If you already have an old version of NUTIL on your iSeries, your data will be loaded from the old version into your new version, by the installation command. Installation is performed as follows:

1. If already installed, ensure that no users are using any NUTIL functions. Ensure that the SCHEDULER subsystem is not currently active. Ensure that Report Distribution is not currently active. No NUTIL activity is allowed whilst this installation is being performed.
2. Sign on as the System Security Officer QSECOFR, or as a user profile that has *ALLOBJ special authority. This installation MUST be performed by a user with *ALLOBJ authority for all programs to function as designed.
3. If NUTIL is in your library list, remove it:

```
RMVLIBLE NUTIL
```

4. Save a copy of your existing version of the NUTIL library (if you have it already installed):

```
SAVLIB LIB(NUTIL) ...
```

You are now ready to commence the installation. The installation routine can be run by one of the following four methods:

**** INSTALLING FROM CD-ROM DISK ****

5(CD). Load the supplied CD-ROM disk into the iSeries CD drive and install the new version of the NUTIL library using the LODRUN command:

```
LODRUN      DEV(*OPT)
```

The supplied CD can also be used to extract all NUTIL documentation. Documentation is stored in the \DOCS directory of the CD and can be accessed using the CD-ROM drive of your PC.

**** INSTALLING FROM .ISO IMAGE FILE ****

5(iso). NUTIL can be installed from a supplied .iso image file in one of two ways:

- The image can be used to burn a CD, and then the CD can used to install the product
- The image file can be installed in the system image catalog and installed directly from there.

Please refer to separate documentation relating to the correct processing for these tasks.

** INSTALLING FROM TAPE **

5(tape). Load the supplied tape onto the tape drive and install the new version of the NUTIL library using the LODRUN command:

```
LODRUN DEV(TAP01)
```

If you are using a tape device other than TAP01, alter the DEVice parameter in the above command accordingly.

** INSTALLING FROM SAVEFILES **

5(savf). You should have a library called NAVANSAV on your system, which contains the following objects:

QINSTNUTIL	The installation program
NUTILSV	Save file containing the NUTIL library objects
NUDTAARASV	Save file containing the NUTIL data area

Install the new version of the NUTIL library using the installation program:

```
CALL PGM(NAVANSAV/QINSTNUTIL) PARM(*SAVF)
```

A pre-check will be made to ensure the objects specified above exist. If they do not, an error message will be sent and the program will end.

6. The procedure will then install NUTIL. At the completion of the procedure you should receive the message

'NUTIL installation procedure completed successfully'

If you do not get this message, the installation procedure was processed incorrectly. You can determine the cause of the error by reviewing the low level messages for the job. If you cannot resolve the problem, you should contact your NUTIL support representative for assistance. You will not be able to continue the installation of this version of NUTIL and you should reinstall your old version of NUTIL.

7. The NUTIL library objects are supplied in compressed form. You should give consideration to decompressing the objects in order to improve user access times. Refer to the CL Reference manual for a discussion of the Decompress Objects (DCPOBJ) command.

8. If you use the Program Generator to create programs for your user applications, you should make sure that your user messages file contains all of the 'run-time' messages used by generated programs. You can do this by using the merge command:

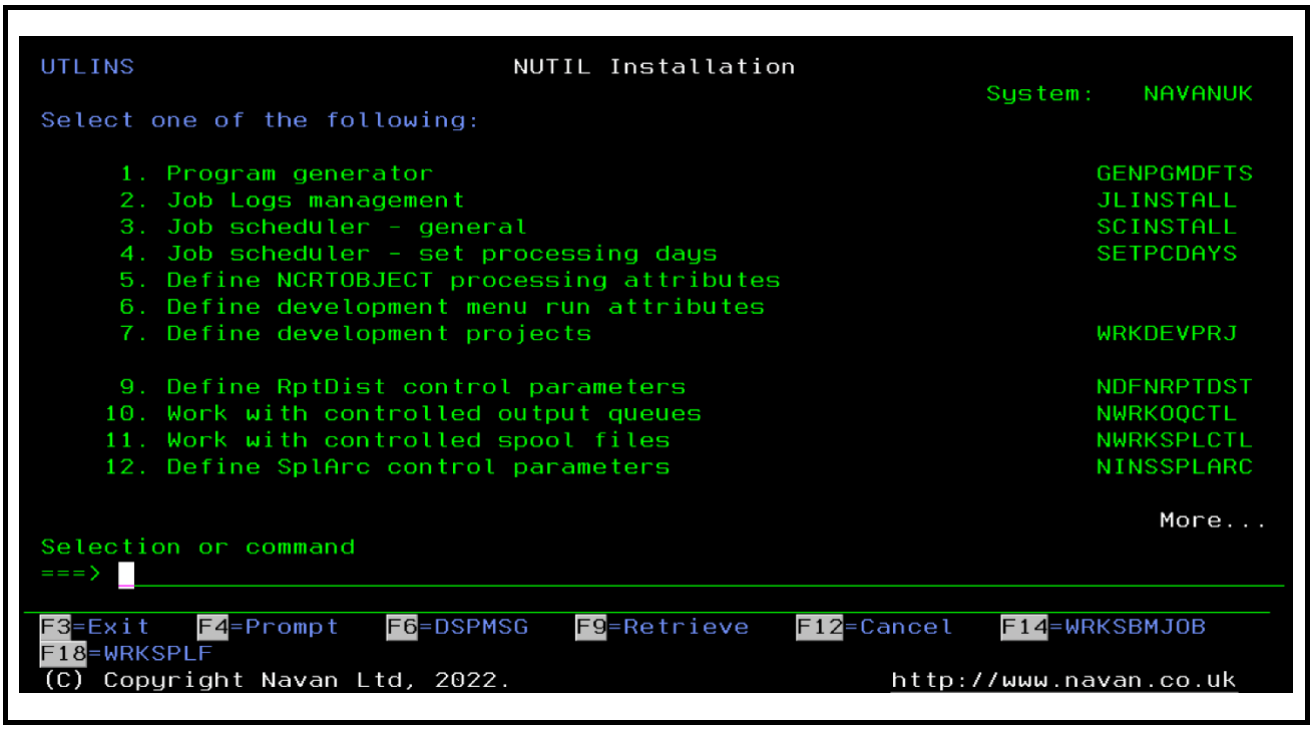
```
MRGMSGF MSGF(NUTIL/RUNMSGF) TOMSGF(userlib/usermsgf)SELECT(*ALL)
```

replacing the 'to message file' parameter entries with your own user message files. You will need to run this MRGMSGF command for each of your user message files.

Run the setup procedures

All set-up procedures can be accessed from the UTLINS menu:

```
GO NUTIL/UTLINS
```



You should first take option 20 and verify the attributes shown for your installation of NUTIL. If the customer number shown is 705027, this means that you have a demonstration copy of NUTIL.

An 'unlock code' is supplied by Navan upon payment of the license fee for using the product. This code can be used in option 30 of this menu to install your product license information into your copy of NUTIL. If you do not have your new 'unlock' code you should contact your Navan representative as soon as possible (you have 3 months use of NUTIL from date of initial installation on a specific iSeries before the product will expire. After that time you must have entered the unlock code to continue using the product).

Next, you should ensure the NUTIL reports will be printed correctly. Refer to the section entitled 'NUTIL reports', later in this manual. If you need to adjust the page settings for printed output you should use option 15 of the UTLINS menu.

GENPGM - Program Generator

If you are intending to use the Program Generator, the Set-up procedure for GENPGM should be run in order to define your installation requirements.

```
                GENPGM System Defaults (GENPGMDFTS)

Type choices, press Enter.

        Date format to display . . . . . DATFMT         *SYSVAL_
        Convert ILE to Eval  . . . . . . CVTEVAL        *YES
        DSPF source mbr name suffix  . . DSPF           D
        Standard Display width . . . . . DSPWDT         80_
        PRTF source mbr name suffix  . . PRTF           P
        Standard Printer file width  . . PRTWDT         132
        CLP source mbr name suffix . . . CLP            C
        Edit code for 8 digit dates  . . EDTDATE        8____
        Edit mask for date display . . . EDTMASK        *YMD
```

Parameter Definitions

The *Date Format* parameter lets you alter the way 6 and 7 digit dates will be displayed and entered in generated programs. The program generator assumes that dates are stored in Year/Month/Day format, but the way they are displayed is dependent upon this parameter.

The *Convert ILE to Eval* parameter defines whether ILE RPG program source will use old "fixed format" RPG Operation Codes, or the equivalent EVAL free-format Operation Code. The entry made here will become the default option shown to a user processing the GENPGM command.

The *DSPF Source member name suffix* parameter defines the character that will be appended to the program name in order to define a related Display file. The default is ‘D’, but the entry can be any letter of the alphabet *provided that* it is not the same as the letter defined for the Printer file suffix name.

> As an example, if the default is used, when the program generator creates a program named GL0001, a related display file will be named GL0001D.

The *Standard display width* parameter defines the maximum allowable length of a display line. The default is 80 but can be changed. When the program generator attempts to generate a subfile it will use this parameter when determining the number of fields it can fit into the subfile line.

The *PRTF Source member name suffix* parameter defines the character that will be appended to the program name in order to define a related Printer file. The default is ‘P’, but the entry can be any letter of the alphabet *provided that* it is not the same as the letter defined for the Display file suffix name.

> As an example, if the default is used, when the program generator creates a program named GL0001, a related printer file will be named GL0001P.

The *Standard printer line width* parameter defines the maximum allowable length of a printed detail line. The default is 132, but can be changed. When the program generator attempts to generate a printer file detail line definition it will use this parameter when determining the number of fields it can fit onto the line.

The *CLP Source member name suffix* parameter defines the character that will be appended to the program name in order to define a related CL program. The default is 'C', but the entry can be any letter of the alphabet.

As an example, if the default is used, when the program generator creates a program named GL0001, a related CL program will be named GL0001C.

The *Edit code for 8 digit dates* (EDTDATE) parameter defines which user-defined edit code will be used to define 8 digit dates. Program Generator support for 8 digit dates is based on the code you enter here.

The specified edit code will be initialised by this command to the edit mask defined in the INTMASK parameter.

The default is edit code 8. If you specify *IGNORE, no Program Generator support of 8 digit dates will be installed.

The *Edit mask for 8 digit dates* (INTMASK) parameter defines the character string that will be used to edit the date field for display purposes. The mask should conform to the date format that your dates are stored in the database file. The value entered here is used to initialise the system *user defined edit description* on your iSeries.

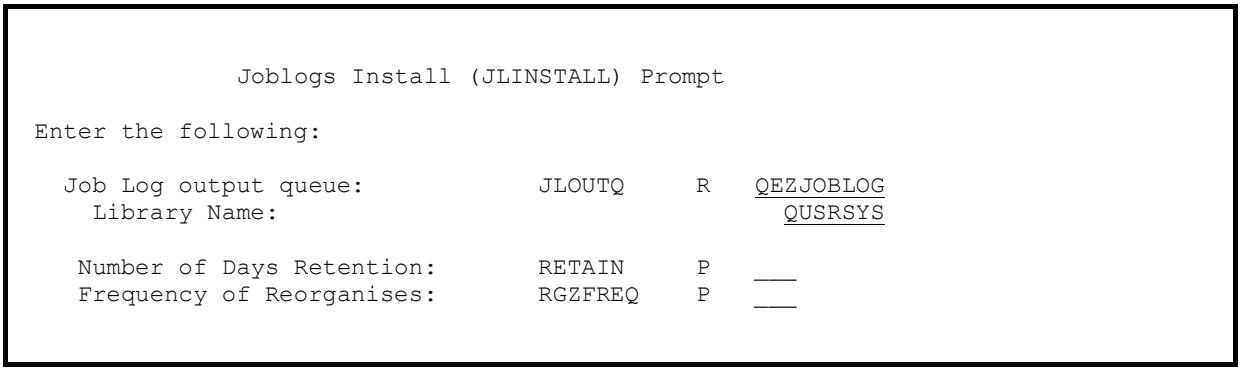
The default value is *YMD, which will display the date as nnnn/nn/nn. Other values are:

*ISO	nnnn-nn-nn
*EUR	nnnn.nn.nn
*USA	nn/nn/nnnn
*JIS	nnnn-nn-nn
*UK	nn/nn/nnnn

The specified edit code number will be used to create the specified user-defined edit description. Please ensure that you are not already using this user-defined edit code for any other purpose. Use the WRKEDTD command to confirm this before proceeding.

Joblog Management

The Set-up procedure for JOBLOGS must be run in order to prepare the product for use. This performed using the JLINSTALL command:



Parameter Definitions

The JLOUTQ (joblog output queue) parameter defines which output queue you have assigned to retain joblogs (when the iSeries is first installed this is defined as PRT01, which should be changed). If the specified joblog output queue does not currently exist this install procedure will create it and attach Printer file QPJOBLOG to it.

> The default queue is QUSRSYS/QEZJOBLOG, which is the queue that *will* be assigned (by OS/400) if you use the Operational Assistant function of OS/400.

The RETAIN (number of days retention) parameter tells the system how long (in days) a joblog is to remain in JOBLOGS before it is automatically deleted. If this entry is zero, all joblogs are retained indefinitely.

The RGZFREQ (frequency of reorganises) parameter tells the application how often (in days) its maintenance routine is to check for obsolete joblogs. If this entry is zero, no test is performed for expired joblogs.

The utility can be changed (using the JLINSTALL command) at any time if you wish to change these parameters at a later date.

Scheduler - Job Scheduler

The install procedure must be run in order to activate the product. This is performed using the SCINSTALL command:

```
              Create Customer SCHEDULER (SCINSTALL) Prompt

Type choices, press Enter:

  Customer Name:                    NAME        'Company_name'_______________
  Date Format:                      DATFMT      *JOB___

  Allow autostart?:                 AUTOSTART   *YES
  Load schedule at start-up?:       AUTOLOAD    *YES
```

The NAME (Customer Name) parameter allows you to enter your company name to 'customise' your Scheduler displays and reports.

The DATFMT (date format) parameter defines how SCHEDULER will format dates for display. The valid entries are:

*JOB	Display them using your job definition
*SYSVAL	Display them using your iSeries system value
*MDY	Display in Month/Day/Year format
*DMY	Display in Day/Month/Year format
*YMD	Display in Year/Month/Day format

The default is *JOB.

The AUTOSTART (Allow Autostart?) parameter is a way of controlling whether scheduler is automatically started by the iSeries system start-up procedure. This needs to be programmed into your iSeries. For a detailed explanation of this please refer to the section 'Start-up methods' later in this manual. You should initially set this parameter to *YES.

The AUTOLOAD (Allow Autoload?) parameter defines whether the job schedule should be loaded when the Job Scheduler is started. This should be installed using *YES value. The parameter should only be changed to *NO if you are performing a special restart of the Scheduler subsystem.

This command can be processed as often as you wish, if you need to change the values you have entered at any time.

Scheduler start-up methods

Scheduler is designed as an automatic system and cannot be considered operational until it has an automatic start-up method implemented on your iSeries.

The SCHEDULER subsystem is designed to run continuously, or at least all of the time the iSeries is powered on and in normal operational mode. For this reason we suggest that a SCHEDULER start-up function be included in your system start-up procedures, in one of two ways:

Method 1 - RECOMMENDED - As an entry in your system start-up procedure. All iSeries installations have a start-up procedure (specified in the QSTRUPPGM system value), that starts all subsystems, printers etc.

Just include the STRSCH (Start SCHEDULER) command in this procedure. This will probably be the preferred method, since you retain control of when the SCHEDULER is operational. An example of a recommended system start-up procedure is included in the NUTIL shell source file (NUTIL/QSHLSRC400, member QSTRUP). The code that needs to be added to your start-up routine is as follows:

```
            DCL        VAR(&SCAJB)      TYPE(*CHAR) LEN(1)


/*============================================================================*/
/*  ...Retrieve the Scheduler Autostart flag to see whether the               */
/*      Job Scheduler should be started...                                    */
/*============================================================================*/
            RTVDTAARA  DTAARA(SCHDTA (127 1)) RTNVAR(&SCAJB)
            MONMSG     MSGID(CPF0000 MCH0000) EXEC(CHGVAR +
                         VAR(&SCAJB) VALUE(Y))


            IF         COND(&SCAJB *NE N) THEN(NUTIL/STRSCH +
                         RESET(*OLD) JOBD(NUTIL/SCHEDULER) +
                         JOBQ(NUTIL/SCHASYNC))
            ELSE       CMD(SNDPGMMSG MSGID(SCH0071) MSGF(NUMSGF) +
                         TOMSGQ(*SYSOPR) MSGTYPE(*DIAG))
```

To determine the name of your current start-up program, use the WRKSYSVAL command to view the system value QSTRUPPGM.

Method 2 - As an Autostart Job. You can add an autostart job entry into your controlling subsystem description, so that when the iSeries powers up, SCHEDULER starts up also. This is performed using the ADDAJE (Add Autostart Job Entry) and ADDRTGE (Add Routing Entry) commands as follows:

```
ADDAJE  SBSD(QCTL) JOB(SCHAJB) JOBD(NUTIL/SCHAJB)
ADDRTGE SBSD(QCTL) SEQNBR(100) CMPVAL('SCHAJB') PGM(NUTIL/SCHAJB)
```

So now when OS/400 starts QCTL, QCTL will, in turn, start SCHEDULER.

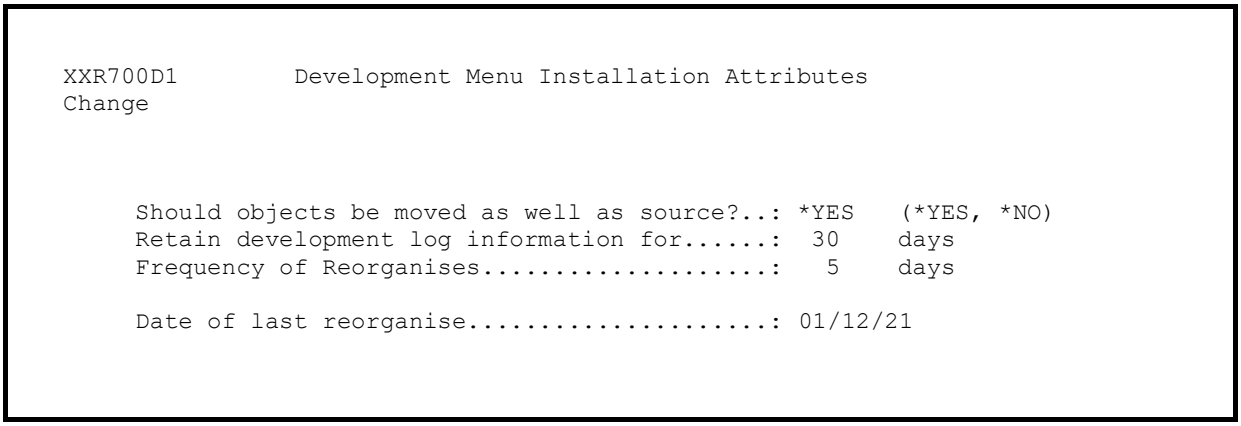
Source code for the Scheduler autostart program, SCHAJB, is retrievable (via the RTVCLSRC Command) if you wish to modify it for your own installation.

Note that you cannot add autostart job entries to an active subsystem. So the only way of implementing this procedure is by creating an alternate controlling subsystem. The method of doing this is described in the OS/400 Programmers guide.

This method can not be used if you are running your iSeries via the QBASE controlling subsystem.

Programmer Development Menu

The ‘define installation attributes’ procedure must be run in order to activate the function. This is performed using option 6 of the UTLINS menu:



Parameter Definitions

The MOVOBJ (move objects?) parameter determines how the movements between development, test and production will occur.

If you answer *YES, an attempt will be made to move the corresponding object as well as the source for development options 52 (move development to test) and 53 (move test to production). The movement of source will be dependent upon the successful move of the object - if the object cannot be moved, then the source move will not be attempted.

If the member attribute is PF or LF, the corresponding object move will not be attempted. This is to ensure the integrity of data objects. If you move a physical or logical file via option 52 or 53 you are only moving the source member. You must then manually recreate the object in the new environment.

The RETAIN (development log retention days) parameter tells the system how long (in days) to keep the Development log information before it is automatically deleted. If this entry is zero, all log records are retained indefinitely.

The FREQUENCY (frequency of reorganises) parameter tells the system how often (in days) its maintenance routine is to check for obsolete log data. If this entry is zero, no test is performed for old information.

These attributes can be altered at any time by re-accessing the menu option.

Development Projects

All development via the Development facility is grouped into *projects*. You define these projects via menu option 7, which processes the WRKDEVPRJ (Work with Development Projects) command. At least one development project must be defined before you can use the Development menu.

The definition of a project is entirely at your discretion. For example, it could be a new application being developed, a new function being developed or an existing application being corrected/upgraded.

The WRKDEVPRJ command has no parameters. After pressing Enter, it will show a list of currently defined projects. If no projects have yet been defined, you will be prompted straight into Project Add mode:

```
XXR701D2              Development Control - Maintenance
Add

  Project Name.......: ACC_INTER
  Project Description: New Accounts Interface

System Account Code: 3916-2214A
Project Status.....: A             Project Manager....:  Sam Snyde
Release Level......: 1             Release Date.......:    0-00-00
Project Start Date.:    151120     Project End Date...:    0-00-00

Library Control         Data            Object           Source
   Master                                                ACCTSMSTR
   Production           ACCTSFILE       ACCTSPGM         ACCTSSRC
   Test                 ACCTSTEST       ACCTSTEST        ACCTSTEST
   Development          ACCTSDEV        ACCTSDEV         ACCTSDEV
   Archive                                               SRCARCHIVE
   PTF/Upgrade                                           ACCTSPTF

Job Description Control
   Production        ACCTSFILE /ACCOUNTS
   Test              ACCTSTEST /ACCOUNTS
   Development       ACCTSDEV  /ACCOUNTS

F3=Exit   F11=Delete   F12=Previous   F16=*SRCF Overrides   F21=Command line
```

From this panel you define the controls for the project. The entries you can make are as follows:

Project Status: Status must be ‘A’ (project active) for work to be allowed on the project. Other valid entries are ‘S’ (project suspended) and ‘C’ (project complete).

> *The status **must** be set to ‘A’ for the project to be available via the Programmer Development Facility Menu.*

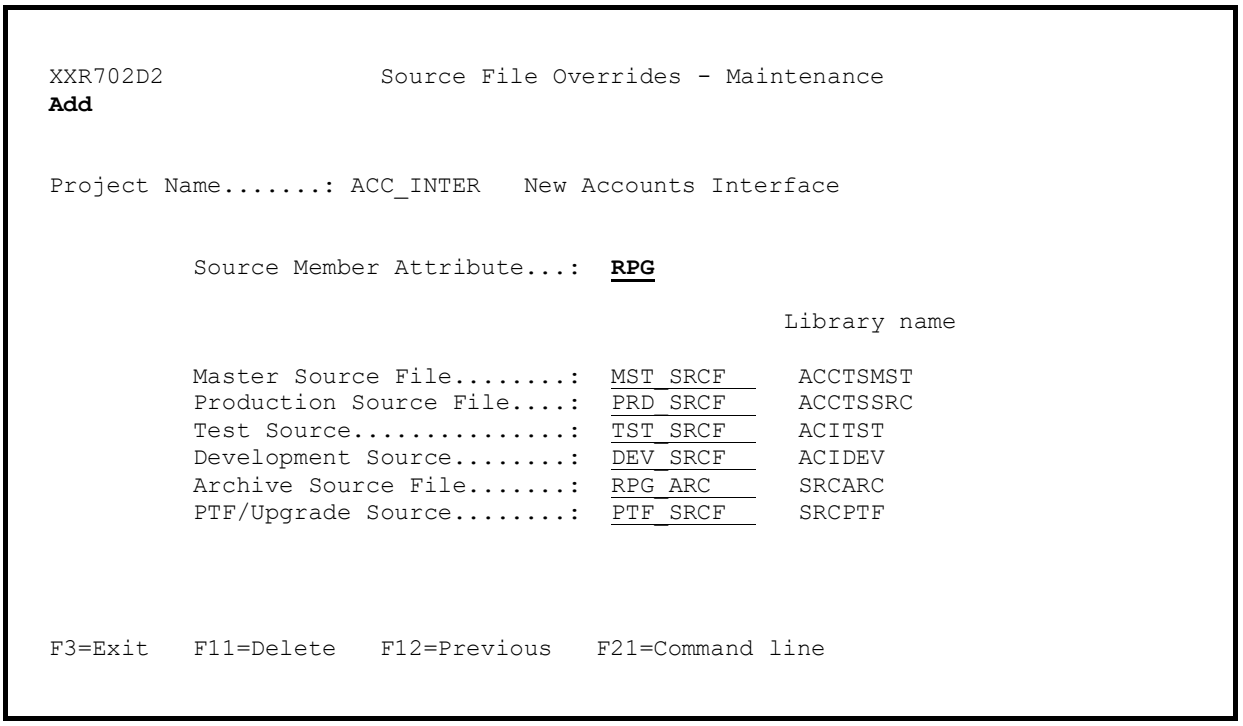
System Account Code: If an entry is made here, any work performed via the development facility menu will be charged to this account number in OS/400 Job Accounting.

Library Control Data: This defines the libraries to be used for this project. The same library name can be used for one or more of the entries (for example, the Production Source and Production Object libraries could be the same library). The ARCHIVE and PTF are optional libraries:

> PTF libraries are not yet supported (future development for NUTIL).
> The ARCHIVE library is optional - If one is entered it will be processed in options 53 and 54 of the Programmer Development Facility menu. If you do not require an archive library, you may enter *NONE for the library name.

Job Description Control: This defines the job descriptions that will be used for any relevant job submissions (object creations, etc).

Also available is a Source File override panel, which is accessed via the F16 function key. Information on source file overrides is only required if you are not using the standard source file naming standard - QCLSRC, QCMDSRC, QRPGSRC etc:

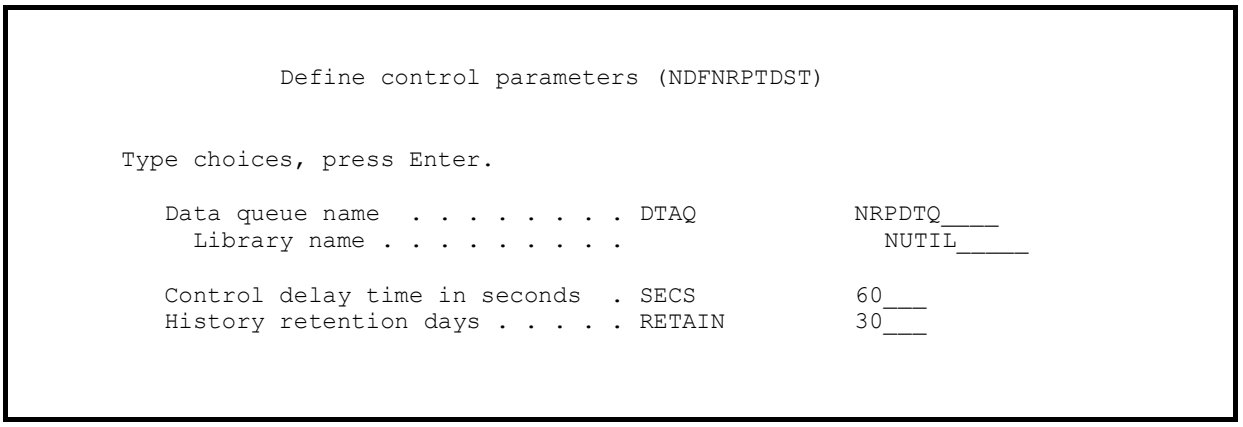


For each of the defined libraries within the project, enter the name of the source file that will be used for this source member type. The entry will be checked to ensure the source file exists in the specified library.

RptDist - Report Distribution

The NDFNRPTDST command is used to initialise the control parameters that are used to run the RptDist distribution control interrogator job.

You cannot start report distribution processing until these parameters have been defined.



Parameter Definitions

DTAQ (data queue name): Specify the name of the controlling data queue. It is advisable to use the default shown unless it conflicts with your own object names.

> The controlling data queue is an internal object that the system will use to determine what spool files are available for processing.

SECS (control delay time): Specify the number of seconds the interrogator job will normally wait before scanning controlled output queues for available spool files.

RETAIN (history retention days): Specify the number of days history you wish to keep of the successful report distributions.

Work with controlled output queues

The NWRKOQCTL command is used to define which output queues on your system that you want the RptDist interrogator job to control. Any spool files arriving on an output queue that is defined here will be eligible for automatic report distribution.

Please refer to *'Work with controlled output queues'* later in this manual for instructions on how to define your control entries.

Work with controlled spool files

The NWRKSPLCTL command is used to define which spool files on your controlled output queues that you want the RptDist interrogator job to control. Any spool files defined here arriving on a controlled output queue will be eligible for automatic report distribution.

Distribution of spool files can be controlled in 4 ways:

- You can send spool files via SNADS (SNS Distribution Services) to any user on your network
- You can send spool files via TCP/IP to any printer queue on a remote system known to your iSeries
- You can send spool files to another output queue on your local iSeries
- You can send spool files (as email .pdf attachments) to internet email addresses.

You can use any, or all, of these options to control your distributions. It is possible to have one original spool file that is distributed to many network users, many remote systems and many output queues. You do, however, need to define the distributions that you require.

The RptDist interrogator job will only process spool file names that have been defined in the spool file control list. You do this via the NWRKSPLCTL command and define in the TYPE parameter which subsection of the spool file table you will work with.

You can only work with (define) one subsection at a time:

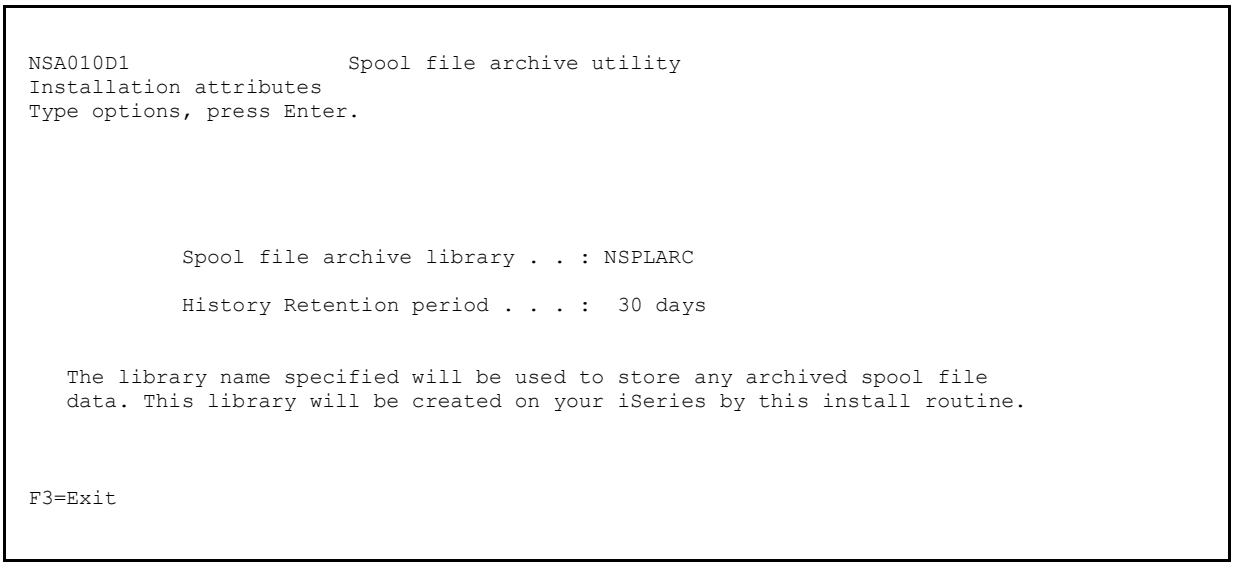
*NETFUSER specifies that you will be defining distributions to be made via SNADS to network users
*TCPUSER specifies that you will be defining distributions to be made via TCP/IP to printer queues on remote systems
*OUTQ specifies that you will be defining distributions to be made to other output queues on your local system
*EMAIL specifies that you will be defining distributions to be sent via email.

Please refer to *'Work with controlled spool files'* later in this manual for instructions on how to define your control entries.

SplArc - Spool file archive utility

The NINSSPLARC command installs the Spool File Archive Utility onto your machine. NINSSPLARC -must- be run before you attempt to use the SplArc utility.

The command has no parameters. When you press enter, the 'install options' work panel will be shown.



The installation parameters are as follows:

Spool file archive library: SplArc will create 'user space' objects to store the image and attributes of your archived spool files. These objects will reside in the library name that you specify here. It is highly recommended that the library name you specify not be used for any purpose other than archiving spool files. The default name is NSPLARC but you can alter it to whatever you choose. If the library name does not already exist on your iSeries it will be created by the installation command.

History retention period: This is the number of days that the spool file image will remain in the archive before it becomes eligible for removal by the SplArc cleanse procedure.

Installation Notes

Security Auditors note

The NUTIL installation procedure is performed by the QINSTAPP installation program, which can be found at SEQNBR 1 on the supplied tape. Should you need to review the installation procedure for security purposes you can do this as follows:

```
CRTSRCPF  FILE(QTEMP/QCLSRC)
RSTOBJ    OBJ(QINSTAPP) DEV(TAP01) SAVLIB(QTEMP) +
            OBJTYPE(*PGM) RSTLIB(QTEMP)

RTVCLSRC  PGM(QTEMP/QINSTAPP) SRCFILE(QTEMP/QCLSRC)
```

The source for the installation program is then in file QTEMP/QCLSRC, member QINSTAPP. Should you have any security objections to this installation procedure, please advise Navan directly.

It is possible (but not recommended) to install NUTIL using a modified version of the retrieved CL source but in doing so *you remove any liability from Navan* for the successful functioning of the installation procedure as well as the successful functioning of the NUTIL program product.

Effect of user modifications on warranty

Any warranty implied or expressed by Navan Ltd relating to this product does not apply to any portion of the product altered by someone other than an authorised Navan Ltd employee.

In other words, user modifications to the NUTIL utility library are not covered by any Navan Ltd warranty or support agreement.

NUTIL reports

As supplied, all NUTIL printed output will be to a page size of 66 lines, with page overflow occurring at line 60. Should you wish to alter this to conform to a different standard you should use option 10 of the UTLINS menu, specifying your own definitions as required.

Object compression

The NUTIL library objects are supplied in compressed form. You should give consideration to decompressing the objects in order to improve user access times. Refer to the CL Reference manual for a discussion of the Decompress Objects (DCPOBJ) command.

Object ownership

Unless there are special processing considerations, objects in NUTIL are owned by QPGMR and are freely accessible to all users on the system (all users have free public access). Where a command performs an operation that is not normally allowable by a 'low' authority user, the program will adopt sufficient authority in order to ensure the function works correctly. All NUTIL command objects are set to ALWLMTUSR(*NO), so that limited capability users can not request them from the command line on a menu.

Operating System configuration

NUTIL is designed to work with a standard iSeries operating system ("i5/OS"), as supplied by IBM. Should any local changes have been applied to any portion of this standard operating system, there is no guarantee of correct operation of the NUTIL system.

Such local changes to i5/OS include any alterations to system object authorisations; changing command parameter defaults; changing command processing environment defaults.

NUTIL internal security checking

Certain functions within NUTIL are 'locked' to the licensing information provided with your copy of NUTIL. Internal system checks within NUTIL functions are made against the following data:

- CPU Serial Number
- CPU Model Number
- System Date

This is to ensure your copy is an authorised version of NUTIL. You should be aware that if any of the CPU control information on your iSeries changes, or you attempt to use the version licensed to your CPU on another iSeries, functions within NUTIL may cease to work.

Your licensing information is displayable via the UTLINS menu option 20 - Display Installation Attributes.

Should you be upgrading your existing CPU to another model, or moving on to another iSeries completely, you will need to contact your Navan representative for an upgrade authorisation, which may be subject to additional licensing fees.

Demonstration versions of NUTIL will automatically cease to function when the expiry date has been reached.

By installing a demonstration version of NUTIL, you are accepting that Navan and its representatives take no responsibility for data lost within NUTIL if your demonstration copy of NUTIL reaches its expiry date, or if an internal security violation is detected by NUTIL.

NUTIL Programmer Environment

An Overview

The NUTIL Programmer Environment is a set of utilities designed to assist programmer development. It consists of the following:

An initial sign on to the environment
'Intelligent' compiler facilities
The capability of interrupting a currently running process to perform another function
A 'popup' programmer function window that can be accessed via the attention key

Installation is very simple - all you have to do is change your user profile.

Installing the NUTIL Programmer Environment

Prior to using any of the NUTIL Programmer environment features, you must ensure that programmers can move objects into the QRPLOBJ (Replaced Objects) library, which is a standard library on your iSeries.

Please have your Security Officer alter the authorisations to this library if necessary.

To access the programmer environment, your user profile has to be changed to call the NUTIL initial program, as follows:

```
                          Change Profile (CHGPRF)

Type choices, press Enter.

  Initial program to call  . . . . INLPGM         inlpgm
    Library  . . . . . . . . . . .                  nutil
```

The Initial Program to call must be set to NUTIL/INLPGM. The user class must be *PGMR or higher. This environment is not available to a user profile with a class of *USER.

After changing the profile, sign off and sign on again using the profile. The first time the user profile is used with the NUTIL initial program it will create the following objects:

- A library with the same name as the User Profile
- A NUTIL programmer control data area in the new library, with the same name as the User Profile.
- An output queue in the new library, with the same name as the User Profile.
- A Job Description in the new library, with the same name as the User Profile.

Defining NUTIL Signon information - NCHGSIGNON

On the initial sign on request you will then be prompted to verify/alter your signon definition, as follows:

```
                    Retrieve SIGNON defaults (NRTVSIGNON)

Type choices, press Enter.

Data Area for SIGNON defaults. . DTAARA       > NEWPGMR
  Library name . . . . . .                    >   NEWPGMR
Initial menu . . . . . . . . . INLMNU         > *SIGNOFF
  Library name . . . . . .                    >
User initial program . . . . . USRINLPGM      > USRINLPGM
  Library name . . . . . .                    >   NUTIL
SETATN program . . . . . . . . SETATNPGM      > SETATN
  Library name . . . . . .                    >   NUTIL
Interactive Debug Mode?. . . . . SYSREQCMD     > *YES
Message queue delivery mode. . . DELVRY       > *NOTIFY
Log CL program commands? . . . . LOGCLPGM     > *YES
Initial library list . . . . . INLLIBL        > NEWPGMR
                                              > NUTIL
                                              > QTEMP
                                              > QGPL
                                              > QIDU
                                              > QOFC
                          + for more values   > QRPG
Source library for STRPGMMNU . . SRCLIB       > *LIBL
Object library for STRPGMMNU . . OBJLIB       > *DFT
Exit program for STRPGMMNU . . . EXITPGM      > NCRTOBJECT
  Library name . . . . . .                    >   NUTIL
Job description name . . . . . JOBD           > NEWPGMR
  Library name . . . . . .                    >   NEWPGMR
Output queue name. . . . . OUTQ           > NEWPGMR
  Library name . . . . . .                    >   NEWPGMR
Request data to call?. . . . . RQSDTA         > '*STRPGMMNU'
Cancel duplicate spool files . . CNLSPLF      > *YES
Current library name . . . . . CURLIB         > NEWPGMR
System Language library. . . . . SYSLNGLIB    > *NONE
User Language library. . . . . USRLNGLIB      > *NONE
Remote Command Environment . . . RMTCMD       > *NO
```

Parameter definitions for this command are as follows:

INLMNU: This is the name of the initial menu to be displayed AFTER the initial request data (see parameter RQSDTA) has been loaded and processed.

USRINLPGM: This is an optional feature that allows you to have the NUTIL Initial Program call a user defined program after it has completed it's initial setup functions.

An example of a User Initial program is included in NUTIL; program USRINLPGM will set the PRTTXT (print text) parameter for the job, based on the user profile and system name. Source for this program can be retrieved using the RTVCLSRC command.

SETATNPGM: This is the name of the program to be called when the attention key is pressed. The default program is SETATN, which is provided in the NUTIL library.

SYSREQCMD: If set to *YES, this parameter allows you to process commands from the System Request menu. Details of this function are provided later in this manual.

DELVRY: This is the mode in which messages are to be presented to your job.

LOGCLPGM: This parameter determines whether CL program logging is to be switched on (CL logging will echo all CL program lines that have been processed to the users job log).

INLLIBL: The initial definition of this parameter is determined by the system value QUSRLIBL. Library NUTIL will then automatically be placed just above library QTEMP.

Note that QTEMP ***must*** be defined as one of the libraries in your system value QUSRLIBL. You can alter this library list to include your own libraries as required.

SRCLIB: This is the name of the source library you normally work in.

OBJLIB: This is the name of the object library you normally compile objects into.

EXITPGM: This is the name of the Programmer Menu exit program. The default, NUTIL/NCRTOBJECT, should be left as the entry (unless you have your own programmer exit program).

JOBD: This is the name of the Job Description that will be used by the NUTIL initial program to set your environment. The library list of this Job Description will always be set to whatever was defined in the INLLIBL parameter above.

OUTQ: This is the Output Queue that your printed output is normally sent to.

RQSDTA: This is your initial request data, which can be one of the following:

'*STRPGMMNU'	displays the IBM Programmer Menu
'*QCMD'	displays the command entry screen without installing attention key handling
'*SETATNOPN'	displays the command entry screen after installing attention key handling
'*QSYSOPR'	displays the SYSTEM menu panel
'*MAIN'	displays the MAIN menu panel
'*STRDEVMNU'	displays the NUTIL Programmer Development Menu
'*STRPDM'	starts the IBM Programming Development Manager
'*NONE'	No request data. Use this in conjunction with the Initial Menu (INLMNU) parameter
any command	causes the command to be processed and then display the command entry screen

The NUTIL Programmer Development menu is very similar to the IBM Programmer menu, except that is handles multiple versions of source and object libraries, so you can keep development and test environments separate from your production environment. Refer later in this manual for more details on STRDEVMNU.

CNLSPLF: This allows you to automatically remove spool files from old compile attempts from the output queue each time you recompile an object (controlled via the NCRTOBJECT exit program).

CURLIB: The name of the Current Library to use. This will default to the programmer library just created, but can be any valid library name or *CRTDFT (use the current system default library).

SYSLNGLIB: The name of the IBM supplied System Language Library to use. An entry here will force that library to the top of the system portion of your library list. This is to provide for the inclusion of IBM National (secondary) language support.

USRLNGLIB: The name of a user generated Language Library to use. An entry here will force that library to the top of the user portion of your library list. This is to provide support for your own multiple language environment libraries.

REMOTE COMMAND ENVIRONMENT: If set to yes, this function allows the use of SNDRMTCMD to this job. Nutil Remote command environment allows you to send a command from any job to be processed at another job that has been enabled by this parameter. This facility is defined further in this manual.

The user signon information entered here can be maintained at any time by using the NCHGSIGNON (Change Signon Information) or NEDTSIGNON (Edit Signon Information) command. This will change the current session, and keep the changes for the next time you sign on to the system.

NUTIL Compiler Facilities

The NUTIL Programmer Exit Program

The NUTIL programmer exit program NCRTOBJECT is an extension of the standard IBM Programmers Menu that performs additional functions when option 3 is used from the menu (to create an object).

The exit program is driven via a control data area in NUTIL. This data area can be maintained by calling program NCO001 (which is also option 5 on menu UTLINS):

Any changes to this information will affect all users using the NUTIL precompiler facilities

```
NCO001D1        NCRTOBJECT control *DTAARA - Maintenance
Change

       NCRTOBJECT pgm in library....:  NUTIL       library name

       Compiler Options Start Stmt..:        1
       Compiler Options End Stmt....:       25

       Pattern for Create options...:  *Z:
       Pattern for Overrides opts...:  *O:
       Pattern for Pre-Compile opts.:  *B:
       Pattern for Post-Compile opts:  *A:

       SBMJOB Library List..........:  *JOBD       *JOBD  *CURRENT
       SBMJOB Output Queue..........:  *JOBD       *JOBD  *CURRENT

       Update Cross Reference?......:  *YES        *YES  *NO
       Process File Dependencies?...:  *PROMPT     *YES  *NO  *PROMPT
       Process Program Dependencies?:  *PROMPT     *YES  *NO  *PROMPT

F3=Exit
```

Parameter Definitions

NCRTOBJECT pgm in library: This parameter should be left as the default (NUTIL).

Compiler Options Start/End: The source member lines number range in which compiler directives might be found.

Patterns for options: Any source member line in the above range that commences with the pattern defined is considered a compiler directive.

Compiler directives are explained in more detail later in this manual.

SBMJOB library list: Defines whether any object creation requests will be attempted with the current users library list or using the library of the Job Description being used.

SBMJOB output queue: Whether to send any compiler printed output to the current user output queue, or to the output queue defined on the Job Description in use.

Update Cross Reference: If set to *YES, a successful compile will automatically cause the NUTIL cross reference database to be updated for this new object (only if the object library has already been defined to the NUTIL XREF Cross Reference facility).

Process File Dependencies: If set to *YES or *PROMPT, the request for the creation of a file will first check to see if any other files are dependent on this file. If the parameter is set to *YES, all dependent files will also be submitted for re-creation. If the parameter is set to *PROMPT, all dependent files will be displayed on a prompt screen before the submission.

Process Program Dependencies: If set to *YES or *PROMPT, the request for the creation of a file will first check the NUTIL Cross Reference database to see if any programs use this file. If the parameter is set to *YES, all dependent programs will be submitted for re-creation. If set to *PROMPT, all dependent programs will be displayed on a prompt screen before the submission.

Using the IBM Programmer Menu with the NCRTOBJECT exit program

When the Programmer menu starts up (if your request data was '*STRPGMMNU') the IBM Programmer Menu is processed via the STRPGMMNU command and the resulting display appears:

```
                          Programmer Menu

Select one of the following:

     1. Start Data File Utility
     2. Work with iSeries Query
     3. CALL NUTIL/NCRTOBJECT                 object name, type, pgm for CMD
     4. Call a program                        program name
     5. Run a command                         command
     6. Submit a job                          (job name), , ,(command)
     7. Go to a menu                          menu name
     8. Edit a source file member             (srcmbr), (type)
     9. Design display format using SDA       (srcmbr), ,(mode)
    90. Sign off                              (*nolist, *list)

Selection . . . . .   3           Parm . . . .   testpgm
Type  . . . . . .     RPG         Parm 2 . . .
Command . . . . . .

Source file . . . .   __________  Source library . . . . . .   NEWPGMR
Object library  . .   NEWPGMR     Job description  . . . . . .   NEWPGMR

F3=Exit        F4=Prompt          F6=Display messages   F10=Command entry
F12=Cancel     F14=Work with submitted jobs             F18=Work with output
```

As you can see, option 3 has changed to 'Call NUTIL/NCRTOBJECT'. Also, the Source/Object Libraries and Job Description parameters have been filled in.

Now that this is installed, there are a few extra facilities provided when using the 'Option 3 - create..' function, as follows:

- If any compiler directives exist in the source member, they are applied to the CRT... command (see the next section of this manual concerning compiler directives).
- Parm 2 parameter: If you enter a '?' here before taking option 3, the create command will be displayed for you before it is submitted. Also, the actual SBMJOB command will be prompted before it is processed. If you enter '*DIAG' when creating an RPG program, then the diagnostic listing will be produced, but the compiler will not be invoked.
- When the attribute is PF or PF38, a check will be made for file dependencies (logical views). If there are any then these will also be submitted for re-creation. The physical file does not get deleted at this point. It is processed in the same way as a program in that it gets moved to QRPLOBJ when the creation of the new object completes successfully. If there is any data in the old version of the file it will be copied back into the new version of the file (using the FMTOPT(*MAP *DROP) parameter). If the file was previously journalled, then journalling of the new version will be re-established. Logical dependencies will only be re-created if the re-creation of the Physical file completes successfully.
- When the attribute is PF, PF38 LF or LF38, a check will be made in the NUTIL Cross Reference database for Program Dependencies. If there are any then these will also be submitted for re-creation. Program dependencies will only be submitted upon the successful creation of the file.
- For ILE source, the source member type that you specify in the "Type" field of the Programmer Menu display will determine what compiler is invoked when you use option 3. You can see the relationship between source member type and the related compile function by positioning your cursor in the "Type" field and pressing F1 to display the help text. As an example, if the Type is entered as "RPGMOD", option 3 will invoke the CRTRPGMOD compiler.

The submitted *object creation* job then performs the following:

- The compiler to invoke is determined, based on the source member type entered in the “Type” field of the Programmer Menu display.
- If any compiler overrides exist in the source member, they are processed and added to the compiler command.
- If any pre-compile directives exist, they are processed.
- If any spool files are still on an output queue from previous compile attempts, and if the CNLSPLF option on the users Signon definition is set to *YES, they are deleted.
- The object creation is attempted. If not successful the job ends.
- If successful, the following steps then occur:
- If the object was a physical file, the old version is moved into library QRPLOBJ and the data from the old file is mapped back into the new version. If journalling was established on the old object, it is detached from the old object and applied to the new object.
- If the NUTIL Cross Reference has been generated for the object library in which this object exists, it will be updated for this object.
- If the object was a physical or logical file, the WRKDBF file attribute database is updated.
- If any post-compile directives exist, they are processed.
- If the object was a physical or logical file and file or program dependencies need to be processed, these are then submitted. Each re-creation is a separate job submitted to the same job queue with the same job description. The JOBPTY (job queue priority) of each submitted job is altered to ensure files are created before programs are processed. These dependencies are processed in exactly the same way as outlined above.

Creating an object from within PDM

The ‘CO’ option from the Work with Members panel of PDM will access the NPDMCRTOBJ command:

```
                NCRTOBJECT access via PDM (NPDMCRTOBJ)

Type choices, press Enter.

Member Name: . . . . . . . . . . MBRNAM       > @CHKDAT
Source File Name:  . . . . . . . SRCFILE      > QCLSRC
  Library Name:  . . . . . . . .              >   NEWPGMR
Object Library:  . . . . . . . . OBJLIB       > NEWPGMR
JOBD to use: . . . . . . . . . . JOBD         > *USRPRF
  Library Name:  . . . . . . . .                *LIBL
```

Parameter Definitions

OBJLIB: This will default to the Source Library name.

JOBD to use: This will default to *USRPRF, which will attempt to use a job description with the same name as the users User Profile.

Note that file and program dependencies are not processed when using the NPDMCRTOBJ command

The submitted job processes in every other way the same as described for the NCRTOBJECT exit program.

Creating an object from within WDSc

WDSc can be configured to access the NWDSCRTOBJ command:

```
                NCRTOBJECT access via WDSc (NWDSCRTOBJ)

Type choices, press Enter.

Object library . . . . . . . . . OBJLIB         @CHKDAT
Source File  . . . . . . . . . . SRCFILE        QCLLESRC
  Library name . . . . . . . . .                 NEWPGMR
Source member name . . . . . . . SRCMBR         NEWPGMR
Option . . . . . . . . . . . . . OPTION         *EVENTF
```

Parameter Definitions

OBJLIB: This will default to the Source Library name.

OPTION: This specifies options to use when the source member is compiled. An Event File should normally be created when you create a module or program from within WDSc. WDSc uses this file to provide error feedback integrated with the WDSc editor.

Note that file and program dependencies are not processed when using the NWDSCRTOBJ command

The submitted job processes in every other way the same as described for the NCRTOBJECT exit program.

Using the generic Object Creation facility

In addition to the facilities provided to re-create objects one at a time there is also the NCRTOBJ Object Creation facility, which allows the processing of generic re-creations:

```
                    Object Creation Facility (NCRTOBJ)

Type choices, press Enter.

Source file name:  . . . . . . . SRCFILE
  Library name:  . . . . . . . .                  *LIBL
Member name: . . . . . . . . . . MBR              *ALL
Object library name: . . . . . . OBJLIB           *SRCLIB
Select member attribute: . . . . ATTR             *ALL
Create option: . . . . . . . . . OPTION           *REPLACE
Job description name:  . . . . . JOBD             *USRPRF
  Library name:  . . . . . . . .                  *LIBL
Output queue name: . . . . . . . OUTQ             *CURRENT
  Library name:  . . . . . . . .                  *LIBL

                           Additional Parameters

Last change date . . . . . . . . CHGDATE          *ALL
Authority  . . . . . . . . . . . AUT              *LIBCRTAUT
```

Parameter Definitions

SRCFILE: The name of the source file to process.

MBR: The member name to process. That can be a specific member, a generic request (for example EG* will process all source members whose names commence with 'EG') or *ALL, which will process all source members in the source file. Where only a single member is to be processed, you MUST enter the specific member attribute (eg *DSPF, *PRTF) in the ATTR parameter below, otherwise the specific member will not be processed.

OBJLIB: The object library to create the objects into. The default of *SRCLIB means that the created objects will be loaded into the library defined above in the *SRCFILE parameter.

ATTR: The source member attribute to process. This provides a filter so that only certain source member types will be processed. For example, 'DSPF38' will only process those source members in the source file where their source attribute is DSPF38. An entry of *ALL will process all members in the generic request.

> *Note:* Where the request is for processing a single source member, you must enter the specific member type here (eg *DSPF, *PRTF).

OPTION: The default of *REPLACE will replace an object of the same name and object type if one already exists in the object library specified. If you specify *CREATE, existing objects will not be replaced. Objects will only be created if one previously did not exist in the object library. Option *DELETE is not frequently used, but it allows you to delete an existing object in the object library without creating a new one.

JOBD: The job Description that will be used for the SBMJOB command for each creation request. The default of *USRPRF will attempt to use a job description with the same name as the users User Profile.

OUTQ: The output queue for the object creation listings.

CHGDATE: This parameter provides a filter to allow you to select only source members that have been changed since a specified date. If you specify a date, the 'Member Last Changed' date on the source member will be compared to this date. Only source members with a change date greater than, or equal to, the date you specify will be selected. If you do not specify a date, all source members (selected by previous parameters) will be selected.

AUT: The authority parameter allows you to specify the default authority for the created object(s). You should refer to the AUT parameter information in the CL Reference Manual for information on the values allowed. The default is *LIBCRTAUT.

Note that file and program dependencies are not processed when using the NCRTOBJ command

Each member to be processed will be submitted as a separate job. Each submitted job processes in every way the same (except regarding dependencies) as described for the NCRTOBJECT exit program.

Source Directives for the Compile Pre-Processor

Instructions for the compile pre-processor may be imbedded at the commencement of a source member using special types of comment lines. Each directive or override may span multiple source lines, connected by a '+' at the end of each line. These instructions are referred to as 'source directives'.

Types of Source Directives

Source directives can be of four types:

Type	Supplied 'Pattern'
Compiler directive	'*Z: '
Pre Compile override	'*O: '
Pre Compile directive	'*B: '
Post Compile directive	'*A: '

They are identified within the source member by a 4 character source code control directive pattern.

The source directive control code patterns shown in this document are the default patterns (shown above) as supplied with NUTIL. They may be different at your installation, as they can be altered using menu UTLINS, option 5.

Compiler Directives

Compiler Directives specify extra parameters to be added to the relevant CRT... command. For example:

```
... Using directives in RPG source
      ************************************************************
      Compiler Directives:
      *Z: CRTRPGPGM  OPTION(*CVTDATE)
      ************************************************************

... Using directives in ILE RPG source
      ************************************************************
      Compiler Directives:
      *Z: CRTBNDRPG OPTIMIZE(*NONE) CVTOPT(*NONE) DFTACTGRP(*NO) +
      *Z:     ACTGRP(*CALLER) BNDDIR(NBNDDIR) +
      *Z:     DBGVIEW(*ALL) OPTION(*NOEXPDDS *NOSHOWCPY *NOEVENTF)
      ************************************************************

  ... Using directives in DDS source
      ************************************************************
      Compiler Directives:
      *Z: CRTPF  MAXMBRS(1) SIZE(*NOMAX)
      ************************************************************

      ************************************************************
      Compiler Directives:
      *Z: CRTDSPF RSTDSP(*YES) DFRWRT(*NO) SHARE(*YES)
      ************************************************************

  ... Using directives in CL source
      /*********************************************************/
      /*  Compiler Directives:                                 */
      /*Z: CRTCLPGM  OPTIMIZE(*YES) LOG(*NO)                   */
      /*********************************************************/

  ... Using directives in CMD source
      /*********************************************************/
      /*  Compiler Directives                                  */
      /*Z: CRTCMD ALLOW(*INTERACT *IPGM) PRDLIB(ACCTSLIB) +     */
      /*Z:        PGM(MTHENDRUN) HLPPNLGRP(ACCTS) HLPPNLID(M1)  */
      /*********************************************************/
```

Note: **Only 1** compiler directive may exist within a source member and the maximum length of the directive is 300 characters. Use a plus (+) sign as a continuation character when a directive does not fit on a single line. The sign must be the last nonblank character within the line requiring continuation. A single directive can be continued for a maximum of 300 character positions in this way.

Compiler directives are loaded into the CRT... command prior to the re-creation job being submitted.

Pre-Compile Overrides

Pre-compile Overrides specify object overrides to be performed before the actual compile takes place. This allows you to compile a program without the actual object existing. This is particularly useful for an RPG program which double-defines a file, once as input and once as update. This avoids the requirement of having 'dummy' copies of a file to satisfy externally described files during a compilation.

Note: More than 1 pre-compile override may exist, and the maximum length of each override is 300 characters. Use a plus (+) sign as a continuation character when the override does not fit on a single line. The sign must be the last nonblank character within the line requiring continuation. A single override can be continued for a maximum of 300 character positions in this way.

```
*************************************************************
* Compiler Directives
*Z: CRTRPGPGM PUBAUT(*NONE) GENOPT(*OPTIMIZE)

* Compiler Overrides
*O: OVRDBF    FILE(@UPFILE) TOFILE(SYSMST00)
*O: OVRDBF    FILE(WRKDATA) TOFILE(SYSMST00)
*************************************************************
```

Pre-Compile overrides are processed *at the commencement* of the submitted re-creation job.

Pre-Compile Directives

Pre-compile Directives specify command requests to be executed before the actual compile is run. For instance, you can create objects in QTEMP necessary for the program to compile.

Note: More than 1 pre-compile directive may exist, and the maximum length of each directive is 2000 characters. Use a plus (+) sign as a continuation character when a directive does not fit on a single line. The sign must be the last nonblank character within the line requiring continuation. A single directive can be continued for a maximum of 2000 character positions in this way.

```
*************************************************************
* Compiler Directives
*Z: CRTRPGPGM PUBAUT(*NONE) GENOPT(*OPTIMIZE)
*
* Pre-Compile Processing
*B: CRTDUPOBJ OBJ(QADSPOBJ) FROMLIB(QSYS) OBJTYPE(*FILE) +
*B:             TOLIB(QTEMP)
*B: ADDLIBLE  USERLIB  *LAST
*************************************************************
```

Pre-Compile directives are processed *prior to* processing the relevant CRT... command.

Post-Compile Directives

Post-compile Directives specify command requests to be executed after the actual compile is run. For instance, you may want to grant object authority to a specific user or duplicate the object into another library.

Post-compile directives will only be processed if the compile was successful.

Note: More than 1 post-compile directive may exist, and the maximum length of each directive is 2000 characters. Use a plus (+) sign as a continuation character when a directive does not fit on a single line. The sign must be the last nonblank character within the line requiring continuation. A single directive can be continued for a maximum of 2000 character positions in this way.

```
*************************************************************
* Compiler Directives
*Z: CRTRPGPGM PUBAUT(*NONE)
*
* Pre-Compile Processing
*B: CRTDUPOBJ OBJ(QADSPOBJ) FROMLIB(QSYS) OBJTYPE(*FILE) +
*B:             TOLIB(QTEMP)
*
* Post-Compile Processing
*A: CHGPGM    PGM(TESTX) RMVOBS(*ALL)
*A: GRTOBJAUT OBJ(TESTX) OBJTYPE(*PGM) USER(QPGMR) AUT(*ALL)
*************************************************************
```

Post-Compile directives are processed *following* the successful completion of the relevant CRT... command.

If the CRT... command terminates abnormally, the Post-Compile directives are not processed.

The NCODFT Data Area

A control data area named NCODFT exists in library NUTIL and is used to hold the default patterns for the various directives/overrides. These can be altered to suit your installation by using the ‘define NCRTOBJECT processing attributes’ option on the UTLINS installation menu.

NUTIL facilities for PDM

Selected features of NUTIL are available to you when you use the IBM Program Development Manager (PDM). To use these, all you do is take F18 from the PDM main menu panel and change your defaults as follows:

```
                              Change Defaults

Type choices, press Enter.

  Object library . . . . . .   *CURLIB      Name, *CURLIB, *SRCLIB

  Save session defaults  . . .  Y           Y=Yes, N=No
  Save/Restore option  . . . .  1           1=Single, 2=All

  Job description  . . . . . .  *USRPRF     Name, *USRPRF, F4 for list
    Library  . . . . . .        __________  Name, *CURLIB, *LIBL

  Change type and text . . . .  Y           Y=Yes, N=No

  Option file  . . . . . .      QAUOOPT __  Name
    Library  . . . . . .        NUTIL ____  Name, *CURLIB, *LIBL
  Member . . . . . .            QAUOOPT __  Name
```

The two entries to change are the user ‘Option file’ (which ensures you use the NUTIL options file) and the ‘Job Description’ to be used (which ensures you use the correct default job description).

Work with members

The NUTIL options file allows you to perform additional options from the ‘Work with’ screens within PDM, as follows:

Option	*Description*
CM	Compare two source members
CO	Access the NPDMCRTOBJ compile processor
CP	Compress/Decompress a source member
CR	Convert RPG source to ILE RPG source
CS	Copy source member
DM	Use the DSPDBF command to display a member (multimember file)
EM	Use the EDTDBF command to edit a database member (multimember file)
ES	Use the EDTDBF command to edit a source member
LI	Start Specification Language Interpreter
LS	Create Program Specification
MM	Move a source member to another source file
MS	Merge two source members
RD	Scan and replace a specified string
RI	Print RPG Indented source listing
RL	Use the IBM RLU utility to print a report layout
RS	Reorganise a source member
SD	Scan for a specified string
SN	Send a Network file
SR	Start REXX procedure
UR	Update (Enhance) ILE RPG source

Note: The NPDMCRTOBJ command performs the same functions as the programmer exit program NCRTOBJECT, from a PDM selection screen

Work with objects

Option	Description
B	Use the DSPDBF command to browse (view) file data
C	Copy a file
E	Use the EDTDBF command to edit file data
I	Display object information
CF	Clear a physical file member
DF	Print the file Data Description layout
DR	Display a file record format
DU	Duplicate an object
MD	Edit a data area
MO	Move an object
OA	Display object authority
QR	Use the RUNQRY command to view the contents of a file
QM	Use the RUNQRY command to view the contents of a file member
RG	Reorganise a physical file member
SQ	Use the NRUNSQL command to process a file
ZF	Clear a physical file, all members

All PDM panels

Option	Description
DM	Display messages
WJ	Work with submitted jobs
WS	Work with spool files

Source Member types supported by the NUTIL Compiler facilities

The following source member types are supported by NUTIL Compiler functions; any other types are ignored.

Source member type		Command processed	Resultant object type	Attribute
CBL	COBOL	CRTCBLPGM	*PGM	CBL
CBLLE *	ILE COBOL/400	CRTCBLMOD	*MODULE	CBLLE
CBLLE *	ILE COBOL/400	CRTBNDCBL	*PGM	CBLLE
CBLMOD	ILE COBOL/400	CRTCBLMOD	*MODULE	CBLLE
CBLBND	ILE COBOL/400	CRTBNDCBL	*PGM	CBLLE
CBL38	COBOL System/38	CRTCBLPGM	*PGM	CBL38
CBL36	COBOL System/36	CRTS36CBL	*PGM	CBL36
CLP	Control Language	CRTCLPGM	*PGM	CLP
CLLE *	ILE Control Language	CRTCLMOD	*MODULE	CLLE
CLLE *	ILE Control Language	CRTBNDCL	*PGM	CLLE
CLMOD	ILE Control Language	CRTCLMOD	*MODULE	CLLE
CLBND	ILE Control Language	CRTBNDCL	*PGM	CLLE
CLP38	System/38 Control Language	CRTCLPGM	*PGM	CLP38
CMD	Command	CRTCMD	*CMD	
CMD38	System/38 Command	CRTCMD	*CMD	
DSPF	Display File	CRTDSPF	*FILE	DSPF
DSPF38	System/38 Display File	CRTDSPF	*FILE	DSPF38
DSPF36	System/36 Display File	CRTS36DSPF	*FILE	DSPF36
DTAARA	Data Area Field Definition	CRTPF	*FILE	PF
DTAQ	Data Queue Field Definition	CRTPF	*FILE	PF
FLDREF	Field Reference File	CRTPF	*FILE	PF
FORMAT	Field Definition	CRTPF	*FILE	PF
ICFF	ICF Communications File	CRTICFF	*FILE	ICFF
KLIST	Key List Field Definition	CRTPF	*FILE	PF
LF	Logical File	CRTLF	*FILE	LF
LF38	System/38 Logical File	CRTLF	*FILE	LF38
MENU	UIM Menu	CRTMNU	*MENU	UIM
MNU36	System/36 Menu	CRTS36MNU	*MENU	MNU36
MSGF36	System/36 Message File	CRTS36MSGF	*MSGF	MSGF36
PF	Physical File	CRTPF	*FILE	PF
PF38	System/38 Physical File	CRTPF	*FILE	PF38
PLIST	Parameter List Field Definition	CRTPF	*FILE	PF
PNLGRP	Panel Group	CRTPNLGRP	*PNLGRP	
PRTF	Printer File	CRTPRTF	*FILE	PRTF
PRTF38	System/38 Printer File	CRTPRTF	*FILE	PRTF38
QMQRY	QueryMgr Query	CRTQMQRY	*QMQRY	
QMFORM	QueryMgr Form	CRTQMFORM	*QMFORM	
RPG	RPG/400	CRTRPGPGM	*PGM	RPG
RPGLE *	ILE RPG/400	CRTRPGMOD	*MODULE	RPGLE
RPGLE *	ILE RPG/400	CRTBNDRPG	*PGM	RPGLE
RPGMOD	ILE RPG/400	CRTRPGMOD	*MODULE	RPGLE
RPGBND	ILE RPG/400	CRTBNDRPG	*PGM	RPGLE
RPG38	System/38 RPG	CRTRPGPGM	*PGM	RPG38
RPG36	System/36 RPG	CRTS36RPG	*PGM	RPG36
RPT	RPG Auto Report	CRTRPTPGM	*PGM	RPG
RPT38	System/38 RPG Auto Report	CRTRPTPGM	*PGM	RPG38
RPT36	System/36 RPG Auto Report	CRTS36RPT	*PGM	RPG36
SPADCT	Spelling Aid Dictionary	CRTSPADCT	*SPADCT	
SQLCBL	DB2/400 QueryMgr COBOL	CRTSQLCBL	*PGM	SQLCBL
SQLCBLLE	ILE DB2/400 QueryMgr COBOL	CRTSQLCBLI	*MODULE	SQLCBLLE
SQLRPG	DB2/400 QueryMgr RPG	CRTSQLRPG	*PGM	SQLRPG
SQLRPGLE	ILE DB2/400 RPG/400	CRTSQLRPGI	*MODULE	SQLRPGLE
SRVPGM	Service Program	CRTSRVPGM	*SRVPGM	
TBL	Table	CRTTBL	*TBL	
SPADCT	Spelling Aid Dictionary	CRTSPADCT	*SPADCT	
WSCST	Workstation Customising Object	CRTWSCST	*WSCST	

***NOTE:** For ILE source types CBLLE, CLLE and RPGLE, the compiler command processed will be dependent on the Compiler Directive in the source member. If the compiler directive specifies that a program is to be created (CRTCBLBND, CRTCLBND, CRTRPGBND) then *that compiler will be invoked*; otherwise the module compiler will be invoked by default.

Processing Commands from the System Request Menu

Overview

If the SYSREQCMD signon parameter is set to *YES, this allows you to process commands from the System Request menu. This is a very handy feature, especially for program debugging. Imagine that you have called a program interactively and after a few minutes you find a problem in the program, but you are not currently in 'debug' mode so you can't display program variables...

Using the SYSREQCMD environment, all you have to do is bring up the system request menu and enter the STRDBG command. Your program is then in debug mode and you can then start to debug it from that point (an example of this follows).

Any command that you are authorised to can be processed in this way. All you have to do is install SYSREQCMD in your job.

The SYSREQCMD command

Basically the function is very simple; when you set your initial program SYSREQCMD parameter to *YES, your initial program processes the NUTIL SYSREQCMD command.

This command can be used at any time to start the SYSREQCMD environment in a job - it does not have to be done from the NUTIL initial program:

```
                  SysRq Command Processor (SYSREQCMD)

Type choices, press Enter.

Message queue: . . . . . . . . . MSGQ            N
Library name:  . . . . . . . . . MSGQLIB         QTEMP
```

SYSREQCMD creates and processes a message queue (default name 'N' in library QTEMP). This message queue has a break message-handling program attached to it.

In order to process a command, all you do is send the command - as a message - to the message queue previously installed.

An example of entering a command at the System Request menu

To send a command to be processed *at any time* (and this includes if your screen is currently locked), all you do is access the System Request menu:

```
                                System Request
Select one of the following:

      1. Display sign on for alternative job
      2. End previous request
      3. Display current job
      4. Display messages
      5. Send a message
      6. Display system operator messages

     80. Disconnect job

     90. Sign off

 Selection  5
```

If you take option 5 you will be given the SNDMSG (Send Message) command prompt. Enter the command as the message and send it to the SYSREQCMD message queue previously defined.

An alternative way to do this is to enter the message on the system request line, which will be displayed at the bottom of your current screen if you press the System Request key WITHOUT pressing enter:

```
                                Command Entry
Type command, press Enter.

   > CALL ACCTS/MTHPROC PARM(*UPD)_____________________________________________
     ___________________________________________________________________________
     ___________________________________________________________________________

F3=Exit    F4=Prompt    F9=Retrieve    F10=Display detailed messages
F12=Cancel              F24=More keys

 5 'strdbg accts/mthproc updprod(*yes)' n                                    __
```

In the above example, the screen is currently locked due to the program ACCTS/MTHPROC. The system request key has been pressed and the request line appears at the bottom of the screen. When the Enter key is pressed the MTHPROC program will be put into debug mode (due to the STRDBG command that was entered on the above example).

The NUTIL Remote Command Facility

Overview

The Remote Command Facility is similar in concept to the SYSREQCMD function previously described, but where SYSREQCMD allows you to interrupt processing on your own screen the Remote Command Facility allows you to interrupt other 'enabled' workstations or jobs currently running on your iSeries (either local or remote, batch or interactive). The restrictions on using this function are that:

- You must have the Remote Command Facility enabled on your screen, either by setting your NUTIL Signon parameter RMTCMD to *YES or by using the STRRMTCMD command at your workstation.
- The 'remote' job you wish to access must have the Remote Command Facility enabled and you must be authorised to send commands to it.

If both of these criteria are met, you have the capability of entering a command at your screen that will be processed within another job.

Starting the Remote Command Environment

For each job that is to access the Remote Command Environment, the Start Remote Command Environment (STRRMTCMD) command must be entered from within that job:

```
                    Start RMTCMD Environment (STRRMTCMD)

Type choices, press Enter.

Users allowed: . . . . . . . . . ALWUSR      __________
                          + for more values  __________
```

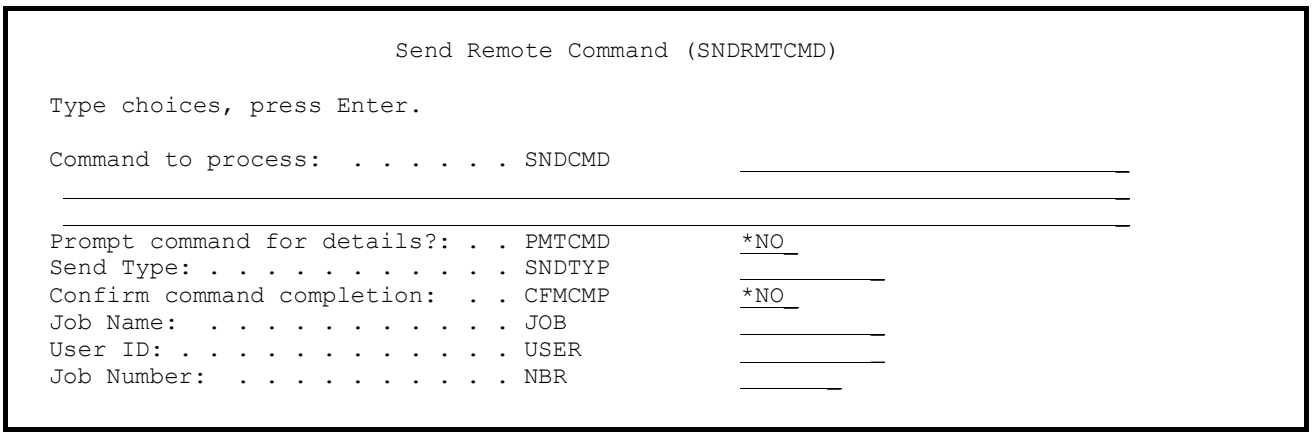
The command will install the Remote Command Environment in the requesting job and allow those users defined in the ALWUSR parameter to send commands to this job.

If your Initial Signon had RMTCMD(*YES) defined, the users allowed to send commands to your job are QSYSOPR, QSECOFR and yourself.

Remote Command Environment is automatically enabled in the NUTIL Scheduler utility for every job scheduled, as well as the actual Schedule Interrogator job. Users authorised are QSECOFR and QSYSOPR.

Sending a Remote Command

To send a Remote Command to another job you use the Send Remote Command (SNDRMTCMD) command:



Parameter Definitions

SNDCMD: The command that is to be processed from within the remote job. Any command that the remote job is authorised to perform can be entered. Remember that you are processing commands at the OTHER job, not this job. Also remember that if you are processing objects in library QTEMP, it is the target job's QTEMP library you are processing, not your own job's QTEMP library.

PMTCMD: If you wish to prompt the SNDCMD entered (to enter additional parameters before sending it), enter *YES.

SNDTYP: You can send the remote command to a Job Name, a User ID or a Job Number. If duplicate Job Names exist or the User ID currently has more than one job active you will be prompted to select the correct one.

If you enter *SELECT you will be given a selection panel showing all jobs currently enabled for Remote Commands.

If you enter *PRV the command will be sent to the job you last sent a Remote Command to.

CFMCMP: If *YES, you will be sent a message confirming the successful/unsuccessful processing of the command at the remote job.

JOB/USER/NUMBER: Enter the identifier for the send request

If you entered *JOB for the SNDTYP, you will prompted to enter the Job Name.
If you entered *USER for the SNDTYP you will be prompted to enter the User ID.
If you entered *JOBNBR for the SNDTYP you will be prompted to enter the Job Number.

The command will only be sent for processing if you are authorised to process commands at the requested job.

Once the command is sent, the remote job will be interrupted (regardless of what it is currently doing) and the command will be processed. Once the command has finished the remote job will continue from where it was interrupted. If you requested a confirmation message, you will then receive a message telling you whether the command completed successfully or not.

Ending the Remote Command Environment

To end the Remote Command Environment for a job, use the End Remote Command Environment (ENTRMTCMD) command. The command has no parameters.

Remote Command Environment cleanup

There is always the possibility that users have signed off without ending the remote command environment. This leaves unwanted information in the control files that should be removed.

To keep the application 'clean', you should use the Reorganise Remote Command Environment (RGZRMTCMD) command at the end of each day to remove any unattached control information. If you are using the NUTIL Scheduler function there is a Standard Job in the Scheduler master files (Scheduler job name RGZ_NUTIL) that will perform this task.

Example of using Remote Commands in a Batch Job

The Remote Command Environment is extremely useful in correcting problems in long running batch jobs. Lets say you are running your month-end processing run and for some reason a work file hasn't been created in the jobs QTEMP library and the job stops and sends a 'File not found' error message to the System Operator.

You can send a remote command to the job (assuming it has been enabled to receive them) to create the file. Then all you have to do is take the 'Retry' option to the System Operator's error message and the program continues processing.

Example of using Remote Commands in an Interactive Job

The Remote Command Environment can be extremely helpful to a 'Helpdesk' operator in helping to solve user problems. Rather than asking the user to perform any corrective actions, the actions can be performed by the helpdesk operator.

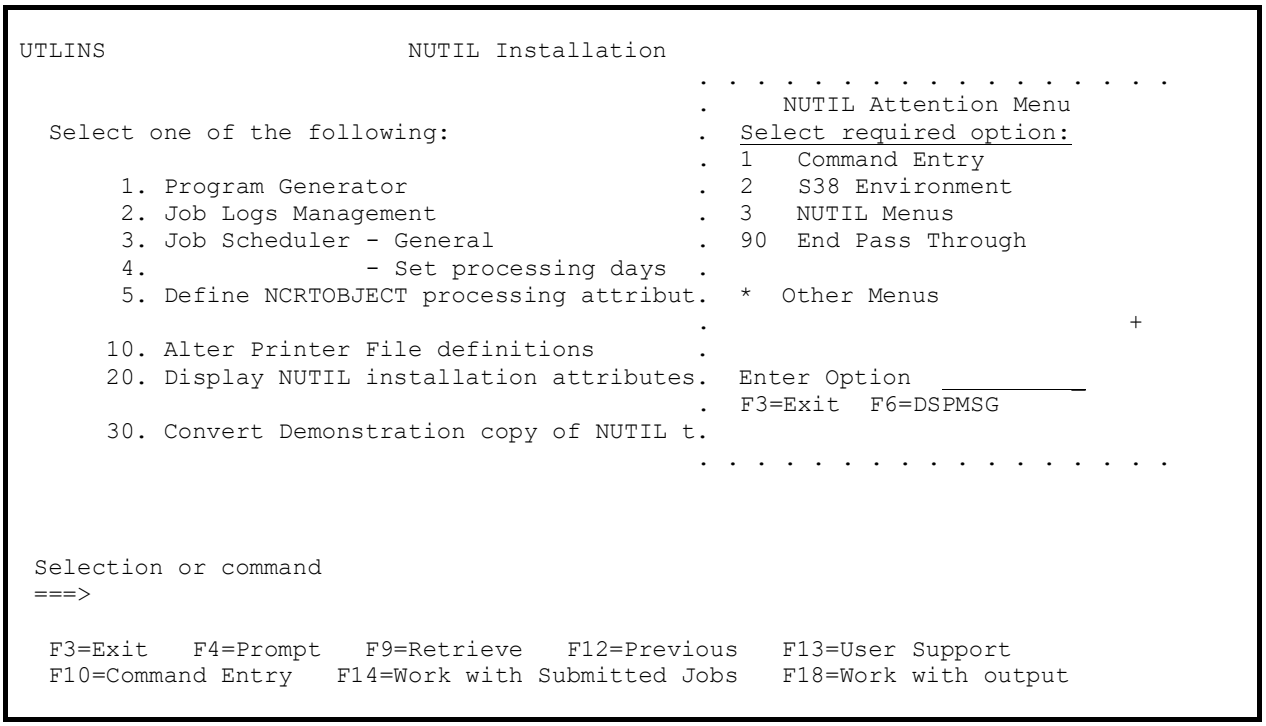
In conjunction with the IBM Start Copy Screen (STRCPYSCN) command the help desk operator can actually see what is happening at the remote workstation as well as performing any necessary corrective actions by using the SNDRMTCMD command.

The NUTIL Programmer 'Popup' menu

If your NUTIL Signon definition has the NUTIL/SETATN program specified in the SETATNPGM parameter, you have access to the NUTIL popup menus. These menus are stored in source file NUTIL/SETATN and can be updated or customised to your own requirements by using the IBM Source Entry Utility (SEU).

The SETATN processing program uses User Defined Data Streams to actually process the 'pop-up' facility, so that the menus will work correctly no matter what mode your screen is currently in (including OfficeVision/400).

As supplied there are three basic menus provided, and the initial menu displayed is the menu stored as member SETATN in source file NUTIL/SETATN:



The definition of 'where' the window actually pops up, and the size of the window is stored in the source member, along with all the processing functions the menu provides. The Page Up/Page Down keys are enabled where there are more options available for the menu than can be shown at one time.

Also included in NUTIL are two simple 'popup' programs:

'@CALC', which is a calculator program and can be accessed from option 'C' on the attention menu window or it can be called directly. Results from previous calculations are retained, but can be cleared using the F12= Reset function. You can exit the panel using F3.

'@CALD', which is a calendar program and can be accessed from option 'D' on the attention menu window or it can be called directly. The panel will initially display the calendar for the current month/year, but can be changed by entering in new values. You can exit the panel using F3.

Job Scheduler Utility

Overview

SCHEDULER is an integrated system designed to run on the IBM iSeries, executing requested jobs in a batch subsystem based on a predefined job schedule. With little or no user intervention, SCHEDULER will run your iSeries all day, every day. Accordingly you achieve:

- Reductions in overtime and manpower needs
- Improved response time

SCHEDULER allows you to:

- Pre-schedule a job to run any time on any day
- allow predefined jobs to run on specific dates
- allow predefined jobs to run on a specific day every week
- allow predefined jobs to run every day, week, month or year
- schedule jobs that are dependent on other jobs
- interface easily from user programs

All jobs that are scheduled to run are listed at the beginning of the day. This may be updated and reprinted at any time during the day. An audit trail may then be produced the next day that provides you with a list of what *did* happen. Any jobs that were not run (due to system powerdown or operator intervention) are also listed.

SCHEDULER runs in its own subsystem (not QBATCH), so that normal day to day operations are not affected by SCHEDULER jobs tying up your production subsystems.

SCHEDULER runs jobs that have been described in a job scheduler file. Each job is defined with time and schedule frequency. At predetermined time intervals, the job scheduler file is read to see if there are any jobs to be run at this time, on this day. If there are, the job is submitted for processing in the SCHEDULER subsystem.

Jobs may be defined in one of two ways:

- a standard job, that is to run on a regular basis (e.g., every day, week, month or year)
- a ‘once off’ job, that is to be run only once

A system data area is used to define and control what days of the week are available to the Scheduler. If a day is designated as ‘not available’, no automatic scheduling will be performed on that day.

A calendar file is used to define and control what calendar dates are not available to the Scheduler. Any dates so designated will still be processed, but automatic scheduling will not be performed (manual override is still allowed if the Scheduler job is running).

Using a command supplied with SCHEDULER, loading of non standard jobs is very similar to the standard SBMJOB (submit job) command, with additional parameters defining the scheduling requirement.

At any time of the day, the current job schedule may be viewed and maintained. Jobs may be added to the schedule, removed from the schedule, or rescheduled to a different start time. The maintenance display shows all jobs that are currently scheduled, allowing easy schedule maintenance.

Also, at any time, standard jobs can be added, maintained or deleted, by an easy to use maintenance display. This allows you to change the scheduling attributes, or to change the actual job itself. Any change to the day or time will affect the current days schedule of jobs immediately.

A job may include a ‘warning’ message to the System Operator, advising that the job is about to be processed. A ‘hold’ may then be placed on the job if it is not required to be run at that stage.

Preparing for Use

Run the Installation Command

First, you need to ensure that the library containing Scheduler is in your library list:

```
ADDLIBLE NUTIL *LAST
```

You can now run the Scheduler install command SCINSTALL, which defines basic control details for Scheduler operation:

```
                          Install SCHEDULER (SCINSTALL)

Type choices, press Enter.

Customer name  . . . . . . . . . NAME           *SAME_____________________

Date format  . . . . . . . . . . DATFMT         *SAME__
Allow autostart? . . . . . . . . AUTOSTART      *YES
Load schedule at start-up? . . . AUTOLOAD       *YES
```

Parameter Definitions

The entry you type in the NAME parameter will appear on screens and reports as your company name.

The way that dates will be shown on displays and reports is controlled by the DATFMT parameter. This is factory set to *JOB (the format is based on your job attributes) and should not need to be changed.

'Allow Autostart?:' is a way of controlling whether scheduler is automatically started by the iSeries system start-up procedure. This needs to be programmed into your iSeries. For a detailed explanation of this please refer to the section 'Start-up methods' later in this manual. You should initially set this parameter to *YES.

'Allow Autoload?:' defines whether the job schedule should be loaded when the Job Scheduler is started. This should be installed using the *YES value. The parameter should only be changed to *NO if you are performing a special restart of the Scheduler subsystem.

This command can be processed as often as you wish, if you need to change the values you have entered at any time.

Set the allowable processing days

You now tell SCHEDULER what days of the week it can perform automatic scheduling and how your processing calendar is defined. This is done using the SETPCDAYS command:

```
           Set SCHEDULER Process Days (SETPCDAYS) Prompt

Type choices, press Enter.

   Process on Mondays?:              PCMON      P    *YES
   Process on Tuesdays?:             PCTUE      P    *YES
   Process on Wednesdays?:           PCWED      P    *YES
   Process on Thursdays?:            PCTHU      P    *YES
   Process on Fridays?:              PCFRI      P    *YES
   Process on Saturdays?:            PCSAT      P    *YES
   Process on Sundays?:              PCSUN      P    *YES

   Number of periods in year:        PCPRD      P    12
   History Retention Days:           PCRTN      P    30
```

Use of the SETPCDAYS command requires the user to have *JOBCTL special authority.

As you can see, the command defaults to the following:

All days of the week are available for auto processing
There are 12 periods in the calendar year

You may change this if you wish, by setting the appropriate days to *YES (they are available) or *NO (they are not).

These parameters only define the days that standard jobs can be automatically scheduled. A non standard job can still be submitted and processed on a day specified as *NO. Also, Standard jobs can still be manually forced onto the job schedule.

The ‘Number of Periods’ parameter tells Scheduler whether you are using a 12 or 13 period calendar. It is important to set this correctly as RunCode 32 jobs (run on the last working day of the month/period) need this value to determine whether period end processing is due.

The ‘History Retention Days’ parameter defines how much history should be kept when a History Cleanse Processing run is performed.

Set the job logging levels

When SCHEDULER runs a job, it needs to know how much job log information you require. This is done using the CHGLOGLVL command:

```
          Alter Scheduler Logging Levels (CHGLOGLVL) Prompt

Type choices, press Enter.

    Log level,NORMAL completion:     NORMAL     P
      Logging Level (0-4):                        2
      Msg severity filter (00 99):                00
      Msg text level:                             *SECLVL
      Log CL program commands?:                   *NO_

    Log level,ABNORMAL completion:   ABNORMAL   P
      Logging Level (0-4):                        4
      Msg severity filter (00 99):                00
      Msg text level:                             *SECLVL
      Log CL program commands?:                   *YES
```

Use of the CHGLOGLVL command requires the user to have *JOBCTL special authority.

As you can see, the command defaults to the following:

Jobs ending normally will be set to	LOG(2 00 *SECLVL) LOGCLPGM(*NO)
Jobs ending abnormally will be set to	LOG(4 00 *SECLVL) LOGCLPGM(*YES)

You may change these defaults if you wish. Refer to the CHGJOB command in the iSeries CL Reference Guide for an explanation of these parameters.

Set the Scheduler Delay

Once SCHEDULER is running, it interrogates the job schedule at a frequency defined by the CHGSCHDLY command. This is factory set to 60 seconds, and at this stage we suggest you leave it that way. If you have previously been running NUTIL you may have already decided on a longer (or shorter) delay.

The CHGSCHDLY command is explained in detail under ‘Controlling the schedule interrogator’, later in this manual.

Use of the CHGSCHDLY command requires the user to have *JOBCTL special authority.

Set the Library Lists

An important consideration in SCHEDULER job processing is the library list on a Job Description that is to be used to run a job.

If used on a Scheduler job, a Job Description library list must conform to the following:

- It must contain every library the job will need
- It must contain library NUTIL
- It must contain library QTEMP

Multiple job descriptions can (and should) be used for this purpose. For example, your order entry job description may be

```
CRTJOBD JOBD(OELIB/OEJOBD) INLLIBL(OEFILE OELIB NUTIL QGPL QTEMP)
```

and your inventory management job description may be

```
CRTJOBD JOBD(IMLIB/IMJOBD) INLLIBL(IMFILE IMLIB NUTIL QGPL QTEMP)
```

In both cases, you can see that user libraries are defined first in the list, then the SCHEDULER library (NUTIL), then system 'general purpose' libraries.

Some notes on Job Descriptions

If library NUTIL does not exist on the library list of a job that is being processed by SCHEDULER, the library will be added in the executing job, POSITION(*LAST).

The job descriptions used to run any job in SCHEDULER should not be defined as USRPRF(*RQD). A valid user profile name should be used, which has sufficient authority to perform the scheduled job.

The SCHEDULER job description, for example, is defined with USRPRF(QSYSOPR). Of course, the user submitting the job to SCHEDULER also needs authority to use the job description.

To take advantage of the system message reply list, all batch job descriptions should specify INQMSGRPY(*SYSRPYL). Refer to the IBM iSeries Programmers Guide for an explanation of this topic.

Scheduler in operation

Starting the Scheduler subsystem

It is important that you read the Scheduler appendix topic ‘Start-up methods’ and implement at least one automatic start-up method for the Scheduler subsystem before commencing regular use of Scheduler.

Scheduler is designed as an automatic system and cannot be considered operational until it has an automatic start-up method implemented on your iSeries.

The Scheduler menu

SCHEDULER functions can be called from the SCHEDULER menu. To access this menu, enter GO NUTIL/UTLSCH, which will then display the menu as follows:

```
 UTLSCH                       Job Scheduler

 Select one of the following:

    1. Work with Standard jobs                          WRKSTDJOB
    2. Submit non standard job                          SBMSCHJOB
    3. Load jobs by Scheduler group code                LODSCH
    4. Force a standard job onto the schedule           FRCSCHJOB

 Daily operations
    5. Work with the current job schedule               WRKSCHJOB
    6. Work with scheduled job history                  DSPSCHHST

 Support functions
   10. Start SCHEDULER                                  STRSCH
   20. End SCHEDULER                                    ENDSCH
   25. Verify schedule extraction                       VFYSCHEXT
   26. Resubmit processing schedule                     RSMSCH

   30. Set installation attributes                      SCINSTALL
   31. Set processing days                              SETPCDAYS
   32. Change the scheduling time delay                 CHGSCHDLY
   33. Calendar file maintenance
   34. Parameter file maintenance

 Reporting functions
   40. Scheduled jobs report
   41. History log report
   42. Standard jobs listing

 Other options
   90. Sign off

Selection or command
===> ____________________________________________________________________

F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel   F14=WRKSBMJOB   F18=WRKSPLF
```

From this menu, all normal SCHEDULER functions can be performed.

Menu Option 1. *Standard jobs maintenance.*
This option allows you to add, change or delete standard jobs.

Menu Option 2. *Load Non-standard job.*
This option is used to load a ‘once-off’ request onto the scheduler.

Menu Option 3. *Load jobs by Group Code.*
This allows you to load a group of jobs grouped by a non date-related code.

Menu Option 4. *Force a standard job onto the schedule.*
This allows you to force a standard job onto the job schedule, bypassing the normal scheduling attributes of the job.

Menu Option 5. *Work with current job schedule.*
This option allows you to interrogate the job schedule.

Menu Option 6. *Work with job history.*
This option allows you to interrogate the log of jobs that were run by SCHEDULER.

Menu Option 10. *Start SCHEDULER*.
This option starts the Scheduler subsystem.

Menu Option 20. *End SCHEDULER*.
This option allows you to terminate the Scheduler subsystem.

Menu Option 25. *Verify schedule extraction.*
This allows you to print a planning list that shows the jobs that will be scheduled for a specific date, based on a date entered.

Menu Option 26. *Resubmit processing schedule*
This allows you select all, or part, of the processing schedule for a specified date, and force it on to the job schedule for another date.

Menu Option 30. *Set installation attributes.*
This allows you to define the company name to appear on reports, and the date format that will be used.

Menu Option 31. *Set processing days.*
This allows you to define the days of the week that are considered standard scheduler processing days.

Menu Option 32. *Change the scheduling timer delay.*
The timer delay is factory set to 60 seconds. This means that the scheduler subsystem will interrogate the job schedule every 60 seconds to see if any jobs are awaiting submission. You may change this delay to suit your own needs, if necessary.

Menu Option 33. *Calendar File Maintenance.*
Any dates that cannot be used for automatic processing by SCHEDULER must be loaded onto the calendar.

Menu Option 34. *Parameter file maintenance.*
Any codes that are used by the SCHEDULER system are defined in a parameter file. This option allows you to maintain those parameters.

Menu Option 40. *Scheduled Jobs report.*
This option lists the current job schedule.

Menu Option 41. *History Log report.*
This option lists the history log.

Menu Option 42. *Standard Jobs list.*
This option will print a list of all standard jobs defined within SCHEDULER.

Loading Jobs

There are four ways to load a job onto the scheduler, depending on the type of job you require loading.

- Where the job is to be run repeatedly, on a day RunCode basis
- Where the job is to be run repeatedly, on a non standard basis
- Where the job is to be run on specific calendar dates
- Where the job is a *one-off* job.

Standard jobs, run on a 'Day RunCode' basis

This is a standard job that is processed by a day-of-the-week, or day-of-the-month 'RunCode'. In this case, when the job is added to the file, you specify the day to run as one of the following codes:

RunCode	*Description*
1 - 31	Run on a specific day of the month (every month)
32	Run on the last working day of the month/period
33	Run on the last working day of the year
34	Run on the last working day of the week
35	Run on the first working day of the week
36	Run on the first working day of the month/period
37	Run on the first working day of the year
40	Run every time STRSCH is performed
41	Run every time the job schedule is interrogated
50	Run every day
51	Run every Monday
52	Run every Tuesday
53	Run every Wednesday
54	Run every Thursday
55	Run every Friday
56	Run every Saturday
57	Run every Sunday
58	Run on Mondays, Wednesdays and Fridays
59	Run on Tuesdays and Thursdays
60	Run Monday thru Friday
70	The job will only run using the job calendar, or by the FRCSCHJOB command

Day RunCodes are DATE related. The schedule loading program picks up the system date, determines what day of the week it is and any special attributes of the date (start of week, end of year, etc), and then selects jobs to be scheduled based on the RunCodes specified on each job.

Jobs loaded with a day RunCode are considered to be permanent jobs. These jobs can only be removed by you deleting the job via the Standard Jobs Maintenance program or, if a job end date is defined, automatically on expiry (a job end date tells SCHEDULER the job has expired and can automatically be deleted).

You can specify up to 5 day RunCodes on each standard job. Each one will be checked in the Scheduler job loading process to determine whether that job should be scheduled or not.

A description of the Standard Jobs Maintenance option appears later in this manual.

Standard jobs, run on a ‘Group Code’ basis

This is a standard job that is processed at a user defined frequency. We have set up 5 Group Codes in the parameter file, which are the more common non-standard job frequencies:

Group Code	Description
WK	Run Weekly
MT	Run Monthly
QT	Run Quarterly
SX	Run Six Monthly
YR	Run Yearly

You may create additional Group Codes (via parameter file maintenance) if you wish.

For instance, you may want to group and submit your housekeeping jobs together, in which case you may add a code ‘HK’ to the parameter file, then attach this Group Code to the jobs it relates to.

Jobs that are to be run using these Group Codes are manually loaded by the user, using the LODSCH command.

Group Codes are NOT date related. It is simply a means of grouping a common set of jobs together, which you may then submit in one step.

Jobs loaded with a Group Code are considered to be permanent jobs. These can only be removed by you deleting the job via the Standard Jobs Maintenance program or, if a job end date is defined, automatically on expiry (a job end date tells SCHEDULER the job has expired and can automatically be deleted).

A description of the Standard Jobs Maintenance option appears later in this manual.

LODSCH - The Load Group Code Jobs command

This command has been defined to allow you to load jobs onto the scheduler via the Group Code. The command prompt looks like this:

```
                Load Job Schedule (LODSCH) Prompt

Enter the following:

  Codes to be run  . . . . . . . . . GROUPCODE  P    __
                                     + for more      __
  Run Date for Jobs  . . . . . . . . RUNDATE    P    ________

  Processing for old jobs  . . . . . OLDJOBS    P    *IGNORE_
  User . . . . . . . . . . . . . . . USER       P    *JOBD_____
```

Use of the LODSCH command requires the user to have *JOBCTL special authority.

Parameter Definitions:

GROUPCODE: You may specify up to 5 Group Codes that are to be selected for processing.

RUN DATE: The date you enter will be the date that the jobs will be run. If you do not enter a date, today's date is assumed.

PROCESSING FOR OLD JOBS: This parameter allows you to determine what to do if a code job is found, but the submission time on the job is less than the current time. Normal scheduling logic is that jobs with a submission time less than the current time will not be considered for processing; this parameter allows you to override normal logic. This parameter only affects processing for a run date of *TODAY.

