Stages Built Into NetRexx Pipelines 4.05 & CMS Pipelines V7R1 and Their Differences

How to Read Syntax Diagrams

Special diagrams (often called railroad tracks) are used to show the syntax of external interfaces.

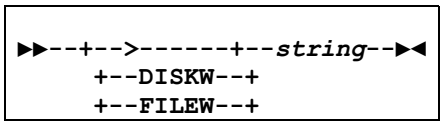
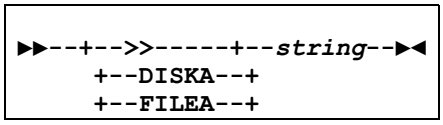
To read a syntax diagram, follow the path of the line. Read from left to right and top to bottom.

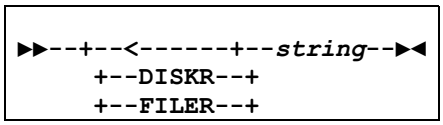
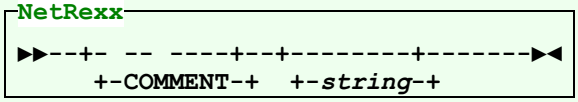
- The ▶▶--- symbol indicates the beginning of the syntax diagram.
- The ---▶ symbol, at the end of a line, indicates that the syntax diagram is continued on the next line.
- The ▶--- symbol, at the beginning of a line, indicates that the syntax diagram is continued from the previous line.
- The ---▶◀ symbol indicates the end of the syntax diagram.

Within the syntax diagram, items on the line are required, items below the line are optional, and items above the line are defaults.

Pipelines Builtin Stages

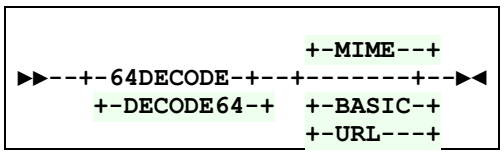
Show Stages Implemented in: ☑ **NetRexx Pipelines:** ☑ **CMS Pipelines:**
Show All Details: ☑ (Double click on a row to turn it on/off.)
☑ **Highlight NetRexx Only / CMS Only:**
Show Only Changes: ◉ **None** ○ **4.05** ○ **3.11** ○ **3.09**

> ***diskw*** ***filew*** *3.09*	**Replace or Create a File**  • delegates to **diskw**.
>> ***diska*** ***filea*** *3.09*	**Append to or Create a File**  • delegates to **diska**.
>>mdsk	**Append to or Create a CMS File on a Mode** • Not implemented in Netrexx Pipelines.
>>mvs	**Append to a Physical Sequential Data Set** • Not implemented in Netrexx Pipelines.
>>oe	**Append to or Create an OpenExtensions Text File** • Not implemented in Netrexx Pipelines.
>>sfs	**Append to or Create an SFS File** • Not implemented in Netrexx Pipelines.
>>sfsslow	**Append to or Create an SFS File** • Not implemented in Netrexx Pipelines.

>mdsk	**Replace or Create a CMS File on a Mode** • Not implemented in Netrexx Pipelines.
>mvs	**Rewrite a Physical Sequential Data Set or a Member of a Partitioned Data Set** • Not implemented in Netrexx Pipelines.
>oe	**Replace or Create an OpenExtensions Text File** • Not implemented in Netrexx Pipelines.
>sfs	**Replace or Create an SFS File** • Not implemented in Netrexx Pipelines.
< *diskr* *filer* *3.09*	**Read a File**  • Implemented as in CMS; delegates to **diskr**.
<mdsk	**Read a CMS File from a Mode** • Not implemented in Netrexx Pipelines.
<mys	**Read a Physical Sequential Data Set or a Member of a Partitioned Data Set** • Not implemented in Netrexx Pipelines.
<oe	**Read an OpenExtensions Text File** • Not implemented in Netrexx Pipelines.
<sfs	**Read an SFS File** • Not implemented in Netrexx Pipelines.
<sfsslow	**Read an SFS File** • Not implemented in Netrexx Pipelines.
-- *comment* *3.09*	**Comment Stage, No Operation**  • delegates to **comment**. • Not in CMS Pipelines; • This is a STAGE, not a programming comment. It must have a SPACE after **--**. • It must have either a stageEnd or pipeEnd. • If used before a driver stage, it must have a pipeEnd.
3277bfra	**Convert a 3270 Buffer Address Between Representations** • Not implemented in Netrexx Pipelines.
3277enc	**Write the 3277 6-bit Encoding Vector** • Not implemented in Netrexx Pipelines.

64decode
decode64
3.11

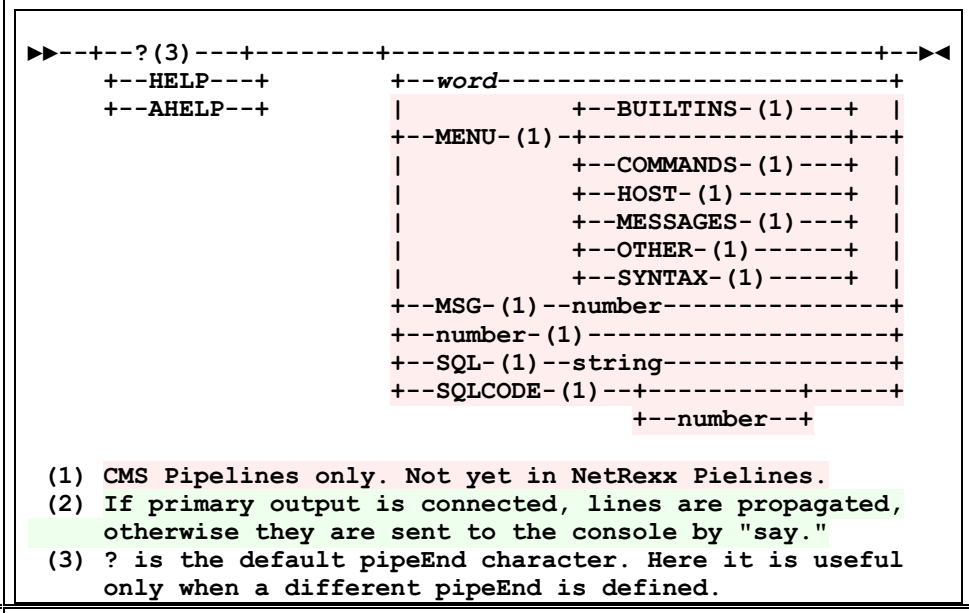
Decode Base-64 Format



- NOTE: CMS is only 64DECODE, and does not have the options; it does MIME.
- BASIC - Output is mapped to a set of characters lying in A-Za-z0-9+/. The encoder does not add any line feed in output, and the decoder rejects any character other than A-Za-z0-9+/.
- URL - Output is mapped to set of characters lying in A-Za-z0-9+_. Output is URL and filename safe.
- MIME - Output is mapped to MIME friendly format. Output is represented in lines of no more than 76 characters each, and uses a carriage return '\r' followed by a linefeed '\n' as the line separator. No line separator is present to the end of the encoded output.
- 3.11: New to NetRexx. Add MIME, BASIC, & URL options.