> If you specify *IGNORE, then any job with a schedule time less than the current time will be ignored by the LODSCH processing.
>
> If you specify *HOLD, then any job with a schedule time less than the current time will be submitted to the job schedule, but in a *HELD status. These held jobs will then need to be released manually (via the WRKSCHJOB command) before they will run. You will be notified if any jobs are submitted in held status.

USER: This specifies which user profile will be associated with the submitted job.

If you specify *JOBD (the default), the user will be taken from the USER parameter of the associated job description.

You can specify *CURRENT to use the same user profile as the currently running job.

All other run parameters are taken from the standard job definition.

The LODSCH command will then read through the standard jobs file and load any job containing one of the Group Codes entered will be scheduled for processing on the date entered.

If the job has already been loaded onto the job schedule for the rundate entered, the job will not be selected again, since jobs may only be loaded once for a specific date/time.

A report will also be printed, as follows:

```
Report: LODSCHP       Scheduler Code Jobs Submitted

Group
Job Name     Description             Code   Time        Notes
*************************************************************************
WRKACTJOB   Display Active Jobs       MO   12.15
SAVEOE      Save of OELIB             BK   23.00
SAVEIM      Save of Inventory Mgmt    BK   23.00
SAVEMK      Save of Marketing Data    BK   23.00    Job already on schedule
OEREPORTS   Periodical Reports OE     RP   23.30
*************************************************************************
       ** End Of Report **        4 Group Code jobs submitted
```

Report Notes:

Submit Date: This is the date the jobs have been submitted to.
Job Name: The name of the job scheduled
Group Code: The Group Code for which the job was selected
Time: The time the scheduler will attempt to process the job

Any job, which has already been loaded onto the current job schedule, will display the message ‘Job already on schedule’. Any job dependencies are only picked up where the dependency has been loaded onto the job schedule. This is because a job may be dependent on day to day running, but when it is run as a code job, the dependency is not required.

Standard jobs, run on a non-standard basis

There are many times when it is necessary to run a job at a time other than on its normal scheduling frequency. The facility exists to force a standard job onto the job schedule, via the FRCSCHJOB command.

FRCSCHJOB - The Force Standard Job command

This command has been defined to allow you to force a standard job onto the job schedule, regardless of its standard scheduling requirement. The command prompt looks like this:

```
                Force Scheduled Job (FRCSCHJOB) Prompt

Enter the following:

  Job Name . . . . . . . . . . . . . JOB        P    *SELECT___
  Submit on Date . . . . . . . . . . SBMDAT     P    _________
  Submit at Time . . . . . . . . . . SBMTIM     P    _________
  Job Dependency . . . . . . . . . . DEPJOB     P    *IGNORE_

  User . . . . . . . . . . . . . . . USER       P    *JOBD_____
  Hold job on job schedule . . . . . HOLD       P    *NO__
```

Parameter Definitions:

JOB: The name of the standard job that you wish to submit. This must be a job that has been defined on the standard jobs master file. If you enter *SELECT, a list of standard jobs will be displayed for you to choose from.

SBMDAT: The date you enter will be the date that the job will be scheduled to run. If you do not enter a date, today is assumed. Special values are:

*D1	The job will be scheduled to run on the date calculated by (today + 1 day)
*D2	The job will be scheduled to run on the date calculated by (today + 2 days)
*D3	The job will be scheduled to run on the date calculated by (today + 3 days)
*D4	The job will be scheduled to run on the date calculated by (today + 4 days)
*D5	The job will be scheduled to run on the date calculated by (today + 5 days)
*D6	The job will be scheduled to run on the date calculated by (today + 6 days)
*D7	The job will be scheduled to run on the date calculated by (today + 7 days)

SBMTIM: The time you enter will be the time that the job will run, on the date specified. The time must be entered in 24 hour clock format. So to have the job run at 3.30pm, enter 1530. If you do not enter a time, the time defined on the standard job definition will be used. Special values are

*NOW	The job will become available for processing immediately. *NOW can only be specified if the SBMDAT is *TODAY.

All other run parameters are taken from the standard job definition.

DEPJOB: This allows you to retain job dependencies or ignore them.

If you enter *LEAVE, the standard job dependency defined for the job will be kept.
If you enter *IGNORE, the scheduled job will not be dependent on another job.

USER: Specifies which user profile will be associated with the submitted job.

If you specify *JOBD (the default), the user will be taken from the USER parameter of the associated job description.
You can also specify *CURRENT to use the same user profile as the currently running job.

HOLD: Specifies whether this job is held at the time that it is put on the job schedule. A job placed on the job schedule in the hold state is held until it is manually released (or ended) via the Work with Scheduled Jobs work panel.

If you enter *NO, the job is not held when it is put on the job schedule.
If you enter *YES, the job is held when it is put on the job schedule until it is manually released (or ended) via the Work with Scheduled jobs work panel.

Submitting jobs to past dates

In some cases it is necessary for the job date to be set to a past date; this is to allow date/time-sensitive processing to be performed using a job date that is based on the scheduled date.

As an example, if a transaction posting run was supposed to run last night, but failed. A re-run can then be forced on to the job schedule with yesterday's date, so that the transaction dates on the postings can be set with the correct date.

Refer to the documentation later in this manual regarding the @RTVSDT routine on how to install this in your programs.

However, if you do force the job on to the job schedule with a past date, it is *highly recommended* that you also specify HOLD*YES) and DEPJOB(*IGNORE):

HOLD(*YES) will stop the job from being released immediately;
DEPJOB(*IGNORE) will ensure no attempt is made to check the completion status of a past dependency

Standard jobs, run on a job date basis

It is sometimes necessary to run standard jobs on a frequency that is not easily defined by the usual methods provided in Scheduler. For this reason there is a Job Calendar attached to each standard job, that can be used to define the specific calendar dates that the job should be run.

This can complement the standard run coding methods, so that you can define a job to be run Monday-Friday (Submit Day=60), as well as on June 15, January 4 and August 12.

Refer to the Work with Standard Jobs (WRKSTDJOB) command for an explanation of setting up a Job Calendar.

A summary on standard jobs

Once a job has been defined to SCHEDULER, there are four ways of getting the job to be processed:

- By the 'Submit Day' RunCode, picked up via the STRSCH (Start Scheduler) extraction program
- By a Job Calendar date, picked up via the STRSCH (Start Scheduler) extraction program
- By a scheduling Group Code, picked up via the LODSCH (Load Job Schedule) command
- By 'forcing the job onto the schedule, using the FRCSCHJOB (Force Standard Job) command.

Non-Standard jobs

A facility also exists to load a 'one-off' job, via the SBMSCHJOB command. This command is similar to the IBM SBMJOB command, but instead of sending the job directly for batch processing, it is sent to the job schedule for processing at some future time/date.

Once the non-standard job is processed, it is removed from the scheduler.

HLDSCHJOB – Hold (currently ready) scheduled job

The Hold Scheduled Job (HLDSCHJOB) command makes a currently scheduled job ineligible for processing by Scheduler. The job is held until it is:

- Released by the Release Scheduled Job (RLSSCHJOB) command
- Manually removed from the job schedule

The command prompt looks like this:

```
                  Hold currently scheduled job (HLDSCHJOB)

Type choices, press Enter.

Job name . . . . . . . . . . . . . JOB
Current schedule date  . . . . . SCHDAT
Current schedule time  . . . . . SCHTIM
```

Parameter Definitions:

JOB: The Job Name parameter specifies the currently scheduled job you wish to hold on the job schedule.

SCHDAT: The Current schedule date parameter specifies the date on which the job is currently scheduled to be run.

SCHTIM: The Current schedule time parameter specifies the time at which the job is currently scheduled to run.

RLSSCHJOB – Release (currently held) scheduled job

The Release Scheduled Job (RLSSCHJOB) command makes a currently held job eligible for processing by Scheduler. The time that the released job actually commences processing will still be dependent upon the scheduled run date and time.

If the date and time occur in the past, the job will be immediately released for processing.

The command prompt looks like this:

```
                  Release currently scheduled job (RLSSCHJOB)

Type choices, press Enter.

Job name . . . . . . . . . . . . . JOB
Current schedule date  . . . . . SCHDAT
Current schedule time  . . . . . SCHTIM
```

Parameter Definitions:

JOB: The Job Name parameter specifies the currently scheduled job you wish to release on the job schedule.

SCHDAT: The Current schedule date parameter specifies the date on which the job is currently scheduled to be run.

SCHTIM: The Current schedule time parameter specifies the time at which the job is currently scheduled to run.

RSMSCH - Resubmit processing schedule

The RSMSCH command allows you select all, or part, of the processing schedule for a specified date, and force it on to the schedule for another date.

The primary purpose of this command is to enable you to reload processing that was missed (due to operational reasons). The command prompt looks like this:

```
                     Resubmit processing schedule (RSMSCH)

Type choices, press Enter.

Extract for date . . . . . . . . EXTDAT
Extract start time . . . . . . . STRTIM         *START
Extract end time . . . . . . . . ENDTIM         *END
Submit on date . . . . . . . . . SBMDAT
Submit in *HELD status?  . . . . HOLD           *YES
Job dependency . . . . . . . . . DEPJOB         *LEAVE
```

Use of the RSMSCH command requires the user to have *JOBCTL special authority.

Parameter Definitions:

Extract for date (EXTDAT) - specifies the date you want to extract processing for (the date the processing would normally run). This is a required parameter.

> *TODAY - The job extraction will be performed for today's date.
> extraction date - Specify the date for which the extraction will be performed. The date you enter must be in job date format.

Extract start time (STRTIM) - specifies the time that the extraction processing will be performed from.

> *START - The extraction will be processed from the start of day.
> extract start time - Specify the time from which the extraction will start.

Extract end time (ENDTIM) - specifies the time that the extraction processing will be performed to.

> *END - The extraction will be processed to the end of day.
> extract end time - Specify the time at which the extraction will end.

Job Submission date (SBMDAT) - specifies the date that Scheduler will process the jobs. This is a required parameter.

> *TODAY - The jobs will be scheduled to run on today's date.
> submit date - Specify the date on which the jobs are to be processed. The date you enter must be in job date format.

Hold on job schedule (HOLD) - specifies whether jobs are to be loaded on to the job schedule in *HOLD status. A job placed on the job schedule in the hold state will not run until it is manually released via the Work with Scheduled Jobs work panel.

> *YES - Jobs are loaded on to the job schedule in *HOLD status and will not run until manually released via the Work with Scheduled Jobs work panel.
> *NO - Jobs are not held when loaded on to the job schedule.

Job Dependencies Option (DEPJOB) - allows you to retain or ignore the job dependencies that have been defined for the jobs.

> *LEAVE - The job dependency (if one is defined) on the job definition is to be retained for job submission. Jobs will only run if the defined dependent job completes normally.
> *IGNORE - The job dependency (if one is defined) on the job definition is to be ignored for job submission. The job will not be dependent on the processing of another job. You can, however, set one manually (using the WRKSCHJOB panel) once the job has been loaded).

SBMSCHJOB - Submit (non-standard) Scheduled Job command

This command has been defined to allow you to load non standard jobs onto the scheduler. The command prompt looks like this:

```
                Submit Scheduled Job (SBMSCHJOB) Prompt

Type choices, press Enter.


  Job Name . . . . . . . . . . . . JOB        R   __________
  Command to run . . . . . . . . . CMD        P   _________________________
   ___________________________________________________________________________
   ___________________________________________________________________________
  Submit on Date . . . . . . . . . SBMDAT     P   *TODAY__
  Submit at Time . . . . . . . . . SBMTIM     R   *NOW____
  Dependency . . . . . . . . . . . DEPJOB     P   *NONE_____
  Description of Job . . . . . . . TEXT       P   _________________________
   ___________

  User . . . . . . . . . . . . . . USER       P   *JOBD_____
  Request Data . . . . . . . . . . RQSDTA     P   *CMD____________________
   ___________________________________________________________________________

  Send Warning Message . . . . . . WARNSYS    P   N

  Job description  . . . . . . . . JOBD       P   SCHEDULER_
    Library name . . . . . . . . .                  NUTIL_____
  Job queue  . . . . . . . . . . . JOBQ       P   SCHEDULER_
    Library name . . . . . . . . .                  NUTIL_____
  Output queue . . . . . . . . . . OUTQ       P   *JOBD_____
    Library name . . . . . . . . .                  *LIBL_____
```

Parameter Definitions

JOB: This is the name that the job will be submitted as. It must be UNIQUE within SCHEDULER.

CMD: This is the command you wish to run. Because this command is used for the request data for the submitted job, this parameter is mutually exclusive with the RQSDTA (Request data) parameter. If both are specified, this parameter (CMD) will be used.

RQSDTA: This parameter allows you to specify the command to be run in the submitted (scheduled) job. Because this command is used for the job request data, this parameter is mutually exclusive with the command to run (CMD) parameter. If both are specified, this parameter (RQSDTA) will be ignored.

SBMDAT: The date you enter will be the date that the job will be run. If you do not enter a date, today's date is assumed.

SBMTIM: The time you enter will be the time that the job will run, on the date specified. The entry you make must be in 24 hour clock format, so to have the job run at 3.30pm, enter 1530. If you do not enter a time, it is assumed that you want the job to be submitted immediately.

DEPJOB: If the job is to be dependent upon the successful completion of another job, enter the dependency job name.

TEXT: This is for you to enter a description of the job.

USER: This specifies which user profile will be associated with the submitted job.

If you specify *JOBD (the default), the user will be taken from the USER parameter of the associated job description.

You can also specify *CURRENT to use the same user profile as the currently running job.

WARNSYS: If you enter a 'Y', the system operator will be sent a message (SCH0230) prior to the job commencing. The operator must respond to this message within one hour of it being sent. The operator's response to the message may be either Y (Continue), N (Do not continue) or R (Reschedule the job and add another 30 minutes to the last attempted start time). If no response is received within one hour, or the system operators message queue is in default mode, the answer is automatically assumed to be Y and the job continues.

JOBD: Enter the name of the job description, which the job is to be submitted with. If you do not enter a Jobd, the SCHEDULER jobd is assumed. It is important to note that the job description chosen must have SCHEDULER in its library list, as well as any libraries upon which the job is to operate.

JOBQ: Enter the job queue to be used. If you do not enter a Jobq, the SCHEDULER jobd is assumed. Unless you have a reason for using a different job queue, we recommend you use the default.

OUTQ: Enter the output queue for printed output. If you do not enter an Outq, the default from the job description is used.

Which parameter should I use, CMD or RQSDTA?

The 'Command to run' in the non-standard job can be specified in either of two parameters, the CMD or the RQSDTA parameter.

You can use either parameter to load the command, but you cannot use both - if both are specified, the RQSDTA parameter will be ignored and the CMD parameter will be used.

You should use

…the CMD parameter if you are just entering a command normally. CMD has the advantage that the command entered can be prompted (via the use of the F4-Prompt function key) for its parameters.

…the RQSDTA parameter if the SBMSCHJOB command is being used within a CL program.

Note that the RQSDTA parameter can contain variable names and expressions; but CMD can not.

STRSCH - Starting Scheduler

Once you have your jobs defined to the system, you need to start the scheduler. This is performed by the STRSCH command (which is also option 10 on the menu):

```
                  Start Scheduler (STRSCH) Prompt

Type choices, press Enter.

  Reset job schedule . . . . . RESET      P   *NO_
  Job description  . . . . . . JOBD       P   SCHEDULER__
   Library name  . . . . . .                   NUTIL_______
  Job queue  . . . . . . . . . JOBQ       P   SCHASYNC___
   Library name  . . . . . .                   NUTIL_______
  Load todays schedule . . . . . LOAD       P   *DFT
  Language library . . . . . . LNGLIB     P   _________
```

Use of the STRSCH command requires the user to have *JOBCTL special authority.

The JOBD and JOBQ parameters are used to submit the Scheduler Interrogator job. Although you are given the option of entering a different job description and job queue, it is suggested that you retain the defaults shown.

The RESET parameter advises the Scheduler Interrogator what to do with the standard jobs that are currently at *RDY (Released, ready for processing) on the Job Schedule, but have not yet been processed. This parameter is extremely useful when the Scheduler has not been used for an extended period. Old jobs that may no longer be required can be cleansed from the Job Schedule.

*RDY - Any *RDY status standard jobs that exist in the Job Schedule are removed. Reload of the Job Schedule then takes place. Non standard jobs are not affected by this parameter; if these are no longer required, they must be manually removed.

*OLD - For any jobs with a scheduling date prior to today, any *RDY (Released, ready for processing) status standard jobs that exist in the Schedule are removed. Reload of the Job Schedule then takes place.

*ALL - All standard jobs that are currently on the job schedule, at any status, are removed. Reload of the Job Schedule then takes place.

*NO - The standard jobs that already exist on the Job Schedule are not removed.

The LOAD parameter advises the Scheduler Interrogator whether to load any new jobs on to the Job Schedule at startup. This parameter is extremely useful when a power failure has occurred, so that it allows for a controlled system restart.

*DFT - The default response will be taken from the value entered in the AUTOLOAD parameter of the SCINSTALL command when you first installed Scheduler. If AUTOLOAD(*YES) was specified, then LOAD(*NOW) will be assumed; if AUTOLOAD(*NO) was specified, then LOAD(*NO) is assumed.

*NO - No new jobs will be loaded on to the Job Schedule at Scheduler startup.

*NOW - Any jobs that should be scheduled today, from the current time onwards, will be loaded on to the job schedule.

*START - All jobs that should be scheduled today, from time 00:00 onwards, will be loaded on to the Job Schedule.

The LNGLIB parameter specifies the name of the library that is added to, or removed from, the Scheduler Interrogator job's library list. The library name entered must start with *'QSYS*'* (system language library) or *'LTF*'* (user language library). Other names entered will be rejected by the command.

The START SCHEDULER command will do the following:

- Test to ensure the SCHEDULER can be started, by checking the Calendar file and the allowable processing days
- Start the SCHEDULER subsystem
- If RESET(*YES) was specified, initially cleanse the job schedule by removing any standard jobs that have not been processed
- Load the scheduler interrogator with jobs that need to be scheduled for today (if the AUTOLOAD flag on the installation command was set to *YES and/or the LOAD parameter requires it)
- List all jobs that are to run TODAY
- List all jobs that were run since the last start of the scheduler

Once started, the interrogator will constantly monitor the job schedule, submitting jobs as and when required.

The interrogator will submit jobs for execution in STRICT DATE/TIME order. So the sequence that jobs will be run is as per the sequence shown on the report. On change of day, if the scheduler is still running, it will perform its own start of day routine. It will list all jobs that are to run on the new day, listing jobs that were run in the previous session and then reset/load the interrogator for the new day.

Since the scheduler subsystem is self-running, there is no need to terminate it at all, unless you are powering down the machine, or you require termination of subsystems for dedicated tasks.

A point to note: let us say you have three jobs that would normally run today:

```
Job_Name____Start_Time
Job_1          14.00
Job_2          15.50
Job_3          16.00
```

Now, you perform STRSCH (Start SCHEDULER) at 3pm (15.00). Only Job_2 and Job_3 will be loaded. Job_1 will not be loaded, because it is scheduled to start PRIOR to the time SCHEDULER was started.

Reports

The following reports are printed by the STRSCH command.

Previous Run history log

```
Report: SCH007P             Scheduler History Log
                                                          Date/Time
Job Name     Description             Schedule Date    Start   End     Status
*************************************************************************
WRKACTJOB   Display Active Jobs     17/10/21 18:00   17/10/21 18:03    *CMP
                                                       17/10/21 18:20
SAVEOE      Save of OELIB           17/10/21 18:30   17/10/21 18:30    *DPX
                                                       17/10/21 18:30
SAVEIM      Save of Inventory Mgmt  17/10/21 19:20   17/10/21 19:30    *CMP
                                                       17/10/21 19:45
SAVEMK      Save of Marketing Data  17/10/21 19:30   17/10/21 19:45    *CLX
                                                       17/10/21 20:02
OEREPORTS   Periodical Reports OE   17/10/21 20:20   17/10/21 20:20    *CMP
                                                       17/10/21 20:55
*************************************************************************
                        ** End Of Report **
```

This report shows the completion status of all jobs that ran in the LAST session of scheduler (the report layout shown above has been condensed for inclusion in this manual).

Report Notes:

Job Name:	The system identifier name for the job
Description:	Descriptive info on the job
Schedule date/time:	The date/time the job was SCHEDULED to start
Date/time started:	The date/time the job ACTUALLY started
Date/time ended:	The date/time the job ended
Status:	The completion status of the job, which may be one of the following:

- ***CMP** The job processing completed normally
- ***RSH** The job was rescheduled due to system operator intervention (via message SCH0230)
- ***CLX** The job processing ended abnormally, due to a CL (command string) error. See the job log for information.
- ***SBX** The job could not be submitted for processing. Refer to the System Operator message queue for information.
- ***DPX** The job was not processed, due to an error in a dependent job.
- ***JBX** The job processing ended abnormally, due to an error in the job header information.
- ***CNL** The job ended, by being cancelled.

Scheduled jobs list

```
Report: SCHEXTP        Scheduler Scheduled Jobs

Job Name     Description                 Date       Time
*************************************************************************
WRKACTJOB    Display Active Jobs       17/10/21     18:00
SAVEOE       Save of OELIB             17/10/21     18:30
SAVEIM       Save of Inventory Mgmt    17/10/21     19:20
SAVEMK       Save of Marketing Data    17/10/21     19:30
OEREPORTS    Periodical Reports OE     17/10/21     20:20

             ** End Of Report **      5 Jobs Submitted
```

This report lists all jobs that will run in THIS session of scheduler.

Report Notes:

Job Name:	The system identifier name for the job
Description:	Descriptive info on the job
Schedule date/time:	The date/time the job is SCHEDULED to start

The Scheduler subsystem

To minimise the effect of scheduled jobs on your normal operations, all scheduled jobs run in their own subsystem (unless you override the STRSCH command defaults).

This subsystem, called SCHEDULER, runs two jobs:

The Schedule Interrogator
The scheduled job currently executing

The interrogator will run all the time SCHEDULER is operational. Its purpose is to test the schedule and load jobs to run dependent upon the date/time and sequence of jobs in the schedule.

The job loaded is submitted (by the interrogator) to run in the SCHEDULER in strict date/time sequence. It is possible to schedule jobs to other subsystems (by using the job description overrides in standard job maintenance). Doing this is only recommended where there are no job dependencies, since no control can be guaranteed over the number of jobs currently executing.

ENDSCH - Ending Scheduler

Once SCHEDULER is started, if you need to terminate it for any reason you must do it by the ENDSCH command, which is also option 20 on the menu:

```
                End Scheduler (ENDSCH) Prompt

Type choices, press Enter.

  How to end . . . . . . . . . . OPTION     P    *CNTRLD
```

Use of the ENDSCH command requires the user to have *JOBCTL special authority.

You have the option for a controlled or immediate termination.

A CONTROLLED termination will wait for the current scheduled job (if any) to complete execution, then terminate the interrogator, then terminate the subsystem.

An IMMEDIATE termination will CANCEL any currently executing job, CANCEL the interrogator and CANCEL the subsystem. As such, it should only be used under extreme circumstances.

Normally, you would not need to use this command. In the case of you wanting to power down the system, providing you define the controlled time delay of the PWRDWNSYS command as being longer than the scheduler time delay, the scheduler subsystem will automatically power itself down, in a controlled manner.

For example, assuming your scheduler time delay is set to 60 seconds, you enter your power down request as follows:

```
PWRDWNSYS  OPTION(*CNTRLD) DELAY(100) .......
```

After 60 seconds, the interrogator will sense that the power down request has been submitted, and will automatically initiate its own ENDSCH (*CNTRLD) command.

As a general note of caution, your delay for the PWRDWNSYS command should always be set to a value which is longer than your longest running procedure. In many cases, people prefer to use the DELAY(*NOLIMIT) so that, if a job falls over or halts in its execution, it is not automatically cancelled by the system once the controlled delay time has expired.

If you power the system down every day you might consider scheduling the PWRDWNSYS command to be the last scheduled job each day, so that the whole thing runs automatically. And if a scheduled job falls over, the PWRDWNSYS command will not be run (since the scheduler is a single stream subsystem).

The Schedule Interrogator

The Scheduler Interrogator is a job that runs in the scheduler subsystem, monitoring the job schedule on a periodical basis, determining whether a job in the schedule is due to run or not. Once it is due to run, the interrogator submits the job for execution.

The frequency at which the interrogator scans the job schedule can be altered, by using the CHGSCHDLY command, which is also option 25 on the menu:

```
          Change Scheduler Delay (CHGSCHDLY) Prompt

Type choices, press Enter.

 Job delay time in seconds  . . . SECS       P    60_
```

The delay is set at the factory to 60 seconds (which is also the command default). If you schedule jobs on a half-hourly basis (so that jobs start at 10am, 10.30am, 11am etc), it would be more efficient to have the scheduler time delay set to 1800.

If you set the delay at a time frequency LONGER than you would normally schedule jobs, the jobs will still be selected for processing, but not necessarily at the requested time. In this case, the jobs will still be run in time sequence, but the time will only reflect a SUGGESTED submit time, not an ACTUAL submit time.

For example, assume the following:

The delay is set to 900 (15 minutes).
You have a job scheduled to be executed at 8.05am.
The interrogator last scanned the schedule at 8.00am

In this case, the job would not be submitted for execution until 8.15am (last scan time PLUS time delay). In most cases this will not be a problem, just a restriction you should be aware of.

Processing at midnight

It is important to note how midnight is handled by the Scheduler Interrogator job.

To have a job scheduled to run at midnight, you must enter the time as 24:00. As such it is considered as apart of today's processing schedule, and not tomorrow. This has important implications on job dependencies, where dependent jobs must be scheduled to run within the same processing day.

Defining standard jobs to Scheduler

WRKSTDJOB – Work with Standard Jobs

The WRKSTDJOB (Work with standard jobs) command allows you to add, maintain or delete standard jobs.

```
SCH001D1                  Work with Standard Jobs
Select

Type options, press Enter.
  2=Change   4=Delete   9=Print

  Job_Name__Status_____________________________________________
_ RGZSCH      H  Reorg Scheduler Files
_ WKSBACK        Save Workshop data files
_ IMDAILY        Inventory Daily processing
_ IM_PRTINV   H  SPAS Release Invoices          No scheduling defined
_ IM_RECAT    H  Full SPAS Recategorisation     No scheduling defined

Position list to:
  IM_CLEANSE_

F3=Exit   F6=Add   F9=DUPSTDJOB
```

Use of the WRKSTDJOB command requires the user to have *JOBCTL special authority.

This panel is the first in a series of displays that allows you to work with standard jobs defined to Scheduler. It provides a list of jobs currently described, allowing you to update information describing those jobs. Note that you can only access a job if you have operational rights to the job description defined for the job.

To change an existing job, type a ‘2’ beside it and press enter.

To delete an existing job, type a ‘4’ beside it and press enter. The job will be checked to ensure it can be deleted, using the following rules:

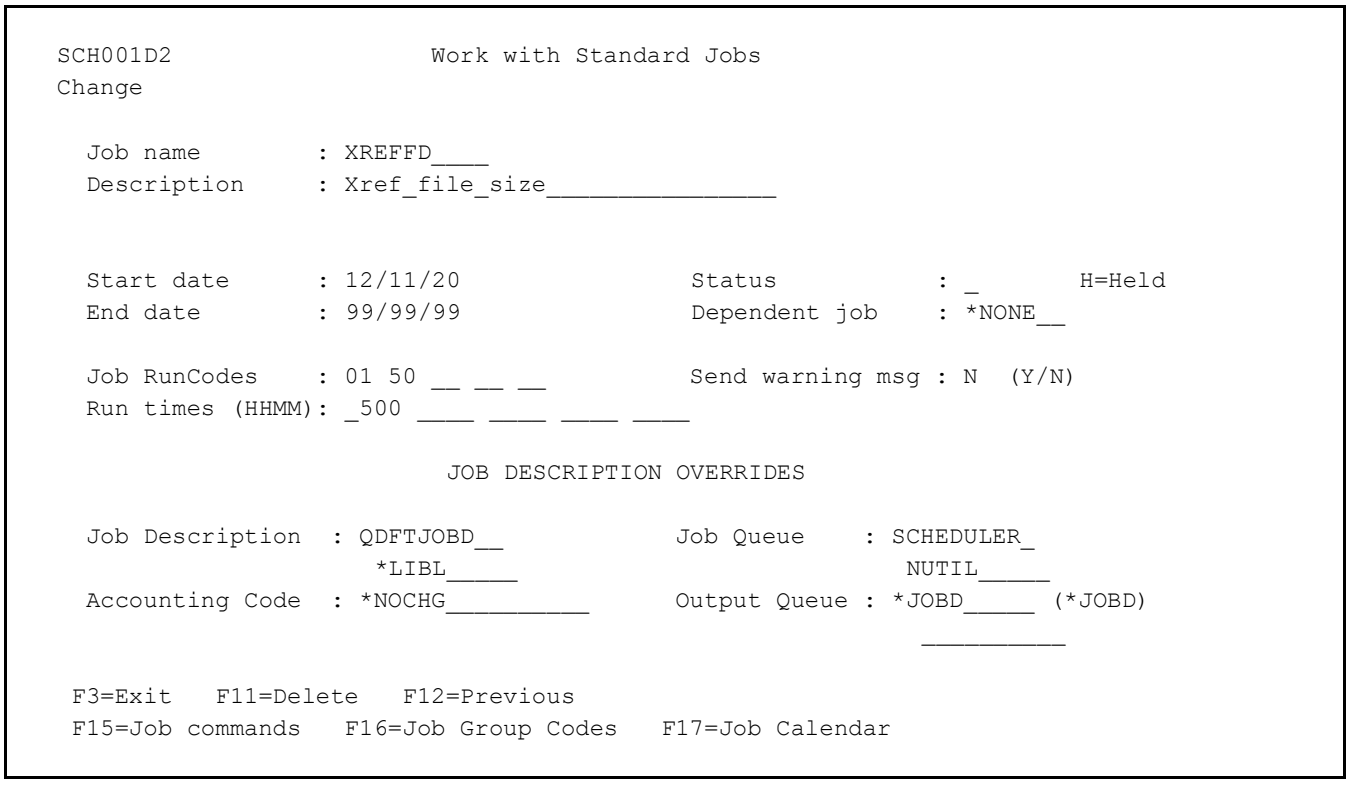
- The job cannot be on the job schedule for processing
- A dependent job cannot be on the schedule for processing
- No job on the master file can have the job as a dependency

Providing all these rules are met, you will then be shown a confirmation screen from which you can request the job be removed from the master file.

To print the definition of an existing job, type a ‘9’ beside it and press enter.

To Add a new job definition, press F6. You will then be prompted to enter the information for the new job.

To Copy an existing job to create a new job press F9. You will then be shown the DUPSTDJOB command prompt. Once copied, you can then alter the details of the new job by typing a ‘2’ beside it.



The information on this screen defines how the job is to be run. Current data (where it exists) is shown, allowing you to make any changes.

JOB NAME: This is a UNIQUE job identifier within SCHEDULER. Use this field as a brief job description. The job will be submitted using this name, thus all system displays (such as WRKACTJOB) will show this name as the job identification.

TEXT: A more detailed description of the job. You may enter anything in this field, as it is only further identification. It is not used in the running of the job.

STATUS: Determines whether Scheduler will consider the job as ‘Available for Scheduling’. If the status is ‘H’ (held), the job will not be automatically scheduled. It can, however, still be forced onto the job schedule by using the FRCSCHJOB command.

START DATE: This is the date that scheduling of the job will commence.

END DATE: This is the date the scheduling of the job will finish, and the job will be automatically set to expired. A date of 99/99/99 means that no end date has been specified.

JOB RUNCODES: This defines when the job is to be run, on a standard interval basis. Valid entries are:

00	The job will only run using job Group Codes, or by the FRCSCHJOB command
1 - 31	Run on a specific day of the month (every month)
32	Run on the last working day of the month/period
33	Run on the last working day of the year
34	Run on the last working day of the week
35	Run on the first working day of the week
36	Run on the first working day of the month/period
37	Run on the first working day of the year
40	Run every time STRSCH is performed
41	Run every time the job schedule is interrogated
50	Run every day
51	Run every Monday
52	Run every Tuesday
53	Run every Wednesday
54	Run every Thursday
55	Run every Friday
56	Run every Saturday
57	Run every Sunday
58	Run on Mondays Wednesdays and Fridays
59	Run on Tuesdays and Thursdays
60	Run Monday thru Friday
70	The job will only run using the job calendar, or by the FRCSCHJOB command

Up to 5 RunCodes can be defined on each job. If RunCode 00, 41 or 70 is specified on a job, then no other RunCodes can be defined for that job.

You should avoid using RunCode 41 if possible, as it affects normal operation of the Schedule Interrogator job.

RUN TIMES: Specify the time of day (in Hours/Minutes) the job is to be run - up to 5 times can be scheduled for each day. A 24 hour clock is used, so 2.15pm is entered as 1415. Up to 5 run times can be specified on each job.

DEPENDENT JOB: If this job is only to be run based on the successful completion of another job, enter the name of that job.

SEND WARNING MESSAGE: If a 'Y' is entered, an inquiry message will be sent to the System Operator (Message SCH0230 in NUMSGF) prior to the running of the job, advising the operator that the job is about to be run. The operator must respond to this message within one hour of it being sent. The operators' response to the message may be either Y (Continue), N (Do not continue) or R (Reschedule the job and add another 30 minutes to the last attempted start time). If no response is received within one hour, or the System Operator's message queue is in default mode, the answer is automatically assumed to be Y and the job continues.

Job Description Overrides

JOB DESCRIPTION: Enter the specific Job Description to be used. If no entry is made, the default job description SCHEDULER will be used.

A very important note: The job description to be used *must* have NUTIL and QTEMP defined in the library list, as well as any libraries that the job is to operate upon. You must also ensure the correct sequencing of the libraries in the library list; unexpected results may occur where this is incorrectly set.

Also, the USER parameter on the job description should have a valid user profile entered. It should not be defined as *RQD (which is the default for the CRTJOBD command).

ACCOUNTING CODE: If the job is to be processed to a specific Job Accounting code, enter the code. A value of *NOCHG will cause the job to be processed with the same Accounting Code as the Schedule Interrogator job.

JOB QUEUE: Enter the specific Job Queue/Library to be used. If no entry is made, job queue SCHEDULER will be used.

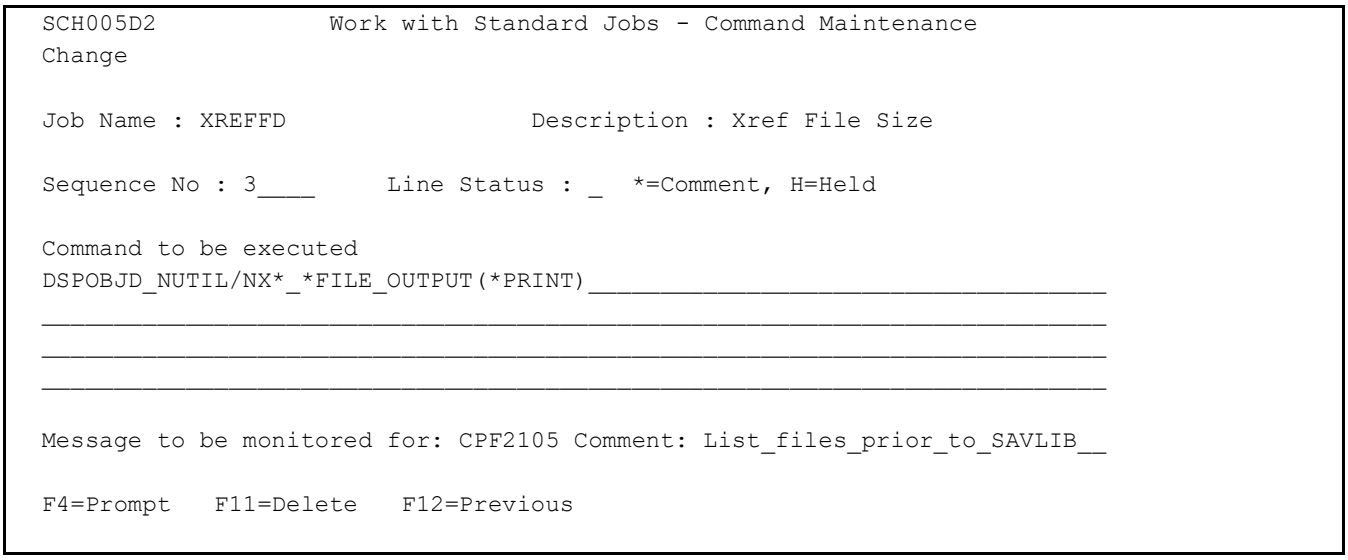
OUTPUT QUEUE: Enter the specific Output Queue/Library to be used. If no entry is made, the OUTQ specified on the job description will be used.

Command Function Keys

F4 allows you to delete the job.
F15 will allow you to maintain the Commands processed by the job
F16 will allow you to maintain the Group Codes this job can be controlled by.
F17 will allow you to maintain the Job Calendar for this job.

After you have maintained the job via this screen, the program will determine whether the modifications you have made to the job require it to be now submitted to the days job schedule. This will occur *only* if the scheduler is currently active.

If the job is already on the job schedule, and the changes you have made require it to be taken off the schedule, you must manually cancel the job from the WRKSCHJOB (work with Scheduled Jobs) work panel.



This screen allows you to update a command on a standard job. If the command line previously existed, the information is loaded onto the screen, allowing you to make any changes necessary.

SEQUENCE NO: This defines what order multiple commands are to be run by this job, in ascending sequence numbers.The entry is in the form 999.99, allowing you to later insert commands (e.g. 3.50 may be later inserted between 3.00 and 4.00).

> You may alter the sequence number by entering a new sequence (it can later be rebuilt using F10). This may not be the same as an existing command sequence number.

LINE STATUS: This defines whether the line will be processed when the job is run by Scheduler.

> If the status is 'H' (held), the command will not get processed.
> If the status is '*' (comment), the line is a comment only and will not be processed.

COMMAND: Any valid command may be entered. To prompt for command parameters, enter the command name and press F4. The command entered will be validity checked.

Comment lines (status '*') are not validity checked, and the prompt function key is disabled.

MESSAGE TO BE MONITORED FOR: Specific errors may be detected and ignored by the CL processor, by entering the message ID to be monitor here. Global messages (e.g. CPF9800) are not allowed.

> If you specify CPF0000 as the message ID to be monitored for, then *all errors* caused by the command line being processed are ignored.

COMMENT: A text reference for this command line. It is purely for documentation purposes.

Scheduler command line substitution variables

The following predefined substitution variables may be used within any Scheduler command line (providing it meets with OS/400 edit rules for the command being used):

Variable	Description		Returned variable definition
&JOB	Executing job	Name	10A
&USER		User Id	10A
&NBR		Number	6A
&OUTQ	Output Queue name		10A
&OUTQLB	Output Queue library name		10A
&SYSNAME	The system ID where the job is running		8A
&ISODATE	The current date, in YYYYMMDD format		8A
&DATE	The current date, in job format		6A
&DATE8	The current date, in extended job format		8A
&CYMDDATE	The current date, CYYMMDD format		7A
&TIME	The current job time, in HHMMSS format		6A
&SXDATE	Scheduled execution date, in YYYYMMDD format		8A
&SXTIME	Scheduled execution time, in HHMM format		4A
&SODATE	Original scheduled execution date, in YYYYMMDD format		8A
&SOTIME	Original scheduled execution time, in HHMM format		4A

If any of the special values are used the job will pick up the appropriate value from its job run attributes (when it is being executed) and insert the value into the command line prior to actually executing the command.

No edit check is performed on the variables you specify during Job Command maintenance. You must ensure the substitution variable and return value is appropriate for the command parameter it is applied to. You must also ensure that the addition of the return variable into the command string does not cause the string to exceed the 256 character length limit for the command.

```
SCH005D1        Work with Standard Jobs - Command Maintenance
Select                     Job Name: WKSBACK

Type options, press Enter.
  2=Change   4=Delete

  _   1.00  INZTAP QTAPE
  _   2.00  SAVLIB P1WKDTA
  _   3.00  DSPOBJD P1WKDTA/*ALL *ALL OUTPUT(*PRINT)


Position list to:
    __1.00

F12=Return   F6=Add   F10=Resequence   F19=Full details
```

Having added/maintained a command, you will then be shown the list of commands that are defined for this job. You may continue to add commands by pressing F6. If you have made a mistake in any of the commands, you may update it by typing a ‘2’ beside the appropriate command sequence number.

As you can see, the review screen only shows the first part of each command line. To see the full command description, press F19.

To rebuild the numbering sequence for the commands, press F10. This will renumber all commands in the job, starting at 10.00 for the first command line and incrementing by 10.00 for each command line.

Once you have entered all the commands for the job, you may return to the Job Review screen by pressing F12.

```
SCH002D1                  Work with Standard Jobs
Select                     Job Name: WKSBACK

Type options, press Enter.
  1=Select

            Group
            Code  Description
         _    FY   FISCAL YEAR END
         1    MT   MONTH END
         _    QT   QUARTER END
         _    SX   SIX MONTH END
         _    WK   WEEK END
         _    YR   YEAR END



F12=Previous
```

This screen lists all Group Codes that have been defined to the system. Those with a '1' beside them are currently in use for this job.

To add a new Group Code for this job, simply type a '1' beside that code.
To remove a Group Code from this job, just blank out the '1' against that code.

To add new Group Codes to the system, you must use the Parameter File Maintenance program.

```
SCH014D1                  Job Calendar Maintenance
                             Job Name: WKS_BACK
Type options, press Enter.
  4=Delete
               I=Job must run on this date
               E=Job must not run on this date

Sel   Date    I/E  Sel    Date    I/E  Sel    Date    I/E  Sel    Date    I/E
 _  17/09/21  I    _    0/00/00   _         0/00/00   _    _    0/00/00   _
 _  17/10/21  E    _    0/00/00   _         0/00/00   _    _    0/00/00   _
 _  17/11/21  I    _    0/00/00   _         0/00/00   _    _    0/00/00   _
 _   0/00/00  _    _    0/00/00   _         0/00/00   _    _    0/00/00   _
 _   0/00/00  _    _    0/00/00   _         0/00/00   _    _    0/00/00   _
 _   0/00/00  _    _    0/00/00   _         0/00/00   _    _    0/00/00   _
 _   0/00/00  _    _    0/00/00   _         0/00/00   _    _    0/00/00   _
 _   0/00/00  _    _    0/00/00   _         0/00/00   _    _    0/00/00   _
 _   0/00/00  _    _    0/00/00   _         0/00/00   _    _    0/00/00   _

F12=Previous
```

This screen lists all calendar dates that the job must, or must not, run - regardless of normal scheduling requirements - provided that the Scheduler is running and the date is a Scheduler Processing day.

To enter a new date, type in the date in one of the available date fields.
To delete an existing date, type a '4' beside it and press enter.

Include/Exclude flag: this field specifies how normal scheduling will be affected for a specific calendar date. This allows you to override normal scheduling codes.

- To *specifically include* the job on the schedule for a calendar date, enter an 'I' in the Include/Exclude field for the date.
- To *specifically exclude* the job from normal scheduling for a calendar date, enter an 'E' in the Include/Exclude field for the date.

DUPSTDJOB - Duplicating standard jobs

The facility exists, via the DUPSTDJOB command, to duplicate a SCHEDULER standard job. The command prompt looks like this:

```
         Duplicate a Standard Job (DUPSTDJOB) Prompt

Type choices, press Enter.

  Duplicate from Job Name  . . . . FROMJOB    R  __________
  NEW Job Name . . . . . . . . . . NEWJOB     R  __________
```

Use of the DUPSTDJOB command requires the user to have *JOBCTL special authority.

Parameter Definitions:

FROMJOB: The name of the existing SCHEDULER job that is to be copied.

NEWJOB: The new name for the duplicated job.

Using this command, it is also possible to perform a *'same-as-except'*; first perform the DUPSTDJOB command, then maintain the job via option 1 of the SCHEDULER menu.

The DUPSTDJOB command is also available from the WRKSTDJOB (work with Standard Jobs) main selection panel, via the F9=Duplicate function key.

RNMSTDJOB – Renaming a standard job

The RNMSTDJOB command allows you rename and existing Schedule standard job:

```
         Rename a Standard Job (RNMSTDJOB) Prompt

Type choices, press Enter.

 Rename Scheduler standard job  . FROMJOB    R  __________
 NEW job name . . . . . . . . . . NEWJOB     R  __________
```

Use of the RNMSTDJOB command requires the user to have *JOBCTL special authority.

Parameter Definitions:

FROMJOB: The name of the existing SCHEDULER job that is to be renamed.

NEWJOB: The new name for the specified job.

Printing the job schedule

At any time you can use the command WRKSCHJOB OUTPUT(*PRINT) to list the contents of the Job Schedule that is currently loaded.

In addition to this you can use the VFYSCHEXT command for analysing a future day's job schedule

VFYSCHEXT – Verify/Analyze Extraction Processing

The VFYSCHEXT command will perform normal Scheduler job load processing logic, based on the specified date and time. A report will be printed, but no jobs will actually be loaded onto the job schedule.

This command is provided to allow you to preview the results of the Scheduler extraction processing, without affecting your normal operations.

The Job Extraction Date parameter specifies the date you want to run the extraction processing for.

The Job Extraction Time parameter specifies the time that the extraction processing will be performed from.

The RPTGRP parameter allows you to select a subset of jobs to extract, based on the Report Group ID defined on the standard jobs.

The ACGCDE parameter allows you to select a subset of jobs to extract, based on the Accounting Code defined on the standard jobs.

Printing a list of standard jobs

The PRTSTDJOB command allows you to print details of standard jobs that have been defined to Scheduler.

The RPTGRP parameter allows you to select a subset of jobs to be printed, based on the Report Group ID defined on the standard jobs.

The SRTSEQ parameter allows you to specify which sequence the report is to be printed in – either by job name of reporting group.

Calendar Maintenance

The Scheduler Calendar is an exception calendar. In other words, a date is assumed to be a normal processing day unless you have specifically defined the date as a NON processing day in the Scheduler Calendar.

```
SCH004D1                  Work with Calendar Exceptions
                          12 period calendar is defined
  Type options, press Enter.
    4=Delete

     Exception   System   Availability
       Date     Available?  From  To      Comments
  _ 18/11/21       N        00:00 24:00   Public_holiday______
  _ 18/12/21       N        00:00 24:00   Christmas_day_______
  _ 19/01/22       N        00:00 24:00   New_years_day_______
  _ 23/01/22       N        12:00 17:00   System_maintenance__
  _  0/00/00       _        00:00 00:00   ____________________
  _  0/00/00       _        00:00 00:00   ____________________
  _  0/00/00       _        00:00 00:00   ____________________

  Position list to:
     _0/00/00

  F3=Exit
```

This work panel allows you change details on exception dates that have already been defined on the calendar, as well as add new dates.

The Calendar File is automatically cleansed by the SCHEDULER loader program, which is part of the STRSCH (Start SCHEDULER) routine. So as dates have expired, they will be automatically deleted.

To add a new date, type in the details on one of the 'empty lines on the display.
To change the details for an existing date, just type over the relevant information.
To delete an existing date, type a '4' beside it and press enter.

The fields displayed are as follows:

DATE: The calendar date that this exception refers to.

SYSTEM AVAILABLE?: This tells Scheduler whether the iSeries is available or not on that date. If 'N', automatic Scheduler processing will not be performed for that date (although manual processing may still be allowed).

AVAILABILITY FROM/TO: You can specify a time availability window for a processing date, so that normal scheduling will only be allowed within the times specified.

P/E CODE: See 13 period calendar notes, below. This option only appears if your calendar is set as a 13 period calendar.

COMMENTS: This is just a descriptive field that allows you to enter an explanation as to why this date is an exception. This will print on the Scheduled Jobs report.

Command Function Keys

F3 will exit this function.

Special notes for users with a 13 period calendar

If your calendar is defined as being a 13 period calendar (shown at the top of the work screen), you are also allowed to enter the dates that PERIOD END and YEAR END processing should occur.

The P/E CODE column can have entries in it that define your Period End (code 32 should be entered) and Year End (code 33 should be entered) processing requirements. Note that a code 33 entry (year end) implies that Period end processing should also occur on that date.

- If ***no*** P/E codes are entered against any exception dates, Scheduler will calculate all dates that processing should occur.
- If ***at least one*** P/E code is entered in the exceptions calendar it indicates to Scheduler that you are manually defining ***all*** P/E dates and so it will not calculate any - *you must define them all*.

Notes on defining calendar Time Availability windows

You can specify a time range for a specified calendar date so that, if the date is available, normal job scheduling will only occur within that time window.

When the Scheduler Interrogator loads the job schedule for a new day, it will determine whether a window has been defined for the date. If it has, any jobs that fall outside of the specified time range will still be loaded on to the job schedule, but in *HLD (held) status. A message will be sent to the System Operator if this occurs.

Any jobs falling outside the time range must be manually adjusted using the WRKSCHJOB “Work with Scheduled Jobs” work panel

Scheduler Inquiry functions

WRKSCHJOB - Working with Scheduled Jobs

The WRKSCHJOB (Work with Scheduled Jobs) panel allows you to display and alter the list of jobs that is currently scheduled for processing:

```
                  Work with Scheduled Jobs (WRKSCHJOB)

Type choices, press Enter.

Start date . . . . . . . . . . . STRDATE        *TODAY__
Output . . . . . . . . . . . . . OUTPUT         *_____
```

Use of the WRKSCHJOB command requires the user to have *JOBCTL special authority.

The STRDATE (Start date) parameter allows you to position the job schedule list to a specific day. So if you have more than one day's scheduling currently loaded you can position to the correct section of the list:

If you specify STRDATE(*START), the list will initially be positioned to the beginning of the job schedule

If you specify STRDATE(*TODAY), which is the default value, the list will initially be positioned based on the current date.

If you specify STRDATE(*NOW) the list will initially be positioned based on the current date and time.

If you specify OUTPUT(*PRINT), a list of jobs will be printed. If you specify OUTPUT(*), which is the default, the 'Work with Scheduled Jobs' panel will be displayed. This screen lists all jobs currently scheduled to run. It also allows the manipulation of any job awaiting processing.

The current job schedule will then be displayed:

```
Select                    Work with Scheduled Jobs
Type options, press Enter.

 2=Change   3=Hold   4=End   5=Display   6=Release           Status: Active
 8=Attributes 9=Print                                       Delay:     300
                                                                  Sbm?
  Job name   Description                   Date   Time   Dependency    Sts

_ WRKACTJOB  Display Active Jobs         17/10/21 21:20  *NONE      Y *CMP
_ OESAVE     Order Entry Backup          17/10/21 21:30  *NONE      Y *ACT
_ IMSAVE     Inventory Backup            17/10/21 21:30  *NONE_____ N *RDY
_ SASAVE     Sales Analysis Backup       17/10/21 21:30  *NONE_____ N *RDY
_ IMDAILY    Inventory Daily Processing  17/10/21 22:00  IMSAVE____ N *RDY
_ START-UP   Daily start-up procedure    17/10/22 07:00  *NONE_____ N *RDY

Position list to:
                                       17/10/21 21:20

F3=Exit   F5=Refresh   F6=DSPMSG   F9=FRCSCHJOB   F10=WRKACTJOB
F16=DSPMSG QSYSOPR   F21=Command line   F23=WRKSCHJOB
```

At the top of the screen, the following is displayed:

Scheduler Status: Will show either ‘Active’, if the scheduler subsystem has been started (via the STRSCH command), or ‘Inactive’, if it is not currently active. The status must be ‘Active’ for jobs to be processed by SCHEDULER.

Delay: SCHEDULER picks up jobs based on a delay time. The delay (set by the CHGSCHDLY command) shows how often SCHEDULER will check the job list to see if any jobs need processing. Thus if the delay is set to 60, every 60 seconds SCHEDULER will check the list and submit any jobs that

…have a start date/time equal to, or less than the current date/time and
…have not yet been scheduled for processing.

Delay will only be shown if SCHEDULER is active.

For each job, the following is displayed:

JOB NAME: The name the job will be/has been processed under.

SCHEDULE DATE: The date the job is to be processed. Normally, this would be today’s date or a future date, but if SCHEDULER has not been started for a while, previous days jobs may still be waiting to be processed. The date may be changed in order to reschedule the job, if required.

SCHEDULE TIME: The time the job is to be started. The time may be changed in order to reschedule the job, if required.

DEPENDENCY: Shows whether the job has a job dependency. If a job name is shown, the dependent job will only run on the successful completion of the dependency job. A job dependency may be changed for this days schedule, if required.

SUBMITTED: Shows you whether SCHEDULER has picked the job up to be processed. If N, the job has not yet been selected by SCHEDULER. If Submitted = 'Y', the job has been selected and submitted to the job queue for processing. This entry should be used in conjunction with the next entry (Status). So that if Submitted = 'Y' and Status = *RDY, this means that the job is sitting in the job queue awaiting processing.

STATUS: Displays the current status of each job, which will be one of the following:

***RDY** The job is awaiting submission
***HLD** The job has been held and will not run until released
***SBM** The job has been submitted to a job queue for processing
***ACT** The job is currently running
***CMP** The job processing completed normally
***RSH** The job was rescheduled due to system operator intervention (via message SCH0230)
***CLX** The job processing ended abnormally, due to a CL (command string) error. See the job log for information.
***SBX** The job could not be submitted for processing. Refer to the System Operator message queue for information.
***DPX** The job was not processed, due to an error in a dependent job.
***JBX** The job processing ended abnormally, due to an error in the job header information.
***CNL** The job ended, by being cancelled.

You can manipulate the scheduled jobs by option codes. You can only access the job if you have operational rights to the job description defined for the job.

Valid job option codes are:

2=*Change a job*. This code will only work if the current job status is *SBM or *ACT. It will allow you to use the CHGJOB (Change job) command for this job.

3=*Hold the job*. If held, a job will not process until released (by option 6). A job can only be held if it has not yet started processing.

4=*End a job*. This code will only work if the current job status is *RDY, *SBM or *ACT. It will cancel a job from the current schedule and perform the ENDJOB (End job) command for the relevant job.

5=*Display job information*. This code will only work once the job is active. It will allow you to use the WRKJOB (Work with job) command for this job.

6=*Release a job*. Jobs previously held by option 3 may be released for processing.

8=*Display attribute information*. This will show all of the job information that has been defined to the scheduler: commands, codes etc.

9=*Print job details*. This will send details of the job to your printer.

The following Command Function keys are active:

3 Exit and return to menu

5 Redisplay (refreshes the panel details with current Schedule status)
6 Perform a DSPMSG to display the users message queue
9 Prompt the FRCSCHJOB (Force Scheduled Job) command
10 Perform the WRKACTJOB (Work with active jobs) command
16 Perform a DSPMSG QSYSOPR (Display the System Operator's message queue) command
21 Display a command entry window
23 Display the Scheduler History work panel (DSPSCHHST)

DSPSCHHST - Displaying the Scheduler Job History log

The DSPSCHHST (Display Scheduler History Log) panel allows you to display job history, from jobs that were run by Scheduler.

```
               Display Scheduler History Log (DSPSCHHST)

 Type choices, press Enter.

       Run Date for Jobs . . . RUNDATE  *TODAY__


F3=Exit    F4=Prompt    F5=Refresh    F12=Cancel
```

The RUNDATE parameter allows you to position the start of the history log display to a specific date. If you leave the entry blank, today's date will be assumed.

The main work panel looks like this:

```
 SCH006D1                  Display Scheduler History
 Select
 Type options, press Enter.
   5=Display job log   8=Display job

        Job Name    Date/Time        Start Date/Time      End Date/Time   Status
   Sel___________(Scheduled)__________(Actual)___________(Actual)__________

    _   DLTOLDLIB 17/10/21  11:30   17/10/21  11:31:20  17/10/21  13:30:05 *CMP
    _   RETEST    17/10/21  13:00   17/10/21  13:30:10  17/10/21  18:00:05 *CMP
    _   RGZSCH    17/10/21  18:00   17/10/21  18:00:10  17/10/21  19:55:55 *CMP
    _   WRKACTJOB 17/10/21  20:00   17/10/21  20:00:00  17/10/21  20:15:25 *CMP
    _   OEBACK    17/10/21  20:00   17/10/21  20:15:30  17/10/21  20:45:25 *CMP
    _   IMBACK    17/10/21  20:00   17/10/21  20:45:30  17/10/21  20:55:15 *CMP
    _   WKSBACK   17/10/21  20:30   17/10/21  20:55:20  17/10/21  21:30:00 *CMP
    _   OEMONTH   17/10/21  21:30   17/10/21  21:30:00  17/10/21  21:55:30 *CMP
    _   IMMONTH   17/10/21  21:30   17/10/21  21:55:30  17/10/21  21:55:32 *CLX
    _   PWRDWNSYS 17/10/21  22:30   17/10/21  22:30:33  17/10/21  22:30:40 *CMP

        Position list to:                                              More...
                  17/10/21  11:30

    F3=Exit   F5=Refresh
```

You can reposition the job history list using the date and time fields at the bottom of the display. The date and time you enter will reposition based on the *Actual Start Date* of the job.

JOB NAME:	The name the job was processed under.
SCHEDULE START:	The date/time the job was scheduled to be processed.
ACTUAL START:	The date/time the job started execution.
ACTUAL END:	The date/time the job completed execution.
STATUS:	Displays the final status of the job, which can be one of the following:

***RDY** The job is still awaiting submission
***HLD** The job has been held and will not run until released
***SBM** The job has been submitted to a job queue for processing
***ACT** The job is currently running
***CMP** The job processing completed normally
***RSH** The job was rescheduled due to system operator intervention (via message SCH0230)
***CLX** The job processing ended abnormally, due to a CL (command string) error. See the job log for information.
***SBX** The job could not be submitted for processing. Refer to the System Operator message queue for information.
***DPX** The job was not processed, due to an error in a dependent job.
***JBX** The job processing ended abnormally, due to an error in the job header information.
***CNL** The job ended, by being cancelled.

By entering a '5' beside a job, the job log for that job will be displayed (providing the job was processed, and the job log is still on the system). Similarly, the entry of an '8' will display the job's execution details. The joblog will be accessed either if it is still on an output queue, or if it has been loaded into JOBLOGS (where installed).

Standard Job Inquiry

This option allows you to display the operational details of a job currently scheduled. It is accessed by taking option '8' from the Schedule Inquiry.

The screens are presented in the sequence shown below. At any time you may return to the schedule inquiry screen by pressing F3.

```
USER_ID                SCHEDULER JOB ATTRIBUTES INQUIRY
WORKSTATION_ID                JOB CONTROL DETAILS

Job name  FRIDAY
Friday Processing                        Start date   : 07/07/08
                                         End date     : 99/99/99

RunCodes     : 55                        Dependent job    : *NONE
Run times    : 1815                      Send warning msg : N  (Y/N)

                          JOB DESCRIPTION OVERRIDES

Job Description  : QDFTJOBD                Job Queue    : SCHEDULER
                   QGPL                                    NUTIL

Accounting Code  : *NOCHG                  Output Queue : *JOBD      (*JOBD)

F3=Exit
```

The information on this screen defines how the job is to be run.

JOB NAME: This is a UNIQUE job identifier within SCHEDULER.

TEXT: A more detailed definition of the job It is not used in the running of the job.

STATUS: Determines whether Scheduler will consider the job as 'Available for Scheduling'.
If the status is 'H' (held), the job will not be automatically scheduled. It can, however, still be forced onto the job schedule by use of the FRCSCHJOB command.

RUN DATE: This is the date that the job is scheduled to run (Only displayed for Non Standard Jobs).

START DATE: This is the date that scheduling of the job will commence (Only displayed for Standard Jobs).

END DATE: This is the date the scheduling of the job will finish, and the job will be automatically set to Expired (Only displayed for standard jobs).

RUNCODES: This defines when the job would run, on a standard interval basis (only displayed for standard jobs). Valid entries are:

00	The job will only run using job Group Codes, or by the FRCSCHJOB command
1 - 31	Run on a specific day of the month (every month)
32	Run on the last working day of the month/period
33	Run on the last working day of the year
34	Run on the last working day of the week
35	Run on the first working day of the week
36	Run on the first working day of the month/period
37	Run on the first working day of the year
40	Run every time STRSCH is performed
41	Run every time the job schedule is interrogated
50	Run every day
51	Run every Monday
52	Run every Tuesday
53	Run every Wednesday
54	Run every Thursday
55	Run every Friday
56	Run every Saturday
57	Run every Sunday
58	Run on Mondays Wednesdays and Fridays
59	Run on Tuesdays and Thursdays
60	Run Monday thru Friday
70	The job will only run using the job calendar, or by the FRCSCHJOB command

RUN TIMES: The time of day (in Hours/Minutes) the job is to be run. A 24 hour clock is used, so 2.15pm is shown as 1415.

DEPENDENT JOB: If this job is only to be run based on the successful completion of another job, the name of the other job is shown here.

SEND WARNING MESSAGE: If a 'Y' is entered, an inquiry message will be sent to the System Operator prior to the running of the job, advising the operator that the job is about to be run.

Job Description Overrides

Job Description: The specific Job Description/Library to be used.

Accounting Code: This code is used to change the job to be processed to a specific Job Accounting code. A value of *NOCHG will cause the job to be processed with the same Accounting Code as the Schedule Interrogator job.

Job Queue: The specific Job Queue/Library to be used.

Output Queue: The specific Output Queue/Library to be used.

```
USER_ID               SCHEDULER JOB ATTRIBUTES INQUIRY
WORKSTATION_ID                  REVIEW COMMANDS

Job name        : FRIDAY                Text : Friday Processing


     1.00  SAVLIB OELIB  DEV(TAP01)  VOL(*MOUNTED) CLEAR(*YES)
     2.00  SAVLIB OEFILE DEV(TAP01)  VOL(*MOUNTED) CLEAR(*YES)
     3.00  CALL OEUPDATE
     4.00  CALL OEREPORTS

F3=Exit   F8=Fold
```

This screen lists all commands currently attached to the job.

The commands will be processed in the sequence they are shown on the display. Only the first 70 characters of each command line is actually displayed. To show the entire command, press F8.

```
USER_ID               SCHEDULER JOB ATTRIBUTES INQUIRY
WORKSTATION_ID                  REVIEW GROUP CODES

Job name        : FRIDAY                Text : Friday Processing

               Group
               Code   Description

                MT    MONTH END
                QT    QUARTER END
                SX    SIX MONTH END
                WK    WEEK END
                YR    YEAR END

F3=Exit
```

This screen lists all the Group Codes that have been defined for this job.

These special codes are used to submit jobs to the scheduler, by use of the LODSCH command (UTLSCH Menu option 3 - Load jobs by special code).

After enter is pressed on this screen, you will return to the schedule inquiry screen.

Scheduler Appendix

The Scheduler Commands

The following commands can be accessed independently of the SCHEDULER main menu, in the same way as standard commands (providing you have NUTIL in your library list):

CHGSCHDLY	Change Scheduler Time Delay
ENDSCH	End SCHEDULER subsystem
DSPSCHHST	Display the schedule history log
FRCSCHJOB	Force the submission of a standard job onto the job schedule
LODSCH	Load Group Code jobs onto the schedule
SBMSCHJOB	Submit (non standard) scheduled job
SETPCDAYS	Set the available processing days
STRSCH	Start the SCHEDULER subsystem
WRKSTDJOB	Work with Standard Jobs
WRKSCHJOB	Work with Scheduled Jobs

On-line help facility

All SCHEDULER screens support the use of the HELP key for online information. The help text displayed is contained in the file SCH_HLPTXT. You may customise the content of the help text using the IBM Source Entry Utility (SEU).

Scheduler Start-up methods

SCHEDULER is designed to run continuously, or at least all of the time the iSeries is powered on and in normal operations mode. For this reason we suggest that a SCHEDULER start-up function be included in your system start-up procedures, in one of two ways:

Method 1 - RECOMMENDED - As an entry in your system start-up procedure. All iSeries installations have a start-up procedure (specified in the QSTRUPPGM system value), which starts all subsystems, printers etc.

Just include the STRSCH (Start SCHEDULER) command in this procedure. This will probably be the preferred method, since you retain control of when the SCHEDULER is operational.

An example of a recommended system start-up procedure is included in the NUTIL shell source file (NUTIL/QSHLSRC400, member QSTRUP). The code that needs to be added to your start-up routine is as follows:

```
           DCL        VAR(&SCAJB)      TYPE(*CHAR) LEN(1)

 /*============================================================================*/
 /*  ...Retrieve the Scheduler Autostart flag to see whether the       */
 /*      Job Scheduler should be started...                            */
 /*============================================================================*/
           RTVDTAARA  DTAARA(SCHDTA (127 1)) RTNVAR(&SCAJB)
           MONMSG     MSGID(CPF0000 MCH0000) EXEC(CHGVAR +
                        VAR(&SCAJB) VALUE(Y))

           IF         COND(&SCAJB *NE N) THEN(NUTIL/STRSCH +
                        RESET(*OLD) JOBD(NUTIL/SCHEDULER) +
                        JOBQ(NUTIL/SCHASYNC))
           ELSE       CMD(SNDPGMMSG MSGID(SCH0071) MSGF(NUMSGF) +
                        TOMSGQ(*SYSOPR) MSGTYPE(*DIAG))
```

Method 2 - As an Autostart Job. You can add an autostart job entry into your controlling subsystem description, so that when the iSeries powers up, SCHEDULER starts up also. This is done by the ADDAJE command as follows:

```
ADDAJE  SBSD(QCTL) JOB(SCHAJB) JOBD(NUTIL/SCHAJB)
ADDRTGE SBSD(QCTL) SEQNBR(100) CMPVAL('SCHAJB') PGM(NUTIL/SCHAJB)
```

So now when OS/400 starts the QCTL controlling subsystem, QCTL will, in turn, start SCHEDULER.

Source code for the autostart program, SCHAJB, is retrievable (via the RTVCLSRC Command) should you wish to modify it for your own installation. Note that you cannot add autostart job entries to an active subsystem. So the only way of implementing this procedure is by creating an alternate controlling subsystem.

The method of doing this is described in the IBM iSeries CL Programmers Guide.

Scheduler and the iSeries system startup routine

As explained in the previous section, the iSeries system startup routine normally has a statement in it to start the NUTIL Job Scheduler automatically (via the STRSCH command). The normal logic for this is:

...if the last pwrdwnsys was normal, the scheduler restarts normally

...if the last pwrdwnsys was abnormal, scheduler is not restarted and must be started manually

However, there are times when you need to bring up the system without having scheduler start it's processing automatically. The way to do this is via the NUTIL/SCINSTALL command:

```
                        Install SCHEDULER (SCINSTALL)

Type choices, press Enter.

Customer name  . . . . . . . . . NAME           *SAME

Date format  . . . . . . . . . . DATFMT         *SAME
Allow autostart? . . . . . . . . AUTOSTART      *NO       <<<<<<<<<
Load schedule at startup?  . . . AUTOLOAD       *YES
```

Normally the AUTOSTART parameter is set to *YES, but it can be temporarily changed. Setting it to *NO will cause the STRSCH in QSTRUP to be ignored (a message is sent to QSYSOPR advising this).

Then, when normal operations are required again, just re-run the SCINSTALL command and set AUTOSTART back to *YES - and then normal system startup logic will be restored.

Job Schedule job dependencies

A powerful feature of SCHEDULER is the ability to make one job dependent upon the successful completion of another. However, the following points should be noted:

a) A non standard job cannot be used as a dependency for another job. This is because a non standard job is deleted from the job schedule once it has completed. Any job dependent upon the non standard job would (if it were possible) therefore end with status *DPX (dependency error).

b) If you schedule a job, and that job submits another job, the completion status of the initial job will only reflect the successful submission of the new job. It will not reflect the completion status of the second submission. Thus if a dependency is defined, the dependent job will run based on the successful submission of the dependency, not the successful completion of the dependency. This is probably explained better by example:

Job	*Command_to_process*	*Dependency*
JOB_1	SBMJOB JOB(BACKUP) RQS('SAVLIB AALIB')	*NONE
JOB_2	CALL PGM(DAYENDRPT)	JOB_1

JOB_2 will run if job BACKUP was successfully submitted. The result of the library save does not affect the dependency.

c) Any time scheduling anomalies between a dependency and its dependent job will be corrected by the Schedule Interrogator. Once again, an example:

Job	*Scheduled Start_Date*	*Scheduled Start_Time*	*Dependency*
JOB_1	06/11	10.50	*NONE
JOB_2	06/11	10.30	JOB_1

JOB_2 is scheduled to start 20 minutes *before* the job it is dependent upon (JOB_1). When the Schedule Interrogator detects this it will attempt to reschedule JOB_2 to submit at 10.51 (ie, 1 minute *after* its dependency). So the above problem will be resolved as:

Job	*Scheduled Start_Date*	*Scheduled Start_Time*	*Dependency*
JOB_1	06/11	10.50	*NONE
JOB_2	06/11	10.51	JOB_1

d) Job dependencies are date related; in other words both jobs must be scheduled to run on the same date. The actual execution date is not considered. Take the following example:

Job	*Scheduled Start_Date*	*Scheduled Start_Time*	*Dependency*
JOB_1	06/11	23.15	*NONE
JOB_2	06/11	23.45	JOB_1

If JOB_1 finishes PROCESSING at 00.30 the next day. JOB_2 will run the next day after its dependent job has completed processing. However, Scheduler does not consider the actual execution date to test a dependency, it tests the requested (or scheduled) date and so the dependency is still true.

Let us take this point one step further and say that JOB_1 and JOB_2 are defined as jobs that are run every day. At midnight the jobs for the new day are loaded onto the job schedule:

Job	*Scheduled Start_Date*	*Scheduled Start_Time*	*Dependency*
JOB_1	06/11	23.15	*NONE
JOB_2	06/11	23.45	JOB_1
JOB_1	07/11	23.15	*NONE
JOB_2	07/11	23.45	JOB_1

JOB_1 from the previous day's schedule completes processing and JOB_2 from the previous day is now ready to run. What is its dependency? It is still JOB_1 from the previous day, since (as we said earlier) it is the *scheduled* date that is important, not the *execution* date. Similarly, JOB_2 for the new day will attach itself to JOB_1 for the new day.

e) Where a job has a dependency, and the dependent job exists on the same days schedule more than once, the *first occurrence* of the job on that day is considered in dependency testing. Look at the following example:

Job	*Scheduled Start_Date*	*Scheduled Start_Time*	*Dependency*
JOB_1	06/11	06.15	*NONE
JOB_1	06/11	18.15	*NONE
JOB_2	06/11	23.45	JOB_1
JOB_1	06/11	20.15	*NONE

JOB_2 will attach itself to the first occurrence of JOB_1 for that day's schedule; in other words it will be dependent on the job JOB_1 that is to process at 06.15.

f) The dependent job will not be submitted to the job queue for processing until the dependency has completed normally:

Job	*Scheduled Start_Date*	*Scheduled Start_Time*	*Dependency*
JOB_1	06/11	08.15	*NONE
JOB_2	06/11	08.16	JOB_1

Let us say that JOB_1 starts processing at 08:35:00 and completes processing (normal completion) at 08:45:00. The Schedule Interrogator job will submit JOB_2 to the job queue for processing on the first interrogation after 08:45:00.

System security

SCHEDULER makes no attempt to override system security in any way. We believe that it is your responsibility to control which users are authorised to functions within your applications.

If your system has work management objects controlled by specific authorisations, make sure you authorise SCHEDULER users to them before attempting to run a scheduled job through them. This includes Job Queues, Job Descriptions, Output Queues, Library Descriptions etc.

> For example, if you attempt to run a SCHEDULER job via a job queue that is not authorised to the user who started SCHEDULER, your Job Schedule display will show that the job was submitted, but will remain forever in *RDY status. If you look in the job log for the SCHEDULER interrogator job you will see why the situation occurred ('*Not authorised to object...*').

So an important note should be made about the Standard Jobs Maintenance screen, as well as the SBMSCHJOB command - both use *LIBL to identify the job queue, job description and output queue as a default (if you do not specify a library name). Make sure you know which of these objects will be picked up from the appropriate library list.

Additional control is also defined to ensure a user is authorised to manipulate individual jobs. The control for this is based on the job description that is defined on the scheduler job definition – if a user attempts to access (or use) a specific Scheduler job a check will be made to ensure the user has operational rights available to the job description defined on the job. If not, access to that job will be denied.

SCHVFYAUT – Verify user security

As supplied, all NUTIL pre-defined Scheduler jobs are normally run by QSYSOPR, using the NUTIL/SCHEDULER Job Description. For these jobs the product will continue to run as before if QSYSOPR has the same authorities as shipped by IBM. For all other standard jobs it is essential that you perform an analysis to ensure they will still function as expected.

For each standard job defined, you should make sure that the user profile specified on the job description has sufficient rights to perform the job. A new program, SCHVFYAUT, has been included with this release and this will print a list of all standard jobs defined, along with user profile information related to each job. You can run the program from the command line, or via menu UTLSCH option 43.

The *LDA - Local Data Area

SCHEDULER itself makes no use of the Local Data Area (*LDA) or the Group Data Area (*GDA).
However, when you use the Submit Scheduled Job (SBMSCHJOB) command, a 'snapshot' of the current *LDA is taken, and passed to the scheduled job, in the same way that the standard Submit Job (SBMJOB) command does.

Unattended processing

The main problem encountered when running the machine unattended is the answering of messages. The best way to overcome any job holdups due to console messages not being answered is to place the system operators message queue in default mode. All OS/400 messages are defined with a default response, which will be used in this case (these are shown in the IBM Messages Guide). We also suggest the system console message queue be treated the same way, if it is signed on. Therefore, when the operator leaves at night (unattended environment), the last task should be to change the queues to default answer mode:

```
CHGMSGQ QSYSOPR  *DFT
```

Then, when the operator returns to work (attended environment), the queues should be returned to notify or break mode:

```
CHGMSGQ QSYSOPR  *BREAK SEV(10)
```

If any of the default responses are not considered suitable for your operation, you can include your own default values to be used for specific messages. For example, CPA0701 is a CL program error detector, with a default reply of 'C' (cancel). You may wish to have a dump printed whenever the error is detected. To do this, add a reply list entry to the system's automatic message reply list as follows:

```
ADDRPYLE SEQNBR(50) MSGID(CPA0701) RPY(D)
```

This method is more preferable than using the WRKMSGD (Work with message descriptions) command to change the default reply on specific messages, simply because you do not have to reapply the changes every time you install a new release of the operating system. Refer to the IBM iSeries CL Programmers Guide for additional information on this topic

Calendar exception processing

The Scheduler calendar exceptions file is provided as a way of stopping the Scheduler Interrogator job from automatically processing jobs on specified dates. If a date is defined as being a 'non-processing day' on the exceptions calendar then no Scheduler jobs will be automatically loaded for that date.

Manual processing can still be performed on a day that is specified as a 'non-processing' day, provided that the Scheduler subsystem is still running.

The only way you can stop *any* Scheduler processing on a specified date is to ensure the Scheduler subsystem is not running on that date.

Considerations for periodical processing

Periodical processing within Scheduler is controlled by job codes 32 (last day of month/period), 33 (last day of the year) and 34 (last day of week).

It is important to understand that these dates will be calculated based on the control flags defined using the SETPCDAYS (set allowable processing days) command. The codes do *not* take into account any calendar exceptions that exist in the Calendar Exceptions file.

This means that if the last processing day of the week, month/period or year is defined as a 'non processing' day in the calendar exceptions file, your end of period processing jobs will be submitted with a 'held' status.

These jobs must then be manually released via the WRKSCHJOB (Work with Scheduled Jobs) control panel.

A message is always sent to the System Operator message queue when this situation occurs, in order to warn you that manual action is required.

Date formats

SCHEDULER, as supplied to you, will display dates in the format defined by whatever format your job is currently using. If you wish to always display dates in a format *other than normal job format*, use the SCINSTALL command to alter the value of the DATFMT parameter:

*DMY for day/month/year
*MDY for month/day/year
*YMD for year/month/day
*JOB for job format
*SYSVAL for system format (QDATFMT)

Note that changing this value will affect all users running any Scheduler commands or functions.
Regardless of your display format, all dates in files will be stored as Year/Month/Day.

@SCHRUN - Scheduler JobStream API

The Scheduler JobStream API allows you to run the commands defined for a standard job - independently of Scheduler. The syntax of the API is

```
CALL NUTIL/@SCHRUN PARM(&JOB)
```

Where &JOB is the name of the standard job defined in the Scheduler database.

The API can be called interactively or submitted for batch processing. The Scheduler History log is updated with the results of the processing, but the job does not appear on the Scheduled Jobs list.

The user requesting the API call is assumed to be the processing user.

Use of the @SCHRUN API requires the user to have *JOBCTL special authority.

@RTVSDT - Retrieve “job scheduled” date

This program will retrieve the “Scheduled date” for a specified job identifier. The scheduled date refers to the date that the job was expected to commence running via the NUTIL Job Scheduler.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Field	*Defn*	*Usage*	*Description*
@JOBNBR	6A	I	Job number
@USER	10A	I	Job user
@NAME	10A	I	Job name
@SCHDATE	8P0	O	Field containing the “Job Scheduled” date, in YYYYMMDD format.

The main purpose of this program is to allow a currently running user process to determine on what date it was ‘supposed’ to run. This is especially important over the midnight period, where the job may be scheduled to run on “Day 1”, but is still running, past midnight, on “Day 2”. A call to @RTVSDT will return the date for Day 1, regardless of the current system date.

> The main use of this is to ensure a month-end or year-end process can still detect its correct year/period, by referring to the scheduled date rather than the actual date.

Where the jobnbr/user/name is not known to Scheduler, a value of 0 will be returned as the “Job Scheduled” date.

Setting job date based on scheduled date

In some cases it may be necessary to set the jobs run date to the date that the job was scheduled to run. The code for this is as follows:

```
DCL         VAR(&JOBNBR)   TYPE(*CHAR) LEN(6)
DCL         VAR(&USER)     TYPE(*CHAR) LEN(10)
DCL         VAR(&NAME)     TYPE(*CHAR) LEN(10)

DCL         VAR(&SCHDATE) TYPE(*DEC)  LEN(8 0)
DCL         VAR(&JOBDT8)   TYPE(*CHAR) LEN(8)
DCL         VAR(&JOBDTE)   TYPE(*CHAR) LEN(6)

RTVJOBA     JOB(&NAME) USER(&USER) NBR(&JOBNBR)

CALL        PGM(@RTVSDT) PARM(&JOBNBR &USER &NAME &SCHDATE)

IF          COND(&SCHDATE *NE 0) THEN(DO)
CHGVAR      VAR(&JOBDT8) VALUE(&SCHDATE)
CVTDAT      DATE(&JOBDT8) TOVAR(&JOBDTE) FROMFMT(*YYMD) +
              TOFMT(*JOB) TOSEP(*NONE)
CHGJOB      DATE(&JOBDTE)
MONMSG      MSGID(CPF0000)
ENDDO
```

Which user profile is used to submit jobs?

An important operational and security consideration for submitted jobs is the user profile that is attached to a job. Scheduler has two distinct methods of loading the user for a processing job:

User profile for manually started jobs

A manually started job is one that has been submitted for scheduling via either the LODSCH (Load job schedule), the SBMSCHJOB (Submit non-standard scheduled job), or the FRCSCHJOB (Force a standard job onto the job schedule) command.

These commands have a USER parameter included on them, which can be set to either

*JOBD (the user profile named on the associated job description will be used) or

*CURRENT (the same user used by the job running the command will be used)

in order to assign the correct user profile for the submitted job to use.

User profile for automatically started jobs

An automatically started job is one that has been loaded onto the job schedule by the standard scheduler start-up (either via the STRSCH command, or at midnight start of processing for a new day).

These jobs are standard jobs (by definition) and will always be submitted using the user profile defined on the related job description.

Authorisation and Security considerations for Job Scheduler User Profiles

Scheduler does not attempt to override normal system authority functions - you must ensure that the user profile specified for processing a particular scheduled job has sufficient authority to run all steps in the specified job.

User Profile considerations for processing Scheduled jobs

Scheduler jobs will be submitted for processing for the User defined on the job description specified for the job. The user profiles to be used for processing Scheduler jobs must adhere to the following requirements:

- Password: The password on the user profile can be a valid, non-expired password, or you can specify *NONE.
- Status: The user profile status must be set to *ENABLED
- Initial program: The initial program parameter of the user profile will be ignored for Scheduler job submissions. As such it can be set to a valid program name, or it can be set to *NONE
- Initial menu: The initial menu parameter of the user profile will be ignored for Scheduler job submissions. As such it can be set to a valid menu name, or it can be set to *SIGNOFF

If the password has expired, or the status is *DISABLED, the job submission will fail.

What happens with USER(*RQD)?

One of the main problems that can occur with Scheduler job submissions is when a job is to be submitted using USER(*JOBD), but the job description has USER(*RQD) specified on it. In this case, the job will take on the user profile that is being used to run the Scheduler Interrogator job (job name SCHEDULER in subsystem SCHEDULER).

As supplied by Navan, this user profile is set to USER(QSYSOPR), as specified in the SCHEDULER job description. While not recommended, this can be changed to another user, and processing for Scheduler will then occur under that profile.

For security reasons you should avoid having jobs submitted using job descriptions defined with USER(*RQD).

Interfacing Scheduler to a Pager product

It is possible to interface the Job Scheduler to a pager product, if you have one installed on your iSeries. Once specified the interface will send Scheduler error messages to the defined pager.

NUTIL does not have a pager utility included; this interface is supplied to allow you to interface NUTIL to an existing Pager utility installed on your iSeries.

To define the interface you need to perform two steps:

Create the Scheduler/Pager interface API program

The source code for the interface program is in source file NUTIL/QSTUFF, member NSNDPGRMSG. You must modify the supplied source code so that it calls the correct pager command for your pager product. Refer to the user guide of your Pager product to determine how to send messages to your Pager.

The supplied program source refers to a command called SNDPGRMSG (send pager message). This is not a standard iSeries command, nor is it a command supplied with NUTIL; it is merely a template shell where you must specify your unique Pager utility message command. The only parameters important for you to user are the &TOPAGER parameter (which specifies the Pager ID you want the messages sent to) and the &MSG parameter (which is the text of the message that is to be sent):

```
0138.00
0139.00 /*************************************************************
0140.00 /* Send the pager message...
0141.00 /*
0142.00 /*        *** YOU MAY NEED TO MODIFY THE SEND COMMAND ***
0143.00 /*
0144.00 /*************************************************************
0145.00              IF         COND(&MSG *NE ' ') THEN(DO)
0146.00              SNDPGRMSG  PAGER(&TOPAGER) MESSAGE(&MSG) TIMES(1) +
0147.00                           INTERVAL(30)
0148.00              ENDDO
```

Compile the program into NUTIL, or into a library that is in the Scheduler library list. It is recommended that you do not leave the NSNDPGRMSG source or program object in the NUTIL library as it will be lost when you install the next release of NUTIL.

NSETPGRDFN - Set Pager Utility definition

You start the interface by using the NSETPGRDFN command to define your Pager utility to NUTIL:

```
               Set Pager Utility definition (NSETPGRDFN)

Type choices, press Enter.

 Pager Utility product library  . PAGERLIB
 Default pager ID . . . . . . . . DFTPAGER
```

The PAGERLIB parameter defines the library name your Pager program product is stored in. This library is added to the library by the NSNDPGRMSG command when sending a pager message. This is a required parameter.

The DFTPAGER parameter defines the pager ID to which messages will be sent. This is a required parameter, but the value entered is not validated by the NSETPGRDFN command.

Once this definition is entered, the Job Scheduler will start sending pager messages (via the NSNDPGRMSG program mentioned above) for any errors that occur in the Scheduler Interrogator job, or in any job that is processed by the Scheduler Interrogator.

Backup and Recovery of Scheduler data

We suggest the following security backups of Scheduler data:

After Installation: Save the library containing SCHEDULER using the SAVLIB (Save Library) command.

Before the library backup can be performed, you must terminate the Scheduler Interrogator job (using the ENDSCH command) if it is currently running.

After Scheduler Master File Maintenance: Save the master files as follows:

```
SAVOBJ  OBJ(SCHMST00 SCHCMD00 SCHJBC00 SCHCOD00 SCHCLD00) +
            OBJTYPE(*FILE)
```

This may be done as a standard job, via SCHEDULER.

Weekly: Save the master files, as above.

Saving Scheduler data as a part of a system backup

If you are saving the NUTIL library (or the Scheduler data files in NUTIL) as a part of another system-wide backup you should end the Scheduler Interrogator job while the save is running.

All you need to do in your CL program that runs your backups is have the ENDSCH command run before your SAVLIB command to end the interrogator; then process the STRSCH RESET(*NO) command straight after the save has completed.

Program Generator

An Overview

The NUTIL Program Generator is a productivity tool for programmers at all levels within your organisation. It rapidly produces proven RPG/400 or ILE RPG/400 code from a comprehensive set of template programs.

You can extend these templates as you wish or tailor them to your standards, enabling simple seamless integration with your current systems and programming standards. The system is driven from a single menu and can be used productively by your staff within 30 minutes of installation.

The Program Generator performs best with database files which have well defined field attributes. Fields which require automatic date formatting and validation must have the date edit code (or data type L) specified on their field attributes to force the appropriate date manipulation source code to be generated. All date fields are assumed to be held on file in Year/Month/Day format; they will be displayed in the format specified in the users job.

General

- Simple to use
- Fully commented programs
- Consistent code
- Creates Display/Printer/CLP source as well as RPG
- Enforces programming design standards
- User can extend or tailor the standard shells supplied

Technical

- Supports Physical/Logical and Join files
- Subfile Page at a time Pagedown/Pageup
- Subfile positioning by key
- Interface to merge your own source code into the generated source
- Automatic date formatting & validation
- Error message handling
- Record locking support
- Help key support
- Display files use Putovr Ovrdta Ovratr

Performance

For an experienced programmer to code and test a File Maintenance program over a file with 40 fields could take 4-5 days if the program was to include the following specifications:

Program controlled Pagedown/Pageup
Ability to load subfile using partial key
Record Locking
Trap and display/print abnormal errors

The Program Generator function within NUTIL can generate the required source code for the above function in a matter of minutes.

Installation

The only installation procedure required for the program generator is to ensure that the defaults to be used are correct; this is achieved by using option 15 of menu UTLGEN (which runs the GENPGMDFTS command). Refer to 'NUTIL Installation Procedures' earlier in this manual for details.

Post install instructions

If you are installing a new version of NUTIL (replacing a previous copy), you should be aware that there may have been additional messages added to the message file used by all program shells.

If you are using the NUTIL/RUNMSGF message file to generate your program shells, you do not need to do any post-installation work. But if you generate shells that refer to another message file you should make sure that your own message files are kept up-to-date. To do this you will need to merge the program generator runtime messages in to your user message file(s):

```
MRGMSGF NUTIL/RUNMSGF  userlib/usermsgf
```

Repeat this for every user message file you need to update. All GENPGM runtime messages are numbered GEN*nnnn*. You should try to avoid using the GEN prefix in your own messages to ensure they do not conflict with NUTIL GENPGM messages.

An important note on keyed sequence files

Most of the supplied Program Shells are designed to work on keyed sequence files.

Where the shell is a 'Display' type, and it requires the file to be keyed, the key is assumed to be physically unique. In other words, regardless of whether the UNIQUE keyword is specified, there should be no possibility of two records having the same key. If this situation occurs then the generated program will operate in one of two ways:

> Pagedown/Pageup will be unpredictable, but most likely will not appear to function
>
> Record selection will be unpredictable, but most likely will choose the first record possessing the key (regardless of the one selected on the display).

As such, you should always ensure the key you are using retains its uniqueness in the file. This is especially important where you are restricting the key to a partial key.

Considerations for key fields containing negative values

As supplied, all Program Shells assume that numeric key fields use positive numeric values only.

If a numeric key field is known (and expected) to contain negative numeric value you will need to manually alter the program source before it work correctly.

Scan the generated source and change the positioning values **for the affected key fields only**:

> Where key field FKEY*nn* is being set to *Zero, change the set to *Loval
> Where key field FKEY*nn* is being set to *ALL'9', change the set to *Hival

Database limitations

The Program Generator does not support files containing

- Null values in null-capable fields
- Floating point data type fields

In both cases program source can be successfully generated, but there is no guarantee of successful operation of the finished (compiled) program objects.

Encoded Vector Index (*EVI) files can not be referenced by the Program Generator.

Date, Time and Timestamp fields are supported in the ILE program shells. In the case of the date data type, dates in the range 1940-2039 are fully supported. You may need to (manually) modify the generated source code to allow the processing of dates outside this range.

The Program Generator menu

All Program Generator functions can be reached from the GO NUTIL/UTLGEN menu:

```
UTLGEN                    Program Generator Utility
                                                        System:   NAVAN
Select one of the following:

     1. Start programmer menu                                    STRPGMMNU
     2. Generate program source                                  GENPGM
     3. Create RPG Merge Source member                           NCRTMRGSRC

  Support functions
    11. Maintain program templates                               WRKMBRPDM
    12. Maintain program template selections                     GEN007

    15. Set system defaults for program generation               GENPGMDFTS

  Other options
    90. Sign off


Selection or command
===>

F3=Exit    F4=Prompt    F6=DSPMSG    F9=Retrieve    F12=Cancel    F14=WRKSBMJOB
F18=WRKSPLF
```

Start Programmer Menu

The Source library and object library name automatically default to the current value of the source library name used by the Program Generator prompt.

The Programmer menu displayed above has been linked to the exit program named NUTIL/NCRTOBJECT. When option 3 is selected on the Programmer menu to create an object, ALL imbedded compiler directives are executed. See section on Source compiler directives.

Generate program source

You use the GENPGM command (which is option 2 on the GO UTLGEN menu) to start the program generation facility. You can specify a default program shell type to use at the command prompt; the default is *PRV which will start the Program Generation prompt screen showing the last shell type that you used.

The command will then display the Program Generator Parameters Screen, as shown below:

```
GEN003D1                    Program Generator Prompt
Select
Type options, press Enter.


Program name to generate . . . . . . . IM4003
Program source type to generate  . . . *ILE  *OPM (RPGIII), *ILE (ILE/RPG)
Program shell type to use  . . . . . . WRKPNL       ?=List
Display/Report title . . . . . . . . . Stock file maintenance
Data file to use . . . . . . . . . . . *LIBL      / ITWSTKRV

Library Name for generated source  . . IMSRC
Create or replace member?  . . . . . . R  C=Create new, R=Replace existing

Number of restricted keys. . . . . . .
Modify field positioning and text? . . Y  Y=Yes, N=No
Incorporate Commitment Control?  . . . N  Y=Yes, N=No
Terminate program with *INLR on? . . . Y  Y=Yes, N=No

Job Description to use . . . . . . . . *LIBL     / IMJOBD


F3=Exit   F4=Exit and update defaults   F6=Process
F21=Command line window
```

Pressing the page down key will display the second format:

```
GEN003D1                  Program Generator Prompt
Select
Type options, press Enter.


Program name to generate . . . . . . . IM4003
Program source type to generate  . . . *ILE
Program shell type to use  . . . . . . WRKPNL
Display/Report title . . . . . . . . . Stock file maintenance
Data file to use . . . . . . . . . . . *LIBL      / ITWSTKRV
Program shell source file  . . . . . .            / *DFT
Message file to use  . . . . . . . . . IMMSGF
Input the *LDA Information?  . . . . . N  Y=Yes, N=No
Single, double, AutoGen columns. . . . G  S=Single, D=Double, G=AutoGen
Maximum width to generate  . . . . . . *DSPF:  80  *PRTF: 132
Start position for DSPF rcd format . . Row  4 Col  2
Table of Workfield name prefixes . . . XZQABCDEFG
ILE Workfield prefix mode  . . . . . . O  O=Overlay, A=Append
ILE Convert for EVAL . . . . . . . . . Y  Y=Yes, N=No
User Merge source file . . . . . . . . *LIBL      /            mbr

F3=Exit   F4=Exit and update defaults   F6=Process
F21=Command line window
```

The GENPGM command is used to generate programs to perform a particular function on a specified file.

The function of the program to be generated is specified by the type of shell program required (for example, a simple single record select and update program). A list of the shell program types may be retrieved by typing a ? in the Program Shell Type parameter.

The generated program is based on a Template program (or *shell*) that uses the fields and keys from the file specified to define any screen or printer files.

There are two shell template source files supplied with NUTIL:

- **QSHLSRC400** contains templates used to create iSeries type programs. The RPG source is Old Program Model (*OPM, or RPG/400) source.
- **QSHLSRCLE** contains templates used to create iSeries type programs. The RPG source is Extended Program Model (*ILE, or ILE RPG/400) source.

New Templates may be created in the shell source files, or you can create shell source files in your own user libraries. For information on how to create these Templates refer to the section *Creating a Program Template* later in this manual. It is our recommendation that you do not change the shells supplied; if you need to modify them you should copy the shells and then modify the copy to your requirements.

The source for the generated program is placed in the following source files in the specified Source Library; please ensure the source files exist before requesting source generation:

- QRPGSRC for *OPM RPG/400 source
- QRPGLESRC for *ILE RPG/400 (RPG/IV) source
- QDDSSRC for DSPF and PRTF source and
- QCLSRC for CL source
- QPNLSRC for PNLGRP source

The Source Type to Generate prompt will determine the type of RPG program source that is generated.

> An entry of *OPM will generate RPG/400 source code into the target source file QRPGSRC. If you specified *DFT for the shell source file to use, an *OPM generation will use NUTIL/QSHLSRC400.
>
> An entry of *ILE will generate ILE RPG/400 source code into the target source file QRPGLESRC. If you specified *DFT for the shell source file to use, an *OPM generation will use NUTIL/QSHLSRCLE.

It is your responsibility to ensure that the shell source file specified contains the source type you are expecting. If you specify that the shell type is *ILE, but the actual source code is RPGIII source, the source that is generated will be unusable. It is our recommendation that you use *DFT as the entry for the shell source file to use.

The member name for the RPG program will be the Program Name specified.

The source member names used by the source generation phase are based on the program name that was entered. If a display, printer or CL source member was generated it is suffixed based on values contained in data area PGDFTS. These suffixes may be changed to suit your installation. Refer to option 15 of the UTLGEN menu.

> *The source created in the source generation phase is not compiled by the program generator.*

The user must perform the object creation using the Programmers Menu linked with the exit program NUTIL/NCRTOBJECT, or via PDM using the command NPDMCRTOBJ (option CO if you are using the NUTIL/QAUOOPT user options file).

The parameters entered by a user are stored after each generation has completed or when F4 is pressed. This allows for ease of use when using the same template many times across different files.

GENPGM parameters

Program Name: This parameter specifies the name of the program to generate. The program will use this name while the associated Display, Printer and CL source members will use the program name with a suffix (as defined in the PGDFTS data area (note that these suffixes can be changed, using option 15 on the UTLGEN menu).

> The program name entered must be a valid name (by iSeries object naming standards) and be 6 characters long. For example if the program name was entered as GEN001, then the display file for this program would be GEN001D.

Program Type: This parameter specifies which shell program to use as a base for the program to be generated. To view a list of the types of program available, enter a ? in the program type field.

Screen Title: Enter the title to appear at the top of the screen or report. All standard shell program templates that perform both Maintenance and Inquiry functions automatically concatenate the literal '- Maintenance' and '- Inquiry' to the screen title depending on how the program is called.

File To Use: Specify which file, and the library, that you require the resulting program to perform its function on. This should be a single record format file, But it may be a Logical or Logical Join file.

> In the case of the MNTDTAARA program shell, you should enter the name of the format file defining the layout of the data area.
>
> In the case of the MENU program shell, you should enter '*NONE' as the file to use.

Source Library: Enter the library name where the generated source code is to be placed. The generator will add the members to QRPGSRC, QRPGLESRC, QDDSSRC, QPNLSRC and/or QCLSRC, depending on the source type created.

Member Option: Specify whether the source members generated are new or are to replace existing source members. Enter either C=Create or R=Replace.

Number of restricted keys: Specify the number of restricted keys you expect the generated program to receive as incoming parameters (excludes any other parameters already defined).

> For example a Selection program is to be generated across an inventory file which has a key of Branch Code and Part Number. You may require the program to receive the Branch code as a parameter so that only parts for that branch will display. In this case you would specify 1 restricted key (Branch)

Modify Fields: This option causes the generator to display a list of all the fields on the file specified. The user may then change the sequence that the fields are to appear on the display or report or remove them altogether from the screen. The user may also change the field name and heading that will appear with the field.