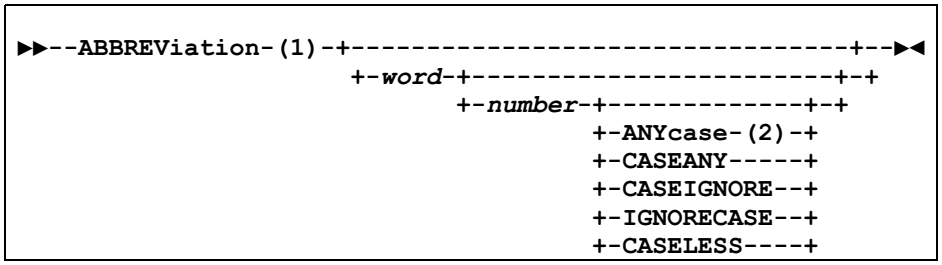
?
ahelp
help

Display Help for Pipelines



abbreviation
abbreviatio
abbreviati
abbreviat
abbrevi
abbrev

Select Records that Contain an Abbreviation of a Word in the First Positions



- (1) ABBREViation must be ABBREV in CMS
- (2) ANYcase must be ANYCASE in CMS

acigroup

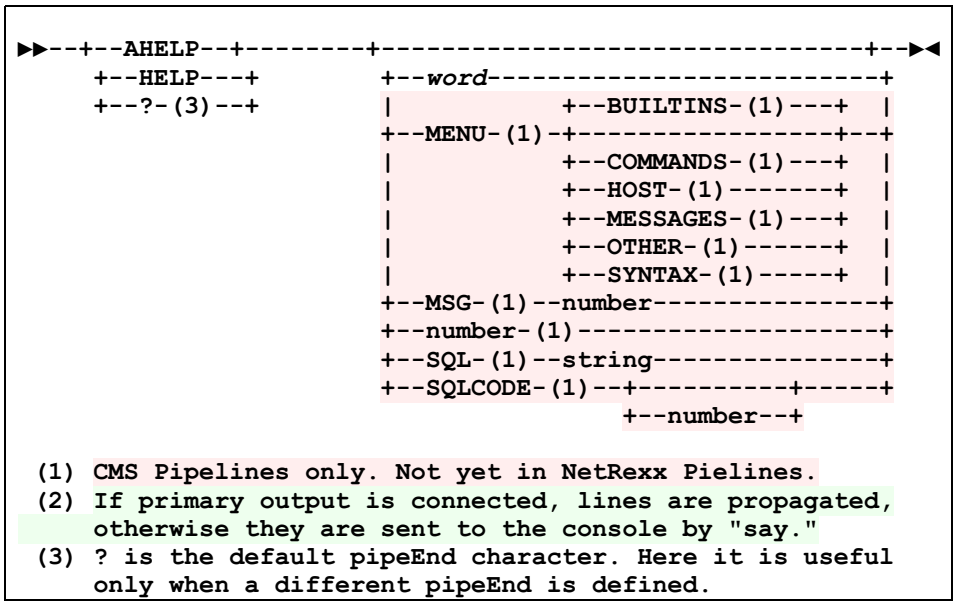
Write ACI Group for Users

- Not implemented in Netrexx Pipelines.

addrdw

Prefix Record Descriptor Word to Records

- Not implemented in Netrexx Pipelines.

adrspace	**Manage Address Spaces** • Not implemented in Netrexx Pipelines.
aftfst	**Write Information about Open Files** • Not implemented in Netrexx Pipelines.
aggrc	**Compute Aggregate Return Code**
ahelp ***help*** ***?***	**Display Help for Pipelines** 

all
4.05

Select Lines Containing Strings (or Not)

```
▶▶--ALL---+----------+--| expression |---▶◀
          +--%debug--+ (6)
          +--%dump---+ (7)
          +--%see----+ (8)
```

Notes:

- (1) "expression" consists of one or more delimitedstrings separated by logical ANDs, ORs, and NOTs, and grouped, if needed, by parentheses.
- (2) "&" is used for AND.
- (3) Since "|" is the default stage separator, "!" may be used for OR.
- (4) Since NetRexx uses "(" and ")" for options -- which are not used in the ALL stage -- "[" and "]" must be used for parentheses.
- (5) CMS Pipelines, having originated on 3270 terminals, uses "¬" for NOT. This symbol is not readily typed on terminals running NetRexx Pipelines, so as alternatives, "\\", used by NetRexx, (it needs to be doubled to "escape" it) or "^", used by KEX, NOT symbols may be used as alternatives.
- (6) %debug (must be lowercase) NetRexx Pipelines writes the logic line to the file ALL.DEBUG in the current directory. Windows may make it all.debug . CMS Pipelines writes the constructed pipeline (of LOCATE and NLOCATE stages) to ALL DEBUG A.
- (7) %dump (must be lowercase) - writes to the primary output stream as the first record. NetRexx Pipelines writes the logic line. CMS Pipelines writes constructed pipeline.
- (8) %see (must be lowercase) - NetRexx Pipelines Only. Writes the logic line to the standard output (terminal).
- CMS Pipelines uses is own logic order. NetRexx Pipelines uses regular NetRexx logic.

Examples:

- ```
  literal NetRexx is Good,NetRexx is Great,NetRexx is Fantastic |
  split , |
  all /a/ |
  cons

  ▶NetRexx is Great
  ▶NetRexx is Fantastic
  ```