Include LDA: This parameter tells the generator whether to include logic for retrieving the Local Data Area into the program or Not. Enter Y for Yes to retrieve the LDA or N for no.

> Note: If *LDA information is requested then a reference file named LDA must exist to externally describe the layout of the Local Data Area. Alternatively the Program Templates must be modified to accommodate your environment.

Message File: Specifies which message file the generated program is to use for messages. If a message file other than RUNMSGF is selected then the RUNMSGF messages must be merged with the nominated message file. This is the responsibility of the person generating the source code.

Row/Column: Enter the Row number and Column number that the first line of information is to show on the display file.

Single/Double: Specify how the main data display panel should be formatted:

> Enter an S to specify that the display file generated should display the file data in one column.
> Enter a D to specify that the display file generated should display the file data in two columns.
> Enter a G to specify that the number of columns to use should be calculated, based on the number of fields to be shown.

Commitment Control: Enter a Y to tell the generator that the resulting program is to incorporate commitment control logic. Enter an N to ignore the option.

LR Indicator: Specify whether the generated program is to terminate with the LR indicator on. Enter Y for yes or N for no. For programs that are repeatedly called within an application, e.g. selection modules, it is best to specify N for this option and so leave the program active.

Shell Source file: Enter the name of the source file containing the template shells to be used to generate the new program. An entry of *DFT will use the default NUTIL shell file for the program type (*OPM or *ILE) being generated.

User merge source file/member: If you are merging user source into the generated program, specify the location of the user source. If the member specified does not exist then the generator enters SEU to allow you to add any functions. If the User Source member is left blank then this function is ignored. Refer to the *User Merge source* section later in this manual for further details

Convert ILE Eval: defines whether ILE RPG program source will use old "fixed format" RPG Operation Codes, or the equivalent EVAL free-format Operation Code:

- MOVEL will be replaced by Eval
- MOVEL (with Blank Padding) will be replaced by Eval
- MOVE (indicators) will be replaced by Eval
- MOVE will be replaced by Evalr
- MOVE (with Blank Padding) will be replaced by Evalr
- Z-ADD will be replaced by Eval
- Z-ADD (with Half-Adjust) will be replaced by Eval(h)
- ADD will be replaced by Eval
- ADD (with Half-Adjust) will be replaced by Eval(h)
- SUB will be replaced by Eval
- SUB (with Half-Adjust) will be replaced by Eval(h)
- MULT will be replaced by Eval
- MULT (with Half-Adjust) will be replaced by Eval(h)
- DIV will be replaced by Eval
- DIV (with Half-Adjust) will be replaced by Eval(h)

Usage of old RPG Operation Codes is (optionally) retained for Legacy support, but usage of EVAL and free-format is the preferred option.

Field Table: This parameter specifies a list of prefixes to be used on fields within the generated program for printer and display file fields. If the new (generated) field name already exists then the next character in the table is used and so on through the table until a unique name is found.

ILE Workfield prefix mode: This parameter defines how prefixes (from the Field Table) are to be applied

O (Overlay): The first character of the original database field is replaced with the selected character from the field table.

A (Append): The selected character, plus an underscore, is appended to the field name

Function keys available are as follows:

F3=Exit the program without generation. (Defaults are not saved)
F4=Exit the program without generation. The parameters entered are saved and will be loaded as defaults when the command is next used.
F6=Process the request.
F21=Display a command line window for command entry.

Modifying data field attributes

After you press the F6=Process function key, the field definitions (attributes) for the specified file will be determined. Depending on whether you requested 'modify field positioning and text' you may then be presented with a screen allowing you to do the following:

Override Column Headings
Resequence Non-Key Fields
Define the usage of the fields on the display
Remove fields from the resulting layout

To alter any field attributes, you should enter a 'Y' in the 'Modify field positioning and Text?' prompt on the GENPGM initial display.

SORT SEQUENCE OF FIELDS TO BE DISPLAYED

On the initial display you are able to change the sort sequence number and format identifier. The sort sequence will determine where the field is positioned in the output layout. The format identifier is used when you have a 'multi-screen' program that has too many fields for one display. Refer to program shells MNT_D2 and MNT_D3 for examples of multi-screen programs.

WARNING - If you make changes on this display, you *must* press enter to accept them before pressing the function key to generate the source. If you do not press enter first, it is assumed that you are ignoring the changes made.

The format identifier is used to separate which fields are to appear on each display when multiple record formats are required. By default this field is always set to '1' and would not normally change.

If the type of program to be generated contains multiple formats then you must nominate the record format that the field will appear on. For instance FLD1> relates to fields which have an identifier of '1', FLD2> relates to fields which have an identifier of '2' etc.

REMOVING FIELDS

You may remove a field from the resulting Display or Printer file, but you should be aware of the following considerations:

- You cannot remove a key field. If you do not wish a key field to be displayed, you should change the field attribute of the field to 'H' (hidden field).
- If the program is a data entry program, and the ADD function will be available to the user, you should not delete data fields from the layout. If you do not wish the user to enter input for a field, you should use field attribute 'H' to hide it.

COLUMN HEADINGS

The column headings are used to format the 'title' of the field, which will be displayed at the top of a report, the top of a subfile, or beside the field on the full screen record layout.

A report title will use all three column headings; A subfile title will use column heading 1 and 2 only. A window subfile title will use column heading 1 only. A full screen record layout will use as much as possible of the three column headings (concatenated to a maximum length of 23 characters).

In general, you should try to keep the column heading length as short as possible. When the generator is creating the formats, it will decide how much room is needed for a field by looking at the longest column heading for the field. If it is longer than the field, then the column heading determines the amount of space for the field.

FIELD ATTRIBUTES

Field attributes determine the usage of a field in a layout. The default usage is B (both input and output):

I=Input field only
O=Output field only
B=Both input and output
H=Hidden field. Will be included in the record format, but will not appear to the user.
C=Control (key) field. Used only in Window programs, to signify that the key will appear in the subfile, but the corresponding 'Position To' control field (in the subfile control record) will not be displayed

The attribute used in the resulting source member will be also be based on the type of program shell being used. For example, if you have specified 'B' or 'I' for a field to be used in a selection program, the generator will assume it to be an 'O' (since selection programs do not allow input).

EDIT CODES AND EDIT WORDS

The edit codes and words used within the file definition are also retrieved and used. You may change the edit code, but not the edit word.

Of special note is edit code Y (for use on date fields), which has special processing considerations within the program generator. The generated program always assumes that a date is displayed in Job date format, but is stored in Year/Month/Day format. Program code is automatically inserted in the generated program to perform reversal and editing functions for every 'date' field.

```
GEN002D1                        Modify data fields
Select
Type options, press Enter.
2=Change  4=Remove
Program IM4003 Stock file maintenance          Based on shell MNT_D1

  Sort  Format    Field      Att Column Heading (1)   Column Heading (2)
    10    1     K SRFRN       B  Franchise            Code
    30    1       SRPRT       B  Part                 Number
    40    1       SRBRN       B  Branch               Code
2   50    1       SRCAT       B  Category             Code
    60    1       STOPS       B  Opening              Stock


Position list to:
    10

F3=Exit F6=Create Source F12=Previous F21=Command Window F19=Submit Create
```

If you requested data field modify, you will be shown screen GEN002D1. This shows a list of fields in the specified file and allows you to manipulate the outcome of the source generation. The fields that can be modified on this panel are as follows:

Select: Type in a 4 to remove the field; the field will not be processed in the source generation. Note that key fields cannot be deleted. Type in a 2 to display and allow change of the full attributes for the field.

Sort Sequence: This will resequence the fields as they will be displayed in a subfile or on a report. To resequence a field type a new sequence number over the existing number. So to resequence field 10 to be displayed after field 30 you type in a number greater than 30 (that is not currently used) and less than the sequence number on the next field after 30.

Format: For a multi format display, type in the format number you wish the field to be displayed on.

Field Attribute: This can be one of the following:

I= Input field only
O=Output field only
B=Both input and output. If the program allows file update, the field will be opened for update.
H=Hidden field. Will be included in the record format, but will not appear to the user.
C=Control (key) field. Used only in Window programs, to signify that the key will appear in the subfile, but the corresponding 'Position To' control field (in the subfile control record) will not be displayed.

The attribute used in the resulting source member will be also be based on the type of program shell being used. For example, if you have specified 'B' or 'I' for a field to be used in a selection program, the generator will assume it to be an 'O' (since selection programs do not allow input).

Output and Hidden fields are still processed within the program, but they cannot be accessed by the user.

Once you have fully completed you field definitions, press F19 to generate the source via a batch job submission, or F6 to generate the source interactively. If the generation is performed in batch, it uses the job description defined on the GENPGM prompt. F12 will return you to the generator prompt and F3 will exit the GENPGM command. F21 will display a command line window for command entry.

```
GEN002D2                        Modify data fields
Change
Type options, press Enter.

File...............: *LIBL   / ITWSTKRV       Record format......: ITRSTKRV
                    Stock master file (work)

Field name.........: SRCAT
Description........: Category code
Field type.........: A                          Field length.......:    2
Decimal precision..:                            Decimal digits.....:
Output buffer posn.:    23

Field sequence No..:    50

Column heading 1...: Category                   Display format No..: 1
Column heading 2...: Code                       Field attribute....: B
Column heading 3...:

Edit code..........:
Edit word..........:

F3=Exit   F11=Remove field   F12=Previous
```

If you requested full attribute detail for a field (by typing a ‘2’ on the previous screen) you will be shown screen GEN002D2, which presents you with full field detail for the field. You can change the following:

Sort Sequence: This will resequence the fields as they will be displayed in a subfile or on a report. To resequence a field type a new sequence number over the existing number. So to resequence field 10 to be displayed after field 30 you type in a number greater than 30 (that is not currently used) and less than the sequence number on the next field after 30.

Column Headings: Column Heading 1, 2 and 3 are used to create headings on printer files and subfiles. Column Heading 1 is normally used when displaying full details for a specific file record.

Format: For a multi format display, type in the format number you wish the field to be displayed on.

Field Attribute: This can be one of the following:

I=Input field only
O=Output field only
B=Both input and output. If the program allows file update, the field will be opened for update.
H=Hidden field. Will be included in the record format, but will not appear to the user.
C=Control (key) field. Used only in Window programs, to signify that the key will appear in the subfile, but the corresponding 'Position To' control field (in the subfile control record) will not be displayed.

The attribute used in the resulting source member will also be based on the type of program shell being used. For example, if you have specified 'B' or 'I' for a field to be used in a selection program, the generator will assume it to be an 'O' (since selection programs do not allow input).

Output and Hidden fields are still processed within the program, but they cannot be accessed by the user.

Edit Code: For numeric fields you can enter the edit code to be used when displaying the field. Note that edit code Y (date edit) has special meaning in the program source generation. A field with edit code Y is assumed to be a date field that is stored in Year/Month/Day format, but displayed in system format and program code is automatically created to perform this function.

You can also request field delete from this screen (F11), but it is only allowed if the field is not a key field.

Maintain program templates

Program Templates are maintained via the WRKMBRPDM (work with members via PDM) command. Take care when maintaining templates, as syntax checking must be switched off to allow the keywords and opcodes to be used in source members.

Maintain program template selections

Select this option if you have created any new program templates.

```
GEN007D1                  Shell Selection - Maintenance
Select

     Shell                                                Shell   Keys   Mult
     Type        Description                              Type  Rqd  Rst  Fmt

     INQ         Inquiry with subfile selection              2   Y    N    N
     INZ         Initialise all Non-Key fields               6   Y    N    N
     MENU        iSeries Menu                                7   N    N    N
     MNT_D1      Inquiry/Maintenance Function (Formats=1)    1   Y    N    N
     MNT_D2      Inquiry/Maintenance Function (Formats=2)    1   Y    N    Y
     MNT_D3      Inquiry/Maintenance Function (Formats=3)    1   Y    N    Y
     MNTCTL      Control File Inquiry/Maintenance            1   Y    N    N
     MNTDTAARA   Maintain Data Area                          1   N    N    N
     MNTINQSEL   Select/Inquiry/Maintenance                  1   Y    N    N
     MNTPGM      Maintenance/Inquiry without prompt          1   Y    N    N

5-Select   3-Shell Source Processed
Key: INQ

F3=Exit
```

The following information needs to be defined for each shell type:

Shell Type: defines the type of program that will be created from the shell:

1 Maintenance program
2 Inquiry program
3 Selection program
4 Prompt program
5 Report program
6 Processing program
7 Menu

Keys Required?: When generating a program, does the user have to specify a keyed file as the file to be processed?

Restricted Keys: Must the user specify at least one restricted key?

Multi Format: Is the Display file a multi format file?

Maintain template shell selections

This screen will appear if Selection details are to be added or amended. It allows you to further define the members attached to a Program Template. This allows you to use the program generator to create display files or printer files as a separate function, without having to create a corresponding RPG program.

```
GEN008D1                Shell Selection Detail - Maintenance
Select
Shell Type: INQ          2 Inquiry with subfile selection

     Source      Source
     Member      Attribute

     INQ           *RPG
     INQD          *DSPF


5-Select
Key: INQ

F3=Exit  F12=Previous
```

For the shell type specified, you must define the templates that are to be used when creating user source. In the above example, the shells INQ (type *RPG) and INQD (type *DSPF) will be used by the GENPGM command when creating source for an INQ shell request.

There are certain rules as to which attribute types are allowed for certain shell types:

Shell Type	Description	Attribute types allowed
1	Maintenance	*RPG, *DSPF, *PRTF, *CLP
2	Inquiry	*RPG, *DSPF, *CLP
3	Selection	*RPG, *DSPF, *CLP
4	Prompt	*RPG, *DSPF, *CLP
5	Report	*RPG, *PRTF, *CLP
6	Processing	*RPG, *PRTF, *CLP
7	Menu	*CLP, *DSPF, *PNL

So, for example, a maintenance shell can have a CL program, both a Display and a Printer file as well as an associated RPG program. It can have any or all of these.

Alter system defaults

The program generator is shipped with the following defaults which may be changed (using the GENPGMFTS command) to suit your installation:

```
          GENPGM System Defaults (GENPGMDFTS) Prompt

 Enter the following:

Date format to display . . . . . DATFMT         *SYSVAL
DSPF source mbr name suffix  . . DSPF           D
Standard Display width . . . . . DSPWDT         80
PRTF source mbr name suffix  . . PRTF           P
Standard Printer file width  . . PRTWDT         132
CLP source mbr name suffix . . . CLP            C
Edit code for 8 digit dates  . . EDTDATE        8____
Edit mask for date display . . . EDTMASK        *YMD
```

Parameter Definitions

The *Date Format* parameter lets you alter the way 6 and 7 digit dates will be displayed and entered in generated programs. The program generator assumes that dates are stored in Year/Month/Day format, but the way they are displayed is dependent upon this parameter.

The *DSPF Source member name suffix* parameter defines the character that will be appended to the program name in order to define a related Display file. The default is ‘D’, but the entry can be any letter of the alphabet *provided that* it is not the same as the letter defined for the Printer file suffix name.

As an example, if the default is used, when the program generator creates a program named GL0001, a related display file will be named GL0001D.

The *Standard display width* parameter defines the maximum allowable length of a display line. The default is 80 but can be changed. When the program generator attempts to generate a subfile it will use this parameter when determining the number of fields it can fit into the subfile line.

The *PRTF Source member name suffix* parameter defines the character that will be appended to the program name in order to define a related Printer file. The default is ‘P’, but the entry can be any letter of the alphabet *provided that* it is not the same as the letter defined for the Display file suffix name.

As an example, if the default is used, when the program generator creates a program named GL0001, a related printer file will be named GL0001P.

The *Standard printer line width* parameter defines the maximum allowable length of a printed detail line. The default is 132, but can be changed. When the program generator attempts to generate a printer file detail line definition it will use this parameter when determining the number of fields it can fit onto the line.

The *CLP Source member name suffix* parameter defines the character that will be appended to the program name in order to define a related CL program. The default is ‘C’, but the entry can be any letter of the alphabet.

As an example, if the default is used, when the program generator creates a program named GL0001, a related CL program will be named GL0001C.

The *Edit code for 8 digit dates* (EDTDATE) parameter defines which user-defined edit code will be used to define 8 digit dates. Program Generator support for 8 digit dates is based on the code you enter here.

The specified edit code will be initialised by this command to the edit mask defined in the INTMASK parameter.

The default is edit code 8. If you specify *IGNORE, no Program Generator support of 8 digit dates will be installed.

The *Edit mask for 8 digit dates* (INTMASK) parameter defines the character string that will be used to edit the date field for display purposes.

The default value is *YMD, which will display the date as nnnn/nn/nn. Other values are:

*ISO	nnnn-nn-nn
*EUR	nnnn.nn.nn
*USA	nn/nn/nnnn
*JIS	nnnn-nn-nn
*UK	nn/nn/nnnn

The specified edit code will be used to create the specified user-defined edit code. Please ensure that you are not already using the edit code for any other purpose.

User specified Merge source code

User Merge Source

The program generator prompt screen allows the entry of a User Merge Source member parameter. This parameter tells the generator that there is some additional *user-defined* RPG source code that is to be merged with the shell template source to produce the final source code.

The GENPGM command allows you to include RPG specification types H*, F, E, D, I, C and O. All specification types may also be included as comment lines. Any H specifications must be comment lines, since only one header specification is allowed and all template programs already have an H specification defined.

Below is an example of each type of specification and an explanation of how it works:

- ⇒ H* Specifications. Header specs may only be added if they are comment lines. These lines will be added to the resulting program after any existing H specifications whether they are comments or not.
- ⇒ F specifications will be included after the existing F Specifications.
- ⇒ E specifications (OPM only) will be included after the existing E Specifications.
- ⇒ D specifications (ILE only) will be included after the existing F specifications.
- ⇒ Compile time arrays will include the compile time data after any existing data at the end of the program.
- ⇒ I specifications will be included after the existing I Specifications.
- ⇒ C specifications are included in the resulting program at predefined points marked in the template program.
- ⇒ O specifications will be included after the existing O specifications.

Creating a Merge Source member

The Create Merge Source command (NCRTRMRGSRC) allows you to create a Merge Source member from a specified program shell. This allows you to then define your own 'merge' code to be added to a subsequent program generation (via the GENPGM command).

```
                  Create RPG merge source member (NCRTMRGSRC)

Type choices, press Enter.

Shell type to process  . . . . . SHELLTYPE      *OPM
From shell source file . . . . . FROMFILE       *DFT
  Library  . . . . . . . . . . .                  *LIBL
From shell template name . . . . FROMSHELL
To RPG merge source file . . . . TOFILE         *DFT
  Library  . . . . . . . . . . .                  *LIBL
To RPG merge source member . . . TOMBR          *FROMSHELL
```

Shell type to process: The shell type specifies which RPG format is used in the program shell template. The possible values are:

*OPM - The program shell template is defined in RPG/400 source format
*ILE - The program shell template is defined in ILE RPG/400 source format

From shell source file: Specifies the name of the shell source file that contains the program shell template being used to extract the user exit points from:

*DFT - The default shell source file for the specified shell type is used. For *OPM shell type this is NUTIL/QSHLSRC400; for *ILE the default is NUTIL/QSHLSRCLE.
shell source file name - specify the name of the shell source file to read.

From shell: Specify the name of the program shell template that is to be used to extract the user exit points from.

To RPG Merge source file: Specifies the name of the merge source file to contain the RPG merge source member that will be created by this command:

*DFT - The default merge source file for the specified shell type is used. For *OPM shell type this is QRPGSRCM; for *ILE the default is QRPGLESRCM.
merge source file name- specify the name of the merge source file to be used.

RPG merge source member: Specifies the name of the merge source member to be created. If this member already exists it will be replaced:

*FROMSHELL - Corresponding from-file and to-file member names will be used.
merge source file name - specify the name of the merge source member to be created.

Layout of a Merge Source member

Each section of user-defined source code in the Merge Source member is grouped using a set of 'work' BEGSR and ENDSR Operation codes; the 'subroutine' name matches the name of the pointer in the template program.

Below is an example of a Merge Source entry point in a template shell:

```
     C           #INIT     BEGSR
USR*: *   ====> Insert Program Initialisation here <====
USR*: *   ====> Insert Program Initialisation here <====
     C*                                                    USR>>
     C*  ...Load subfile Page Size...                       USR>>
     C                     Z-ADD10        @MAXSZ  30       USR>>
      *
     C                     ENDSR
```

All source code lines between the BEGSR and the ENDSR delimiters will be included at the point specified by the pointer #INIT in the template program. The 'work' subroutines used in the Merge Source member are for grouping control only – they are not transferred to the generated source program. Only the source code within the work BEGSR/ENDSR delimiters is actually transferred.

Any 'real' subroutines that appear after the precompile marker '`ADD::`' (which may appear in columns 1-5 of the source code) will be added to the generated source program, including the required BEGSR and ENDSR statements, after the last C specification.

O specifications will be added to the generated source program after all other C specifications. This includes any user-defined C specifications.

Comment lines specified by simply an asterisk will *not* be included in the resulting program.

Controlling Markers

The following markers are used to identify the points where the user-defined code stored in the Merge Source member is to be included in the generated source program:

```
H:::: *  <<==== Additional H specifications begin here ====>>
F:::: *  <<==== Additional F specifications begin here ====>>
E:::: *  <<==== Additional E specifications begin here ====>>
I:::: *  <<==== Additional I specifications begin here ====>>
USR:: *          #INIT
USR*: *   ====> Insert Program Initialisation here <====
USR*: *   ====> Insert Program Initialisation here <====
     C*                                                    USR>>
     C*  ...Load subfile Page Size...                      USR>>
     C                     Z-ADD10        @MAXSZ  30       USR>>
O:::: *  <<==== Additional O specifications begin here ====>>
ADD:: *  <<==== Add User Defined Subroutines here      ====>>
```

If the literal '`USR>>`' appears in the comment area of the C specification and a Merge Source member has been specified on the GENPGM prompt, then all '`USR>>`' source lines will be ignored from the template program, as they will have previously been included from the Merge Source member.

Creating a program template

This is an advanced function. You should not attempt to create or process your own program templates unless there is a special need to do so.

You should not modify the program shells supplied with NUTIL. Instead, you should take a copy of one supplied and use the copy as a base for a new shell. The copy must not be put back in to the NUTIL shell source file as this may be reloaded when installing a new release of the product.

Program Templates are normal source specifications, which consist of additional special substitution values recognised by the Program Generation Utility. The purpose of this section of the manual is to allow you to better understand how they are written.

Once you have an understanding of Program templates you will then be able to alter the supplied set of templates to suit your organisations standards or even define your own.

If you create a new program template then you must select option 12 of the Program Generator menu to add it to the valid types of selections and add the names of the template source members to the Selections detail file.

Precompile Labels

RPG Precompile Labels are markers used in the Template source code to define the processing level for Precompile Keywords. They are entered in positions 1 to 5 of the source line and take the following format:

Label	*Purpose*
`*>>>>`	Comment line only. Not carried forward to Generated source
`K>>>>`	Repeat Precompile Keyword for ALL key fields
`KD>>>`	Repeat Precompile Keyword for all DATE key fields
`KND>>`	Repeat Precompile Keyword for all NON-DATE key fields
`R>>>>`	Repeat Precompile Keyword for ALL Key fields *GT No of Keys supplied on Prompt (i.e., No of keys to be restricted)
`RND>>`	Repeat Precompile Keyword for all NON-DATE key fields *GT No of Keys supplied on Prompt (i.e., No of keys to be restricted)
`R<<<<`	Repeat Precompile Keyword for ALL Key fields *LE No of Keys supplied on Prompt (i.e., No of keys to be restricted)
`F>>>>`	Repeat Precompile Keyword for ALL fields
`FD>>>`	Repeat Precompile Keyword for date/time/timestamp fields only
`FND>>`	Repeat Precompile Keyword for all NON-DATE fields
`N>>>>`	Repeat Precompile Keyword for ALL Non-Key fields
`NND>>`	Repeat Precompile Keyword for all NON-DATE Non-Key fields
`CMT>>`	Include code if Commitment Control requested
`LDA>>`	Include *LDA code if requested
`LR>>>`	Include code if LR to be SETON
`OV>>>`	Include code if Override to a temporary update file
`RST>>`	Include code if Restricted Keys exist
`H::::`	Include User Defined Program Description here
`F::::`	Include User Defined F Specifications here
`E::::`	Include User Defined E Specifications here
`I::::`	Include User Defined I Specifications here
`O::::`	Include User Defined O Specifications here
`USR::`	Include User Defined Entry Point here
`USR*:`	User Defined Entry Point (Text Only)
`ADD::`	Include User Defined Subroutines here

For the above table, a *date* field has one of the following characteristics:

- The field is defined in the database file with EDTCDE(Y) as the edit code
- The field is defined in the database file with the same edit code that was set as the *8 digit date edit code* when installing the program generator (the default is EDTCDE(8)).
- The field is defined in the database file as having the date data type (data type L)

A *time* field is one that is defined in the database file as being data type(T)

A *timestamp* field is one that is defined in the database file as being data type(Z)

Note that date, time, and timestamp data type fields can only be processed correctly by the *ILE type shells.

DSPF Precompile Labels are markers used in the Template source code to define the processing level for Precompile Keywords. They are entered in positions 1 to 5 of the source line and take the following format:

Label	*Purpose*
`*>>>>`	Comment line only. Not carried forward to Generated source
`SFL>>`	Insert field names for subfile record. Also returns column headings as 80 bytes
`SFLU>`	Insert field names for SFL Update
`SFLW>`	Insert field names for SFL Work file (used by MNTSFL)
`CTL>>`	Insert field names for subfile control record (excluding Restricted Keys)
`CTL+>`	Insert field names for subfile control record (including Restricted Keys)
`CTL<<`	Insert field names for restricted fields
`CTLU>`	Insert field names for SFLCTL update
`ALKY>`	Insert all KEY field names for Input format
`KEY>>`	Insert Non-Restricted KEY field names for Update/Add format
`KEY<<`	Insert Restricted KEY field names for Update/Add format
`KEYR>`	Insert Non-Restricted KEY field names for Input format
`KEYR<`	Insert Restricted KEY field names for Input format
`FLD>>`	Insert NON-KEY fields for either Input/Output format
`FLD1>`	Insert NON-KEY fields for either Input/Output format 1
`FLD2>`	Insert NON-KEY fields for either Input/Output format 2
`FLD3>`	Insert NON-KEY fields for either Input/Output format 3
`FLD4>`	Insert NON-KEY fields for either Input/Output format 4
`FLD5>`	Insert NON-KEY fields for either Input/Output format 5
`FLD6>`	Insert NON-KEY fields for either Input/Output format 6
`FLD7>`	Insert NON-KEY fields for either Input/Output format 7
`FLD8>`	Insert NON-KEY fields for either Input/Output format 8
`FLD9>`	Insert NON-KEY fields for either Input/Output format 9

Note: `SFL>>, CTL<<, CTL>> and CTL+>` require the start line number and starting position to be specified.

PRTF Precompile Labels are markers used in the Template source code to define the processing level for Precompile Keywords. They are entered in positions 1 to 5 of the source line and take the following format:

Label	*Purpose*
`*>>>>`	Comment line only. Not carried forward to Generated source
`PRT>>`	Insert field names for Printer file detail record
`PRT<<`	Insert field names for restricted fields. Also returns column heading as 132 bytes

Note: `PRT<<` and `PRT>>` require the starting print position to be specified.

Precompile operation codes

Precompile Operation Codes appear in RPG specifications under the 'Operation Code' field and may be one of the following:

Code Insertion OpCodes

A section of code will be included for each field of the appropriate data type, to perform the relevant process:

<YDJ>	Reverse date from *YMD to *JOB format
<JDT>	Reverse date from *JOB to *YMD format
<CDJ>	Numeric Date field validation using *JOB form at

The following are valid for *ILE shells only:

<CVD>	Convert numeric date to date data field
<CVM>	Convert numeric time to time data field
<CV8>	Convert numeric 6P0 date to numeric 8P0 date
<CDT>	Date data type field validation using *JOB format
<CTM>	Time data type field validation
<CTS>	Timestamp data type field validation

Code Inclusion OpCodes

One line of code will be included for each field:

<CPY>	Copy ALL fields, according to data type
<CPA>	Copy all NON-DATE fields, according to data type
<INZ>	Initialise ALL fields, according to data type
<*NE>	Insert first RPG operand as IFNE, followed by ORNE
<*EQ>	Insert first RPG operand as IFEQ, followed by ANDEQ
<*NQ>	Insert first RPG operand as IFNE, followed by ANDNE
<*OR>	Insert first RPG operand as IFEQ, followed by OREQ

The following are valid for *ILE shells only:

<*EQLE>	Insert first RPG operand as Ifne, followed by Andeq
<*NELE>	Insert first RPG operand as Ifne, followed by Orne
<*NQLE>	Insert first RPG operand as Ifne, followed by Andne
<*ORLE>	Insert first RPG operand as Ifeq, followed by Oreq

Precompile keywords

Precompile Keywords are variables used in the Template source to define a literal or field name which appears in Factor1, Factor2 or the Result field of RPG specifications and may be one of the following:

<FFLD>	Key field name as FKEY01, FKEY02, FKEY03 etc
<fFLD>	Same as <FFLD>, but date/time/timestamp fields use WKEY
<IFLD>	Field name of current input file field
<iFLD>	Same as <IFLD>, but date/time/timestamp fields use WKEY
<NFLD>	Key field name as NKEY01, NKEY02, NKEY03 etc
<DFLD>	Field name of current display field
<dFLD>	Same as <DFLD>, but date/time/timestamp fields use WKEY
<HFLD>	Hidden field used by Precompile Label <SFLW>
<WFLD>	Work field used by Precompile Label <SFLW>

Field names

<*ALL9>	*ALL'9'
<*LOVAL>	Low Values
<*HIVAL>	High Values
<*BLANK>	*Blank or *Zero depending on field type

Literals

<*IN>	replaced by the appropriate error indicator and is valid only in the RESULT field

Error Indicator

For example:

```
......L0N01N02N03Factor1+++OpcdeFactor2+++ResultLenDHHiLoEqComment+++++
      *
      *  ...Initialise ALL display fields...
F>>>>C                    <CPY><*BLANK>  <DFLD>
      *
      *  ...Copy Display Fields to Output File...
F>>>>C                    <CPY><DFLD>    <IFLD>
*>>>> *
*>>>> *  ...Reverse any date fields defined with EDTCDE(Y)...
F>>>>C           <IFLD>   <JDT><IFLD>
     C                    WRITERECORD
```

Precompile variables

RPG Source Code: The following variables denoted by <> are supported by the Program Generator for source attribute RPG and may appear anywhere within a line of source code:

<PGMN>	6 Char Program name as entered on Prompt
<TITLE>	Program title as entered on Prompt
<IFFILE>	File-Id on which program based as entered on Prompt
<UFFILE>	Replaced by dummy file @UPFILE
<RCDFMT>	Record format name of <IFFILE>
<GENDTE>	Source creation date in System date format
<MSGF>	Message file holding error messages

For *OPM program generation, File name and Record Format name is restricted to a maximum length of 8 characters (this is a language restriction, not a NUTIL restriction). Names longer than 8 characters will be truncated. *ILE program generation allows 10 character names.

DSPF Source Code: The following variables denoted by <> are supported by the Program Generator for source attribute DSPF and may appear anywhere within a line of source code:

<PGMN>	6 Char Program name as entered on Prompt
<IFFILE>	File-Id on which program based as entered on Prompt
<TITLE>	Program title as entered on Prompt
<RCDFMT>	Record format name of <IFFILE>
<GENDTE>	Source creation date in System date format
<CHDR11>	Bytes 1 thru 34 of a 78 byte Column Heading (1)
<CHDR12>	Bytes 35 thru 68 of a 78 byte Column Heading (1)
<CHDR13>	Bytes 69 thru 78 of a 78 byte Column Heading (1)
<CHDR21>	Bytes 1 thru 34 of a 78 byte Column Heading (2)
<CHDR22>	Bytes 35 thru 68 of a 78 byte Column Heading (2)
<CHDR23>	Bytes 69 thru 78 of a 78 byte Column Heading (2)
<CHDR31>	Bytes 1 thru 34 of a 78 byte Column Heading (3)
<CHDR32>	Bytes 35 thru 68 of a 78 byte Column Heading (3)
<CHDR33>	Bytes 69 thru 78 of a 78 byte Column Heading (3)

PRTF Source Code: The following variables denoted by <> are supported by the Program Generator for source attribute PRTF and may appear anywhere within a line of source code:

<PGMN>	6 Char Program name as entered on Prompt
<IFFILE>	File-Id on which program based as entered on Prompt
<TITLE>	Program title as entered on Prompt
<RCDFMT>	Record format name of <IFFILE>
<GENDTE>	Source creation date in System date format
<CHDR11>	Bytes 1 thru 34 of a 132 byte Column Heading (1)
<CHDR12>	Bytes 35 thru 68 of a 132 byte Column Heading (1)
<CHDR13>	Bytes 69 thru 102 of a 132 byte Column Heading (1)
<CHDR14>	Bytes 103 thru 132 of a 132 byte Column Heading (1)
<CHDR21>	Bytes 1 thru 34 of a 132 byte Column Heading (2)
<CHDR22>	Bytes 35 thru 68 of a 132 byte Column Heading (2)
<CHDR23>	Bytes 69 thru 102 of a 132 byte Column Heading (2)
<CHDR24>	Bytes 103 thru 132 of a 132 byte Column Heading (2)
<CHDR31>	Bytes 1 thru 34 of a 132 byte Column Heading (3)
<CHDR32>	Bytes 35 thru 68 of a 132 byte Column Heading (3)
<CHDR33>	Bytes 69 thru 102 of a 132 byte Column Heading (3)
<CHDR34>	Bytes 103 thru 132 of a 132 byte Column Heading (3)

CLP Source Code: The following variables denoted by <> are supported by the Program Generator for source attribute CLP and may appear anywhere within a line of source code:

<PGMN>	6 Char Program name as entered on Prompt
<TITLE>	Program title as entered on Prompt
<IFFILE>	File-Id on which program based as entered on Prompt
<UFFILE>	Replaced by dummy file @UPFILE
<RCDFMT>	Record format name of <IFFILE>
<GENDTE>	Source creation date in System date format
<MSGF>	Message file holding error messages

Source directives for the Pre Compile processor

Instructions for the compile pre-processor may be imbedded at the commencement of a source member using special types of comment lines. Each directive or override may span multiple source lines, connected by a '+' at the end of each line. These instructions are referred to as 'source directives'.

Types of source directives

Source directives can be of four types:

Type	*Supplied 'Pattern'*
Compiler directive	'*Z: '
Pre Compile override	'*O: '
Pre Compile directive	'*B: '
Post Compile directive	'*A: '

Note that the patterns shown in this document are the default patterns supplied with NUTIL. They may be different at your installation, as they can be altered using menu UTLINS, option 5).

COMPILER DIRECTIVES

Compiler Directives specify extra parameters to be added to the relevant CRT... command. For example:

... Using directives in RPG source

```
     ****************************************************************
     *  Compiler Directives                                         *
     *Z: CRTRPGPGM  GENOPT(*OPTIMIZE)
     ****************************************************************
```

... Using directives in DDS source

```
     ****************************************************************
     *  Compiler Directives                                         *
     *Z: CRTPF  MAXMBRS(1) SIZE(*NOMAX)
     ****************************************************************

    A****************************************************************
    A*  Compiler Directives                                         *
    A*Z: CRTDSPF RSTDSP(*YES) DFRWRT(*NO) SHARE(*YES)    +
    A*Z:         SHARE(*YES)
    A****************************************************************
```

... Using directives in CL source

```
    /****************************************************************/
    /*  Compiler Directives                                         */
    /*Z: CRTCLPGM  OPTIMIZE(*YES) RMVOBS(*ALL)                      */
    /****************************************************************
```

... Using directives in CMD source

```
/*************************************************************/
/*  Compiler Directives                                      */
/*Z: CRTCMD    ALLOW(*INTERACT *IPGM *EXEC) PRDLIB(ACCTSLIB)  +     */
/*Z:             PGM(MTHENDRUN)                             */
/*************************************************************
```

Note: Only 1 compiler directive may exist within a source member and the maximum length of the directive is 300 characters.

Compiler directives are loaded into the CRT... command prior to the re-creation job being submitted.

PRE COMPILE OVERRIDES

Precompile Overrides specify object overrides to be performed before the actual compile takes place. This allows you to compile a program without the actual object existing. This is particularly useful for an RPG program which double-defines a file, once as input and once as update. This avoids the requirement of having 'dummy' copies of a file to satisfy externally described files during a compilation.

Note: More than 1 pre compile override may exist, but the maximum length of each override is 300 characters.

```
*************************************************************
*  Compiler Directives                                      *
*Z: CRTRPGPGM PUBAUT(*NONE) GENOPT(*OPTIMIZE)
*  Compiler Overrides                                       *
*O: OVRDBF    FILE(@UPFILE) TOFILE(SYSMST00)
*O: OVRDBF    FILE(WRKDATA) TOFILE(SYSMST00)
*************************************************************
```

Pre-Compile overrides are processed at the commencement of the submitted re-creation job.

PRE COMPILE DIRECTIVES

Precompile Directives specify command requests to be executed before the actual compile is run. For instance, you can create objects in QTEMP necessary for the program to compile.

Note: More than 1 pre compile directive may exist, but the maximum length of each directive is 300 characters.

```
*****************************************************************
*  Compiler Directives                                          *
*Z: CRTRPGPGM PUBAUT(*NONE) GENOPT(*OPTIMIZE)
*  Pre-Compile Processing                                       *
*B: CRTDUPOBJ OBJ(QADSPOBJ) FROMLIB(QSYS) OBJTYPE(*FILE) +
*B:             TOLIB(QTEMP)
*B: ADDLIBLE  USERLIB  *LAST
*****************************************************************
```

Pre-Compile directives are processed prior to processing the relevant CRT... command.

POST COMPILE DIRECTIVES

Post compile Directives specify command requests to be executed after the actual compile is run. For instance, you may want to grant object authority to a specific user or duplicate the object into another library. Post compile directives will only be processed if the compile was successful.

Note: More than 1 post compile directive may exist, but the maximum length of each directive is 300 characters.

```
*****************************************************************
*  Compiler Directives                                          *
*Z: CRTRPGPGM PUBAUT(*NONE)
*  Pre-Compile Processing                                       *
*B: CRTDUPOBJ OBJ(QADSPOBJ) FROMLIB(QSYS) OBJTYPE(*FILE) +
*B:             TOLIB(QTEMP)
*  Post-Compile Processing                                      *
*A: CHGPGM    PGM(TESTX) RMVOBS(*ALL)
*A: GRTOBJAUT OBJ(TESTX) OBJTYPE(*PGM) USER(QPGMR) AUT(*ALL)
*****************************************************************
```

Post-Compile directives are processed following the successful completion of the relevant CRT... command.

If the CRT... command terminates abnormally the Post-Compile directives are not processed.

The NCODFT data area

A control data area named NCODFT exists in library NUTIL and is used to hold the default patterns for the various directives/overrides. Although not recommended, these can be altered to suit your installation by using the UTLINS menu option 5 – *'Define NCRTOBJECT processing attributes'.*

A summary of shells supplied

The following section outlines the basic function of each shell supplied with the Program Generator. These definitions refer to the iSeries style shells found in NUTIL/QSHLSRCLE and NUTIL/QSHLSRC400.

*ENTRY parameter lists

The parameter list for the program generated will depend on whether you have specified any 'restricted' keys for the program. If there any restricted keys, they will be appended to the standard parameter list for the program type being generated:

```
C           *ENTRY      PLIST
C                       PARM      standard parm 1
C                       PARM      standard parm 2
                          .
                          .
C                       PARM      restricted key 1
C                       PARM      restricted key 2
                          .
                          .
```

Restricted keys are used as input to the program and the value of each restricted key is 'locked' (i.e., it is not changed within the program).

Date fields in restricted keys

IMPORTANT NOTE: **Where a date field is passed in the restricted key *ENTRY parameter list, it is assumed to be in *JOB date format.**

Regardless of the way a date field is held on the database, the *ENTRY parameter list will always refer to the 6 digit work field loaded within the program. The field will contain the restricted key value in *JOBRUN format.

Help key processing

The processing of the HELP key is the same in all ‘display’ type programs. An attempt is made to display a source file member on the screen, using the NUTIL supplied program @HELP.

When the user presses the HELP or F1 key, the program passes control to the @HELP program. The name of the source member accessed is the same as the Screen ID (shown at the top of the screen display).

The source file is assumed to be *LIBL/QHLPTXT, which is defined as a constant in the program. If no helptext member is found, the user will be advised accordingly.

Commented code also exists in the generated Display file source member in order to assist you in enabling the HLPDOC method of displaying help text. Refer to the DDS manual for a discussion on using documents as help text.

QSHLSRCLE considerations

The shells in QSHLSRCLE take advantage of facilities that are provided in the ILE programming environment. Because of this they function slightly differently to QSHLSRC400 shells.

- RPG and CL source will create either program source (RPGBND or CLP) or module source (RPGLE or CLLE), depending on the function of the shell.

If the shell performs a standalone process it will create program source. An example of a standalone process is the MNT_D1 shell, which will maintain a specified database file.

If the shell performs a function that works in conjunction with other functions to provide a complete process it will create module source. An example of a function is the CLLE source member in the PRT_QRY shell. This module is bound together with the RPGLE module to perform the complete PRT_QRY process.

- The calls to the standard NUTIL support functions like @RVSDAT8 and @RTVDAT8 are bound calls. However they are dynamically bound calls via the NBNDDIR binding directory. Doing this means that the final program objects do not contain the bound modules for these support functions; when the program is called by the user they are accessed via the binding directory. For this reason you must include the NUTIL binding directory and service programs when distributing your user applications.

- The source attribute for program source is related to the type of object that the shell expects to create. For example the MNT_D1 program source has attribute RPGBND and the compiler directive in the source member is a CRTRPGBND command. If you change the source member attribute you should also ensure that the compiler directive relates to the new attribute:

Source Attribute	*Compiler directive*	*Object created*	
RPGLE	CRTRPGMOD	*MODULE	ILE
RPGMOD	CRTRPGMOD	*MODULE	ILE
RPGBND	CRTBNDRPG	*PGM	ILE
RPG	CRTRPGPGM	*PGM	OPM
CLLE	CRTCLMOD	*MODULE	ILE
CLP	CRTCLPGM	*PGM	OPM

NUTIL ILE Condition Handler procedure @ECHDL

If an ILE RPG procedure is called by an ILE CL procedure in the same activation group, and the caller is monitoring for status or notify messages, then the ILE CL caller may get control back prematurely because of a notify or status message that the ILE RPG procedure was trying to ignore.

This may mean that a global MONMSG CPF0000 in the CL caller program incorrectly gets control back from the RPG procedure, simply because of an “untrapped” informational message that was issued to the RPG - causing the program to terminate abnormally.

In order to ensure that error messages are handled at the correct level, the source code for ILE RPG programs generated by GENPGM includes logic that registers a default ILE Condition Handler (@ECHDL, which is in the NSRVPGM service program) to control this:

- Status, informational and warning messages are ignored *(action = handle)* and processing continues at the next machine instruction (normally the next RPG operation) following the error.
- Anything higher than a warning message is forwarded *(action = percolate)* to the *PSSR exception handler subroutine.

For more information on ILE Condition Handers, please refer to the IBM "ILE Concepts" manual SC41-5606-05. Also refer to the IBM manual "ILE RPG Programmers Guide" SC09-2507-03

DBTRIGGER - Database trigger processor

Source members generated

*RPGLE *PGM

This shell is available from QSHLSRCLE only.

This program functions as a database trigger program for the file specified. As supplied it performs no actual processing function; you will need to modify the generated source to define your trigger requirements.

The program can handle all trigger events, but the generated program will initially support only the following:

*INSERT	*AFTER	(after a record has been added to the file)
*UPDATE	*BEFORE	(before a record has been updated in the file)
*UPDATE	*AFTER	(after a record has been updated in the file)
*DELETE	*AFTER	(after a record has been deleted from the file)

However it is a very simple matter to alter the generated source to handle all trigger events.
Further information on database trigger programs can be found in the IBM iSeries Database Programming Guide.

Source Generation Notes
When requesting the source code generation for this program, use the name of the file for which the trigger program is to be created.

Restricted key function is not valid for this program.
You should consider the performance benefits that will be obtained by leaving the *INLR indicator off.

Program Modification Notes
The format of the *ENTRY parameter list is predefined within DB2/400 and must not be altered. There are sufficient comments within the program source to explain each of the fields and the values it can contain.

The data structure RcdImage provides the trigger program with the record image (file field data) feedback for the trigger event and the contents of the data structure will vary depending on where in the program it is accessed:

Subroutine SR1000 will be processed when the trigger event is a database ‘insert’ operation. The RcdImage data structure will contain an image of the record that has been added to the file.

Subroutine SR2000 will be processed when the trigger event is a database ‘delete’ operation. The RcdImage data structure will contain an image of the record that has been deleted from the file.

Subroutine SR3000 will be processed when the trigger event is a database ‘update’ operation.

If the event is *BEFORE the update has occurred, the RcdImage data structure will contain an image of the record that is being updated in the file.

If the event is *AFTER the update has occurred, the RcdImage data structure will contain an image of the record that has been updated in the file.

The loading of the data structure is handled by pointer arithmetic, which accesses the input parameter (data structure) DbTrgDs to load it correctly, based on the relevant trigger event. Due to the complex nature of the code, you should avoid altering any of the standard logic that manipulates the input parameters and data structures for this program. We recommend that you try to confine your modifications to the appropriate points in SR10000, SR2000 and SR3000.

Adding trigger events

The TrgEvent (trigger event) field is used to determine what trigger event caused the program to be called. The value will be either:

1=Insert
2=Delete
3=Update

The TrgTime (trigger time) field is used to determine at what point in the event the program was called:

2=Before the event has occurred
1=After the event has occurred

If you need to add an event that is not handled in the base version of this program (for example, an *INSERT *BEFORE) it is a simple matter of evaluating these 2 fields to control the processing logic of the event. You should also add the ADDPFTRG post-compiler directive at the top of the program source for the new event, so that it will get added to the database file when the program is created.

Compilation Notes
The RPGLE source will generate a program object. Following successful completion of the program compilation step, the 4 compiler directives will then add the trigger program to the database file.

CVT_DAT - Date fields conversion processing

Source members generated

*PRTF	
*CLLE	*MODULE
*RPGLE	*MODULE

This shell is available in QSHLSRCLE only.

This process is designed to assist you in converting your existing data files to new file layouts which use the new date and time data types. The program makes the assumption that your old data file

has dates in 6P0 or 7P0 fields stored in yymmdd format
has times in 6P0 fields stored in hhmmss format

and that your new data file

has dates in date data (type 'L') fields
has times in time data (type 'T') fields

Other than this redefinition of date and time field attributes, the format of the old and the new file must be exactly the same. Any date and time fields are checked for valid data before the copy is performed - an invalid date or time will be ignored and a *LOVAL will be stored in it's place (the system defined low value for date is '0001-01-01' and for time is '00.00.00')

A control report will list the total number of records that have been copied to the new file.

**ENTRY parameter list*

Field	*Type*	*Len*	*Usage*	*Description*
OLD_LIB	*CHAR	10	*IN	The name of the library which contains the old data file
NEW_LIB	*CHAR	10	*IN	The name of the library which contains the new data file

Source Generation Notes
When requesting the source code generation for this program, use the name of the new file (making sure you specify the name of the library containing the new version of the file) as the file to be processed.

Program Modification Notes
When copying the date to the new file, the conversion of date fields uses the standard IBM 'date windowing' technique for determining the century of the new date:

if the year is greater than or equal to 40 then the date is a 20th century date (1900)
If the year is less than 40 then the date is a 21st century date (2000)

If this logic is not correct for your application (for example your application may use dates earlier than 1940) then you will need too add your own code to the program to overlay the century portion of the date data with the correct century.

Compilation Notes
The RPGLE and CLLE source will generate modules and then a binding step in the CLLE source member will create the final program.

Program Operation
The program assumes that the data is still stored in the old data file (in the old library) and that the new file (in the new data library) is currently empty.

The program will read records from the old file, convert the data and write the converted record to the new file.

CVT_DAT8 - Date fields conversion processing (6/7 digit to 8 digit)

Source members generated
*PRTF
*RPGLE *PGM

This shell is available in QSHLSRCLE only.

This process is designed to assist you in converting your existing data files to new file layouts, where your conversion was from a 6 (or 7) digit date field to an 8 digit date field. The program makes the assumption that your old data file was in 6P0 (or 7P0) form and you have altered the field length to 8P0.

The edit code on the field in the old version of the file would have been EDTCDE(Y), but this must also be changed to EDTCDE(8), or the edit code you have specified as being your edit code for 8 digit dates, for the new file layout. The code for date data conversion is based upon detecting this field attribute so, if the field is not defined correctly, the conversion code will not be generated.

Other than this redefinition of date field attributes, the format of the old and the new file must be exactly the same.

The program will process all records in the specified file, updating the data in the altered date fields by adding the century to the year portion of the date.

A control report will list the total number of records that have been copied to the new file.

Program Modification Notes
When updating the date data in the new file, the conversion of date fields uses the standard IBM 'date windowing' technique for determining the century of the new date:

Years in the range 28-99 will be converted to 1928-1999
Years in the range 00-27 will be converted to 2000-2027

If this logic is not correct for your application (for example your application may use dates earlier than 1928) then you will need too add your own code to the program to overlay the century portion of the date data with the correct century.

Compilation Notes
The RPGLE source will generate an *ILE bound object program.

Program Operation
The program assumes that you have re-created the file with the new date edit code and that the data has been copied into the new file.

The program will read records in the new file, convert the date data from 6 digits (*YMD) to 8 digits (*ISO) and update the record.

INQ - File Inquiry

This program functions as a general all-purpose file inquiry.

Source members generated

*DSPF
*RPG

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering a '1' beside one or more records shown the user can then display full detail for the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

Command Function Keys Enabled

F3=Exit Program
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D2 - Display

This screen shows the 'full' record detail for every record selected by the user.

If enter is pressed, and multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F12=Return to Select screen
HELP=Display help text

INQ_PMT - File Inquiry via key prompt

This program differs from the standard INQ shell in that the file key is entered via an initial prompt. The display screen then shows the record details for that file key value. The Page up and Page down keys allow the user to read backwards and forwards within the file data. This program functions as a general all-purpose file inquiry.

Source members generated

*DSPF
*RPG

Screen D1 - Select

The user will be presented with a prompt screen in which an entry is made for each of the key fields.

The entry is validated (valid record exists with this key) and if acceptable the program continues to screen D2. If the user presses the 'page down' key instead of the enter key, the program will use the key values entered in the prompt fields as a positional pointer for file read (SETLL then READ) and the program will continue to screen D2 and display the first record found. If no key values are entered, Page down will cause the first record in the file to be shown. The reverse logic is true for the Page up key, which will position the file for a reverse read (SETGT then READP).

Command Function Keys Enabled

ENTER=Use key as specific file record
PAGEDOWN=Use key as SETLL value for READ
PAGEUP=Use key as SETGT value for READP
F3=Exit Program
HELP=Display help text

Screen D2 - Display

This screen shows the 'full' record detail for the record selected by the user.

If Page down is pressed, the program will then display the next record in the file. If end of file is reached the program will start again at the beginning of the file.

If Page up is pressed, the program will then display the previous record in the file. If beginning of file is reached the program will start again at the end of the file.

Command Function Keys Enabled

F3=Exit Program
F12=Return to Select screen
HELP=Display help text
PAGEDOWN=Display next record in file
PAGEUP=Display previous record in file

INZ - Initialise non-key fields

Source members generated
*RPG
*PRTF

The program will process all records in the specified file and for each record found will set each field selected in the program generation (other than the file keys) to zeroes or blanks, depending on the type of field being processed.

A control total is printed on an audit report, showing how many records were actually processed.

The program generated from this shell is very handy for situations where you wish to reset some fields in a file. The generated source code can be modified to suit your specific needs. For example, where you wish to reset all status codes in all records.

In most cases you will need to modify the generated source program to include your own processing requirements. These should be placed in one of the following subroutines:

SR1000 Process file data

MENU - Basic Menu using a display file

Source members generated

*DSPF
*CLP

This shell generates a basic iSeries menu, defined using a display file and CLP program

In all cases you will need to modify the generated code to insert your user menu options (The source code shows you exactly where to locate your user options).

Source generation options
When requesting the source code generation for a menu, enter file name *NONE.

Source modification
After generating the source members, the main areas of change are as follows:

Display file
To find the location where menu options are inserted into the source, use SEU FIND string '<OPT>'

Replace the <OPT> locator with the menu option number to be displayed
Replace the <OPTTXT> locator with the descriptive text to be displayed for the option
Replace the <OPTCMD> locator with the command name to be processed by the option (this appears at the right of the menu option on the display)

These 3 fields must be defined for each menu option to be displayed.

This is a normal display file layout and can be altered using the Screen Design Aid (SDA).

CL Program
To find the location where help text is inserted into the source, use SEU FIND string '<OPT>'

Replace the <OPT> locator with the menu option number relating to the help text
Replace the <TXT> locator with the descriptive text for the option
Replace the <CMD> locator with the command to be processed by the option

The program source shows how to code both an interactive call and a batch job submit

Processing code must be added for each menu option that has been defined.

MNU_PNL - Menu using a UIM panel group

Source members generated
*PNLGRP

This shell generates a basic iSeries menu, defined using the UIM Panel Group Definition language. The syntax for this language is specified in the IBM iSeries Application Display Programming Guide. An example of a menu generated from this shell is the main NUTIL menu.

In all cases you will need to modify the generated code to insert your user menu options (The source code shows you exactly where to locate your user options).

Source generation options
When requesting the source code generation for a menu, enter file name *NONE.

Source modification
After generating the PNLGRP source member, the main areas of change are as follows:

Define menu options

To find the location where menu options are inserted into the source, use SEU FIND string

'<<==== Menu Options are loaded here ====>>'

The :MENUI tag defines each menu option to be available on the menu

Replace the <OPT> locator with the menu option number to be displayed
Replace the <OPTTXT> locator with the descriptive text to be displayed for the option
Replace the <OPTCMD> locator with the command name to be processed by the menu option (this appears at the right of the menu option on the display)
Replace the <CMD> locator with the command to be processed by the option

A :MENUI tag must be defined for each menu option to be displayed.

You can split the menu into groups of options by enclosing a group of options in a separate :MENUGRP/:EMENUGRP area. By default your menu options are loaded into one group, which is titled 'Main menu options', but you can easily alter this.

Define help for menu options

To find the location where help text is inserted into the source, use SEU FIND string

'<<==== Help for Menu Options is loaded here ====>>'

The :HELP/:EHELP. tags define the help text to be displayed for a specific menu option.

Replace the <OPT> locator with the menu option number relating to the help text
Replace the <OPTTXT> locator with the descriptive text for the option
The actual help text is then keyed in on the line after the :P. tag

If you do not have a :HELP/:EHELP. group defined for a specific option on the menu the user will get a message 'Help text not available' if they try to retrieve it. Also, the general help text for the menu panel will display a message 'Help text is incomplete'.

MNT_D1 - File Maintenance/Inquiry

Source members generated

*DSPF
*RPG

This program operates as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If no parameters are passed, or if the parameter is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR3100 Editing specifically for *ADD (key field checks)
SR4100 Editing specifically for *UPDATE
SR4500 Editing for both *ADD and *UPDATE

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering the selection code beside one or more records shown the user can then display full detail for the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

The selection code to be used is dependent on the way the program was called. If it is a maintenance call, the selection code is '2'. If it is an inquiry call, the selection code is '1'.

By pressing F6 (available in Maintenance mode only) the 'Add New Record' function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record (Maintenance mode only)
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D2 - Display

This screen shows the 'full' record detail for every record selected by the user. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change non-key field data, or by pressing F11 delete the record.

If enter is pressed and the data is valid the file is updated (Maintenance mode). If multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record (Maintenance Update mode only)
F12=Return to Select screen
HELP=Display help text

MNT_D2 - File Maintenance/Inquiry via 2 formats

Source members generated

*DSPF
*RPG

This program operates in a similar way to MNT_D1, as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If no parameters are passed, or if the parameter is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

The main reason you would use this in preference to MNT_D1 is where there are more than 20 fields to be updated. MNT_D2 allows you to place fields on two screen formats, whereas MNT_D1 has only one 'processing' screen format. You define which fields from the record are to be displayed on the two processing screens in the 'Modify Data Fields' prompt when generating the source (the 'format' column).

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR3100 Editing specifically for *ADD (key field checks)
SR4110 Editing specifically for *UPDATE, screen D2
SR4210 Editing specifically for *UPDATE, screen D3
SR4500 Editing for both *ADD and *UPDATE, all screens

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering the selection code beside one or more records shown the user can then display full detail for the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

The selection code to be used is dependent on the way the program was called. If it is a maintenance call, the selection code is '2'. If it is an inquiry call, the selection code is '1'.

By pressing F6 (available in Maintenance mode only) the 'Add New Record' function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record (Maintenance mode only)
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D2 - Display first processing format

This screen shows the first part of the 'full' record detail for the record being processed. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change non-key field data, or by pressing F11 delete the record. If either enter or pagedown is pressed, the data entered on the screen will be edited and, if valid, the user will be shown screen D3.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record (Maintenance Update mode only)
F12=Return to Select screen
PAGEDOWN=Display screen D3
HELP=Display help text

Screen D3 - Display second processing format

This screen shows the second part of the 'full' record detail for the record being processed. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change non-key field data, or by pressing F11 delete the record. If enter is pressed and the data entered is valid the file is updated. If multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record (Maintenance Update mode only)
F12=Return to Screen D2
PAGEUP=Return to Screen D2
HELP=Display help text

MNT_D3 - File Maintenance/Inquiry via 3 formats

Source members generated

*DSPF
*RPG

This program operates in a similar way to MNT_D2, as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If no parameters are passed, or if the parameter is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

The main reason you would use this in preference to MNT_D2 is where there are more than 40 fields to be updated. MNT_D3 allows you to place fields on three screen formats, whereas MNT_D2 has only two 'processing' screen formats. You define which fields from the record are to be displayed on the two processing screens in the 'Modify Data Fields' prompt when generating the source (the 'format' column).

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR3100 Editing specifically for *ADD (key field checks)
SR4110 Editing specifically for *UPDATE, screen D2
SR4210 Editing specifically for *UPDATE, screen D3
SR4310 Editing specifically for *UPDATE, screen D4
SR4500 Editing for both *ADD and *UPDATE, all screens

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering the selection code beside one or more records shown the user can then display full detail for the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

The selection code to be used is dependent on the way the program was called. If it is a maintenance call, the selection code is '2'. If it is an inquiry call, the selection code is '1'.

By pressing F6 (available in Maintenance mode only) the 'Add New Record' function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record (Maintenance mode only)
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D2 - Display first processing format

This screen shows the first part of the 'full' record detail for the record being processed. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change non-key field data, or by pressing F11 delete the record. If either enter or pagedown is pressed, the data entered on the screen will be edited and, if valid, the user will be shown screen D3.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record (Maintenance Update mode only)
F12=Return to Select screen
PAGEDOWN=Display screen D3
HELP=Display help text

Screen D3 - Display second processing format

This screen shows the second part of the 'full' record detail for the record being processed. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change non-key field data, or by pressing F11 delete the record. If either enter or pagedown is pressed, the data entered on the screen will be edited and, if valid, the user will be shown screen D4.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record (Maintenance Update mode only)
F12=Return to screen D2
PAGEUP=Return to screen D2
PAGEDOWN=Display screen D4
HELP=Display help text

Screen D4 - Display third processing format

This screen shows the third part of the 'full' record detail for the record being processed. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change non-key field data, or by pressing F11 delete the record. If enter is pressed and the data entered is valid the file is updated. If multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record (Maintenance Update mode only)
F12=Return to Screen D3
PAGEUP=Return to Screen D3
HELP=Display help text

MNT_PMT - File Maintenance/Inquiry via key prompt

This program differs from the standard MNT_D1 shell in that the user modifies records one at a time, based on the key field entries made via an initial prompt. For each of the restricted keys the user is required to make an entry. The maintenance display then shows the specific record retrieved for the key entered.

Source members generated

*DSPF
*RPG

This program operates as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If no parameters are passed, or if the parameter is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR2100 Editing specifically for prompt screen
SR3100 Editing specifically for *ADD (key field checks)
SR4100 Editing specifically for *UPDATE
SR4500 Editing for both *ADD and *UPDATE

Screen D1 - Select

The user will be presented with a prompt screen in which an entry is made for each of the restricted keys requested. The entry is validated (valid record exists with this partial key) and if acceptable the program continues to screen D2.

By pressing F6 (available in Maintenance mode only) the 'Add New Record' function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record (Maintenance mode only)
HELP=Display help text

Screen D2 - Display

This screen shows the 'full' record detail for the record selected by the user. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change non-key field data, or by pressing F11 delete the record.

If enter is pressed and the data is valid the file is updated (Maintenance mode). The user will then be returned to the Select screen.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record (Maintenance Update mode only)
F12=Return to Select screen
HELP=Display help text

MNTCTL - File Maintenance/Inquiry via single work panel

Source members generated

*DSPF
*RPG

This program operates as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If no parameters are passed, or if the parameter is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

The main difference between this program and MNT_D1 is that all processing is performed on the one screen display (the user performs maintenance at the bottom of the screen format). As such, the use of the program is limited to data files that have only a short record format, such as control files.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR2130 Editing for both *ADD and *UPDATE

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering the selection code beside one of the records shown the user 'pulls down' the record to the bottom of the screen for processing. Only one record may be selected at a time.

The selection code to be used is dependent on the way the program was called. If it is a maintenance call, the selection code is '2'. If it is an inquiry call, the selection code is '1'.

To add a new record to the file, the user enters the full details at the bottom of the screen.

Command Function Keys Enabled

F3=Exit Program
F11=Delete selected record (Maintenance mode only)
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

MNTCTLRCD - Control Record Maintenance/Inquiry

Source members generated

*DSPF
*RPG

This program operates as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If no parameters are passed, or if the parameter is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

The main difference between this program and the MNTCTL shell is that it will access only one record from the file, Relative Record Number 1, as it assumes that this is the file control record.

In maintenance mode, if the record does not exist it will be created. If it does exist, it will be accessed for update.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR4500 Editing for both *ADD and *UPDATE

Screen D1 - Process

This screen shows the 'full' record detail for relative record number 1 in the file. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change field data. If enter is pressed and the data is valid the file is updated (Maintenance mode).

Command Function Keys Enabled

F3=Exit Program
HELP=Display help text

MNTDTAARA - Data Area Maintenance/Inquiry

Source members generated

*DSPF
*RPG

This program operates as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If no parameters are passed, or if the parameter is 'M', the program is called for maintenance and the file can be updated. If the parameter is 'I' the program is called for Inquiry only and the data area information is only displayed.

This program assumes that the data area already exists, and that a 'format file' has been created which defines the layout of the data area contents. When requesting the source code to be generated, the file name you specify is used as the format for the data area.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR4500 Editing for the *UPDATE function

Screen D1 - Select

The user will be shown the current contents of the data area.

If the program is in Maintenance mode the user may update the data area area contents. If any of the displayed fields are altered by the user the data is edit checked. If valid it is redisplayed to the user for confirmation and if the user presses enter again the data area is updated and the program ends.

Command Function Keys Enabled

F3=Exit Program
HELP=Display help text

MNTINQSEL - Record selection, allowing Maintenance/Inquiry

Source members generated

*DSPF
*RPG

This shell is available from QSHLSRC400 only.

**ENTRY parameter list*

Field	*Type*	*Len*	*Usage*	*Description*
@FUNC	*CHAR	1	*IN	Program Function - 'M' (Maintenance) or 'I' (Inquiry)
@INKC	*CHAR	1	*OUT	Exit function requested - '1' (yes) or '0' (no)
all key fields	*	*	*OUT	Details of the last record processed

This program operates as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If no parameters are passed, or if the parameter is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

The program functions in the same way as MNT_D1, but results of the program are returned to the calling program; firstly whether F3 (exit) was pressed and secondly the contents of the key fields of the last record processed by the user.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR3100 Editing specifically for *ADD (key field checks)
SR4100 Editing specifically for *UPDATE
SR4500 Editing for both *ADD and *UPDATE

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering the selection code beside one or more records shown the user can then display full detail for the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

The selection code to be used is dependent on the way the program was called. If it is a maintenance call, the selection code is '2'. If it is an inquiry call, the selection code is '1'.

By pressing F6 (available in Maintenance mode only) the 'Add New Record' function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record (Maintenance mode only)
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D2 - Display

This screen shows the ‘full’ record detail for every record selected by the user. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change non-key field data, or by pressing F11 delete the record.

If enter is pressed and the data is valid the file is updated (Maintenance mode). If multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record (Maintenance Update mode only)
F12=Return to Select screen
HELP=Display help text

MNTRRN - File Maintenance/Inquiry via Relative Record number

Source members generated

*DSPF
*RPG

This program operates as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If no parameters are passed, or if the parameter is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

The program functions in the same way as MNT_D1, except that processing is by Relative Record Number and not Key.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR3100 Editing specifically for *ADD (key field checks)
SR4100 Editing specifically for *UPDATE
SR4500 Editing for both *ADD and *UPDATE

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). The Relative Record Number of each record is displayed on the left hand side of the screen. Page down will display the next 10, Page up the previous 10.

Alternatively the user can enter a Relative Record Number to perform a SETLL (Set Lower Limits and Read) type of function.

By entering the selection code beside one or more records shown the user can then display full detail for the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

The selection code to be used is dependent on the way the program was called. If it is a maintenance call, the selection code is '2'. If it is an inquiry call, the selection code is '1'.

By pressing F6 (available in Maintenance mode only) the 'Add New Record' function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record (Maintenance mode only)
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D2 - Display

This screen shows the ‘full’ record detail for every record selected by the user. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change non-key field data, or by pressing F11 delete the record.

If enter is pressed and the data is valid the file is updated (Maintenance mode). If multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record (Maintenance Update mode only)
F12=Return to Select screen
HELP=Display help text

MNTSFL - File maintenance via subfile data entry

Source members generated

*DSPF
*RPG

The program allows the maintenance of fields from the subfile, rather than having to request each record to be displayed and then changed.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in the following subroutine:

SR4500 Editing for both *ADD and *UPDATE

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

Data is allowed to be corrected (if required) directly to the subfile. Each record changed in any way is edited.

All records changed must be valid before the file is updated. Record deletion is performed by typing a ‘4’ beside the appropriate line and pressing Enter. Record addition is performed by entering the new data on an ‘empty’ subfile line.

Command Function Keys Enabled

F3=Exit Program
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

PDWIN – Pop-up window display panel

This program displays a pop-up window containing data from a specific record in the specified file. This is made possible by the use of the DDS Window keywords provided in OS/400. Window display programs are ideally suited to providing the user access to non-essential information that may be occasionally required.

This code is not designed to run as a standalone program; the intention is for it to be (manually) merged in to a process that requires a pop-up window view function. The main display code is defined in subroutine SR9921, allowing easy definition of an F21=Display information window" function within an existing display program.

Special care is needed when defining the fields to be displayed in the window:

A maximum of 10 lines (one field per line) can be displayed for the selected record. The maximum length of each displayed line is 45 characters and you should enter this in the GENPGM prompt when generating the source.

You should keep the length of the field descriptive text as short as possible, to ensure as much data as possible can fit in the 45 character line length limitation. Once generated, you should manually alter the display file code for record format W1 and reduce the displayed descriptive text to the minimum readable length possible, giving more room for your record data.

Source members generated

QSHLSRC400	*QSHLSRCLE*
*RPG	RPGLE *MODULE
*DSPF	DSPF

*ENTRY parameter list

Field	*Type*	*Len*	*Usage*	*Description*
all key fields	*	*	*IN	Record key for the record selected

Screen W1 – Display selected record information

The window panel will pop up on the users display, showing the fields from the selected record. When the user presses enter, control will pass back to the caller.

PMT - Prompt all keys fields and validate

Source members generated

*DSPF
*RPG

This is a ‘pre-process’ type program. The user is prompted to enter the key fields in the file. The entry is validated and if acceptable a process program is called.

In all cases you will need to modify the generated source program to include the name of the ‘process’ program to call, as well as your own edit checking requirements (if any). These should be placed in one of the following subroutines:

SR4500 Editing
SR5000 Call the ‘process’ program

Screen D1 - Select

The user will be presented with a prompt screen in which an entry is made for each of the key fields in the file. The entry is validated (valid record exists with this key) and if acceptable a ‘process’ program is called.

Command Function Keys Enabled

F3=Exit Program
HELP=Display help text

PMT_DE - Restricted key prompt and Data Entry

Source members generated
*DSPF
*RPG

This program is designed as a data entry function. The initial prompt allows the user to enter a restricted key (for example, the document reference number), and then a subfile data entry screen allows the user to enter multiple lines for that reference.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR0600 Editing for the initial prompt screen
SR2110 Loads the subfile entries to the database
SR4500 Editing for the data entry subfile

Screen D1 - Select

The user will be presented with a prompt screen in which an entry is made for each of the restricted key fields in the file.

Command Function Keys Enabled

F3=Exit Program
HELP=Display help text

Screen D2 - Data Entry

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record) within the key restriction specified on the initial prompt screen. Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function. Where insufficient records exist in the file, the user will be provided with blanks lines to use for data entry.

Data is allowed to be corrected (if required) directly to the subfile. Each record changed in any way is edited.

All records changed must be valid before the file is updated. The actual file update is not performed until the *F10=Process* function key is used.

Command Function Keys Enabled

Enter=Edit the input, no update
F3=Exit Program
F10=Edit the input and, if no errors, update the file
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

PMTINQ - Prompt restricted keys and Inquiry

This program differs from the standard INQ shell in that the file key is restricted via an initial prompt. For each of the restricted keys the user is required to make an entry. The prompt subfile then only shows the records within that restriction.

Source members generated

*DSPF
*RPG

Screen D1 - Select

The user will be presented with a prompt screen in which an entry is made for each of the restricted keys requested.

The entry is validated (valid record exists with this partial key) and if acceptable the program continues to screen D2.

Command Function Keys Enabled

F3=Exit Program
HELP=Display help text

Screen D2 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file for the restricted key entered (one line per record). Page down will display the next 10, Page up the previous 10.

Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering a '1' beside one or more records shown the user can then display full detail for the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

Command Function Keys Enabled

F3=Exit Program
PAGEDOWN=Display next 10 records
PAGEUP=Display previous 10 records
HELP=Display help text

Screen D3 - Display

This screen shows the 'full' record detail for every record selected by the user.

If enter is pressed, and multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F12=Return to Select screen
HELP=Display help text

PMTMNT - Prompt restricted keys and maintenance/Inquiry

This program differs from the standard MNT_D1 shell in that the file key is restricted via an initial prompt. For each of the restricted keys the user is required to make an entry. The prompt subfile then only shows the records within that restriction.

Source members generated

*DSPF
*RPG

The program functions as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If no parameters are passed, or if the parameter is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR3100 Editing specifically for *ADD (key field checks)
SR4100 Editing specifically for *UPDATE
SR4500 Editing for both *ADD and *UPDATE
SR5100 Editing for the initial prompt

Screen D1 - Select

The user will be presented with a prompt screen in which an entry is made for each of the restricted keys requested. The entry is validated (valid record exists with this partial key) and if acceptable the program continues to screen D2.

Command Function Keys Enabled

F3=Exit Program
HELP=Display help text

Screen D2 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file within the key restriction (one line per record). Page down will display the next 10, Page up the previous 10.

Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering the selection code beside one or more records shown the user can then display full detail for the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

The selection code to be used is dependent on the way the program was called. If it is a maintenance call, the selection code is '2'. If it is an inquiry call, the selection code is '1'.

By pressing F6 (available in Maintenance mode only) the 'Add New Record' function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record (Maintenance mode only)
PAGEDOWN=Display next 10 records
PAGEUP=Display previous 10 records
HELP=Display help text

Screen D2 - Display

This screen shows the ‘full’ record detail for every record selected by the user. If the program is in Inquiry mode the user may only display the record. If the program is in Maintenance mode the user may change non-key field data, or by pressing F11 delete the record.

If enter is pressed and the data is valid the file is updated (Maintenance mode). If multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record (Maintenance Update mode only)
F12=Return to Select screen
HELP=Display help text

PMT_RST - Prompt restricted key fields and validate

Source members generated

*DSPF
*RPG

This is a ‘pre-process’ type program. The user is prompted to enter the key fields in the file. The entry is validated and if acceptable a process program is called. It functions in exactly the same way as PMT, but only for a requested number of key fields (rather than all keys).

In all cases you will need to modify the generated source program to include the name of the ‘process’ program to call, as well as your own edit checking requirements (if any). These should be placed in one of the following subroutines:

SR4500 Editing
SR5000 Call the ‘process’ program

Screen D1 - Select

The user will be presented with a prompt screen in which an entry is made for each of restricted keys requested. The entry is validated (valid record exists with this partial key) and if acceptable a ‘process’ program is called.

Command Function Keys Enabled

F3=Exit Program
HELP=Display help text

PRC - Process records

Source members generated
*RPG

The program will process all records in the specified file. No actual processing is done - the code that is generated will just move the contents of the fields into themselves and update the file. For this reason the source will always have to be modified.

The program generated from this shell is very handy for situations where you wish to process some fields in a file.

PRC_R - Process records, with control totals report

Source members generated

*RPG
*PRTF

The program will process all records in the specified file. A control total is printed on an audit report, showing how many records were actually processed.

No actual processing is done - the code that is generated will just move the contents of the fields into themselves and update the file. For this reason the source will always have to be modified.

The program generated from this shell is very handy for situations where you wish to process some fields in a file.

PRT_ALL - Print records

Source members generated
*RPG
*PRTF

The program will process all records in the specified file and for each record found will print a line of detail on the resulting report.

At the end of the report a control total is printed, showing how many records were actually processed.

The printer file is defined with the following attributes:

PAGESIZE(66 132)	66 lines per page, 132 characters per line
CPI(12)	12 characters per inch

The program generated from this shell is very handy for situations where you wish to create a program to process all records in a file and print the results. The generated source code can be modified to suit your specific needs. For example, where you wish to change the contents of some fields in all records.

Source Generation notes
On the GENPGM prompt you must set the 'maximum width to generate' for the *PRTF entry to 132 for the source generation to work correctly.

PRT_198 - Print records (compressed print)

Source members generated
*RPG
*PRTF

The program will process all records in the specified file and for each record found will print a line of detail on the resulting report.

At the end of the report a control total is printed, showing how many records were actually processed.

The program is the same as the PRT_ALL shell, except that the printer file prints to a maximum line length of 198. The printer file is defined with the following attributes:

PAGESIZE(66 198)	66 lines per page, 198 characters per line
CPI(15)	15 characters per inch

The program generated from this shell is very handy for situations where you wish to create a program to process all records in a file and print the results. The generated source code can be modified to suit your specific needs. For example, where you wish to change the contents of some fields in all records.

Source Generation notes
On the GENPGM prompt you must set the 'maximum width to generate' for the *PRTF entry to 198 for the source generation to work correctly.

PRTDTAARA - Print Data Area contents

Source members generated

*PRTF
*RPG

The program will read the contents of a specified data area print a line of detail on the resulting report.

The printer file is defined with the following attributes:

PAGESIZE(66 132)	66 lines per page, 132 characters per line
CPI(12)	12 characters per inch

Source Generation notes

On the GENPGM prompt you must set the ‘maximum width to generate’ for the *PRTF entry to 132 for the source generation to work correctly.

This program assumes that the data area already exists, and that a ‘format file’ has been created which defines the layout of the data area contents. When requesting the source code to be generated, the file name you specify is used as the format for the data area.

PRT_L0 - Print records using a restricted key

Source members generated
*RPG
*PRTF

The program will process all records in the specified file for the restricted key defined and for each record found will print a line of detail on the resulting report.

At the end of the report a control total is printed, showing how many records were actually processed.

The printer file is defined with the following attributes:

PAGESIZE(66 132)	66 lines per page, 132 characters per line
CPI(12)	12 characters per inch

PRT_L1 - Print records with control level break

Source members generated
*RPG
*PRTF

The program will process all records in the specified file and for each record found will print a line of detail on the resulting report.

'Level break' processing is performed on the first key field of the file - The key is printed at the top of the page and on each change of key a new page is started.

At the end of the report a control total is printed, showing how many records were actually processed.

The printer file is defined with the following attributes:

PAGESIZE(66 132)	66 lines per page, 132 characters per line
CPI(12)	12 characters per inch

PRT_QRY - Print records via OPNQRYF selection

Source members generated

QSHLSRC400	*QSHLSRCLE*	
*CLP	CLLE	*MODULE
*RPG	RPGLE	*MODULE
*PRTF	PRTF	

The program will process records in the specified file selected via OPNQRYF and for each record found will print a line of detail on the resulting report.

At the end of the report a control total is printed, showing how many records were actually processed.
The CLP program generated will contain a 'shell' OPNQRYF statement (QRYSLT *ALL) that you must tailor to your own requirements.

The printer file is defined with the following attributes:

PAGESIZE(66 132)	66 lines per page, 132 characters per line
CPI(12)	12 characters per inch

The program generated from this shell is very handy for situations where you wish to create a program to process selected records in a file and print the results.

The generated source code can be modified to suit your specific needs. For example, where you wish to change the contents of some fields in some records.

In most cases you will need to modify the generated source program to include your own processing requirements. These should be placed in one of the following subroutines:

SR1000	Process file data

QSHLSRCLE Compilation Notes
The RPGLE and CLLE source will generate modules and then a binding step in the CLLE source member will create the final program.

PRT_SQL - Print records with SQL select/retrieve/order

Source members generated

*SQLRPGLE
*PRTF

The program will process records in the specified file and for each record selected will print a line of detail on the resulting report.

At the end of the report a control total is printed, showing how many records were actually processed.

The program is similar to the PRT_ALL shell, except that it includes SQL statements that can be customised to perform select/retrieve/order operations.

The printer file is defined with the following attributes:

PAGESIZE(66 132)	66 lines per page, 132 characters per line
CPI(12)	12 characters per inch

The generated source code can be modified to suit your specific needs. For example, where you wish to change the contents of some fields in some records.

Source Generation notes

On the GENPGM prompt you must set the ‘maximum width to generate’ for the *PRTF entry to 132 for the source generation to work correctly.

SEL - Selection program

This selection program is ideally used when, for example, a user has to select from a choice of selections; in your main program you specify that if the user enters a '?' in a field the program calls this program to display a list to choose from.

Source members generated

QSHLSRC400	*QSHLSRCLE*
*RPG	RPGLE *MODULE
*DSPF	DSPF

**ENTRY parameter list*

Field	*Type*	*Len*	*Usage*	*Description*
@INKC	*CHAR	1	*OUT	Exit function requested - '1' (yes) or '0' (no)
@INKL	*CHAR	1	*OUT	Return function requested - '1' (yes) or '0' (no)
all key fields	*	*	*OUT	Details of the record selected

The program functions as a record selection program where the results of the program are returned to the calling program; firstly whether F3 (exit) or F12 (return) was pressed and secondly the contents of the key fields of the record selected by the user.

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering a '1' beside one of the records shown the user 'selects' that record and the details of the file key are returned to the calling program.

The results of the function keys 3 and 12 are also passed back - this allows the calling program to perform a 'full' exit if F3 was pressed.

Command Function Keys Enabled

F3=Exit Program
F12=Return
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

SELW - Selection program using a 'popup' selection panel

This selection program is similar to the SEL program, but the display appears to the user in a 'pop-up' window. This is made possible by the use of the DDS Window keywords provided in OS/400. Window selection programs are ideally suited to parameter code selection functions.

Special care is needed when defining the fields to be displayed in the window:

The maximum length of the subfile line is 34 characters. You should enter this in the GENPGM prompt when generating the source.

You **must** specify on the GENPGM prompt that you wish to alter field positioning and text. You must change the definition of all key fields: All key fields **must** have a field attribute of either

'C' (key control, displayed in subfile), or
'H' (key hidden, not displayed in subfile).

Failure to do this will mean the source code generated will not work as intended.

Only column heading 1 is used for the subfile 'header' line. You should keep the length of the text as short as possible, to ensure as much data as possible can fit in the 34 character limitation.

Source members generated

QSHLSRC400	*QSHLSRCLE*
*RPG	RPGLE *MODULE
*DSPF	DSPF

*ENTRY parameter list

Field	*Type*	*Len*	*Usage*	*Description*
@INKC	*CHAR	1	*OUT	Exit function requested - '1' (yes) or '0' (no)
@INKL	*CHAR	1	*OUT	Return function requested - '1' (yes) or '0' (no)
all key fields	*	*	*OUT	Details of the record selected

The program functions as a record selection program where the results of the program are returned to the calling program; firstly whether F3 (exit) or F12 (return) was pressed and secondly the contents of the key fields of the record selected by the user.

Screen D1 - Select

The user will first be presented with a 'popup' window containing a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. There is no 'position to' function within a window.

By entering a '1' beside one of the records shown the user 'selects' that record and the details of the file key are returned to the calling program.

The results of the function keys 3 and 12 are also passed back - this allows the calling program to perform a 'full' exit if F3 was pressed.

Command Function Keys Enabled

F3=Exit Program
F12=Return
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

SELWIN - Selection program using a 'popup' selection panel

This selection program is similar to the SELW program, but for two differences:

- The window is larger
- The user can use the familiar "position to" key positioning logic for the subfile.

Special care is needed when defining the fields to be displayed in the window; the maximum length of the subfile line is 60 characters. You should enter this in the GENPGM prompt when generating the source.

Only column heading 1 is used for the subfile 'header' line. You should keep the length of the text as short as possible, to ensure as much data as possible can fit in the 60 character limitation.

Source members generated

QSHLSRC400	*QSHLSRCLE*
*RPG	RPGLE *MODULE
*DSPF	DSPF

*ENTRY parameter list

Field	*Type*	*Len*	*Usage*	*Description*
@INKC	*CHAR	1	*OUT	Exit function requested - '1' (yes) or '0' (no)
@INKL	*CHAR	1	*OUT	Return function requested - '1' (yes) or '0' (no)
all key fields	*	*	*OUT	Details of the record selected

The program functions as a record selection program where the results of the program are returned to the calling program; firstly whether F3 (exit) or F12 (return) was pressed and secondly the contents of the key fields of the record selected by the user.

Screen D1 - Select

The user will first be presented with a 'popup' window containing a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10.

By entering a '1' beside one of the records shown the user 'selects' that record and the details of the file key are returned to the calling program.

The results of the function keys 3 and 12 are also passed back - this allows the calling program to perform a 'full' exit if F3 was pressed.

Command Function Keys Enabled

F3=Exit Program
F12=Return
PAGEDOWN=Display next 19 records in file
PAGEUP=Display previous 19 records in file
HELP=Display help text

SEL_CPF - Selection program using CPF pagekey control

This selection program is ideally used when, for example, a user has to select from a choice of selections; in your main program you specify that if the user enters a '?' in a field the program calls this program to display a list to choose from.

It functions in the same way as the SEL shell except that CPF controls subfile paging, so if used it should only be against small files.

Source members generated

QSHLSRC400	*QSHLSRCLE*
*RPG	RPGLE *MODULE
*DSPF	DSPF

**ENTRY parameter list*

Field	*Type*	*Len*	*Usage*	*Description*
all key fields	*	*	*OUT	Details of the record selected

The program functions as a record selection program where the results of the program (the file key of the record selected) are returned to the calling program.

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering a '1' beside one of the records shown the user 'selects' that record and the details of the file key are returned to the calling program.

Command Function Keys Enabled

F12=Return
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

WRKINQ - File inquiry using a 'Work With' panel

Source members generated

*DSPF
*RPG

This program operates as an Inquiry function. It is similar in concept to the WRKPNL maintenance shell, providing the same style user interface.

In some cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR5100 Editing specifically for *DISPLAY
SR9100 Processing for 'Locked record' handling

**ENTRY parameter list*

Field	*Type*	*Len*	*Usage*	*Description*
@FUNC	*CHAR	1	*IN	Program function - 'I' (Inquiry)

The parameter to be passed is an optional parameter - it can be ignored. The purpose is to provide a similar call interface as used in the WRKPNL shell.

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering the selection code "5=Display" beside one or more records shown the user can then perform the requested function on the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

Command Function Keys Enabled

F3=Exit Program
F17=Position subfile to beginning-of-file
F18=Position subfile to end-of-file
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D5 - Display

This screen shows the 'full' record detail for every record selected by the user.

If enter is pressed and if multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F12=Return to Select screen
HELP=Display help text

WRKPNL - File maintenance using a 'Work With' panel

Source members generated

*DSPF
*RPG

This program operates as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If the 'program function' is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

The program is similar in concept to the MNT_D1 shell, but more powerful; it is based on the methodology of the various 'work with' panels that exist on the iSeries.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR2100 Editing specifically for *UPDATE and *COPY
SR3100 Editing specifically for *ADD
SR4100 Editing specifically for *DELETE
SR5100 Editing specifically for *DISPLAY
SR8000 Common Editing for *ADD, *UPDATE and *COPY
SR9100 Processing for 'Locked record' handling

**ENTRY parameter list*

Field	*Type*	*Len*	*Usage*	*Description*
@FUNC	*CHAR	1	*IN	Program function - 'M' (Maintenance) or 'I' (Inquiry)

It is important to note that in this shell the 'program function' parameter is a required parameter and must be passed.

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering a selection code beside one or more records shown the user can then perform the requested function on the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

The selection code to be used is dependent on the way the program was called.

If it is an inquiry call, the selection code is '5'.
If it is a maintenance call, the selection codes available are 2=Change, 3=Copy, 4=Delete and 5=Display.

By pressing F6 (available in Maintenance mode only) the ‘Add New Record’ function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record (Maintenance mode only)
F17=Position subfile to beginning-of-file
F18=Position subfile to end-of-file
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D2 - Change

This screen shows the ‘full’ record detail for every record selected by the user. The user may change non-key field data.

If enter is pressed the screen data is validated (no update occurs). If F10 is pressed the data is validated and, if acceptable, the file is updated.

If multiple selections were made, the next record will then be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F10=Accept (change record)
F12=Return to Select screen
HELP=Display help text

Screen D3 - Copy

This screen shows the ‘full’ record detail for each ‘copy from’ record selected by the user. The user may change all field data; at least one of the key fields must be changed so that the newly copied record will have it’s own unique key.

If enter is pressed the screen data is validated (no update occurs). If F10 is pressed the data is validated and, if acceptable, the file is updated.

If multiple selections were made, the next record will then be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F10=Accept (copy record)
F12=Return to Select screen
HELP=Display help text

Screen D4 - Delete

This screen shows the ‘full’ record detail for each record selected by the user. The user then has the option to press F11 to delete the record shown or F12 to ignore the delete.

If multiple selections were made, the next record will then be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F11=Accept (delete record)
F12=Return to Select screen
HELP=Display help text

Screen D5 - Display

This screen shows the ‘full’ record detail for every record selected by the user.

If enter is pressed and if multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F12=Return to Select screen
HELP=Display help text

WRKPNLRRN – RRN File maintenance using a 'Work With' panel

Source members generated

*DSPF
*RPG

This program operates as both a Maintenance and an Inquiry function, depending on the parameter passed to the program when it is called. If the 'program function' is 'M', the program is called for maintenance and the file is opened for Update. If the parameter is 'I' the program is called for Inquiry only and the file is opened for Read only.

The program is similar in concept to the WRKPNL shell, but is intended for files that have no key definition – access is via Relative Record Number.

In most cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR2100 Editing specifically for *UPDATE and *COPY
SR3100 Editing specifically for *ADD
SR4100 Editing specifically for *DELETE
SR5100 Editing specifically for *DISPLAY
SR8000 Common Editing for *ADD, *UPDATE and *COPY
SR9100 Processing for 'Locked record' handling

**ENTRY parameter list*

Field	*Type*	*Len*	*Usage*	*Description*
@FUNC	*CHAR	1	*IN	Program function - 'M' (Maintenance) or 'I' (Inquiry)

It is important to note that in this shell the 'program function' parameter is a required parameter and must be passed.

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter a record number to perform a SETLL (Set Lower Limits and Read) type of function.

By entering a selection code beside one or more records shown the user can then perform the requested function on the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

The selection code to be used is dependent on the way the program was called.

If it is an inquiry call, the selection code is '5'.
If it is a maintenance call, the selection codes available are 2=Change, 3=Copy, 4=Delete and 5=Display.

By pressing F6 (available in Maintenance mode only) the ‘Add New Record’ function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record (Maintenance mode only)
F17=Position subfile to beginning-of-file
F18=Position subfile to end-of-file
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D2 - Change

This screen shows the ‘full’ record detail for every record selected by the user. The user may change non-key field data.

If enter is pressed the screen data is validated (no update occurs). If F10 is pressed the data is validated and, if acceptable, the file is updated.

If multiple selections were made, the next record will then be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F10=Accept (change record)
F12=Return to Select screen
HELP=Display help text

Screen D3 - Copy

This screen shows the ‘full’ record detail for each ‘copy from’ record selected by the user. The user may change all field data; at least one of the key fields must be changed so that the newly copied record will have it’s own unique key.

If enter is pressed the screen data is validated (no update occurs). If F10 is pressed the data is validated and, if acceptable, the file is updated.

If multiple selections were made, the next record will then be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F10=Accept (copy record)
F12=Return to Select screen
HELP=Display help text

Screen D4 - Delete

This screen shows the ‘full’ record detail for each record selected by the user. The user then has the option to press F11 to delete the record shown or F12 to ignore the delete.

If multiple selections were made, the next record will then be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F11=Accept (delete record)
F12=Return to Select screen
HELP=Display help text

Screen D5 - Display

This screen shows the ‘full’ record detail for every record selected by the user.

If enter is pressed and if multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F3=Exit Program
F12=Return to Select screen
HELP=Display help text

QEZUSRCLNP - iSeries system cleanup program

Source members generated
*CLP

This shell generates a basic iSeries user defined cleanup program. In all cases you will need to modify the generated code to insert your user requirements.

When requesting the source code generation for this program, enter file name *NONE.

User cleanup is explained in the IBM Operational Assistant information in the IBM iSeries System Operations Guide. After performing it's normal system cleanup functions (such as job logs and output queues) OA will attempt to run the first program in the library list called QEZUSRCLNP.

Requirements for creating this program are therefore:

- Operational Assistant Cleanup must be enabled, via the CHGCLNUP command
- The compiled program MUST be called QEZUSRCLNP
- The compiled program must be placed in a library which is *higher* in the system library list than QSYS.

QEZPWROFFP - iSeries system power-off program

Source members generated
*CLP

This shell generates a basic iSeries user defined system power off program. In some cases you will need to modify the generated code to insert your user requirements.

When requesting the source code generation for this program, enter file name *NONE.

If power scheduling has been enabled on the iSeries, the system will attempt to power down the iSeries at the specified time, by running the first program in the library list called QEZPWROFFP.

Requirements for creating this program are therefore:

- Power scheduling must be enabled, via the CHGPWRSCD command
- The compiled program MUST be called QEZPWROFFP
- The compiled program must be placed in a library which is *higher* in the system library list than QSYS.

QSTRUP - iSeries Start-up program

Source members generated
*CLP

This shell generates a basic iSeries start-up program. In all cases you will need to modify the generated code to insert your user requirements.

When requesting the source code generation for this program, enter file name *NONE.

The iSeries system start-up program is enabled by altering the system value QSTRUPPGM, to point to the program object compiled from this generated source. The system value is altered by using the command

```
WRKSYSVAL QSTRUPPGM
```

'Order Entry' style shells

Most of the shells supplied are 'general purpose' in design, where they act on one file only. In the case of an 'order entry' type function, it is usually necessary to process two files; the order header (one record per order) and the order detail (multiple records per order).

To assist you in generating these types of functions, there are four shells supplied:

OE1_PMT	Order Entry, Header details via prompt
OE1_SFL	Order Entry, Header details via subfile select
OE2	Order Entry, Line details
OE2SFL	Order Entry, Line details (Subfile data entry)

For one order entry process you will need to generate

one of the *OE1* style programs (to process the order header) and
one of the *OE2* style programs (to process the order details).

Within the generated source, you will find comments relating to the merging of the two display files generated. This is a suggested method of development; where the two programs will share the same display file.

The files are assumed to be based on the following key logic:

File	Key
Order Headers	Order number
Order Details	Order number Line number

The *OE1* program would process based on the primary key information (order number) and pass this to the *OE2* program, which would process all records within this key restriction.

In the case of the ILE shells (QSHLSRCLE) it is assumed that the OE1 and OE2 functions are modules that will be combined with a CLLE module to become a complete order entry process within the one program object.

OE1_PMT - Order Header details (via prompt)

This program allows the user to add a new order header, or maintain an existing one based on the key field entries made via an initial prompt. For each of the restricted keys the user is required to make an entry. The maintenance display then shows the specific record retrieved for the key entered. After maintenance of the header details is complete, a call is made to the *OE2* details program.

Source members generated

QSHLSRC400	QSHLSRCLE
*CLP	CLLE *MODULE
*RPG	RPGLE *MODULE
*DSPF	DSPF

In all cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR3100 Editing specifically for *ADD (key field checks)
SR4100 Editing specifically for *UPDATE
SR4300 Order deletion processing
SR4500 Editing for both *ADD and *UPDATE
SR5000 Call to Order detail processing program
*INZSR Program initialization

Required changes are defined in the source members by a '---->' in columns 1 to 5. These lines are generic and MUST be modified before an attempt is made to compile the program.

Screen D1 - Select

The user will be presented with a prompt screen in which an entry is made for each of the restricted keys requested.

The entry is validated (valid record exists with this partial key) and if acceptable the program continues to screen D2.

By pressing F6 the 'Add New Record' function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record
HELP=Display help text

Screen D2 - Display

This screen shows the 'full' record detail for the record selected by the user. The user may change non-key field data, or by pressing F11 delete the record. If F11 is taken, all order detail lines are deleted from the detail file, as well as the header being removed from the header file.

If enter is pressed and the data is valid the file is updated and the order detail processing program is called, passing the restricted file key (order number).

Upon return from the Order detail processing program, the user is returned to the Select screen.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record
F12=Return to Select screen
HELP=Display help text

QSHLSRCLE Program Modification Notes

You will need to modify the CRTPGM post-compile directives in the CLLE source. The MODULE parameter must contain the names CLLE module, the RPGLE module for the *OE1* program and the RPGLE module for the *OE2* program.

QSHLSRCLE Compilation Notes

The RPGLE and CLLE source will generate modules and then a binding step in the CLLE source member will create the final program.

Because this final binding step will bind all modules in the Order Entry function (including the *OE2 Order_Details* module) you should create the CLLE module last.

OE1_SFL - Order Entry header details (via subfile select)

This program is similar to OE1_PMT, in that it allows the user to add a new order header, or maintain an existing one. The difference between the two programs is the initial select prompt - in this program the user is provided with a subfile list of existing order headers to select from. The maintenance display then shows the specific record retrieved for the key entered.

After maintenance of the header details is complete, a call is made to the *OE2* details program.

Source members generated

QSHLSRC400	QSHLSRCLE
*CLP	CLLE *MODULE
*RPG	RPGLE *MODULE
*DSPF	DSPF

In all cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR3100 Editing specifically for *ADD (key field checks)
SR4100 Editing specifically for *UPDATE
SR4300 Order deletion processing
SR4500 Editing for both *ADD and *UPDATE
SR5000 Call to Order detail processing program
*INZSR Program initialization

Required changes are defined in the source members by a `'---->'` in columns 1 to 5. These lines are generic and **must** be modified before an attempt is made to compile the program.

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record). Page down will display the next 10, Page up the previous 10. Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering a '1' beside one or more records shown the user can then display full detail for the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

By pressing F6 the 'Add New Record' function will allow the user to enter new records into the file.

Command Function Keys Enabled

F3=Exit Program
F6=Add new record
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D2 - Display

This screen shows the 'full' record detail for the record selected by the user. The user may change non-key field data, or by pressing F11 delete the record. If F11 is taken, all order detail lines are deleted from the detail file, as well as the header being removed from the header file.

If enter is pressed and the data is valid the file is updated and the order detail processing program is called, passing the restricted file key (order number).

Upon return from the Order detail processing program, the user is returned to the Select screen.

Command Function Keys Enabled

F3=Exit Program
F11=Delete Record
F12=Return to Select screen
HELP=Display help text

QSHLSRCLE Program Modification Notes

You will need to modify the CRTPGM post-compile directives in the CLLE source. The MODULE parameter must contain the names CLLE module, the RPGLE module for the *OE1* program and the RPGLE module for the *OE2* program.

QSHLSRCLE Compilation Notes

The RPGLE and CLLE source will generate modules and then a binding step in the CLLE source member will create the final program.

Because this final binding step will bind all modules in the Order Entry function (including *the OE2 Order_Details* module) you should create the CLLE module last.

OE2 - Order Entry line details (via subfile select)

Source members generated

QSHLSRC400	QSHLSRCLE
*RPG	RPGLE *MODULE
*DSPF	DSPF

This program will provide a function similar to the MNT_D1 shell, for the Order Lines associated with an Order header processed by one of the *OE1* programs. Based on the restricted key (Order number) passed from the *OE1* program, this program will allow the user to maintain the line details for the order.

In all cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in one of the following subroutines:

SR3100 Editing specifically for *ADD (key field checks)
SR4100 Editing specifically for *UPDATE
SR4500 Editing for both *ADD and *UPDATE
*INZSR Program initialization

Required changes are defined in the source members by a `'---->'` in columns 1 to 5. These lines are generic and MUST be modified before an attempt is made to compile the program.

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record), for the specified order number. Page down will display the next 10, Page up the previous 10.

Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

By entering a '1' beside one or more records shown the user can then display full detail for the specific record. Where more than one record was selected the user will be presented with each selection (by pressing the enter key).

By pressing F6 the 'Add New Record' function will allow the user to enter new records into the file.

Command Function Keys Enabled

F6=Add new record
F10=Order complete
F12=Return
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

Screen D2 - Display

This screen shows the ‘full’ record detail for every record selected by the user, who may change non-key field data, or by pressing F11 delete the record.

If enter is pressed and the data is valid the file is updated. If multiple selections were made, the next record will be displayed. If only one selection was made the user will be returned to the Select screen. To bypass multiple selections the user can press F12 to return without seeing any more of the selections made.

Command Function Keys Enabled

F11=Delete Record
F12=Return to order line Select screen
HELP=Display help text

QSHLSRCLE Compilation Notes

The RPGLE source will generate the program module for the *Order_Details* function. The final binding step in the OE1 CLLE module will bind this *OE2* module into the final *Order_Entry_Function* program object.

OE2SFL - Order Entry line details (via subfile data entry)

Source members generated

QSHLSRC400	QSHLSRCLE
*RPG	RPGLE *MODULE
*DSPF	DSPF

This program will provide a function similar to the MNTSFL shell, for the Order Lines associated with an Order header processed by one of the *OE1* programs. Based on the restricted key (Order number) passed from the *OE1* program, this program will allow the user to maintain the line details for the order.

In all cases you will need to modify the generated source program to include your own edit checking requirements. These should be placed in the following subroutine:

SR4500 Editing for both *ADD and *UPDATE

Required changes are defined in the source members by a `'---->'` in columns 1 to 5. These lines are generic and MUST be modified before an attempt is made to compile the program.

Screen D1 - Select

The user will first be presented with a list screen, showing details of the first 10 records in the file (one line per record), for the specified order number. Page down will display the next 10, Page up the previous 10.

Alternatively the user can enter key fields to perform a SETLL (Set Lower Limits and Read) type of function.

Data is allowed to be corrected (if required) directly to the subfile. Each record changed in any way is edited.

All records changed must be valid before the file is updated.

Record deletion is performed by typing a ‘4’ beside the appropriate line and pressing Enter. Record addition is performed by entering the new data on an ‘empty’ subfile line.

Command Function Keys Enabled

F10=Order complete
F12=Return
PAGEDOWN=Display next 10 records in file
PAGEUP=Display previous 10 records in file
HELP=Display help text

QSHLSRCLE Compilation Notes

The RPGLE source will generate the program module for the *Order_Details* function. The final binding step in the *OE1* CLLE module will bind this *OE2* module into the final *Order_Entry_Function* program object.

Cross Reference Utility

An Overview

XREF (NUTIL Cross Reference Utility) is a programmer productivity tool providing immediate access to documentation and cross reference information. Information provided allows you to determine where (and how) a file is used, where a field is used, program summary information listing all objects accessed, and so on.

Information is presented via the screen with the option to produce a print of the displayed data.

Changes to the ”LIVE” system can be reflected immediately in the cross reference inquiries, using the tools provided in NUTIL.

XREF can be split into three broad functions :

1. Cross Reference Generation
2. Cross Reference Inquiry
3. Programmer Development Facility

The Cross Reference Documentation facility

This function has been developed to assist the data processing department keep all systems documentation accurate and up to date. It achieves this by storing all relevant information in its own Data Base, that is created once only for a new system, and kept up to date each time an object is modified or a new object added to the production library.

> Therefore it is important that any object creation is performed using the NUTIL programmer exit program NCRTOBJECT, the generic create utility NCRTOBJ, or via the NUTIL PDM interface NPDMCRTOBJ. Alternatively you must ensure that the GENOBJXRF command is processed every time you create an object by a means other than using the NUTIL programmer exit program NCRTOBJECT, the generic create utility NCRTOBJ, or via the NUTIL PDM interface NPDMCRTOBJ.

Once the Cross Reference Data Base is created for an application, all objects within the application will exist in the Data Base and are available for inquiry.

The Database created in XREF is stored by an ‘application’ (or system) name that is specified during the generation of each application. The database will be generated for all objects that exist in the specified Object Library, and thereafter will be stored by the Object Library name for future access.

If more than one application exists in the cross reference database, the inquiry programs will expect the user to enter the application for the relevant inquiry. If no application name is entered, the screen will display a ‘window’ list of applications defined and expect the user to select one of the applications displayed.

There are three major limitations within the XREF function:

- Internally described RPG programs and all HLL program types other than RPG may be run through XREF but will not be included in the ”Field in which Program Inquiry”.
- The ”Field in which Program Inquiry” is only valid for files that have been defined by field in DDS specifications. Files that have only been defined as one block of data will appear as such on this inquiry.
- The ”Field in which Program Inquiry” will not show fields used withinin ILE free-form RPGLE code.

Using Cross Reference

The main XREF menu can be accessed in one of three ways:

1 from option 5 of the NUTIL main menu,
2 by a ‘GO UTLXRF’ command, or
3 by the XREF command.

By selecting option 1 on the UTLXRF main menu the Cross Reference Generation menu UTLXR2 will be displayed.

All options, other than option 3, should only be selected by a person with sufficient user authority to access all objects in the libraries to be processed. This is because the system needs authority to all the objects to be included in the cross reference data base.

Options 1, 2 and 6 cause a job to be submitted. The default job description XREF is used for this purpose. The purpose of a default job description is consistency - you can be sure where the jobs are and how they are processing.

Option 3 causes the Cross Reference Inquiry menu UTLXR3 to be displayed. From here you can perform all inquiries into the previously generated Cross Reference Database.

Generate X-REF Data Base for an Entire Application

Option 1 on the Generation Menu performs the GENAPPXRF (Generate Application Cross Reference) command. This option has two functions:

1. To define the application to XREF
2. To initially load the Cross Reference information for an application.

The GENAPPXRF command requires you to the following parameters:

OBJECT LIBRARY:	Name of the library containing the objects you wish to place in the XREF data base.
RPG SOURCE:	Name of the library/file containing RPG source relating to the object library (if any).
DDS SOURCE:	Name of the library/file containing DDS source relating to the object library (if any).
CLP SOURCE:	Name of the library/file containing CLP source relating to the object library (if any).

You should be aware that the submitted job is a long running process, the actual time it takes being purely dependent on the number of objects in the specified library (and the size of their related source members).

Generate X-REF Data Base for an Individual Object

Option 2 on the Generation Menu performs the GENOBJXRF (Generate Object Cross Reference) command. This command is similar to the GENAPPXRF command, but performs the generation for a single object, rather than all objects in the specified library.

The GENOBJXRF command will only generate cross reference information for the object if the relevant application cross reference has previously been generated via option 1 of the generation menu.

The GENOBJXRF command requires you to enter the following parameters:

OBJECT:	Name of the object to be added to the cross reference
OBJTYP:	The type of the object to be added to the cross reference data base.
SOURCE:	The name of the library/file containing any source member relating to the object. The source member name will be assumed to be the same as the object name.

Inquiries Menu

The inquiry menu is option 3 from the UTLXR2 menu (which can also be accessed by a ‘GO UTLXR3’ or via the ‘XREF’ command). No options on this menu should be selected until some cross reference data has been generated (i.e. options 1 on menu UTLXR2 have been run).

Options on this menu should not be selected whilst any XREF data base generation is in progress.

Information retrieved via the options on this menu will only be accurate if all changes to the system have been added to the database. Therefore it is important that any object creation is performed using the NUTIL programmer exit program NCRTOBJECT, the generic create utility NCRTOBJ, or via the NUTIL PDM interface NPDMCRTOBJ.

Alternatively you must ensure that the GENOBJXRF command is processed every time you create an object by a means other than using the NUTIL programmer exit program NCRTOBJECT, the generic create utility NCRTOBJ, or via the NUTIL PDM interface NPDMCRTOBJ.

Some examples of using the inquiries menu:

Example 1. Determine the amount of development effort required to change the size of a field

Use option 5 (field in which file inquiry) to determine which files will have to be recreated.

Use option 4 (file cross reference inquiry) for each file displayed by option 5 to determine which logical files and programs will have to be recreated.

Use option 6 (field in which program inquiry) to determine which programs may need modification.

If your files are defined by field but internally defined within some programs use only options 5 and 4. If the files are defined by size only (i.e. a block of data) use only option 4.

Example 2. Determine the effects of removing an obsolete file from the system

Use option 4 (file cross reference inquiry) to determine if there are any logical files incorporating the physical file to be removed. It will also list any programs which still reference the file.

Example 3. Determine what action is required when a program terminates abnormally

Use option 7 (program flow inquiry) to determine which programs were not executed and what files have been or should have been affected.

Program List Inquiry

You will be prompted for System Reference, which is the name (object library) that you entered when the cross reference was generated. If the system reference that you entered is not found in the cross reference data base an error message will be displayed.

Having entered a valid system reference, the program list display will appear. This will show all programs which exist for the system reference selected, together with the type (e.g. RPG) and the associated description.

If you press F6 the program list report will be printed, showing the same information as was displayed.

Only those programs which have been created will show even though there may be source code in the specified source file.

Program Cross Reference Inquiry

You will be prompted for Program Name. If this is not found in the cross reference data base an error message will be displayed.

Having entered a valid program name, the program cross reference display will be shown. This display may be composed of up to four types of list :

CALLED BY: If the program specified is called by any other programs, these will be shown together with their type, description and the statement number where the call occurs.

FILES USED: If the program accesses any files, these will be shown together with each member of the file, the member description and the member usage.

*ENTRY PLIST: If the program specified expects to be passed any parameters when it is called these will be listed together with their type, size and the statement number where they are specified.

CALLING: If the specified program calls any other programs these will be listed together with the statement number where the call is issued, the called program type and description.

If you press F6 the program cross reference report will be printed showing the same information as was displayed. The report is formatted to fit on A4 stationery so that it can be used as formal documentation. The page heading (which has a default of ”Navan Utilities”) may be changed by keying the following command:

```
CHGDTAARA NUTIL/HED954 VALUE('AAAAA')
```

Where AAAAA is the required heading of up to 50 characters.

Programs which have source but no corresponding object cannot be inquired upon, however they may be shown in the CALLING list. Created programs with no source will only show the FILES USED list. Files with no source code will appear on the FILES USED list.

File Field Inquiry

You will be prompted for File Name, if this is not found in the cross reference data base an error message will be displayed.

Having entered a valid file name, the file fields display will appear This shows all the fields contained in the file together with their type, size, description and, if relevant, the reference field and file.

If you press F6 the file field report will be printed showing the same information as was displayed. However, if you require printed layouts of this kind, you are advised to use the NUTIL command DSPFILE, which lists the information in a much better format than the one provided by XREF.

Files with no source code can be reviewed, however files which have not been created (although source is present) cannot be displayed.

File Cross Reference Inquiry

You will be prompted for File Name. If this is not found in the cross reference data base an error message will be displayed.

At this point you have two options: If you press F6, you will be shown the keylist and select/omit parameters of this file (as well as all its related logical views, if any). If you press enter, you will be shown all uses of the file (as well as all its related logical views, if any).

1. Enter was pressed:

If a logical file was selected, then all programs that use that logical will be displayed.

If a physical file was selected, then the following information will be displayed:

a) a list of programs that use the physical file
b) then the name of the logical that is alphabetically first
c) a list of all programs that use the logical file from (b)
d) the next logical file in alphabetical sequence.

In the list of programs that use the file, you may get a status indicator that shows when XREF could not determine whether the specified file was used in one of the programs listed:

An ***e1*** status flag indicates that the program uses a file with the same file name, but *LIBL is specified for the library name. So this file MAY be used, depending upon the contents of the library list when the program is run.

An ***e2*** status flag indicates that the program uses a file with the same file name, but a different library name. This may indicate that you are using duplicate copies of the file in multiple libraries.

2. F6 was pressed:

You will be shown the key information for the file selected, along with any select/omit logic defined (if the file was a logical file).
If the selected file was a physical file, you will also be shown the same level of information of each related logical file.

FILE CROSS REFERENCE - FILE/PROGRAM REFERENCES

If the file specified was a logical file the display will show the based on physical files and a list of all programs that reference that logical file.

If the file specified was a physical file the display will list all programs that reference that physical, then (in alphabetical sequence) each logical file that uses the specified physical followed by each program that references that logical view.

If you press F6 the file cross reference report will be printed showing the same information as was displayed.

FILE CROSS REFERENCE - FILE KEY SEQUENCE

If instead of pressing enter on the cross reference inquiry screen you pressed F6 then the Key Sequence display will be shown.

If the file specified was a logical file the display will show the 'based on' physical files and a list of all the key fields for that logical file.

If the file specified was a physical file the display will list all key fields for that physical, then (in alphabetical sequence) each logical file that uses the specified physical followed by the key fields for that logical view.

If you press F6 the file key sequence report will be printed showing the same information as was displayed.

If the file has no source then no key field data will be available.

If a program uses a file the file can be accessed even if it has no source and is not created.

Field in which Files Inquiry

You will be prompted for Field Name, if this is not found in the cross reference data base an error message will be displayed.

Having entered a valid field name the 'field in which file' display will be shown. This is a list of all physical and device files containing the specified field, also shown is the file library, the field type, field length and file description.

If you press F6 the 'field in which file' report will be printed showing the same information as was displayed.

Field in which Program Inquiry

You will be prompted for Field Name, if this is not found in the cross reference data base an error message will be displayed.

Having entered a valid field name the 'field in which program' display will be shown. This is a list of all programs (which have source code) which use specified field, also shown is the program type, program library, statement number, usage, field type, field length and program description.

If you press F6 the 'field in which program' report will be printed showing the same information as was displayed.

Non-RPG programs, and RPG programs with internally described files, will not appear in this inquiry.

Fields within RPGLE free-form code will not appear in this inquiry

Program Flow Listing

You will be prompted for Program Name, if this is not found in the cross reference data base an error message will be displayed.

Having entered a valid program name the 'program flow' report will be printed. This shows the same information as the program cross reference inquiry. However each program called will also have its program cross reference data shown.

For example, suppose program A calls program B and then Program C and in turn program B calls program B1. The reporting sequence will be:

A, B, C, B1

Remove an Application from the XREF database

Option 6 on menu UTLXR2 processes the RMVAPPXRF (Remove application from Cross Reference) command.

You will be prompted to enter the Application name to remove. When you press enter, a job will be submitted to remove the specified application references from XREF.

The submitted job may take a considerable time to run; its duration depends not only upon the size of the application to be deleted but also upon the size of the remaining applications.

Remove an Object from the XREF database

Option 7 on menu UTLXR2 processes the RMVOBJXRF (Remove object from Cross Reference) command.

You will be prompted to enter the Object name and type to remove. When you press enter, all references to that object will be removed from XREF.

Programmer Development Facility

The Programmer Development Facility function of XREF provides a more ‘intelligent’ version of the Programmers menu supplied by IBM, in that it ensures that a programmer never manipulates source members in a ‘Production’ source file.

Also, whenever any Production source member is replaced by a newer version, a copy is ‘archived’ so that a previous version is always available should it ever be necessary to revert to using an older version.

The Programmer Development Facility is set up to work around an installation which has up to five types of program library:

- MASTER: These libraries contain ”MASTER” versions of source code. These masters are considered as ‘read-only’ versions, and are never updated by this system. Master libraries are optional; you may not have this requirement.
- PRODUCTION: These libraries contain the ”LIVE” objects. For instance you may have one library containing all the programs and one for the files, or you may have a library for each application. These are the libraries that are in use by users of the system.
- TEST: These libraries are used to store modifications where development has finished and testing is in progress. These are the libraries used by system verification personnel, where the modification is to be approved before being installed in the production system.
- DEVELOPMENT: These libraries are used to store modifications in progress, as well as any new development. These are the libraries used by programmers for testing and development purposes.
- ARCHIVE: These libraries are used to store the prior versions of production source code. Archive libraries are optional; you may not have this requirement.

Programmers should only be allowed to manipulate source members in the Development libraries (this is the prime objective of this facility).

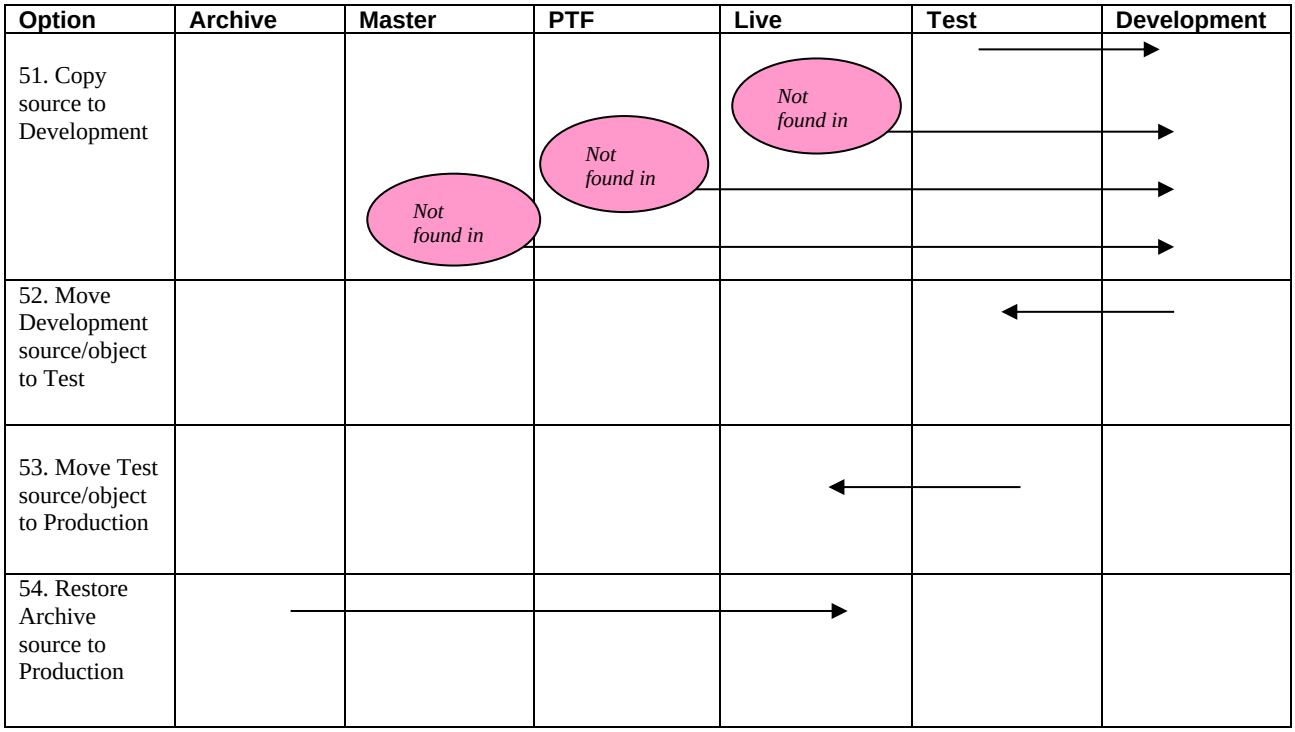
The only time the Production source is accessed is when a new version of a program is being installed in the user environment.

When a production source member is replaced by a new version, the old version of the source is always archived, rather than just replaced.

The source/object flow within the development facility

It is important to understand where the Development menu option causes the source to be *moved*, rather than just copied.

The following diagram shows the movement of source and objects between the defined environments, when using the Development menu options:



Legend:

Menu option 51 - Copy source to Development allows you to obtain the required source member and load it into the development environment.

The source is obtained from one of the defined environments, based on the following rules:

- You cannot move the source to Development if a Development copy already exists.
- If the source exists in the Test library, it will be moved (SO) into Development. IF NOT
- If the source exists in the Production library, it will be copied (SC) into Development. IF NOT
- If the source exists in the PTF/Upgrade library, it will be copied (SC) into Development. IF NOT
- If the source exists in the Master Source library, it will be copied (SC) into Development.

Note:

SC Source only is copied
SM Source only is moved for types PF and LF
SO Source and object are moved

Menu option 52 - Move Development source/object to Test
Menu option 53 - Move Test source/object to Production
Menu option 54 - Restore Archive source to Production

The Development Audit Log

Whenever an option on the Programmer Development Facility menu is used to move a source member or to create an object an audit record is written to file DEVLOG00. This audit file can be displayed using the DSPDEVLOG (Display Development Log) command.

Entries in file DEVLOG00 cannot be changed or deleted. If the file gets too large you must use a special program (XXC799) to cleanse it. This program works on the retention parameters defined via menu UTLINS, option 6. The program is included in the Scheduler job RGZ_NUTIL, which is designed to run on a monthly basis.

Installing your Development environment

If you do not already have development, test and archive libraries, you will need to set them up before using this facility. You do this by using the IBM CRTLIB (Create Library) and CRTSRCPF (Create Source Physical File) commands.

You will also need separate job descriptions with different initial library lists so that objects are created referencing the production objects (your 'live' library list), another which references the test objects (your 'test' library list) and another which references the development objects (your 'development' library list).

Defining Development Environment attributes

Before you can start using the Programmer Development Environment, you must define some controlling attributes. This is achieved using option 6 of the UTLINS menu:

```
XXR700D1        Development Menu Installation Attributes
Change



     Should objects be moved as well as source?..:*YES     (*YES, *NO)

     Retain development log information for......: 30days
     Frequency of Reorganises....................:  5days

     Date of last reorganise.....................: 09/02/15



F3=Exit
```

The 'should objects be moved' parameter defines how the movements between development, test and production will occur.

If you answer *YES, an attempt will be made to move the corresponding object as well as the source for development options 52 (move development to test) and 53 (move test to production). The movement of the source will be dependent on the movement of the object - if the object cannot be moved, the source move will not be attempted.

If the member attribute is PF or LF, the corresponding object move will not be attempted. This is to ensure the integrity of data objects. If you move a physical or logical file via option 52 or 53 you are only moving the source member. You must then manually recreate the object in the new environment (using option 55, 56 or 57).

The RETAIN parameter tells the system how long (in days) to keep the Development log information before it is automatically deleted. If this entry is zero, all log records are retained indefinitely.

The FREQUENCY parameter tells the system how often (in days) its maintenance routine is to check for obsolete log data. If this entry is zero, no test is performed for old information.

The entries made can be altered at any time by re-accessing the UTLINS menu option.

The WRKDEVPRJ Command

All development via this facility is grouped into PROJECTS. You define these projects via the WRKDEVPRJ (Work with Development Projects) command.

The definition of a project is entirely at your discretion. For example, it could be a new application being developed, a new function being developed or an existing application being corrected/upgraded.

The WRKDEVPRJ command has no parameters. After pressing Enter it will show a list of currently defined projects. If no projects have yet been defined, you will be prompted straight into Project Add mode:

```
XXR701D2              Development Control - Maintenance
Add

 Project Name.......: ACC_INTER
 Project Description: New Accounts Interface

 System Account Code: 3916-2214A
 Project Status.....: A              Project Manager....:  Sam Snyde
 Release Level......: 1              Release Date.......:    0/00/00
 Project Start Date.:    150198      Project End Date...:    0/00/00


    Library Control      Data          Object          Source
    Master                                           ACCTSMSTR
    PTF/Upgrade                                      ACCTSPTF
    Production          ACCTSFILE      ACCTSPGM      ACCTSSRC
    Test                ACCTSTEST      ACCTSTEST     ACCTSTEST
    Development         ACCTSDEV       ACCTSDEV      ACCTSDEV
    Archive                                          SRCARCHIVE

    Job Description Control
    Production        ACCTSFILE /ACCOUNTS
    Test              ACCTSTEST /ACCOUNTS
    Development       ACCTSDEV  /ACCOUNTS

F3=Exit   F11=Delete   F12=Previous   F16=*SRCF Overrides   F21=Command line
```

From this panel you define the controls for the project. The entries you can make are as follows:

Project Status: Status must be ‘A’ for work to be allowed on the project. Other valid entries are ‘S’ (project suspended) and ‘C’ (project complete). Note, however, that the status MUST be ‘A’ for it to be available to the Programmer Development Facility Menu.

System Account Code: If an entry is made here, any work performed via the development facility menu will be charged to this account number in OS/400 Job Accounting.

Library Control Data: This defines the libraries to be used for this project. The same library name can be used for one or more of the entries (for example, the Production Source and Production Object libraries could be the same library).

The ARCHIVE and PTF are optional libraries. PTF is not yet supported (future development for NUTIL). The ARCHIVE library is optional - If one is entered it will be processed in options 53 and 54 of the Programmer Development Facility menu. If you do not require an archive library, you may enter *NONE for the library name.

Job Description Control: This defines the job descriptions that will be used for any relevant job submissions (object creations, etc).

Also available is a Source File override panel, which is accessed via the F16 function key. Information on source file overrides is only required if you are not using the standard source file naming standard - QCLSRC, QCMDSRC, QRPGSRC etc:

```
XXR702D2              Source File Overrides - Maintenance
Add

Project Name.......: ACC_INTER   New Accounts Interface

          Source Member Attribute...:  RPG
                                                  Library
          Master Source File........:  MST_SRCF    ACCTSMST
          Production Source File....:  PRD_SRCF    ACCTSSRC
          Test Source...............:  TST_SRCF    ACITST
          Development Source........:  DEV_SRCF    ACIDEV
          Archive Source File.......:  RPG_ARC     SRCARC
          PTF/Upgrade Source........:  PTF_SRCF    SRCPTF


F3=Exit   F11=Delete   F12=Previous   F21=Command line
```

For each of the defined libraries within the project, enter the name of the source file that will be used for this source member type. The entry will be checked to ensure the source file exists in the specified library.

The STRDEVMNU Command

The Programmer Development Facility menu is similar in concept to the IBM Programmers Menu. It is started using the STRDEVMNU command:

```
                    Start PGMR Development Menu (STRDEVMNU)

Type choices, press Enter.

Development Project: . . . . . > ACC_INTER     Name, *PRV, *SELECT
Log Requests (*YES *NO): . . . .   *YES        *YES, *NO
Allow changes (*YES *NO):  . . .   *YES        *YES, *NO
Option 3 exit program: . . . . .   NCRTOBJECT  Name
  Library Name:  . . . . . . . .     NUTIL      Name, *LIBL
Delete option: . . . . . . . . .   *DLT         *DLT, *PROMPT, *NODLT


F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

The command is similar to the IBM STRPGMMNU (Start Programmer Menu) command, with the following exceptions:

PROJECT: You must specify which project you are working on. The entry will be checked to ensure it is a valid project, and that the project is at status ‘A’ (Active).

If you specify *PRV, the previous project you worked on via the STRDEVMNU command will be used.

If you specify *SELECT, you will be given a list of currently active projects to select from.

If you are using the NUTIL Programmer Environment, you can have the programmer use this as the initial work panel by entering *STRDEVMNU as the initial request data in the NCHGSIGNON command. This will initiate the interactive job using the command STRDEVMNU PROJECT(*PRV).

You must ensure that the programmer always signs on with the Development library list. You can do this by specifying the Development Job Description on the programmer’s user profile. Programmers should never use the Production library list, other than to move completed Development work into a production library.

The Programmer Development Facility Menu looks like this:

```
 ACC_INTER             Programmer Development Facility Menu
 New Accounts Interface                                 System:  NAVAN

 Select one of the following:
      1. Start Data File Utility
      2. Work with iSeries Query
      3. CALL NUTIL/NCRTOBJECT                 object name, type, pgm for CM
      4. Call a program                        program name
      5. Run a command                         command
      6. Submit a job                          (job name), , ,(command)
      7. Go to a menu                          menu name
      8. Edit a source file member             (srcmbr), (type)
      9. Design format using SDA/RLU           (srcmbr)
                                                                   More...
 Selection . . . . .    ____          Parm . . . .  _________
 Type  . . . . . .      __________    Parm 2 . . .  ____________________
 Command . . . . . .    ____________________________________________
                        ____________________________________________
                    Test         Development   Archive
Source library  . . ACCTSTEST    ACCTSDEV      SRCARC
Object library  . . ACCTSTEST    ACCTSDEV
Data library  . . . ACCTSTEST    ACCTSDEV      F21=Change Project
Job description . . ACCOUNTS     ACCOUNTS      F23=Display Project Info

F3=Exit  F4=Prompt  F6=DSPMSG  F10=Command entry  F12=STRPGMMNU  F14=WRKSBMJOB
```

The entries within the work panel are based on the entries made in the STRDEVMNU command, as well as from entries retrieved from the Project definition.

Once you are onto the menu, it is possible to change to another project at any time by using the F21 function key. This will display a list of active projects and allow you to select the project to work on.

Because of the additional options available on this menu (compared to the normal IBM Programmer Menu) you must use the PAGE keys to display all of the options:

```
 ACC_INTER             Programmer Development Facility Menu
 New Accounts Interface                                 System:  NAVAN

 Select one of the following:
   51. Copy source to Development
   52. Move Development source/object to Test
   53. Move Test source/object to Production
   54. Restore Archive source to Production
   55. Create Development object
   56. Create Test Object
   57. Create Production object
   58. Work with Development source
   60. Display Development Log
   90. Sign off                              (*nolist, *list)      Bottom

 Selection . . . . .      _____      Parm . . . .  __________
 Type  . . . . . .        _________  Parm 2 . . .  ____________________
 Command . . . . . .      ____________________________________________
 ____________________________________________________________________

                     Test         Development   Archive
Source library  . . ACCTSTEST    ACCTSDEV      SRCARC
Object library  . . ACCTSTEST    ACCTSDEV
Data library  . . . ACCTSTEST    ACCTSDEV      F21=Change Project
Job description . . ACCOUNTS     ACCOUNTS      F23=Display Project Info

F3=Exit  F4=Prompt  F6=DSPMSG  F10=Command entry  F12=STRPGMMNU  F14=WRKSBMJOB
```

The menu options perform as follows:

Option 1. Start DFU (Data File Utility) performs in exactly the same way as the standard IBM Programmers Menu. It accesses the STRDFU command.

Option 2. Start Query performs in exactly the same way as the standard IBM Programmers Menu. It accesses the STRQRY command.

Option 3. Create an Object performs in exactly the same way as the standard IBM Programmers Menu, using the exit program defined on the STRDEVMNU command. The source file will be the DEVELOPMENT source and the Object library will be the DEVELOPMENT object library. The job description used for submission will be the DEVELOPMENT job description. Please refer to the chapter entitled "NUTIL Compiler Facilities", earlier in this manual, for a discussion of the NUTIL Programmer Exit Program and how it invokes compilers based on the source member type.

Option 4. Call a program performs in exactly the same way as the standard IBM Programmers Menu, by calling the requested program. The library name will be the DEVELOPMENT object library.

Option 5. Run a command performs in exactly the same way as the standard IBM Programmers Menu. It will process the command entered on the Command line.

Option 6. Submit a job performs in exactly the same way as the standard IBM Programmers Menu, submitting the command entered on the command line. The job description used for submission will be the DEVELOPMENT job description.

Option 7. Go to a menu performs in exactly the same way as the standard IBM Programmers Menu. It accesses the GO command.

Option 8. Edit a source file member performs in the same way as the standard IBM Programmers Menu, only a little better. It will access EITHER the STRSEU or EDTTXT (if it exists on your machine) command, depending on the source member attribute. The source file will be the DEVELOPMENT source.

Option 9. Design format using SDA/RLU performs in the same way as the standard IBM Programmers Menu, only a little better. It will access EITHER the STRSDA or STRRLU command, depending on the source member attribute. The source file will be the DEVELOPMENT source.

Option 51. Copy source to Development allows you to obtain the required source member and load it into the development environment.

The source is obtained from one of the defined environments, based on the following rules:

> You cannot move the source to Development if a Development copy already exists.
> If the source exists in the Test library, it will be moved into Development. IF NOT
> If the source exists in the Production library, it will be copied into Development. IF NOT
> If the source exists in the PTF/Upgrade library, it will be copied into Development. IF NOT
> If the source exists in the Master Source library, it will be copied into Development.

An important note on the move from Test to Development: If the source member is to be copied from the Test environment back into the Development environment, the system assumes that you are returning it for rework.

Thus if the Installation Attributes specify that Object moves are to occur also, it will move the corresponding object back from Test into Development as well (this does not occur if the member attribute is PF or LF).

Option 52. Move Development source/object to Test will move the specified source member into the Test environment. Depending on installation attributes, it will also move a corresponding object (if one exists) *provided that* the source member attribute is not PF or LF.

If an object move is to be attempted (Installation attributes specify *YES, and member attribute is not PF or LF), the source move will be dependent on the object move: If the object cannot be moved, the source move will not be attempted.

Option 53. Move Test source/object to Production will move the specified source member into the Production environment. Depending on installation attributes, it will also move a corresponding object (if one exists) *provided that* the source member attribute is not PF or LF.

If an object move is to be attempted (Installation attributes specify *YES, and member attribute is not PF or LF), the source move will be dependent on the object move: If the object cannot be moved, the source move will not be attempted.

Prior to the source move occurring, if an Archive library is specified for this project, a copy of the Production source is moved into the Archive source file.

Option 54. Restore Archive source to Production source allows you to return the Archived copy of the Production source back into Production. It is only available if an Archive library has been defined for this project.

Prior to the source move occurring, the current Production source is moved back into the Test source file.

Option 55. Create development object from development source performs the same function as option 3, using the exit program defined on the STRDEVMNU command. The source file will be the DEVELOPMENT source and the Object library will be the DEVELOPMENT object library. The job description used for submission will be the DEVELOPMENT job description. Please refer to the chapter entitled "NUTIL Compiler Facilities", earlier in this manual, for a discussion of the NUTIL Programmer Exit Program and how it invokes compilers based on the source member type.

Option 56. Create Test object from Test source performs the same function as option 3, using the exit program defined on the STRDEVMNU command. The source file will be the TEST source and the Object library will be the TEST object library. The job description used for submission will be the TEST job description. Please refer to the chapter entitled "NUTIL Compiler Facilities", earlier in this manual, for a discussion of the NUTIL Programmer Exit Program and how it invokes compilers based on the source member type.

Option 57. Create Production object from Production source performs the same function as option 3, using the exit program defined on the STRDEVMNU command. The source file will be the PRODUCTION source and the Object library will be the PRODUCTION object library. The job description used for submission will be the PRODUCTION job description.

Option 58. Access Development source via PDM accesses the DEVELOPMENT source member via the WRKMBRPDM command.

Option 60. Display Development Log performs the DSPDEVLOG (Display Development Log) command. This displays a log of all changes made (within the project being processed) to source, as well as any object creations. It will only show history information for options performed on the Programmer Development Facility menu.

Option 90. Sign off performs in exactly the same way as the standard IBM Programmers Menu. It access the SIGNOFF command.

The WRKDEVLCK Command

There is also a facility to stop the use of any Development Facility Menu Option by any or all users within a specific project. This is defined by using the WRKDEVLCK (work with Development Locks) command:

```
                  Work with DEVMNU Option Locks (WRKDEVLCK)

Type choices, press Enter.

Development User:  . . . . . .   *ALL          Name, *ALL
Development Project: . . . . .   ACC_INTER     Name, *SELECT



F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

Parameter Definitions

Development User: You can enter a specific User Profile which you intend to control, or you can specify *ALL for all users within the project.

Development Project: Specify the project for which you are defining option locks. If you specify *SELECT, wou will be given a list of current projects.

Using the information entered in the command parameters, you will then be given a panel showing all the locks that have been currently defined against the specified user in the specified project:

```
XXR703D1              Development Menu Locks - Maintenance

Type options, press Enter.
  4=Delete

User Profile: *ALL        Project Name: ACC_INTER

      Menu
      Option   Description of Option Locked
    _  52      Move Development source/object to Test
    _  56      Create Test Object
       __
       __
       __
       __
       __
       __

    Position list to:
       53

F3=Exit
```

To add a new lock for the user, enter the menu option number that is to be locked. You will then be given the title of the option for your own verification of the correct option being locked.

To remove an existing lock, type a ‘4’ beside the option and it will be unlocked.

Work with Database Files

An Overview

WRKDBF is a PDM styled database file manipulation tool that allows easy access to the main operations normally performed on database files. In addition it provides file display and edit capabilities not provided by standard OS/400.

WRKDBF allows you to

.. Edit and display database file data
.. Copy files (either via CRTDUPOBJ or CPYF)
.. Save and restore files
.. Clear, Reorganise, Delete and Move files
.. Start and end journalling
.. Display file, field and database relationship information
.. Work with Query or Data File Utilities
.. Cleanse files by selection string
.. Send/Receive network files

All from the one display panel.

The EDTDBF function of WRKDBF is a full screen database file update utility designed to run on the IBM iSeries. It is capable of displaying data base records in both physical and logical files in either keyed or arrival sequence without any programming effort.

For performance reasons, EDTDBF supports a maximum record length of 4096 bytes.

EDTDBF allows you to

.. display journal records extracted by the **DSPJRN** command
.. display records in keyed sequence
.. display records in arrival sequence
.. display records in hexadecimal format
.. display record contents by field name
.. add records by field name
.. update records by field name
.. copy records by field name
.. delete records
.. print records by field name
.. window by field name
.. scan records for a character string
.. scan records for a hexadecimal string

EDTDBF does NOT support the following:

.. updating of Join files
.. updating of logical files based on multiple members
.. detailed data views of join logical files with an outer join (see below)
.. *NULL values in null-capable fields
.. Floating point data type fields

.. Encoded Vector Index (*EVI) files

The DSPDBF function of WRKDBF provides all the facilities of the EDTDBF except for the file updating capability.

DSPDBF Join Logical File Limitation

In a Join Logical (left outer join) database file, the OS/400 database I/O feedback area returns the relative record number (RRN) of the record as it resides in the ***primary*** Physical File in the join - it ***is not*** the "RRN of the record" in the Join Logical (*JLF)

Because of this, the *SETLL and *RRN main browse displays show the Relative Record Number of the record "correctly"; as that is the number returned to us in the feedback area. This may mean you will see more than one record with the same relative record number (as this is a reflection of the primary file in the join).

In the *KEY browse display you see the records sequenced differently - so it looks correct on the main browse display, but if you select either of them (to view the full record details) they will still show only the same (first) record details.

As such, because of this system limitation, the "detailed processing" of data represented in Join Logical views may not produce the expected result. It is recommended that the IBM Query product be used to view data in Join Logical Files, especially those defined as Left Outer Joins.

Working with Database files

The WRKDBF utility is accessed using the WRKDBF (Work with Database files) command:

```
                Work with Database Files (WRKDBF)

Type choices, press Enter.

 File name  . . . . . . . . . . . FILE          sch*
   Library name . . . . . . . . .                 nutil

 File Attribute . . . . . . . . . FILEATR       *ALL
```

FILE: The name of the file(s) to be displayed. If library name is not entered, then *LIBL is used. Generic entries are allowed.

FILEATR: The file attribute to select, which may be:

*PF	select Physical files only
*LF	select Logical files only
*ALL	select all files

Note that source files are not processed by this selection, only data files. You can, however use the EDTDBF or DSPDBF commands directly if you wish to process source files (although you would normally use the IBM Source Entry Utility STRSEU for this purpose).

Based on the entries made, WRKDBF will then build up a file list and display the WRKDBF main function panel:

```
WRKDF1D1              Work with Database files          System:  NAVAN

 Type options, press Enter.

 2=Edit  3=Copy (CRTDUPOBJ)  4=Delete  5=Display  6=Clear  7=Rename
 8=Display file description  9=Save  10=Restore  11=Move  12=Work with file ..


    File        Library     Attr    Description

 __ SCHCLD00    NUTIL       PF      Calendar Exceptions File
 __ SCHCMD00    NUTIL       PF      Standard Job Commands File
 __ SCHCOD00    NUTIL       PF      Standard Job Run Codes
 __ SCHCOD01    NUTIL       LF      Standard Job Run Codes, Code Sequence
 __ SCHEXTP     NUTIL       PRTF    Standard Jobs Submitted
 __ SCHHST00    NUTIL       PF      Scheduler Run History Log
 __ SCHHST01    NUTIL       LF      History Log (JHLIST='N')
 __ SCHHST02    NUTIL       LF      History Log by Exec Date/Time/Job
 __ SCHJOB00    NUTIL       PF      Scheduled Jobs File
 __ SCHJOB01    NUTIL       LF      Scheduled Jobs File (SHSBM='N')

 Position to file: SCHCLD00   NUTIL                              More..
 Parameters for options, or command
 ===>

 F3=Exit  F4=Prompt  F5=Refresh  F6=Create  F9=Retrieve  F10=Command entry
 F12=Cancel  F23=More options
```

If you are familiar with OS/400 Programming Development Manager (PDM) this screen will seem familiar to you. That is because it was designed to operate in a similar way.

The main area of the screen lists all of the files selected for processing and on the left-hand side of each file name is an option entry area.

At the top of the screen is listed all of the options available (because there are so many options available you must use F23 to see the rest...). At the bottom of the screen is a command entry line, which also doubles as a parameter entry field. Also at the bottom is a list of command keys available.

The 'Position to file' entry allows you to position the list to a specific file name, rather than using the page keys to find it.

WRKDBF Options

The WRKDBF panel options operate on the specific file they are entered against. If you request multiple selections (by typing in options on more than one line) you will process all of those files.

The parameter line at the bottom of the screen allows you to enter additional command options. For example you may require a file to be reorganised in *KEYFILE sequence, so you enter option 16 beside the file to process and 'KEYFILE(*FILE)' on the parameter line.

Each of the options available is explained below, but for full details of OS/400 standard commands please refer to the IBM iSeries Control Language Reference Guide.

Option 1 - Display Basic Information

This option will display the following screen of information about the file selected:

```
WRKDF1D2                Work with Database Files         System:  NAVAN
Display

        File...............: SCHCLD00
        Library............: NUTIL
        File Attribute.....: PF

        Description........: Calendar Exceptions File

        Creation date......: 021015
        Creation time......: 154443

        Number of formats..: 00001
        Number of members..: 00001

        External Descriptn.: Y
        Access path type...: K
        Select/omit?.......: N
        Join Logical?......: N


F3=Exit   F12=Previous
```

This a display of the basic information on the file that is used by the WRKDBF command.

Option 2 - Edit Database file

This option will perform the NUTIL EDTDBF (Edit Database file) command, which is described in detail in this manual. If the file has more than one member, you will be prompted to enter the member name.

Option 2 is only available for Database files.

Option 3 - Copy (Using CRTDUPOBJ)

This option will perform the OS/400 CRTDUPOBJ (Create Duplicate Object) command. You will be prompted to enter additional command parameters.

Care should be taken when using this option on a Logical file, as the Physical file on which the newly created Logical file will be based-on can vary. The topic is discussed in more detail in the OS/400 reference manual.

Option 4 - Delete

This option will perform the OS/400 DLTF (Delete File) command. You will be prompted to accept the request before the file delete is performed.

Option 5 - Display Database file

This option will perform the NUTIL DSPDBF (Display Database file) command, which is described in detail in this manual. If the file has more than one member, you will be prompted to enter the member name.

Option 5 is only available for Database files.

Option 6 - Clear

This option will perform the OS/400 CLRPFM (Clear Physical file member) command. If the file has more than one member, you will be prompted to enter the member name.

Option 6 is only available for Physical files.

Option 7 - Rename

This option will perform the OS/400 RNMOBJ (Rename Object) command. You will be prompted to enter additional command parameters.

After successfully renaming the file it will not be accessible from the WRKDBF panel until the F5 (Refresh) function is performed.

Option 8 - Display File Description

This option will perform the OS/400 DSPFD (Display File Description) command.

You are not prompted for the command and the output is assumed to be to the display. If you wish to print a listing, specify 'OUTPUT(*PRINT)' on the parameter line before pressing ENTER.

Option 9 - Save

This option will perform the OS/400 SAVOBJ (Save Object) command. You will be prompted for additional command parameters.

Option 10 - Restore

This option will perform the OS/400 RSTOBJ (Restore Object) command. You will be prompted for additional command parameters.

Option 11 - Move

This option will perform the OS/400 MOVOBJ (Move Object) command. You will be prompted to enter additional command parameters.

After successfully moving the file it will not be accessible from the WRKDBF panel until the F5 (Refresh) function is performed.

Option 12 - Work with file

This option will perform the WRKDBFMBR (Work with Database file members) command which is an extended function of the NUTIL EDTDBF (Edit Database file) command, described in detail in this manual.

WRKDBFMBR allows you to manipulate members in a multi member file.

Option 12 is only available for Database files.

Option 13 - Change Description

This option will perform the OS/400 CHGPF or CHGLF (Change Physical/Logical file) command, dependent upon the file attribute. You will be prompted to enter additional command parameters.

Option 13 is only available for Physical and Logical files.

Option 14 - Clean

This option will perform the NUTIL CLNPFM (Cleanse Physical File member) command. You will be prompted to enter additional command parameters.

CLNPFM allows you to selectively remove or archive records from a file by entering selection information. Correct syntax for the QRYSLT parameter is described in the OS/400 Control Language Reference manual, under the OPNQRYF (Open Query File) command.

For example, STATUS *EQ “D” will remove all records from the file where the field STATUS has a D in it.

Option 14 is only available for Physical files.

Option 15 - Copy (using CPYF)

This option will perform the OS/400 CPYF (Copy File) command. You will be prompted to enter additional command parameters.

Option 15 is only available for Physical and Logical files.

Option 16 - Reorganise

This option will perform the OS/400 RGZPFM (Reorganise Physical file member) command. If the file has more than one member, you will be prompted to enter the member name.

Option 16 is only available for Physical files.

Option 17 - Query

This option prompts the OS/400 WRKQRY command.

Option 18 - Update Data (DFU)

This option will perform the OS/400 UPDDTA (Update Data) command, which will generate a temporary DFU over the file for you to update file data.

Option 18 is only available for Physical and Logical files.

Option 19 - Start Journalling

This option will perform the OS/400 STRJRNPF (Start Journalling Physical file) command. You will be prompted to enter additional parameters.

Option 19 is only available for Physical files.

Option 20 - End Journalling

This option will perform the OS/400 ENDJRNPF (End Journalling Physical file) command. You will be prompted to enter additional parameters.

Option 20 is only available for Physical files.

Option 21 - Display Database Relations

This option will perform the OS/400 DSPDBR (Display Database Relations) command.

You are not prompted for the command and the output is assumed to be to the display. If you wish to print a listing, specify 'OUTPUT(*PRINT)' on the parameter line before pressing ENTER.

Option 21 is only available for Physical files.

Option 22 - Display File Field Descriptions

This option will perform the OS/400 DSPFFD (Display File Field Descriptions) command.

You are not prompted for the command and the output is assumed to be to the display. If you wish to print a listing, specify 'OUTPUT(*PRINT)' on the parameter line before pressing ENTER.

Option 23 - Send Network File

This option will perform the OS/400 SNDNETF (Send Network File) command. You will be prompted to enter additional parameters.

Option 24 - Receive Network File

This option will perform the OS/400 RCVNETF (Receive Network File) command. You will be prompted to enter additional parameters.

WRKDBF command function keys

The WRKDBF panel function keys are as follows:

F3	Exit the WRKDBF utility.
F4	Prompt the command entered on the command line.
F5	Rebuild the work display. This will refresh the screen and correct the list where files have been moved renamed or deleted.
F6	The CMDCRT (Create Commands) OS/400 menu panel is displayed.
F9	Retrieve the last command entered on the command line.
F10	Display a command entry screen.
F12	Exit the WRKDBF utility.
F23	More options. Because of the number of options available to you there are two lists that can be displayed at the top of the screen. The F23 key lets you switch between the two lists.

Editing and Displaying Database file data

The EDTDBF and DSPDBF commands

Behind options 2 and 5 of the WRKDBF panel are two NUTIL commands, namely EDTDBF (Edit Database file data) and DSPDBF (Display Database file data). While these commands are part of the WRKDBF utility they can be accessed from a normal command entry screen. The following section of the manual describes their operation.

EDTDBF and DSPDBF Display Modes

There are three display methods that may be selected upon entry or by command key:

1. .. *RRN display file by relative record number
2. .. *KEY display file by key sequence
3. .. *SETLL display file by key sequence and allow file positioning

*RRN: If the file contains NO key fields, then this is the only display method available. However, logical files may also be displayed in ARRIVAL sequence provided NO SELECT/OMIT statements were used when creating the logical view. This is an ideal method of determining records that have been added by a physical or logical file by simply by displaying the last few records in the file. The record number displayed in position 73-79 of the EDTDBF display is the actual relative record number held on file.

*KEY: To display a file in key sequence, then the file must have a keyed access path, otherwise the *RRN display method must be used. The record number displayed in position 73-79 of the EDTDBF display is a program generated record number (to enable EDTDBF to keep track of which record it is processing) and NOT the actual relative record number held on file.

*SETLL: To display a file in key sequence, then the file must have a keyed access path, otherwise the *RRN display method must be used. The record number displayed in position 73-79 of the EDTDBF display is the actual relative record number held on file.

To show which mode you are currently using, you will see a two character ‘file type’ followed by the selected display mode (*RRN, *KEY or *SETLL) on line 1 of the EDTDBF main panel:

AR - Arrival Sequence
KF - Keyed, First In
KU - Keyed, Unique

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

  Control: __________ Scan: _____________________________    __1   _151  W: 000
  Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

  0000001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
  0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
  0000003 ---MR   MILES     A T                S-oa?ACCTS     ---‘LONG GREEN’
  0000004 ---MS   OWENS     I U                A—E-EDP       ---15 ARVONSON P
  0000005 --?MS   BROWN     I M                A-/--EDP       ---9978 RISEBANT
```

To switch between the various display modes (if this is allowed, dependent upon file type) you simply type ‘KEY’ ‘RRN’ or ‘SETLL’ in the control field at the top of the screen.

Generating Record Format Information

The WRKDBFGEN (Generate WRKDBF Record Format Information) command is used to generate Record Format Information over a whole library or for a generic object name. This method of storing record format information over a set of files will ensure that extended display, update and print functions are available immediately, otherwise they have to be generated at the time an extended EDTDBF function has been requested.

The prompt looks like this:

```
         Generate DSPFFD Information (WRKDBFGEN) Prompt

Type choices, press Enter:

  File name . . . . . .  FILE       R
    Library name  . . . . . .                    *LIBL
```

Re-Organizing Record Format Information

It is recommended that the WRKDBFRGZ (Reorganise WRKDBF File Information) command be run periodically to remove redundant Record Format Information for files that no longer exist.

The command has no parameters.

Using the EDTDBF and DSPDBF commands

The EDTDBF and DSPDBF commands operate in the same way, the main difference being that EDTDBF allows you to change file information, whereas DSPDBF only allows you to display it. The command prompt for both functions is similar, and looks something like this:

```
             Edit Database file data (EDTDBF) Prompt

Type choices, press Enter:

  File name:                        FILE       R
    Library name:                                    *LIBL

  Member name:                      MBR            *FIRST

  Record format creation check      CHKFMT         *YES
```

FILE: The name of the file to be displayed. If library name is not entered, then the library list is used to search for the specified object.

MBR: Member name to be displayed, if not entered then the first member in the object will be assumed.

CHKFMT: The WRKDBF utility maintains a database containing the attributes of database files on your iSeries. To ensure these are correct, this parameter will access the control information for the file and verify it has not changed since the last time it was accessed by the WRKDBF utility. If it has changed, and the CHKFMT parameter was set to *YES, it will automatically rebuild the control information for this file.

If your file contains multiple members and you don't know the names of the members, then specify *SELECT and EDTDBF will display a list of ALL members within the object and allow you to select the member you wish to display:

The member selection display will look something like this:

```
NUTIL                     WORK WITH DATA BASE FILE
WS51                          MEMBER SELECTION

File    : QDDSSRC
Library : NUTIL

   Member      Text Description                                 No. Records

_ CHKOBJ      Check Source/Object Compatibility                      19
_ CHKOBJP     Check Source/Object Printer File                       49
_ CHKSRC      Check Source/Object Compatibility                      20
_ CHKSRCP     Check Source/Object Printer File                       44
_ DSPHLPD     DSPHLP Display Help Facilty                            51
_ MNUAUT      Menu Authority Master File                             26
_ MNUCHK      Menu Machine Serial No. Master File                    24


To Display member - Key "5" and press ENTER
To Remove  member - Key "4" and press ENTER

F3=Exit
```

Rebuilding WRKDBF Information after using CHGPF

If you use the IBM CHGPF command to change a file descriptor, it is possible that the next usage of WRKDBF may not pick up the fact that the file has changed. This is due to the fact that the level ID of the file is not always changed when CHGPF is used.

If you do use CHGPF against a file known to be used with WRKDBF, we highly recommend you issue the WRKDBFGEN command to rebuild the WRKDBF information for the changed file.

```
            Build WRKDBF information (WRKDBFGEN) Prompt

Type choices, press Enter:

  File name:                        FILE       R
    Library name:                                 *LIBL
```

FILE: The name of the file for information to be rebuilt. If library name is not entered, then the library list is used to search for the specified object.

Displaying journal data

The DSPDBF command is also capable of allowing you to view formatted data from journals. It does this by accessing the outfile created by the DSPJRN (Display Journal Entries) command:

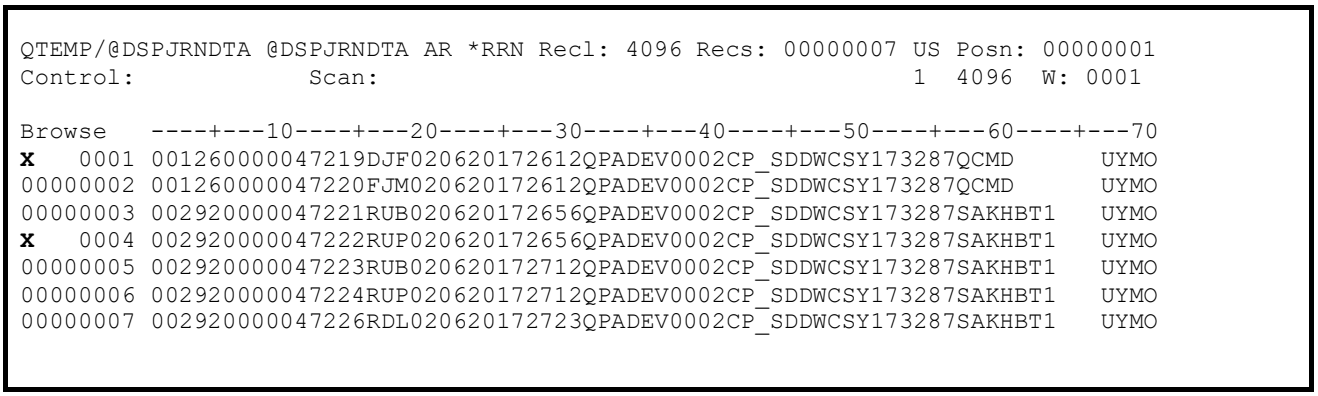
DSPJRN JRN(*journal_name*) FILE((*file_name*)) OUTPUT(*OUTFILE) OUTFILFMT(*TYPE1)
OUTFILE(QTEMP/@DSPJRNDTA) ENTDTALEN(3971)

This feature will only work correctly where the record length of the journalled file is less than 3972. As shown on the DSPJRN command sample above you must specify the output file format as *TYPE1 data and the entry data length of 3971. If you do not do this, journalled information will not display correctly. The name of the outfile is not important, you can specify whatever name you choose.

Once you have created the outfile from the DSPJRN you can run the DSPDBF command to display the contents:

DSPDBF FILE(QTEMP/@DSPJRNDTA)

The first display that will be shown is the standard DSPDBF display panel:



```
QTEMP/@DSPJRNDTA @DSPJRNDTA AR *RRN Recl: 4096 Recs: 00000007 US Posn: 00000001
Control:            Scan:                                    1  4096  W: 0001

Browse   ----+---10----+---20----+---30----+---40----+---50----+---60----+---70
x   0001 001260000047219DJF020620172612QPADEV0002CP_SDDWCSY173287QCMD      UYMO
00000002 001260000047220FJM020620172612QPADEV0002CP_SDDWCSY173287QCMD      UYMO
00000003 002920000047221RUB020620172656QPADEV0002CP_SDDWCSY173287SAKHBT1   UYMO
x   0004 002920000047222RUP020620172656QPADEV0002CP_SDDWCSY173287SAKHBT1   UYMO
00000005 002920000047223RUB020620172712QPADEV0002CP_SDDWCSY173287SAKHBT1   UYMO
00000006 002920000047224RUP020620172712QPADEV0002CP_SDDWCSY173287SAKHBT1   UYMO
00000007 002920000047226RDL020620172723QPADEV0002CP_SDDWCSY173287SAKHBT1   UYMO
```

You can see (at the top of the display) that the record length of the outfile is shown as 4096. This is made up as follows:

Field	*Attributes*		*From*	*To*	*Text Description*
JOENTL	5 S	0	1	5	Length of entry
JOSEQN	10 S	0	6	15	Sequence number
JOCODE	1 A		16	16	Journal Code
JOENTT	2 A		17	18	Entry Type
JODATE	6 A		19	24	Date of entry: Job date format
JOTIME	6 S	0	25	30	Time of entry: hour/minute/second
JOJOB	10 A		31	40	Name of Job
JOUSER	10 A		41	50	Name of User
JONBR	6 S	0	51	56	Number of Job
JOPGM	10 A		57	66	Name of Program
JOOBJ	10 A		67	76	Name of Object
JOLIB	10 A		77	86	Objects Library
JOMBR	10 A		87	96	Name of Member
JOCTRR	10 S	0	97	106	Count or relative record number changed
JOFLAG	1 A		107	107	Flag: 1 or 0
JOCCID	10 S	0	108	117	Commit cycle identifier
JOINCDAT	1 A		118	118	Incomplete Data: 1 or 0
JORES	7 A		119	125	Not used
JOESD	3971 A		126	4096	Entry Specific Data

Positions 1-125 are information relating to the journal entry; positions 126-4096 (field JOESD) contain the variable data which shows before and/or after images of the database record for a record level journal entry.

In the same way as for a normal database file, you can select a specific journal entry for display by typing an ‘X’ in the first position of the RECNBR field.

For a file or member level journal entry, the detail display will appear like this:

```
                Display Journal Record

File name    : DATIL/UYMOLATE        Member
Job name     : 173287/CP_SDDWCSY/QPADEV0002
Program name : QCMD        Journal Date : 07/01/15 Time : 14:26:12

Journal code      : D  Database file operation
Journal entry type: JF Start journaling for file


F3=Exit    F12=Previous
```

As you can see, the information for the selected journal entry has been formatted based on the journal code in the file. In this example this is a database operation at file level and so there is no "entry specific" data to be shown.

```
                          Display Journal Record

File name    : DATIL/UYMOLATE        Member UYMOLATE       Rcdfmt : EMPLDTA
Job name     : 173287/CP_SDDWCSY/QPADEV0002               Rcdlen :   167
Program name : SAKHBT1     Journal Date : 07/01/15 Time : 14:26:56

Journal code      : R  Record level operation
Journal entry type: UP After-image of record updated in physical file member

Field      Key     Attr ----+---10----+---20----+---30-- Text Description
EMPNO      K1  A    5P0 00003                           Employee Number
TITLE               5A  MR                              Employee Title
NAME               30A  MILES     A T                   Employee Name
STATUS              1A  S                               Status
STRDT               8P0 20030615                        Commencement Date
DEPT               10A  ACCTS                           Department
CLASS               5P0 00102                           Classification
ADDR1              30A  'LONG GREEN'                    Employee Address 1
ADDR2              30A  GRANTON                         Employee Address 1
STATE              10A  OPL                             State
PCODE              10A  6534                            Postal Code
TELE               15A  566 1024                        Telephone Number

                                                                    More...

F3=Exit    F12=Previous   F23=Fold
```

In this second example we see a database operation at record level; in this case it is the "after" image of a record that has been changed. When a record level change is detected by DSPDBF, the contents of the "Entry Specific Data" in field JOESD are formatted based on the file name, so instead of seeing a "raw data representation" of field JOESD in the display you see the formatted layout of the database record.

Command Function Keys

Function Key Summary

Command Function keys are ‘soft coded’, which means they are supplied in a file which allows you to change their assignments or definitions. The definitions are stored in file NUTIL/WRKDBFP. As supplied, the command keys function as follows:

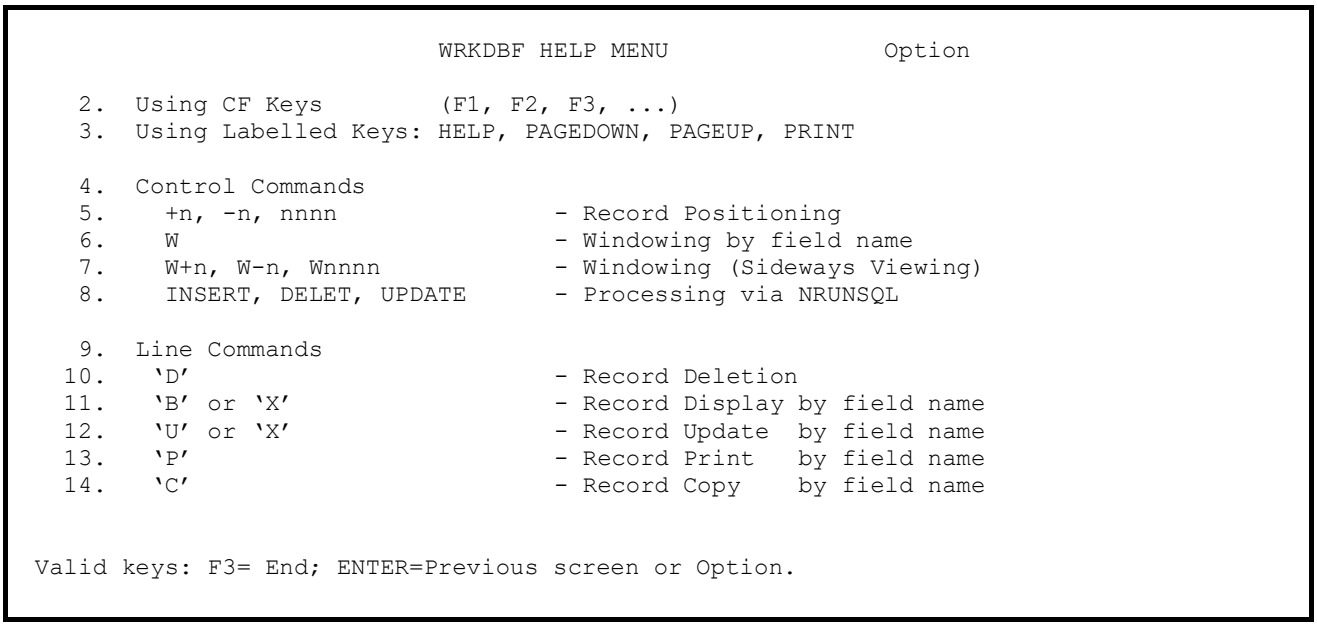
Function Key	*Supplied Assignment*
1	Display Help
2	Switch between upper and lower case entry
3	Exit
4	Prompt field names for *SETLL
5	Refresh/Cancel pending operations
6	Add record
7	Switch hex on/off
8	Switch between hex format 1 and 2
9	Switch between edit and browse modes
12	Previous
13	Generate file field information
15	Position file using key information in scan field
16	Scan forwards
17	Scan backwards
19	Window left
20	Window right
21	Display command line window
22	Switch to *RRN mode
23	Switch to *KEY mode
24	Switch to *SETLL mode

F1=Display Help

The F1 or the HELP key (either can be used) is used to display help about a specific function within the WRKDBF program product. Position the cursor on the specific area of the edit panel you require help information for and press the HELP key.

You can also type ‘HELP’ in the control field to access a full help facility.

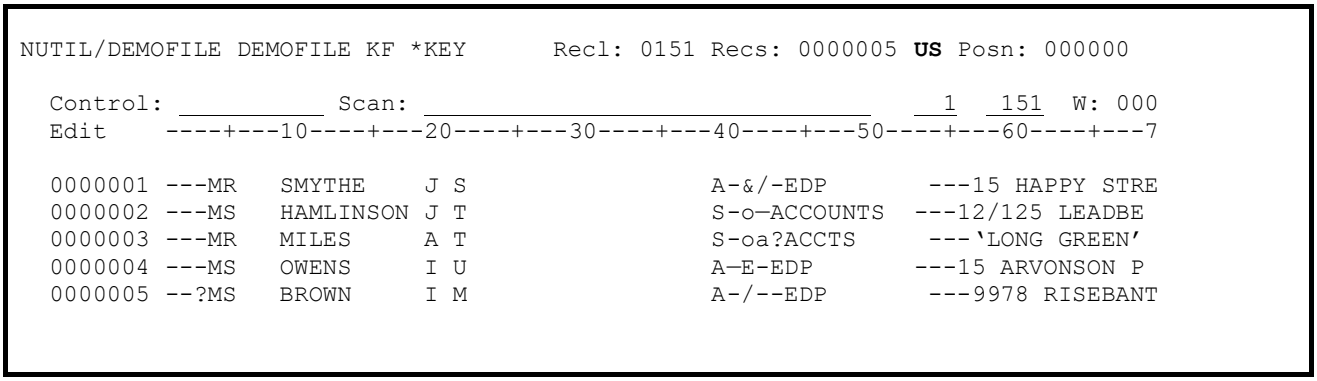
The HELP facility main menu looks like this:



F2=Switch Upper/Lower Case

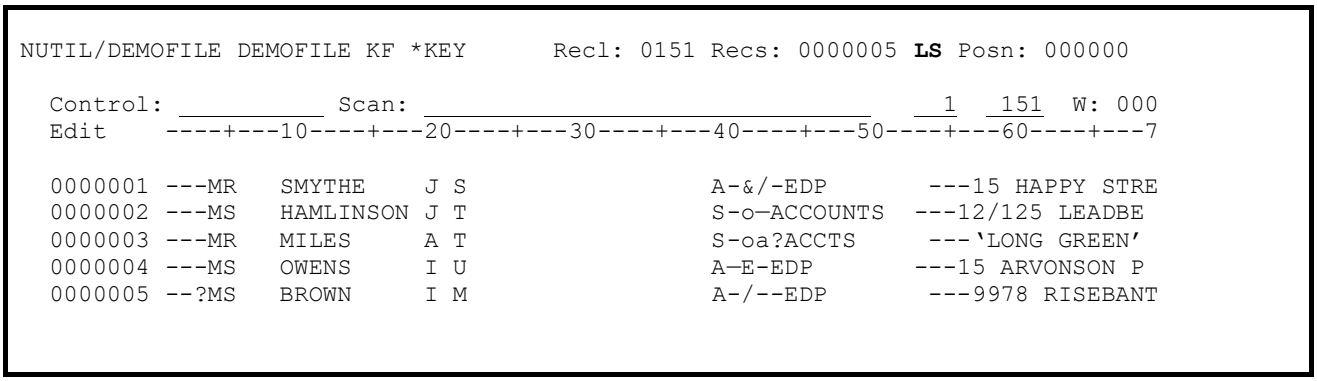
Pressing F2 (or typing 'CASE' in the control field) switches back and forth from upper to lower case. Your keying is never translated, so you see the case as you key it.

On the status line of the EDTDBF Display, there is a two character indicator for the current Upper (US) or Lower (LS) case in position 64 thru 65.



```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

  Control: __________ Scan: _____________________________    1   151  W: 000
  Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

  0000001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
  0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
  0000003 ---MR   MILES     A T                S-oa?ACCTS     ---'LONG GREEN'
  0000004 ---MS   OWENS     I U                A—E-EDP       ---15 ARVONSON P
  0000005 --?MS   BROWN     I M                A-/--EDP       ---9978 RISEBANT
```

*After F2 is pressed status changes from US to LS:



```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 LS Posn: 000000

  Control: __________ Scan: _____________________________    1   151  W: 000
  Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

  0000001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
  0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
  0000003 ---MR   MILES     A T                S-oa?ACCTS     ---'LONG GREEN'
  0000004 ---MS   OWENS     I U                A—E-EDP       ---15 ARVONSON P
  0000005 --?MS   BROWN     I M                A-/--EDP       ---9978 RISEBANT
```

F3=Exit Display

Pressing F3 or typing EXIT in the control field will exit (with update) from the display.

F4=Prompt Key Fields for Position by Key Operation

Pressing F4 or typing ‘K’ in the control field will display a subfile of ALL key fields and allow data to be keyed into these fields. When ENTER is pressed then the data in these key fields are used by the SETLL operation to re-position the file.

This function is only valid when the file is in SETLL mode.

F5=Reset/Cancel Pending Operations

Pressing F5 will cancel any pending operations and refresh the current display.

F6=Add a Record

Pressing F6 or typing ‘ADD’ in the control field will display a subfile of ALL fields and allow data to be entered into these fields.

This function is only valid when the file is in EDIT mode.

F7=Switch Hexadecimal Mode On/Off

Pressing F7 or typing ‘HEX’ in the control field switches back and forth from Character display to Hexadecimal display.

Hexadecimal display format 1 looks like this:

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

Control: __________ Scan: _____________________________    __1   _151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 ---MR   SMYTHE    J S                 A-&/-EDP       ---15 HAPPY STRE
0000001 001DD444EDEECC4444D4E44444444444444444C0565CCD4444444015FF4CCDDE4EEDC
0000001 00F490002483850000102000000000000000001801F547000000002F1508177802395
0000001
0000002 ---MS   HAMLINSON J T                 S-o—ACCOUNTS  ---12/125 LEADBE
0000002 002DE444CCDDCDEDD4D4E44444444444444444E0732CCCDEDEE44012FF6FFF4DCCCCC
0000002 00F42000814395265010300000000000000000002801F133645320000F1211250351425
0000002
0000003 ---MR   MILES     A T                 S-oa?ACCTS     ---‘LONG GREEN’
0000003 003DD444DCDCE44444C4E44444444444444444E0746CCCEE444440127DDDC4CDCCD74
0000003 00F490004935200000103000000000000000002802F133320000000FD3657079555D0
0000003
0000004 ---MS   OWENS     I U                 A—E-EDP       ---15 ARVONSON P
0000004 004DE444DECDE44444C4E44444444444444444C0978CCD4444444010FF4CDEDDEDD4D
0000004 00F42000665520000090400000000000000000001802F547000000005F1501956526507
0000004
0000005 --?MS   BROWN     I M                 A-/--EDP       ---9978 RISEBANT
0000005 006DE444CDDED44444C4D44444444444444444C0618CCD4444444012FFFF4DCECCCDE
0000005 00F42000296650000090400000000000000000001812F547000000005F9978099252153
0000005
```

F8=Switch Hexadecimal Display Format

Pressing F8 switches back and forth between Hexadecimal Display format 1 and Hexadecimal Display format 2. Format 2 looks like this:

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

Control: __________ Scan: _____________________________    __1   _151  W: 000
Edit    --------+---------10--------+---------20--------+---------30--------+

0000001  - - - M R       S M Y T H E         J   S
0000001 00001FD4D9404040E2D4E8E3C8C540404040D140E2404040404040404040404040404
0000002  - - - M S       H A M L I N S O N   J   T
0000002 00002FD4E2404040C8C1D4D3C9D5E2D6D540D140E3404040404040404040404040404
0000003  - - - M R       M I L E S           A   T
0000003 00003FD4D9404040D4C9D3C5E24040404040C140E3404040404040404040404040404
0000004  - - - M S       O W E N S           I   U
0000004 00004FD4E2404040D6E6C5D5E24040404040C940E4404040404040404040404040404
0000005  - - ? M S       B R O W N           I   M
0000005 00006FD4E2404040C2D9D6E6D54040404040C940D4404040404040404040404040404
```

F9=Switch Update Mode On/Off

Pressing F9 switches the Update mode On or Off. If you are not authorised to update the displayed data base file, then this command key has no effect. F9 will only work for the EDTDBF command. It is not available from the DSPDBF command.

F13=Generate Record Format Information

Pressing F13 generates Record Format Information to allow extended functions to be performed, such as displaying, updating or printing data in unpacked format.

To ensure that Record Format Information is current then Function Key 13 should always be used to regenerate field attribute information before using any of the extended display, update or print functions.

F15=Position File by Full or Partial Char/Hex Key

Pressing F15 will re-position the data base file using the SETLL operation (the Scan data is used as the key). The key is entered as Scan data as either character or hexadecimal format.

This function is only valid when the file is in SETLL mode.

The key can be entered in either Character or Hex format. For Character format, ENTER the key field(s) as Scan data and press F15:

```
NUTIL/WRKDBFP WRKDBFP KF *SETLL      Recl: 0126 Recs: 0000067 US Posn: 000000

Control: __________ Scan: Line_____________________________   1   151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 A         Prompt field names for Record Addition        *ADD
0000002 ADD       Prompt field names for Record Addition        *ADD
0000003 B         Go to bottom of file                          *RECORD    99
0000004 CAPS      Switch between Upper and Lower case           *CASE
0000005 CASE      Switch between Upper and Lower case           *CASE
0000006 CMD       Command Entry Line                            CALL QUSCMDLN
0000007 CMDK01    Display HELP Text                             DSPHLP    WRK
0000008 CMDK02    Switch between Upper and Lower case           *CASE
```

For Hex format, ENTER the hexadecimal key field as Scan data and press F15:

```
NUTIL/WRKDBFP WRKDBFP KF *SETLL      Recl: 0126 Recs: 0000067 US Posn: 000000

Control: __________ Scan: X'C3D4C4D2F0F1'__________________   1   151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 A         Prompt field names for Record Addition        *ADD
0000002 ADD       Prompt field names for Record Addition        *ADD
0000003 B         Go to bottom of file                          *RECORD    99
0000004 CAPS      Switch between Upper and Lower case           *CASE
0000005 CASE      Switch between Upper and Lower case           *CASE
0000006 CMD       Command Entry Line                            CALL QUSCMDLN
0000007 CMDK01    Display HELP Text                             DSPHLP    WRK
0000008 CMDK02    Switch between Upper and Lower case           *CASE
```

F16=Scan Forwards using Char/Hex mode

Pressing F16 will scan forwards for a nominated character or hexadecimal string. Scanning stops at the first occurrence of the scan string found for each record and the cursor is positioned at the beginning of the located character string.

For a character format scan, enter the required scan string and press F16:

```
NUTIL/WRKDBFP WRKDBFP KF *SETLL      Recl: 0126 Recs: 0000067 US Posn: 000000

Control: __________ Scan: Text_____________________________   1   151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 A         Prompt field names for Record Addition        *ADD
0000002 ADD       Prompt field names for Record Addition        *ADD
0000003 B         Go to bottom of file                          *RECORD    99
0000004 CAPS      Switch between Upper and Lower case           *CASE
0000005 CASE      Switch between Upper and Lower case           *CASE
0000006 CMD       Command Entry Line                            CALL QUSCMDLN
0000007 CMDK01    Display HELP Text                             DSPHLP    WRK
0000008 CMDK02    Switch between Upper and Lower case           *CASE
```

For a Hex format scan, enter the required hexadecimal scan string and press F16:

```
NUTIL/WRKDBFP WRKDBFP KF *SETLL      Recl: 0126 Recs: 0000067 US Posn: 000000

Control: __________ Scan: X'C8C5D3D7'______________________   1   151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 A         Prompt field names for Record Addition        *ADD
0000002 ADD       Prompt field names for Record Addition        *ADD
0000003 B         Go to bottom of file                          *RECORD    99
0000004 CAPS      Switch between Upper and Lower case           *CASE
0000005 CASE      Switch between Upper and Lower case           *CASE
0000006 CMD       Command Entry Line                            CALL QUSCMDLN
0000007 CMDK01    Display HELP Text                             DSPHLP    WRK
0000008 CMDK02    Switch between Upper and Lower case           *CASE
```

F17=Scan Backwards using Char/Hex mode

Pressing F17 will scan backwards for a nominated character or hexadecimal string. Scanning stops at the first occurence of the scan string found for each record and the cursor is positioned at the beginning of the located character string.

For a character format scan, enter the required scan string and press F17:

```
NUTIL/WRKDBFP WRKDBFP KF *SETLL      Recl: 0126 Recs: 0000067 US Posn: 000000

Control: __________ Scan: Text_____________________________   1   151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 A         Prompt field names for Record Addition        *ADD
0000002 ADD       Prompt field names for Record Addition        *ADD
0000003 B         Go to bottom of file                          *RECORD    99
0000004 CAPS      Switch between Upper and Lower case           *CASE
0000005 CASE      Switch between Upper and Lower case           *CASE
0000006 CMD       Command Entry Line                            CALL QUSCMDLN
0000007 CMDK01    Display HELP Text                             DSPHLP    WRK
0000008 CMDK02    Switch between Upper and Lower case           *CASE
```

For a Hex format scan, enter the required hexadecimal scan string and press F17:

```
NUTIL/WRKDBFP WRKDBFP KF *SETLL      Recl: 0126 Recs: 0000067 US Posn: 000000

Control: __________ Scan: X'C8C5D3D7'______________________   1   151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 A         Prompt field names for Record Addition        *ADD
0000002 ADD       Prompt field names for Record Addition        *ADD
0000003 B         Go to bottom of file                          *RECORD    99
0000004 CAPS      Switch between Upper and Lower case           *CASE
0000005 CASE      Switch between Upper and Lower case           *CASE
0000006 CMD       Command Entry Line                            CALL QUSCMDLN
0000007 CMDK01    Display HELP Text                             DSPHLP    WRK
0000008 CMDK02    Switch between Upper and Lower case           *CASE
```

F19=Window Left 70 Characters

Pressing F19 will window left 70 characters (which is a full screen).

F20=Window Right 70 Characters

Pressing F20 will window right 70 characters (which is a full screen).

F21=Display Command Line Window

Pressing F21 or typing ‘CMD’ in the control field will cause a command line window to appear at the bottom of the screen.

F22=Switch Display Mode to Arrival Sequence

Pressing F22 or typing ‘RRN’ in the control field switches the display mode to arrival sequence.

Logical files which select or omit records can not be accessed by arrival sequence, therefore they are displayed in keyed sequence.

```
NUTIL/WRKDBFP WRKDBFP KF *RRN        Recl: 0126 Recs: 0000067 US Posn: 000000

Control: __________ Scan: _____________________________   __1   _151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 A         Prompt field names for Record Addition        *ADD
0000002 ADD       Prompt field names for Record Addition        *ADD
0000003 B         Go to bottom of file                          *RECORD    99
```

F23=Switch Display Mode to Keyed Sequence

Pressing F23 or typing ‘KEY’ in the control field will switch the display mode to Keyed sequence. If the data base file being displayed is a non-keyed physical file, then the display sequence will be Arrival.

```
NUTIL/WRKDBFP WRKDBFP KF *KEY        Recl: 0126 Recs: 0000067 US Posn: 000000

Control: __________ Scan: _____________________________   __1   _151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 A         Prompt field names for Record Addition        *ADD
0000002 ADD       Prompt field names for Record Addition        *ADD
0000003 B         Go to bottom of file                          *RECORD    99
```

F24=Switch Display Mode to Position by Key

Pressing F24 switches the display mode to Position by Key. For more information on how to position the file, refer to F4 and F15 Help.

If the data base file being displayed is a non-keyed physical file then it will be displayed in Arrival sequence.

```
NUTIL/WRKDBFP WRKDBFP KF *SETLL      Recl: 0126 Recs: 0000067 US Posn: 000000

Control: __________ Scan: _____________________________   __1   _151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 A         Prompt field names for Record Addition        *ADD
0000002 ADD       Prompt field names for Record Addition        *ADD
0000003 B         Go to bottom of file                          *RECORD    99
```

PAGEDOWN Key

Depending on the current display format being used the next 4, 9 or 19 records will be displayed.

PAGEUP Key

Depending on the current display format being used the previous 4, 9 or 19 records will be displayed.

PRINT Key

Pressing the PRINT key causes the current display to be printed to the associated work station printer or, if there is no associated work station printer, to the default printer device.

Special Control Commands

POSITION (Record Positioning)

A change in record positioning on the Browse/Update Display can be requested as follows:

Keying in a new record number in the Control field to position that record at the top of the display.

You can also key in '+12' into the Control field to page down 12 records, or a '-5' in the Control field to page up 5 records.

To position the file at the 7th record, enter a '7' in the control field:

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

 Control: 7          Scan: _____________________________     1   151  W: 000
 Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

 0000001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
 0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
```

To position at the last record on file, enter 'B' (Bottom) or 'E' (End of File) as the control command.

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

Control: B          Scan: _____________________________     1   151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
```

WINDOWING (How to perform Windowing)

The data that is displayed on the Browse/Update Display can be thought of as a 70-character-wide and 20-record-long window into your data file member. The window can be varied horizontally by keying in a window line number (W+n, W-n, Wn).

For example, to start your window at position 35 of each data record, type ‘W35’ in the Control field on the display. The current window will be displayed following the W: on the Browse/Update heading line.

For example: To Window across to position 35 of the record.

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

Control: W35_____ Scan: _____________________________   1   151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
```

WINDOWING (How to perform Windowing by Field Name)

The data that is displayed on the Browse/Update Display can be thought of as a 70-character-wide and 20-record-long window into your data file member. The window can be varied horizontally by keying in a field name in the Scan data field.

For example, to start your window at a certain field name, type ‘W’ in the Control field and the field name required as Scan data.

The current window will be displayed following the W: on the Browse/Update heading line.

For example :

To Window across to field ADDR1 of the record.

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

Control: W          Scan: ADDR1                              1   151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
```

If the field name is not known, type ‘W’ in the control field with no field name in the Scan field. The screen will display a list of the record format field from which to select, as follows:

```
Object  : DEMOFILE       Work with Data Base File          Rcdfmt : EMPLDTA
Library : NUTIL                                            Rcdlen :   151

    Field          Pos  Attributes      Text Description

  _ EMPNO       K    1     5  P  0     Employee Number
  _ TITLE            4     5  A        Employee Title
  _ NAME             9    30  A        Employee Name
  _ STATUS          39     1  A        Status
  _ STRDT           40     6  P  0     Commencement Date
  _ DEPT            44    10  A        Department
  _ CLASS           54     5  P  0     Classification
  _ ADDR1           57    30  A        Employee Address 1
  _ ADDR2           87    30  A        Employee Address 1
  _ STATE          117    10  A        State
  _ PCODE          127    10  A        Postal Code
  _ TELE           137    15  A        Telephone Number
```

Type a “1” beside the field on which to position the window and press ENTER.

Extended Line Commands

DELETE (Delete a record in the file)

To delete a record, enter a ‘D’ in the first position of the RECNBR field.

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

 Control: __________ Scan: ____________________________    1   151  W: 000
 Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

 D 00001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
 0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
```

If the data base file is being displayed and update mode has not been enabled, then any ‘D’ entered will subsequently be ignored.

DISPLAY (Display a record, formatted by field name)

To display a specific record formatted by field name, simply enter a 'X' in the first position of the RECNBR field. This will cause a subfile of ALL fields and their contents to be displayed.

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

Control: __________ Scan: ____________________________    _1_  _151_  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

X 00001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
```

If the editor is in EDIT mode, the fields will be displayed and allow updating of the information. If the editor is in BROWSE mode the information will be displayed, but not update-capable.

```
Object  : DEMOFILE       Work with Data Base File          Rcdfmt : EMPLDTA
Library : NUTIL                  Edit                      Rcdlen :   151
Member  : DEMOFILE                                         Case   : Upper

  Field        Attributes   ----+---10----+---20----+---30 Text Description
  EMPNO      K      5  P  0  00001                         Employee Number
  TITLE             5  A     MR                            Employee Title
  NAME             30  A     SMYTHE    J S                 Employee Name
  STATUS            1  A     A                             Status
  STRDT             6  P  0  850615                        Commencement Date
  DEPT             10  A     EDP                           Department
  CLASS             5  P  0  00125                         Classification
  ADDR1            30  A     15 HAPPY STREET               Employee Address 1
  ADDR2            30  A     FUNTOWN                       Employee Address 1
  STATE            10  A     MISS                          State
  PCODE            10  A     2865                          Postal Code
  TELE             15  A     052 2045                      Telephone Number


  F3=Exit   F11=Delete   F12=Upper/Lower Case   F24=Print Rcd
```

UPDATE (Update a record in the file)

Refer to the previous topic, DISPLAY. Update works in exactly the same way as Display, but Update is available only from the EDTDBF command.

COPY (Copy a record in the file)

To copy a record by field name simply enter a 'C' in the first position of the RECNBR field. This will cause a subfile of all fields in the selected record (and the field contents) to be displayed and allow them to be updated and a new record will be written to the data base.

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

 Control: __________ Scan: _____________________________   __1  __151  W: 000
 Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

 C 00001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
 0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
```

```
Object  : DEMOFILE       Work with Data Base File          Rcdfmt : EMPLDTA
Library : NUTIL                  Copy                      Rcdlen :   151
Member  : DEMOFILE                                         Case   : Upper

  Field        Attributes   ----+---10----+---20----+---30 Text Description
  EMPNO      K      5  P  0  00001                         Employee Number
  TITLE             5  A     MR                            Employee Title
  NAME             30  A     SMYTHE    J S                 Employee Name
  STATUS            1  A     A                             Status
  STRDT             6  P  0  850615                        Commencement Date
  DEPT             10  A     EDP                           Department
  CLASS             5  P  0  00125                         Classification
  ADDR1            30  A     15 HAPPY STREET               Employee Address 1
  ADDR2            30  A     FUNTOWN                       Employee Address 1
  STATE            10  A     MISS                          State
  PCODE            10  A     2865                          Postal Code
  TELE             15  A     052 2045                      Telephone Number


  F3=Exit   F11=Delete   F12=Upper/Lower Case   F24=Print Rcd
```

To avoid duplicate keys, you should ensure that you update the information in at least one of the displayed key fields.

PRINT (Print a record in the file)

To print a record by field name simply enter a ‘P’ in the first position of the RECNBR field. This will cause ALL fields and their contents to be printed.

```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

Control: __________ Scan: _____________________________    1   151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

P 00001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS  ---12/125 LEADBE
```

Other control commands

DELETE (delete records from the file)

See *'Processing via the NRUNSQL command'* later in this manual.

DSPDBR (Display Data Base Relations)

The Display Data Base Relations (DSPDBR) command provides relational information about data base files. The information identifies the physical files that are dependent on a specific file, those files that use a specific record format, or the members that are dependent on a specific member. This command can be used to actually display or print the information.

Restrictions: Before you can display each file specified, you must have operational authority for, or own, the file.

DSPFD (Display File Description)

The Display File Description (DSPFD) command displays one or more types of information that is retrieved from the file descriptions of one or more database and/or device files. The information is displayed for each file that has the specified name that is found in all libraries named in the specified library list to which the user has access. The information can be displayed, printed, or directed to a data base file.

Restrictions: Before you can display each file specified, you must have operational authority for or own the file. Also, of the libraries specified, by the library qualifier, only the libraries for which you have read rights are searched for files. If TYPE(*ALL), TYPE(*MBR), or TYPE(*MBRLIST) is specified and the file is a physical file, one data authorisation is required for member information.

DSPFFD (Display File Field Description)

The Display File Field Description (DSPFFD) command provides field information for the data base file being displayed. This command can be used to actually display or print the information.

Restrictions: Before you can display each file specified, you must have operational authority for, or own, the file.

DSPMSG (Display Messages)

The Display Message (DSPMSG) command is used by the work station user to display the messages received at the specified message queue. If the message queue is not allocated to the job in which this command is entered, it is implicitly allocated by this command for the duration of the command.

When the messages are displayed, options are also displayed that allow the user to remove one or more messages from the queue or enter a reply to each inquiry message.

DSPOBJD (Display Object Description)

The Display Object Description (DSPOBJD) command displays the names and attributes of the data base file being displayed. This command can be used to actually display or print the information.

Restrictions: Before you can display each file specified, you must have operational authority for, or own, the file.

DUPKEYS (Detecting duplicate keys in the file)

This will prompt the NRUNSQL command with a preformatted SQL SELECT statement designed to detect duplicate keys. Replace the text **key_field*n*** with the names of all of the key fields in the file. For a basic description of SQL statements see *'Processing via the NRUNSQL command'* later in this manual.

E (Position File at Last Record) or B (Goto Bottom of File)

The End of File command is used to position the file at the last record.

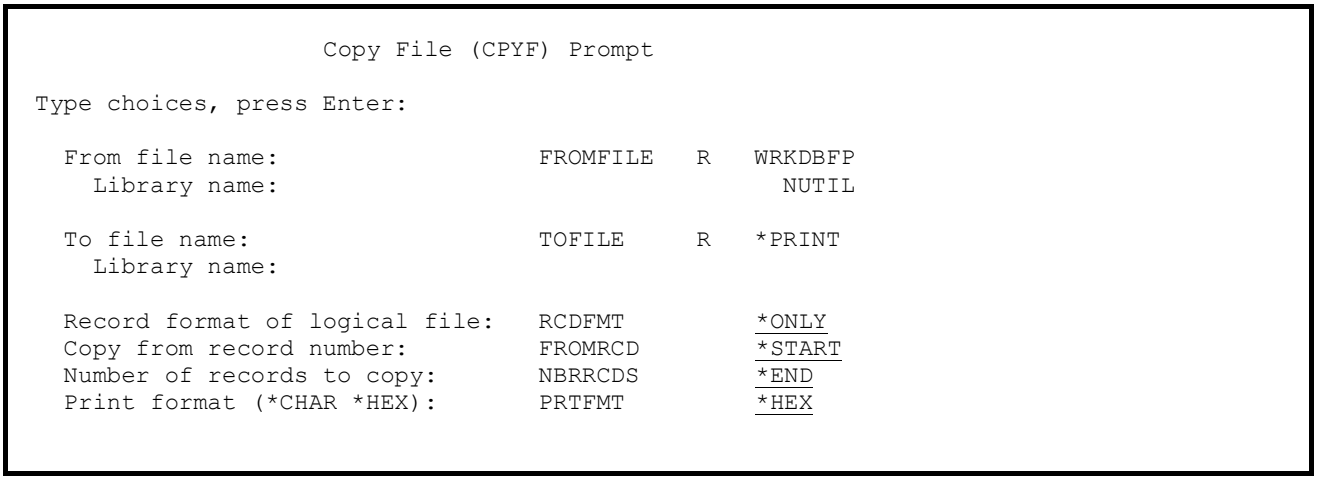
```
NUTIL/DEMOFILE DEMOFILE KF *KEY      Recl: 0151 Recs: 0000005 US Posn: 000000

Control: E________ Scan: _____________________________    __1   _151  W: 000
Edit    ----+---10----+---20----+---30----+---40----+---50----+---60----+---7

0000001 ---MR   SMYTHE    J S                A-&/-EDP       ---15 HAPPY STRE
0000002 ---MS   HAMLINSON J T                S-o—ACCOUNTS   ---12/125 LEADBE
```

INSERT (Insert records into the file)

See *'Processing via the NRUNSQL command'* later in this manual.

PRINT (Print File by Record Range using CPYF)

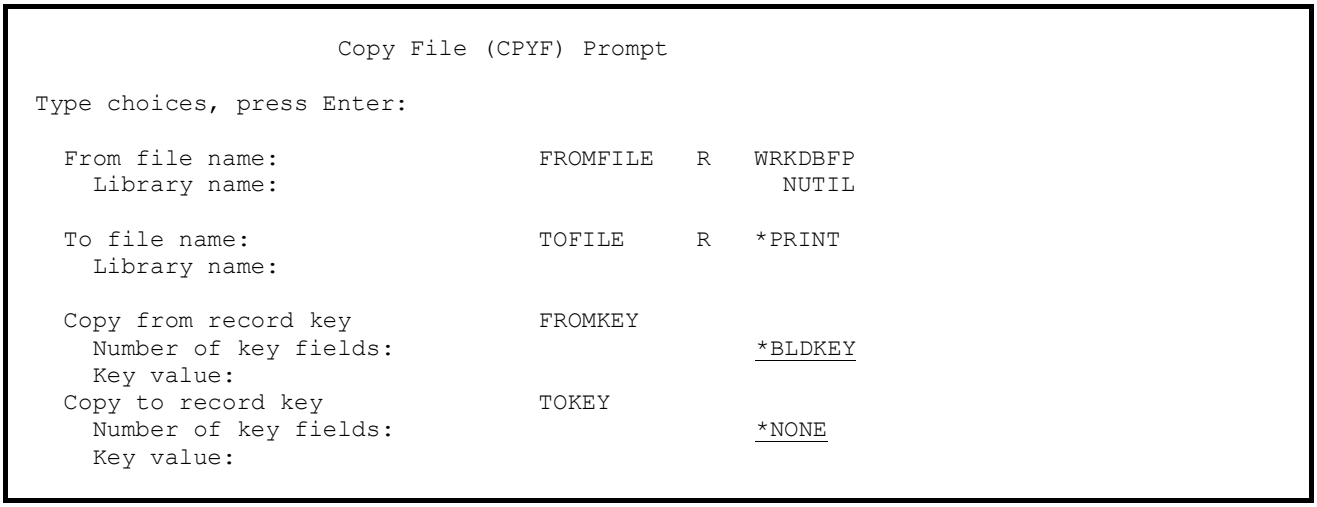
The Print (PRINT) command allows the displayed file to be printed using the CPYF command. It may be printed from beginning to end, or by a selected record range in character or hexadecimal mode.



```
                    Copy File (CPYF) Prompt

Type choices, press Enter:

  From file name:                  FROMFILE   R   WRKDBFP
    Library name:                                   NUTIL

  To file name:                    TOFILE     R   *PRINT
    Library name:

  Record format of logical file:   RCDFMT         *ONLY
  Copy from record number:         FROMRCD        *START
  Number of records to copy:       NBRRCDS        *END
  Print format (*CHAR *HEX):       PRTFMT         *HEX
```

PRINTK (Print File by Key Range using CPYF)

The Print (PRINTK) command allows the displayed file to be printed using the CPYF command. It may be printed from beginning to end, or by a selected key range in character or hexadecimal mode.



```
                    Copy File (CPYF) Prompt

Type choices, press Enter:

  From file name:                  FROMFILE   R   WRKDBFP
    Library name:                                   NUTIL

  To file name:                    TOFILE     R   *PRINT
    Library name:

  Copy from record key             FROMKEY
    Number of key fields:                        *BLDKEY
    Key value:
  Copy to record key               TOKEY
    Number of key fields:                        *NONE
    Key value:
```

SELECT (Select records in the file)

See *'Processing via the NRUNSQL command'* later in this manual.

UPDATE (Update records in the file)

See *'Processing via the NRUNSQL command'* later in this manual.

UPDDTA (Update file using DFU temporary program)

The UPDDTA control command will cause the OS/400 command UPDDTA to be called. This command generates a temporary DFU (Data File Utility) program allowing you to update the file.

WRKACTJOB (Work with Active Jobs)

The Work with Active Jobs (WRKACTJOB) command displays performance and status information for the active jobs in the system.

WRKJOB (Display Job)

The WRKJOB command displays, for the specified user job, any of the following information:

. Job status attributes
. Job definition attributes
. Job execution attributes
. Program invocation stack information
. Spooled file information
. Job lock information
. Commitment control status
. Library list information
. Open file information
. File override information

WRKJOBQ (Work with Job Queue)

The Work with Job Queue command displays the overall status of all job queues or the detailed status of a specific job queue. The status of the queues may change while the command is being executed.

Restrictions: If only one job queue is to be displayed, the user must have read rights for the queue to be displayed or he must have job control rights in his user profile and the job queue must have OPRCTL(*YES) attribute. If all the job queues are to be displayed, the user needs authority only for the WRKJOBQ command.

WRKSBMJOB (Display Submitted Jobs)

The WRKSBMJOB command displays the status of all jobs submitted by the user.