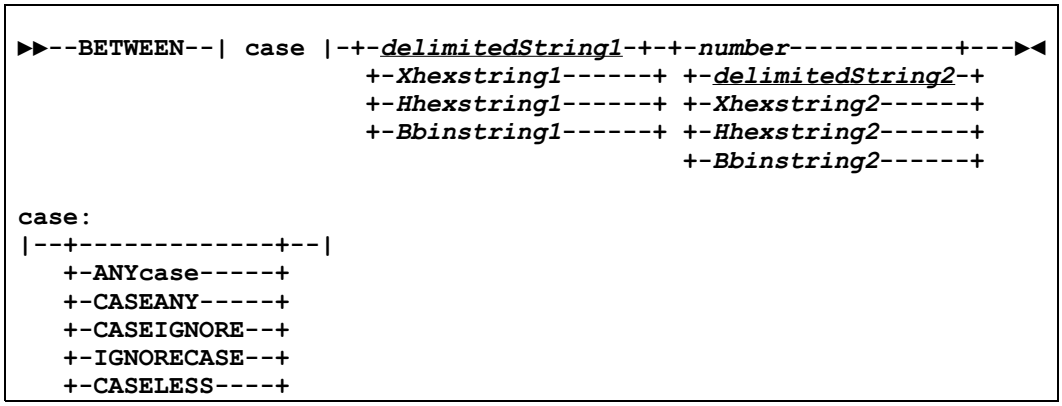
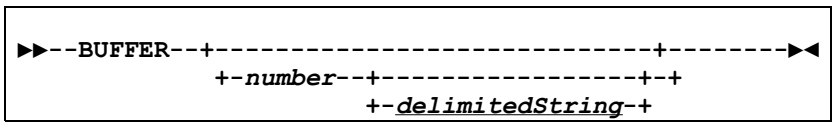
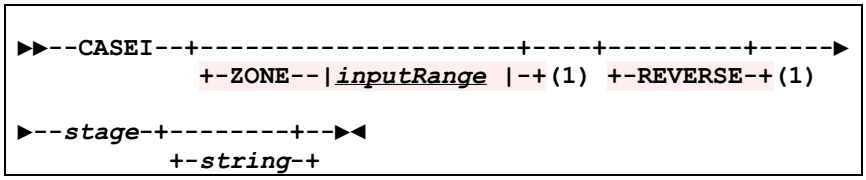
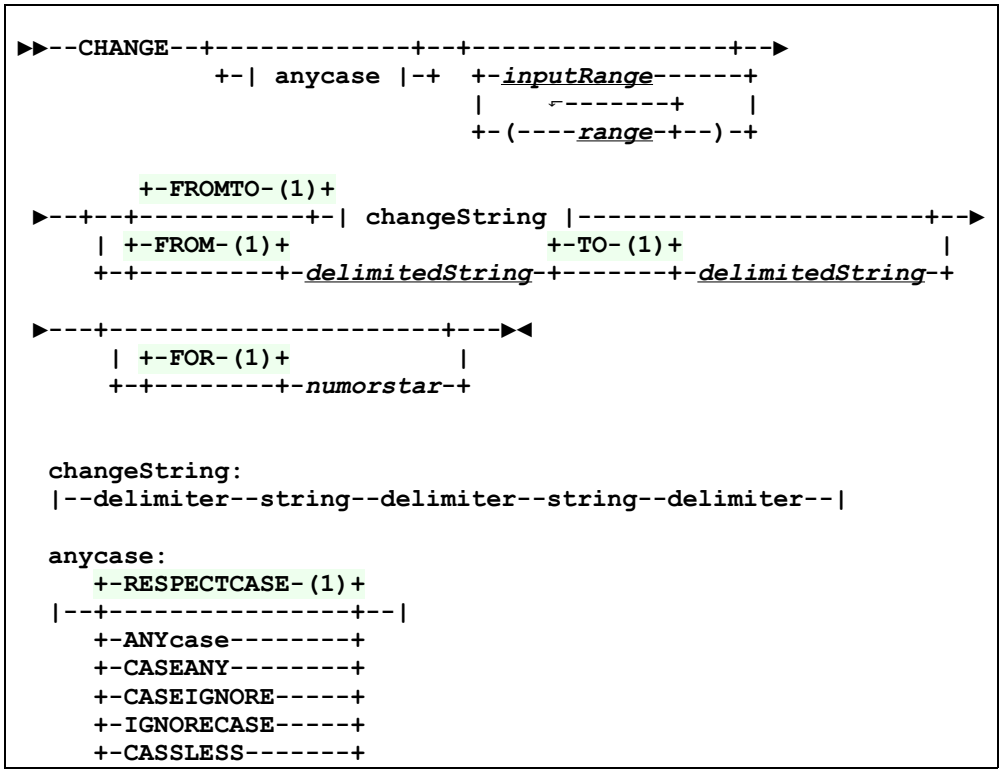
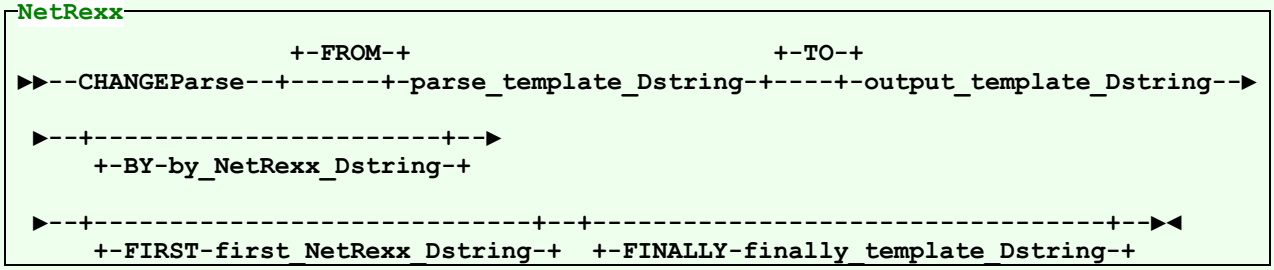
- ```
  literal NetRexx is Good,NetRexx is Great,NetRexx is Fantastic |
  split , |
  all / G/ & [/oo/ ! /F/] |
  cons

  ▶NetRexx is Good
  ```

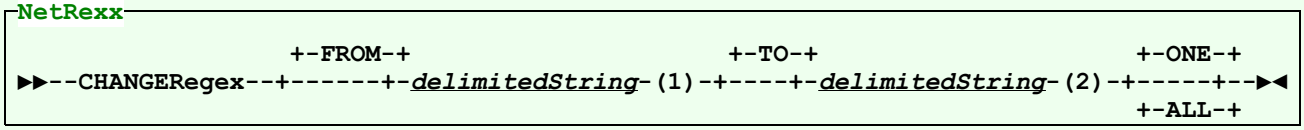
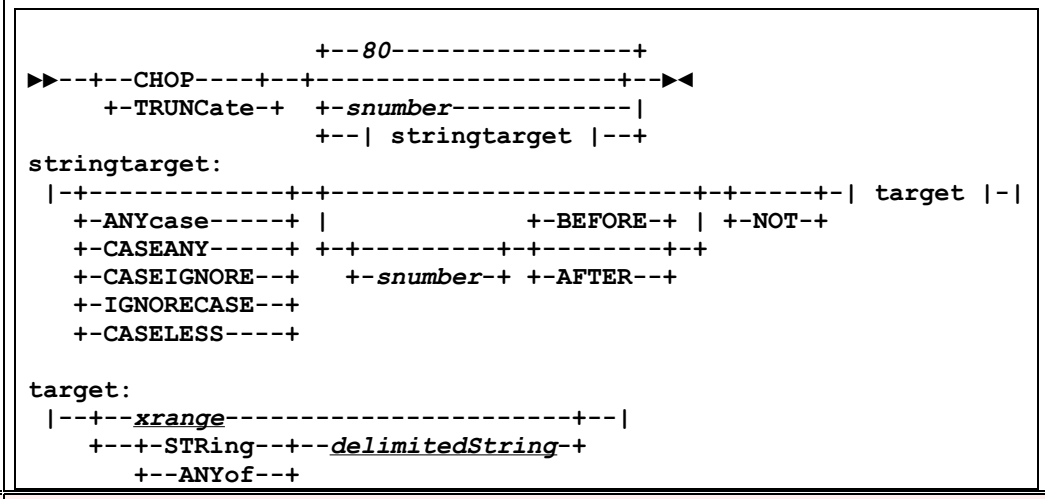
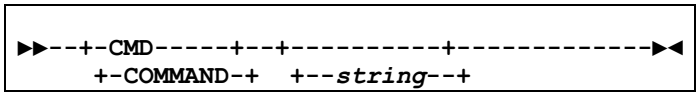
- ```
  literal NetRexx is Good,NetRexx is Great,NetRexx is Fantastic |
  split , |
  all /R/ & [/oo/ ! /F/] |
  cons

  ▶NetRexx is Good
  ▶NetRexx is Fantastic
  ```