Note that Jobs submitted with the WRKSBMJOB(*NO) parameter specified on the SBMJOB, SBMCRDJOB, SBMDBJOB, or SBMDKTJOB commands are not displayed by this command.

WRKSPLF (Work with Spooled Files)

The WRKSPLF command displays a list of all spooled files generated by the user.

WRKOUTQ (Work with Output Queue)

The Work with Output Queue command displays the overall status of all output queues or the detailed status of a specific output queue. The status of the queues may change while the command is being executed.

Restrictions: If only one output queue is to be displayed, the user must have read rights for the queue to be displayed or he must have job control rights in his user profile and the output queue must have OPRCTL(*YES) attribute. If all the output queues are to be displayed, the user needs authority only for the WRKOUTQ command.

Processing via the NRUNSQL command

The NRUNSQL allows you to run an interactive SQL statement, without SQL. This can be handy for 'batch' style processing of a database file.

The following provides a brief review of the more common SQL statements, which can be used with the NRUNSQL command.

The INSERT statement

INSERT can be used to add a new record to a file.

```
INSERT INTO file_name (field1, field2, field3)
VALUES (value1, value2, value3)
```

The INTO clause names the fields that are to be loaded. The VALUES clause specifies a value to insert into the related field.

General convention requires that you specify each field in the record format and a value to be assigned to each field.

The DELETE statement

DELETE can be used to delete records from a file.

DELETE FROM file_name WHERE field1 = value1

The WHERE clause specifies the conditions under which a record will be deleted from the file. If you do not supply a WHERE clause, all records in the file will be deleted.

```
DELETE from EMPMAST   where STATUS = 'D'
DELETE from PARTSFILE where USECOUNT < 5
```

The UPDATE statement

UPDATE can be used to update records in a file.

```
UPDATE file_name SET field1 = value1, field2 = value2
WHERE field3 = value3
```

The SET clause specifies the fields you want to update and the values you wish to assigned to the updated fields. The value you assign can be

.another field name, so you can replace the contents of field1 with the contents of field2

```
SET field8 = field9
```

...a constant

```
SET field8 = 'A'
```

...a null value, using the keyword NULL (the field must be defined as NULL capable)

```
SET field8 = NULL
```

...an expression, where the results of the expression will be loaded into the field

```
SET field8 = field9 + field10
```

The WHERE clause specifies the conditions under which a record will be updated in the file. If you do not supply a WHERE clause, all records in the file will be updated.

```
UPDATE EMPMAST  set SALARY = 0  where STATUS = 'D'
```

The SELECT statement

SELECT can be used to view records in a file.

```
SELECT field1, field2 FROM file_name WHERE field3 = value3
SELECT * FROM file_name WHERE field1 = value1
```

The first form of the statement will retrieve only the fields specified; the second form (SELECT *) will retrieve all fields.

The WHERE clause specifies the conditions under which a record will be retrieved from the file. If you do not supply a WHERE clause, all records in the file will be retrieved.

```
SELECT * from EMPMAST   where SALARY > 25000
```

The OUTPUT(*PRINT) parameter for the NRUNSQL command is only valid if the SQL statement parameter specified includes a SELECT statement.

General rules within SQL statement definition

Basic predicates compare two values; valid ones are as follows:

Predicate	Meaning
field1 = field2	field1 is equal to field2
field1 ¬= field2	field1 is not equal to field2
field1 <> field2	field1 is not equal to field2
field1 < field2	field1 is less than field2
field1 > field2	field1 is greater than field2
field1 >= field2	field1 is greater than or equal to field2
field1 <= field2	field1 is less than or equal to field2
field1 ¬< field2	field1 is not less than field2
field1 ¬> field2	field1 is not greater than field2

EDTDBF/DSPDBF Function Key chart

The following function key assignments are the default values as supplied by Navan. These may be different at your installation.

Function Key	*Description*
1	Display HELP Text
2	Switch between Upper and Lower case
3	Exit Display
4	Prompt field names for SETLL
5	Refresh/Cancel Pending Operations
6	Add new record
7	Switch Hex On/Off
8	Switch Hex Display Format
9	Switch mode - browse/edit
12	Previous
13	Generate file field information
15	Position file using full/partial Char/Hex Key
16	Scan Forwards using Char/Hex data
17	Scan Backwards using Char/Hex data
19	Window left (-70 bytes)
20	Window right (+70 bytes)
21	Command Entry Line
22	Switch sequence to Arrival *RRN
23	Switch sequence to Keyed *KEY
24	Switch sequence to Keyed/SETLL *SETLL

EDTDBF/DSPDBF Control Keywords

The following control keyword assignments are the default values as supplied by Navan. These may be different at your installation.

Control Keyword	*Description*
ADD	Prompt field names for Record Addition
B (or E)	Go to bottom of file
CAPS	Switch between Upper and Lower case
CASE	Switch between Upper and Lower case
CMD	Command Entry Line
DELETE	Delete records via the NRUNSQL command
DSPDBR	Display Data Base Relations
DSPFD	Display File Description
DSPFFD	Display File Field Description
DSPFILE	Print Data Description Layout
DSPMSG	Display Messages
DUPKEYS	Detect duplicate keys in a file
EXIT	Exit Display
EXPIRY	Display Expiry date
GEN	Generate DSPFFD Information
HELP	Display HELP Text
HEX	Switch Hex On/Off
HEXFMT	Switch Hex Display Format
INSERT	Insert records via the NRUNSQL command
K	Prompt field names for SETLL
KEY	Switch display sequence to keyed
PAGEDOWN	Page Down
PAGEUP	Page Up
PRINT	Print File using CPYF by RRN range
PRINTK	Print File using CPYF by KEY range
PROMPT	Prompt field names for SETLL
RESET	Refresh/Cancel Pending Operations
ROLLDOWN	Roll Down
ROLLUP	Roll Up
RRN	Switch display sequence to Arrival
SCANB	Scan Backwards using Char/Hex data
SCANF	Scan Forwards using Char/Hex data
SELECT	Select (for view) records via the NRUNSQL command
SETLL	Switch display sequence to Keyed with SETLL
T (or 1)	Go to top of file
UPDATE	Update records via the NRUNSQL command
UPDDTA	Update data using DFU (IBM Data File Utility)
W	Window by Field Name
WRKACTJOB	Work with Active Jobs
WRKJOB	Work with Job
WRKJOBQ	Work with Job Queue
WRKOBJD	Work with Object Description
WRKOUTQ	Work with Output Queues
WRKSBMJOB	Work with submitted jobs
WRKSPLF	Work with spool files

Job Logs Management Facility

An Overview

Normal OS/400 operation of job log processing requires retaining hundreds of logs on an output queue. Apart from resulting in a slow response when displaying the output queue, this method also increases the number of jobs in the system, and eventually affects system performance.

The aim of the NUTIL Job Logs Management Facility is to allow the retention of joblogs for specified number of days without holding the logs on an output queue. The logs are loaded into a database file and cross referenced for future access. The job log is then removed from the output queue, effectively removing the job from the system.

The system operates by maintaining two joblog files: an index of the logs, and a data file which contains the text information of each job log.

The files are loaded upon request, via a command supplied. It is usually more convenient if this job is run as part of a routine, perhaps during a housekeeping routine, or as part of a regular start-up routine. A standard job (LODJLOG) is supplied in the Job Scheduler to perform this function.

Once the job log information is in the JOBLOGS files, it can be accessed via the supplied NWRKJOBLOG command.

Routines are also provided to define how long the stored joblogs are to be retained in the JOBLOGS database. If a retention period is defined the logs will be automatically removed when they have exceeded the retention period.

JOBLOGS in operation

JLINSTALL - Installing the JOBLOGS facility

The Job Logs Management Facility will not operate unless the installation routine has been performed. Refer to the NUTIL Installation Guide for information regarding the JLINSTALL command.

LODJLOG - Load active Joblogs to History

This is the main maintenance routine for the JOBLOGS facility. Run on a regular basis, it loads job logs into the JOBLOGS database and removes expired joblogs on the specified frequency.

```
            Load active Joblogs to History (LODJLOG)

Type choices, press Enter:

  Severity filter                  SEV            0
  Maximum pages filter             MAXPAGES       *NOMAX
  Ignore *SYS System logs          IGNSYSLOGS     *NO
  Output                           OUTPUT         *PRINT
```

The joblog maintenance routine performs a number of basic functions:

- Loads all joblogs for jobs (with a completion code greater than, or equal to, the severity filter specified on the command prompt) on the specified joblog output queue into the JOBLOGS Database and creates a controlling index record for each in the JOBLOGS Index. The MAXPAGES filter is an optional one, intended to allow you to filter out joblogs that are excessively large. The IGNSYSLOGS filter allows you to bypass processing of system job logs. If requested, an audit report is also printed which lists all of the joblogs that have been processed.
- Clears the joblog output queue
- On the specified Reorganisation frequency, removes expired joblogs from JOBLOGS

Because of the nature of the routine, it should be submitted for batch processing (or using SCHEDULER).

No special library list is needed for the submission job description as LODJLOG looks after itself, adding any libraries it needs.

NWRKJOBLOG - Work with archived Job Logs

The NWRKJOBLOG command allows you to access the joblog information stored in the JOBLOGS database. Stored joblogs can be viewed, printed and removed using the options provided by the work panel in this command.

```
                Work with archived Job Logs (NWRKJOBLOG) Prompt

Type choices, press Enter:

  Extract Date:                    STRDATE    P   *TODAY
  Allow delete operation:          ALWDLT     P   *YES
```

Use of the NWRKJOBLOG command requires the user to have *SPLCTL special authority.

Parameter definitions

The STRDATE (Start Date for extract) parameter allows you to start listing joblogs from a specified date. If you specify *START, the display list will start from the first joblog stored in the joblogs database. If you specify *TODAY, the display list will start at today's date.

Alternatively, you may enter a specific date for the display list to start at. You must enter the required date in job date format.

The ALWDLT (Allow joblog deletion) parameter specifies whether the user will be allowed to access the option for deleting a stored joblog.

NWRKJOBLOG - Joblog display list for selection

```
JLM210D1                  Work with archived Job Logs
Select

Type options, press Enter.
  4=Delete   5=Display   6=Print   7=PDF   8=Display Index entry

                                                        Total   Termination
  Opt   Date      Time      User        Job       Number  Pages    Severity

                                                      ..SUBSET fields
      17/10/21  5:51:58 QSYS        SCPF        018104  ..POSITIONING fields

    _ 17/10/21  5:51:58 QSYS        SCPF        018104       2
    _ 17/10/21  5:52:09 QSYS        QSYSARB     018002       4
    _ 17/10/21  6:42:47 QSYSOPR     AA_SAVE     018137       2        30
    _ 17/10/21  6:43:25 QSYSOPR     OPR_MSGQHD  018138       1
    _ 17/10/21  7:02:42 QSYSOPR     RGZ_NUTIL   018140       2
    _ 17/10/21  7:02:42 QSYSOPR     IEW_INTFAC  018141       1
    _ 17/10/21  7:47:56 QSYSOPR     LODJLOG     018142       2
    _ 17/10/21 18:01:05 QSYSOPR     OPR_MSGQDF  018148       1
    _ 17/10/21 18:01:20 QSYSOPR     RLS_NIGHTQ  018149       1

  F3=Exit   F9=Request data   F15=Current logs   F17=Top   F18=Bottom
```

Based on the start date entered, the list of archived joblogs will be displayed. Use the Pagedown and Pageup keys to scroll through the list. The list displays

Date the joblog was extracted (by the LODJLOG command)
Job identity (User ID, Job Name, Job Number)
The number of pages in the joblog
The termination severity of the job
The initial job request (use F9 to display it)

The first set of data entry fields (under the User, Job and Number columns) are SUBSET fields.

If you wish to display only the entries for a specific user, job name or job number, enter the known information in the SUBSET fields at the top of the screen. Generic entries are allowed in the SUBSET fields.

The second set of data entry fields (under the Date, Time, User, Job and Number columns) are POSITIONING fields.

If you wish to point the list to start at a specific date, user ID, job name or job number, enter the known information in the POSITIONING fields at the top of the screen and press enter.

Opt: This field (the option entry column) allows you to enter an operation to be performed on a specific joblog entry. Multiple option entries may be made. For each joblog entry displayed you may enter one of the following options:

4=Delete this joblog. This option will only be available if the ALWDLT parameter of the NWRKJOBLOG command was set to *YES.
5=Display the joblog detail
6=Print the joblog detail
7=Print the joblog detail to a .pdf document (see below)
8=Display the index entry for the joblog.

Function key 3 will exit the NWRKJOBLOG command.

Function key 9 will show the jobs initial command request, as well as the information shown above.
Function key 15 will display the current Joblogs output queue.

Function key 17 will reposition the list to the first entry in the JOBLOGS database.
Function key 18 will reposition the list to the last entry in the JOBLOGS database.

Function key 21 will display a pop-up command line window.

NWRKJOBLOG – convert joblog to .pdf document

If you use Option 7=PDF, the selected job log will be converted in to a .pdf document, ideal for using as an email attachment (it can be sent from your machine using the NUTIL/NSNDSMTPML command).

The finished .pdf document will be created in the IFS '/tmp' folder and will be named "JOBLOG_FOR_JOB_*job-number_user_job-name*.pdf"

NWRKJOBLOG - joblog data display

```
BROWSE FILE DISPLAY          FILE: NUTIL/JLPJLDT          MEMBER: JLPJLDT

  Control: W6____      Record:        1    Column:     6    Record size:   136
 ....1....+....2....+....3....+....4....+....5....+....6....+....7....+....8...

5769SS1 V5R4M0 000526                    JOB LOG

  JOB NAME - IEW_INTFAC        USER - QSYSOPR           NBR - 018141        JOB

TIME   MSGID  SEV TYPE  MESSAGE TEXT

090010 CPF1124 00 INFO  Job 018141/QSYSOPR/IEW_INTFAC started 17/10/21 07:00
                        QBATCH; entered system 17/10/21 09:00:12.
090012 CPI1125 00 INFO  Job 018141/QSYSOPR/IEW_INTFAC submitted to job queue
                         018090/QSYSOPR/WS41.
                   RQS  -CALL IE100
090015 CPF4028 10 DIAG  Open of mbr IEPTIEX file IEPTIEX changed to SEQONLY(*NO
090015 CPF4030 10 DIAG  Keyed sequence ignored for open of mbr IEPTIEX file IEP
090016 CPF4011 10 DIAG  Buffer length longer than rcd for file IEPFILE
090016 CPF4011 10 DIAG  Buffer length longer than rcd for file IEPACC mbr IEPAC
```

This screen will display as a result of entering option '5' beside any of the display list entries, giving you a display of the full joblog detail. The utility uses the NUTIL/DSPDBF command (or the IBM command DSPPFM if DSPDBF is not available) to provide this function.

RGZJLOG - Remove expired joblogs from the JOBLOGS database

Every time the LODJLOG command is processed the last thing it does is remove expired joblogs based on a defined frequency (refer to the explanation of the JLINSTALL command, in this manual).

Should it be necessary for you to perform a reorganise before the next one is automatically due, you may do so by running the RGZJLOG command. This command will check every joblog to see whether it has expired, based on today's date and the retention days specified on the JLINSTALL command.

For example, if you specified on your install

```
JLINSTALL  RETAIN(120) RGZFREQ(90)
```

it would mean that the automatic reorganise will occur every 90 days, removing any logs that are over 120 days old. You could request the processing of the RGZJLOG command at any time within the 90 days to perform an unscheduled reorganise.

It is important to note that the cleanse is processed against the "date extracted" (the date the joblog was loaded into the joblogs database) and not the job date.

Joblog Operational Considerations

Unattended processing considerations

As a precaution relating to file size, we have defined the Joblog data file so that it will normally be capable of loading joblogs without operator intervention. However there may be cases where an exceptionally large joblog may cause advisory messages to be sent to the system operator, to notify that the joblog data file has reached maximum size. At this stage a response is required from the system operator to either cancel the operation (the joblog is not loaded into the data file) or continue the process (the file size is increased). The job will wait until a response is received before continuing.

During regular daily operations, this would be quite normal. However, this joblog loading facility is usually processed after normal working hours, so you should plan for this situation to occur. There are three options available to you:

1. Change the file definition so that the restrictions on member size are removed

   CHGPF NUTIL/JLPJLDT SIZE(*NOMAX)

   ...this is a permanent change...

2. Before the system operator leaves each evening, change the operator message queue to default mode, which would mean that the oversized joblog would not be loaded (message CPA5305 has a default reply of 'C')

   CHGMSGQ QSYSOPR *DFT

3. Use the system reply list to respond to the message. First use the WRKRPYLE command to add a default response of 'I' (ignore and increase the member size to accommodate the joblog) for message ID CPA5305. Then, before the system operator leaves in the evening, change the operator message to default mode. This will then allow oversized joblogs to be loaded.

   ADDRPYLE seq_nbr MSGID(CPA5305) RPY('I')

   ...this is a permanent change...

   CHGMSGQ QSYSOPR *DFT

The Job Scheduler function of NUTIL has two standard jobs defined, to automatically control the altering of the system operator message queue. Each evening you should run the OPR_MSGQDF (Change the queue to *DFT mode) job and each morning run the OPR_MSGQHD (Change the queue to *HOLD mode) job. You should ensure the queue is not allocated to a job at those times.

Joblog retention considerations

Most installations will only want to keep joblogs for jobs that terminated abnormally. Others will want to retain all logs. The determination as to whether a joblog will be produced for a job (and what level of information should be recorded) is controlled by the LOG parameter on the Job Description that is used to submit a user job.

To retain logs only for those jobs terminating abnormally your job descriptions should be specified as follows:

```
CRTJOBD JOBD(FRED) ....  LOG(4 10 *NOLIST)
```

the *NOLIST entry on the LOG parameter signifies that a joblog will only be created if the submitted job terminates abnormally.

For each joblog retained within the JOBLOGS database, there is one record created in the index file (JLPINDX) and many records added to the data file (JLPJLDT).

For example, Job 015685/IEW_INTFAC/JONES_JS has one record defined in JLPINDX and the actual joblog text data is stored in file JLPJLDT, keyed by the job number (015685).

The amount of information loaded to JLPJLDT is variable, depending on the size of the job, the message severity filter and the logging level defined.

Operational considerations

If you are writing your own programs to access the joblogs data we recommend the procedure we use, which is to select a joblog from the index file JLPINDX and then, using a READ loop, to access the specific joblog data stored in JLPJLDT.

Upgrading or Reinstalling the machine Operating System

During an upgrade or reinstall the operating system, it may be possible that the JOBLOGS management facility becomes disconnected from the joblogs output queue.

As a general precaution to ensure normal functioning of this utility, you should always re-run the JLINSTALL installation command every time you upgrade, or re-install, the iSeries operating system.

RptDist Report Distribution Facility

The Navan RptDist Report Distribution facility is designed to allow you to distribute a spool file to multiple users, or locations, without any special programming required on your existing user applications.

Spool files can be distributed to

- users on the local machine
- users on other iSeries machines connected to your network (via SNADS)
- users on machines other than iSeries machines (via TCP/IP)
- user email addresses (as .pdf attachments)

The original spool file can be retained for archiving or it can be automatically removed after it has been successfully distributed.

Control is confined to specific output queues that are defined within a control list.

Specific spool file names are then defined which, combined with the list of controlled output queues, determines the spool files that will be processed by the facility.

A control interrogator job is started by the user to process the required spool file distributions. The job can be started and stopped as required, but would normally be active all day. As spool files arrive on any controlled output queue this job will scan the distribution control files and determine how to distribute the report.

Installing RptDist

The objects required to run the RptDist Report Distribution Facility are supplied as a part of NUTIL. However, they must be initialised to your system before they are capable of functioning. Installation is done from the RptDist menu:

```
:: ADDLIBLE_NUTIL___    If NUTIL is not already in your list
:: GO_UTLRPT________    Starts the RptDist main menu
```

The RptDist main menu is then displayed:

```
UTLRPT               RptDist Report Distribution Control

Select one of the following:

 Define control parameters
   1. Define RptDist control parameters                    NDFNRPTDST
   2. Work with controlled output queues                   NWRKOQCTL
   3. Work with controlled spool files                     NWRKSPLCTL

 Support functions
  10. Start RptDist Distribution Controller                NSTRRPTDST
  11. Display the RptDist message queue                    DSPMSG
  12. Display the RptDist history log                      NDSPRPTLOG

  20. End RptDist Distribution Controller                  NENDRPTDST

  30. Change the RptDist Controller delay                  NCHGRPTDLY

  40. Cleanse the RptDist history log                      Call NDSCLN

 Reporting functions
  51. Print the RptDist history log                        NDSPRPTLOG

Selection or command
===> ____________________________________________________________
F3=Exit   F4=Prompt   F9=Retrieve   F12=Cancel   F14=WRKSBMJOB   F18=WRKSPLF
```

All RptDist functions can be performed from this menu. It is also possible to run the commands individually, from a normal command line. Command names are shown at the right of the menu option.

Before RptDist can be used, you must specify what the facility is responsible for controlling. You do this using the ‘define control parameters’ options on the menu.

Define RptDist control parameters

The NDFNRPTDST command, which is also menu option 1, is used to initialise the control parameters that are used to run the distribution control interrogator job. You cannot start report distribution processing until these parameters have been defined.

```
                    Define control parameters (NDFNRPTDST)

Type choices, press Enter.

  Data queue name  . . . . . . . . DTAQ          NRPDTQ____
    Library name . . . . . . . . .                 NUTIL_____

  Control delay time in seconds  . SECS          60___
  History retention days . . . . . RETAIN        30___
  From Email name  . . . . . . . . FROMNAME

  From Email address . . . . . . . FROMEML

  Email (SMTP) server name . . . . SMTPHOST      *LOCALHOST
```

Parameter definitions

DTAQ: Specify the name of the controlling data queue. It is advisable to use the default shown unless it conflicts with your own object names.

The controlling data queue is an internal object that the system will use to determine what spool files are available for processing.

SECS: Specify the number of seconds the interrogator job will normally wait before scanning controlled output queues for available spool files.

RETAIN: Specify the number of day's history you wish to keep of the successful report distributions.

FROMNAME: Specify the name of the sender of any email distributions sent via NRptDist.

FROMEML: Specify the internet email address of a person or organization.

SMTPHOST: Specify the mail (SMTP) server with which the command will communicate.

Work with controlled output queues

The NWRKOQCTL command, which is also menu option 2, is used to define which output queues on your system that you want the RptDist interrogator job to control. Any spool files arriving on an output queue that is defined here will be eligible for automatic report distribution.

Use of the NWRKOQCTL command requires the user to have *SPLCTL special authority.

Important note: *Changes made to the Output Queue control list will be processed from the next time that the RptDist interrogator job starts. If the interrogator job is currently running you must stop and restart it for the changes to become effective.*

```
NDS005D1             Output Queue Control - Maintenance
Select

Type options, press Enter.

2=Change

   Output queue to
   Control    Library   Sts     Control Start   Defined by    Last assigned

_  ACCOUNTS   QUSRSYS    A      16/01/21 060000 QSYSOPR       17/10/21 172724
_  MARKETING  QUSRSYS    A      16/01/21 060000 BRUCE_J       17/10/21 191056
_  SALES      QUSRSYS    A      16/01/21 060000 BRUCE_J       17/10/21 191056


F3=Exit   F6=Add
```

Any output queues that have already been defined will be displayed. If no output queues have yet been defined you will be shown a display to add new output queues to be controlled (see below). To change or remove a queue displayed, enter a ‘2’ beside it and press Enter. To add more output queues, press F6.

```
NDS005D2     Output Queue Control - Maintenance
Change



                Output Queue name..: WAREHOUSE
                  Library name.....:   QUSRSYS

                Control Status.....: A         A=Active, H=Held


                Control defined on.: 16/05/21
                at time............: 14:22:12
                by user............: BRUCE_J

                Last processed.....: 17/11/21
                Time processed.....: 14.47.55


F3=Exit    F11=Delete    F12=Previous
```

For add, enter the name of the output queue that should be controlled by the interrogator job. If you specify *LIBL for the library name, the actual library name will be resolved - you must have the library in your library list while you are running the NWRKOQCTL command.

The actual library name (and not *LIBL) will be stored in the control file.

For delete, if the output queue specified is the one you wish to delete, press F11.

To change an output queue entry (for example, if you move the output queue from one library to another) you must delete the old control record and create a new one for the new library.

You can suspend the processing of an output queue by RptDist by setting the status to 'H'. To reactivate the output queue you change the status back to 'A'.

Output queue tricks and techniques

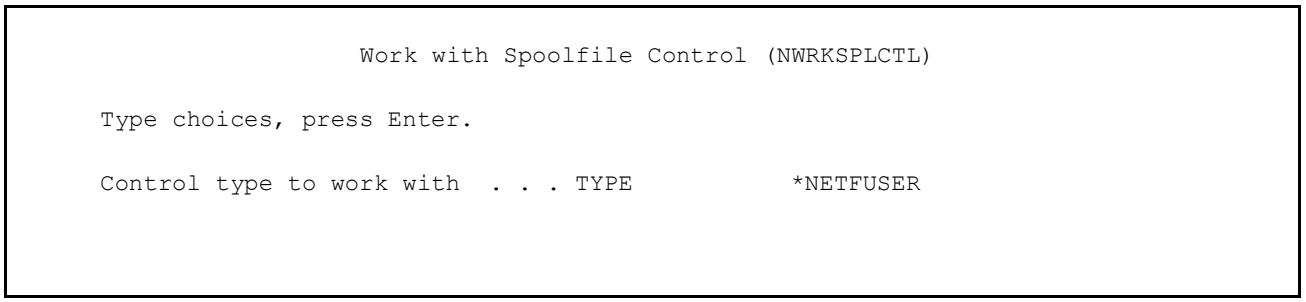
- You should only define output queues here that never have a writer attached to them. If a writer is attached it is possible that the spool file will be processed and printed before the RptDist interrogator has had a chance to distribute the spool file.
- The best method for distribution is to create special output queues specifically for the purpose of report distribution. You then change your user job descriptions (or job definitions) to write spool files to these special output queues instead of the normal queues that are attached to writers.

Work with controlled spool files

The NWRKSPLCTL command, which is also menu option 3, is used to define which spool files on your controlled output queues that you want the RptDist interrogator job to control. Any spool files defined here arriving on a controlled output queue will be eligible for automatic report distribution.

Use of the NWRKSPLCTL command requires the user to have *SPLCTL special authority.

Changes made to the Spool file control list will be effective immediately. If the RptDist interrogator job is currently running it will act immediately on any changes you make to the Spool file control list.



Distribution of spool files can be controlled in four ways:

1. You can send spool files via SNADS (SNA Distribution Services) to any user on your network
2. You can send spool files via TCP/IP to any printer queue on a remote system known to your iSeries
3. You can send spool files to another output queue on your local iSeries
4. You can send spool files (as email .pdf attachments) to internet email addresses.

You can use any, or all, of these options to control your distributions. It is possible to have one original spool file that is distributed to many network users, many remote systems and many output queues.

You do, however, need to define the distributions that you require. The RptDist interrogator job will only process spool file names that have been defined in the spool file control list. You do this via the NWRKSPLCTL command and define in the TYPE parameter which subsection of the spool file table you will work with. You can only work with (define) one subsection at a time:

*NETFUSER specifies that you will be defining distributions to be made via SNADS to network users

*TCPUSER specifies that you will be defining distributions to be made via TCP/IP to printer queues on remote systems

*OUTQ specifies that you will be defining distributions to be made to other output queues on your local system

*EMAIL specifies that you will be defining distributions to be sent via email.

Defining distributions to local output queues

Using the NWRKSPLCTL TYPE(*OUTQ) command will allow you to work with the control table for defining spool file distribution to local output queues:

```
NDS010D1           RptDist control, *OUTQ - Maintenance
Select

Type options, press Enter.
2=Change   12=Work with masks

    Spoolfile              From                      To                        Dlt
    name          Seq  Sts  Output Queue              Output Queue              Spl

_   AC0251P        10   A   *ANY                      QUSRSYS/PRT01              N
_   MR0015P        10   A   QUSRSYS/MARKETING         QUSRSYS/PRT01              N
_   MR0016P        20   A   *ANY                      QUSRSYS/PRT01              N

F3=Exit   F6=Add   F10=Resequence
```

Any spool file control entries that have already been defined will be displayed. If no entries have yet been defined you will be shown a display to add new spool files to be controlled (see below).

To change or remove an entry displayed type a ‘2’ beside it and press Enter.

To work with data masks for the entry, type a '12' beside it and press Enter.

To add more entries, press F6.

To rebuild the numbering sequence for the spool file control entries, press F10.

```
NDS010D2           RptDist control, *OUTQ - Maintenance
Add

                     Spoolfile Name.....: MR0015P___  *ALL, spool file
                     Sequence number....: ___11
                     Record status......: A           A=Active, H=Held

                     From Output Queue..: *ANY______  *ANY, output queue
                      Library name.......: __________

                     From Formstype.....:  __________ *ANY, forms type
                     From UsrDta........:  __________ *ANY, user data

                     To Output Queue....: ARCHIVE___
                      Library name.......: QUSRSYS___

                     To Formstype.......: *SAME_____  *SAME, formstype
                     To Copies..........: _1

                     Hold spoolfile.....: Y           Y=Yes, N=No
                     Save spoolfile.....: Y           Y=Yes, N=No
                     Delete spoolfile...: N           Y=Yes, N=No


F3=Exit    F11=Delete    F12=Previous
```

For add or change, enter the details of the spool file control:

Spool file name: Specify the name of the spool file. This is a unique name that identifies the actual spool file on the output queue. The special value of *ALL will process all spool files on a specified output queue. *ALL cannot be used if *ANY is used as the name of the From Output Queue.

Sequence number: Specify the sequence number that identifies this control entry for the spool file name. Actual report distribution for the spool file name is processed in ascending sequence number order. Sequence numbering can be rebuilt using the F10=Resequence option from the previous screen.

Status: If the status is set to 'A', then the control entry will be available for processing. If the status is set to 'H' then the control entry will be ignored. Setting the status to 'H' allows you to temporarily suspend processing of the control entry.

From Output Queue: Specify the name of the output queue where this spool file name will be controlled from. If you specify *ANY, then the spool file name will be processed on any output queue that is currently under the control of the RptDist interrogator job. If you specify an output queue name, the output queue must exist on the local system.

If you specify *LIBL for the library name, the actual library name will be resolved - you must have the library in your library list while you are running the NWRKSPLCTL command.

The actual library name (and not *LIBL) will be stored in the control file.

From Formstype: Specify the formstype to be controlled by this control entry. If you specify a formstype, only spool entries arriving at the controlled output queue with this formstype will be considered for processing. The special value of *ANY ignores formstype testing.

From UsrDta: Specify the User Data to be controlled by this control entry. If you specify a User Data test, only spool entries arriving at the controlled output queue with this User Data will be considered for processing. The special value of *ANY ignores User Data testing.

To Output Queue: Specify the name of the output queue where this spool file name will be copied to. The output queue must exist on the local system.

If you specify *LIBL for the library name, the actual library name will be resolved - you must have the library in your library list while you are running the NWRKSPLCTL command.

The actual library name (and not *LIBL) will be stored in the control file.

To Formstype: Specify the formstype for the copied spool file. If you specify *SAME the formstype of the original spool file will be retained.

To Copies: Specify the number of copies for the copied spool file. The default is 1 copy.

Hold spoolfile: Specify whether you want the status of the original spool file set to *HOLD after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be set to status HELD.
If you specify N=No the status of the original spool file will not be changed.

Save spoolfile: Specify whether you want the status of the original spool file set to SAVE(*YES) after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be set to SAVE(*YES).
If you specify N=No the SAVE attribute of original spool file will not be changed.

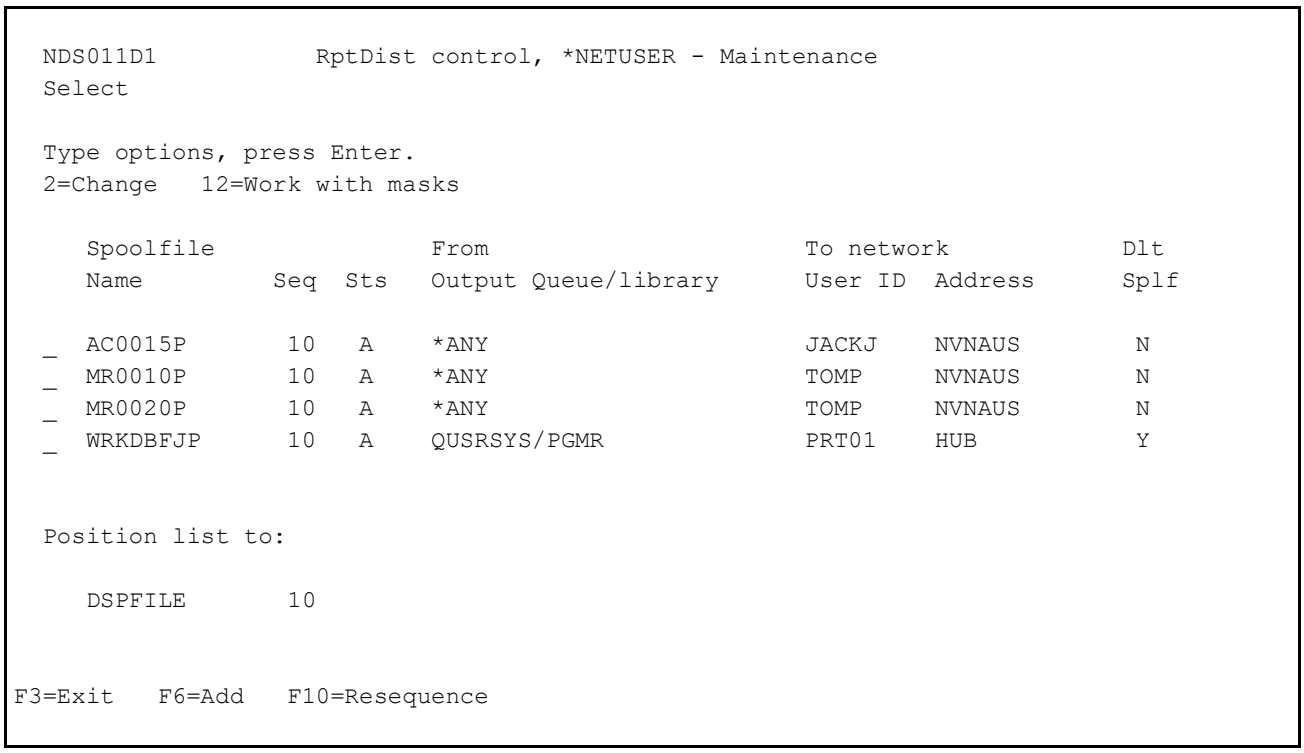
Delete spoolfile: Specify whether you want the original spool file removed from the system after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be deleted.
If you specify N=No the original spool file will remain on the system.

For delete, if the control entry displayed is the one you wish to delete, press F11.

Defining distributions to network users

Using the NWRKSPLCTL TYPE(*NETFUSER) command will allow you to work with the control table for defining spool file distribution to network users via SNADS:



```
NDS011D1           RptDist control, *NETUSER - Maintenance
Select

Type options, press Enter.
2=Change   12=Work with masks

   Spoolfile               From                      To network            Dlt
   Name          Seq  Sts  Output Queue/library      User ID  Address      Splf

_  AC0015P        10   A    *ANY                      JACKJ    NVNAUS        N
_  MR0010P        10   A    *ANY                      TOMP     NVNAUS        N
_  MR0020P        10   A    *ANY                      TOMP     NVNAUS        N
_  WRKDBFJP       10   A    QUSRSYS/PGMR              PRT01    HUB           Y


Position list to:

   DSPFILE        10


F3=Exit   F6=Add   F10=Resequence
```

Any spool file control entries that have already been defined will be displayed. If no entries have yet been defined you will be shown a display to add new spool files to be controlled (see below).

To change or remove an entry displayed type a ‘2’ beside it and press Enter.

To work with data masks for the entry, type a '12' beside it and press Enter.

To add more entries, press F6.

To rebuild the numbering sequence for the spool file control entries, press F10.

```
NDS011D2           RptDist control, *NETUSER - Maintenance
 Add

                     Spoolfile Name.....: DSPFILE___  *ALL, spool file
                     Sequence number....: ___10
                     Record status......: A           A=Active, H=Held

                     From Output Queue..: *ANY______  *ANY, output queue name
                      Library name.......: __________

                     From Formstype.....:  __________ *ANY, forms type
                     From UsrDta........:  __________ *ANY, user data

                     To Network User ID.: PRT01___ Network Address..: HUB_____

                     To Formstype.......: *SAME_____  *SAME, formstype
                     To Copies..........: _1

                     Hold spoolfile.....: Y           Y=Yes, N=No
                     Save spoolfile.....: Y           Y=Yes, N=No
                     Delete spoolfile...: N           Y=Yes, N=No


  F3=Exit    F11=Delete    F12=Previous
```

For add or change, enter the details of the spool file control:

Spool file name: Specify the name of the spool file. This is a unique name that identifies the actual spool file on the output queue. The special value of *ALL will process all spool files on a specified output queue. *ALL cannot be used if *ANY is used as the name of the From Output Queue.

Sequence number: Specify the sequence number that identifies this control entry for the spool file name. Actual report distribution for the spool file name is processed in ascending sequence number order. Sequence numbering can be rebuilt using the F10=Resequence option from the previous screen.

Status: If the status is set to 'A', then the control entry will be available for processing. If the status is set to 'H' then the control entry will be ignored. Setting the status to 'H' allows you to temporarily suspend processing of the control entry.

From Output Queue: Specify the name of the output queue where this spool file name will be controlled from. If you specify *ANY, then the spool file name will be processed on any output queue that is currently under the control of the RptDist interrogator job. If you specify an output queue name, the output queue must exist on the local system.

If you specify *LIBL for the library name, the actual library name will be resolved - you must have the library in your library list while you are running the NWRKSPLCTL command.

The actual library name (and not *LIBL) will be stored in the control file.

From Formstype: Specify the formstype to be controlled by this control entry. If you specify a formstype, only spool entries arriving at the controlled output queue with this formstype will be considered for processing. The special value of *ANY ignores formstype testing.

From UsrDta: Specify the User Data to be controlled by this control entry. If you specify a User Data test, only spool entries arriving at the controlled output queue with this User Data will be considered for processing. The special value of *ANY ignores User Data testing.

To Network User ID/Address: Specify where the spool file is to be sent to. You must ensure that there is a valid directory entry in your distribution services configuration tables to handle the transmission. Use the CFGDSTSRV (configure distribution services) command or the WRKDIRE (work with directory entries) command to define your network attributes as necessary.

To Formstype: Specify the formstype for the copied spool file. If you specify *SAME the formstype of the original spool file will be retained.

To Copies: Specify the number of copies for the copied spool file. The default is 1 copy.

Hold spoolfile: Specify whether you want the status of the original spool file set to *HOLD after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be set to status HELD.
If you specify N=No the status of the original spool file will not be changed.

Save spoolfile: Specify whether you want the status of the original spool file set to SAVE(*YES) after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be set to SAVE(*YES).
If you specify N=No the SAVE attribute of original spool file will not be changed.

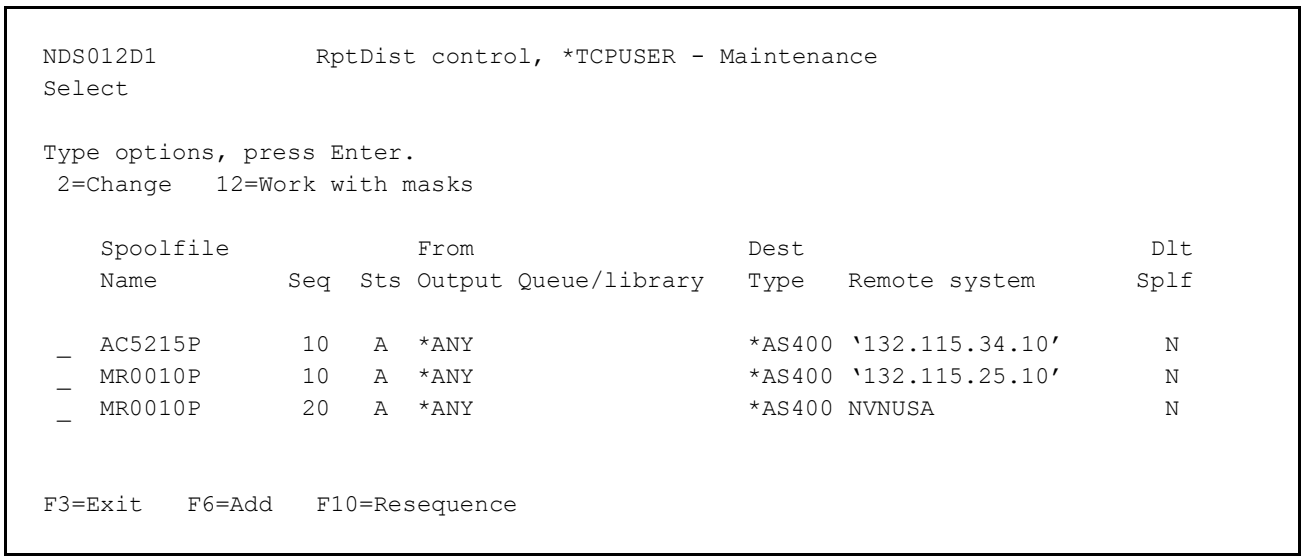
Delete spoolfile: Specify whether you want the original spool file removed from the system after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be deleted.
If you specify N=No the original spool file will remain on the system.

For delete, if the control entry displayed is the one you wish to delete, press F11.

Defining distributions to remote systems

Using the NWRKSPLCTL TYPE(*TCPUSER) command will allow you to work with the control table for defining spool file distribution to remote systems via TCP/IP:



```
NDS012D1           RptDist control, *TCPUSER - Maintenance
Select

Type options, press Enter.
 2=Change   12=Work with masks

    Spoolfile             From                    Dest                          Dlt
    Name          Seq  Sts Output Queue/library   Type   Remote system        Splf

 _  AC5215P        10    A  *ANY                   *AS400 '132.115.34.10'       N
 _  MR0010P        10    A  *ANY                   *AS400 '132.115.25.10'       N
 _  MR0010P        20    A  *ANY                   *AS400 NVNUSA                N


F3=Exit    F6=Add    F10=Resequence
```

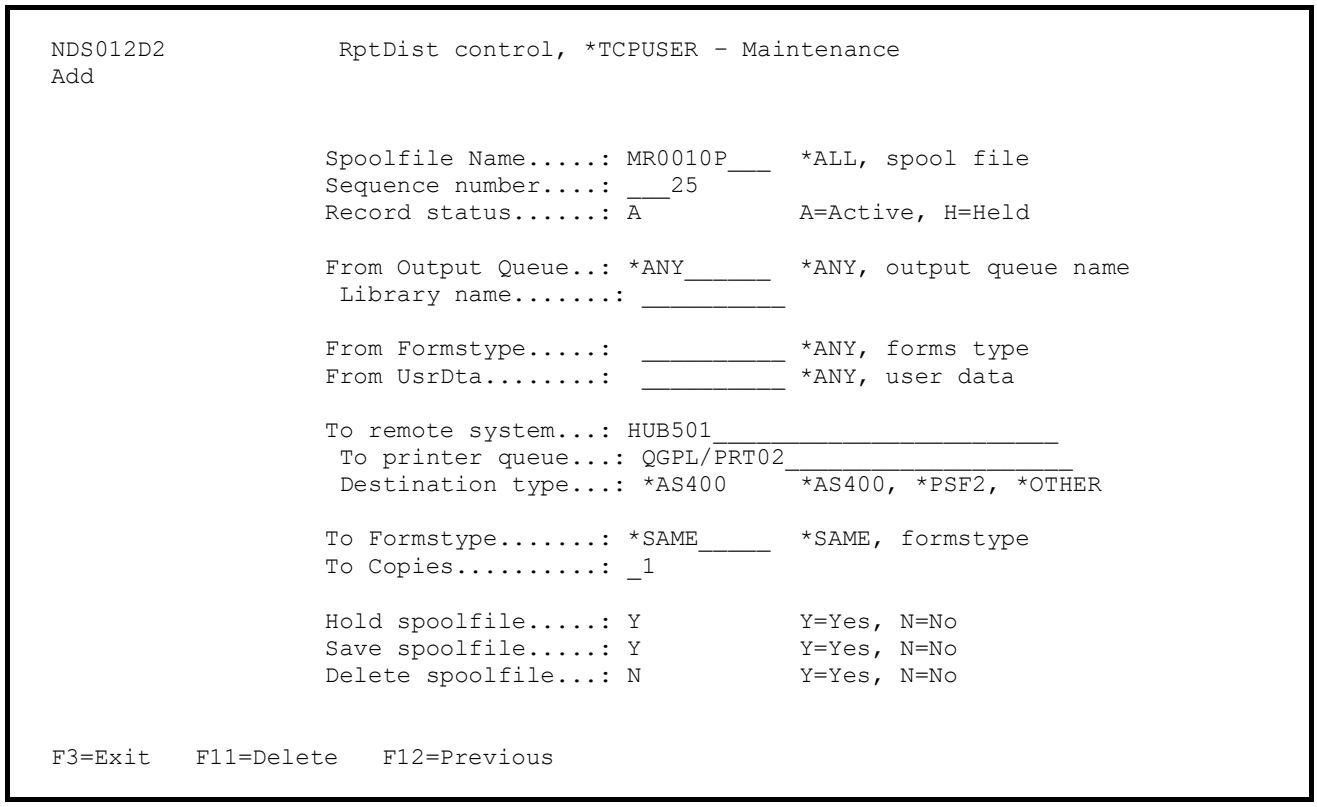
Any spool file control entries that have already been defined will be displayed. If no entries have yet been defined you will be shown a display to add new spool files to be controlled (see below).

To change or remove an entry displayed type a '2' beside it and press Enter.

To work with data masks for the entry, type a '12' beside it and press Enter.

To add more entries, press F6.

To rebuild the numbering sequence for the spool file control entries, press F10.



For add or change, enter the details of the spool file control:

Spool file name: Specify the name of the spool file. This is a unique name that identifies the actual spool file on the output queue. The special value of *ALL will process all spool files on a specified output queue. *ALL cannot be used if *ANY is used as the name of the From Output Queue.

Sequence number: Specify the sequence number that identifies this control entry for the spool file name. Actual report distribution for the spool file name is processed in ascending sequence number order. Sequence numbering can be rebuilt using the F10=Resequence option from the previous screen.

Status: If the status is set to 'A', then the control entry will be available for processing. If the status is set to 'H' then the control entry will be ignored. Setting the status to 'H' allows you to temporarily suspend processing of the control entry.

From Output Queue: Specify the name of the output queue where this spool file name will be controlled from. If you specify *ANY, then the spool file name will be processed on any output queue that is currently under the control of the RptDist interrogator job. If you specify an output queue name, the output queue must exist on the local system.

If you specify *LIBL for the library name, the actual library name will be resolved - you must have the library in your library list while you are running the NWRKSPLCTL command.

The actual library name (and not *LIBL) will be stored in the control file.

From Formstype: Specify the formstype to be controlled by this control entry. If you specify a formstype, only spool entries arriving at the controlled output queue with this formstype will be considered for processing. The special value of *ANY ignores formstype testing.

From UsrDta: Specify the User Data to be controlled by this control entry. If you specify a User Data test, only spool entries arriving at the controlled output queue with this User Data will be considered for processing. The special value of *ANY ignores User Data testing.

To Remote System/Printer queues: Specify where the spool file is to be sent to. You must ensure that there is a valid routing entry in your TCP/IP configuration tables to handle the transmission. Use the CFGTCP (configure TCP/IP) command to define your network attributes as necessary.

Note that on some non-iSeries systems the printer queue name may be case sensitive. To preserve lower case characters in the printer queue name, enclose it in single quotes 'PrintQ'.

If you are specifying an iSeries output queue you enter it in the *library_name/output_queue_name* format. If you do not specify a library name, QUSRSYS is assumed.

Destination type: Specify the system type for the remote system. The way that the spool file is sent is dependent upon this parameter:

*AS400: The remote system is an iSeries. *SCS (SNA character strings) data streams are left intact.

*PSF2: The remote system is using Print Services Facility/2. *SCS data streams are converted to ASCII for transmission. *AFP (Advanced Function Printing) data streams are left intact.

*OTHER: The remote system is not an iSeries and it is not using Print Services Facility/2. *SCS data streams are converted to ASCII before transmission.

To Formstype: Specify the formstype for the copied spool file. If you specify *SAME the formstype of the original spool file will be retained. This parameter is ignored if the destination type is not *AS400.

To Copies: Specify the number of copies for the copied spool file. The default is 1 copy. This parameter is ignored if the destination type is not *AS400.

Hold spoolfile: Specify whether you want the status of the original spool file set to *HOLD after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be set to status HELD.
If you specify N=No the status of the original spool file will not be changed.

Save spoolfile: Specify whether you want the status of the original spool file set to SAVE(*YES) after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be set to SAVE(*YES).
If you specify N=No the SAVE attribute of original spool file will not be changed.

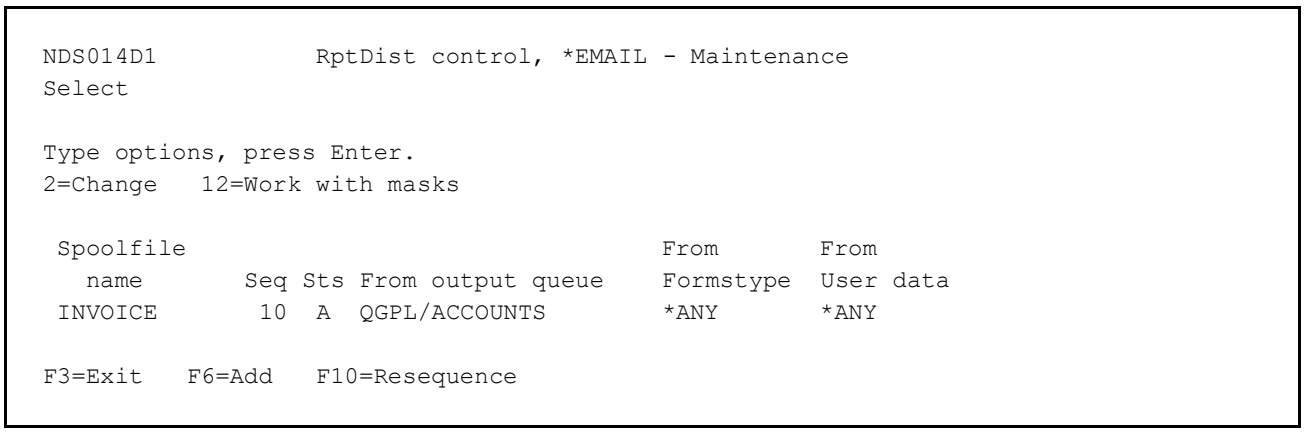
Delete spoolfile: Specify whether you want the original spool file removed from the system after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be deleted.
If you specify N=No the original spool file will remain on the system.

For delete, if the control entry displayed is the one you wish to delete, press F11.

Defining distributions to email addresses

Using the NWRKSPLCTL TYPE(*EMAIL) command will allow you to work with the control table for defining spool file distribution to be sent to email addresses:



```
NDS014D1           RptDist control, *EMAIL - Maintenance
Select

Type options, press Enter.
2=Change   12=Work with masks

 Spoolfile                                  From       From
   name       Seq Sts From output queue     Formstype  User data
 INVOICE       10  A  QGPL/ACCOUNTS         *ANY       *ANY

F3=Exit   F6=Add   F10=Resequence
```

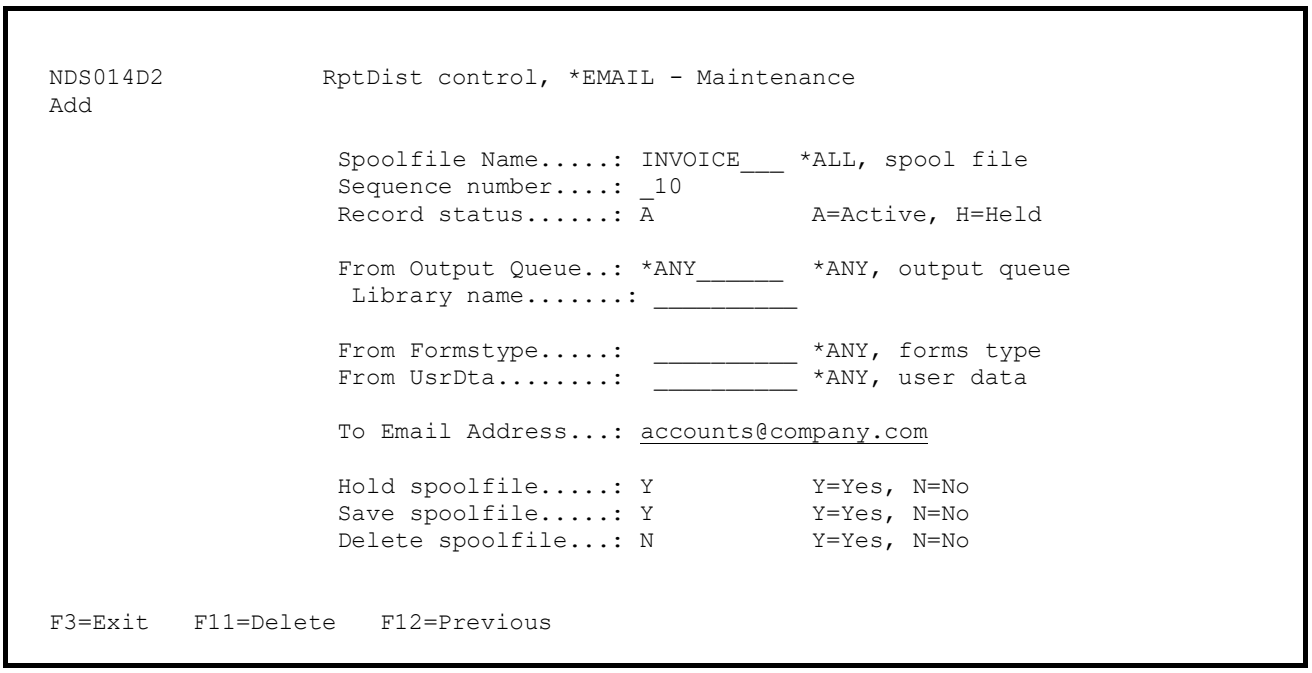
Any spool file control entries that have already been defined will be displayed. If no entries have yet been defined you will be shown a display to add new spool files to be controlled (see below).

To change or remove an entry displayed type a ‘2’ beside it and press Enter.

To work with data masks for the entry, type a '12' beside it and press Enter.

To add more entries, press F6.

To rebuild the numbering sequence for the spool file control entries, press F10.



For add or change, enter the details of the spool file control:

Spool file name: Specify the name of the spool file. This is a unique name that identifies the actual spool file on the output queue. The special value of *ALL will process all spool files on a specified output queue. *ALL cannot be used if *ANY is used as the name of the From Output Queue.

Sequence number: Specify the sequence number that identifies this control entry for the spool file name. Actual report distribution for the spool file name is processed in ascending sequence number order. Sequence numbering can be rebuilt using the F10=Resequence option from the previous screen.

Status: If the status is set to 'A', then the control entry will be available for processing. If the status is set to 'H' then the control entry will be ignored. Setting the status to 'H' allows you to temporarily suspend processing of the control entry.

From Output Queue: Specify the name of the output queue where this spool file name will be controlled from. If you specify *ANY, then the spool file name will be processed on any output queue that is currently under the control of the RptDist interrogator job. If you specify an output queue name, the output queue must exist on the local system.

If you specify *LIBL for the library name, the actual library name will be resolved - you must have the library in your library list while you are running the NWRKSPLCTL command.

The actual library name (and not *LIBL) will be stored in the control file.

From Formstype: Specify the formstype to be controlled by this control entry. If you specify a formstype, only spool entries arriving at the controlled output queue with this formstype will be considered for processing. The special value of *ANY ignores formstype testing.

From UsrDta: Specify the User Data to be controlled by this control entry. If you specify a User Data test, only spool entries arriving at the controlled output queue with this User Data will be considered for processing. The special value of *ANY ignores User Data testing.

To Email Address: Specify the email address that is to receive a copy of the Spool File. The Spool File will be converted in to a .pdf document and sent (as an attachment) to the email address specified.

Hold spoolfile: Specify whether you want the status of the original spool file set to *HOLD after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be set to status HELD.
If you specify N=No the status of the original spool file will not be changed.

Save spoolfile: Specify whether you want the status of the original spool file set to SAVE(*YES) after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be set to SAVE(*YES).
If you specify N=No the SAVE attribute of original spool file will not be changed.

Delete spoolfile: Specify whether you want the original spool file removed from the system after the distribution has been successfully processed.

If you specify Y=Yes the original spool file will be deleted.
If you specify N=No the original spool file will remain on the system.

For delete, if the control entry displayed is the one you wish to delete, press F11.

Spool file tricks and techniques

- Be careful to avoid processing loops when defining your distributions. You should ensure that a spool file will not be distributed such that it automatically appears again on a local output queue that is under RptDist control; if this happens the RptDist interrogator job will process the spool file again.
- We recommend that you only use the 'delete spool file after distribution' option when the application that generated the spool file has a reprint capability.
- If you do decide to have the spool file deleted after use, make sure you have the deletion performed on the last sequence entry for the spool file name. The RptDist interrogator will process the control entries in the following sequence:

  *NETFUSER entries, in ascending sequence order
  *TCPUSER entries, in ascending order
  *EMAIL entries, in ascending order
  *OUTQ entries, in ascending order

As an example, consider the following control table:

Spoolfile_name	*Type*	*SeqNbr*	*Delete*
RPT001P	*NETFUSER	10	N
RPT001P	*NETFUSER	20	N
RPT001P	*TCPUSER	10	**Y**
RPT001P	***OUTQ**	**10**	**N**

The *OUTQ control entry will never be processed, because the spool file would have been deleted by the *TCPUSER entry.

Defining spool file masks

An additional level of control can be applied to your distribution entries, by defining a spool file mask.

Spool file masks assist in filtering specific spool files based on the data contained in the spool file itself. A mask is a comparison field, so that if the mask defined matches the spool file being tested, the requested distribution will occur.

You define a mask by typing a '12' in the selection field for the distribution you wish to filter. The mask you define is specific to this control entry. After pressing Enter you will be shown a list of masks that have been currently defined for the requested control entry:

```
NDS013D1                  Work with Spool File Masks
Select

Type options, press Enter.
  2=Change   3=Copy   4=Delete   5=Display

Type: *NETFUSER   Spoolfile Name: RPT001     Sequence Number:    10

              Row       Column
   Line    From  To    From  To    Search spool pattern (mask)

      1     1   10      1   50    Branch 15
      1     1   10      1   50    Branch 18



F3=Exit   F6=Add   F12=Previous   F17=Top   F18=Bottom
```

In the above example, the distribution of the spool file RPT001 will occur if the text string *Branch 15* or the text string *Branch 18* is found anywhere in the area of the spool file data:

starting at line 1, print positions 1 to 50 and

ending at line 10, print positions 1 to 50.

To change an entry displayed enter a '2' beside it and press enter.

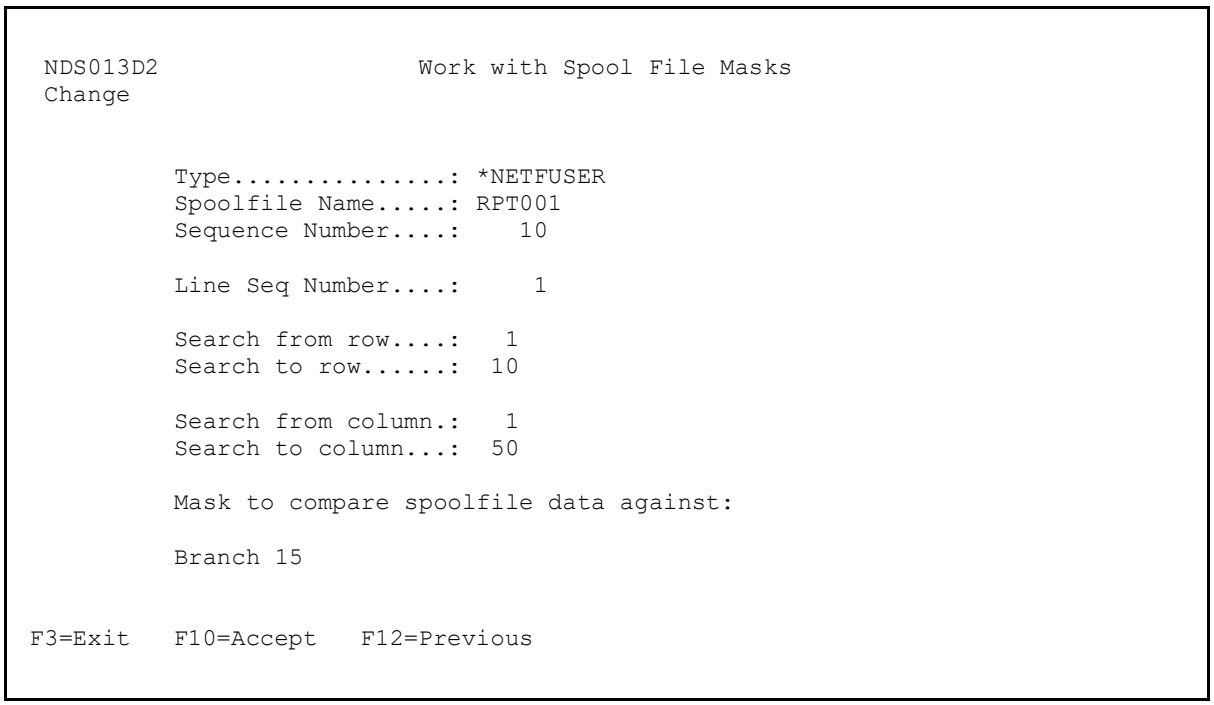
To copy an entry displayed enter a '3' beside it and press enter.

To delete an entry displayed enter a '4' beside it and press enter.

To display an entry displayed enter a '5' beside it and press enter

To add more entries, press F6.

Upon selection of an entry, or in adding a new entry, you will be shown the detail for the mask:



Line

Sequence number: Specify the sequence number that identifies this control entry for the spool file name. Mask comparison testing for the spool file name is processed in ascending sequence number order. Sequence numbering can later be rebuilt using the F10=Resequence option from the main control screen.

Search from/to row: This defines the lines on the spool file to be scanned for a match

Search from/to column: This defines the columns that should be scanned for a match - within the specified line range.

Mask to compare: Specify the text string that is to be used to scan the area of the spool file defined. If the string is found anywhere within the area specified, then the report is selected for Report Distribution. The comparison is not case-sensitive; you can use upper or lower case characters in the mask. Do not use quotation marks unless they are to be included in the string to be compared against

Press F10 to accept the entries made. If you are deleting an entry, press F11 to delete it.

RptDist in operation

NSTRRPTDST - Starting the RptDist Interrogator job

The RptDist Interrogator job is started by using the NSTRRPTDST command:

```
           Start the RptDist processor (NSTRRPTDST)

Type choices, press Enter.

 Reset data queue?  . . . . . . . RESET          *YES
 Job description name . . . . . . JOBD           SCHEDULER_
   Library name . . . . . . . . .                  NUTIL_____
 Job queue name . . . . . . . . . JOBQ           QUSRNOMAX_
   Library name . . . . . . . . .                  QSYS______
```

Parameter Definitions

RESET: Specifies what to do with any spool file entries that are currently listed on the control queue but have not yet been processed. This parameter is extremely useful when RptDist has not been used for an extended period. Old entries that may no longer require distribution can be cleansed from the control queue.

*YES will remove old entries from the control queue. You must manually send any old entries if necessary.

*NO will leave any old (not yet processed) entries on the control queue and the interrogator job will start processing them as soon as it starts up.

JOBD: The Job Description parameter defines the name of the job description to be used to submit the RptDist interrogator job. It is recommended that the default shown is used.

JOBQ: The Job Queue parameter defines the name of the job queue in which the RptDist Interrogator job will be placed. It is recommended that the default shown is used.

NENDRPTDST - Ending the RptDist Interrogator job

The RptDist interrogator job is designed to run all day, every day. It will end by itself automatically if it detects that the subsystem it is working in is ending. You can, however, end the RptDist interrogator job at any time by using the NENDRPTDST command:

```
              End the RptDist processor (NENDRPTDST)

  Unlink outq control? . . . . . . UNLINK         *YES
```

When you press enter, a 'trigger' message is sent to the Interrogator job to end. The RptDist job will end immediately after it has finished processing the current spool file it is working on. If it is not currently processing any spool file it will end directly.

UNLINK: The *Unlink outq control* parameter determines whether the RptDist controlling data queue should be detached from the controlled output queues.

The possible values are:

*YES - The link between the controlled output queues and the controlling data queue is removed. Spool files arriving on the output queues will not be logged to the controlling data queue until Report Distribution processing is restarted.

*NO - The link between the controlled output queues and the controlling data queue is retained. Spool files arriving on the output queues will continue to be logged to the controlling data queue.

Restarting Report Distribution

One problem that frequently occurs is when spool files arriving on controlled output queues are not being distributed whilst the RptDist processor is not running.

This can be controlled, using the parameters on the Start and End Report Distributions commands:

If you end Report Distribution with

NENDRPTDST UNLINK(*NO)

and then start it later with

NSTRRPTDST RESET(*NO)

then any spool files not yet distributed from the controlled output queues will be sent as soon as the RptDist processor starts up again.

Displaying the RptDist distribution history log

You can display or print the distribution log entries by using the NDSPRPTLOG command:

```
            Display RptDist history log (NDSPRPTLOG)

  Type choices, press Enter.

  Start date . . . . . . . . . . . FROMDATE       *CURRENT
  Start time . . . . . . . . . . . FROMTIME       *AVAIL_
  Ending date  . . . . . . . . . . ENDDATE        *CURRENT
  Ending time  . . . . . . . . . . ENDTIME        *AVAIL_
  Output . . . . . . . . . . . . . OUTPUT         *_____
```

Parameter definitions

FROMDATE: Specify the date (in job format) to start the display from. If you specify *CURRENT the log will be displayed for the current date only.

FROMTIME: Specify the time (in *hhmmss* format) to start the display from. If you specify *AVAIL, all available entries from the start date specified will be displayed.

ENDDATE: Specify the date (in job format) to end the display at. This parameter is ignored if you specified *CURRENT in the FROMDATE parameter.

ENDTIME: Specify the time (in *hhmmss* format) to end the display at. The end time is directly related to the end date specified. If you specify *AVAIL, all available entries for the end date specified will be displayed.

OUTPUT: Specify whether you wish to display the entries or write them to a spool file.

*PRINT: The log entries will be written to a spool file

*: If the job is an interactive job, the log entries will be displayed. Otherwise they will be written to a spool file.

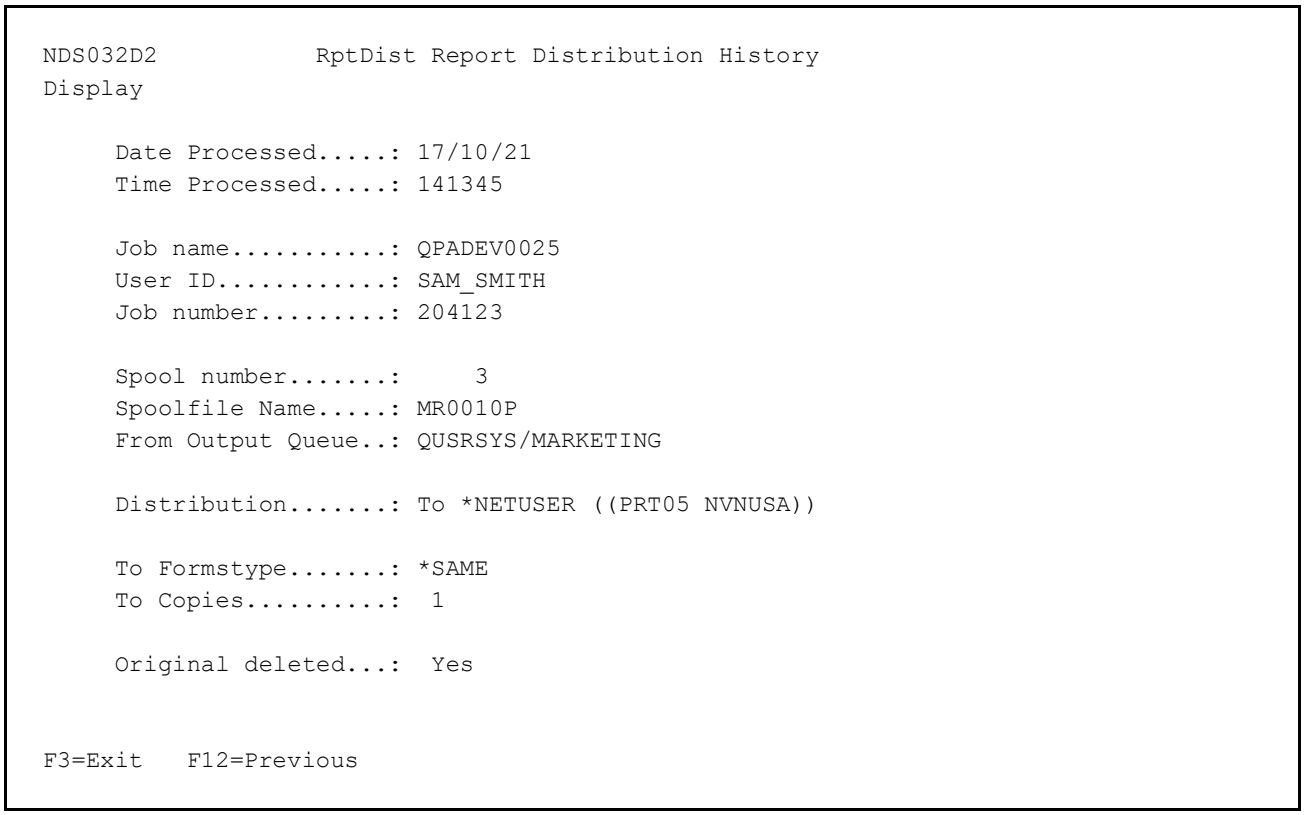
Note that old entries may no longer exist in the log file if the Log Cleanse processing has been performed.

Displaying the distribution history log

If you requested OUTPUT(*) from an interactive job you will see the log entries on the display:

```
NDS032D1           RptDist Report Distribution History
Select

Type options, press Enter.
 1=Select
                                                    Spool     Spoolfile
    Date     Time    From job/user/job nbr          number    name

 _  17/10/21 141345  QPADEV0025 SAM_SMITH  204123       3     MR0010P
 _  17/10/21 141424  QPADEV0012 QSYSOPR    205122       4     WRKDBFJP


Position list to:
    17/10/21 141345  QPADEV0025 SAM_SMITH_ 204123     __3

F3=Exit
```

The first display will show a summary of the distributions that were performed. To display the details of a distribution you select it by typing a ‘1’ beside it and pressing Enter.



```
NDS032D2           RptDist Report Distribution History
Display

     Date Processed.....: 17/10/21
     Time Processed.....: 141345

     Job name...........: QPADEV0025
     User ID............: SAM_SMITH
     Job number.........: 204123

     Spool number.......:     3
     Spoolfile Name.....: MR0010P
     From Output Queue..: QUSRSYS/MARKETING

     Distribution.......: To *NETUSER ((PRT05 NVNUSA))

     To Formstype.......: *SAME
     To Copies..........:  1

     Original deleted...:  Yes


F3=Exit   F12=Previous
```

Date/Time processed: The timestamp that shows when the spool file was distributed.

Job/User/Job number: The job identifier for the job that created the original spool file.

Spool number: The number of the original spool file within the job that created it.

Spool file name: The name of the spool file that was distributed.

From output queue: The name of the output queue and library where the original spool file was distributed from.

Distribution: How the spool file was distributed.

> To *NETUSER: The spool file was distributed using the SNDNETSPLF command, to the network user shown.
>
> To *TCPUSER: The spool file was distributed using the SNDTCPSPLF command, to the remote system and printer queue shown.
>
> To *OUTQ: The spool file was distributed, using the NUTIL DUPSPLF (duplicate spool file) command, to the local output queue shown.
>
> To *EMAIL: The spool file was distributed, using the NUTIL NCVTSPLSTM (Convert to .pdf) and NSNDSMTPML (send SMTP mail) commands, to the email address shown.

To Formstype/Copies: If the attributes of the spool file were changed before distribution, they are shown.

Original deleted: If Yes, the original spool file was deleted after this distribution was made.

Changing the RptDist Interrogator job delay interval

You can alter the interrogator delay at any time by using the NCHGRPTDLY command:

```
                  Change RptDist Delay (NCHGRPTDLY)

  Type choices, press Enter.

    Job delay time in seconds  . . . SECS           60
```

Parameter definitions

SECS: Specify the number of seconds the interrogator job will normally wait before scanning controlled output queues for available spool files. This change is effective immediately.

Cleansing the RptDist distribution history log

Each successful spool file distribution will generate an entry in the RptDist report distribution log. This file should be cleansed periodically to remove old entries from the log file. Menu option 40 will cleanse the log based on the Retain days parameter that you specified in the installation of the RptDist function. Any entries with a processing date older than the number of days will be removed from the log.

The current version of Scheduler has this cleanse routine included in the RGZ_NUTIL standard job.

RptDist Internals: how the Interrogator job works

Using the parameters entered in the NSTRRPTDST (Start RptDist Interrogator) command a job called NRPTDIST is submitted to the job queue specified. This job is an asynchronous job, which will normally stay active continuously until the subsystem it is working in is ended.

Interrogator start-up processing

The Interrogator job is started in batch, using the User Profile name that is defined on the job description's USRPRF parameter (this is normally QSYSOPR). This user must have a directory entry in the distribution directory, so that the job can send spool file transmissions.

If the controlling data queue does not currently exist, it is created.

The RESET parameter is then checked. If *YES, the controlling data queue is cleared.

An outq processor job will then be started in QSYSWRK, one for each controlled output queue. The job name will be the same as the output queue name being controlled (where more than one queue with the same name exists, a sequence number will be appended to the queue name in order to create the job name).

All output queues that have been defined on the output queue control table are then linked to a controlling data queue. There is one controlling data queue for each output queue. All new spool file entries arriving at these controlled output queues will now be available for automatic distribution.

A message advising that RptDist report distribution has commenced is then sent to the system operator message queue. The interrogator job now goes into a suspended state and awaits spool files arriving onto any of the controlled output queues.

How the spool files are processed

When a spool file arrives on a controlled output queue, it sends a trigger to the control data queue. This trigger is immediately detected by the interrogator job, which then attempts to determine whether the spoolfile should be considered for distribution.

The spoolfile name is used to scan the spool file control tables. If a match is found the output queue name is then compared to the control table entry. If both match (or the spool file name matches and the control entry has OUTQ(*ANY) specified) the spool file is distributed according to the details specified on the control table entry:

*NETFUSER table entries will be distributed using the SNDNETSPLF command
*TCPUSER table entries will be distributed using the SNDTCPSPLF command
*EMAIL table entries will be distributed using the NUTIL NSNDSMTPML command
*OUTQ table entries will be distributed using the NUTIL DUPSPLF command

After the distribution command has been processed successfully, an entry is written to the distribution log file.

If Hold=Yes is specified on the control table entry, the status of the original spool file is then set to *HELD.

If Save=Yes is specified on the control table entry, the save attribute of the original spool file is changed to SAVE(*YES).

If Delete=Yes is specified on the control table entry, the original spool file is then deleted.

If the distribution command fails, a message will be sent to the system operator message queue specifying the failure that occurred. No log entry will be written and the original spool file will remain intact. No further processing on the original spool file will be attempted by the Interrogator job.

The sequence that spool file distribution entries are processed by the interrogator is as follows:

*NETFUSER entries, in ascending sequence order
*TCPUSER entries, in ascending order
*EMAIL entries, in ascending order
*OUTQ entries, in ascending order

It is important to note that the Interrogator will attempt to process ALL distributions for each spool file. Thus it is possible for one spool file to be sent to many users, on many systems.

Interrogator 'Normal termination' processing

The interrogator will normally terminate when either the subsystem it is running in is terminated, or the NENDRPTDST command is used. In either case, the interrogator will finish processing the spool file it is currently working on (if any). All output queues currently linked to the interrogator job will then be unlinked and the job will end. A message advising normal completion will be sent to the system operator message queue.

Interrogator 'Abnormal termination' processing

The interrogator will abnormally terminate when an ENDJOB OPTION(*IMMED) is issued against it, when a processing failure occurs, or the iSeries ends abnormally. In any case a message will be sent to the system operator message queue advising the failure of the job.

Because the interrogator did not perform its normal end processing, this means that all controlled output queues are still linked to the controlling data queue.

It is important to understand how this affects the restart of the interrogator job:

If you specify RESET(*NO), any spool files that arrived at the controlled output queues since the interrogator ended abnormally will be processed and distributed first. Then normal functions will resume.

If you specify RESET(*YES), any spool files that arrived at the controlled output queues since the interrogator ended abnormally will be ignored. Normal processing will commence. You must manually distribute any spool files that were generated while the interrogator job was not running.

Starting and stopping the Interrogator

Although you can start and stop the Interrogator using the menu options, it is recommended that you have standard jobs in Scheduler to start and stop the Interrogator job. The interrogator would normally be active every working day, but having a standard job defined ensures that there is an automatic way of restarting the job if it is terminated for any reason.

The current version of Scheduler has standard jobs STR_RPTDST and END_RPTDST defined to start and stop the RptDist report interrogator. After you have defined the control parameters you can activate these standard jobs by using the WRKSTDJOB (Work with Scheduler Standard Jobs) command and removing the hold flag from the jobs.

Recovering from damaged data queues

On rare occasions the controlling data queue object may become damaged and require rebuilding. Damage like this can occur when one process (like a library save) locks the controlling data queue whilst the output queue is trying to send it a trigger entry for a new spoolfile that has arrived on the output queue.

You can verify that this failure is occurring by using the WRKACTJOB SBS(QUSRWRK) command.

RptDist controller jobs can be identified by looking in the “Function” column, where they all have the same entry “PGM-NRPTDISTQ”. Now look at the “Status” column for these jobs. In general the status of an RptDist controlling job will be showing as “DEQW”. If, over a period of 5 minutes, the job fails to return to “DEQW” status it means the job has a potential damage problem. If it stays in “SELW” it is definitely in trouble. Take note of the job name (it is usually the same name as the output queue it controls)

To rebuild the data queue you need to make sure report distribution is ended. Rebuilding the controlling data queue object will mean that any spoolfiles currently on that output queue will need to be manually processed.

To rebuild a controlling data queue:

1) End Report Distribution, using the NENDRPTDST command.

2) Once you have done this, use the WRKSBSJOB SBS(QUSRWRK) command to verify that there are no RptDist controlling jobs still running. You can identify these by looking at the “Function” column. If you see any jobs that have function PGM-NRPTDISTQ you should now end them manually, using option 4 of the WRKSBSJOB display panel. Make sure all RptDist controlling jobs have actually ended before proceeding (it may be necessary to force an immediate end for the job).

3) Do a DSPDTAARA DTAARA(NUTIL/NRPCTL). In positions 11-20 you will see the name of the library where your data queues are stored

4) Do a DSPDBF FILE(NUTIL/NRPOUTQ) and find the entry for the output queue having problems. At the end of that record (posn 51-60) you will see the name of the controlling data queue, which is usually the same name as the output queue.

For this example we assume the library name is NUTIL and the data queue name is PRT01; just use your correct values in the following information.

5) Delete the data queue: DLTDTAQ NUTIL/PRT01

6) Now start report distribution again, using the NSTRRPTDST command. The startup routine will detect that a controlling data queue object does not exist and create it again automatically.

A final note, if you are doing a library save for the library that has the RptDist controlling data queues in it, you should always make sure that report distribution is ended whilst your library save is running.

SplArc Spool file archive utility

The Navan SplArc spool file archiving utility allows you to move spooled files in to archive storage. From there you can later display and generate copies of the original spooled file output.

The SplArc functions can be accessed from the GO UTLSPL menu:

```
UTLSPL                SplArc Spool file archive utility


Select one of the following:

  Define control parameters
     1. Define SplArc control parameters                          NINSSPLARC

  SplArc operations
    10. Archive a spooled file                                    NARCSPLF
    11. Cleanse spool files                                       CLNSPLF

    30. Work with archived spool files                            NWRKSPLARC

    40. Cleanse the SplArc spool file archive                     NRGZSPLA

  Other options
    90. Sign off



Selection or command
===>

F3=Exit    F4=Prompt    F6=DSPMSG    F9=Retrieve    F12=Cancel    F14=WRKSBMJOB
F18=WRKSPLF
```

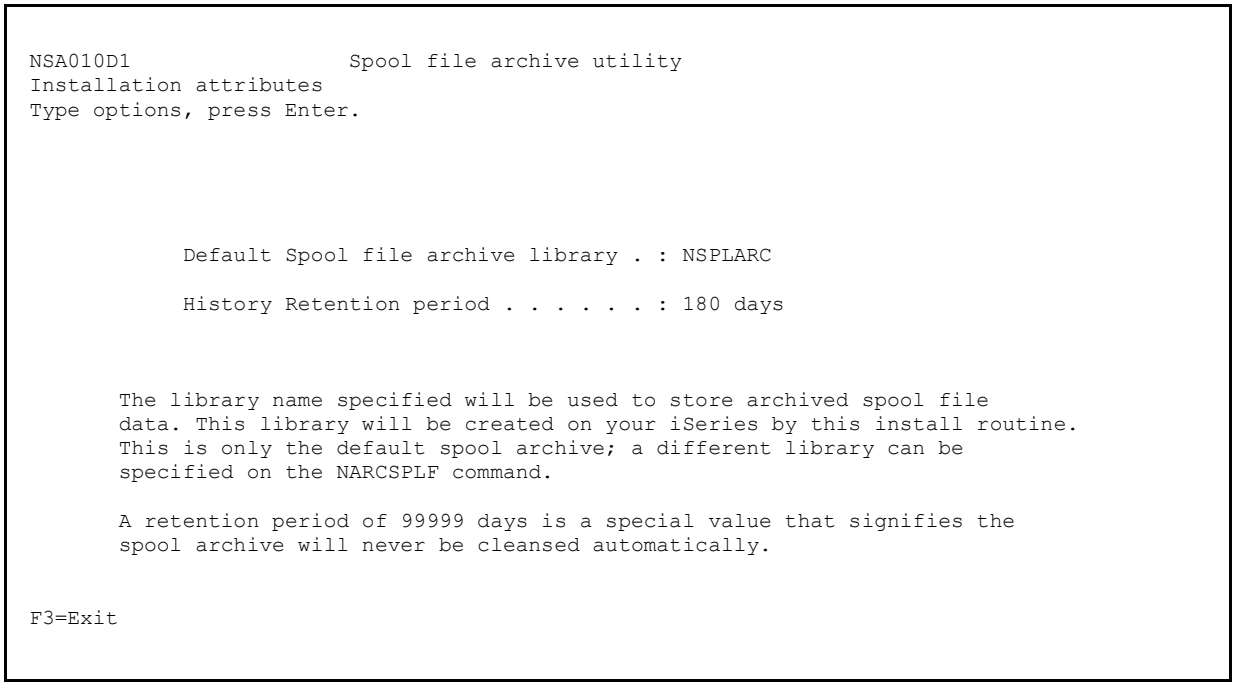
Before Splarc can be used it must be installed; this can be achieved by taking option 1 from the menu.

Running the SplArc installation command

Option 1 on the UTLSPL command runs the NINSSPLARC (Install SplArc) command:

The NINSSPLARC command installs the Spool File Archive Utility onto your machine. NINSSPLARC -must- be run before you attempt to use the SplArc utility. You must have *SPLCTL rights defined on your user profile to be able to run this command.

The command has no parameters. When you press enter, the 'install options' work panel will be shown.



The installation parameters are as follows:

Spool file archive library: SplArc will create special 'user space' objects to store the image and attributes of your archived spool files. By default, these objects will reside in the library name that you specify here. It is highly recommended that the library name you specify not be used for any purpose other than archiving spool files. The default name is NSPLARC but you can alter it to whatever you choose. If the library name does not already exist on your iSeries it will be created by this installation command.

History retention period: This is the number of days that the spool file image will remain in the archive before it becomes eligible for removal by the SplArc cleanse procedure.

An entry of 99999 days signifies that no automatic cleansing of the archived spool files should occur.

Archiving spooled files

There are two main ways of moving spooled files in to the SplArc archive, you can move a specific file in there or you can use the existing NUTIL cleanse command to do it automatically for you. Both of these methods are discussed below.

NARCSPLF - Archive Spooled file

The NARCSPLF command will transfer a spool file from an output queue in to the spool file archive. It removes the original spool file once it has been transferred to the archive.

Command parameters are as follows:

```
                        Archive Spooled File (NARCSPLF)

Type choices, press Enter.

Spooled file . . . . . . . . . . FILE
Job name . . . . . . . . . . . . JOB            *
  User . . . . . . . . . . . . .
  Number . . . . . . . . . . . .
Spooled file number  . . . . . . SPLNBR         *LAST

                           Additional Parameters

Spool Archive library  . . . . . ARCLIB         *DFT
```

Spooled file: Specifies the name of the spooled file to process and move to the spool file archive. This is a required parameter.

Job name: Specifies the name of the job that created the spooled file to be processed. An '*' in the Job parameter signifies that the current job created the spooled file.

Spooled file number: Specifies the number of the job's spooled file that is to be processed.

> An entry of *LAST signifies that the spooled file with the highest number and the specified file name is used.
> An entry of *ONLY signifies that only one spooled file in the job has the specified file name; therefore, the number of the spooled file is not necessary.

The Spool archive library: Specifies which library should be used to store the archived spool objects.

> *DFT - The archived spool objects will be stored in the default archive library (as defined during installation of the Spool Archive Utility).
> spool-archive-library-name: The archived spool objects will be stored in the specified archive library name. This library must exist on the iSeries prior to performing this command.

CLNSPLF - Cleanse spooled file

If you specify CLNOPT(*ARCHIVE) on the CLNSPLF command, any spooled file selected will be moved from the output queue into the selected SplArc archive library.

For a full discussion of the CLNSPLF command please refer to the Utility Commands section later in this manual

NWRKSPLARC - Working with archived spool files

Once spooled files are moved to the SplArc archive they are no longer considered spooled data by the iSeries system. To process them you must use the WRKSPLARC (work with archived spool files) command:

Use of the NWRKSPLARC command requires the user to have *SPLCTL special authority.

```
                  Work with archived spool files (WRKSPLARC)

Type choices, press Enter.

Display list start date  . . . . STRDATE        *TODAY
Display list sequence  . . . . . LSTSEQ       > *USER
Spool Archive to display . . . . ARCLIB         *ALL
Allow processing operations .. . ALWPRC         *YES
```

The command parameters are as follows:

Display list start date: The STRDATE parameter allows you to start listing spool files from a specified job date (note that 'Job date' refers to the date that the spooled file was originally created, before it was moved in to the archive).

An entry of *TODAY will start the list from today's date.
An entry of *START will show all archive entries

List Sequence: The LSTSEQ parameter specifies which sequence the entries should be shown in:

*DATE - The entries will be shown in date sequence.
*NAME -The entries will be shown in job name sequence.
*USER -The entries will be shown in user ID sequence.
*NUMBER -The entries will be shown in job number sequence.

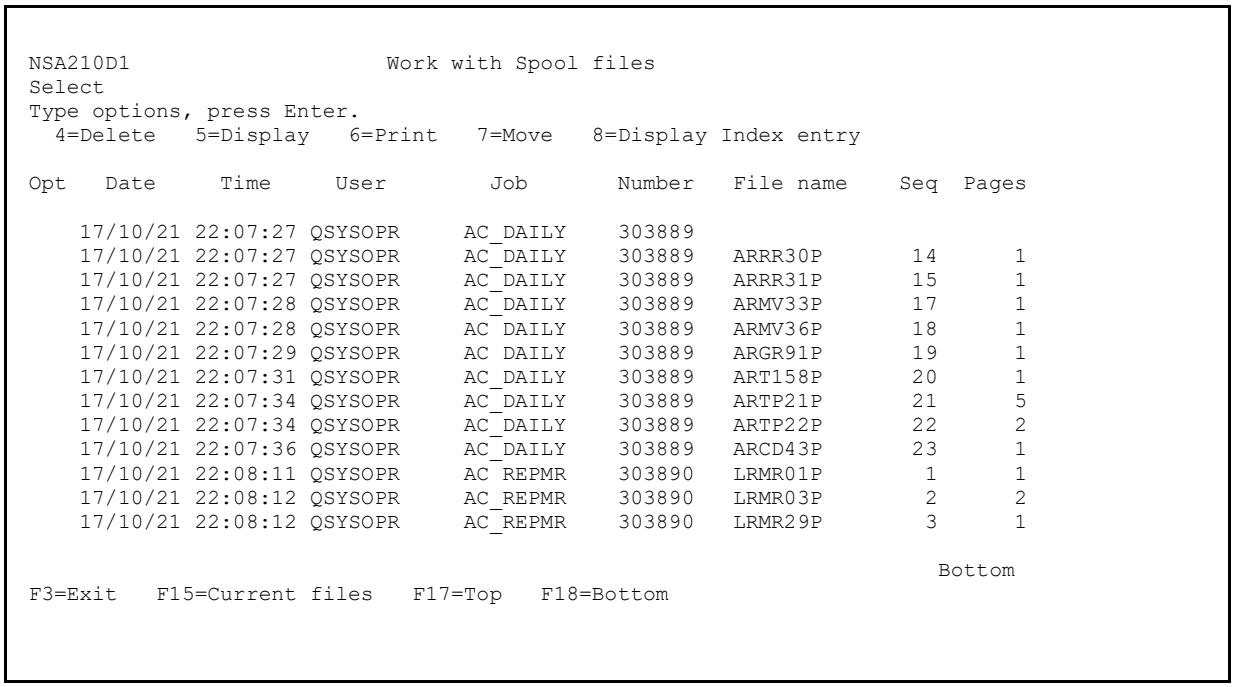
Spool archive library: The ARCLIB parameter specifies which archived spool files can be accessed by the in the work panel:

*ALL -All archived spool file entries will be displayed to, and can be accessed by, the user.
*DFT -The work panel will only display spool files that have been archived to the default archive library (as defined during installation of the Spool Archive Utility). The user will be restricted to performing actions against entries stored in the default archive library.
spool-archive-library-name - The work panel will only display spool files that have been archived to the specified archive library name. The user will be restricted to performing actions only against those entries stored in the specified archive library.

Allow processing operations: Specifies whether the user can move/delete archive index entries and spooled file data. An entry of *YES will enable the 4=Delete and 7=Move options of the work panel.

From here you will then be shown the main 'work with' panel:

Working with archive index entries



```
NSA210D1                    Work with Spool files
Select
Type options, press Enter.
  4=Delete   5=Display   6=Print   7=Move   8=Display Index entry

Opt   Date     Time     User        Job       Number   File name    Seq  Pages

    17/10/21 22:07:27 QSYSOPR     AC_DAILY    303889
    17/10/21 22:07:27 QSYSOPR     AC_DAILY    303889   ARRR30P       14      1
    17/10/21 22:07:27 QSYSOPR     AC_DAILY    303889   ARRR31P       15      1
    17/10/21 22:07:28 QSYSOPR     AC_DAILY    303889   ARMV33P       17      1
    17/10/21 22:07:28 QSYSOPR     AC_DAILY    303889   ARMV36P       18      1
    17/10/21 22:07:29 QSYSOPR     AC_DAILY    303889   ARGR91P       19      1
    17/10/21 22:07:31 QSYSOPR     AC_DAILY    303889   ART158P       20      1
    17/10/21 22:07:34 QSYSOPR     AC_DAILY    303889   ARTP21P       21      5
    17/10/21 22:07:34 QSYSOPR     AC_DAILY    303889   ARTP22P       22      2
    17/10/21 22:07:36 QSYSOPR     AC_DAILY    303889   ARCD43P       23      1
    17/10/21 22:08:11 QSYSOPR     AC_REPMR    303890   LRMR01P        1      1
    17/10/21 22:08:12 QSYSOPR     AC_REPMR    303890   LRMR03P        2      2
    17/10/21 22:08:12 QSYSOPR     AC_REPMR    303890   LRMR29P        3      1

                                                                      Bottom
F3=Exit   F15=Current files   F17=Top   F18=Bottom
```

The list shows the spooled files currently stored in the selected SplArc archive. Each line in the list represents one archived spooled file.

Opt (Option) - Use this column to perform different operations on individual spool files. Type the option number against the spool file you wish to process and press enter. You can type the same option next to more than one spool file at one time, and you can type different options next to different spool files at one time.

- 4= Delete (if enabled) allows you to remove the spooled file from the SplArc archive
- 5= Display allows you to display the spooled file
- 6= Print allows you to generate a spooled file on to your current output queue.
- 7= Move (if enabled) allows you to move the spool file from one archive library to another.
- 8= Display index entry allows you to view the summary index entry for this archived spool file.

Function keys are as follows:

- F3= Exit the work with panel
- F15= View your current output queue. This allows you to access spooled files you have re-created (using the option 6=Print function)
- F17= Brings you back to the top of the list
- F18= Brings you to the bottom of the list
- F21= Allows you to display a command entry window

Cleansing the SplArc spooled file archive

Left unattended, the SplArc archive library can grow very large. As such you should monitor the amount of archived data you keep. There are two ways of reducing the storage allocated to the SplArc utility

You can manually remove spooled files from the archive by using option 4=Delete from the 'work with archived spool files' panel discussed previously

You can remove spooled files automatically based on the length of time they have been stored in the SplArc archive. This is done using the NRGZSPLA command as described below.

NRGZSPLA - Cleanse the SplArc archive

The NRGZSPLA command checks the Spool File archive and removes any archived spool files that are older than the Retention days specified in the SplArc control information (loaded by the NINSSPLARC installation command).

The command has no parameters.

For each index entry found that is older than the calculation retention period, this command will remove the index entry and delete the archived spool image data.

It is recommended that the command be submitted to batch for processing, as it may be a long running process.

Accessing SplArc entries from user defined programs

You may find that your own development projects require access to spooled file data stored in the SplArc spool file archive. For this reason we have developed a set of 'internal' SplArc commands that can be used by your programs to display, print or delete SplArc entries.

An example program is supplied in source file NUTIL/QSTUFF, source member PRCSPLA. This example shows how you could write your own program to remove any archived spool file that is greater than 100 pages in length.

The internal commands provided are

- NDSPSPLA - Display spool file
- NPRTSPLA - Print spool file
- NMOVSPLA – Move spool file to another archive library
- NDLTSPLA - delete spool file

The commands cannot be used from a command line, only from within programs. All of these internal commands access the spooled file data using an internal index number, which can be found in the NSPINDX SplArc index file, field NSUSIN.

It is highly recommended that the NSPINDX index file be used as the main driver file for any user processing of the SplArc database. It is also highly recommended that you process the entries *only* via the internal commands supplied. For a listing of the fields within this file please use the NUTIL/DSPRCDFMT or NUTIL/DSPFILE command to process file NUTIL/NSPINDX.

NUTIL Utility commands and programs

An overview

NUTIL utility commands and programs have been developed to fill in areas not covered by standard OS/400. These commands function in the same way as standard OS/400 commands, and all support the command prompt facility from a command entry screen.

Unless specified, NUTIL utility commands make no attempt to override system security. Any use of a NUTIL utility command is subject to normal OS/400 object security.

Since many of the commands are interrelated, it is suggested that you do not attempt to split the library by moving objects to other libraries.