alter

Change the contents of records, from one character to another

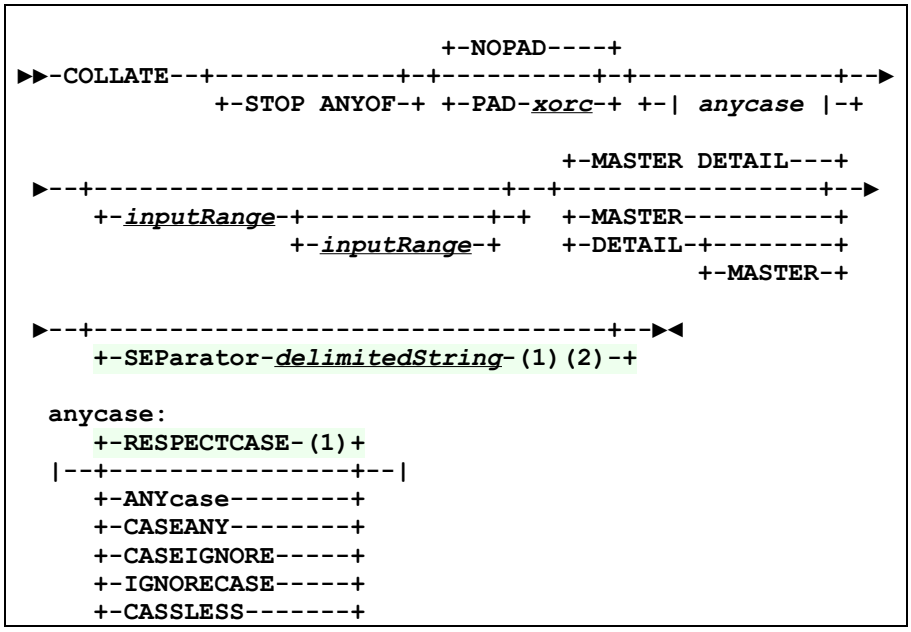
```
NetRexx

              +--,--+
            ┌-+-----+---------------------------------------+
            |   +-FROM-+                +-TO-+                |
▶▶--ALTER--v---+------+--+-char1---+--+----+--+-char2---+--+--▶◀
                          +-decimal-+          +-decimal-+
                          +-Hhex----+          +-Hhex----+
                          +-Xhex----+          +-Xhex----+
```

- An variation on the theme of Xedit's ALTER.
- There are pairs of char1s and char2s, optionally separated by commas.
- For each pair of char1 and char2, this changes ALL char1s to char2, like Xedit's 4th and 5th parameters were 1 and 1.
- The chars can be single characters, 2 or 3 digit decimal numerical representations, or beginning with H, h, X, or x hexidecimal representations.
- Also see TRANSLATE / XLATE.
- Not in CMS Pipelines.
```

alserv	**Manage the Virtual Machine's Access List** • Not implemented in Netrexx Pipelines.
apldecode	**Process Graphic Escape Sequences, Old APL language** • Not implemented in Netrexx Pipelines.
aplencode	**Generate Graphic Escape Sequences, Old APL language** • Not implemented in Netrexx Pipelines.
append	**Put Output from a Device Driver after Data on the Primary Input Stream** ▶▶--APPEND--*string*------▶◀
array	**Read or Write an Array** • Pipes for NetRexx
arraya	**Read or Write an Array** • Pipes for NetRexx
arrayr	**Read or Write an Array** • Pipes for NetRexx
arrayw	**Read or Write an Array** • Pipes for NetRexx
asatomc	**Convert ASA Carriage Control to CCW Operation Codes. Old printer control** • Not implemented in Netrexx Pipelines.
asmcont	**Join Multiline Assembler Statements** • Not implemented in Netrexx Pipelines.
asmfind	**Select Statements from an Assembler File as XEDIT Find** • Not implemented in Netrexx Pipelines.
asmnfind	**Select Statements from an Assembler File as XEDIT NFind** • Not implemented in Netrexx Pipelines.
asmxpnd	**Expand Joined Assembler Statements** • Not implemented in Netrexx Pipelines.
beat	**Mark when Records Do not Arrive within Interval** • Not implemented in Netrexx Pipelines.

between
3.09

Select Records Between Labels



bfs
hfs

Read or Append File in the Hierarchical File System

- Not implemented in Netrexx Pipelines.

bfsdirectory
bfsdir
hfsdirectory
hfsdir

Read Contents of a Directory in a Hierarchical File System

- Not implemented in Netrexx Pipelines.

bfsquery
bfsq
hfsquery
hfsq

Write Information Obtained from OpenExtensions into the Pipeline

- Not implemented in Netrexx Pipelines.

bfsreplace
bfsrep
hfsreplace
hfsrep

Replace the Contents of a File in the Hierarchical File System

- Not implemented in Netrexx Pipelines.

bfsstate
bfsstat
hfsstate
hfsstat

Obtain Information about Files in the Hierarchical File System

- Not implemented in Netrexx Pipelines.

bfsxecute
bfsx
hfsxecute
hfsx

Issue OpenExtensions Requests

- Not implemented in Netrexx Pipelines.

block

Block to an External Format

- Not implemented in Netrexx Pipelines.

browse
brw

Display Data on a 3270 Terminal

- Not implemented in Netrexx Pipelines.

buffer

Buffer Records



c14to38

Combine Overstruck Characters to Single Code Point. Old printer.

casei

Run Selection Stage in Case Insensitive Manner



- (1) CMS Pipelines only.

change

Substitute Contents of Records



- (1) NetRexx Pipelines only.

changeparse
changepars
changepar
changepa
changep
3.11

Change the contents of records, using Rexx Parse. Calculations can be done.

NetRexx



- Records are parsed via the parse_template_delimited_string.
- Variables are named $n, where n is 1 to 9.
- The by_NetRexx_delimited_string is interpreted. This is 0 or more semicolon separated NetRexx statements, probably using the $n variables, which can have the value altered.
- Other variables may be used, and are persistent while the stage is active, so can be used as accumulators.
- The values of the variables are put into the output_template_delimited_string replacing $n.
- For a literal $n that won't be changed, use $$n.
- A first_NetRexx_delimited_string, if present, is interpreted before reading any record from the primary input steam. This is 0 or more semicolon separated NetRexx statements, probably using the $n variables. Any variables used in the by_NetRexx_delimited string must be defined here.
- A finally_template_delimited_string, if present, is written as a final output record after the primary input stream is finished, using the $n's.
- Any keyword phrases must, in any order, follow any non-keyworded FROM & TO phrases.
- This is NetRexx Pipelines only, not in CMS.

Examples:

- `changeparse / 2 $1 +1/ /The second letter is "$1". $$1 won't be changed./`
- `changeparse from / 2 $1 +1/ to /The second letter is "$1". $$1 won't be changed./`
- `changeparse from / . $2 . 50 $5 +5 / to /The product is $1/ by /$1 = $2 * $5/`
- ```
  changeparse from / . $2 . 50 $5 +5 / ,
  to /The product is $1/ ,
  by /$1 = $2 * $5;$3 = $3 + $1/ ,
  first /$3 = 0/ ,
  finally /$3 is the total/
  ```
  ```

changeregex ***changerege*** ***changereg*** ***changere*** ***changer*** ***3.09***	**Substitute Contents of Records using Java Regular Expressions** **NetRexx**  - Uses the Java RegEx classes and its dialect of RegEx. See Java's **Pattern** class and **replaceFirst** and **replaceAll** methods of **String** for full documentation. - (1) First, FROM, delimitedString is a Java *RegEx expresion* for what is to be replaced. - (2) Second, TO, delimitedString is the replacement string. It may contain elements from the first one. - Ths is NetRexx Pipelines only, not in CMS.
chop ***truncate*** ***truncat*** ***trunca*** ***trunc***	**Truncate the Record** 
cipher	**Encrypt and Decrypt Using a Block Cipher** - Not implemented in Netrexx Pipelines.
ckddeblock	**Deblock Track Data Record** - Not implemented in Netrexx Pipelines.
cmd ***command***	**Issue OS Commands, Write Response to Pipeline**  - input stream 0 is for commands - input stream 1 is stdin - output stream 0 is stdout - output stream 1 is the return code - output stream 2 is stderr
cms	**Issue CMS Commands, Write Response to Pipeline '**

collate
3.11

Collate Streams

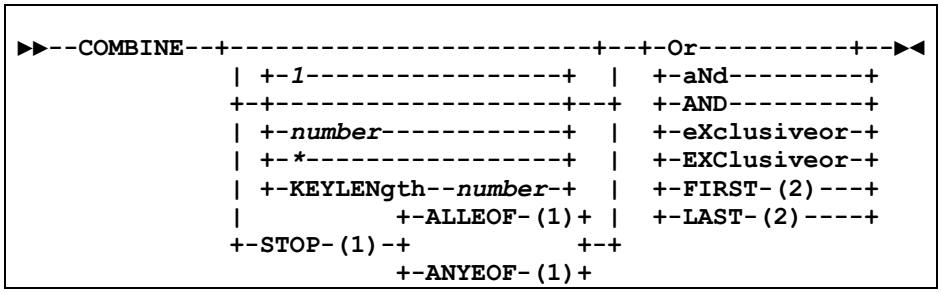


- Not implemented in Netrexx Pipelines.

- (1) NetRexx Pipelines only.
- (2) delimitedString record is put before each Master Record
  (or after if DETAIL MASTER order) on the primary output stream.
- 3.11 New to NetRexx Pipelines. Add SEParator option.