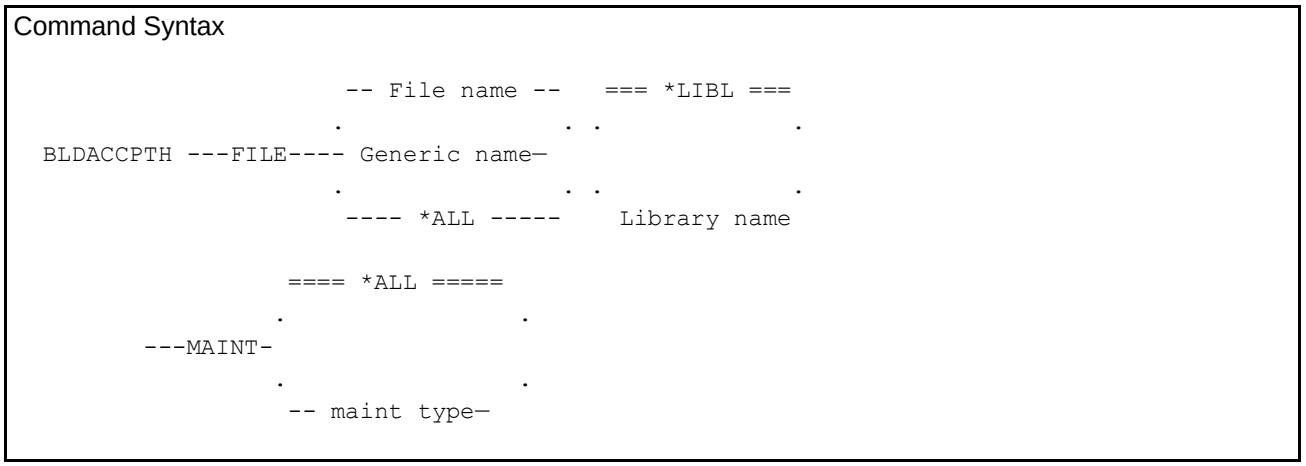
NUTIL utility commands can be accessed via menus or can be entered from a command line. If you enter ‘GO UTLCMD’ from a command entry line you will be shown the NUTIL Utility commands menu.

Commands

BLDACCPTH - Build/Rebuild File Access Paths

The BLDACCPTH command will attempt to build/rebuild an access path over the file(s) requested. This is particularly advantageous where you use MAINT(*DLY) a lot, or where the system has terminated abnormally, possibly requiring access paths to be rebuilt. The command prompt is as follows:

Command Syntax



Parameter Definitions

The MAINT parameter allows you to specify which access path maintenance type you wish to attempt to rebuild.

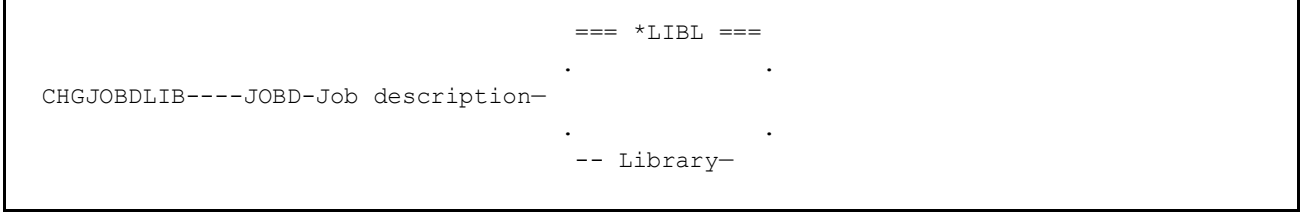
You may specify *IMMED, *DLY, *REBLD or *ALL.

CHGJOBDLIB - Set JOBD to current user library list

This command sets the library list of the nominated Job Description to be the same as the current interactive library list.

This command will update the job description specified directly.

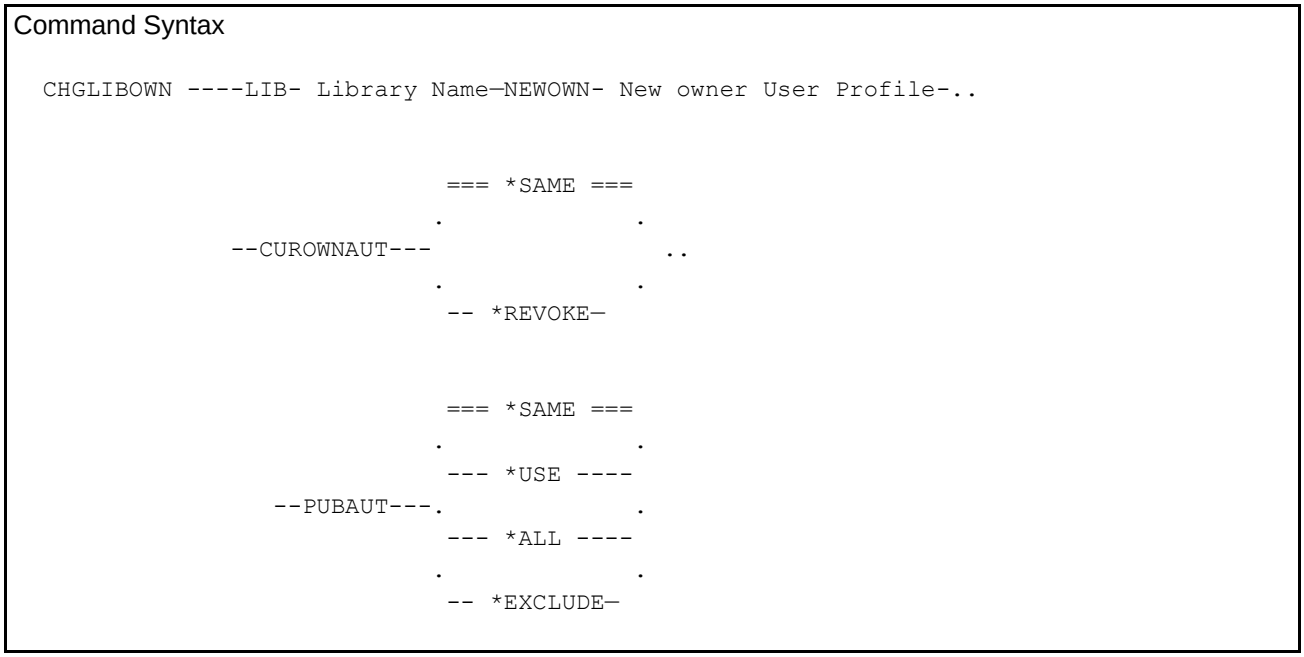
Command Syntax



CHGLIBOWN - Change Library Ownership

This command alters the object ownership of all objects in a library (and the library objects itself) to a new owner. Optionally, it can also set the *PUBLIC authority to the objects in the library at the same time.

Command Syntax

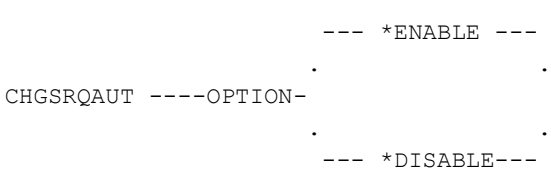


The current owner of each object in the library will lose ownership of the object and then will either retain the same authority rights to the object or lose them completely, depending on the CUROWNAUT parameter.

CHGSRQAUT - Change authority to use System Request

This command allows you to enable/disable system request functions for a user job. It is extremely useful in user application programs, to be able to stop a user from ending the currently running function by the method of using option 2 (end previous request) of the system request menu.

Command Syntax



The command will apply the request to the user profile associated with the current job.

If the option is *DISABLE, authority to the System Request menu and the ENDRQS command will be removed from the user profile.

If the option is *ENABLE, authority will be restored.

CHKACTJOB - Check whether job is active in a subsystem

This command allows you to determine whether a specified job is currently active in a specified subsystem. This is especially useful in batch jobs, where you know that the job you are submitting cannot run if another known job is currently active.

Command Syntax

```
CHKACTJOB ----JOB- Job name to test --- SBS- Subsystem to check
```

If the specified job is currently active in the specified subsystem, escape message CHK0001 will be sent.

CHKDDMLNK - Check DDM Link Status

This command allows you to determine if a Distributed Data Management link is available to the specified remote system. If it is found to be unavailable an escape message will be sent; your user job can then monitor for this message using the MONMSG command.

Command Syntax

```
CHKDDMLNK ----RMTSYS- Remote system to test
```

A connection will be attempted to the remote system specified using the currently logged in user. If the connection is successful message DDM0010 will be issued. If the connection is unsuccessful, escape message DDM0001 will be sent to the calling program. This message can be monitored for in the user program.

CLNPFM - Cleanse Physical File member

This command will remove records from a file, based on extraction criteria provided.

The extraction selection criteria is in the form of an OPNQRYF (Open Query File) QRYSLT (Query Select) parameter and is subject to the same syntax rules as that parameter.

You can optionally archive the removed records to a separate archive file, allowing you to keep historical data.

Command Syntax

```
   CLNPFM---FILE- file library/name ---MBR- member name --...

             QRYSLT- Selection criteria --...

             CLNOPT- Cleanse Option --...

             RGZPFM- Reorganise Option --...

             ARCLIB- Archive Library—CRTFILE- Create Archive?
```

Parameter Definitions

CLNOPT: The cleanse option may be either *REMOVE or *ARCHIVE.

*REMOVE will remove (delete) the selected records from the file, whereas
*ARCHIVE will move (copy, then delete) the selected records from the file into an archive file.

RGZPFM: The reorganise option may be either *YES or *NO.

*YES will attempt a Reorganise Physical File Member (RGZPFM) command after the file has been cleansed.

ARCLIB: The name of the library which contains the archive file. The archive file must be at the same level ID as the file being cleansed, as well as having the same name. A record format level check will be performed to ensure the files are compatible before archiving actually commences.

CRTFILE: Determines whether archive file creation is required or not.

If *NO, the file is assumed to already exist.
If *YES, an archive file will be created, if required.

Take the following examples:

```
CLNPFM  ACCTS/EMPMASTR QRYSLT('STATUS *EQ "D"')
```

This example will remove all records in file ACCTS/EMPMASTR where the status field is set to 'D'.

```
CLNPFM  ACCTS/EMPMASTR QRYSLT('TRMDTE *LT 020630')
```

This example will remove all records in file ACCTS/EMPMASTR where the employee termination date (field TRMDTE) is older than June 30. 2002.

```
CLNPFM  ACCTS/EMPMASTR QRYSLT('MNAME *CT "SMITH"')
                       CLNOPT(*ARCHIVE) ARCLIB(ACCTHIST) CRTFILE(*YES)
```

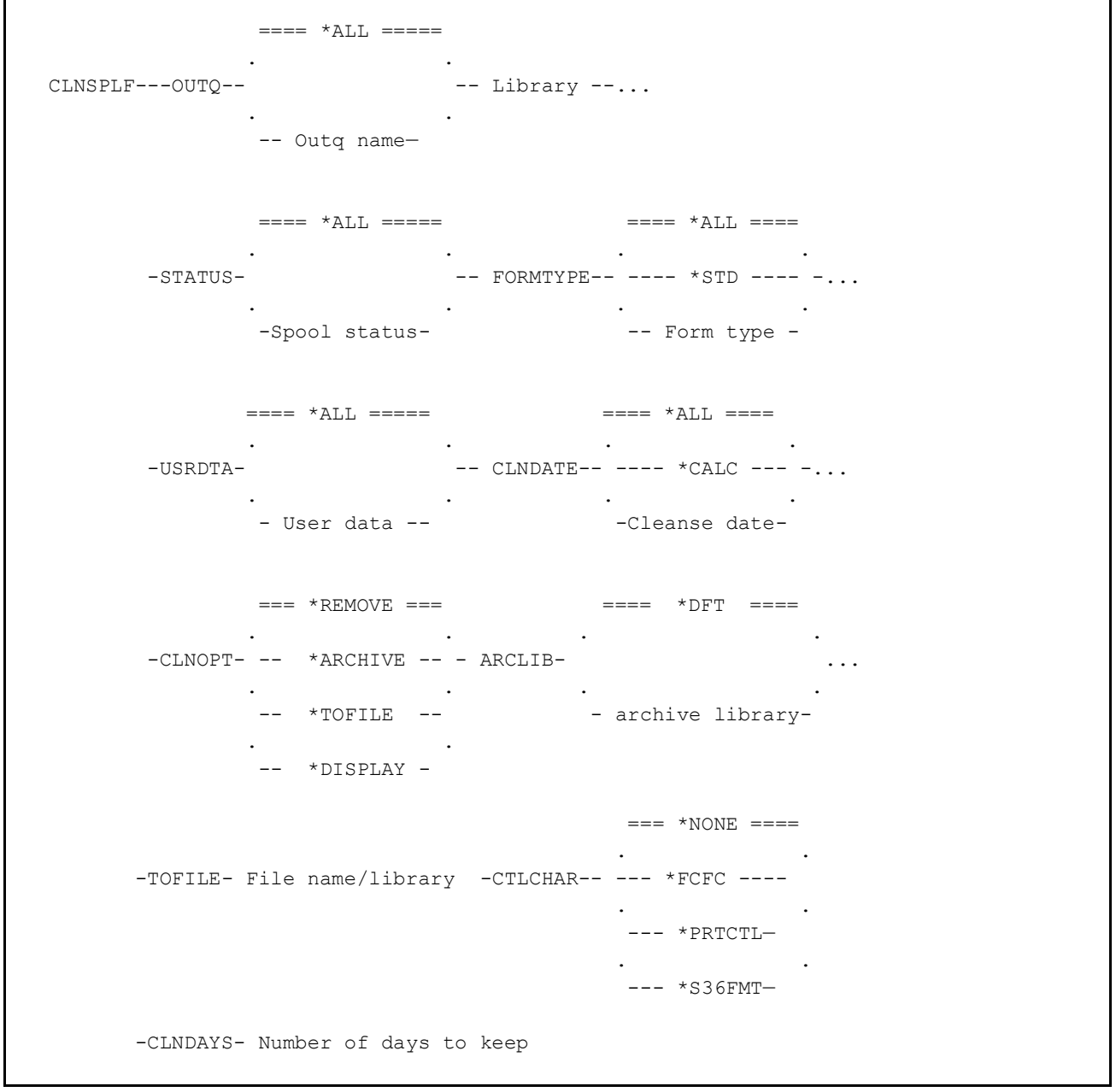
This example will move all records in file ACCTS/EMPMASTR, where the employee name field (MNAME) contains the character string 'SMITH', into an archive version of the file ACCTHIST/EMPMASTR. If the specified archive does not yet exist, it will be created.

CLNSPLF - Cleanse Spool Files

This command allows you to perform group processing of spool files on your iSeries. Spool files can be removed or archived based on input parameters entered. You must have *SPLCTL rights defined on your user profile to be able to run this command.

An *AND relationship exists between the selection parameters; for example if you enter specific values for two parameters, then the spool file must meet both criteria for it to be selected.

Command Syntax



Parameter Definitions

OUTQ: The output queue parameter specifies which output queue to process. You may enter a specific output queue name, or *ALL.

USER: The Select for User parameter allows you to select only spool files for a specific User ID, or you may enter *ALL for all users.

FILE: The Select spool file name parameter allows you to identify a specific spool file name to process, or you may enter *ALL to process all names.

STATUS: The Select spool file status parameter allows you to select only those spool files at a specific status. The valid values are:

*ALL	Status selection will be ignored
*RDY	Those ready to be printed will be processed
*HLD	Those currently held will be processed
*OPN	Those currently open will be processed
*CLO	Those currently closed will be processed
*SAV	Those currently saved will be processed
*WTR	Those currently being written will be processed

FORMTYPE: The Select formtype parameter allows you to select only those spool files using a specific formtype, or you may enter *ALL for all formtypes.

USRDTA: The Select user data parameter allows you to select only those spool files that have specific user data in their spooling attributes, or you may enter *ALL to ignore selection based on user data.

CLNDATE: The Select prior to date parameter allows you to select only those spool files that were created before a specified date. You may enter a specific cleanse date, or you can have the command calculate a cleanse date by specifying *CALC and entering a ‘number of days to keep’ value in the CLNDAYS parameter. Entering *ALL will ignore selection based on creation date.

CLNDAYS: This parameter is required if the CLNDATE parameter was entered as *CALC. The command will use the ‘number of days to keep’ entry made here in the calculation of a cleanse date. You may enter a number between 0 and 365, where 0 is today, 1 is yesterday, and so on.

CLNOPT: The cleanse option may be either *REMOVE, *ARCHIVE, *TOFILE or *DISPLAY.

*REMOVE will remove (delete) the selected spool files;
*ARCHIVE will move the selected spool files to the SplArc spool (discussed earlier in this manual)
*TOFILE will move the selected spool files into a user defined physical database file
*DISPLAY will display each spool file selected.

If you specify CLNOPT(*ARCHIVE), you must specify the additional parameters:

ARCLIB: specifies which library should be used to store the archived spool objects. If a library name is not entered, it is assumed that the archived spool objects will be stored in the default archive library (as defined during installation of the Spool Archive Utility).

If you specify CLNOPT(*TOFILE), you must specify the additional parameters:

TOFILE: specifies which user defined physical database file is to be used to store the copied data from the selected spool files. If this file does not exist at the time of running CLNSPLF, the process will fail (spool file will not be processed)..

CTLCHAR: specifies which print control characters (if any) are to replace the spooled file's internal print control characters. The available options are:

*NONE No print control characters are created
*FCFC The first character of every record will contain one of the ANSI forms control codes
*PRTCTL The first four characters of every record will contain skip/space before values
*S36FMT The format of the records will be the same as that created by the IBM System/36 COPYPRT procedure.

Copying Spooled file data to a data file (*TOFILE)

If your spool files are to be copied to a data file, you must create a physical database file in which they will be stored.

The record length of the file should be made sufficiently long enough to store the print line data PLUS the forms control characters requested (if any).

Each spool file copied to the TOFILE will be stored as a separate member in the file, so the file you create must allow multiple members. The name of the archive member created in the file will be made up from the job number and the spool number, prefixed by an 'S':

Job number (105245/TESTUSR/LISTSPL), Spool number 0001
will create member S105245001 in the specified TOFILE.

The member text for the TOFILE member will contain the information relating to the original spool file:

From	*To*	*Description*
1	10	Job name
11	20	User ID
21	26	Job number
31	40	Spool file name
41	44	Spool number

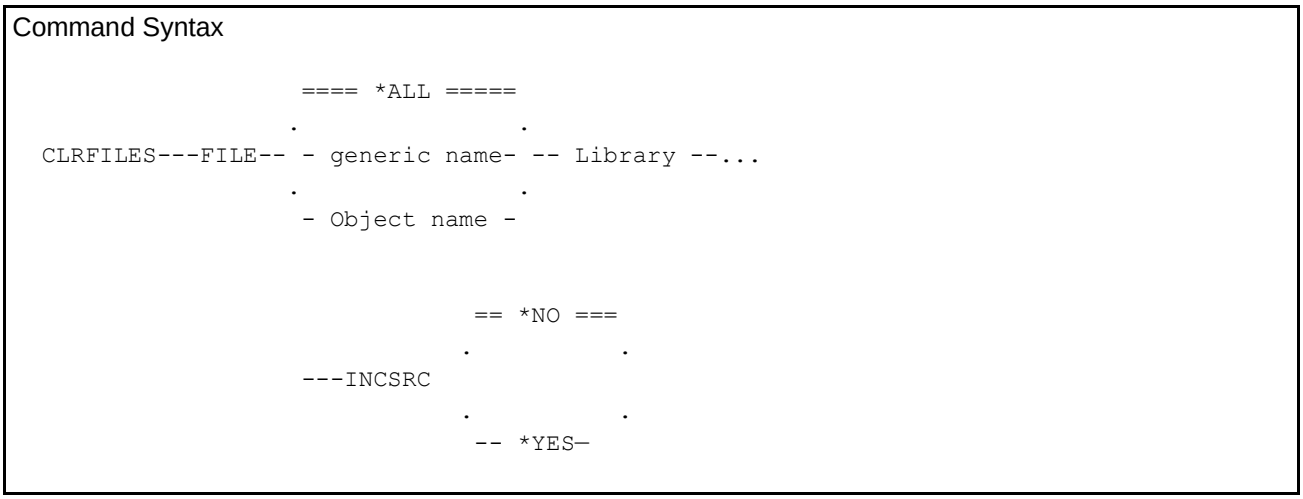
Archiving spooled file data to the SplArc spool archive utility

For information on this topic refer to the "SplArc – Spooled File Archiving" section of this manual.

CLRFILES - Clear Library Physical Files

This command allows you to clear (remove all data, including deleted records) a group of physical files in a specified library.

Command Syntax



The request may be generic, so that a request of ‘GL*’ will clear all gl... files within the specified library. ALL members are cleared within each file, including source files (if selected).

Source files can be included or excluded using the INCSRC parameter.

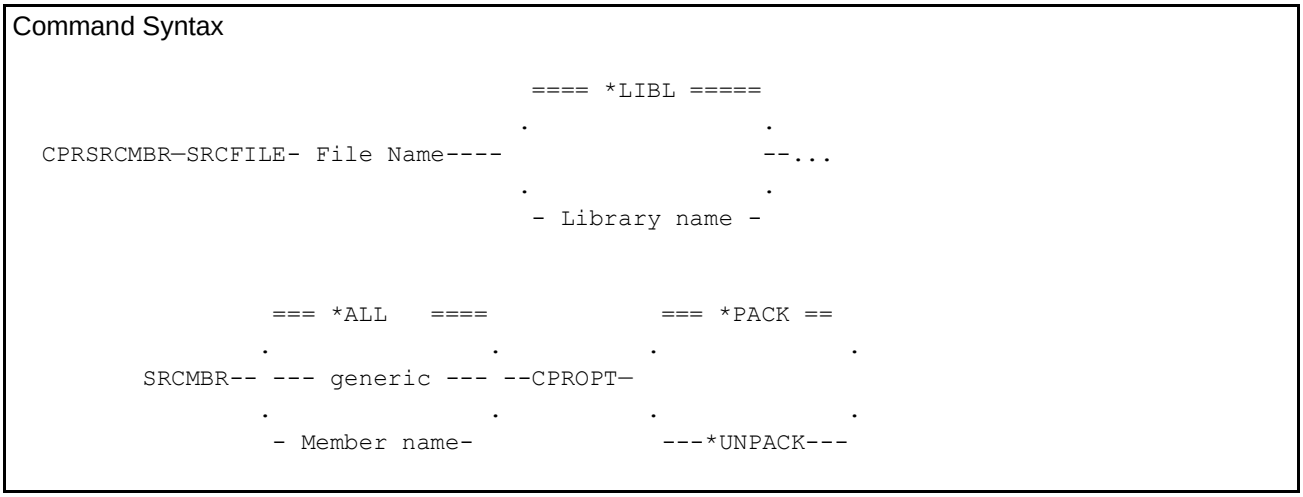
> An entry of *YES will include source files in the process, whereas an entry of *NO will ignore them.

CPRSRCMBR - Compress/Decompress Source File Member

This command allows you compress or decompress a source file member.

Compression can typically save 80% of disk storage currently held by your source files. The resulting compression ratio is reported upon successful compression of a source member.

Command Syntax



The CPROPT parameter specifies which function you are performing, either

*PACK (Compress a source member) or
*UNPACK (Decompress a previously compressed source member).

The source member name can be a generic (or *ALL) entry, allowing you to process a group of source members, or an entire source file.

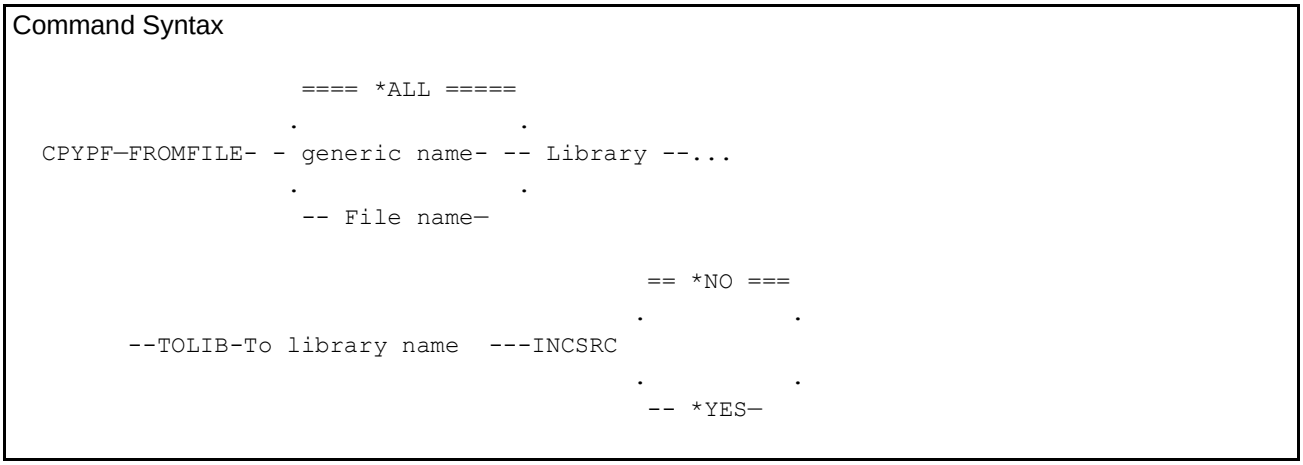
***WARNING** - Do not attempt to access a compressed source member via SEU. The Source Entry Utility is not capable of displaying the compressed data and will cause SEU to terminate abnormally.*

Whilst this does not affect the integrity of the iSeries or the source member being accessed, the situation should be avoided.

CPYPF - Copy Physical File(s)

This command allows you to copy data from Physical files in one library to Physical files in another library.

Command Syntax



Parameter Definitions

Where the file exists in the TOLIB, and data exists in the FROMLIB file, the data will be copied using FMTOPT(*MAP *DROP).

Where the file exists in the TOLIB, and no data exists in the FROMLIB file, the TOLIB file will be cleared using CLRPFM.

Where the file does not exist in the TOLIB, the file will be created in the TOLIB using CRTDUPOBJ DATA(*YES).

All members in the FROMFILE are copied to the TOFILE.

Where the member exists in the TOLIB file, the member will be copied using MBROPT(*REPLACE).

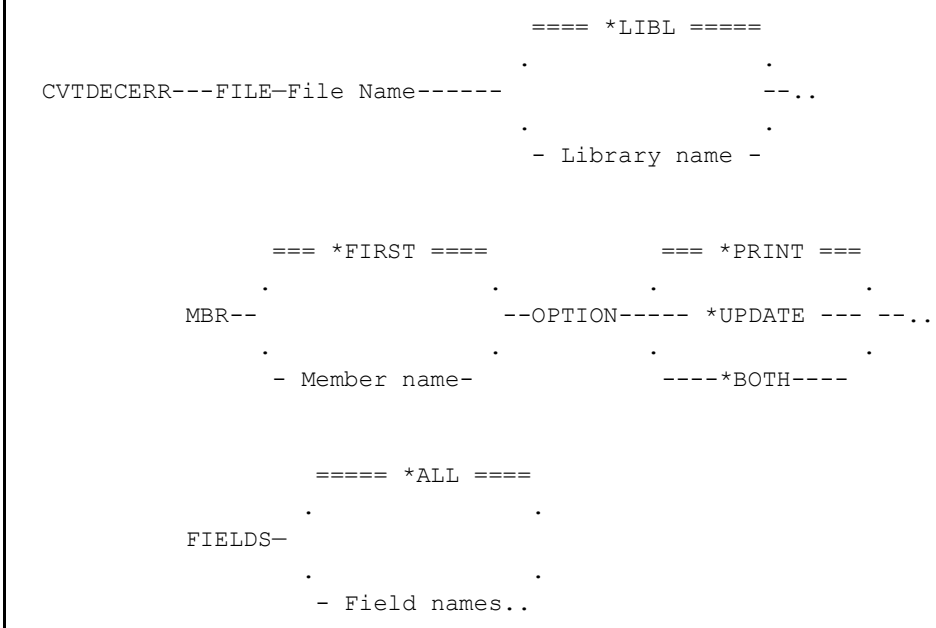
Where the member does not exist in the TOLIB file, the member will be copied using MBROPT(*ADD).

Source files can be included or excluded using the INCLSRC parameter. An entry of *YES will include source files in the copy, whereas an entry of *NO will ignore them.

CVTDECERR - Report/Convert Decimal Data Errors in a file

This command allows you to test the contents of numeric fields in a physical file in order to determine whether any decimal data errors exist in the data.

Command Syntax



Parameter Definitions

The OPTION parameter allows either

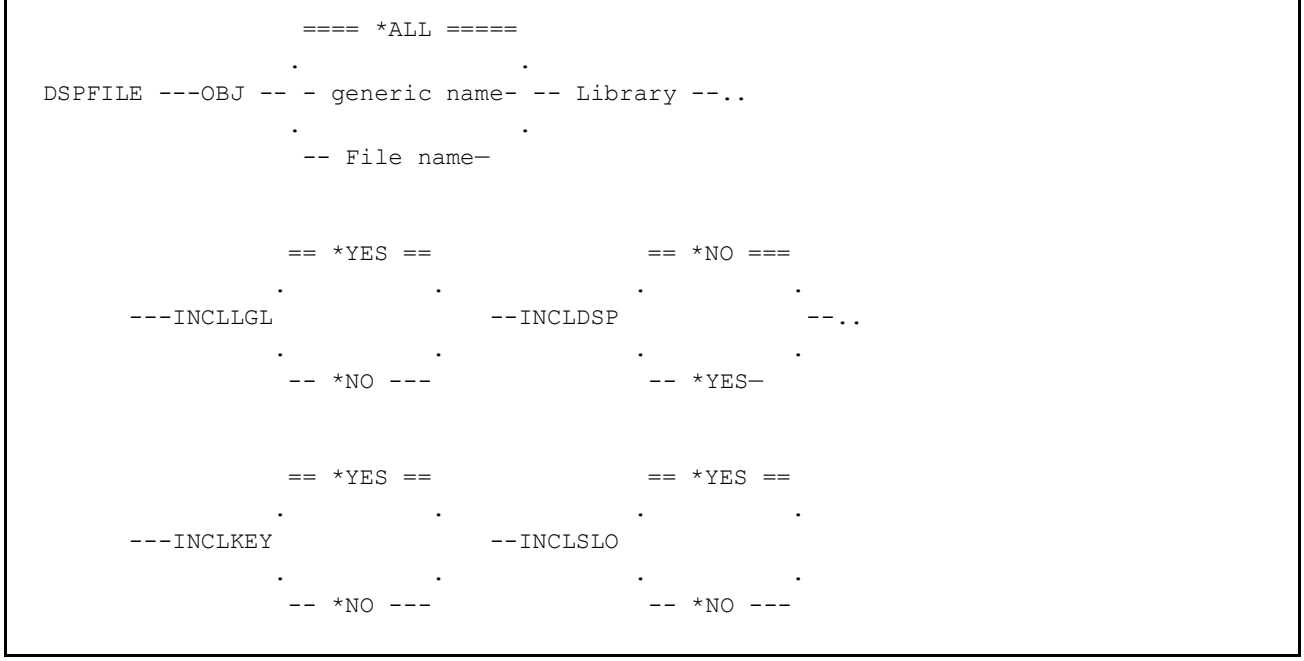
*PRINT - report any decimal data errors encountered
*UPDATE - correct any errors found, by resetting the field(s) in error to zero
*BOTH - reset decimal errors encountered and print any errors found

The FIELDS parameter allows you to specify which fields in the file are to be tested. If you do not specify any field names, all fields will be processed.

DSPFILE - Print a Data Description Record Layout

This command will print a formatted record format layout for a specified file. Options are available to print the key fields (where applicable) and to indicate if any fields (logical file) are used as select/omit fields.

Command Syntax



Parameter Definitions

The INCLLGL (Include logical files?) and INCLDSP (Include display file?) allows you to filter out object types not required.

The INCLKEY (Include Access Path Information) parameter allows you to specify whether to print key field information on the report.

The INCLSLO (Include Select/Omit Information) parameter allows you to specify whether to flag those fields that are used in a logical view for Select or Omit criteria.

DSPLIBSIZ - Display the size of a library

This command calculates the size of a library, based on the objects within the library.

It will only work correctly where the user has read authority on the objects within the library.

Command Syntax

```
DSPLIBSIZ ---- LIB (Library name)
```

The size of the library, and the total number of objects read, will be returned in a message (SIZ0001). The message is displayed as a completion message.

DSPOBJINF - Display Object Information

This command displays an informational panel, which provides you with the basic attributes of an object (in a similar way to DSPOBJD)

.

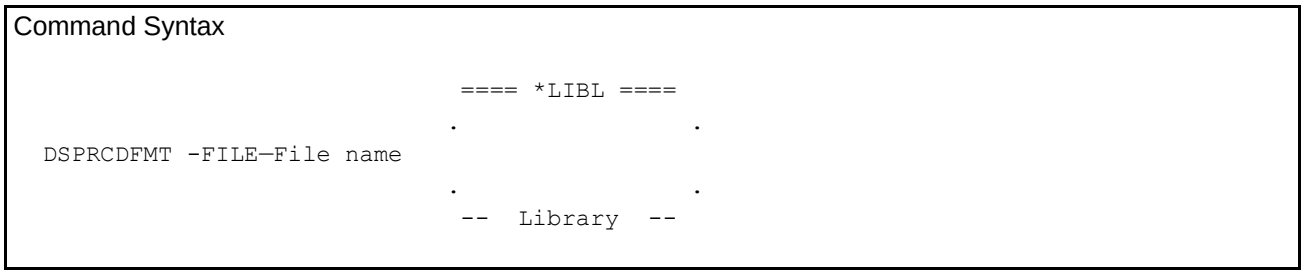
Command Syntax

```
                                ==== *libl =====
                               .                .
 DSPOBJINF -OBJ—Object name                    -OBJTYPE - Object type
                               .                .
                                - Library name -
```

DSPRCDFMT - Display File Record Format

This command displays file field information based on the file requested.

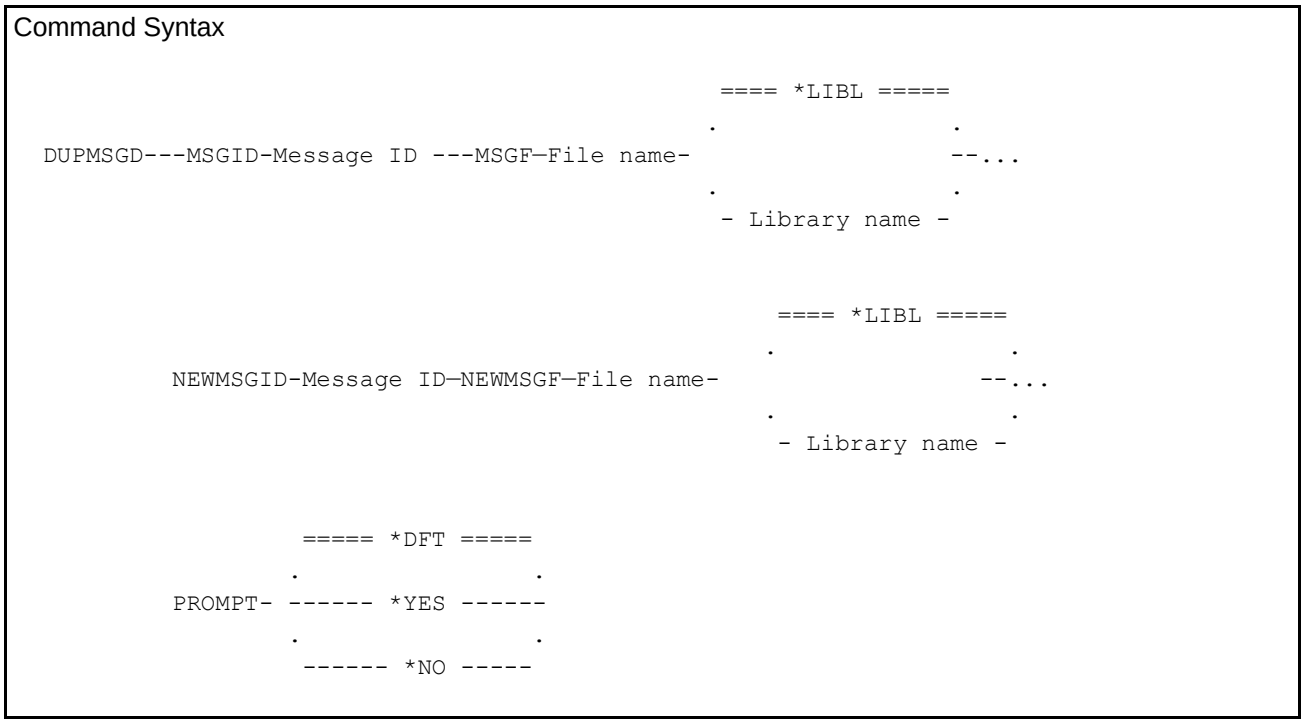
Command Syntax



DUPMSGD - Duplicate Message Description

This command allows you to duplicate a message file message, with the option to then modify the duplicated message.

Command Syntax



The advantage of using this over CHGMSGD is that the text of the existing message is displayed for you to make modifications to, rather than having to re-type the entire message text.

The PROMPT option will allow you to edit the new message if either

the parameter is *YES or
the parameter is *DFT and the command is run interactively.

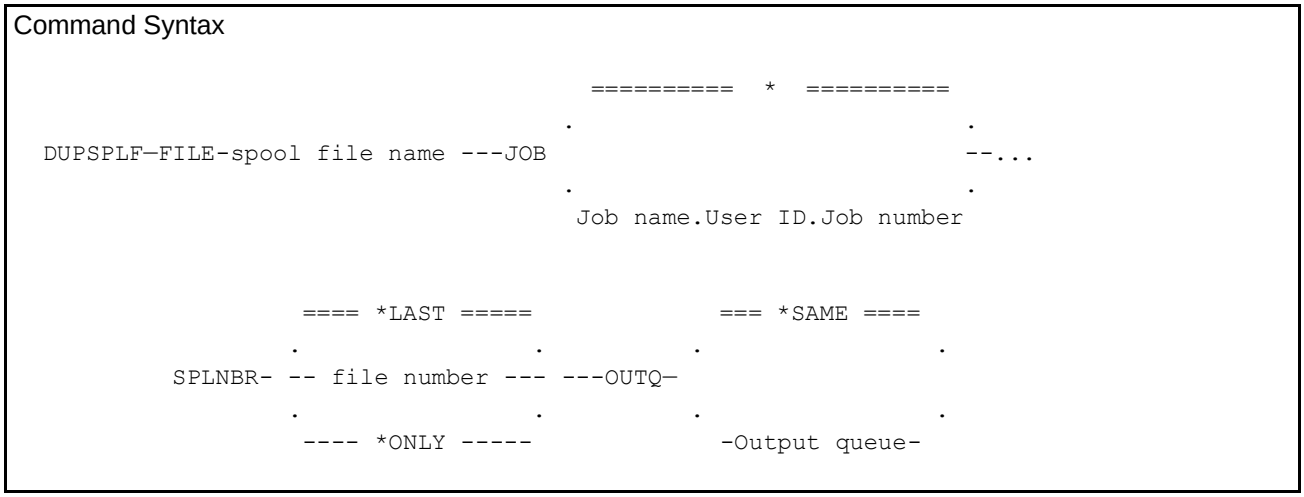
This command should not be used for messages which have substitution variables embedded in them. Embedded substitution variables will not be generated into the new message.

DUPSPLF - Duplicate spool file

This command allows you to duplicate a spool file. The spool file named is copied and placed on a named output queue. Where the job name is not specified, the current job is assumed.

*Graphics spool files and *ASCII output spool files are not supported by this command.*

Command Syntax



Parameter Definitions

Spooled file name (FILE): Specify the name of the spool file this command is to duplicate. This is a required parameter.

Job name (JOB): Specify the qualified name of the job that created the spooled file. The possible values are:

*	The job that issued this command is the job that created the spooled file.
job name	Specify the name of the job that created the spooled file.
user name	Specify the user name that identifies the user profile under which the job is run.
job number	Specify the system-assigned job number.

Spooled file number (SPLNBR): Specify the unique number of the spooled file in the job that is to be duplicated. The possible values are:

*LAST	The spooled file with the highest number and the specified file name is used.
*ONLY	Only one spooled file in the job has the specified file name; therefore, the number of the spooled file is not necessary.
spooled file number	Specify the number of the spooled file whose attributes are to be retrieved.

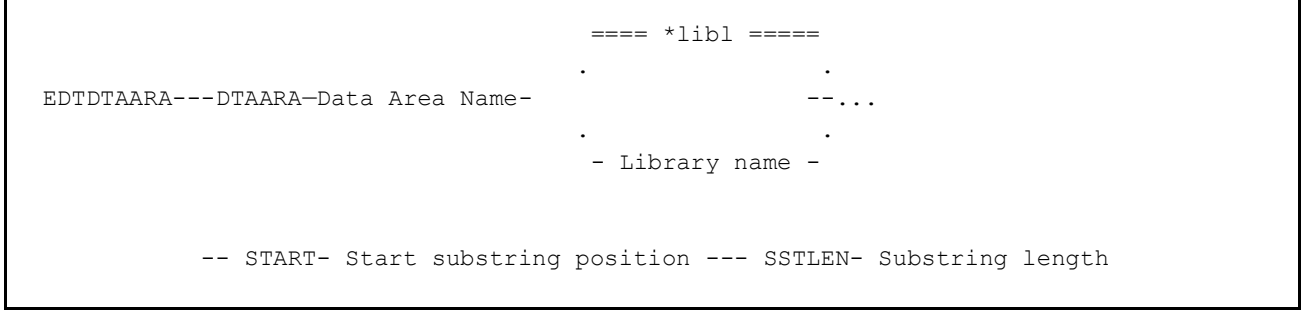
Output Queue (OUTQ): The Output Queue parameter specifies the name of the output queue where the duplicated spooled file is to be placed. The possible values for output queue name are:

*SAME	The duplicated spooled file will be placed in the same output queue where the original spooled file is located.
Output queue name	Specify the name of the output queue to be used.

EDTDTAARA - Edit Data Area contents

This command allows you to edit the contents of a Data Area.

Command Syntax



The advantage of using this over CHGDTAARA is that the information currently stored in the data area is displayed for you to make modifications to, rather than having to re-type the entire contents.

Because of a restriction in command prompting, only 512 characters of a data area can be maintained at any time (this is the reason for the START and SSTLEN parameters).

If you wish to maintain any data from positions 513 upwards, you must use these parameters to define where you wish to start the maintenance, as well as the number of characters to be maintained.

If *ANY* special character (anything other than letters and numbers) information is contained within the portion of the data area that is being maintained, then *ALL* information in that area will be displayed in hexadecimal format.

EDTPSTDFT - Edit PassThru Defaults

The EDTPSTDFT command allows you to maintain the default data stored for remote machines that is used by the NPASTHR command.

The command has no parameters

The first screen displayed will be a list of all locations that are currently defined. If no locations have been defined you will receive a maintenance screen to allow you to add locations:

```
NPA002D1            NPASTHR Connections - Maintenance
Select

Type options, press Enter.
  2=Change

   Remote                                      Connection
   System                                      Type
   LONDON    Head Office, London               Pass thru
   NEWYORK   Head Office, New York             Pass thru
   SBA01     Local SubArea network             Telnet
   PARIS     Head Office, Paris                Pass thru
   SYDNEY    Head Office, Sydney               Pass thru
   TOKYO     Head Office, Tokyo                Pass thru
   WASHTON   Head Office, Washington           Pass thru

Position list to:
   LONDON

F3=Exit   F6=Add
```

From this screen you can edit any of the existing entries by typing a ‘2’ beside the entry and pressing the enter key. To add new entries you press F6=Add.

The maintain entry screen then displays, allowing you to edit the entry:

```
NPA002D2            NPASTHR Connections - Maintenance
Change

       Remote System ID (Alias) . LONDON   Name
       Connection type  . . . . . N        N=Network Passthru, T=TELNET TCP
       Connection available?. . . Y        Y=Yes, N=No
       Description. . . . . . . . Head Office, London

For a PASS THROUGH connection:
Remote System Name . . . . S44CB098           Name, *SYSID, *CNNDEV
Virtual Controller . . . . VRTCTL01           Name, *NONE
Mode . . . . . BLANK              Name, *NETATR
Local location name. . . . *LOC               Name, *LOC, *NETATR
Remote Network Identifier. NAVNET             Name, *LOC, *NETATR, *NONE
APPC Device (default). . . *LOC               Name (for *CNNDEV), *LOC

For a TELNET TCP/IP connection:
Remote system name . . . .


                                              Name, *INTNETADR
Internet address . . . . .                    Format nnn.nnn.nnn.nnn

F3=Exit   F11=Delete   F12=Previous
```

This screen allows you to maintain information about a target passthrough location.

Remote System ID: This is the identifier that will be used by the NPASTHR command to retrieve the location information. It does not have to be the Remote System name; it can be an alias.

So, for example, the remote system name might be S52A0653 - you might make the ID ‘USA’. When the user accesses this machine they type NPASTHR USA.

> The ID field locates the information in the NPASTHR defaults file
> the Remote System name locates the target machine.

Connection Type: This flag determines the type of connection that will be made.

> An entry of ‘N’ signifies this is a network passthru, using the STRPASTHR command.
> An entry of ‘T’ signifies this is a Telnet passthru, using the STRTCPTELN command.

Connection Available?: This flag allows you to lock the location and stop NPASTHR using this entry.

> An entry of ‘Y’ (Yes) allows access.
> An entry of ‘N’ (No) will stop access via the NPASTHR command (the user will be issued a ‘not available’ escape message).

The parameters for a **Network Passthru** connection are:

Remote system name: This is the name of the remote location that will be the target of a passthrough session. Special values are:

*SYSID - The system name is the same as the system identifier (the previous field on the screen)
*CNNDEV - Use the APPC devices specified with the APPC device parameter

Description: Descriptive text that helps identify the target location. This is displayed by the NPASTHR command when an attempt is made to access the system.

Virtual Controller: This is the name of the virtual controller on the remote system that is used to do passthrough jobs. If you specify *NONE, this means that no controller is specified.

Mode: This specifies the mode name that will be used. If you specify *NETATR, this means that the mode specified on the network attributes will be used.

Remote Network Identifier: This is the network ID of the network where the remote location resides. The following special values are allowed:

*LOC - Any remote network ID may be used.
*NETATR - The local network ID specified on the network attributes is used.
*NONE - The remote location does not support network identifiers.

APPC Device: Specifies the name of the device description that will be used to do the passthrough. If you specified *CNNDEV as the Remote System Name, you must enter a connection device here. If you specify *LOC, the Remote System Name specifies the name to be used.

The parameters for a **TCP Telnet** connection are:

The RMTSYS Remote System parameter allows you to specify the name of the remote TELNET server system to connect to. You can specify the remote system name or *INTNETADR (use the internet address specified in the INTNETADR parameter).

The INTNETADR Internet Address parameter allows you to specify the internet address to connect to. An internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255.

To cancel the EDTPSTDFT command, press F3.
To cancel the editing of this location, press F12.
To delete this location, press F11.

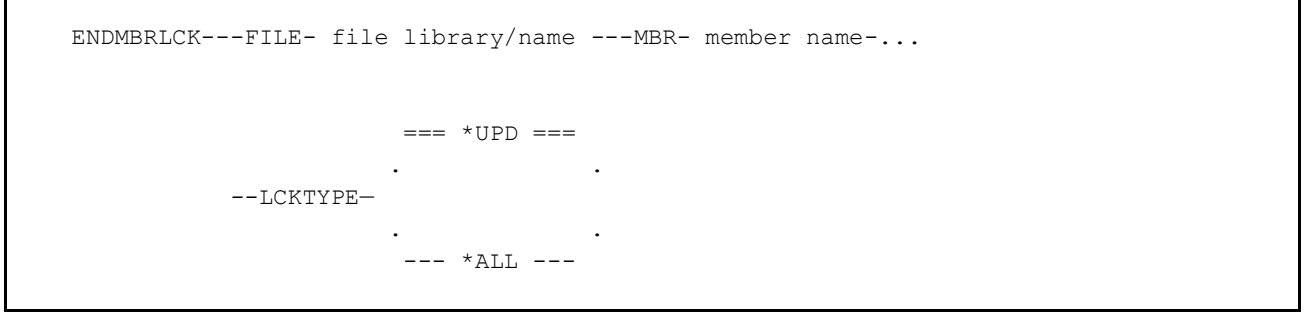
ENDMBRLCK - End jobs holding member locks

This command will end any jobs that currently have a member lock on the specified file/member. The command is especially useful where you are trying to obtain an exclusive lock on a file, but are unable to.

Any jobs found to have a lock on the file are ended (controlled cancel) and an audit report lists those jobs ended by this command.

This command has been replaced by the NPRCOBJLCK command, defined later in this section of the manual. The ENDMBRLCK command will be retained for compatibility purposes, but your future development should be directed towards using the new NPRCOBJLCK command.

Command Syntax



Parameter Definitions

The LCKTYPE Lock Type To End parameter allows you to specify whether all jobs accessing the file should be ended, or only those jobs holding update locks on the file/member specified.

MRGLIB - Merge Libraries

This command allows you to merge the contents of two libraries into one library.

Command Syntax

```
          MRGLIB ---FROMLIB (From Library) ---TOLIB (To library)-...

                    --WORKLIB (work library to be created)
```

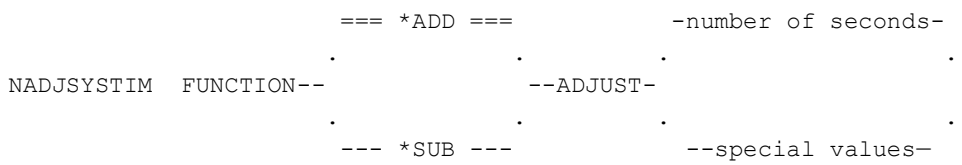
Where a duplicate of an object already exists in the TO library you have the option of either specifying a work library (in which case the TO library object is moved there first) or replacing the object (by not specifying a work library).

NADJSYSTIM - Adjust system time

The NADJSYSTIM command allows you to adjust the time on the system clock by a specified number of seconds.

The command does not allow you to adjust the time if the adjustment value causes the system date to change.

Command Syntax



Parameter Definitions

FUNCTION specifies whether you want to *ADD or *SUBtract the specified time adjustment value.

The ADJUST value allows you to define the number of seconds for the adjustment. The special values for this parameter are as follows:

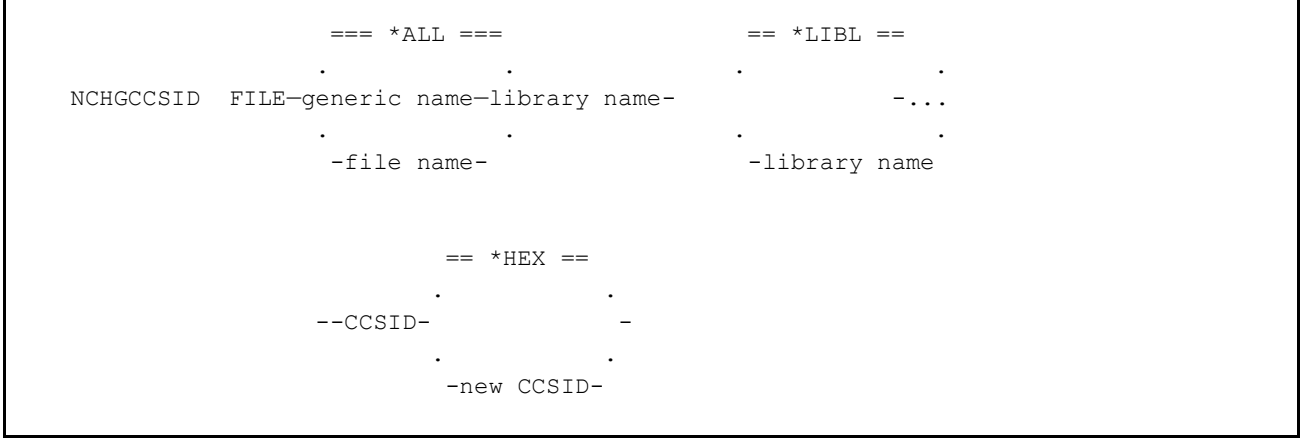
**MN15*	The adjustment is 15 minutes
**MN30*	The adjustment is 30 minutes
**MN45*	The adjustment is 45 minutes
**HR1*	The adjustment is 1 hour
**HR2*	The adjustment is 2 hours
**HR3*	The adjustment is 3 hours
**HR4*	The adjustment is 4 hours
**HR5*	The adjustment is 5 hours

You can only adjust the time within the current date. If the specified adjustment would cause the time to roll in to the next (or the previous) date, the time will not be changed and an exception message will be sent.

NCHGCCSID - Change physical file CCSID

The NCHGCCSID command allows you to alter the CCSID of a specified file, or group of files.

Command Syntax



Parameter Definitions

The FILE parameter allows you to specify a single file name to process, or a generic name to process a group of file, or *ALL to process all files in the specified library name.

The CCSID parameter specifies the CCSID that should be applied to the file name specified. This must be a valid Coded Character Set identifier as defined in the Database Reference guide.

The default for this parameter is *HEX, which specifies that CCSID 65535 is used. This indicates that character data in the file is treated as bit data and is not subject to CCSID conversion.

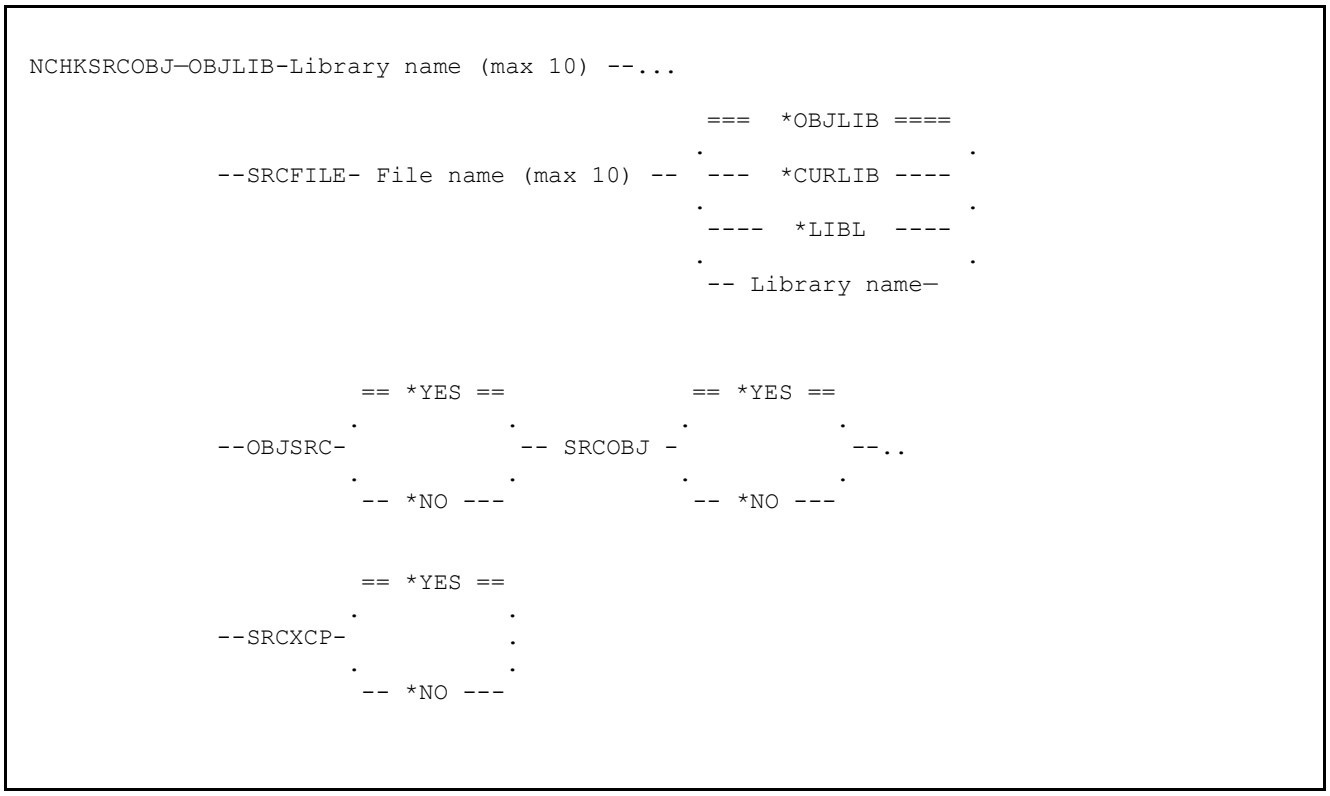
NCHKSRCOBJ - Check source/object compatibility

This command allows you to perform an audit check on your application libraries to determine any problems that may exist between your source and objects. Such inconsistencies include:

- Source not found for object
- Object not found for source
- Source change date incompatible with object creation date

Reports will be printed showing any possible problems detected by the command.

Command Syntax



Parameter Definitions

OBJLIB: Specify the name of the library containing the objects you want to audit. You can enter up to 10 object library names to be included in the audit. This is a required parameter and requires at least one object library name to be entered.

SRCFILE: Specify the name of the source file containing the source definition for the objects you are auditing. You can enter up to 10 source files to be included in the audit.

The possible library values are:

*OBJLIB: The source file is in the same library named in the object library parameter.
*LIBL: The library list is used to locate the source file
*CURLIB: The source file is in the current library. If no current library is defined for this job, QGPL is assumed.
Library name: Specify the name of the library where the source file resides.

The three optional parameters allow you to specify which of the comparison reports are to be printed:

OBJSRC (Print object/source exceptions): This report may include objects which were created from source files not specified in the SRCFILE parameter of the command; objects that weren't created from source; objects which have had the source attributes removed from the object definition.

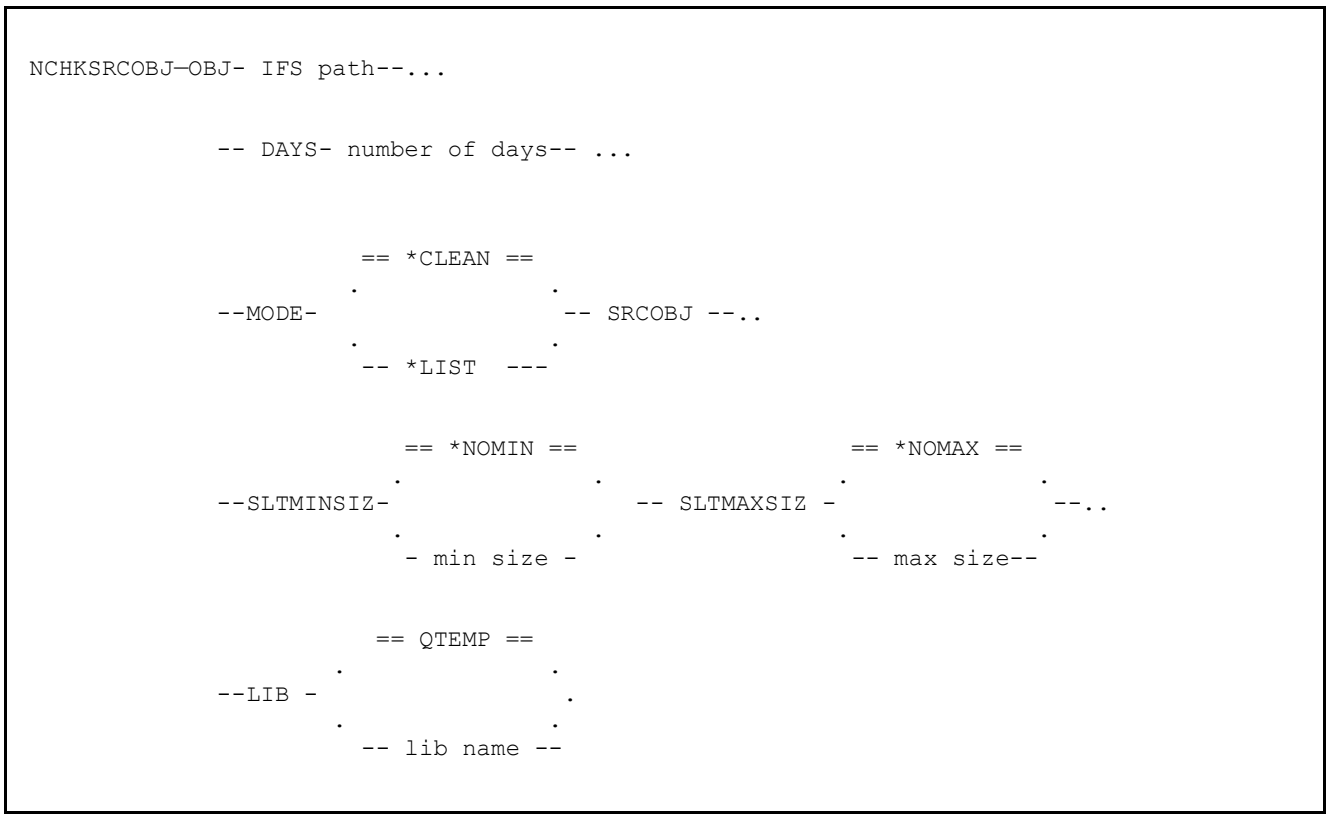
SRCOBJ (Print source/object exceptions): Specifies whether the processing should print a list of all source members for which a corresponding object could not be found in any of the object libraries specified.

SRCXCP (Print Source exceptions): Specifies whether the processing should print a list of objects that may have been created from source that is different from the current source. This is determined by comparing the source timestamp on the object to the 'last changed' timestamp on the actual source member.

NCLNIFSDIR – Cleanse IFS directory

The NCLNIFSDIR command allows you remove stream files from a specified IFS directory, based on object age and size. Results are logged in an outfile for review.

Command Syntax



Parameter Definitions

OBJ: Specifies which path name (or directory) is to be processed.

DAYS: The num number of days entered in this parameter will be used to calculate a cleanse date. You may enter a number of days between 0 and 365, where 0 is today, 1 is yesterday, and so on. Stream files older than the calculated date will be processed.

MODE: The Processing Mode defines what the command will do with IFS stream files meeting the selection criteria:

*CLEAN - Stream files meeting the selection criteria will be deleted. The log outfile will list all stream files removed.

*LIST - Stream files meeting the selection criteria will be listed in the log outfile only. they will not be deleted.

SLTMINSIZ: Specifies the lower limit of the object size range (in KB) for the stream file selection. Objects under the minimum size specified will be ignored by the command.

*NOMIN - There is no lower size limit on the selection range.

minimum object size - Specify the lower limit (in KB) of the object size range for the stream file selection

SLTMAXSIZ: Specifies the upper limit of the object size range (in KB) for the stream file selection. Objects exceeding the maximum specified will be ignored by the command.

The possible values are:

*NOMAX - There is no upper size limit on the selection range.

maximum object size - Specify the upper limit (in KB) of the object size range for the stream file selection.

LIB: The *OUTFILE library parameter specifies which library the NCIFSLOG log outfile should be placed in. If the file already exists in the specified library, it will be replaced.

NCLNJRNRCV – Cleanse library journal receivers

The Cleanse Library Journal Receivers utility command allows the processing of older Journal Receivers in a specified library.

Parameter Definitions

Library name (LIB): Specifies the name of the library to be processed. This is a required parameter.

Number of days to keep (CLNDAYS): The value you enter in this parameter will be used to calculate a cleanse date. This is a required parameter.

You may enter a number of days between 0 and 365, where 0 is today, 1 is yesterday, and so on. Journal Receivers older than the calculated date will be processed for removal.

NCPYDBFIFS – Copy Database file to the IFS

The Copy Database File to IFS (NCPYDBFIFS) command copies a database file member to an IFS stream file. Optional conversion of the data and reformatting is performed when copying a database file member.

Any overrides in effect for the database file member are not used by this command.

Parameter Definitions

From file (FROMFILE): Specifies the database file that contains the records to be copied. The file must be either a source physical file that has fields or a program described file. Externally described files and source physical files containing fewer or more than three fields are not permitted by this command.

This is a required parameter.

From member (FROMMBR): Specifies the database file member in the from-file that is to be copied.

To stream file (TOSTMF): Specifies the path name of the stream file to which data is copied. All directories in the path name must exist. New directories are not created. If the stream file does not exist, it is created.

Stream file option (STMFOPT): Specifies whether the copy operation replaces, adds, or fails to copy the records in a stream file if a stream file with the specified name already exists. If the stream file does not exist, it is created:

*NONE - No records are copied and the operation will fail.

*ADD - The records are added to the end of the existing stream file.

*REPLACE - The records replace the existing stream file records.

Data conversion options (CVTDTA): Specifies the process for converting the data from the database file member to the stream file:

*AUTO - The data is converted during the copy operation using the coded character set identifier (CCSID) of the stream file and the database file CCSID.

*TBL - The data is converted using a conversion table. Only single-byte character sets are supported. The conversion table must be specified by the Conversion table (TBL) parameter. If a conversion table is not available, the operation will fail.

*NONE - Only the removal of the sequence numbers and date stamp from source physical files and the optional insertion of specified line-formatting characters into the stream file are performed Database file CCSID to stream file CCSID conversion of other characters is not performed.

Database file CCSID (DBFCCSID): Specifies the method of obtaining the database file CCSID:

*FILE - The database file CCSID is used, unless it is 65535. If the database file CCSID is 65535, and the file is not a program-described file, the operation will fail. If the database file CCSID is 65535, and the file is a program-described file, the default job CCSID is used.

1-65533 - Specify the database file coded character set identifier (CCSID).

Stream file CCSID (STMFCCSID): Specifies the method of obtaining the stream file coded character set identifier (CCSID) used for data conversion:

*STMF - If the stream file exists, and data conversion is requested, the CCSID associated with the stream file is used to perform the conversion. If the stream file does not exist, the source database file CCSID is used and associated with the stream file.

*STDASCII - A CCSID in the IBM PC Data encoding scheme (x2100) is computed. If the stream file exists, and the stream file CCSID is not the same as the computed value, the operation will fail. If the stream file does not exist, the computed CCSID is associated with the target stream file and is used for data conversion if it is requested.

*PCASCII - A CCSID in the Microsoft Windows encoding scheme (x4105) is computed. ("Microsoft" and "Windows" are registered trademarks of Microsoft Corporation). If the stream file exists, and the stream file CCSID is not the same as the computed value, the operation will fail. If the stream file does not exist, the computed CCSID is associated with the target stream file and is used for data conversion if it is requested. This option allows the resulting data to be used by Microsoft Windows applications.

1-65533 - Specify the CCSID used. If the stream file exists, this option is only valid if the CCSID associated with the stream file is the same as the specified value. Otherwise, the operation will fail. If the stream file does not exist, the specified CCSID is associated with the stream file when it is created.

Conversion table (TBL): Specifies the path name of the conversion table used to convert data from the database file member to the stream file.

Note: This parameter is required and valid only if CVTDTA(*TBL) is specified.

End of line characters (ENDLINFMT): Specifies the end-of-line characters to insert into the stream file during the copying of records. If one of the end-of-line character options is selected (ENDLINFMT(*FIXED) is not specified) the database file records are transformed to variable-length stream file text lines as they are copied. Each database file record is trimmed of any trailing blanks. Then, the data is converted to the destination data format (if specified) and the end-of-line character is appended to the end of the text line. The text line is copied to the stream file.

*CRLF - Carriage-return followed by line-feed is appended to the end of each line.

*LF - Line-feed is appended to the end of each line.

*LFCR - Line-feed followed by carriage-return is appended to the end of each line.

*FIXED - The lines in the stream file are written as fixed length records. CR and LF characters are not added at the end of each line, trailing blanks are not removed from the end of each record. The length of the stream file records equals the length of the database file records. If CVTDTA(*AUTO) is specified, the converted data will not be allowed to contract or expand. Therefore, the encoding schemes of the stream file CCSID and database file CCSID must be compatible. For example, if a stream file had a single-byte encoding scheme and the database file had a double-byte encoding scheme, the command will fail.

Authority (AUT): Specifies the method used to assign authority information to the stream file.

This parameter is ignored if the stream file already exists.

*DFT - The owner of the stream file will be granted *RWX data authority to the stream file. The primary group and *PUBLIC will have *NONE data authority to the stream file. Object authorities will be based on the object authorities for the directory where the stream file is to be created. The auditing value of the database file will be copied to the stream file.

*INDIR - The authority information for the stream file is based on the authority for the directory where the stream file is to be created. The stream file is assigned the same public authority, private authorities, primary group, primary group authority, and authorization list as the directory in which it is created. The auditing value assigned to the stream file is controlled by the directory's create object auditing value. If the target file system does not support the *INDIR special value, the command will fail.

*FILE - The authority information for the stream file is based on the authority for the object specified on the From file member or save file (FROMMBR) parameter. The stream file is assigned the same public authority, private authorities, primary group, primary group authority, authorization list, and auditing value as the member or save file being copied. If the target file system does not support one or more of these values, the unsupported values will be ignored. each line.

*INDIRFILE - The authority information for copied objects is initially based on the authority for the directory where the objects are to be created. Then, authority information from the object specified on the FROMMBR parameter will be copied to the target object. The stream file is assigned the same public authority, private authorities, primary group, primary group authority, authorization list, and auditing value as the member or save file being copied, as well as any additional private authorities obtained from the directory. The resulting authority information will be similar to that produced by copying and pasting objects using the System i Navigator. If the target file system does not support the *INDIRFILE special value, the command will fail.

NCPYIFSDBF – Copy IFS Stream file to a Database file

The Copy IFS Stream file to Database file (NCPYIFSDBF) command copies an IFS stream file to a Database file. Optional conversion of the data and reformatting is performed when copying a database file member.

Any overrides in effect for the database file member are not used by this command.

Parameter Definitions

From stream file (FROMSTMF): Specifies the path name of the stream file from which data is copied. All directories in the path name must exist.

To file (TOFILE): Specifies the database file to which data is copied. The file must be either a source physical file that has fields or a program described file.
This is a required parameter.

To member (TOMBR): Specifies the database file member in the to-file to which data is to be copied.

Member option (MBROPT): Specifies whether the copy operation replaces, adds, or fails to copy the records if the object specified on the To file member (TOMBR) parameter already exists.

> *NONE - No records are copied and the operation will fail if the object exists.
> *ADD - The copied records are added to the end of the existing records.
> *REPLACE - The copied records replace the existing records.

Data conversion options (CVTDTA): Specifies the process for converting the data from the stream file to the database file member.

> *AUTO - The data is converted during the copy operation using the coded character set identifier (CCSID) of the stream file and the database file CCSID.
> *TBL - The data is converted using a conversion table. Only single-byte character sets are supported. The conversion table must be specified on the Conversion table (TBL) parameter.
> *NONE - Only insertion of the sequence numbers and date stamp to source physical files and the optional removal of specified line-formatting characters from the stream file are performed. Stream file CCSID to database file CCSID conversion of other characters is not performed.

Stream file CCSID (STMFCCSID): Specifies the method of obtaining the stream file coded character set identifier (CCSID) used for data conversion.

> *STMF - The CCSID associated with the stream file specified on the From stream file (FROMSTMF) parameter is used for data conversion.
> *PCASCII - Use the CCSID associated with the stream file specified on the From stream file (FROMSTMF) parameter to compute a CCSID in the Microsoft Windows encoding scheme (x4105). This CCSID is used if data conversion is requested. This option allows data to be converted properly if the data was created using Microsoft Windows. ("Microsoft" and "Windows" are registered trademarks of Microsoft Corporation). For example, if the CCSID associated with the stream file is 37, the stream file data is instead assumed to be in CCSID 1252 for data conversion.
> 1-65533 - Specify the stream file CCSID to be used for data conversion.

Database file CCSID (DBFCCSID): Specifies the method of obtaining the database file CCSID used for data conversion.

> *FILE - The database file CCSID is used, unless it is 65535. If the database file CCSID is 65535, and the file is not a program-described file, the operation will fail. If the database file CCSID is 65535, and the file is a program-described file, the default job CCSID is used.
> 1-65533 - Specify the database file coded character set identifier (CCSID). This option is valid only if the database file CCSID is 65535 or is the same as the CCSID specified. Otherwise, the operation will fail.

Conversion table (TBL): Specifies the path name of the conversion table to be used to convert data from the stream file to the database file member

Note: This parameter is required and valid only if CVTDTA(*TBL) is specified.

End of line characters (ENDLINFMT): Specifies the end-of-line characters which are recognized in the stream file during copying of records. All records are transformed to a fixed-length format as they are copied to the database file. The fixed-length is equal to the length of the database file records to which they are copied. If one of the end-of-line character options is selected (ENDLINFMT(*FIXED) is not specified) the stream file is read up to first occurrence of that character. The end-of-line character is stripped from the record. The remainder of the record is padded with blanks. The data is converted to the destination data format (if specified) and copied to the database file member. If a record is too long to fit in the fixed-length format, it is truncated, then copied.

*ALL - Any single or double character combination of carriage-return and line-feed is appended to the end of each line.
*CRLF - Carriage-return followed by line-feed is appended to the end of each line.
*LF - Line-feed is appended to the end of each line.
*LFCR - Line-feed followed by carriage-return is appended to the end of each line.
*FIXED - Text lines in the stream file are considered fixed-length records of the same length as the database file records to which they are to be copied. Any encountered CR, LF, or EOF characters are not stripped from the stream file. Tab expansion is not allowed and the Tab character expansion (TABEXPN) parameter is not valid. If the last record in the stream file does not fill the database file record, that record is padded with blanks.
If CVTDTA(*AUTO) is specified, the converted data will not be allowed to contract or expand. Therefore, the encoding schemes of the stream file CCSID and database file CCSID must be compatible. For example, if a stream file had a single-byte encoding scheme and the database file had a double-byte encoding scheme, the command will fail.

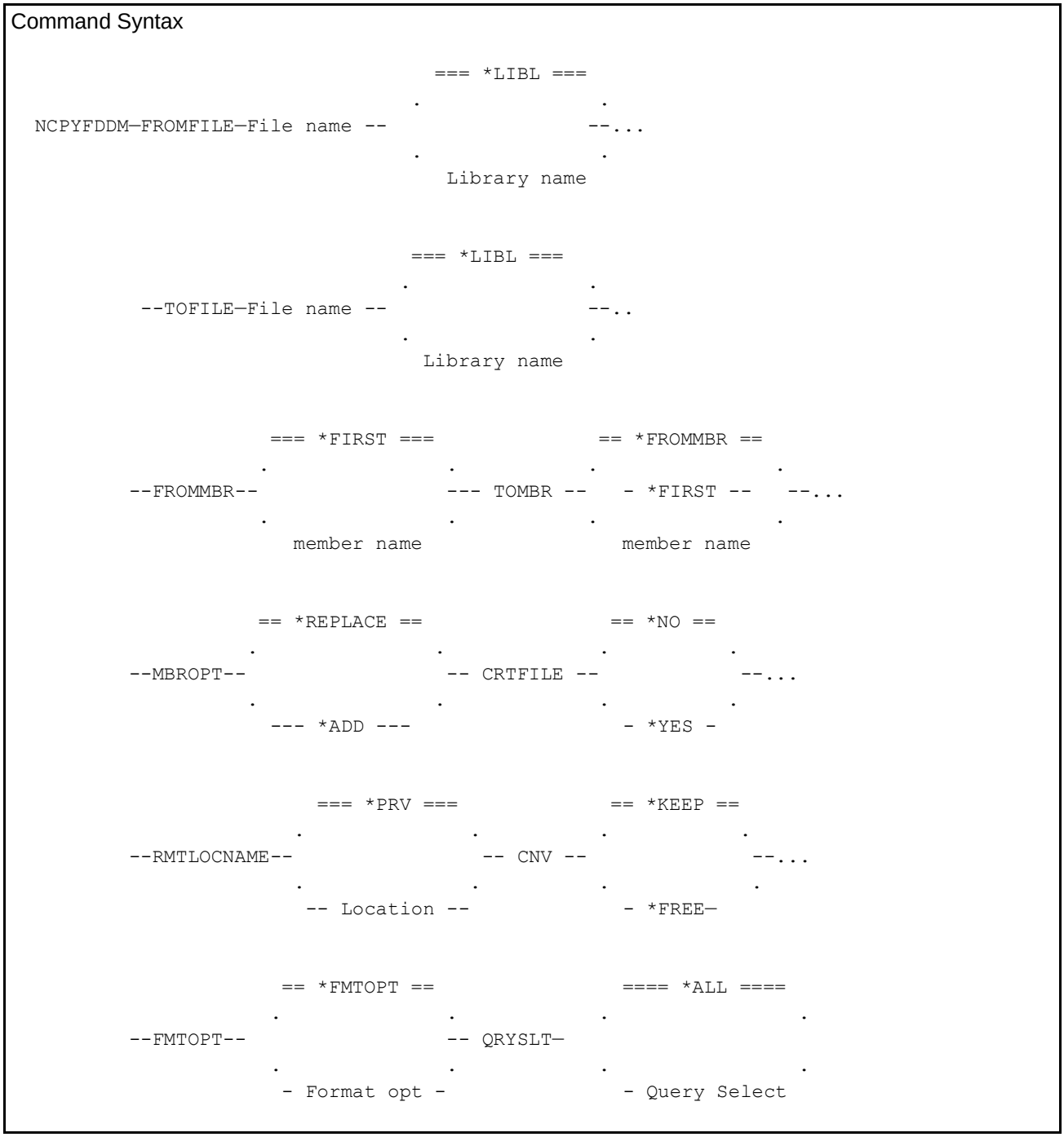
Tab character expansion (TABEXPN): Specifies whether embedded tab characters are expanded to blanks up to the next eight-character tab position.

*YES - Tab characters are not copied to the database file member. Any encountered tab character is expanded with blanks up to the next tab position.
Note: If ENDLINFMT(*FIXED) is specified, *YES is not a valid value for this parameter.
*NO - Tab characters are copied to the database file member. No tab expansion occurs.

NCPYFDDM - Copy a file using DDM

The NCPYFDDM command allows you to copy a file from this machine (the source machine) to a remote machine (the target machine).

Command Syntax



Parameter Definitions

FROMFILE: allows you to specify the name and library of the file (on the source system) to be sent. Any overrides currently in force against the specified file are ignored.

TOFILE: allows you to specify the name and library of the file to which records are copied, on the remote system.

CRTFILE: specifies whether a physical file is created (on the target system) if the specified to-file does not exist.

RMTLOCNAME: specifies the remote location name that is to be the target of this file copy. A special value of *PRV allows you to use the remote location name used on the last NCPYFDDM attempt (this is only valid if the command has been previously used in this session).

KEEP DDM conversation: allows you to leave the communications connection open to the target system after this command has completed. The advantage of this is when you are performing multiple requests of this command, the communication between the two machines is already established (thus making additional requests much faster).

*FREE - The connection to the target machine is dropped after completion of the processing of this command.

*KEEP - The connection to the target machine remains active after successful completion of the processing of this command.

Connection will be dropped, however, if a RCLRSC (Reclaim Resource) command is issued for this program invocation level.

FMTOPT: specifies what field-level record format processing (if any) is done.

*NONE - No field mapping or dropping is done.

*NOCHK - If the record formats of the files are different, the copy operation will continue despite the differences - record data is copied directly (left to right) from one file to the other.

*CVTSRC - This value is used when the *from* and *to* files are different types; one is a source file and the other is a data file.

*MAP - Fields with the same name in the from-file and the to-file record formats are copied, and any fields in the to-file that do not exist in the from-file format are set to their default value. Mapped fields may have different starting positions in the from-file and to-file record formats.

*DROP - This value must be used if any of the field names in the from-file record format do not exist in the to-file record format.

QRYSLT: allows you to send a subset of the records in the file, the subset being selection by the use of an Open Query File filter. Refer to the OPNQRYF command in the IBM iSeries CL Reference Guide for further details of this parameter.

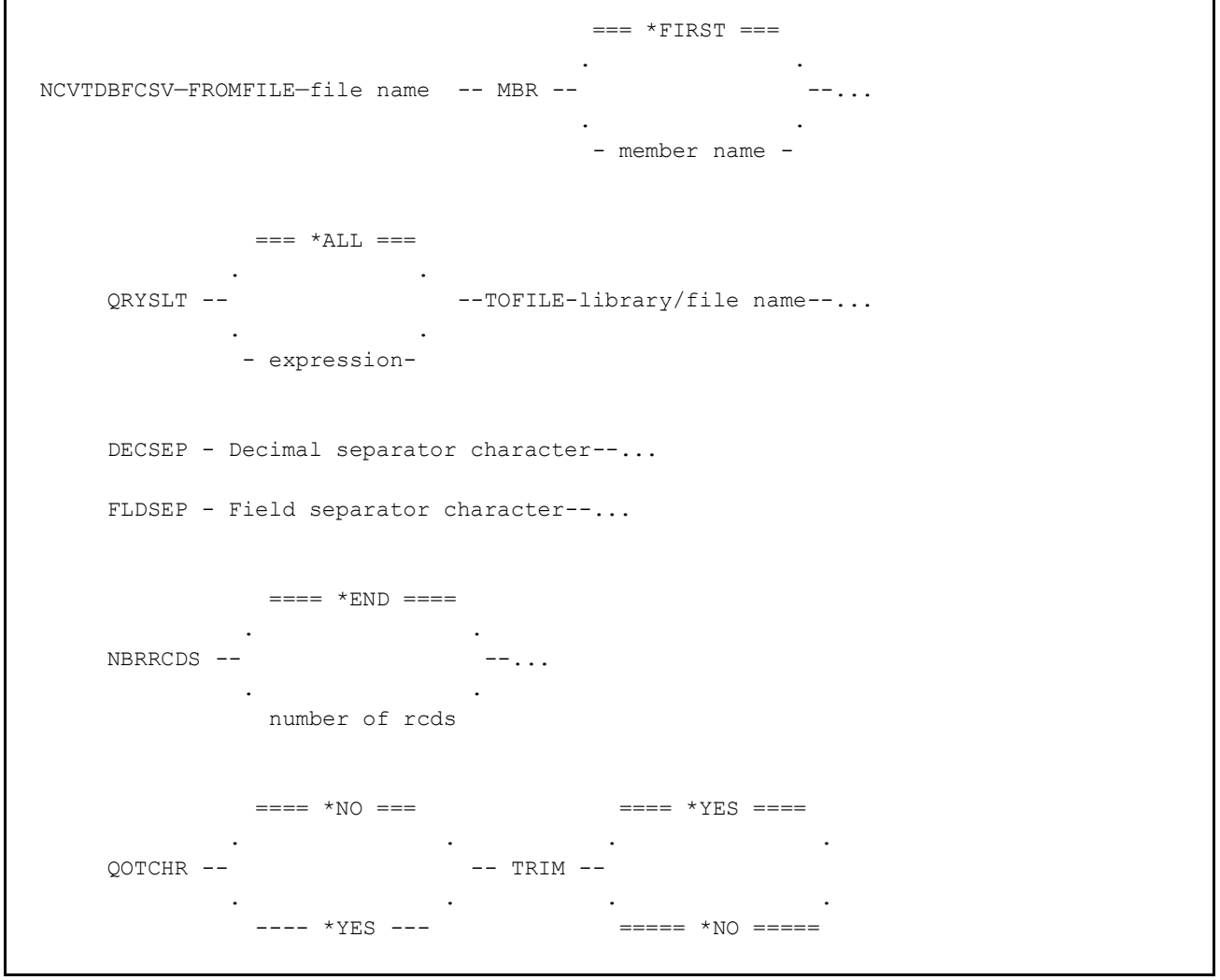
NCVTDBFCSV - Convert database file to CSV data file

The NCVTDBFPCD command allows you to convert a DB2/400 database file into a CSV (Comma Separated Variable) type data file, suitable for uploading to a non-iSeries system.

Optional extraction selection criteria in the form of an Open Query File query select (QRYSLT) statement can be used and is subject to the same syntax rules as that parameter. Refer to the IBM iSeries CL Reference Guide for a full discussion of the OPNQRYF (Open Query File) command.

The command can only process physical files, with a maximum record length of 8192 bytes.

Command Syntax



Parameter Definitions

FROMFILE: specifies the name and library of the database file to process. This is a required parameter.

MBR: allows you to specify the name of the file member to be processed.

QRYSLT: specifies the selection values used to determine the records that are to be copied.

*ALL - No record selection is to be performed. All records in the file are available for copying.

Query selection - Specify an expression (contained in apostrophes) that describes the values used to determine which records are selected. You can specify any logical expression formed from relationships (such as *EQ and *NE) of field and constant values or functions of field and constant values. At least one field name must be specified in each relationship.

TOFILE: Specifies the name and library of the CSV data file to create. The file name must not already exist in the specified library. The member name created in this file will be the same as the file name specified.

DECSEP: Specifies the character to be used as the decimal separator in a numeric field that contains decimal positions.

'.'	A full-stop character is used as the decimal separator.
*ASTERISK	An asterisk character '*' is used as the decimal separator.
*COLON	A colon character ':' is used as the decimal separator.
*COMMA	A comma character ',' is used as the decimal separator.
*DOT	A dot character '.' is used as the decimal separator.
*PERIOD	A period character '.' is used as the decimal separator.
*SEMICOLON	A semicolon character ';' is used as the decimal separator.
*SLASH	A slash character '/' is used as the decimal separator.
*STOP	A stop character '.' is used as the decimal separator.
*TAB	A tab character x'05' is used as the decimal separator.

decimal separator character - Specify the character to use as a decimal separator. The character used cannot be the same as the character defined as the field separator character (in parameter FLDSEP).

FLDSEP: Specifies the character to be used as the field separator, or delimiter. This is used by programs to determine how the items in the file are separated.

','	A comma character is used as the field separator.
*ASTERISK	An asterisk character '*' is used as the field separator.
*COLON	A colon character ':' is used as the field separator.
*COMMA	A comma character ',' is used as the field separator.
*DOT	A dot character '.' is used as the field separator.
*PERIOD	A period character '.' is used as the field separator.
*SEMICOLON	A semicolon character ';' is used as the field separator.
*SLASH	A slash character '/' is used as the field separator.
*STOP	A stop character '.' is used as the field separator.
*TAB	A tab character x'05' is used as the field separator.
*CARAT	A carat character '^' is used as the field separator.

field separator character - Specify the character to use as the field separator. The character used cannot be the same as the character defined as the decimal separator character (in parameter DECSEP).

NBRRCDS: specifies the maximum number of records to be copied.

*END - All records in the from-file are copied, dependent upon the selection criteria (QRYSLT parameter).

Number of records - Specify the maximum number of records that are to be copied.

QOTCHR: specifies whether any character fields in the file should be enclosed in double quotation marks, as required by some non-iSeries systems.

*NO - Character fields within the file will not be enclosed within double quotation marks.

*YES - All Character fields within the file will be enclosed within double quotation marks.

TRIM: Specifies whether any trailing blanks in character fields should be trimmed (removed).

*YES - Any trailing blanks detected in character fields will be trimmed.

*NO - Trailing blanks will be left in the character fields.

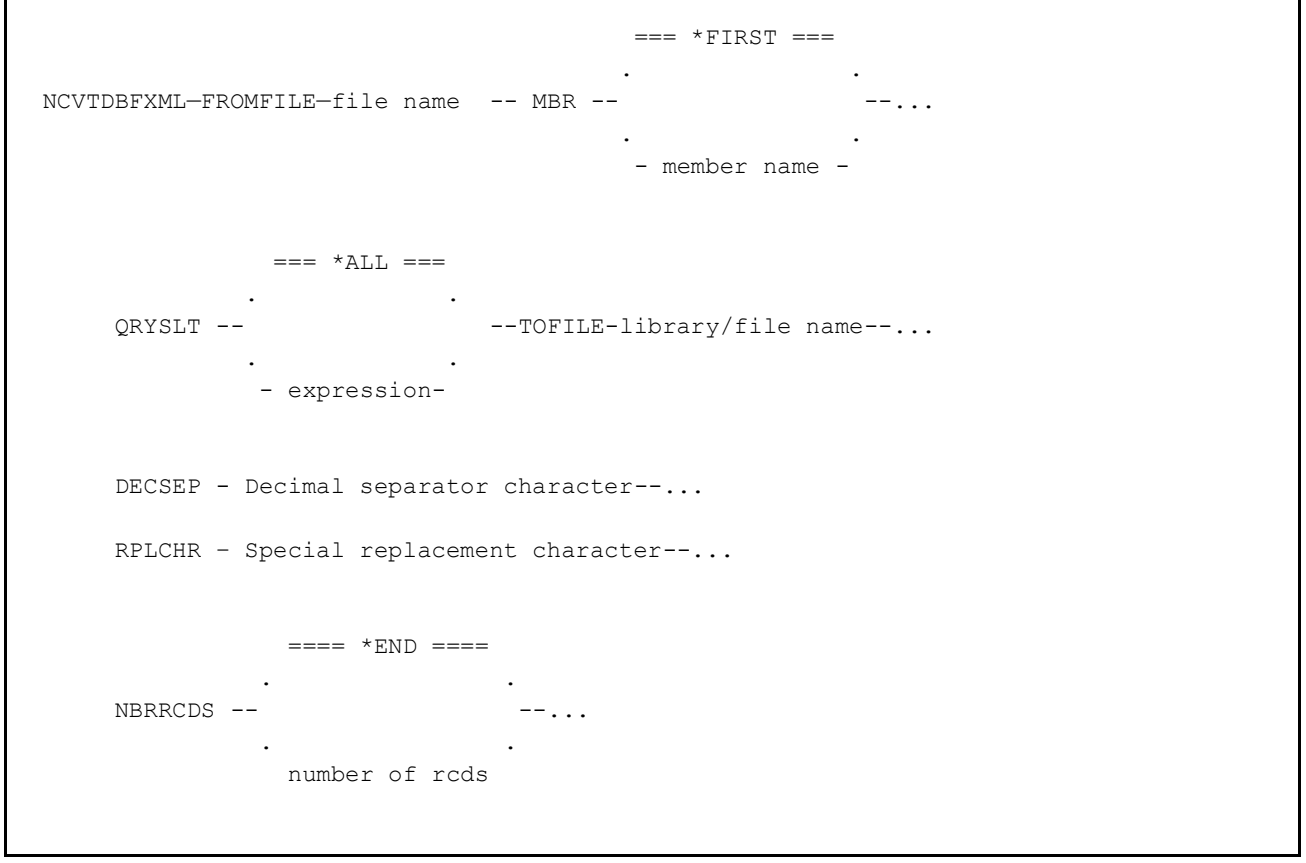
NCVTDBFXML - Convert database file to XML data file

The NCVTDBFXML command allows you to convert a DB2/400 database file into an XML type data file, suitable for uploading to a non-iSeries system.

Optional extraction selection criteria in the form of an Open Query File query select (QRYSLT) statement can be used and is subject to the same syntax rules as that parameter. Refer to the IBM iSeries CL Reference Guide for a full discussion of the OPNQRYF (Open Query File) command.

The command can only process physical files, with a maximum record length of 8192 bytes.

Command Syntax



Parameter Definitions

FROMFILE: specifies the name and library of the database file to process. This is a required parameter.

MBR: allows you to specify the name of the file member to be processed.

QRYSLT: specifies the selection values used to determine the records that are to be copied.

*ALL - No record selection is to be performed. All records in the file are available for copying.

Query selection - Specify an expression (contained in apostrophes) that describes the values used to determine which records are selected. You can specify any logical expression formed from relationships (such as *EQ and *NE) of field and constant values or functions of field and constant values. At least one field name must be specified in each relationship.

TOFILE: Specifies the name and library of the XML data file to create. The file name must not already exist in the specified library. The member name created in this file will be the same as the file name specified.

DECSEP: Specifies the character to be used as the decimal separator in a numeric field that contains decimal positions.

'.'	A full-stop character is used as the decimal separator.
*ASTERISK	An asterisk character '*' is used as the decimal separator.
*COLON	A colon character ':' is used as the decimal separator.
*COMMA	A comma character ',' is used as the decimal separator.
*DOT	A dot character '.' is used as the decimal separator.
*PERIOD	A period character '.' is used as the decimal separator.
*SEMICOLON	A semicolon character ';' is used as the decimal separator.
*SLASH	A slash character '/' is used as the decimal separator.
*STOP	A stop character '.' is used as the decimal separator.
*TAB	A tab character x'05' is used as the decimal separator.

decimal separator character - Specify the character to use as a decimal separator.

RPLCHR: Specifies the character to be used for replacing any detected special characters in element names. By definition, element names must start with a letter or an underscore character. The rest of the name is any combination of letters, digits, underscores, periods, hyphens or colons. Names can not have embedded spaces or start with any form of the string "XML". Element name is case sensitive.

*UNDERSCORE An underscore character '_' is used for special character replacement.

*DASH A dash character '-' is used for special character replacement.

NBRRCDS: specifies the maximum number of records to be copied.

*END - All records in the from-file are copied, dependent upon the selection criteria (QRYSLT parameter).

Number of records - Specify the maximum number of records that are to be copied.

NCVTDBFSLK - Convert database file to SLK data file

The NCVTDBFSLK command allows you to convert iSeries database data into a stream file in .slk format, suitable for sending to a non-iSeries system.

The .slk (SYmbolic LinK) format is an ASCII text file that can be imported directly into Spreadsheet applications, such as Microsoft Excel™. Information regarding the .slk format can be found in Wikipedia

The command can only process physical files, with a maximum record length of 8192 bytes.

Due to compatibility issues between Excel™ and the .slk format this command can only process a maximum of 65535 rows of data.

Note also that versions of Excel issued after Excel 2013 *may not be compatible* with the output of this command.

Command Syntax

```
NCVTDBFSLK—STM—SQL selection statement--...

     TODIR-Target Directory/Path--...

     TOSTMF-Target Stream file name--...

     STMFCODPAG-Stream file code page--...

     NULLVALUE-Replacement character(s)--...

     DTAAUT-data authorities--...

     OBJAUT-Object authorities--...
```

Parameter Definitions

STM - Specifies a valid interactive SQL select statement for obtaining the input data.

TODIR - Specifies the full path name of the directory to which the stream file is copied. All directories and subdirectories in the path name must exist; new directories are not created by this command.

TOSTMF - Specifies the stream file name to which data is copied.

STMFCODPAG - Specifies the method of obtaining the stream file code page and the CCSID used for data conversion.

NULLVALUE - Specifies the replacement characters to be used when null values are detected in the selected data.

DTAAUT- Specifies the data authorities to be assigned to the stream file.

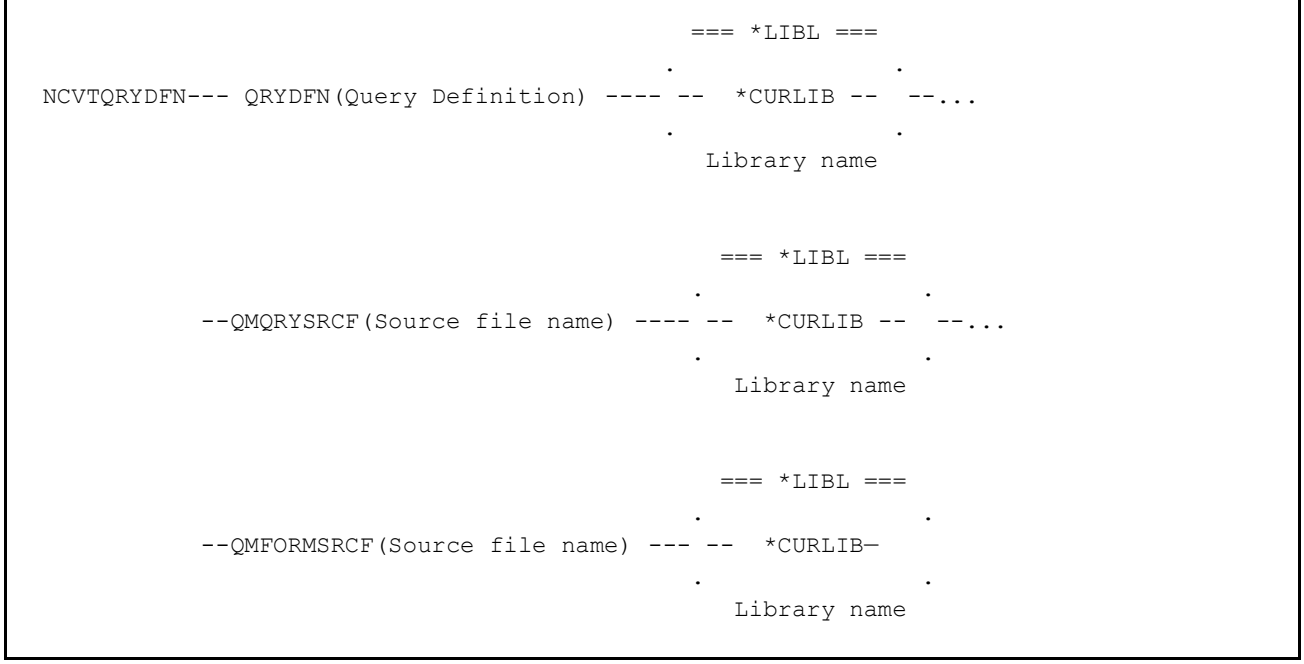
OBJAUT- Specifies the object authorities to be assigned to the stream file

NCVTQRYDFN - Convert QUERY/400 *QRYDFN to Query Management/400

The NCVTQRYDFN (Convert Query Definition object to Query Management) command allows you to convert a Query Definition, which was created using the Query/400 Program Product, into equivalent Query Management/400 source and objects.

Using the *QRYDFN object specified, the command will generate the QMQRY and QMFORM source into the specified source files and then attempt to generate *QMQRY and *QMFORM objects based on the retrieved source.

Command Syntax



Parameter Definitions

QRYDFN: The Query Definition parameter allows you to specify which *QRYDFN object to process. The newly generated *QMQRY and *QMFORM objects will be created in the library specified in this parameter.

QMQRYSRCF: The *QMQRY source file parameter specifies the name of the source file into which the QMQRY source member will be created.

> Note that the system default name for this source file is QQMQRYSRC. The source file used for QMQRY source should have a record length of 91 characters.

QMFORMSRCF: The *QMFORM source file parameter specifies the name of the source file into which the QMFORM source member will be created.

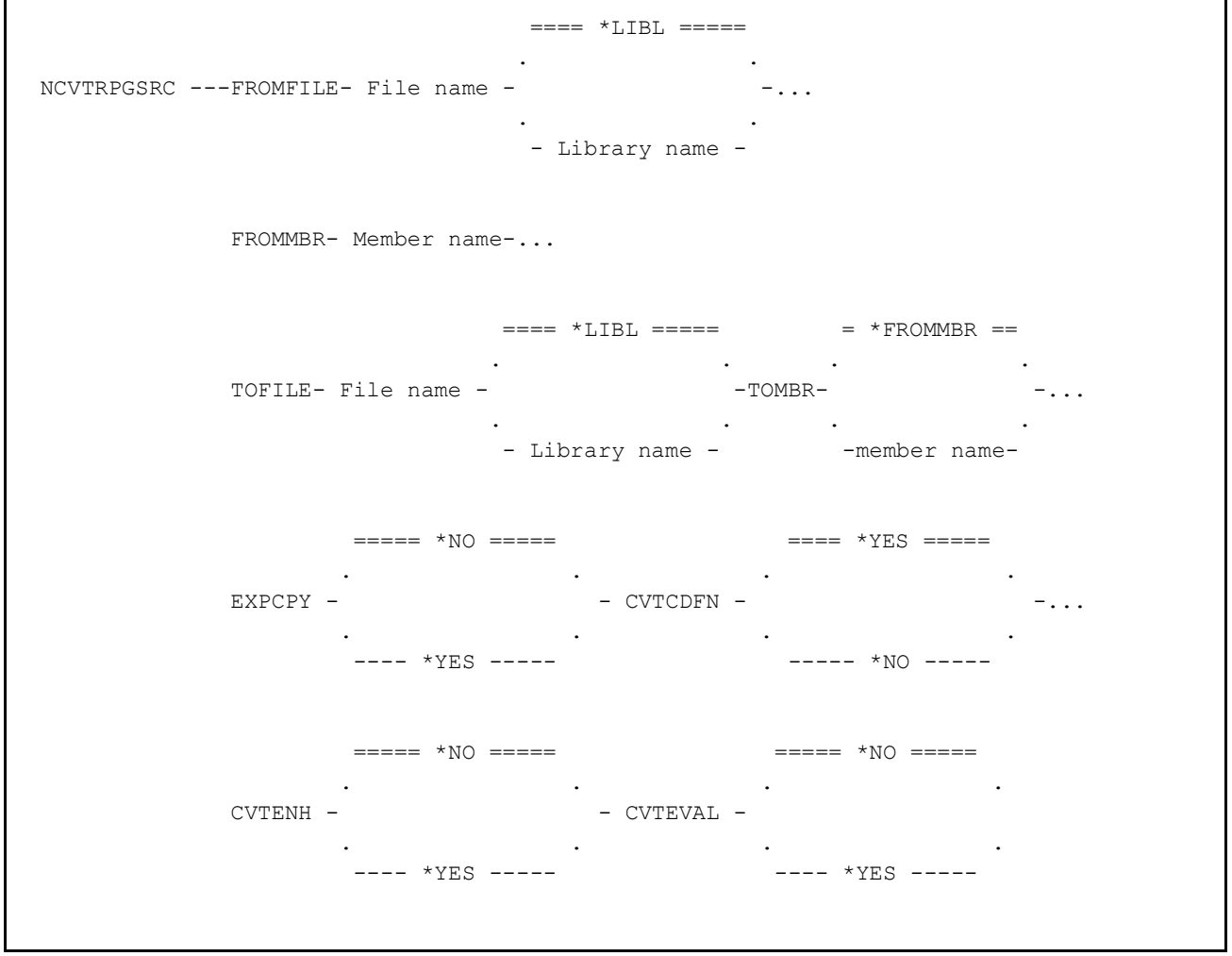
> Note that the system default name for this source file is QQMFORMSRC. The source file used for QMFORM source should have a record length of 162 characters.

NCVTRPGSRC - Convert RPG/400 to ILE RPG source code

The NCVTRPGSRC command allows you to convert RPG III or RPG/400 source code to ILE RPG source code.

In addition to the processing done by the IBM CVTRPGSRC command, this command converts field names from upper to lower case and also (optionally) generates Data specifications (‘D’ specs) to define any fields that were previously defined in calculation specification lines.

Command Syntax



Parameter Definitions

FROMFILE: Specifies the name of the source file containing the RPG III or RPG/400 source member to be read for conversion. This is a required parameter.

FROMMBR: allows you to specify the name of the source member to be processed for conversion.

TOFILE: Specifies the name of the target source file, where the converted source member will be stored. This is a required parameter.

> The TOFILE file specified must exist and should have a record length of 122 characters or more. The default name for an ILE RPG source file is QRPGLESRC.

TOMBR: allows you to specify the name of the target (converted) source member.

Enter the member name for the target (converted) source member. If the member currently exists, it will be replaced. If the member name does not currently exist in the target source file, it will be created.

An entry of *FROMMBR specifies that the target converted source is the same as the member name specified in the From Member parameter.

EXPCPY: specifies whether any /COPY member statements in the program source are expanded in the converted source member.

Note: If the source member is of type RPT or RPT38, EXPCPY(*YES) or EXPCPY(*NO) has no effect because the auto report program will always expand the /COPY members.

*NO - Do not expand the /COPY file member(s) into the converted source. The /COPY statement remains in place in the converted source member.

*YES - Expand the /COPY file member(s) into the converted source.

CVTCDFN: specifies whether any fields that are defined in calculation specification lines (in positions 49-51) should be converted in to D (Data) specification line field definitions.

Note: If the source member is a /COPY member you should specify CVTCDFN(*NO), otherwise the converted /COPY member may cause compiler errors due to program specification line types being out of sequence.

*YES - Any fields defined in positions 49-51 on a C Spec will be converted in to D (Data) specifications. The field definition will be removed from the C specification line and a new D specification line will be added to the converted source, to define the field.

*NO - Any fields defined in positions 49-51 will not be converted in to D (Data) specifications. The C spec field definition will remain unchanged in the converted source.

CVTENH: specifies whether you wish source code enhancements to be applied. These additional enhancements relate to the structured operation codes IF, DO, CAS and WHEN.

Note: Selecting enhanced code conversion requires the source code to be resequenced, to allow the code changes to occur. In doing so, any comments on positions 1-5 of source lines (other than array and table data) will be dropped in the resulting source member.

> *YES - Enhanced code conversion will occur. Each column-defined IF*xx*, DO*xx*, CAS*xx* and WHEN*xx* statement detected will be converted in to it's free format equivalent. Any comments detected in positions 1-5 of the source code line will be dropped.
>
> *NO - Enhanced code conversion will not occur. Source will be converted on a line-for-line basis.

CVTEVAL: specifies whether the converted program source will retain usage of any old "fixed format" RPG Operation codes, or convert any usage to an equivalent EVAL free-format Operation Code:

> *YES - Converted program source will use the EVAL Operation Code.
>
> *NO - Converted program source will retain any usage of old RPG Operation Codes.

For CVTEVAL(*YES), the following conversions will occur:

- MOVEL will be replaced by Eval
- MOVEL (with Blank Padding) will be replaced by Eval
- MOVE (indicators) will be replaced by Eval
- MOVE will be replaced by Evalr
- MOVE (with Blank Padding) will be replaced by Evalr
- Z-ADD will be replaced by Eval
- Z-ADD (with Half-Adjust) will be replaced by Eval(h)
- ADD will be replaced by Eval
- ADD (with Half-Adjust) will be replaced by Eval(h)
- SUB will be replaced by Eval
- SUB (with Half-Adjust) will be replaced by Eval(h)
- MULT will be replaced by Eval
- MULT (with Half-Adjust) will be replaced by Eval(h)
- DIV will be replaced by Eval
- DIV (with Half-Adjust) will be replaced by Eval(h)

Conversion Notes:

Div - subsequent use of the MVR operation requires manual attention (replace with %Rem)

Any result field definitions in C specs will be dropped

Move operations (replaced by Evalr) may require manual attention after conversion. The old move operation code does not have a direct equivalent in ILE - the closest is Evalr. But Evalr is far more formalised (and restrictive) in it's capability.

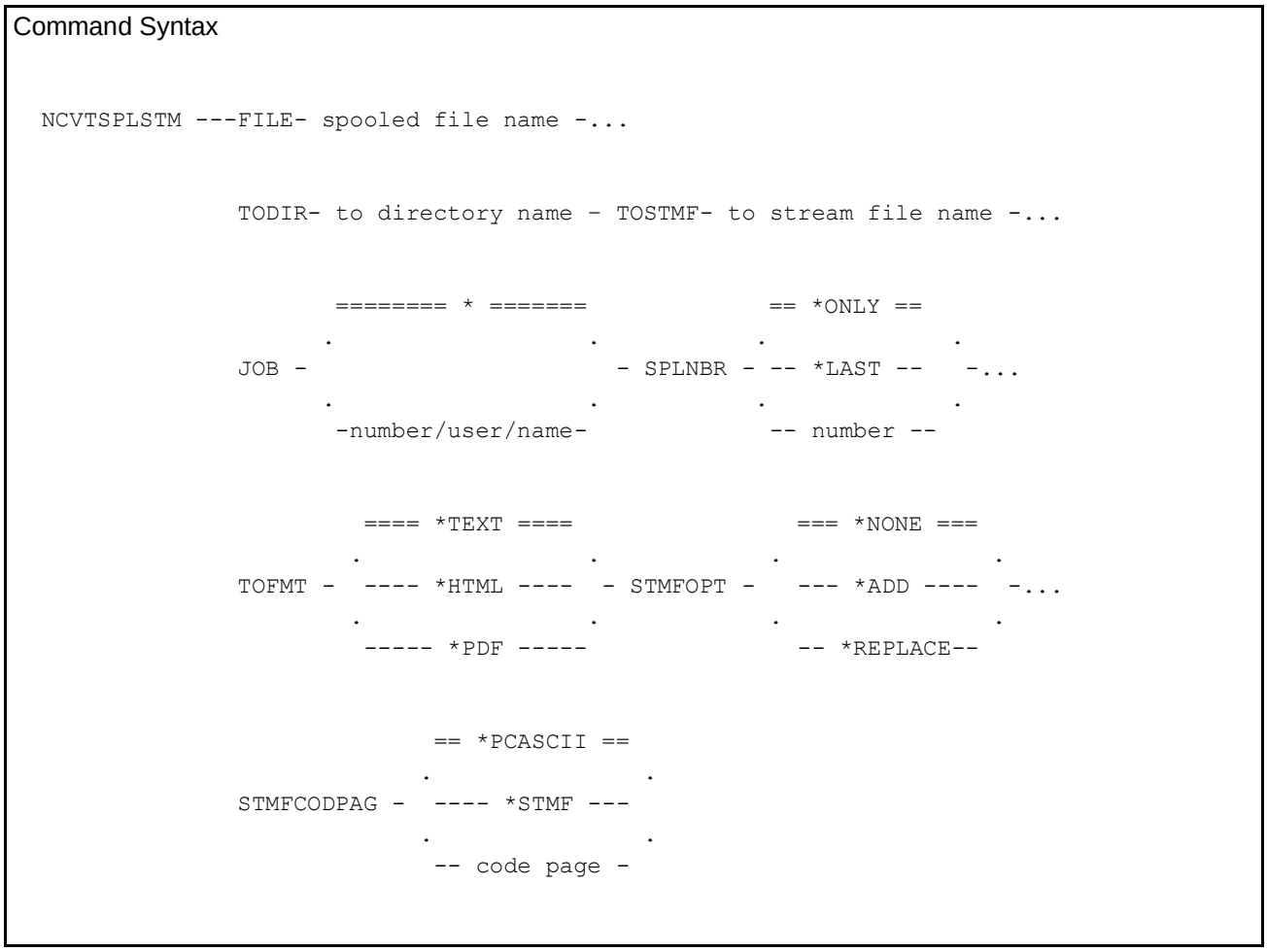
NCVTSPLSTM - Convert spooled file to stream file

The NCVTSPLSTM command lets you convert a spooled file in to a stream file, to be stored in the iSeries integrated file system (IFS) root file system.

Using shared folders, the resulting stream file can be opened in a PC application such as a document reader, spreadsheet or a browser. The stream file can also be sent as an email attachment (for example, via the NSNDSMTPML command), published on the web, etc.

> *Note:* NCVTSPLSTM does not currently support conversion of Advanced Function Printing Data Stream (AFPDS) spool files.

Command Syntax



Parameter Definitions

FILE: Specifies the name of the spooled file that is to be copied to a stream file. This is a required parameter.

TODIR: Specifies the full path name of the directory to which the stream file is copied. All directories and subdirectories in the path name must exist; new directories are not created by this command. This is a required parameter.

Note: The QSYS file system does not allow attributes to be set, so if the path name specified on the TOSTMF parameter is a QSYS member, diagnostic messages will appear in the joblog. The diagnostic messages will not prevent the copy operation from completing successfully and can be ignored.

TOSTMF: Specifies the stream file name to which data is copied. This is a required parameter.

JOB: Specifies the name of the job that created the spooled file whose data records are to be copied.

* - The job that issued this command is the job that created the spooled file

number/user/name - Specify the name of the job that created the spooled file

SPLNBR: Specifies the number of the spooled file, from the job whose data records are to be copied.

*ONLY - Only one spooled file in the job has the specified file name; therefore, the number of the spooled file is not necessary

*LAST - The spooled file with the highest number and the specified file name is used

spooled file number: Specify the number of the spooled file whose data records are to be copied.

TOFMT: Specifies the required format of the resulting stream file.

*TEXT - Each line of the spooled file is represented by a single record in the text file and terminated by a carriage return or line feed pair. Blank lines in the spooled file will not appear in the resulting text file. The resulting stream file should have a “.txt” suffix assigned to ensure correct PC file associations.

*HTML - The program formats the report for viewing in a browser application (such as Microsoft Internet Explorer) to appear similar in layout to original spooled file. Line spacing and formatting is retained wherever possible. The resulting stream file should have a ".htm" (or ".html") suffix assigned to ensure correct PC file associations.

*PDF - The program formats the report for viewing via a reader utility for Portable Document Files (such as the Adobe Acrobat Reader application). Line spacing and formatting is retained wherever possible. The resulting stream file should have a ".pdf" suffix assigned to ensure correct PC file associations.

STMFOPT: Specifies whether the copy operation replaces, adds, or fails to copy the records in a stream file if a stream file with the specified name already exists. If the stream file does not exist, it is created.

*NONE - No records are copied and an error condition is created

*ADD - The records are added to the end of the existing stream file records

*REPLACE - The records replace the existing stream file records

PAGSIZ: Specifies, for *PDF output, the page size definition to use.

*A4 - the page size will conform to A4 paper (21cm width, 29.7cm height)

*LETTER - the page size will conform to LETTER paper (21.59cm width, 27.94cm height)

PAGRTT: Specifies, for *PDF output, the page orientation to use.

*PORTRAIT - Prints a document with the short edge of the paper at the top of the page.

*LANDSCAPE - Prints a document with the long edge of the paper at the top of the page.

*AUTO - Indicates that automatic rotation of output is done to fit the printed data on the form.

STMFCODPAG: Specifies the method of obtaining the stream file code page and the CCSID used for data conversion.

*PCASCII If the stream file exists, this option is valid only if the code page associated with the stream file is the same as the specified value. Otherwise, the operation will fail

*STMF If the stream file exists, and data conversion is requested, the CCSID equivalent of the code page associated with the stream file is used to perform the conversion. If the stream file does not exist, the code page equivalent of the source database file CCSID is used and associated with the stream file

code-page-number Specify the code page to be used. If the stream file exists, this option is only valid if the code page associated with the stream file is the same as the specified value. Otherwise, an error condition is created. If the stream file does not exist, the specified code page is associated with the stream file when it is created

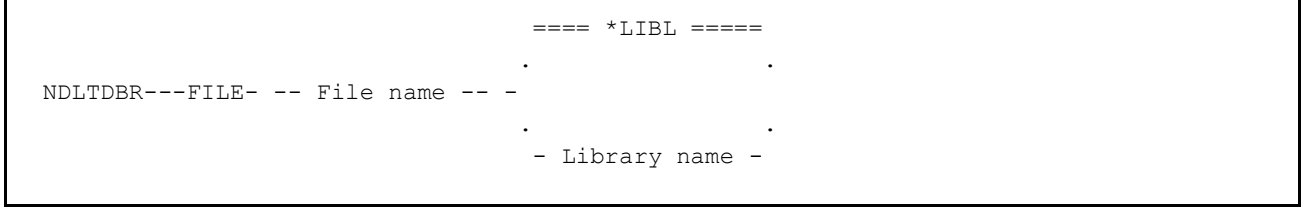
NDLTDBR - Delete database file relations

The NDLTDBR command will delete any files that are dependent upon the file specified.

A report will be printed showing the dependencies that existed before the deletion processing commences.

Note: The specified file itself is not deleted, only the files that are dependent upon it.

Command Syntax



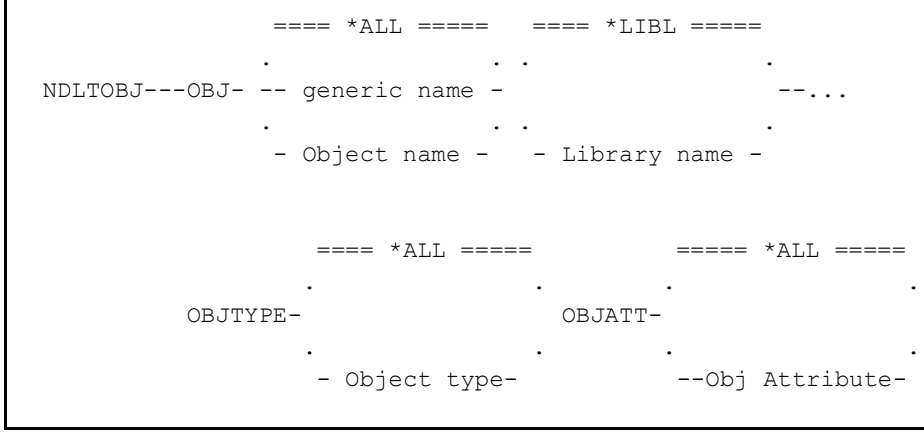
The files deleted by the command can be determined by reviewing the low level messages in the job log.

NDLTOBJ - Delete objects by object type

This command allows you to delete a group of objects by a specific object type and, optionally, specific object attribute.

For example, deletion of all RPG programs within a specific application. Deletions may be generic, so that a request of 'GL*' will delete all gl.. objects of the specified object type within a specified library.

Command Syntax



The objects deleted by the command can be determined by reviewing the low level messages in the job log.

NDUPOBJ - Duplicate objects

This command allows you to duplicate objects.

It is similar to the CRTDUPOBJ command, but has the added advantages of allowing generic selection of objects, and (optionally) deleting any existing object in the destination library.

Command Syntax

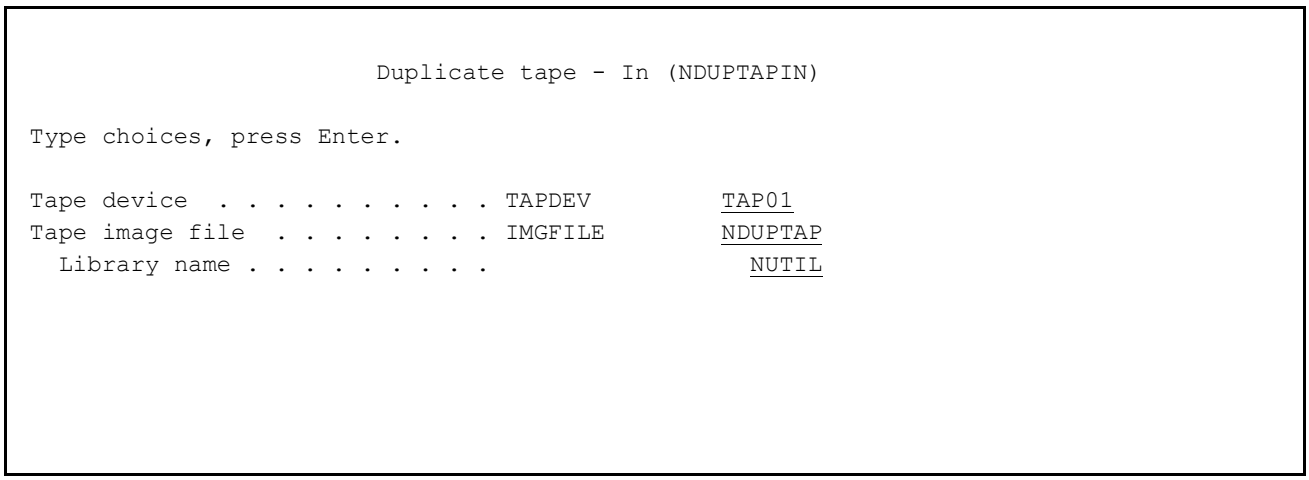
```
                --- *ALL ---
               .             .
NDUPOBJ—OBJ--  Object name   -Object library ---OBJTYPE-Object type --...
               .             .
                -- *generic -

  --TOLIB-To library name ---DATA-*yes/*no ---REPLACE-*yes/*no
```

NDUPTAPIN - Duplicate tape (in)

The NDUPTAPIN command is the first command in a set of tape duplication routines that only requires one tape drive. NDUPTAPIN reads the source tape and stores the tape image on disk. Then the NDUPTAPOUT command reads that stored image file to write a copy of the tape.

While the density and length of the target tape does not have to be the same as the source tape, you must ensure that the full tape image from the source tape will fit on the target tape. The capacity of the target tape must be equal to, or greater than, the capacity of the source tape.

This also applies to a multi-volume tape set, where each tape must be copied to a separate target tape (tapes cannot be appended).



TAPDEV: Specifies the name of the device from which the tape is copied. This is a required parameter.

IMGFILE: Specifies the name and library of the tape image file that is used to store the images of the data from the tape to be copied. The file must not currently exist; it will be created by the command. The default name for the file is NDUPTAP, but you may use any name you like. This allows you to store many tape images on your disk.

The possible library values are:

NUTIL -The tape image file will be stored in the NUTIL product library.

*LIBL - The library list is used to locate the tape image file.

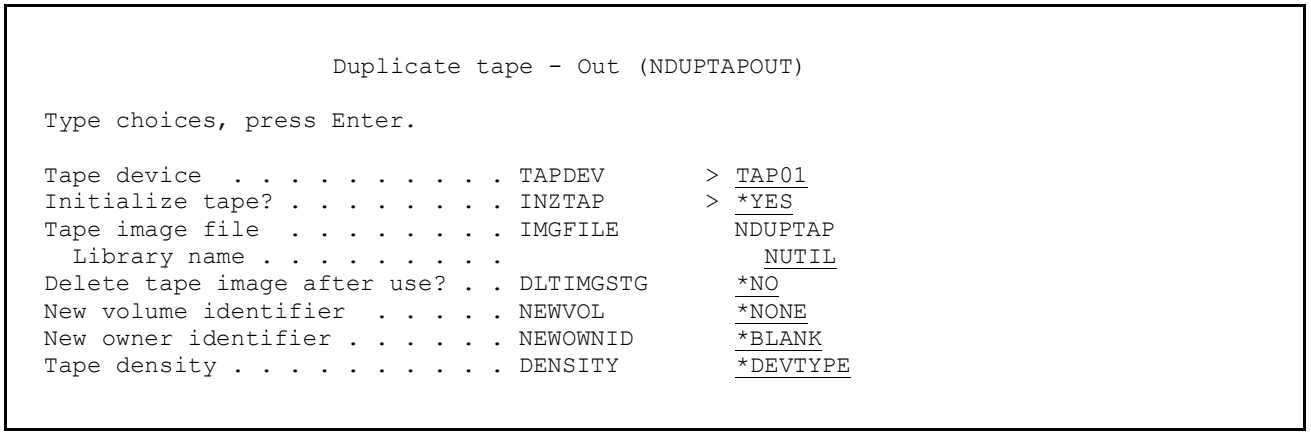
*CURLIB - The tape image file is in the current library. If no current library is defined for this job, QGPL is assumed.

Library name - Specify the name of the library where the tape image file resides.

NDUPTAPOUT - Duplicate tape (out)

The NDUPTAPOUT command is the second command in a set of tape duplication routines that only requires one tape drive. NDUPTAPOUT reads the tape image file that was stored on disk by the NDUPTAPIN command and uses it to write copies of the original source tape.

While the density and length of the target tape does not have to be the same as the source tape, you must ensure that the full tape image will fit on the target tape. The capacity of the target tape must be equal to, or greater than, the capacity of the source tape.



```
                  Duplicate tape - Out (NDUPTAPOUT)

Type choices, press Enter.

Tape device  . . . . . . . . . . TAPDEV       > TAP01
Initialize tape? . . . . . . . . INZTAP       > *YES
Tape image file  . . . . . . . . IMGFILE        NDUPTAP
  Library name . . . . . . . . .                  NUTIL
Delete tape image after use? . . DLTIMGSTG      *NO
New volume identifier  . . . . . NEWVOL         *NONE
New owner identifier . . . . . . NEWOWNID       *BLANK
Tape density . . . . . . . . . . DENSITY        *DEVTYPE
```

TAPDEV: Specifies the name of the device from which the tape is copied. This is a required parameter.

INZTAP: Specifies whether the command should first initialize the tape before starting to copy the stored images to it. If you do require the tape to be initialized you will be prompted for the initialization parameters.

Please note that the target tape for the copy must be empty. The possible values are:

*NO - Tape initialization is not required. The tape is already initialized correctly.

*YES - Tape initialization should be performed before commencing the copy.

IMGFILE: Specifies the name and library of the tape image file that contains the stored images of the data from the tape to be copied. This is the file name you specified in the IMGFILE parameter of the NDUPTAPIN command.

DLTIMGSTG: Specifies whether the command should delete the tape image file after completing the copy. The tape image file can be deleted at a later time using the DLTF (Delete File) command.

The possible values are:

*NO - The tape image file should be left on disk.

*YES - After successfully copying the tape image to tape, the tape image storage file is to be deleted.

NEWVOL: Specifies the volume identifier for a tape being initialized for use as a standard labelled tape. If no volume identifier is specified, the tape is initialized for use as an unlabeled tape.

NEWOWNID: Specifies the identifier of the tape owner to write in the volume label.

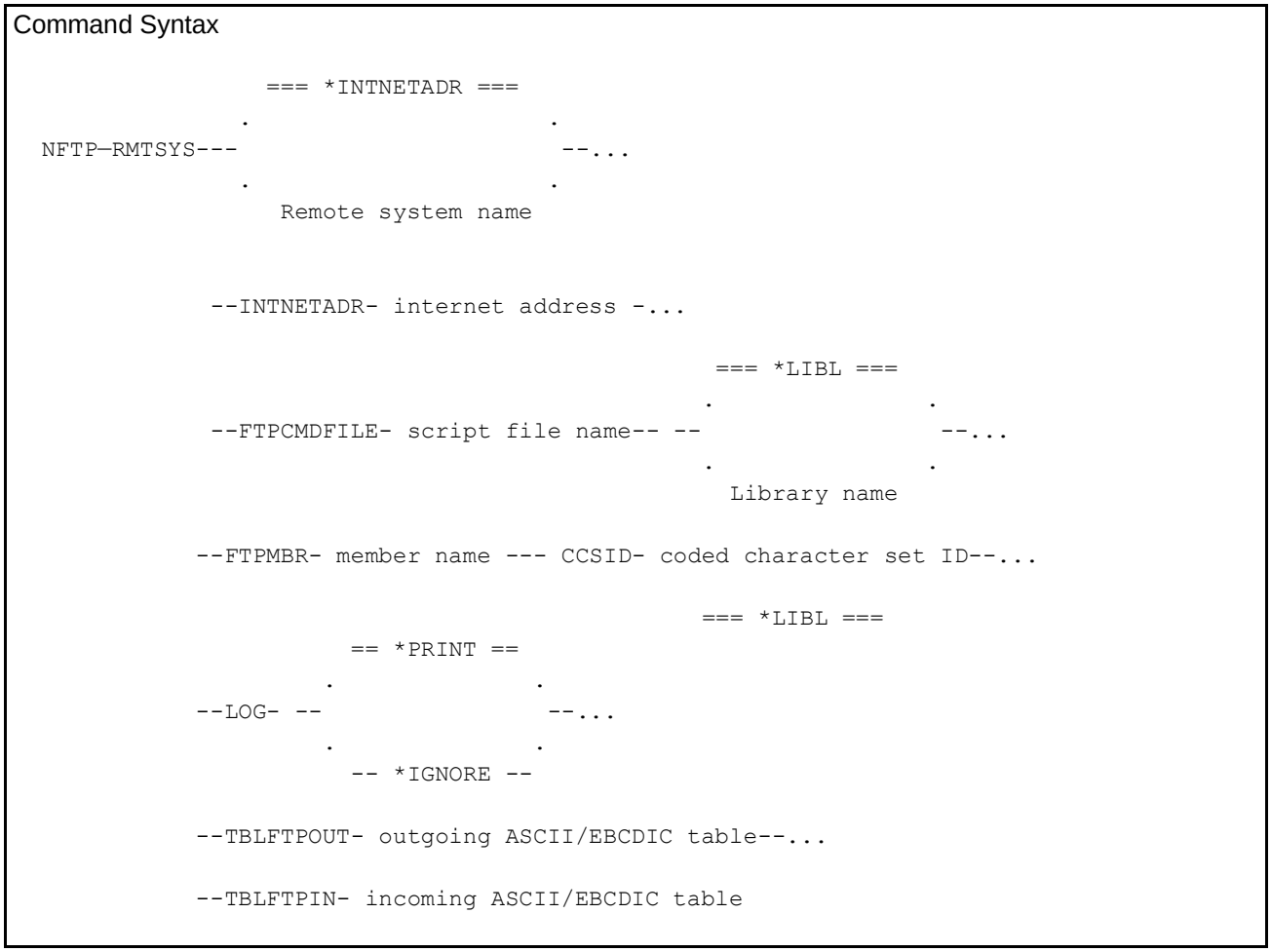
DENSITY: Specifies the recording format of the data to be written on the tape. Allowable values can be seen by typing a ? into the field and pressing enter. The default *DEVTYPE will initialize the tape based on the type of tape drive being used.

NFTP - Start batch TCP/IP file transfer

The Start Batch TCP/IP File Transfer (NFTP) command is used to start the File Transfer Protocol (FTP) client application that transfers files between systems using the Transmission Control Protocol/Internet Protocol (TCP/IP). The FTP subcommands processed by NFTP are stored in a *'processing script'*, which is stored in a source file member.

For more details concerning FTP subcommands and processing scripts, refer to the IBM iSeries TCP/IP Configuration and Reference Guide.

Command Syntax



Parameter Definitions

RMTSYS (Remote System) - Specifies the remote system name to which or from which the processing will occur. The remote system name must be valid, and the system must be able to communicate with the local system.

INTNETADR (Internet address) - If *INTNETADR was specified in the RMTSYS parameter, this specifies the internet address of the remote system to which or from which the file transfer takes place. The internet address is specified in the form *nnn.nnn.nnn.nnn*, where *nnn* is a decimal number ranging from 0 through 255.

FTPCMDFILE (FTP subcommand script file) - Specifies the source file name where the FTP subcommands are kept.

FTPMBR (FTP subcommands script member name) - Specifies the source member in the FTP subcommands file that contains the FTP subcommands to be processed. Example FTP scripts can be found in file NUTIL/QSTUFF, members NFTPGET and NFTPPUT.

LOG (Print FTP transmission log) - Specifies whether the FTP transmission log should be printed upon completion of the FTP processing. The log shows the full conversation between the source and target machines. The possible log values are:

*PRINT- The log is printed.

*IGNORE- The log is not printed.

CCSID (Coded character set ID) - Specifies the ASCII coded character set identifier (CCSID) that is used for single-byte character set (SBCS) ASCII file transfers when the FTP TYPE mode is set to ASCII. ASCII file transfers are also assumed when no TYPE subcommand has been issued. The CCSID value chosen is the default used by the FTP client for ASCII-to-EBCDIC and EBCDIC-to-ASCII mapping. Mapping is determined using the specified ASCII CCSID and the EBCDIC CCSID of the job. The default (*DFT) will use the CCSID value of 00819 (ISO 8859-1 8-bit ASCII).

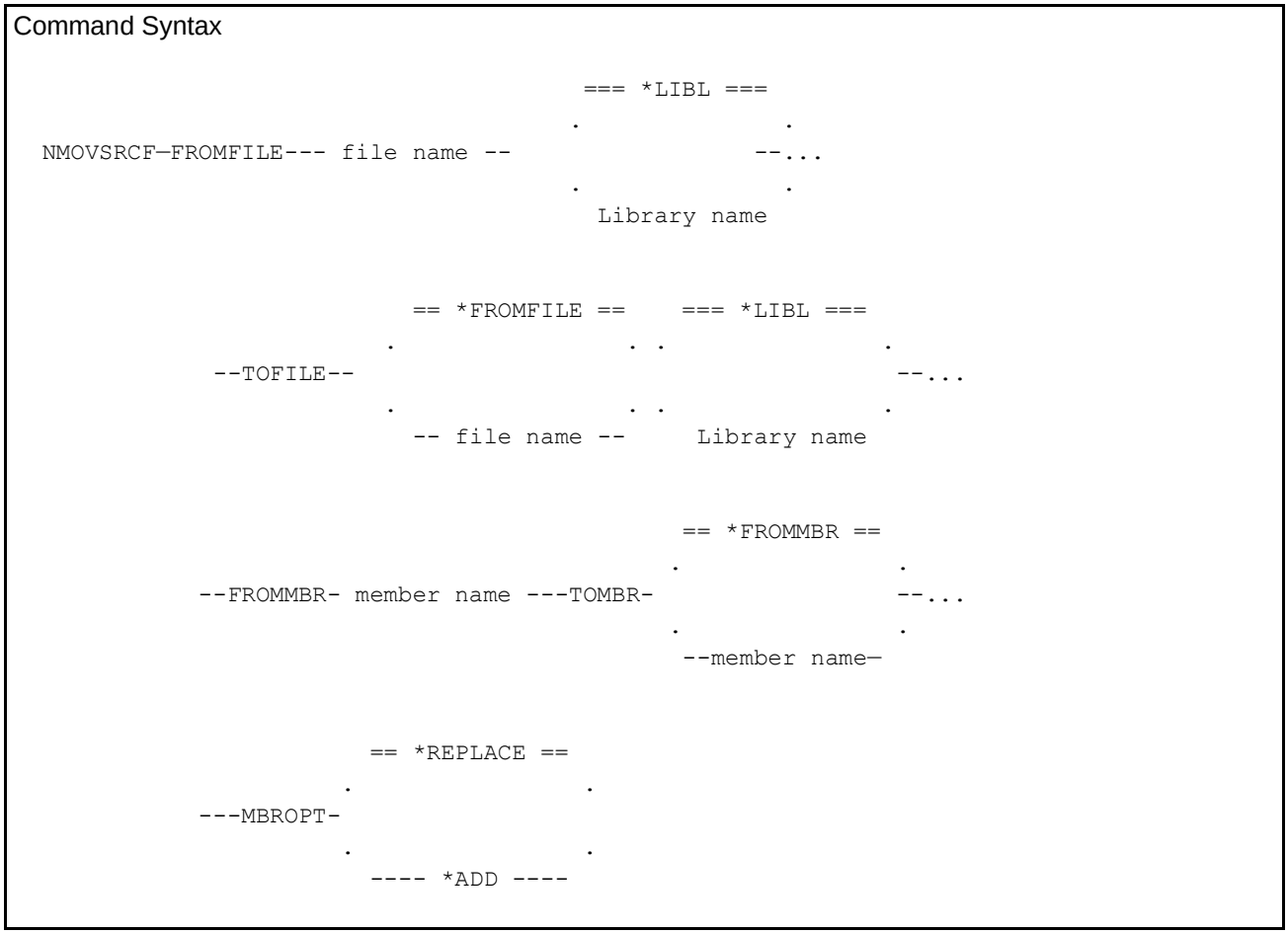
TBLFTPOUT (Outgoing ASCII/EBCDIC table) - Specifies the table object that is to be used to map all outgoing data in the FTP client. Outgoing data is mapped from EBCDIC to ASCII. The default (*CCSID) is that the CCSID parameter is used to determine the outgoing mapping

TBLFTPIN (Incoming ASCII/EBCDIC) table) - Specifies the table object that is to be used to map all incoming data in the FTP client. Incoming data is mapped from ASCII to EBCDIC. The default (*CCSID) is that the CCSID parameter is used to determine the incoming mapping

NMOVSRCF - Move Source File Members

The NMOVSRCF command allows you to move a source member from one source file to another source file.

Command Syntax



Parameter Definitions

FROMFILE: Specifies the source physical file currently containing the member that is to be moved. Specify the name of the source file that is to be used. The possible library values are:

*LIBL: The library list is used to locate the source file.
*CURLIB: The source file is in the current library. If no current library is defined for this job, QGPL is assumed.
Library name: Specify the name of the library where the source file resides.

TOFILE: Specifies the source physical file into which the specified member is to be moved. Specify the name of the source file that is to be used. A special value of *FROMFILE allows you to specify that the source member is being moved to a new source member in the same file. The possible library values are:

*LIBL: The library list is used to locate the source file.
*CURLIB: The source file is in the current library. If no current library is defined for this job, QGPL is assumed.
Library name: Specify the name of the library where the source file resides.

FROMMBR: Specifies the name of the source member in the from-file that is to be moved. Specify the name of the member that is to be used. This is a required parameter.

TOMBR: Specifies the name of the source member that receives the moved source records. The possible to-member values are:

> *FROMMBR: Corresponding from-file and to-file member names are used.
> Member name: Specify the name of the source file member that receives the moved source records.

MBROPT: Specifies whether the moved source records replace, or are added to, existing data. This parameter is only meaningful if the TOMBR already exists. The possible member option values are:

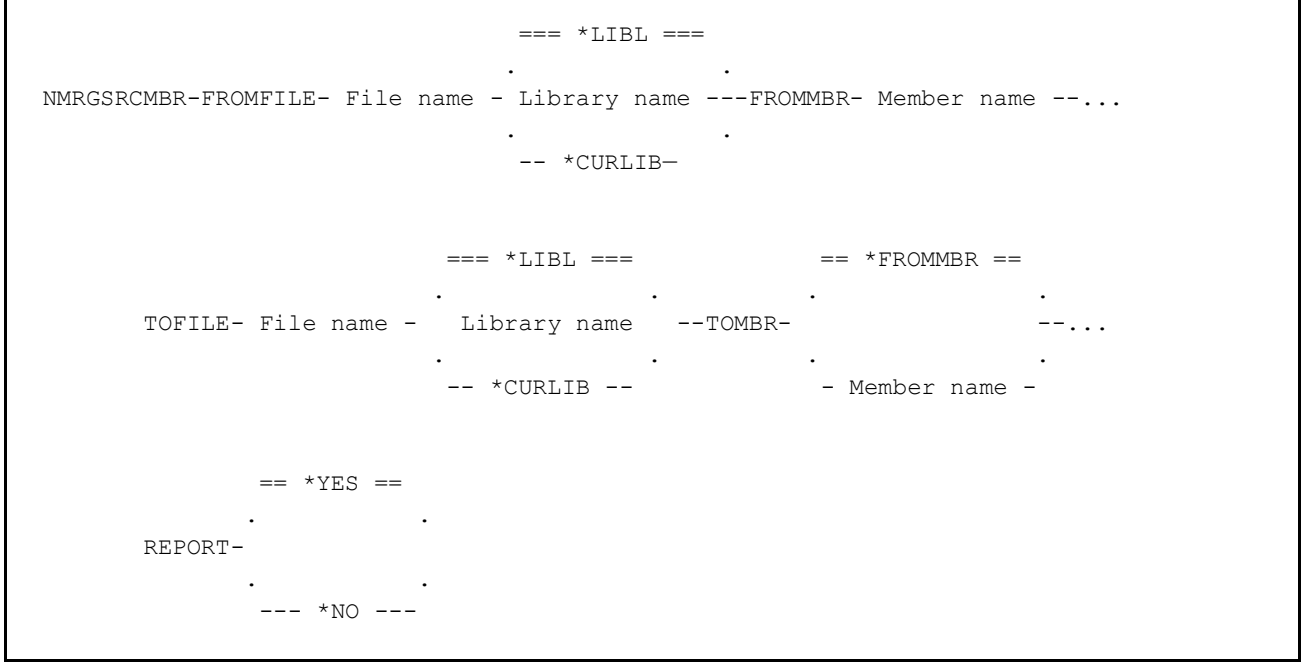
> *REPLACE: The system clears the existing member (if it already exists) and adds the new records.
> *ADD: The system adds the new records to the end of the existing records.

NMRGSRCMBR - Merge two source members

The NMRGSRCMBR command allows you to merge updates from maintenance file members to target file members.

This is a front-end for the IBM MRGSRC (Merge Source) command. Full details of the way this command works are contained in the IBM iSeries File Compare and Merge Utility Guide. You should fully understand the workings of the IBM MRGSRC command before attempting to use the Navan NMRGSRCMBR command.

Command Syntax



Parameter Definitions

FROMFILE: Specifies the source physical file containing the updates to merge. This is a required parameter.

The possible library values are

*LIBL: The library list is used to locate the source file
*CURLIB: The source file is in the current library. If no current library is defined for this job, QGPL is assumed.
Library name: Specify the name of the library where the source file resides

FROMMBR: Specifies the source member containing the updates to merge. Specify the name of the member that is to be used. This is a required parameter.

TOFILE: Specifies the source physical file into which the updates are to be merged. This is a required parameter.

The possible library values are

*LIBL: The library list is used to locate the source file
*CURLIB: The source file is in the current library. If no current library is defined for this job, QGPL is assumed.
Library name: Specify the name of the library where the source file resides

TOMBR: Specifies the source member into which the maintenance updates are to be applied. The possible member values are:

> *FROMMBR: Corresponding from-file and to-file member names are used.
> Member name: Specify the name of the source file member into which the maintenance updates are to be applied.

REPORT: Specifies whether the *from* and *to* members are to be compared, and the differences reported, after the merge processing has completed. The possible report values are:

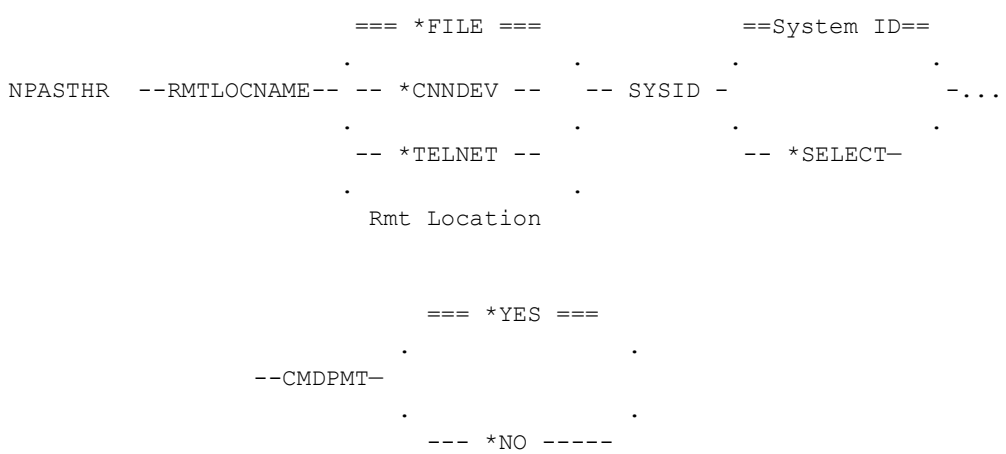
> *NO: No reporting will be performed by the command.
> *YES: Before merge processing commences, a control report will be printed, showing the number of areas that have been found to be different between the source members. After merge processing has completed a comparison report will be printed showing, in detail, the line differences between the from member and the to member.

NPASTHR - Start Remote Connection

The NPASTHR command is a 'front-end' utility for the standard iSeries remote connection commands STRPASTHR (Start PassThrough) and STRTCPTELN (Start TCP/IP Telnet) which makes connection a much simpler process for the user, because it obtains most of the communications parameters for the relevant command from a user configured defaults file based upon the name of the remote machine being accessed.

The remote connections to other machines must be defined using the EDTPSTDFT (Edit Passthrough Defaults) command before the NPASTHR command can be used to access the remote systems. Only connections defined in the PassThrough defaults file can be accessed by the NPASTHR command.

Command Syntax



Parameter definitions below are the main ones used for this command. For full details of all command parameters, refer to the EDTPSTDFT (Edit NPASTHR connections) command.

The RMTLOCNAME (remote location name) parameter specifies one of the following:

*FILE - The remote location name is to be obtained from the Connection Defaults file, for the Remote System ID specified.

*TELNET - Connect, using TCP/IP TELNET, to the system defined the Remote System (RMTSYS parameter) prompt.

*CNNDEV - Use the APPC device specified in the APPC device (CNNDEV parameter) prompt.
Remote location name - Specify the remote location name that is to be the target of this (Passthrough) session.

Where the Remote Location Name (RMTLOCNAME) parameter is specified as *FILE, the SYSID Remote System Identifier parameter is used to determine which entry is to be retrieved from the Connection Defaults file. This parameter is only used when the Remote Location Name (RMTLOCNAME) parameter was entered as *FILE.

*SELECT - Select the System Identifier from a list.
System identifier - Enter the System Identifier. This must be a valid entry in the Connection Defaults file.

When you enter Remote Location Name (RMTLOCNAME) as *FILE, connection information is then retrieved from the passthrough defaults file. If you do not specify *FILE you must provide all connection information.

The RMTUSER (Remote User) parameter allows you to specify the user profile for automatic signon to the target system. If a profile is specified for this parameter and password security is active on the target

system, (*NONE) is not valid for the User Password (RMTPWD) parameter. The target system must be configured to allow automatic signon, otherwise a failure message will be sent back to the source system.

*NONE - No user profile name is sent, and no automatic signon occurs.
*CURRENT - The user profile of the job using this command is sent.
User profile name - Specify a user profile name to use that exists on the target system.

The RMTPWD (Remote Password) parameter allows you to specify the password for automatic signon to the target system. The target system must be configured to allow automatic signon, otherwise a failure message will be sent back to the source system.

This password is sent to the target system without being encrypted in any way.

*NONE - No password is sent. If a User Profile has been specified on the User Profile parameter (RMTUSER), this value is not allowed.

Password - Specify a password being sent to the target system to verify the signon information of the user specified in the RMTUSER parameter.

The CMDPMT (Command Prompting) parameter allows you to display the connection command (either STRPASTHR or STRTCPTELN, depending on connection type) prompt before it is actually performed. This then allows you to retrieve standard information from the connection defaults file, but alter one or more values for this specific connection attempt.

Ending an NPASTHR session on the target machine

Where the session is a PassThrough session

NPASTHR starts each passthrough connection session as a ‘group job’ to allow you to have multiple sessions open to multiple machines at the same time. Another advantage of using the group jobs concept is that you can suspend the passthrough sessions on the target systems.

If you use the TFRPASTHR command to return from the target system back to the source system, the passthrough session on the target machine remains open (suspended mode).

When you next enter the NPASTHR command again for that same System ID you will be returned to the suspended target session (provided the target job is still available).

To completely end a passthrough session on the target machine, use the ENDPASTHR (End Passthrough) command.

Where the session is a TELNET session

If the remote system *is not* an iSeries: Many TELNET server implementations automatically end the TELNET session when you log off the target machine. Where the session doesn’t automatically end you should be able to press the Attention key and then select option 99 (end TELNET session) from the ‘Send TELNET control functions’ menu.

If the remote system *is* an iSeries: Use the SIGNOFF command and specify ENDCNN(*YES) for the end connection parameter.

NPRCOBJLCK - Process object locks

The Process Object Locks (NPRCOBJLCK) command allows you to retrieve object locks that are currently held on a specified object. Based on parameters entered, jobs holding locks will either be sent a message, or they will be ended. The command is especially useful where you are trying to obtain an exclusive lock on a file, but are unable to.

Command Syntax

```
NPRCOBJLCK---OBJ- object library/name ---OBJTYPE- object type-...

           --MBR- member name-...

                       === *SNDMSG ===
                      .               .
           --OPTION—                    -...
                      .               .
                       ---- *END ----

           --MSG- message text-...

                       === *UPD ===
                      .             .
           --LCKTYPE—
                      .             .
                       --- *ANY ---
```

Parameter Definitions

MBR (Member name) - Where the object being processed is a database file, specifies the member name to process.

*NONE - No member locks are processed, but object level locks are processed.
*FIRST - The first member in the named file is processed.
*ALL - Member locks for all the members in the file are processed.
Member name - Specify the name of the database file member for which locks are processed.

OPTION (processing option) - Specifies the processing that the command should perform when it detects a lock on the specified object.

*SNDMSG - Send a break message to the job holding the lock. The text of the message is supplied in the MSG parameter of this command.
*END - Jobs holding locks on the specified object should be ended, based on the lock type specified in the LCKTYPE parameter of this command. Any jobs ended will be printed on an audit report at the end of the processing.

MSG (message to send) - For OPTION(*SNDMSG), specifies the immediate message that is being sent.

LCKTYPE (Lock type to end) - For OPTION(*END), specifies whether all jobs accessing the file should be ended, or only those jobs holding update locks on the specified object.

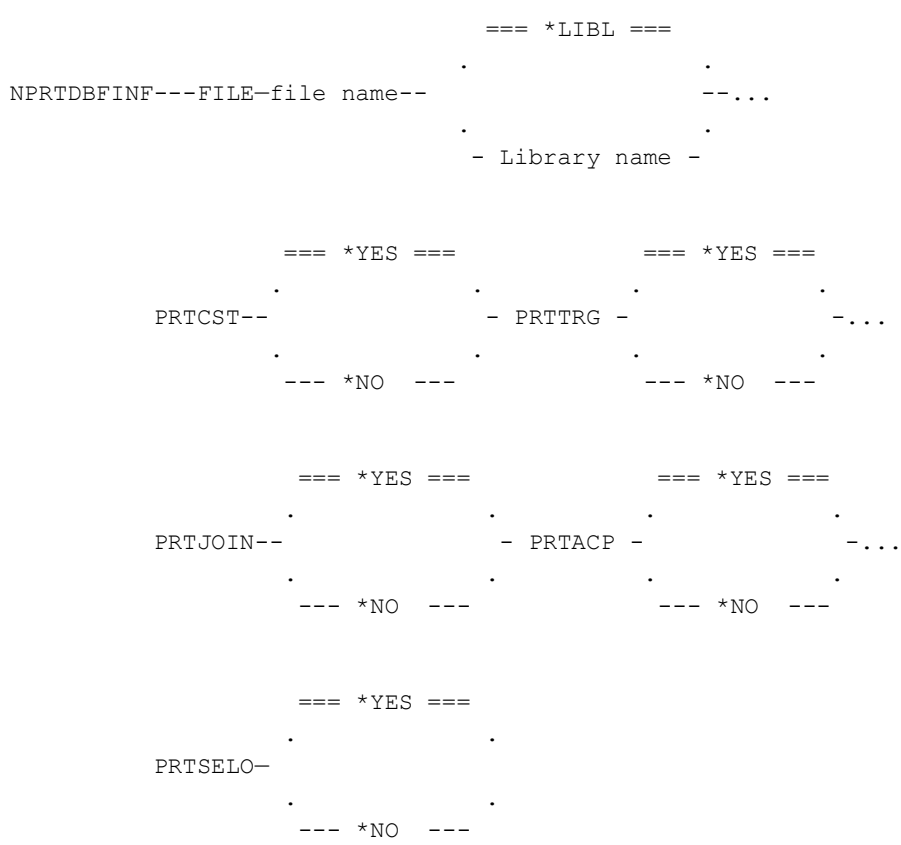
*UPD - Only end jobs that are currently holding an update lock on the specified object.
*ALL - End any jobs that currently have any lock on the object specified.

NPRTDBFINF - Print database information

The Print database file information (NPRTDBFINF) command provides a report detailing selected information retrieved from the attributes of a specified physical database file.

Details of all dependent files are also printed on the report.

Command Syntax



For the physical file name specified (and all files dependent on the specified file), the optional parameters allow you to define the information to be printed on the report:

PRTCST	Print database constraints
PRTTRG	Print database triggers
PRTJOIN	Print join criteria
PRTACP	Print access paths
PRTSELO	Print select/omit criteria

NPRTDOC - Print text document

This command will print a text document that is stored in a source file. It is similar in concept to the PRTDOC command that previously existed on the IBM System/38. For more details on the NPRTDOC command, please see the separate manual.

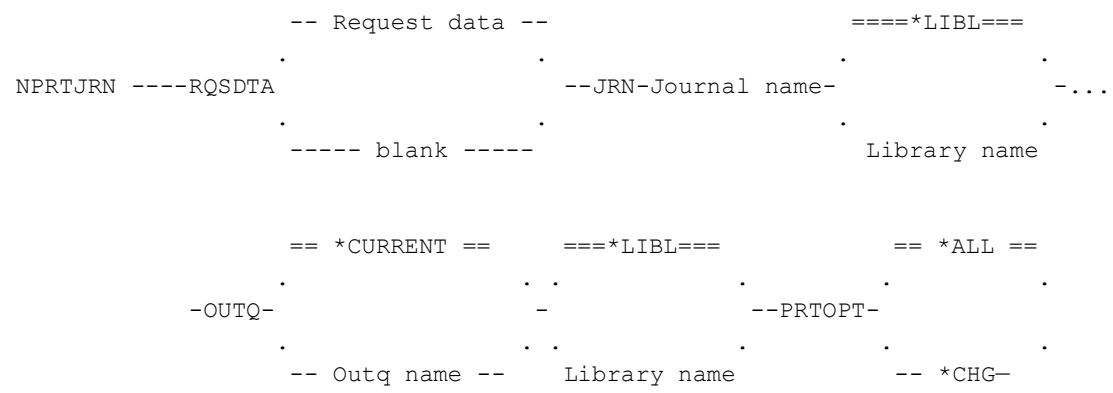
NPRTJRN - Print Journal Images

The Print Journal (NPRTJRN) command provides a 'front-end' to iSeries system journalling, in that it will provide formatted listings of journal entries processed within a specified timeframe. It can work in one of two ways:

- By specifying a request to process, it will run the request and then print listings of all journal entries caused by the processing of that request
- If no request is specified (the OS/400 Display Journal command is prompted), you can print prior journal information from other requests that have been processed

File record images are formatted and printed using the file field definition database that is used by the NUTIL Work with Database files (WRKDBF) utility.

Command Syntax



Parameter Definitions

RQSDTA (Request data) -you may enter any valid command. Upon completion of the Request Data processing, the results of the journalled processing of that request will then printed by the NPRTJRN command.

If you do not enter anything in this parameter, it will be assumed that you wish to print journal entries from previous processing. In this case, you will be later prompted to enter the parameters for the Display Journal (DSPJRN) command.

JRN (Journal name) - specify the name of the journal to process.

OUTQ (Output queue name) - specify the name of the output queue to use. The default of *CURRENT will use the output queue currently specified in the job attributes.

PRTOPT (Print Option) - this can be either

> *ALL - All file fields in each journal entry are printed. Where the data has changed in any field the literal **** *CHG* **** will print beside that field.
>
> *CHG - The only file fields printed are the key fields and those fields where the data contents have changed.

NPRTJRN, using the 'Request data' parameter

You may enter any valid command in the RQSDTA parameter. Upon completion of the Request Data processing, the results of the journalled processing of that request will then printed by the NPRTJRN command.

This is extremely useful as a program debug tool - you can call a program and enter data; then when you exit your program the NPRTJRN command will print out the data changes caused by the execution of the program.

You must remember that the command will only print changes that have been recorded in the specified journal; If a file is not journalled, or it is journalled to a different journal, it will not be included in the report.

Using NPRTJRN to print previous journal entries

If you do not enter anything in the RQSDTA parameter, it will be assumed that you wish to print journal entries from previous processing. In this case, you will be later prompted to enter the parameters for the OS/400 Display Journal (DSPJRN) command. Some of the entries will already be filled in for you. Refer to the IBM iSeries Control Language Reference Guide for a description of the DSPJRN command.

The entries extracted from the journal will then be formatted and printed by the NPRTJRN command.

NPRTJRN Report examples

The next 3 pages show examples of the printed output from the NPRTJRN command. The database file being processed was journaled with IMAGES(*BOTH) specified.

General notes on report information

- The report can process a file with a maximum record length of 4096 bytes.
- File key fields for the record (where the file is a keyed file) will be shown as *key 1, key 2 … key n*
- Numeric field contents will show the decimal positions correctly. A '.' will be used as the decimal separator character.
- Numeric field contents will show negative signs where necessary.
- Where unprintable data is detected in a field, the contents of the field will be printed in hexadecimal format.

NPRTJRN – EXAMPLE OF A JOURNALED DATABASE PUT OPERATION

```
 NUTIL                                                                                                  Date:
 NPRTJRN: Print Journaled Images                                                                        Time:
 System: NAVAN                                                                                          Page:    1
               B E F O R E  I M A G E                                    A F T E R  I M A G E
-------------------------------------------------------------------------------------------------------------------------
  Sequence   Date     Time     Type   Job        User       Job No  Program     Object     Library    Member       *RRN
      1856 06/05/29 18:10:33   PT    QPADEV0102 QPGMR      012942  PAY001      DEMOFILE   NUTIL      DEMOFILE       6
-------------------------------------------------------------------------------------------------------------------------
           ....+....1....+....2....+....3....+....4....+....5                ....+....1....+....2....+....3....+....4....+....5
                                                        Key 1 EMPNO      05214
                                                              TITLE      MISS
                                                              NAME       HARRISON
                                                              STATUS     A
                                                              STRDT      19980701
                                                              DEPT       ACCTS
                                                              CLASS      00720
                                                              ADDR1      3/175 JAMES LANE
                                                              ADDR2      HAPPYTOWN
                                                              STATE      CT
                                                              PCODE      2015-120
                                                              TELE       555 1234
                                                              PAYDT      00000000
                                                              PAYYTD     0000000.00
                                                              ALWYTD     0000000.00
                                                              TRMDT      00000000
                                                              RTMDT      00000000
```

A database PUT operation will show no details in the *Before Image* section of the report. The *After Image* section of the report will show the details of the database record that was added.

Sequence:	Journal entry sequence number
Date/Time:	The timestamp for when the entry was recorded in the journal
Type:	The database operation that caused the journal entry (in this case *PT* means database PUT)
Job/User/Job number:	The name of the job that caused the journal entry to be recorded
Program:	The name of the program that caused the journal entry to be recorded
Object/Library/Member:	The name of the database object that caused the journal entry to be recorded
*RRN:	The relative record number of the database record that caused the journal entry to be recorded

*NPRTJRN – EXAMPLE OF A JOURNALED DATABASE UPDATE OPERATION, WITH PRTOPT(*ALL) SPECIFIED*

```
                                                                                                                    Date:
NPRTJRN: Print Journaled Images                                                                                     Time:
System: NAVAN                                                                                                       Page:    1
              B E F O R E  I M A G E                                          A F T E R  I M A G E
-------------------------------------------------------------------------------------------------------------------------------
  Sequence   Date     Time     Type   Job        User       Job No  Program     Object     Library    Member       *RRN
      1864 06/05/29 18:11:22   UP    QPADEV0102 QPGMR      012942  PAY001      DEMOFILE   NUTIL      DEMOFILE        6
-------------------------------------------------------------------------------------------------------------------------------
           ....+....1....+....2....+....3....+....4....+....5                ....+....1....+....2....+....3....+....4....+....5
EMPNO      05214                                                 Key 1 EMPNO      05214
TITLE      MISS                                                        TITLE      MISS
NAME       HARRISON                                              CHG   NAME       HARRISON  F A
STATUS     A                                                           STATUS     A
STRDT      19980701                                                    STRDT      19980701
DEPT       ACCTS                                                       DEPT       ACCTS
CLASS      00720                                                       CLASS      00720
ADDR1      3/175 JAMES LANE                                      CHG   ADDR1      5/175 JAMES LANE
ADDR2      HAPPYTOWN                                                   ADDR2      HAPPYTOWN
STATE      CT                                                          STATE      CT
PCODE      2015-120                                                    PCODE      2015-120
TELE       555 1234                                                    TELE       555 1234
PAYDT      00000000                                                    PAYDT      00000000
PAYYTD     0000000.00                                                  PAYYTD     0000000.00
ALWYTD     0000000.00                                            CHG   ALWYTD     0000015.00-
TRMDT      00000000                                                    TRMDT      00000000
RTMDT      00000000                                                    RTMDT      00000000
```

A database UPDATE operation will show details of the record before update in the *Before Image* section of the report. The *After Image* section of the report will show the details of the database record after the update was performed. Any fields that have been changed will show a **CHG** flag next to them in the *After* image.

Sequence:	Journal entry sequence number
Date/Time:	The timestamp for when the entry was recorded in the journal
Type:	The database operation that caused the journal entry (in this case *UP* means database UPDATE)
Job/User/Job number:	The name of the job that caused the journal entry to be recorded
Program:	The name of the program that caused the journal entry to be recorded
Object/Library/Member:	The name of the database object that caused the journal entry to be recorded
*RRN:	The relative record number of the database record that caused the journal entry to be recorded

*NPRTJRN – EXAMPLE OF A JOURNALED DATABASE UPDATE OPERATION, WITH PRTOPT(*CHG) SPECIFIED*

```
                                                                                                                  Date:
 NPRTJRN: Print Journaled Images                                                                                  Time:
 System: NAVAN                                                                                                    Page:    1
              B E F O R E  I M A G E                                        A F T E R  I M A G E
-------------------------------------------------------------------------------------------------------------------------------
  Sequence   Date     Time     Type   Job        User       Job No  Program     Object     Library    Member       *RRN
      1864 06/05/29 18:11:22   UP    QPADEV0102 QPGMR      012942  PAY001      DEMOFILE   NUTIL      DEMOFILE        6
-------------------------------------------------------------------------------------------------------------------------------
           ....+....1....+....2....+....3....+....4....+....5                ....+....1....+....2....+....3....+....4....+....5
EMPNO      05214                                                 Key 1 EMPNO      05214
NAME       HARRISON                                                    NAME       HARRISON  F A
ADDR1      3/175 JAMES LANE                                            ADDR1      5/175 JAMES LANE
ALWYTD     0000000.00                                                  ALWYTD     0000015.00-
```

This example shows the same journal entry as the previous example, but in this case the NPRTJRN command was specified with parameter PRTOPT(*CHG). The report will list only the key fields of the record and the contents of the fields that were changed.

Sequence:	Journal entry sequence number
Date/Time:	The timestamp for when the entry was recorded in the journal
Type:	The database operation that caused the journal entry (in this case *UP* means database UPDATE)
Job/User/Job number:	The name of the job that caused the journal entry to be recorded
Program:	The name of the program that caused the journal entry to be recorded
Object/Library/Member:	The name of the database object that caused the journal entry to be recorded
*RRN:	The relative record number of the database record that caused the journal entry to be recorded

NPRTJRN – EXAMPLE OF A JOURNALED DATABASE DELETE OPERATION

```
NUTIL                                                                                                    Date:
NPRTJRN: Print Journaled Images                                                                          Time:
System: NAVAN                                                                                            Page:    1
               B E F O R E  I M A G E                                    A F T E R  I M A G E
-------------------------------------------------------------------------------------------------------------------------
  Sequence   Date     Time     Type   Job        User       Job No  Program     Object     Library    Member       *RRN
      1866 06/05/29 18:11:26   DL    QPADEV0102 QPGMR      012942  WRKDBF1     DEMOFILE   NUTIL      DEMOFILE       6
-------------------------------------------------------------------------------------------------------------------------
           ....+....1....+....2....+....3....+....4....+....5                ....+....1....+....2....+....3....+....4....+....5
                                                          Key 1 EMPNO      05214
                                                                TITLE      MISS
                                                                NAME       HARRISON  F A
                                                                STATUS     A
                                                                STRDT      19980701
                                                                DEPT       ACCTS
                                                                CLASS      00720
                                                                ADDR1      5/175 JAMES LANE
                                                                ADDR2      HAPPYTOWN
                                                                STATE      CT
                                                                PCODE      2015-120
                                                                TELE       555 1234
                                                                PAYDT      00000000
                                                                PAYYTD     0000000.00
                                                                ALWYTD     0000015.00-
                                                                TRMDT      00000000
                                                                RTMDT      00000000
```

A database DELETE operation will show no details in the *Before Image* section of the report. The *After Image* section of the report will show the details of the database record that was deleted.

Sequence:	Journal entry sequence number
Date/Time:	The timestamp for when the entry was recorded in the journal
Type:	The database operation that caused the journal entry (in this case *DL* means database DELETE)
Job/User/Job number:	The name of the job that caused the journal entry to be recorded
Program:	The name of the program that caused the journal entry to be recorded
Object/Library/Member:	The name of the database object that caused the journal entry to be recorded
*RRN:	The relative record number of the database record that caused the journal entry to be recorded

NPRTJRNHST - Print Journal Images

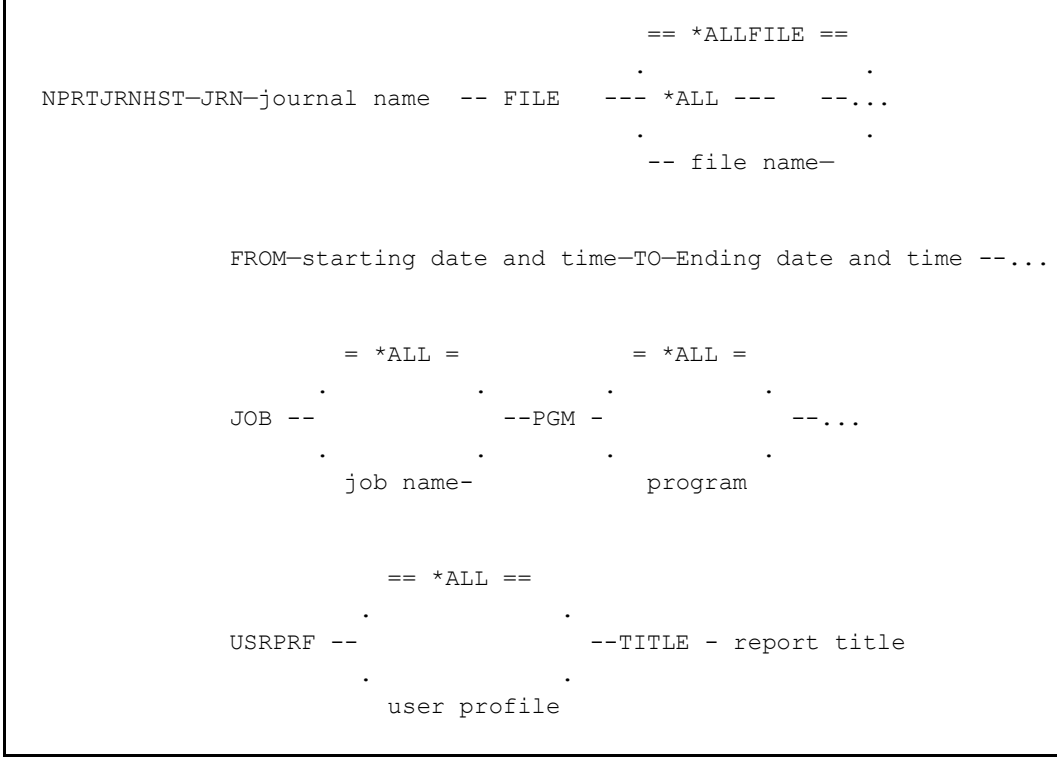
The Print Journal (NPRTJRNHST) command provides a 'front-end' to iSeries system journalling, in that it will provide formatted listings of entries processed within a specified timeframe.

The command is similar to (and uses the same processing engine as) the NUTIL NPRTJRN command. However this command can be used in all processing environments on the iSeries, whereas the NPRTJRN command can only be used interactively.

File record images are formatted and printed using the file field definition database that is used by the NUTIL Work with Database files (WRKDBF) utility.

You must remember that the command will only print journal entries that have been recorded in the specified journal; if a file is not journalled, or it is journalled to a different journal, it will not be included in the report.

Command Syntax



Parameter Definitions

JRN (Journal name) - specifies the qualified name of the journal to process. This should be the name of the journal used by the files in your application program. This is a required parameter.

FILE (Journalled physical file name) - specifies the qualified name of the physical database file that is to be reported in the Journalled Images report.

*ALLFILE - For all files attached to the journal, all journal entries in the current receiver of the specified journal will be reported.

*ALL - For all files (in the specified library only) attached to the journal, all journal entries in the current receiver of the specified journal will be reported. Library name must be specified.

File name - Specify the name of the file (which must be attached to the specified journal) to process.

PRTOPT (print option) - allows you to specify the basic level of detail to print.

*ALL will cause all fields in each journal entry to print (the fields that have changed will have the literal **** CHG **** printed beside them).

*CHG will print only key fields and fields that have changed.

OUTQ (output queue) - specifies the name of the output queue to use.

FROM (starting date and time) - specifies the date and time of the first journal entry to process.

A from date value of *CURRENT will use the system date as the starting date.

TO (ending date and time) - specifies the date and time of the last journal entry to process.

A to date value of *CURRENT will use the system date as the ending date.

JOB (job name) - specifies that processing will be limited to the journal entries for the qualified job name specified.

*ALL will signify the processing is not limited to journal entries for a specific job; the entry of a fully qualified job name (job name, user name and job number) will cause only the entries for that job to be printed.

PGM (program name) - specifies that processing will be limited to the journal entries caused by the processing of a specified program.

*ALL will signify the processing is not limited to journal entries for a specific program; the entry of a program name will cause only the entries for that job to be printed.

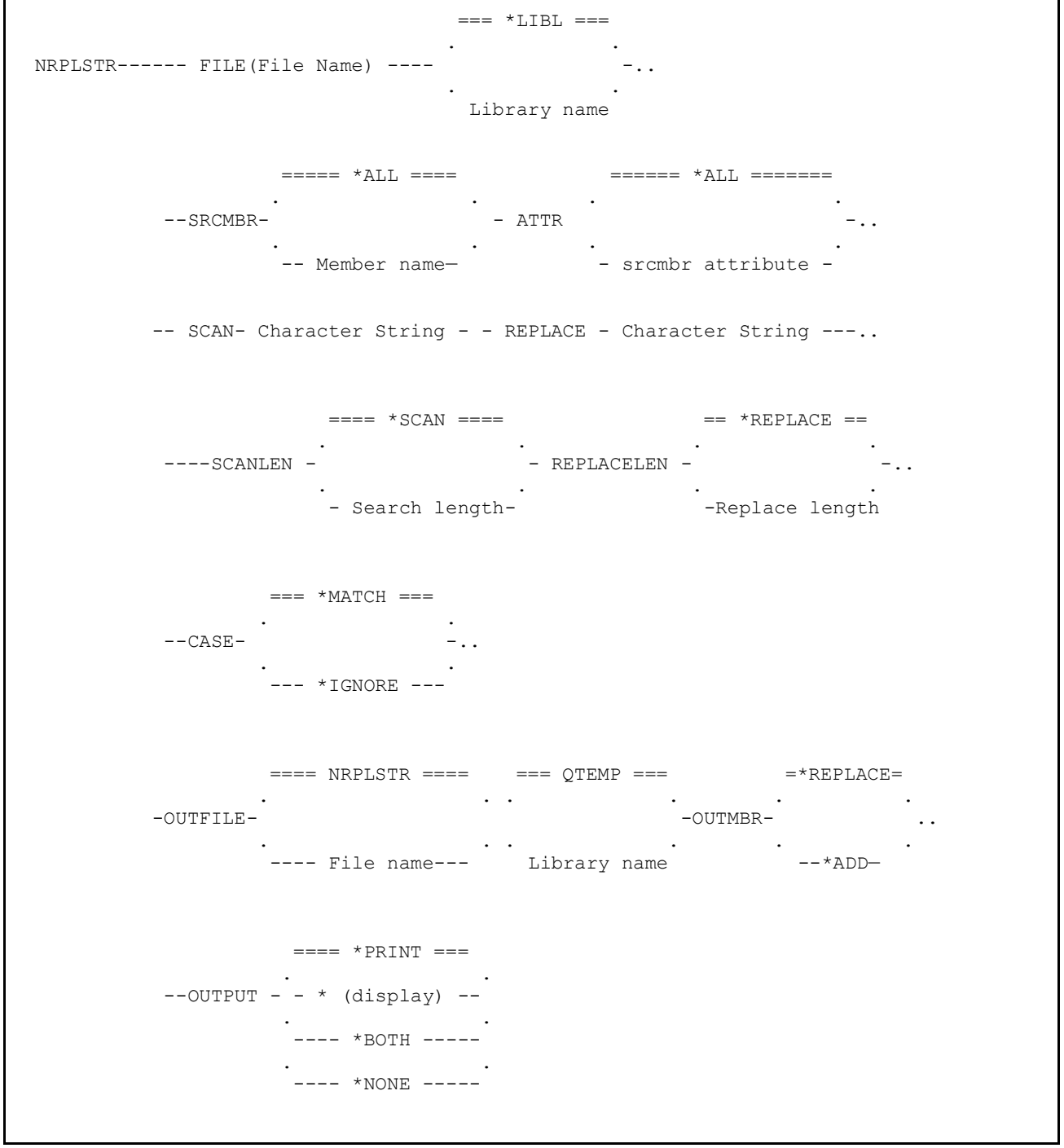
USRPRF (user profile) - specifies that processing will be limited to the journal entries created for a specified user profile name. The user name identifies the user profile under which the job was run that caused the journal entries to be created.

*ALL will signify that the processing is not limited to journal entries for a specific user profile; the entry of a user name will cause only the entries for that user profile to be printed.

NRPLSTR - Scan and Replace character string in Source file

This command works in a similar way to the NSCNSTR Scan String command (described later in this manual), but in this case, where a match is found on the SCAN string it is replaced using the contents of the REPLACE string.

Command Syntax



Parameter Definitions

Most parameter definitions are covered in the following examples, but the following explains ones not covered.

ATTR: This allows you to perform a selective replacement, based on the source member attribute. So, for example, you could perform a replacement in QDDSSRC, but only perform the replacement in printer files.

SCANLEN: This defines the length of the scan string. If *SCAN, then the length is the number of characters in the scan string (including embedded blanks).

For example if the scan string is 'DATE' then where *SCAN is specified the length used will be 4. To allow for trailing blanks, you must enter the length to be used. So for a scan string of 'DATE*bbb*' (where *b* signifies blank), you must enter a length of 7.

REPLACELEN: This defines the length of the replacement string. If *REPLACE, then the length is the number of characters in the replacement string (including embedded blanks).

For example if the replace string is 'DATE' then where *REPLACE is specified the length used will be 4. To allow for trailing blanks, you must enter the length to be used. So for a replace string of 'DATE*bbb*' (where *b* signifies blank), you must enter a length of 7.

If the scan length is *greater than or equal* to the replace length, then *all* occurrences of the scan string on the same source line will be replaced.

If the scan length is *less than* the replace length, then only the *first* occurrence of the scan string on the source line will be replaced.

CASE: This allows you to specify whether or not the scan is case sensitive:

*MATCH: The scan will be made for an exact match with the scan string specified.
*IGNORE: The scan will be based on the scan characters specified, but all characters will be converted to upper-case before the scan is made.

OUTFILE: Specifies the name and library of the database output file to which the output of the command is directed. The default file is QTEMP/NRPLSTR. The output file uses the NUTIL/NSCNWK00 file as a model for the output.

OUTMBR: Specifies how the output information is to be stored in the output file:

*REPLACE: Clear any existing file before starting to add data to it
*ADD: Add new records to the end of any existing data.

Examples of using NRPLSTR

```
NRPLSTR SRCFILE(PRODLIB/QRPGSRC) SRCMBR(*ALL) SCAN(CUSNAM)
        REPLACE(CUSNAM1)
```

This command will replace all occurrences of the character string ‘CUSNAM’ that exist in PRODLIB/QRPGSRC with the character string CUSNAM1 and print an audit list of all changes made.

```
NRPLSTR SRCFILE(PRODLIB/QRPGSRC) SRCMBR(*ALL) SCAN(CUSNAM)
        REPLACE(CUSNAM1) OUTFILE(MYLIB/REPLCUS)
```

This command will replace all occurrences of the character string ‘CUSNAM’ that exist in PRODLIB/QRPGSRC with the character string CUSNAM1 and print an audit list of all changes made.

In addition a file called REPLCUS will be created in library MYLIB containing the source records that matched the scan string.

NRTVJOBDA - Retrieve *JOBD Attributes

The Retrieve Job Description Attributes (NRTVJOBDA) command is used in a CL program to retrieve the value of one or more attributes defined within the job description and place the values into the specified CL variables.

For complete information about any of the returned values, you should refer to the IBM iSeries CL Reference Guide for details about the CRTJOBD (Create Job Description) command.

This command is valid only within a CL program.

Command Syntax

```
NRTVJOBDA—JOBD—job description name--...

               RTNLIB   - CL variable name for RTNLIB--...   10A
               USER     - CL variable name for USER--...     10A
               DATE     - CL variable name for DATE--...      8A
               SWS      - CL variable name for SWS--...       8A
               JOBQ     - CL variable name for JOBQ--...     10A
               JOBQLIB  - CL variable name for JOBQLIB--...  10A
               JOBPTY   - CL variable name for JOBPTY--...    1A
               HOLD     - CL variable name for HOLD--...     10A
               OUTQ     - CL variable name for OUTQ--...     10A
               OUTQLIB  - CL variable name for OUTQLIB--...  10A
               OUTPTY   - CL variable name for OUTPTY--...    1A
               PRTDEV   - CL variable name for PRTDEV--...   10A
               PRTTXT   - CL variable name for PRTTXT--...   30A
               INLLIBL  - CL variable name for INLLIBL--... 275A
               INLLIBL1 - CL variable name for INLLIBL1--. 2750A
               SYNTAX   - CL variable name for SYNTAX--...    6A
               ENDSEV   - CL variable name for ENDSEV--...    2P0
               LOGLEVEL - CL variable name for LOGLEVEL--...  1A
               LOGSEV   - CL variable name for LOGSEV--...    2P0
               LOGTEXT  - CL variable name for LOGTEXT--...  10A
               LOGCLPGM - CL variable name for LOGCLPGM--... 10A
               RTGDTA   - CL variable name for RTGDTA--...   80A
               RQSDTA   - CL variable name for RQSDTA--...  256A
               INQMSGRPY- CL variable name for INQMSGRPY--.. 10A
               DEVRCYACN- CL variable name for DEVRCYACN--.. 13A
               TSEPOOL  - CL variable name for TSEPOOL--...  10A
               ACGCDE   - CL variable name for ACGCDE--...   15A
               TEXT     - CL variable name for TEXT--...     50A
```

The return variable parameters allow you to specify a CL variable name which will return the relevant value to your program, for the job description specified.

You can request as many, or as few, of the return variables as you wish.

CL var for RTNLIB - Specifies the name of a variable used to return the name of the library that contains the job description.

If *LIBL or *CURLIB was specified for the name of the library in the JOBD parameter, the value returned is the name of the library where the job description was found.

Example of using the NRTVJOBDA command

```
DCL  VAR(&LIB)     TYPE(*CHAR) LEN(10)
DCL  VAR(&LIBLIST) TYPE(*CHAR) LEN(2750)

NRTVJOBDA  JOBD(*LIBL/MASTER) RTNLIB(&LIB) INLLIBL1(&LIBLIST)
```

The current job's library list will be used to locate the job description MASTER.

The actual library name where the object was found is retrieved and returned in program variable &LIB.

The library list defined on the job description is retrieved and returned in program variable &LIBLIST.

Notes on the retrieved Job Description Initial Library List

There are two variables that can be returned with the Job Description Library List:

- &INLLIBL is the variable that retrieves the first 25 libraries in the user portion of the job description library list
- &INLLIBL1 is the variable that retrieves the full 250 libraries in the user portion of the job description library list.

The &INLLIBL variable is retained only for purposes of backwards compatibility with previous user programming that references this command. All future use of the NRTVJOBDA command should reference the &INLLIBL1 parameter when retrieving the job description library list.

NRTVLIBL - Retrieve full job library list

The Retrieve Job Library List (NRTVLIBL) command is used in a CL program to retrieve information relating to the library list of the current job. This is used for Version 5 of OS/400, where a job library list has been expanded in comparison to earlier OS/400 releases. You can no longer use the standard IBM RTVJOBA command to retrieve the full library list after Version 5 has been installed.

The command has no parameters and is valid only within a CL program.

Command Syntax

```
NRTVLIBL--...

               SYSLIBL   - CL variable name for RTNLIB--...   165A
               PRDLIBL   - CL variable name for PRDLIBL--...   22A
               CURLIB    - CL variable name for CURLIB--...    11A
               USRLIBL   - CL variable name for USRLIBL--... 2750A
```

The return variable parameters allow you to specify a CL variable name which will be used to return the relevant value for the current job.

You can request as many, or as few, of the return variables as you wish.

CL var for SYSLIBL - Specifies the name of a variable used to return the system portion of the job library list. A maximum of 15 library entries can be returned. Each library in the list will be separated by a blank space.

CL var for PRDLIBL - Specifies the name of a variable used to return the product portion of the job library list. A maximum of 2 library entries can be returned. Each library in the list will be separated by a blank space.

CL var for CURLIB - Specifies the name of a variable used to return the current library portion of the job library list. A maximum of 1 library entry can be returned. Each library in the list will be separated by a blank space.

CL var for USRLIBL - Specifies the name of a variable used to return the user portion of the job library list. A maximum of 250 library entries can be returned. Each library in the list will be separated by a blank space.

NRTVSPLFA - Retrieve Spoolfile Attributes

The Retrieve Spoolfile Attributes (NRTVSPLFA) command is used in a CL program to retrieve the value of one or more attributes of a specified spoolfile and place the values into the specified CL variables.

For complete information about any of the returned values, you should refer to the IBM iSeries CL Reference Guide for details about the WRKSPLFA (Work with Spoolfile Attributes) command. The definition of spoolfile attributes can be found in the OVRPRTF (Override with Printer File) command discussion.

This command is valid only within a CL program.

Command Syntax

```
NRTVSPLFA—FILE—spool file name--...

                    ==== * ====                 == *LAST ==
                .                .            .              .
          JOB -                   - SPLNBR --  -- *ONLY --   -...
                .                .            .              .
                 number/user/name             -spool number-


  FORMTYPE  - CL variable name for FORMTYPE--...   10A
  SCHEDULE  - CL variable name for SCHEDULE--...   10A
  STATUS    - CL variable name for STATUS--...     10A
  PAGES     - CL variable name for PAGES--...       5P0
  DEVTYPE   - CL variable name for DEVTYPE--...    10A
  ACGCDE    - CL variable name for ACGCDE--...     15A
  USRDTA    - CL variable name for USRDTA--...     10A
  USRDFNDTA - CL variable name for USRDFNDTA--... 255A
  COPIES    - CL variable name for COPIES--...      2P0
  LPI       - CL variable name for LPI--...         5P1
  CPI       - CL variable name for CPI--...         5P1
  PAGELEN   - CL variable name for PAGELEN--...     3P0
  PAGEWIDTH - CL variable name for PAGEWIDTH--...   3P0
  MSRMETHOD - CL variable name for MSRMETHOD--...   7A
  OVRFLW    - CL variable name for OVRFLW--...      3P0
  PAGRTT    - CL variable name for PAGRTT--...      5A
  JUSTIFY   - CL variable name for JUSTIFY--...     3P0
  DUPLEX    - CL variable name for DUPLEX--...      7A
  FOLD      - CL variable name for FOLD--...        4A
  ALIGN     - CL variable name for ALIGN--...       4A
  PRTQLTY   - CL variable name for PRTQLTY--...    10A
  FORMFEED  - CL variable name for FORMFEED--...    8A
  DRAWER    - CL variable name for DRAWER--...      3A
  MULTIUP   - CL variable name for MULTIUP--...     3P0
  UOM       - CL variable name for UOM--...         5A
  PRTTXT    - CL variable name for PRTTXT--...     30A
  FILESEP   - CL variable name for FILESEP--...     3P0
  REDUCE    - CL variable name for REDUCE--...      5A
```

Parameter Definitions

FILE: Specifies the name of the spooled file for which you wish to retrieve the attributes. This is a required parameter.

JOB: Specifies the name of the job that created the spooled file. The possible values are:

* The job that entered this command is the job that created the spooled file

job-name: Specify the name of the job that created the spooled file. If no job qualifier is given, all jobs currently in the system ar searched for the simple name of the job.

user-name: Specify the user name that identifies the user profile under which the job is run.
job-number: Specify the system-assigned job number.

SPLNBR: Specifies the unique number of the spooled file in the job whose attributes are to be retrieved. The possible values are:

*ONLY- Only one spooled file in the job has the specified file name; therefore, the number of the spooled file is not necessary.
*LAST- The spooled file with the highest number and the specified file name is used.
spooled-file-number- Specify the number of the spooled file with the specified file name whose attributes are displayed.

Return Variables

The return variable parameters allow you to specify a CL variable name, which will be used to return the relevant value for the spoolfile specified. You can request as many, or as few, of the return variables as you wish.

Example of using the NRTVSPLFA command

```
DCL  VAR(&USRDTA)   TYPE(*CHAR) LEN(10)

NRTVSPLFA  FILE(WRKDBFP) JOB(*) SPLNBR(*LAST) USRDTA(&USRDTA)
```

This example will retrieve the attributes of the most recent spoolfile named WRKDBFP that has been created in the current job. The contents of the User Data field for the spoolfile will be returned to the program in variable &USRDTA.

NRTVSQLSRC - Retrieve SQL source

The NRTVSQLSRC command allows you to create equivalent SQL source statements for an existing database file.

Command Syntax

```
   NRTVSQLSRC—FILE—file name--...

                         === QSQLSRC ===      === *LIBL ===
                       .                  .   .              .
          -- SRCFILE--                     -                  - ...
                       .                  .   .              .
                        -- source file --     -library name-


          -- SRCMBR-- source member name- ...


                          === *FILE ===               = *SYS =
                        .                  .        .         .
           -- DFTRDBCOL--                    - NAMING--          - ...
                        .                  .        .         .
                         -collection name-            - *SQL -


                        == *DFT ==
                       .           .
           -- RTVOPT--  -  *VIEW -
                       .           .
                        - *INDEX -
```

Parameter Definitions

FILE: Specifies the name and library of the database file to process. This is a required parameter.

SRCFILE: Specifies the name and library of the previously created database source file into which the SQL source statements are being written.

SRCMBR: Specifies the name of the database source file member into which the SQL source statements are being written. If the name is not specified, the database file name is used. If the member existed before the command was run, it is cleared before any source statements are written into it. If the member did not exist, it is created.

If the default entry of *FILE is specified, the name of the database file is used as the member name.

DFTRDBCOL: Specifies the name of the collection identifier used for qualifying names of the tables, views, and indexes. If the default entry of *FILE is specified, the collection name is the same as the library name of the database file specified in the FILE parameter.

NAMING: Specifies the naming convention to use when qualifying the SQL object name.

*SYS - the iSeries naming convention (collection/object) is used.
*SQL - the SQL naming convention (collection.object) is used.

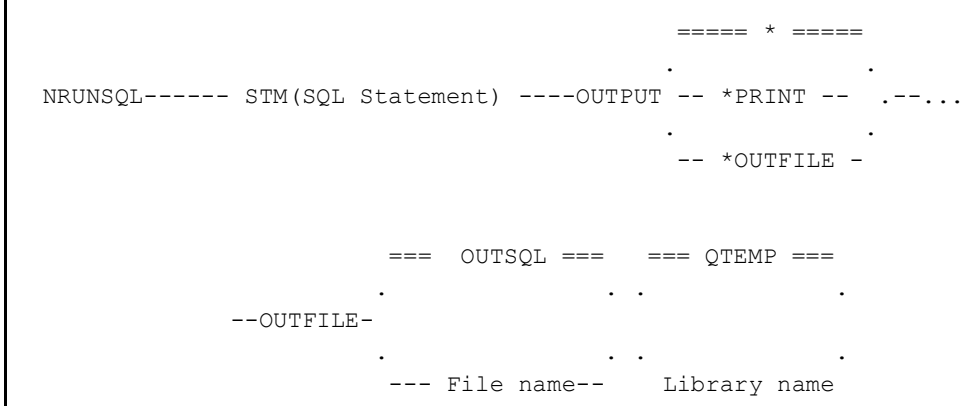
RTVOPT: Specifies, when a logical file is being processed, whether to retrieve the key definition or the select/omit criteria. The parameter is ignored if the database object is not a logical file.

*DFT - Where the logical file is keyed, with no select/omit criteria defined, an SQL index will be generated. Otherwise a view will be created. If the logical file is keyed and also has select/omit criteria defined, a view will be created but a warning message will also be issued.
*VIEW - The Select/Omit criteria should be used to create an SQL view.
*INDEX - The Key field criteria should be used to create an SQL index.

NRUNSQL - Run an SQL statement

The NRUNSQL (Run SQL statement) command allows you to run an interactive SQL statement regardless of whether the IBM DB2/400 SQL program product is installed on the iSeries.

Command Syntax



Parameter Definitions

STM (SQL Statement) - Specify a valid interactive SQL statement. Refer to the IBM iSeries SQL Reference Guide for a full discussion of the Interactive (Structured Query Language) statements.

OUTPUT - Specifies, *for an SQL SELECT statement only*, where the output is to be sent; either to display, print, or to a database file.

OUTFILE - Where OUTPUT(*OUTFILE) was requested for an SQL SELECT, this parameter specifies the name of the database file to receive the output.

Basic SQL info for the NRUNSQL command

The following provides a brief review of the more common SQL statements, which can be used with the NRUNSQL command.

THE INSERT STATEMENT

INSERT can be used to add a new record to a file.

```
INSERT INTO file_name (field1, field2, field3)
VALUES (value1, value2, value3)
```

The INTO clause names the fields that are to be loaded. The VALUES clause specifies a value to insert into the related field.

General convention requires that you specify each field in the record format and a value to be assigned to each field.

THE DELETE STATEMENT

DELETE can be used to delete records from a file.

```
DELETE FROM file_name WHERE field1 = value1
```

The WHERE clause specifies the conditions under which a record will be deleted from the file. If you do not supply a WHERE clause, all records in the file will be deleted.

```
DELETE from EMPMAST   where STATUS = 'D'
DELETE from PARTSFILE where USECOUNT < 5
```

THE UPDATE STATEMENT

UPDATE can be used to update records in a file.

```
UPDATE file_name SET field1 = value1, field2 = value2
WHERE field3 = value3
```

The SET clause specifies the fields you want to update and the values you wish to assigned to the updated fields. The value you assign can be

...another field name, so you can replace the contents of field1 with the contents of field2

SET field8 = field9

...a constant

SET field8 = 'A'

...a null value, using the keyword NULL (the field must be defined as NULL capable)

SET field8 = NULL

...an expression, where the results of the expression will be loaded into the field

SET field8 = field9 + field10

The WHERE clause specifies the conditions under which a record will be updated in the file. If you do not supply a WHERE clause, all records in the file will be updated.

```
UPDATE EMPMAST  set SALARY = 0  where STATUS = 'D'
```

The SELECT statement

SELECT can be used to view records in a file.

```
SELECT field1, field2 FROM file_name WHERE field3 = value3
SELECT * FROM file_name WHERE field1 = value1
```

The first form of the statement will retrieve only the fields specified, whereas the second form (SELECT *) will retrieve all fields.

The WHERE clause specifies the conditions under which a record will be retrieved from the file. If you do not supply a WHERE clause, all records in the file will be retrieved.

```
SELECT * from EMPMAST   where SALARY > 25000
```

OUTPUT(*PRINT) is only valid if the SQL statement parameter specified includes a SELECT statement.

General rules within SQL statement definition

Basic predicates compare two values; valid ones are as follows:

Predicate	Purpose
field1 = field2	field1 is equal to field2
field1 ¬= field2	field1 is not equal to field2
field1 <> field2	field1 is not equal to field2
field1 < field2	field1 is less than field2
field1 > field2	field1 is greater than field2
field1 >= field2	field1 is greater than or equal to field2
field1 <= field2	field1 is less than or equal to field2
field1 ¬< field2	field1 is not less than field2
field1 ¬> field2	field1 is not greater than field2
field1 <> field2	field1 is not equal to field2
field1 <> field2	field1 is not equal to field2

Example uses of the NRUNSQL command

ACCESSING A SPECIFIC FILE MEMBER

"True SQL" has no real understanding of the OS/400 concept of multi-member files; as such it assumes you will be working on the first member of the file.

You can access a specific member in a multimember file by creating an ALIAS that points to it. In the following example we are accessing member ACCTS2004 in file ACCTSMAST:

```
NRUNSQL 'create alias QTEMP/accts for ACCTSMAST(ACCTS2004)'

NRUNSQL 'select * from QTEMP/accts'

NRUNSQL 'Drop alias QTEMP/accts'
```

Step one creates the Alias, which directs access to the specific member. Step two performs the required processing on the member. Step 3 closes the Alias that was created in Step 1.

USING SQL TO ACCESS A FILE BY RELATIVE RECORD NUMBER (RRN)

Display first 10 rows (records) of data:

```
NRUNSQL STM('SELECT * FROM file_name fetch first 10 rows only')
```

Select by RRN:

```
NRUNSQL STM('SELECT * FROM file_name WHERE rrn(file_name) = 5')
```

Order by RRN and show RRN:

```
NRUNSQL STM('SELECT rrn(file_name), field_1, field_2, field_n FROM
file_name order by rrn(file_name)')
```

USING SQL TO UPDATE VIA RELATIVE RECORD NUMBER (RRN)

Get the RRN:

```
SELECT RRN(file_name) as storedRRN, file_name.* FROM file_name
       WHERE criteria_to_access_record
```

Update by RRN

```
UPDATE file_name
       SET fields_to_update
       WHERE RRN(file_name) = storedRRN
```

NSCNDBFDAT - Scan database file for possible date fields

The NSCNDBFDAT command allows you to scan a database file definition for possible date fields. The results of the analysis will be supplied on a printed report.

The scan logic is as follows:

- Scan for fields where the field text, field name, column headings or field alias name contains one of the common text strings normally used for naming date fields: DAT, DTE, DT, DATE, YEAR, MONTH, DAY, YR, YY, MTH, MM, DAY, DY, DD.
- Scan for fields of a specified length (the default is to detect field length 6 and 7)
- Scan for fields that have EDTCDE(Y) defined for them
- Scan for fields that have EDTWRD(‘ / / ‘), EDTWRD(‘ - - ‘) or EDTWRD(‘ . . ‘) defined for them.

A list of the results will be printed (and optionally loaded into an output file) and a *match field* column on the report will advise you of the scan condition detected.

Command Syntax

Parameter Definitions

FILE: Specifies the name, or generic name, of the database file(s) to be processed. A special value of *ALL for the file name allows you to scan all database files in the specified library. This is a required parameter.

The possible library values are:

> *LIBL: All the libraries in the user and system portions of the job's library list are searched.
> *CURLIB: The current library for the job is searched. If no current entry exists in the library list, QGPL is used.
> Library name: Specify the name of the library to be searched.

MSCFILE: Specify the name of the file containing the multiscan source member, for multiscan searches. The default file is NUTIL/QSCANSRC.

MSCMBR: Specifies the name of the source member containing the multiscan scanning criteria to be used. Refer to the command description for the NSCNSTR (Scan String) command for an explanation on how to define multiscan source members. The possible member values are:

> NSCNDAT: The source member supplied by Navan for use in this command is to be used.
> Member name: Specify the name of the source member containing the multiscan scanning criteria to be used. Examples of multiscan source members can be found in the source file NUTIL/QSCANSRC.

OUTFILE: Specifies the name and library of the database output file to which the output of the command is directed. The default file is QTEMP/NSCNDBFDAT. The output file uses the NUTIL/NSCNWK10 file as a model for the output.

OUTMBR: Specifies how the output information is to be stored in the output file. *REPLACE will clear any existing file before starting to add data to it; *ADD will add new records to the end of any existing data.

FLDCHK*n*: Specifies the field lengths that should be scanned for and reported:

> The FLDCHK2 parameter will detect fields defined as 2A (alpha), 2S0 (signed numeric) or 2P0 (packed numeric)
>
> The FLDCHK3 parameter will detect fields defined as 3A (alpha), 3S0 (signed numeric) or 3P0 (packed numeric)
>
> The FLDCHK4 parameter will detect fields defined as 4A (alpha), 4S0 (signed numeric) or 4P0 (packed numeric)
>
> The FLDCHK5 parameter will detect fields defined as 5A (alpha), 5S0 (signed numeric) or 5P0 (packed numeric)
>
> The FLDCHK6 parameter will detect fields defined as 6A (alpha), 6S0 (signed numeric) or 6P0 (packed numeric)
>
> The FLDCHK7 parameter will detect fields defined as 7A (alpha), 7S0 (signed numeric) or 7P0 (packed numeric)
>
> The FLDCHK8 parameter will detect fields defined as 8A (alpha), 8S0 (signed numeric) or 8P0 (packed numeric)
>
> The FLDCHK9 parameter will detect fields defined as 9A (alpha), 9S0 (signed numeric) or 9P0 (packed numeric)

NSCNOBJDAT - Scan object for possible date fields

The NSCNOBJDAT command allows you to scan a device file (printer file, display file, etc) definition for possible date fields. The results of the analysis will be supplied on a printed report.

The scan logic is as follows:

- Scan for fields where the field text, field name, column headings or field alias name contains one of the common text strings normally used for naming date fields: DAT, DTE, DT, DATE, YEAR, MONTH, DAY, YR, YY, MTH, MM, DAY, DY, DD.
- Scan for fields of a specified length (the default is to detect field length 6 and 7)
- Scan for fields that have EDTCDE(Y) defined for them
- Scan for fields that have EDTWRD(‘ / / ‘), EDTWRD(‘ - - ‘) or EDTWRD(‘ . . ‘) defined for them.

A list of the results will be printed (and optionally loaded into an output file) and a *match field* column on the report will advise you of the scan condition detected.

Command Syntax

Parameter Definitions

FILE: Specifies the name, or generic name, of the device file(s) to be processed. A special value of *ALL for the file name allows you to scan all device files in the specified library. This is a required parameter.

The possible library values are:

*LIBL:	All the libraries in the user and system portions of the job's library list are searched.
*CURLIB:	The current library for the job is searched. If no current entry exists in the library list, QGPL is used.
Library name:	Specify the name of the library to be searched.

MSCFILE: Specify the name of the file containing the multiscan source member, for multiscan searches. The default file is NUTIL/QSCANSRC.

MSCMBR: Specifies the name of the source member containing the multiscan scanning criteria to be used. Refer to the command description for the NSCNSTR (Scan String) command for an explanation on how to define multiscan source members. The possible member values are:

NSCNDAT:	The source member supplied by Navan for use in this command is to be used.
Member name:	Specify the name of the source member containing the multiscan scanning criteria to be used. Examples of multiscan source members can be found in the source file NUTIL/QSCANSRC.

OUTFILE: Specifies the name and library of the database output file to which the output of the command is directed. The default file is QTEMP/NSCNOBJDAT. The output file uses the NUTIL/NSCNWK10 file as a model for the output.

OUTMBR: Specifies how the output information is to be stored in the output file. *REPLACE will clear any existing file before starting to add data to it; *ADD will add new records to the end of any existing data.

FLDCHK*n*: Specifies the field lengths that should be scanned for and reported:

The FLDCHK2 parameter will detect fields defined as 2A (alpha), 2S0 (signed numeric) or 2P0 (packed numeric)

The FLDCHK3 parameter will detect fields defined as 3A (alpha), 3S0 (signed numeric) or 3P0 (packed numeric)

The FLDCHK4 parameter will detect fields defined as 4A (alpha), 4S0 (signed numeric) or 4P0 (packed numeric)

The FLDCHK5 parameter will detect fields defined as 5A (alpha), 5S0 (signed numeric) or 5P0 (packed numeric)

The FLDCHK6 parameter will detect fields defined as 6A (alpha), 6S0 (signed numeric) or 6P0 (packed numeric)

The FLDCHK7 parameter will detect fields defined as 7A (alpha), 7S0 (signed numeric) or 7P0 (packed numeric)

The FLDCHK8 parameter will detect fields defined as 8A (alpha), 8S0 (signed numeric) or 8P0 (packed numeric)

The FLDCHK9 parameter will detect fields defined as 9A (alpha), 9S0 (signed numeric) or 9P0 (packed numeric)

NSCNSRCDAT - Scan source file for possible date fields

The NSCNSRCDAT command allows you to scan a source file member for possible date fields. The results of the analysis will be supplied on a printed report.

The scan logic is as follows:

- Scan for the use of one of the common text strings normally used for naming date fields: DAT, DTE, DT, DATE, YEAR, MONTH, DAY, YR, YY, MTH, MM, DAY, DY, DD.
- Optionally scan for internally defined fields (in RPG source)
- Optionally omit comment lines from the scan

A list of the results will be printed (and optionally loaded into an output file) and a match field will advise you of the scan condition detected.

Command Syntax

Parameter Definitions

FILE: Specifies the name of the source file to be processed. A special value of *ALL for the file name allows you to scan all source file in the specified library. This is a required parameter.

The possible library values are:

*LIBL:	All the libraries in the user and system portions of the job's library list are searched.
*CURLIB:	The current library for the job is searched. If no current entry exists in the library list, QGPL is used.
Library name:	Specify the name of the library to be searched.

MBR: The Member Name parameter allows you to specify the name of the source member to be processed.

The possible member values are:

*ALL:	Scan all source members in the specified source file.
*FIRST:	The first member in the specified file is processed.
Member name:	Specify the name of the member to be processed.

MSCFILE: Specify the name of the file containing the multiscan source member, for multiscan searches. The default file is NUTIL/QSCANSRC.

MSCMBR: Specifies the name of the source member containing the multiscan scanning criteria to be used. Refer to the command description for the NSCNSTR (Scan String) command for an explanation on how to define multiscan source members. The possible member values are:

NSCNDAT:	The source member supplied by Navan for use in this command is to be used.
Member name:	Specify the name of the source member containing the multiscan scanning criteria to be used. Examples of multiscan source members can be found in the source file NUTIL/QSCANSRC.

OUTFILE: Specifies the name and library of the database output file to which the output of the command is directed. The default file is QTEMP/NSCNSRCDAT. The output file uses the NUTIL/NSCNWK00 file as a model for the output.

OUTMBR: Specifies how the output information is to be stored in the output file. *REPLACE will clear any existing file before starting to add data to it; *ADD will add new records to the end of any existing data.

OPTION: Specifies processing options to customise the scan. The possible option values are:

*OMTDDSCMT: Omit DDS Comments. DDS Comment lines will not be included in the scan.

*OMTRPGCMT: Omit RPG Comments. RPG Comment lines will not be included in the scan.

RPGDFN: Specifies which RPG syntax should be used when scanning the source member(s). This then allows the command to analyse any program-defined field definitions:

*NONE: Do not check RPG syntax. The specified source member(s) do not contain RPG source code.

*OPM: Perform OPM RPG syntax checks. The RPG column positions are based on the standard RPG program specifications used for S/36, S/38 and RPGIII source.

*ILE: Perform ILE RPG syntax checks. The RPG column positions are based on the ILE RPG/400 program specifications.

RPGCHK*n*: Specifies whether the RPG scan should report fields that are internally defined. Note that these parameters are only used in the command processing if the RPGDFN parameter is either *OPM or *ILE:

The RPGCHK2 parameter will detect fields defined as 2A (alpha), 2S0 (signed numeric) or 2P0 (packed numeric)

The RPGCHK3 parameter will detect fields defined as 3A (alpha), 3S0 (signed numeric) or 3P0 (packed numeric)

The RPGCHK4 parameter will detect fields defined as 4A (alpha), 4S0 (signed numeric) or 4P0 (packed numeric)

The RPGCHK5 parameter will detect fields defined as 5A (alpha), 5S0 (signed numeric) or 5P0 (packed numeric)

The RPGCHK6 parameter will detect fields defined as 6A (alpha), 6S0 (signed numeric) or 6P0 (packed numeric)

The RPGCHK7 parameter will detect fields defined as 7A (alpha), 7S0 (signed numeric) or 7P0 (packed numeric)

The RPGCHK8 parameter will detect fields defined as 8A (alpha), 8S0 (signed numeric) or 8P0 (packed numeric)

The RPGCHK9 parameter will detect fields defined as 9A (alpha), 9S0 (signed numeric) or 9P0 (packed numeric)

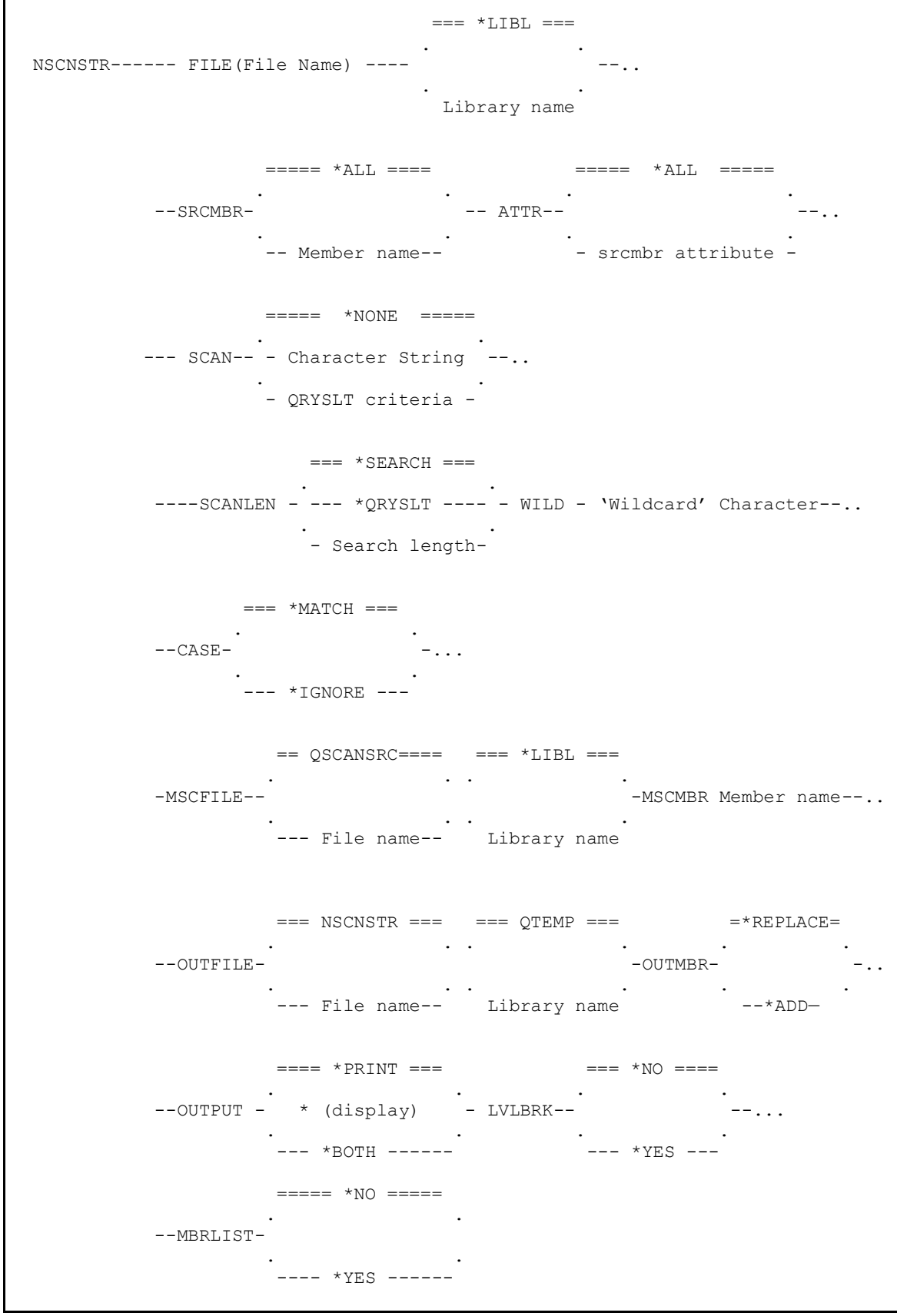
Internally defined fields will be marked on the report (and in the *OUTFILE) with the match string shown as '*Internal RPG FldLen Defn'*.

NSCNSTR - Scan Source file for a given character string

This command will print a list of all occurrences of a given character string within a source member/file.

The search may be either by single character string, multiple character strings (contained in a named MultiScan source member) or by QRYSLT (query selection) criteria.

Command Syntax



Parameter Definitions

Most parameter definitions are covered in the following examples, but the following explains ones not covered.

SCANLEN: Where a scan string is used, this defines the length of the string. If *SEARCH, then the length is the actual number of characters in the scan string.

For example if the scan string is 'DATE' then where *SEARCH is specified the length used will be 4. To allow for trailing blanks, you must enter the length to be used. So for a scan string of 'DATE*bbb*' (where *b* signifies blank), you must enter a length of 7.

CASE: This allows you to specify whether or not the scan is case sensitive:

*MATCH: The scan will be made for an exact match with the scan string specified.

*IGNORE: The scan will be based on the scan characters specified, but all characters will be converted to upper-case before the scan is made.

MBRLIST: Do you require a list showing the names of all source members scanned? Defaults to no.

LVLBRK: Do you require a change of page every time a new member is scanned? This defaults to no, in which case only a line skip is performed.

MSCFILE: Specify the name of the file containing the multiscan source member, for multiscan searches. This parameter is only valid if SCAN(*NONE) is specified. The default file is NUTIL/QSCANSRC.

MSCMBR: Specifies the name of the source member containing the multiscan scanning criteria to be used. This parameter is only valid if SCAN(*NONE) is specified. See below for an explanation on how to define multiscan source members. Specify the name of the source member containing the multiscan scanning criteria to be used. Examples of multiscan source members can be found in the source file NUTIL/QSCANSRC

OUTFILE: Specifies the name and library of the database output file to which the output of the command is directed. The default file is QTEMP/NSCNSTR. The output file uses the NUTIL/NSCNWK00 file as a model for the output.

OUTMBR: Specifies how the output information is to be stored in the output file:

*REPLACE: Clear any existing file before starting to add data to it

*ADD: Add new records to the end of any existing data.

OUTPUT: Specifies where the output of the command is to be directed. Regardless of the entry made here, the output will always exist in the OUTFILE specified above.

*PRINT - The results of the command will be printed on a report.

* - The results of the command will be displayed at the workstation, for an interactive job.

*BOTH - The results of the command will be printed on a report and, for an interactive job, they will also be displayed at the workstation.

For OUTPUT(*PRINT) and OUTPUT(*BOTH), two reports will be printed. The first report will print a detailed list of every match on found. The second report will print a summary list, showing a count of lines detected.

Examples of using NSCNSTR by scan

The following examples use the IBM QCLSCAN facility (refer IBM iSeries Programmers Guide) to determine whether the search match is met.

```
NSCNSTR FILE(PRODLIB/QRPGSRC) SRCMBR(*ALL) SCAN(CUSNAM)
```

This command will print a list of all occurrences of the character string ‘CUSNAM’ that exist in PRODLIB/QRPGSRC.

```
NSCNSTR FILE(PRODLIB/QRPGSRC) SRCMBR(*ALL) SCAN(CUSNAM)
OUTFILE(MYLIB/SCANCUS)
```

This command will print a list of all occurrences of the character string ‘CUSNAM’ that exist in PRODLIB/QRPGSRC. In addition a file called SCANCUS will be created in library MYLIB containing the source records that matched the scan string.

```
NSCNSTR FILE(PRODLIB/QRPGSRC) SRCMBR(*ALL) SCAN(Z-ADD#DATE)
WILD(#)
```

This command will print a list of all occurrences of the character string `'Z-ADD#DATE'` that exist in PRODLIB/QRPGSRC. The WILD(#) will signify that any character in that position will be accepted as valid for the scan argument.

Thus a match would be found on `'Z-ADDTDATE'` as well as `'Z-ADDUDATE'`.

```
NSCNSTR FILE(PRODLIB/QRPGSRC) SRCMBR(*ALL) SCAN(*NONE) +
MSCFILE(MYLIB/QSCANSRC) MSCMBR(DATES)
```

This command will perform a ‘MultiScan’ multiple scan search and print a list of all occurrences of all scan arguments listed in source member DATES in file MYLIB/QSCANSRC.

For an example of setting up a MultiScan multiple scan argument source member, some have been set up in NUTIL/QSCANSRC. You can specify up to 200 ‘Include’ arguments and up to 200 ‘Exclude’ arguments in a single MultiScan source member (if you specify more, only the first 200 specified will be used).

NSCNSTR using Open Query File

NSCNSTR has also been interfaced to the IBM Open Query File function so you can optionally search the source data at record level.

Specifying *QRYSLT as the SCANLEN parameter automatically invokes OPNQRYF, and the SCAN parameter becomes an OPNQRYF QRYSLT (query Selection Criteria) parameter. In this case, the entries made in the SCAN parameter of the NSCNSTR command would be bound by the rules of the QRYSLT parameter of the IBM OPNQRYF command. Refer to the IBM iSeries Programmers Guide for more information on the OPNQRYF command.

The source file field names used for QRYSLT on a source member are:

Field Name	*Description of field*
SRCSEQ	Line sequence number
SRCLIN	Source line data (If searching for a character string you should use the *CT operator)
SRCDAT	Source change date (in YYMMDD)

*EXAMPLES OF USING NSCNSTR VIA THE *QRYSLT OPTION*

```
NSCNSTR PRODLIB/QRPGSRC  SRCMBR(*ALL) SCAN('SRCDAT *GE 971215') +
SCANLEN(*QRYSLT)
```

This command will use OPNQRYF to print a list of all source lines in PRODLIB/QRPGSRC that have a revision date of greater than December 15, 1997.

```
NSCNSTR PRODLIB/QRPGSRC  SRCMBR(*ALL) SCAN('SRCDTA *CT UDATE') +
SCANLEN(*QRYSLT)
```

This command will use OPNQRYF to print a list of all source lines in PRODLIB/QRPGSRC that have the text string 'UDATE' in them.

```
NSCNSTR PRODLIB/QRPGSRC  SCAN('SRCDTA *CT "MOVEL*HIVAL" *OR +
SRCDTA *CT "MOVEL*LOVAL")  SCANLEN(*QRYSLT)
```

This command will use OPNQRYF to print a list of all source lines in PRODLIB/QRPGSRC that have the text string 'MOVEL*HIVAL' or the text string 'MOVEL*LOVAL' in them.

Defining MultiScan source members

The MultiScan feature of NSCNSTR allows you to define more complex scan criteria. The search arguments are not actually specified on the command, they must be defined in a source physical file source member that you then assign to the scan. You can specify up to 200 'Include' arguments and up to 200 'Exclude' arguments in a single MultiScan source member (if you specify more, only the first 200 includes and the first 200 excludes that are specified will be used).

You define a NSCNSTR scan as a MultiScan search by specifying SCAN(*NONE) for the scan parameter and entering the MSCFILE and MSCMBR parameters.

For an example of setting up a MultiScan multiple scan argument source member, some have been set up in NUTIL/QSCANSRC. The layout for the source line source data is as follows:

From	To	Type	Description
1	2	numeric	The length of the scan argument
3	3	alpha	The argument type, either I=Include or E=Exclude
4	73	alphanumeric	The scan argument

As an example, this is an excerpt from the scan member MESSAGES, which scans for the use of messaging commands in CL programs:

```
* I         1         2         3         4         5
*nE12345678901234567890123456789012345678901234567890123
**************************************************
*Scan Excludes
*Scan Includes
09ISNDPGMMSG
06ISNDMSG
06ISNDRPY
05IMSGID
06IMSGDTA
```

There are no excludes defined in this example scan.

The first include argument specifies that a scan should be made for the string 'SNDPGMMSG' (which has a length of 9 bytes. The second specifies a scan for the string 'SNDMSG' which is 6 characters long.

The purpose of the scan length is to allow you to specify leading or trailing blanks in the scan criteria:

```
09Idiscount
```

This example would find a match with the string 'Ex-discount ' but would not match to the string 'discounted' because the lack of a trailing blank causes a no-match.

NSNDSMTPML – Send SMTP Mail

The Send SMTP Mail (NSNDSMTPML) command allows you to send an "Internet compliant" mail from the iSeries.

Command Syntax

```
NSNDSMTPML------ FROM- mail address; description- ..

                                                 ==== *TO ====
                                                .             .
                TO- mail address; description- .. ---- *CC ---- - ...
                                                .             .
                                                 ==== *BCC ===

                  === NSNDSMTPML ===
                 .                  .
          --FILE-                    -... MBR- member name -...
                 .                  .
                  --- file name ----


          --SUBJECT- subject line - ...


          --ATTACHMENT- path name - ...


                      == *LOCALHOST ==
                     .                .
          --SMTPHOST-
                     .                .
                      --- host name---
```

The command works with the SMTP (mail) server directly by TCP/IP socket interface. You should enable TCP/IP and configure domain/host name information on the iSeries before running the command. You should also configure the iSeries to use a DNS (Domain Name System) server or 'hosts' file for the command to resolve IP address from TCP/IP host name.

System date/time is used when generating the date information in the mail header. Make sure that system values QDATE, QTIME and QUTCOFFSET are set correctly.

The command creates a temporary stream file in directory specified in the TMPDIR parameter. The path name will be: '/(tmpdir)/SM_(job number)-(user name)-(job name)_(date).(time).TXT'

For example:

'/tmp/SM_90605-TCPIP-QPADEV0006_1999-01-20-17.33.19.215.TXT'

Wait time (time out interval) will depend on the socket communication phase. Usually, the command will "time out" in 1 to 3 minutes if the SMTP server doesn't respond.

Restrictions:

1. Mail body text (physical file member specified by the FILE parameter) and attachment files (stream file(s) specified by the ATTACHMENT parameter) can be sent. Attachment files cannot be QSYS.LIB file system objects.
2. The maximum number of recipients ('to' and 'cc' and 'bcc') is 30.
3. The 'Content-type' of the mail body will be set to 'text/plain'. Other text formats such as HTML or RTF are not supported. For attachment files, 'Content-type' will be set to 'application/octet-stream'.
4. For (safe) US-ASCII characters, no encoding will take place. For unsafe ASCII and 8bit ISO-8859-1 characters, Q-encoding will be used to encode mail header and quoted-printable encoding will be used for mail body. For Japanese characters, B-encoding will be used to encode mail header and body text will be converted to ISO-2022-JP. For attachment files, Base64 encoding is always used. Other MIME encodings (such as nesting) are not supported.
5. EBCDIC Japanese CCSIDs (5026/5035/1390/1399) will be converted to ISO-2022-JP. You cannot send a mail by other Japanese codes such as Shift-JIS(x-sjis) or EUC. SBCS katakana will be replaced by DBCS katakana. Non-JIS defined DBCS characters will be replace by thick '=' character or the command terminates in error depending on NONJISDBCS parameter.

Parameter Definitions

FROM: Specify the sender of the mail:

Mail Address - Specify the internet mail address of a person or organization.
Description - Specify mail user information such as employee name or organization name.

TO: Specify recipients who are to receive the mail. The maximum number of recipients is 30.

Mail Address - Specify the internet mail address of a person or organization.
Description - Specify mail user information such as employee name or organization name.
Recipient Type - Either:
*TO - The user is the primary recipient of the mail.
*CC ("Carbon Copy") - The user is receiving a copy of the mail sent to the primary recipient. This copy recipient is identified on the distribution as a secondary receiver of the distribution.
*BCC ("Blind Carbon Copy") - The user is receiving a copy of the mail sent to the primary recipient, but this copy recipient is not identified on the distribution as a receiver of the distribution.

FILE: Specify the database file and library name which contains mail body text. The file should be a source physical file (with record length 92) or a data physical file (with record length 80). An example Mail Body text file has been defined in NUTIL, file name NSNDSMTPML. You may use this as your Mail Body file, or use it as a template for creating your own.

MBR: Specifies the database file member name which contains the mail body text to be sent.

SUBJECT: Specify mail subject (title). This is an optional parameter but it is recommended you always specify a mail subject.

ATTACHMENT: Specify the full path name of the stream file that will be sent as a mail attachment. A maximum of 5 attachments can be specified. For more information on specifying path names, refer to Chapter 2 of the CL Reference, SC41-5722.

REPLYTO: Specify the "reply to" mail address. If this parameter is specified, most mail clients will send replies to this mail address, instead of the sender address.

Mail Address - Specify the internet mail address of a person or organization.
Description - Specify mail user information such as employee name or organization name.

SMTPHOST: Specify the mail (SMTP) server with which the command will communicate.

*LOCALHOST - The iSeries machine running the command is the SMTP server. The iSeries must be configured and running as an SMTP server.
host-name - Specify the mail (SMTP) server TCP/IP host name.

NONJISDBCS: Specify how the command should act when it detects undefined JIS characters in the mail header or mail body text. This applies only to the Japanese (CCSID 5026/5035/1390/1399) environment.

*ABORT - Command aborts, mail not sent.
*REPLACE - Convert undefined JIS character(s) to thick DBCS '=' and send mail.

HDRCCSID: Specify the method of obtaining the job CCSID to encode mail header.

*DFTJOBCCSID: The default job CCSID is used.
coded-character-set-identifier: Specify valid EBCDIC CCSID.

DBFCCSID: Specify the method of obtaining the mail body text file CCSID.

*DFTJOBCCSID- The default job CCSID is used.
*FILE - The database file CCSID is used, unless it is 65535.
coded-character-set-identifier - Specify valid EBCDIC CCSID.

TMPDIR: Specify the directory in which the command creates the temporary workfile. The Root ('/') directory cannot be specified.

'/TMP' - Directory '/TMP' is used as working directory
directory-name - Specifies the path name of the directory being used.

NSTRJRNPF - Start (generic) journal physical files

The NSTRJRNPF command is a generic file version of the standard STRJRNPF command, which allows you start journaling changes made to a specific file, or set of files, to a specific journal.

```
                  Start journal physical files (NSTRJRNPF)

 Type choices, press Enter.

 Physical files to be journaled   FILE
   Library  . . . . . .                *LIBL
 Journal  . . . . . . . . . . . JRN
   Library  . . . . . .                *LIBL
 Record images  . . . . . . . . IMAGES     *BOTH
 Journal entries to be omitted  . OMTJRNE    *OPNCLO
```

FILE - Specifies the name of the physical file(s) whose changes are written to the journal. This is a required parameter. The possible values are:

file name - Specify the full name of the file. Journaling is started for this file only.

generic* file name - Specify a generic name. Journaling is started for all files whose names begin with the specified characters.

The possible library values are:

*LIBL - The library list is used to locate the file.

*CURLIB - The current library for the job is used to locate the file. If a current library is not defined for the job, QGPL is used.

Library name - Specify the name of the library where the file resides.

JRN - Specifies the name and library of the journal that will receive the file change journal entries. This is a required parameter.

The possible library values are:

*LIBL - The library list is used to locate the journal.

*CURLIB - The current library for the job is used to locate the journal. If a current library is not defined for the job, QGPL is used.

Library name - Specify the name of the library where the journal resides.

IMAGES - Specifies the kinds of record images to be written to the journal for changes to records in the file.

*BOTH - The system writes both before and after images to the journal for changes to records in this file.

*AFTER - Only after images are written to the journal for changes to records in this file.

OMTJRNE - Specifies the journal entries that are omitted.

*OPNCLO - Open and close entries are omitted. Open and close operations on the specified file members do not create open and close journal entries.

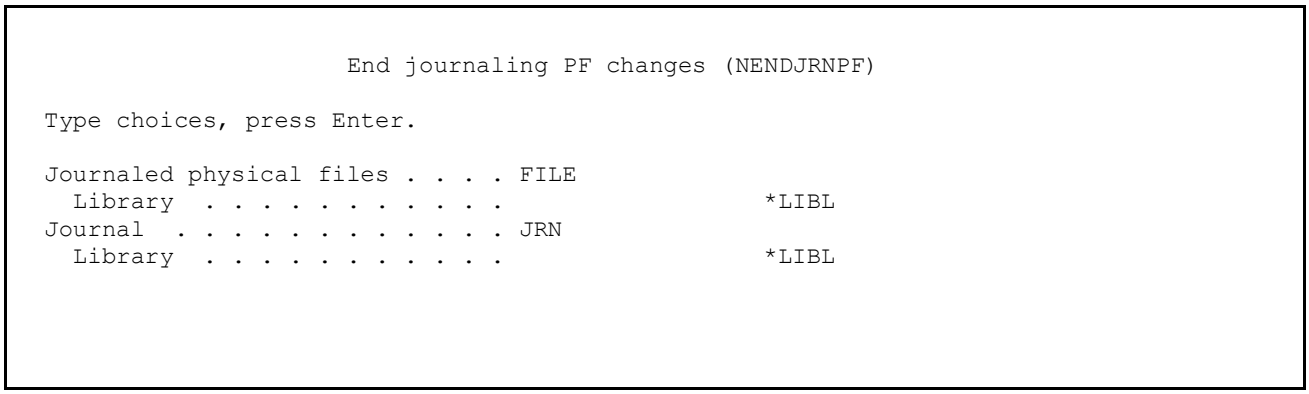
*NONE - No journal entries are omitted.

NENDJRNPF - End (generic) journaling changes

The NENDJRNPF command is a generic file version of the standard ENDJRNPF command, which allows you end journaling for a specific file, or set of files.

Restrictions

The *generic file processing function within this command may only work in it's current form if it is used in an English language version of OS/400. This is because it relies on spooled output from the WRKJRNA command to obtain a list of files currently journaled. Other language versions of OS/400 may format this report differently.



FILE . Specifies the name of the physical file(s) whose changes will no longer be journaled. This is a required parameter.

file name - Specify the full name of the file. Journaling is ended for this file only.

*ALL - All physical files currently being journaled to the specified journal no longer have their changes journaled.

generic* file name - Specify a generic name. Journaling is ended for all files, currently being journaled to the specified journal, whose names begin with the specified characters.

The possible library values are:

*LIBL - The library list is used to locate the file.

*CURLIB - The current library for the job is used to locate the file. If a current library is not defined for the job, QGPL is used.

Library name - Specify the name of the library where the file resides.

JRN - Specifies the name and library of the journal to which changes in the indicated files are currently being journaled. This is a required parameter.

The possible library values are:

*LIBL - The library list is used to locate the journal.

*CURLIB - The current library for the job is used to locate the journal. If a current library is not defined for the job, QGPL is used.

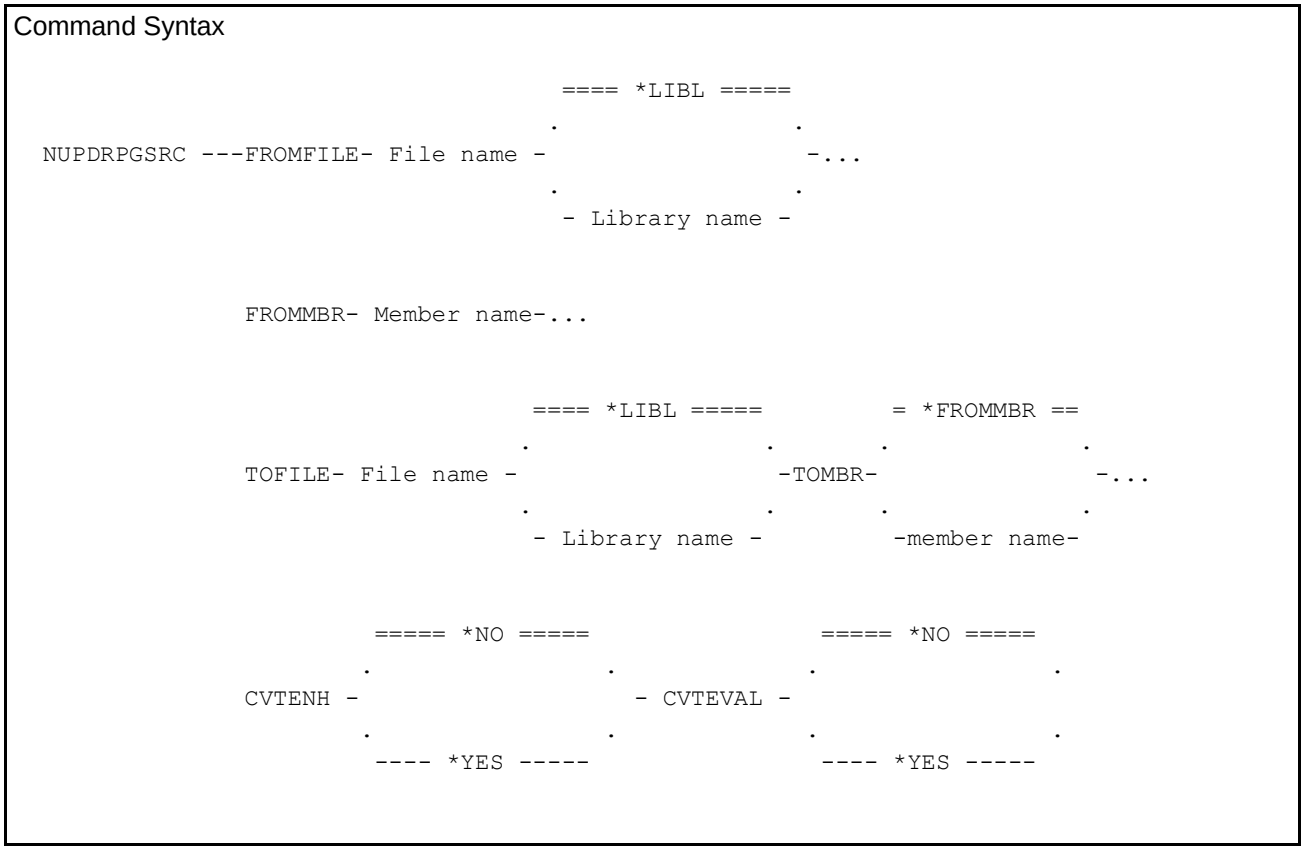
Library name - Specify the name of the library where the journal resides.

NUPDRPGSRC - Update (Enhance) ILE RPG source code

The NUPDRPGSRC command allows you to 'reprocess' your ILE RPG source code to potentially improve its appearance and readability.

Note: The determination of whether the readability of source code is improved is an objective one. You should use the CMPPFM (Compare file members) command to see the changes that have been made in the 'improved' version before deciding to replace your old source.

Command Syntax



Parameter Definitions

FROMFILE: Specifies the name of the source file containing the existing ILE RPG source member to be read for update. This is a required parameter.

FROMMBR: allows you to specify the name of the source member to be processed for update.

TOFILE: Specifies the name of the target source file, where the updated source member will be stored. This is a required parameter.

> The TOFILE file specified must exist and should have a record length of 122 characters or more. The default name for an ILE RPG source file is QRPGLESRC.

TOMBR: allows you to specify the name of the target (updated) source member.

> Enter the member name for the target (updated) source member. If the member currently exists, it will be replaced. If the member name does not currently exist in the target source file, it will be created.
>
> An entry of *FROMMBR specifies that the target converted source is the same as the member name specified in the From Member parameter. Note that target source member can not replace the source member that is being read for update.

CVTENH: specifies whether you wish source code enhancements to be applied. These additional enhancements relate to the structured operation codes IF, DO, CAS and WHEN.

Note: Selecting enhanced code conversion requires the source code to be resequenced, to allow the code changes to occur. In doing so, any comments on positions 1-5 of source lines (other than array and table data) will be dropped in the resulting source member.

> *YES - Enhanced code conversion will occur. Each column-defined IF*xx*, DO*xx*, CAS*xx* and WHEN*xx* statement detected will be converted in to it's free format equivalent. Any comments detected in positions 1-5 of the source code line will be dropped.
>
> *NO - Enhanced code conversion will not occur. Source will be converted on a line-for-line basis.

CVTEVAL: specifies whether the updated program source will retain usage of any old "fixed format" RPG Operation codes, or convert any usage to an equivalent EVAL free-format Operation Code:

> *YES - Converted program source will use the EVAL Operation Code.
>
> *NO - Converted program source will retain any usage of old RPG Operation Codes.

For CVTEVAL(*YES), the following conversions will occur:

- MOVEL will be replaced by Eval
- MOVEL (with Blank Padding) will be replaced by Eval
- MOVE (indicators) will be replaced by Eval
- MOVE will be replaced by Evalr
- MOVE (with Blank Padding) will be replaced by Evalr
- Z-ADD will be replaced by Eval
- Z-ADD (with Half-Adjust) will be replaced by Eval(h)
- ADD will be replaced by Eval
- ADD (with Half-Adjust) will be replaced by Eval(h)
- SUB will be replaced by Eval
- SUB (with Half-Adjust) will be replaced by Eval(h)
- MULT will be replaced by Eval
- MULT (with Half-Adjust) will be replaced by Eval(h)
- DIV will be replaced by Eval
- DIV (with Half-Adjust) will be replaced by Eval(h)

Conversion Notes:

Div - subsequent use of the MVR operation requires manual attention (replace with %Rem)

Any result field definitions in C specs will be dropped

Move operations (replaced by Evalr) may require manual attention after conversion. The old move operation code does not have a direct equivalent in ILE - the closest is Evalr. But Evalr is far more formalised (and restrictive) in it's capability.

NVFYLNK – Verify IFS object link

The Verify IFS Object Link command checks IFS object existence and verifies the user's authority for the object before trying to access it. If the object exists and the user has the proper authority for the object, no error messages are sent to the user.

When the command is run, the system searches for the specified object. If the object is found, the system verifies that the user is authorized to that object as specified for the Authority (AUT) parameter. If the object is not found or the user does not have the authorities specified for the AUT parameter, an error message is sent to the user.

Command Syntax

```
NVFYLNK ---OBJ- IFS object - ...


               AUT - check for authority
```

Parameter Definitions

OBJ - Specifies the path name of the object to verify. This object must already exist.

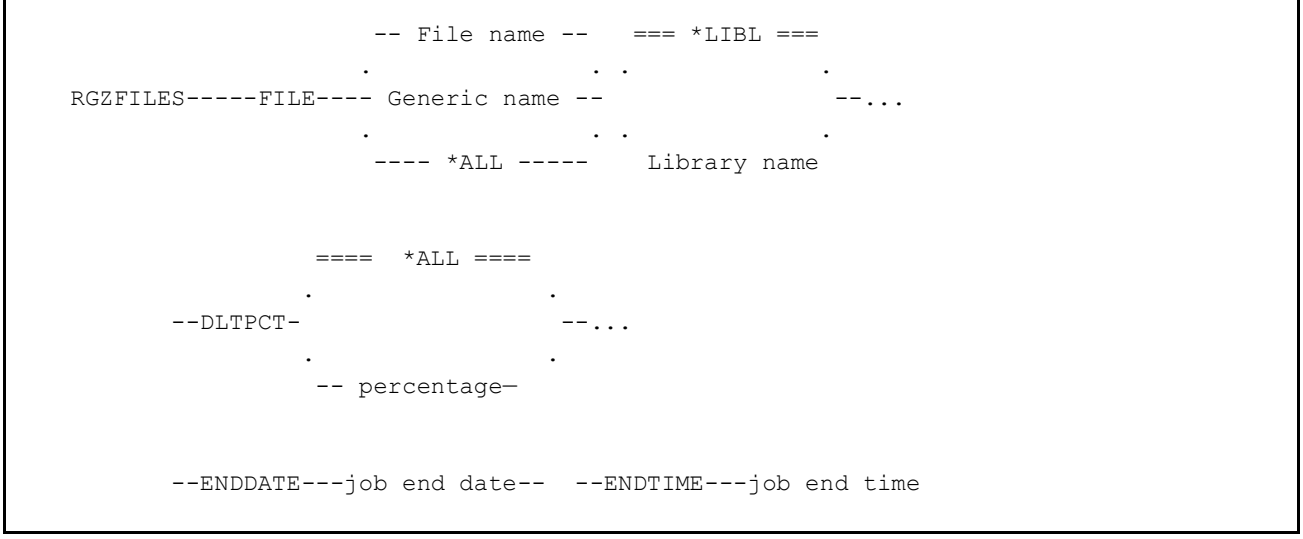
AUT - Specifies the authority to verify. The possible values are:

*EXISTS:	Checks only whether the specified object exists.
*R:	Checks whether user has access to object attributes.
*RW:	Checks access to the object attributes and rights to change the object.
*RWX:	Checks for authority to all operations on the object except those that are limited to the owner or controlled by the object rights.
*RX:	Checks authority for access to the object attributes and use of the object.
*W:	Checks whether user has change rights to the object.
*WX:	Checks whether user has use and change rights to the object.
*X:	Checks whether user has use rights to the object.

RGZFILES - Reorganise Library Physical Files

This command allows you to reorganise multiple files in a library with one command.

Command Syntax



ALL members in a selected physical file will be eligible for reorganisation. Reorganisation will only occur if the percentage of deleted records equals or exceeds the requested parameter.

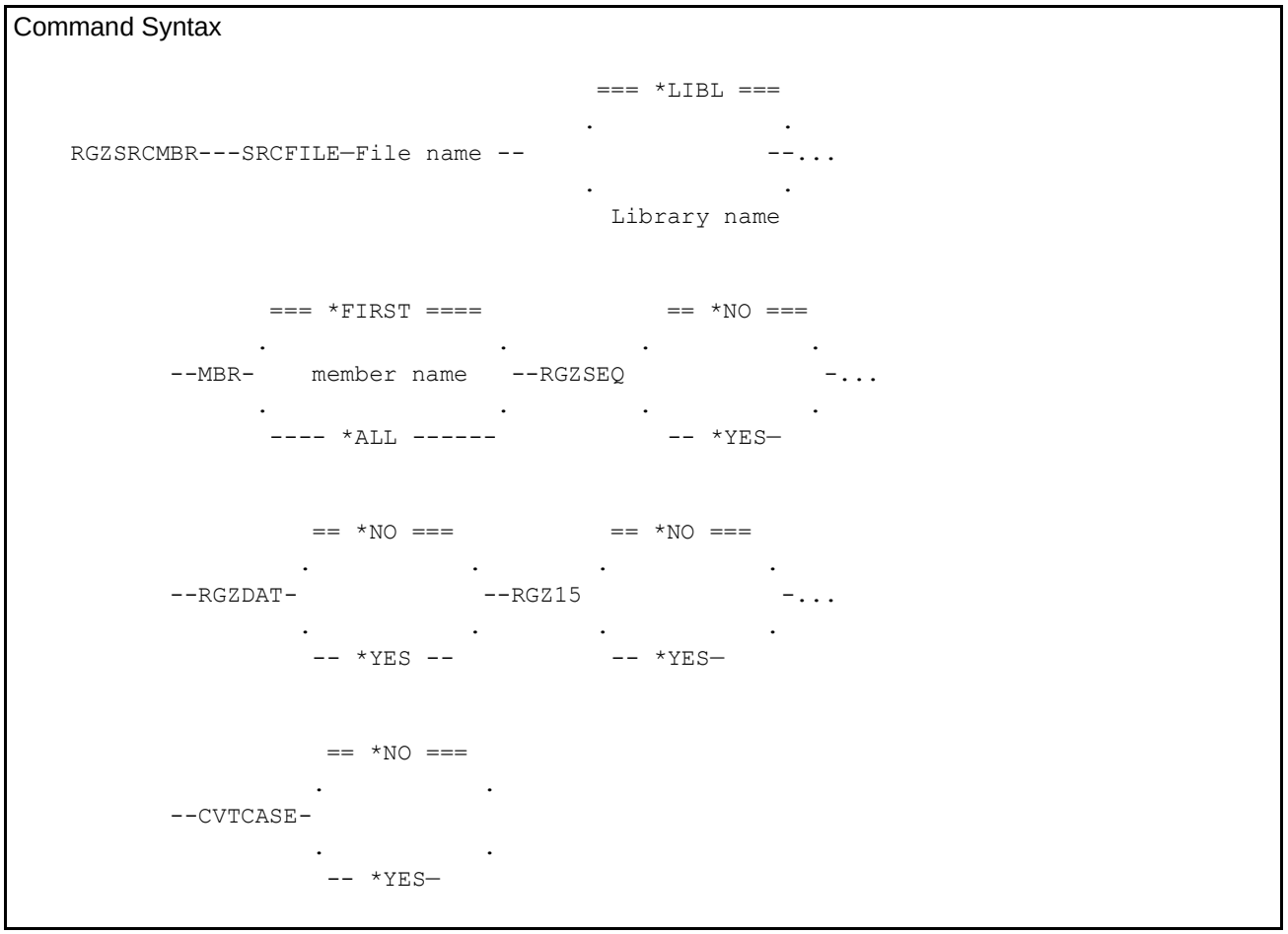
As this could be a long running process it should only be used in a batch environment. If the command is accessed from the UTLCMD menu it will automatically be submitted for batch processing.

The optional ENDDATE and ENDTIME parameters can be used to ensure that the job finishes before a specified time. If the time specified is reached before all files have been reorganised, the job will end at that point. This is a timestamp that will be used by the processing program to ensure no file reorganisation will be started AFTER that specified time, so if one is already in process when the time is reached, then the job will complete processing after that file has been reorganised.

RGZSRCMBR - Reorganise Source Member

This command allows you to reorganise the information in a source file member. The command can only be used on source physical files.

Command Syntax



Parameter Definitions

RGZSEQ: If set to *YES, the line numbering in the specified source member will be re-sequenced, starting at 1.00 with an increment value of 1.00.

RGZDAT: If set to *YES, the source change date field in all lines of the source member will be reset to zero.

RGZ15 : If set to *YES, any data in columns 1 to 5 of all source lines in the source member will be blanked out. If the data is an array definition it is not affected by this parameter.

*YES is only valid for source member types RPG, RPGLE, DSPF, PRTF, LF and PF.

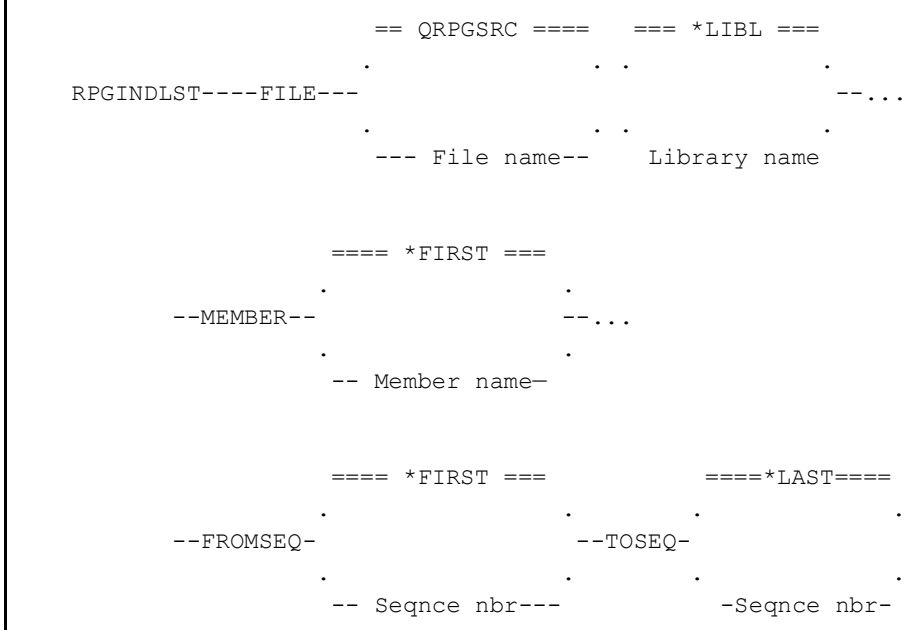
CVTCASE: If set to *YES, the program source will be converted to lower case characters. This is useful where you have converted and RPG source member into an RPGLE source member.

*YES is only valid for source member type RPGLE.

RPGINDLST - Formatted RPG Source Listing

This command will print a formatted RPG source listing for the source member specified. All DO groups are shown indented for simpler recognition.

Command Syntax



DO, IF and WHere groups are indented by level of indentation. An example of this would be:

RPG Program		***Indented Listing***		
A	IFEQ B	*B001*		A IFEQ B
C	OREQ D	*001*		C OREQ D
X	DOWLT25	*B002*	...	X DOWLT25
	ADD 1 X	*002*	...	ADD 1 X
	WRITEFMTA	*002*	...	WRITEFMTA
	ENDDO	*E002*	...	ENDDO
	ELSE	*X001*		ELSE
	WRITEFMTB	*001*		WRITEFMTB
	ENDIF	*E001*		ENDIF

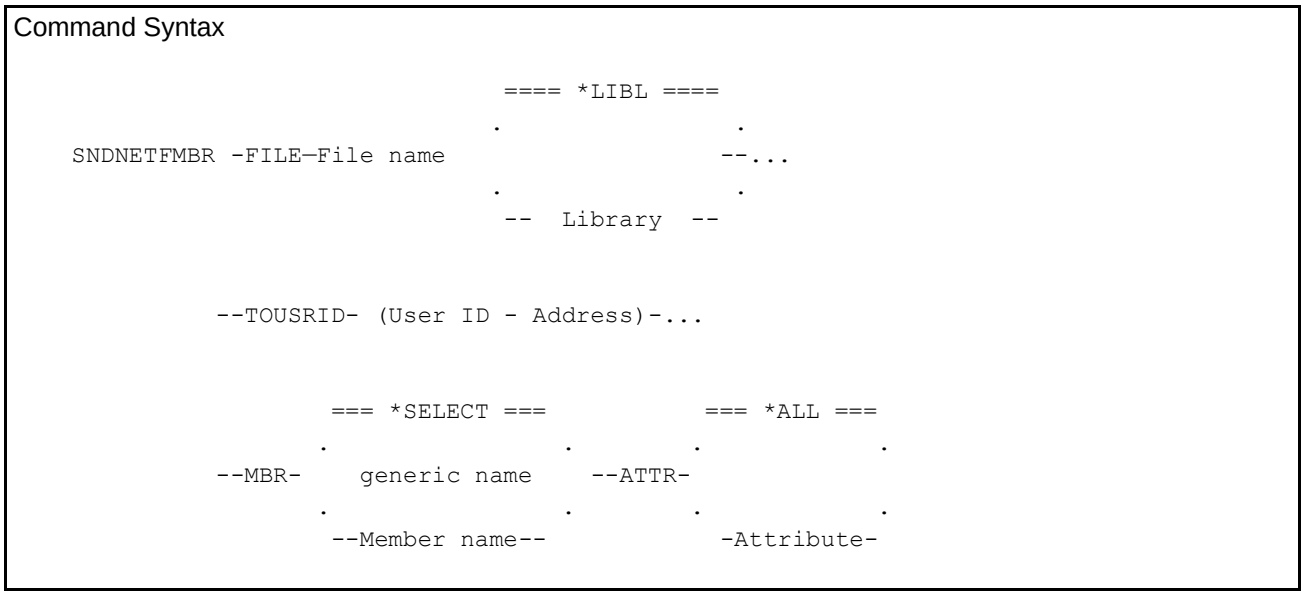
The FROMSEQ and TOSEQ parameters allow you to print a portion of the specified program, using ‘from’ and ‘to’ line sequence numbers.

When using this you should be careful, as the indentation logic may not be fully contained within the line numbers specified (and thus the displayed indentation may be incorrect). You should only use the range criteria within a functional subset of the program, for example, a subroutine.

SNDNETFMBR - Send Network File Member

This command allows the sending of one, or many, members in a file via SNADS.

Command Syntax



The SNDNETF command is performed for each member within the generic group (and source attribute, if specified) requested.

Selection of the member to send is also available. Where selection is requested, only one member can be selected to send.

The select function should only be used where the selected file has less than 100 members, as the selection list may take some time to build.

TIME - Display system time and date

This command will display a message showing the current system time and date.

Command Syntax

```
TIME   (Command has no parameters)
```

WRKACTSBS - Work with Active Subsystem

The Work with Active Subsystem (WRKACTSBS) command allows you to work with jobs running in a specified subsystems in the system, and jobs that are on any job queue for that subsystem. If one of the jobs shown on the subsystem display is selected, additional information about that job can be displayed.

Command Syntax

```
                                 ==== *LIBL ====
                                .                .
WRKACTSBS -SBS—Subsystem name
                                .                .
                                 --  Library  --
```

Parameter Definitions

SBS (Subsystem name) - specifies the name and library of the subsystem description to be worked with. The subsystem must be currently active to be processed by this command.

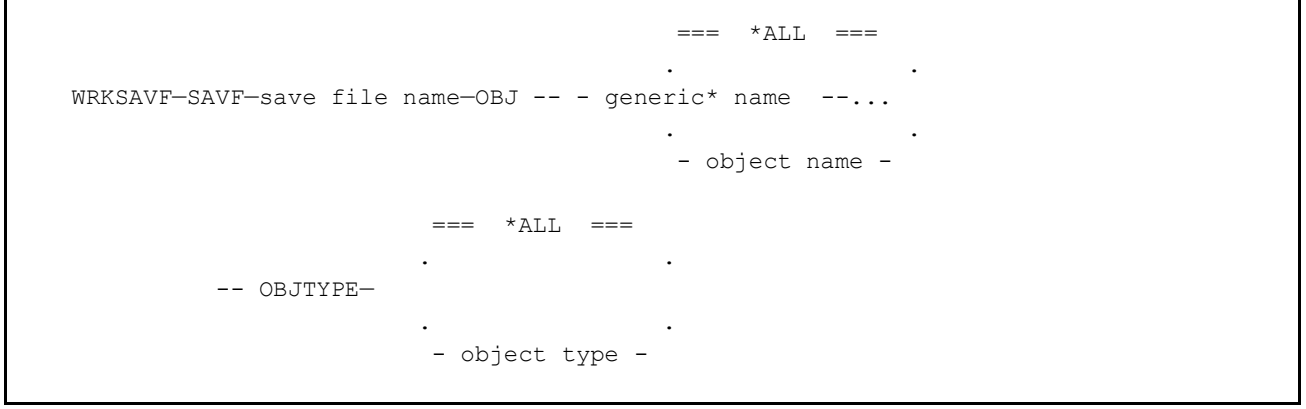
Specify the name of the subsystem. All active jobs in this subsystem are displayed and access to the job queues for that subsystem is also made available.

WRKSAVF - Work with Save File

The Work with Save File (WRKSAVF) command allows you to view the contents of a Save File and selectively restore objects from it.

Using the parameters you specify, the command will build a list panel, showing the saved objects within the save file. For each object listed you then have the option to restore it to your system.

Command Syntax



Parameter Definitions

SAVF (Save file name) - specifies the name of the save file to be processed. This is a required parameter.

OBJ (Saved object name) - specifies the name (or generic name) of the saved object(s) within the save file that you wish to work with.

> *ALL - All saved objects in the save file will be displayed in the list panel.
> Generic* object name - Specify the generic name for the objects you require.
> Object name - Specify the name of the object you require.

OBJTYPE (Saved object type) - specifies the object type of the saved object within the save file that you wish to work with.

> *ALL - All object types in the save file will be available to work with.
> Object type - Specify the object type you require.

Callable program modules

In addition to the utility commands, also included are some generic function program modules, which are used by NUTIL functions but can also be used by your own user programs.

These module programs are always identified by their program name, which always has an '@' as the first character of the program name. The module programs that can be used by your own programs are explained in this section of the manual.

The RPG routines all remain open after the first program call, in order to improve response times on program calls:

The OPM routines can be closed by the RCLRSC (Reclaim Resources) command.

The ILE routines can be closed by the RCLACTGRP (Reclaim Activation Group) command:

Standalone ILE programs run in the QILE activation group
Service programs run in the QSRVPGM activation group

Service Programs and the NUTIL Binding Directory

There are 2 service programs supplied with NUTIL:

- NSRVPGM is the general service program for all NUTIL callable modules except the 'Y2K Conversion Assistant' date conversion routines
- NDCSRV is the service program containing *only* the 'Y2K Conversion Assistant' date conversion modules

These service programs are also referenced in the NUTIL NBNDDIR binding directory. As a general rule you should always access the NUTIL service program modules via the NBNDDIR binding directory instead of directly accessing the service program. This ensures any changes to the structure/sequencing of the service programs (and their module imports/exports) will not affect the operation of any programs you have created that use them.

When creating your program you just specify the binding directory on your CRT command:

```
CRTPGM...     BNDDIR(NUTIL/NBNDDIR)
CRTBNDRPG...  BNDDIR(NUTIL/NBNDDIR)
CRTBNDCL...   BNDDIR(NUTIL/NBNDDIR)
CRTBNDCBL...  BNDDIR(NUTIL/NBNDDIR)
CRTBNDRPG...  BNDDIR(NUTIL/NBNDDIR)
```

You should define the NUTIL binding directory after any of your own specific binding directory entries, to ensure the correct procedure is accessed by your program call.

All NUTIL Service Programs are set to run in activation group QSRVPGM.

They can be closed by the RCLACTGRP (Reclaim Activation Group) command.

Prototypes and /COPY

Prototypes for using NUTIL callable functions can be found in source file NUTIL/PROTYPES. To use one of the pre-defined prototypes, just use the standard /COPY statement in your program:

```
/COPY NUTIL/PROTOTYPES,@RVSDAT8_H
```

Date Handling Overview

Date handling for 6 and 7 digit dates

NUTIL date processing routines for 6 and 7 digit dates use a full 7 digit date field, where the first digit in all instances is a century digit.

Year/Month/Day is Century/Year/Month/Day - CYYMMDD
Month/Day/Year is Century/Month/Day/Year - CMMDDYY
Day/Month/Year is Century/Day/Month/Year - CDDMMYY

This allows you to handle the changeover to the 21st century (century digit is set to 0 or 1, where the 20th century = 0).

When a 6 digit date is passed to one of the utility programs, a 6 digit date will be returned (the century digit is ignored).

In general, the NUTIL date routines for 6 and 7 digit dates will only work on dates in the range 1940 – 2039

For a 7 digit date, where the century digit is passed as zero, the basic windowing rule will be applied - that years less than 40 are century 0 (19*nn*) and years greater than 39 are 1 (20*nn*). This is especially important to note for the @RVSDAT and @GETDAT modules.

Date handling for 8 digit dates

All NUTIL 8 digit date handling routines use an 8P0 parameter variable for passing dates and a 4A parameter variable for passing date type.

The date formats used by NUTIL date handlers are as follows:

Format name	*Description*	*Format*	*Data length*	*Formatted Length*	*Example*
*MDY	Month/Day/Year	mm/dd/yy	6	8	06/27/03
*DMY	Day/Month/Year	dd/mm/yy	6	8	27/06/03
*YMD	Year/Month/Day	yy/mm/dd	6	8	02/06/27
*CYMD	Century/Year/Month/Day	cyy/mm/dd	7	9	002/06/27
*JUL	Julian	yy/ddd	5	6	02/179
*LONGJUL	Extended Julian	yyyy/ddd	7	8	1902/179
*ISO	International Standards Organisation	yyyy-mm-dd	8	10	1999-06-27
*USA	IBM USA Standard	mm/dd/yyyy	8	10	06/27/1999
*EUR	IBM European Standard	dd.mm.yyyy	8	10	27.06.1999
*JIS	Japanese Industrial Standard Christian Era	yyyy-mm-dd	8	10	1999-06-27
*JOB	Current job format	See below	6	8	
*JB8	Full job format	See below	8	10	

With *YMD, *MDY and *DMY you can pass the routine a 6 digit date and the routine will validate it as though it were a date in the range 1940 - 2039.

It is important to understand that the *YMD, *MDY and *DMY formats will only work on dates in the range 1940 - 2039. For dates outside this range you must use another date format.

The special value of *JOB will assume the date format of the job running the date routine, which will resolve to either a *YMD, *MDY or *DMY (6 digit date) format. This is also known as *JOBRUN format.

The special value of *JB8 can be used in the @RVSDAT8 module to use the *JOBRUN format, but process a full 8 digit date.

@CHKDAT - Check date for validity

This program will test a given date for validity. If invalid, a value of 9999999 will be returned as the date. Note that the program remains open after the first call, to improve response.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Chkd7	7P0	I/O	Field containing the date
£Dfi7	1A	I	Date format in, either 'D' Day/Mth/Year 'M' Mth/Day/Year 'Y' Year/Mth/Day 'S' System format 'J' Job format

Where a 6 digit date is passed, a 6 digit date will be returned (i.e. the century digit is ignored). Note that the date passed must be valid date data within standard rules. A zero date is considered invalid.

The valid date range for this procedure is January 1, 1940, to December 31, 2039. Dates outside of this range will be treated as invalid dates.

@CHKDAT8 - Check 8 digit date for validity

This program will test a given date for validity. If the date passed to the routine is invalid, a value of 99999999 will be returned as the date. Note that the program remains open in the default activation group after the first call, to improve response.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Chkd8	8P0	I/O	Field containing the date
£Dfi8	4A	I	Date format in, either *DMY Day/Mth/Year *MDY Mth/Day/Year *YMD Year/Mth/Day *ISO System format *EUR EUR format *JIS JIS format *JUL JUL format *USA USA format *JOB Job format

Note that the date passed must be valid date data within standard rules. A zero date is considered invalid.

Example of usage

This code will test field ScreenDate to see whether it contains a valid date in the current *JOB format. If it fails the check then the program will send an ‘Invalid date’ message to the user.

```
     *  ...Check Date is valid (only if entered)...
    C     ScreenDate    Ifne      0
    C     ScreenDate    Andne     999999
    C                   Call      '@CHKDAT8'
    C                   Parm      ScreenDate    £Chkd8
    C                   Parm      '*JOB'        £Dfi8

     *  ...Date does not conform to Job Format...
    C     £Chkd8        Ifeq      99999999
    C                   Move      *ON           *IN98
    C                   Move      'GEN0061'     Nmsgid
    C                   Endif
    C                   Endif
```

Example of Prototyped Call

```
     /COPY NUTIL/PROTOTYPES,@CHKDAT8_H

    C                   CALLP     @CHKDAT8(£Chkd8:£Dfi8)

     *  ...Date does not conform to Job Format...
    C                   If        £Chkd8  = 99999999
    C                   Eval      *IN99   = *ON
    C                   Eval      NMsggid = 'GEN0008
    C                   EndIf
    C                   EndIf
```

@CHKTIM - Check time for validity

This program will test a given time for validity. Also available as a bound call via the NSRVPGM service program.

If invalid, a value of 999999 will be returned as the time. Note that the program remains open after the first call, to improve response.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Chktm	6P0	I/O	Field containing the time

The time is assumed to be in hour/minute/second (*hhmmss*) format.

@CHKVN - Check name for validity

This program will test a given name to ensure it conforms to standard OS/400 object naming conventions. Refer to the IBM iSeries CL Reference Guide, under the topic *'Rules for specifying names'* for further information. Also available as a bound call via the NSRVPGM service program.

The validity check tests the following:

- The first character can be A-Z, or one of the allowed special characters for the installed character set
- The rest of the name can be A-Z, an allowed special character, 1-9, _ or .
- Minimum length of the name is 1.

If invalid, a value of 'Y' will be returned in the error flag parameter. Note that the program remains open after the first call, to improve response.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Chknm	10A	I	Field containing the name to be checked
£Nmerr	1A	O	Error flag. If invalid name, the 'Y' is returned

@CLCDAT – Calculate an 8 digit date, based on start date and duration

This program will calculate a date, given a base (starting) date and a duration to add or subtract from it. It can also return the day number or the week, the day name and the month name for the calculated date.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
Required parameters			
£Clcd8	8P0	I/O	Field containing the date
£Dfi8	4A	I	Date format in, either *DMY Day/Mth/Year *MDY Mth/Day/Year *YMD Year/Mth/Day *ISO System format *EUR EUR format *JIS JIS format *JUL JUL format *USA USA format *JOB Job format
£DurUnit	3P0	I	Number of units to add to, or subtract from, the input date
£DurType	8A	I	Duration type for the units specified, either *DAY *DAYS *MONTH *MONTHS *YEAR *YEARS
Optional parameters			
£Day	1P0	O	Day number of the week, where Monday is 1, to Sunday (7)
£DayName	10A	O	Day name (Monday, Tuesday, etc)
£MthName	10A	O	Month name (January, February etc)

Day and Month names text

Any “text descriptors”, like day and month names, are stored in the NUTIL/NUMSGF message file.

The message identifiers are NTX0001 to NTX0012 (month names) and NTX0013 to NTX0019 (day names).

If you wish to change these to display a language other than English, use the WRKMSGD (Work with Message Descriptions) command to change the text (maximum 10 characters for each message).

@CLCLOG - Calculate Logarithm

This program calculates the logarithm, either common (log to base 10) or natural (log to base e), based on an input number. Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Number	15P9	I/O	Field containing the input number. Also the field in which the calculated logarithm will be returned.
£LogType	1A	I	Calculation type, either '1' Common log '2' Natural log

If an error occurs in the calculation process, the number returned will be 0.

Note that the program remains open after the first call, to improve response.

@CMPDTS – Compare Date/Timestamps

This program will compare two timestamps and return the difference between the two, in seconds.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
Required parameters			
£Date1	8P0	I	Date 1, in *ISO format
£Time1	6P0	I	Time 1, in *HMS format
£Date2	8P0	I	Date 2, in *ISO format
£Time2	6P0	I	Time 2, in *HMS format
£Diff	11P0	O	Time difference, in seconds

It is assumed that, in normal operational circumstances, the Date1/Time1 timestamp will be *earlier than* the Date2/Time2 timestamp (see below).

Special notes:

- If any of the 4 input values are invalid, or if a program failure occurs, the value returned in output variable £Diff will be 99999999999
- If the Date1/Time1 timestamp is *greater (later) than* the Date2/Time2 timestamp, the value returned in output variable £Diff will be *negative*
- If the Date1/Time1 timestamp is *less (earlier) than* the Date2/Time2 timestamp, the value returned in output variable £Diff will be *positive*
- If the Date1/Time1 timestamp is *equal to* the Date2/Time2 timestamp, the value returned in output variable £Diff will be *zero*

The values in the input variables must be valid dates and times, in the format shown.

@CVTCDT - Convert 8 digit date from/to *CYMD format

This program will either convert an 8 digit date into *CYMD (CYY/MM/DD) format, or convert a *CYMD date into 8 digit date format, depending on the conversion type parameter passed. Also available as a bound call via the NSRVPGM service program.

The valid date range for this procedure is January 1, 1928, to December 31, 2071. Dates outside of this range will be treated as invalid dates.

The century digit in the *CYMD form of date is set to 0 for dates in the range 1928 through 1999 and is set to 1 for years 2000 through 2071.

If the expected input date is invalid, a value of 99999999 (or 9999999) will be returned as the output date.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Cvtd8	8P0	I/O	Date in *YYYYMMDD format
£Cvtd7	7P0	I/O	Date in *CYYMMDD format
£CvtType	5A	I	Conversion type, either *FROM (from *CYMD to 8 digit date) or *TO (to *CYMD from 8 digit date)

@CVTJUL - Convert date from/to Julian format

This program will either convert a date from *CYMD format into Julian (*YYDDD) format, or convert a Julian date into *CYMD format, depending on the conversion type parameter passed. Also available as a bound call via the NSRVPGM service program.

The valid date range for this procedure is January 1, 1940, to December 31, 2039. Dates outside of this range will be treated as invalid dates.

If the input date is invalid, a value of 9999999 (or 99999) will be returned as the output date.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Cvtd7	7P0	I/O	Date in *CYMD format
£Cvtd5	5P0	I/O	Date in Julian format
£CvtType	5A	I	Conversion type, either *FROM (from Julian to *CYMD) or *TO (to Julian from *CYMD)

Where a 6 digit date is passed in the £Cvtd7 parameter, a 6 digit date will be returned (i.e. the century digit is ignored).

@CVTJUL8 - Convert 8 digit date from/to Julian format

This program will either convert a date from *YYYYMMDD format into 'long' Julian (*YYYYDDD) format, or convert a long Julian date into *YYYYMMDD format, depending on the conversion type parameter passed.

Also available as a bound call via the NSRVPGM service program.

The valid date range for this procedure is August 24, 1928, to May 9, 2071. Dates outside of this range will be treated as invalid dates.

If the input date is invalid, a value of 99999999 (or 9999999) will be returned as the output date.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Cvtd8	8P0	I/O	Date in *YYYYMMDD format
£Cvtd7	7P0	I/O	Date in Long Julian (*YYYYDDD) format
£CvtType	5A	I	Conversion type, either *FROM (from long Julian to *CYMD) or *TO (to long Julian from *CYMD)

@ExCmd - Process a command

This procedure allows the direct processing of a command from within an ILE program. The command string can be up to 2000 characters long, and a return code will advise whether the command was processed successfully or not.

Available only via the NSRVPGM service program.

Procedure Prototype

```
* Returns 0 if execution successful; otherwise 1.
 D @ExCmd          PR            10I 0
 D  £Cmd                       2000A
```

Usage from within ILE RPG program

```
  * Return code from @ExCmd
 D  £Rc                          10I 0

 C                   Eval      £Rc = @ExCmd('chgjob ' +
 C                             %trim(JobNbr) + '/'  +
 C                             %trim(JobUsr) + '/'  +
 C                             %trim(JobNam) + ' OUTQ(' +
 C                             %trim(Outq)) + ')

 C                   If        £Rc = 0
  *                            ... Command processed OK
 C                   Else
  *                            ... £Rc = 1; Command processing failed
C                   Endif
```

In this example, @ExCmd will execute the CHGJOB command and set the output queue to the name specified in the variable OUTQ.

If the execution of the command is successful, Variable £RC will return a value of 0; if it fails the value of £RC will be 1.

@ERRMSG - Error message handler

This program will perform a function on a program or external message queue, based on the input parameters. It can be used to

- Clear a program message queue
- Send a message to a program message queue
- Send a program status message

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£MsgID	7A	I	Message ID to send
£Msgf	10A	I	Message file containing MSGID
£Msgdta	120A	I	Message data to include in message

Special Usage of the @ERRMSG program

If the £MsgID parameter is set to the special value of blanks, the program message queue of the calling program will be cleared.

If the £MsgID parameter is set to the special value of *STATUS, a program status message will be displayed to the user, using the text passed in the £Msgdta parameter as the status message.

@GETDAT - Calculate a date, based on start date and no of days

This program will calculate a date, given a base (starting) date and a number of days to add or subtract from it. It can also return the day number or the week, the day name and the month name for the calculated date.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
Required parameters			
£Clcd8	7P0	I/O	Field containing the date
£Dfi	1A	I	Date format in, either 'D' Day/Mth/Year 'M' Mth/Day/Year 'Y' Year/Mth/Day 'S' System format 'J' Job format
£Days	3P0	I	Number of days to add to, or subtract from, the input date
£Day	1P0	O	Day number of the week, where Monday is 1, to Sunday (7)
Optional parameters			
£DayName	10A	O	Day name (Monday, Tuesday, etc)
£MthName	10A	O	Month name (January, February etc)

Where a 6 digit date is passed, a 6 digit date will be returned (i.e. the century digit is ignored).

Notes on 7 digit dates

The century portion of a 7digit date is implied, as follows:

- Years in the range 40-99 will be treated as 1940-1999
- Years in the range 00-39 will be treated as 2000-2039
- Any dates outside of the range 1940-2039 are invalid dates

Examples of usage:

If the program was passed:

```
CALL @GETDAT   PARM(120488 D 28 0)
```

Then the date returned would be 100588.

And if the program was passed:

```
CALL @GETDAT   PARM(120488 D -5 0)
```

Then the date returned would be 070488.

In the case of century changeover, if the date passed was

```
CALL @GETDAT   PARM(0291299 D 5 0)
```

Then the date returned would be 1030100 (3rd January, 2000).

Day name is only returned if 5 or 6 parameters were passed in the call to the @GETDAT program. For example, if the program was called as follows:

```
CALL @GETDAT   PARM(900615 Y 10 0 '          ')
```

Then the date returned would be 900525
the day number returned would be 1
the day name returned would be 'Monday '

And if the program was passed:

```
CALL @GETDAT   PARM(900615 Y 10 0)
```

Then the date returned would be 900525

the day number returned would be 1
the day name parameter would not be processed.

Month name is only returned if 6 parameters were passed. For example, if the program was called as follows:

```
CALL @GETDAT   PARM(900615 Y 10 0 '          ' '          ')
```

Then the date returned would be 900525
the day number returned would be 1
the day name returned would be 'Monday '
the month name returned would be 'June '

And if the program was passed:

```
CALL @GETDAT   PARM(900615 Y 10 0 '          ')
```

Then the date returned would be 900525
the day number returned would be 1
the day name returned would be 'Monday '
the month name parameter would not be processed.

@GETDAT8 - Calculate an 8 digit date, based on start date and no of days

This program will calculate a date, given a base (starting) date and a number of days to add or subtract from it. It can also return the day number or the week, the day name and the month name for the calculated date.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
Required parameters			
£Clcd8	8P0	I/O	Field containing the date
£Dfi8	4A	I	Date format in, either *DMY Day/Mth/Year *MDY Mth/Day/Year *YMD Year/Mth/Day *ISO System format *EUR EUR format *JIS JIS format *JUL JUL format *USA USA format *JOB Job format
£Days	3P0	I	Number of days to add to, or subtract from, the input date
£Day	1P0	O	Day number of the week, where Monday is 1, to Sunday (7)
Optional parameters			
£DayName	10A	O	Day name (Monday, Tuesday, etc)
£MthName	10A	O	Month name (January, February etc)

@GETDAYS - Get the number of days between two dates

This program will calculate the number of days difference between the two dates supplied. Invalid dates will be treated as 0000000.

Also available as a bound call via the NSRVPGM service program.

The valid date range for this procedure is January 1, 1940, to December 31, 2039. Dates outside of this range will be treated as invalid dates.

Where an error in dates occurs, the number of days parameter will return 999999 to the calling program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Date1	7P0	I	'From' Date in *CYMD format
£Date2	7P0	I	'To' Date in *CYMD format
£Days	6P0	O	The number of days calculated between the dates

Where a 6 digit date is passed, a 6 digit date will be returned (i.e. the century digit is ignored).

@GETDYS8 - Get the number of days between two 8 digit dates

This program will calculate the number of days difference between the two dates supplied. Invalid dates will be treated as 00000000.

Also available as a bound call via the NSRVPGM service program.

The valid date range for this procedure is August 24, 1928, to May 9, 2071. Dates outside of this range will be treated as invalid dates.

Where an error in dates occurs, the number of days parameter will return 999999 to the calling program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Date1	8P0	I	'From' Date, in *YYYYMMDD format
£Date2	8P0	I	'To' Date, in *YYYYMMDD format
£Days	6P0	O	The number of days calculated between the dates

@GETDOW - Get the day-of-the-week number for a given date

This program will determine the day-of-the-week number for a given date. Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Date8	8P0	I	Field containing the date, in *YYYYMMDD format
£Day	1P0	O	Day number of the week, where 1=Monday, though to 7=Sunday

If the date passed to the routine is invalid for the format specified, a value of 9 will be returned as the day number.

@GETTIM - Calculate time

This program will calculate a time, based on a start time and a duration to add or subtract. Also available as a bound call via the NSRVPGM service program.

If invalid, a value of 999999 will be returned as the time. Note that the program remains open after the first call, to improve response.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Time	6P0	I/O	Field containing the time
£Hours	5P0	I	Field containing the hours duration to apply
£Minutes	5P0	I	Field containing the minutes duration to apply
£Seconds	5P0	I	Field containing the seconds duration to apply

The input time is assumed to be in hour/minute/second (*hhmmss*) format.

Depending on the values in each of the duration parameters, the hours duration will be applied to the time, then the minutes and finally the seconds.

No error is returned if the calculated time ‘wraps’ to a future or previous date.

@GETWKNR - Get the ISO week number for a given date

This program will determine the week number-of-the-year for a given date. The week number is based on the Internationals Standards Organisation standard for week number definition.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Date7	7P0	I	Field containing the date, in *CYMD format
£Weeknr	2P0	O	ISO week number of the year

Date can be passed as either a 6 digit number (YYMMDD) or a 7 digit number (CYYMMDD). If a 6 digit number is passed, the century digit will be assumed to be 0.

If the date passed to the routine is invalid, a value of 0 will be returned as the week number.

@GETWKN8 - Get the ISO week number for a given date

This program will determine the week number-of-the-year for a given date. The week number is based on the Internationals Standards Organisation standard for week number definition.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Date8	8P0	I	Field containing the date, in *YYYYMMDD format
£Weeknr	2P0	O	ISO week number of the year

If the date passed to the routine is invalid, a value of 0 will be returned as the week number.

@RANDOM - Generate a Random Number

This program generates random numbers.

Also available as a bound call via the NSRVPGM service program.

The first call to the program generates a ‘seed’ number based on the current date/time. The initial seed number is based on the formula:

```
f(z) = az mod m
                where a = 16807
                      m = 2147483647
```

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Loval	10P0	I	Lower limit of range
£Hival	10P0	I	Upper limit of range
£Rndn	10P0	O	Random number

The lower limit and upper limit parameters allow you to specify the limits between which the random number can be generated. So the following call:

```
CALL @RANDOM   PARM(15  45  0)
```

would generate a random number between 15 and 45.

Note that the program remains open after the first call, to improve response.

@RPLXML - Remove/Replace invalid XML characters in string

This program will remove or replace any invalid XML characters in a text string.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Input	250A	I/O	Text string to validate
£Action	4A	I	Action to perform (*RMV/*RPL)

Valid £Action codes are

*RMV	Invalid characters will be replaced with blanks
*RPL	Invalid characters will be replaced with its XML equivalent

For the *RMV function, each invalid character will be replaced by a blank character; so the length of the output string will be the same as the input string.

For the *RPL function, the following will be performed:

Invalid character		*Will be replaced by*
&	Ampersand	&amp;
<	Less than	&lt;
>	Greater than	&gt;
'	Apostrophe	&apos;
"	Quotation mark	&quot;

Due to the single character being replaced by multiple characters, the length of the output string will be greater than the input string.

@RTVASI - Retrieve Auxiliary Storage information

This program will retrieve Auxiliary Storage information relating to disk usage of the system.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£SysAsp	7P0	O	Storage capacity of System ASP 1 (in Mb)
£SysAspC	7P4	O	Percentage of System ASP currently used
£TotalAsp	7P0	O	Total auxiliary storage on system (in Mb)
£CurUnpUse	7P0	O	Current unprotected storage used (in Mb)
£MaxUnpUse	7P0	O	Maximum unprotected storage used (in Mb)

@RTVDAT - Retrieve current system date/time stamp

This program will retrieve the current system date and time and return the date in the format specified.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Dfo	1A	I	Date format in, either 'D' Day/Mth/Year 'M' Mth/Day/Year 'Y' Year/Mth/Day 'S' System format 'J' Job format
£Sysd7	7P0	O	Field containing the system date
£Systm	6P0	O	Field containing the time

Where a 6 digit date parameter is passed, a 6 digit date will be returned (i.e. the century digit is ignored).

Notes on 7 digit dates

The century portion of a 7digit date is implied, as follows:

- Years in the range 40-99 will be treated as 1940-1999
- Years in the range 00-39 will be treated as 2000-2039
- Any dates outside of the range 1940-2039 are invalid dates

@RTVDAT8 - Retrieve current 8 digit system date/time stamp

This program will retrieve the current system date and time and return the date in *ISO format.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Sysd8	8P0	O	Field containing the system date in YYYYMMDD format
£Systm	6P0	O	Field containing the time

@RTVDTI - Retrieve date information

This program will return the following information about a given date:

- The day number of the week (where Monday is 1, Sunday is 7)
- The day number of the year
- The ISO week number of the year

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Date	7P0	I	Field containing the date in *CYMD format
£DayOfWeek	1P0	O	Day number of the week
£DayOfYear	3P0	O	Day number of the year
£WeekOfYr	2P0	O	Week number of the year

Date can be passed as either a 6 digit number (YYMMDD) or a 7 digit number (CYYMMDD). If a 6 digit number is passed, the century digit will be assumed to be 0.

@RTVDTS - Retrieve *ISO date/time stamp

This program will retrieve the current system date and time and return the date in *ISO format. Retrieved time includes the milliseconds portion of the timestamp.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Isodt	8P0	O	Field containing the system date in YYYYMMDD format
£Systm9	9P0	O	Field containing the time in HHMMSSsss

The result is the same as a CALL to the @RTVDAT8 program, but includes milliseconds in the time parameter.

@RTVDFM - Retrieve system/job date formats

This program will retrieve the values for the system date format and the current job date format.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£SysFmt1	1A	O	System date format, where Y=Year/Month/Day M=Month/Day/Year D=Day/Month/Year
£JobFmt1	1A	O	Current job date format, where Y=Year/Month/Day M=Month/Day/Year D=Day/Month/Year

@RTVDF8 - Retrieve 8 digit system/job date formats

This program will retrieve the values for the system date format and the current job date format.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£SysFmt	4A	O	System date format, where *YMD=Year/Month/Day *MDY=Month/Day/Year *DMY=Day/Month/Year
£JobFmt	4A	O	Current job date format, where *YMD=Year/Month/Day *MDY=Month/Day/Year *DMY=Day/Month/Year

@RTVEMLADR - Retrieve SMTP Email Address

This program will retrieve the SMTP email address for a specified user. Information is retrieved from the system directory.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£UserID	8A	I	Field containing the User ID in the System Distribution Directory
£UserAddr	8A	I	Field containing the User Address in the System Distribution Directory
£EmailAddr	256A	O	Field containing the user's SMTP email address

If the specified user does not have an SMTP email address defined I the System Distribution Directory, the value returned will be blanks.

@RTVIPA - Retrieve the Session IP address

This program will retrieve the TCP/IP address for the current signed-on workstation session.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£IPAddress	15A	O	Field containing the IP address in dotted decimal form *nnn.nnn.nnn.nnn*

If the current job is not a workstation session connected via TCP/IP, the value returned will be blanks.

@RTVJDT8 - Retrieve current 8 digit job date/time stamp

This program will retrieve the current job date and time. Date is returned in *ISO format.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Jobd8	8P0	O	Field containing the job date in YYYYMMDD format
£JobTm	6P0	O	Field containing the time

@RTVMSG - Retrieve Message

The Retrieve Message API is used to retrieve a specified predefined message from a message file and to copy the First Level Text in to a returned variable.

Substitution values can be specified in the £MSGDTA parameter (as a single character string containing one or more concatenated message data fields) to replace the substitution variables in the predefined message text.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Msgid	7A	I	Message ID to retrieve
£Msgf	10A	I	Message file containing the Message ID
£MsgDta	80A	I	Message data to include in the returned message
£Msg	80A	O	Returned message text

£MSGID specifies the message identifier of the predefined message that is being retrieved from the specified message file. The message ID must already exist in the message file specified.

£MSGF specifies the name of the message file containing the predefined message. The user library list is used to determine the location of the message file; all libraries in the job's library list are searched until the first match is found.

£MSGDTA specifies the substitution values that are used in the retrieved message if the predefined message contains substitution variables. If more than one substitution variable exists, the message data is concatenated in to this parameter; the formatting of the concatenated substitution data in to the message text is then performed based on the message data definition in the predefined message.

£MSG specifies the name of the character variable in the program into which the First Level Text of the retrieved message is copied.

@RVSDAT - Reformat date to a new format

This program will reformat a given date, from one date format to another. The input date will then be returned to the requesting program in the requested format.

Also available as a bound call via the NSRVPGM service program.

If the date is invalid, a value of 9999999 will be returned as the date. If the date passed to the program is either 0 or 999999, it will be ignored by the program. Note that the program remains open after the first call, to improve response.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Sysd7	7P0	I/O	Field containing the date
£Dfi7	1A	I	Date format in, either 'D' Day/Mth/Year 'M' Mth/Day/Year 'Y' Year/Mth/Day 'S' System format 'J' Job format
£Dfo7	1A	I	Date format out, either 'D' Day/Mth/Year 'M' Mth/Day/Year 'Y' Year/Mth/Day 'S' System format 'J' Job format

Where a 6 digit date is passed, a 6 digit date will be returned (i.e. the century digit is ignored).

Notes on 7 digit dates

When not passed as input the century portion of a 7digit date is implied, as follows:

- Years in the range 40-99 will be treated as 1940-1999
- Years in the range 00-39 will be treated as 2000-2039
- 6 digit dates outside of the range 1940-2039 are invalid dates

@RVSDAT8 - Reformat 8 digit date to a new format

This program will reformat a given date, from one date format to another. The input date will then be returned to the requesting program in the requested format.

Also available as a bound call via the NSRVPGM service program.

If the date passed to the routine is invalid for the format specified, a value of 99999999 will be returned as the date.

A date of 0 is an invalid date.
A date of 99999999 will be ignored by the program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Isodt	8P0	I/O	Field containing the date
£Dfi	4A	I	Date format in, either *DMY Day/Mth/Year *MDY Mth/Day/Year *YMD Year/Mth/Day *ISO System format *EUR EUR format *JIS JIS format *JUL JUL format *USA USA format *JOB Job format (6 digits) *JB8 Job format (8 digits)
£Dfo	4A	I	Date format out, either *DMY Day/Mth/Year *MDY Mth/Day/Year *YMD Year/Mth/Day *ISO System format *EUR EUR format *JIS JIS format *JUL JUL format *USA USA format *JOB Job format (6 digits) *JB8 Job format (8 digits)

Notes on 8 digit dates

The century portion of the date is implied for date formats *DMY, *MDY and *YMD, as follows:

- Years in the range 40-99 will be treated as 1940-1999
- Years in the range 00-39 will be treated as 2000-2039
- 6 digit dates outside of the range 1940-2039 are invalid dates

The century portion of the date is a required entry for date formats *ISO, *USA, *EUR, *JIS. If not entered, the century portion will be treated as 00.

Example of usage

This code will convert the contents of the FileDate field from *ISO to *JOB format and move the result into the field ScreenDate.

```
 *  ...Reverse date from *ISO format to *JOB format...
C     FileDate      Ifne      0
C     FileDate      Andne     99999999
C                   Call      '@RVSDAT8'
C     ScreenDate    Parm      FileDate      @Date8
C                   Parm      '*ISO'        @Dfi8
C                   Parm      '*JOB'        @Dfo8
C                   Else
C                   Z-add     0             ScreenDate
C                   Endif
```

@RTVOSL - Retrieve the current i5/OS Operating System version

This program will retrieve the i5/OS Operating System version installed.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Version	6A	O	Field containing the release level in the form *VnRnMn*

@RTVPLI - Retrieve the Operating System primary language identifier

This program will retrieve the primary language load identifier for the current i5/OS Operating System version installed.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£LangID	4A	O	Field containing the primary language identifier.

A list of the valid language identifier codes can be found in the IBM i5/OS Globalization (National Language Support) Guide.

A code of ‘2924’ indicates that English is the primary language installed.

@RTVRC – Retrieve current Record Count for specified file

This program will return the current number of non-deleted records in the specified file. If the specified file or member is not found, the returned record count will be zero.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£File	10A	I	Field containing the File Name
£Lib	10A	I	Field containing the Library Name
£Mbr	10A	I	Field containing the File Member Name
£RcdCnt	10P0	O	Return field containing the number of current records

Special values allowed for *£Lib* Library Name parameter:

*LIBL
*CURLIB

Special values allowed for *£Mbr* Member Name parameter:

*FIRST
*LAST

@RTVRL – Retrieve Record Length of specified file

This program will return the record length of the specified file. If the specified file is not found, the returned record length will be zero.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£File	10A	I	Field containing the File Name
£Lib	10A	I	Field containing the Library Name
£RcdLen	5P0	O	Return field containing the file record length

Special values allowed for *£Lib* Library Name parameter:

*LIBL
*CURLIB

@RTVSYSA - Retrieve system attributes

This program will retrieve information relating to the iSeries system (or partition) that the current job is running on.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Sysnm	8A	O	Field containing the System Name
£Srlnbr	8A	O	Field containing the System Serial number
£Model	4A	O	Field containing the System Model number

@RTVSVI - Retrieve server information

This program will retrieve information about the currently configured iSeries NetServer (iSeries Support for Windows Network).

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£SvrNm	15A	O	Field containing the Server Name
£SvrNmp	15A	O	Field containing the pending Server Name
£DomNm	15A	O	Field containing the Domain Name
£DomNmp	15A	O	Field containing the pending Domain Name
£Text	50A	O	Field containing the Text Description

@RTVTIM - Retrieve current system time

This program will retrieve the current system time in the format requested.

Also available as a bound call via the NSRVPGM service program.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£TimeFmt	1A	O	Time format requested, where S=System time G=GMT
£Time	6P0	O	Field containing the system time in HHMMSS format

The return of GMT Time (or *Coordinated Universal Time, UTC*) is dependent on the correct setting of the system value QUTCOFFSET.

QUTCOFFSET is the system value containing the coordinated universal time offset, which is a value that defines the time difference between the local time and UTC (also known as Greenwich Mean Time, or Zulu). Refer to the IBM iSeries Work Management guide for further details.

If GMT time was requested and QUTCOFFSET is not set, the program will return 999999 as the time.

If GMT time was requested and the system country code is GB (United Kingdom), the program will assume the system time is being requested.

@SNDSTS - Send a status message

This program will format a message and send it as a status message.

It is mainly supplied for use by RPG programs, due to the language restriction that they cannot directly send a status message.

Parameter List

Parameter	*Defn*	*Usage*	*Description*
£Msgid	7A	I	Message ID to send
£Msgf	10A	I	Message file containing MSGID
£StsDta	120A	I	Status Message data to include in message

If the £Msgid parameter is blank, the standard IBM message ID CPF9898 will be assumed and the contents of the message data parameter (£StsDta) will be sent as the status message.

The status message data (£StsDta) is used in one of two ways; either it contains the message to be sent (£Msgid is blank) or it contains variable data that is to be loaded into the message specified (£Msgid) before it is sent as a message.

Date 'Conversion Assistant' modules

The following NUTIL 'Date Conversion Assistant' programs can be used for manipulating your date data between 6 and 8 digit date formats.

Conversion Module	*Purpose*
@DCDDMMYY	Convert DDMMYY (day/month/year) format to DDMMYYYY
@DCMMDDYY	Convert MMDDYY (month/day/year) format to MMDDYYYY
@DCYY	Convert YY (year) format to YYYY
@DCYYDDD	Convert YYDDD (julian) format to YYYYDDD (longjul)
@DCYYMMDD	Convert YYMMDD (year/month/day) format to YYYYMMDD
@DCYYPP	Convert YYPP (year/period) format to YYYYPP
@DCYYPPWW	Convert YYPPWW (year/period/week) format to YYYYPPWW
@DCYYPPWWD	Convert YYPPWWD (year/period/week/day) format to YYYYPPWWD
@DCYYWW	Convert YYWW (year/week) format to YYYYWW
@DCYYWWD	Convert YYWWD (year/week/day) format to YYYYWWD

In all cases of the above, the following is true:

- The call has two parameters, *DateVarIn* (date variable input) and *DateVarOut* (date variable output).
- Where the value of *DateVarIn* is either 0 or all 9's it will be ignored by the conversion (not @DCYY)
- No value judgement is made on the input data; it is assumed that you are passing the conversion module valid date data in the form expected.
- Years in the range 28-99 will be converted to 1928-1999
- Years in the range 00-27 will be converted to 2000-2027
- The program remains open after the first call, to improve response.
- The program can either be called directly, or as a bound call via the NUTIL NDCSRV service program (which can be assessed from the NBNDDIR binding directory).

@DCDDMMYY - Convert DDMMYY to DDMMYYYY

This program will convert a Day/Month/Year date variable into full 4 digit year representation.

Parameter List

Field	*Defn*	*Usage*	*Description*
@DdMmYy	6P0	I	DateVarIn, in DDMMYY format
@DdMmYyyy	8P0	O	DateVarOut, in DDMMYYYY format

@DCMMDDYY - Convert MMDDYY to MMDDYYYY

This program will convert a Month/Day/Year date variable into full 4 digit year representation.

Parameter List

Field	*Defn*	*Usage*	*Description*
@MmDdYy	6P0	I	DateVarIn, in MMDDYY format
@MmDdYyyy	8P0	O	DateVarOut, in MMDDYYYY format

@DCYY - Convert YY to YYYY

This program will convert a Year date variable into full 4 digit year representation.

Parameter List

Field	*Defn*	*Usage*	*Description*
@Yy	2P0	I	DateVarIn, in YY format
@Yyyy	4P0	O	DateVarOut, in YYYY format

@DCYYDDD - Convert YYDDD to YYYYDDD

This program will convert a Julian date variable into full 4 digit year representation (LongJulian).

Parameter List

Field	*Defn*	*Usage*	*Description*
@YyDdd	5P0	I	DateVarIn, in YYDDD format
@YyyyDdd	7P0	O	DateVarOut, in YYYYDDD format

@DCYYMMDD - Convert YYMMDD to YYYYMMDD

This program will convert a Year/Month/Day date variable into full 4 digit year representation.

Parameter List

Field	*Defn*	*Usage*	*Description*
@YyMmDd	6P0	I	DateVarIn, in YYMMDD format
@YyyyMmDd	8P0	O	DateVarOut, in YYYYMMDD format

@DCYYPP - Convert YYPP to YYYYPP

This program will convert a Year/Period date variable into full 4 digit year representation.

Parameter List

Field	*Defn*	*Usage*	*Description*
@YyPp	4P0	I	DateVarIn, in YYPP format
@YyyyPp	6P0	O	DateVarOut, in YYYYPP format

@DCYYPPWW - Convert YYPPWW to YYYYPPWW

This program will convert a Year/Period/Week date variable into full 4 digit year representation.

Parameter List

Field	*Defn*	*Usage*	*Description*
@YyPpWw	6P0	I	DateVarIn, in YYPPWW format
@YyyyPpWw	8P0	O	DateVarOut, in YYYYPPWW format

@DCYYPPWWD - Convert YYPPWWD to YYYYPPWWD

This program will convert a Year/Period/Week/DayOfTheWeek date variable into full 4 digit year representation.

Parameter List

Field	*Defn*	*Usage*	*Description*
@YyPpWwD	7P0	I	DateVarIn, in YYPPWWD format
@YyyyPpWwD	9P0	O	DateVarOut, in YYYYPPWWD format

@DCYYWW - Convert YYWW to YYYYWW

This program will convert a Year/Week date variable into full 4 digit year representation.

Parameter List

Field	*Defn*	*Usage*	*Description*
@YyWw	4P0	I	DateVarIn, in YYWW format
@YyyyWw	6P0	O	DateVarOut, in YYYYWW format

@DCYYWWD - Convert YYWWD to YYYYWWD

This program will convert a Year/Week/DayOfTheWeek date variable into full 4 digit year representation.

Parameter List

Field	*Defn*	*Usage*	*Description*
@YyWwD	5P0	I	DateVarIn, in YYWWD format
@YyyyWwD	7P0	O	DateVarOut, in YYYYWWD format

Business Contingency Planning

In the event of a serious failure of the CPU for which you have licensed this copy of NUTIL, it may be necessary to temporarily transfer your business processing to another CPU.

While it is not possible to temporarily transfer your NUTIL product license to another CPU, it is possible to temporarily run your copy of NUTIL on another machine. The way to do this is as follows:

Using your most recent backup copy of NUTIL, restore the library from your offline storage media onto the temporary CPU. Now type the following commands *exactly as shown* on a command line:

```
ADDLIBLE    LIB(NUTIL)
CHGDTAARA   DTAARA(NUTIL (17 8))  VALUE(*DCS)
```

Now type CALL NU9000 and press *enter* to display the NUTIL Installation Attributes panel. Verify that there is no ‘copyright violation’ error message displayed, and that an expiry date 15 days from now has been established for this temporary copy.

NUTIL will now function correctly on the temporary CPU. Please note that this copy is installed as a demonstration version and is bound by the terms and conditions of a demonstration version as explained in your license agreement.

> Please note that this Disaster Contingency Support option *cannot be extended beyond 15 days*. Any attempt to re-use the *DCS option will result in a copyright violation message and NUTIL will cease to function.

Once your operations on your main CPU have been re-established you should remove this copy of NUTIL from the temporary CPU.

Support Contacts

Europe, Middle East and Africa

Navan (UK) Limited
The Courtyard
Brooklands Broughton
Milton Keynes MK16 0HU
England

Email: helpdesk@navan.co.uk

Rest of the World

Navan Australia Pty Ltd
22 Torrington Road
Maroubra
NSW 2035
Australia

Email: helpdesk@navan.com.au

The latest version of this manual, as well as the latest PTF's to this product, can always be downloaded from the Navan Internet Website, which can be found at

http://www.navan.co.uk