combine

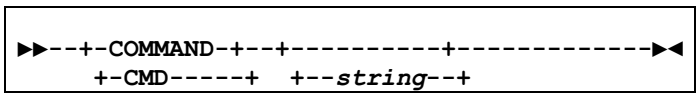
Combine Data from a Run of Records



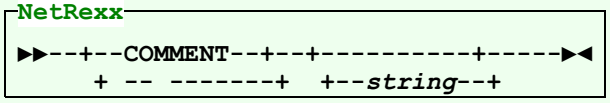
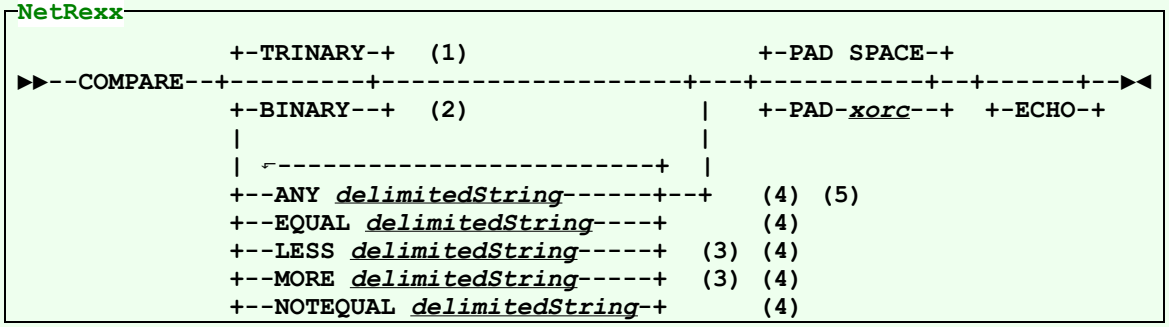
- (1) Only for use with secondary input streams. Only options from this column usable with any secondary input streams.
  (This is poorly documented in CMS Pipelines. This is a best guess of their intentions.)
- (2) Not usable with STOP and secondary streams.

command
cmd

Issue OS Commands, Write Response to Pipeline

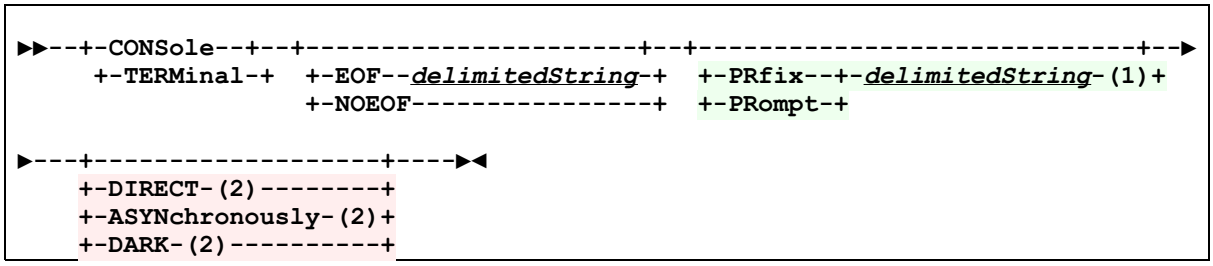


- input stream 0 is for commands
- input stream 1 is stdin
- output stream 0 is stdout
- output stream 1 is the return code
- output stream 2 is stderr

comment **--** ***3.09***	**Comment stage** NetRexx  • Not in CMS Pipelines; • This is a STAGE, not a programming comment. It must have a SPACE after --. • It must have either a stageEnd or pipeEnd. • If ended with a stageEnd, it passes records through on primary input to output streams. • If ended with a pipeEnd, it does NOT pass records through. • If used before a driver stage, it must have a pipeEnd.
compare	**Compare Primary and Secondary Streams, Write the Result** NetRexx  • (1) -1 = Primary is shorter/less, 0 = equal, 1 = Secondary is shorter/less • (2) 0 = equal, 1 = not equal • (3) Primary is LESS/shorter (or MORE/longer) than secondary • (4) *DStrings* can use any of the following escapes (or the lowercase) for the unequal situation: ◦ \C (count) for the record number, ◦ \B (byte) for column number ◦ \P (primary) for the primary stream record ◦ \S (secondary) for the secondary stream record ◦ \L (Least) for the stream number that is shorter, -1 if equal ◦ \M (Most) for the stream number that is longer, -1 if equal • (5) Equal or not, this DString precedes any of the others. • (6) This is NetRexx Pipelines only, not included in CMS • (7) In reporting \P & \S, control charactors, except new line, \n, are transliterated to [blob, 219.d2c()] • (8) Without ECHO, this stops and reports at first non-compare. With ECHO, each primary input is reported; after first non-compare primary input stream records continue to be read and reported, but no testing is done. • (9) Options work in any order • Input streams: ◦ 0: Data 1 ◦ 1: Data 2 • Output streams: ◦ 0: Result (single record, possibly multiple lines) ◦ 1: Last primary record read at first no match, or end of stream ◦ 2: Last secondary record read at first no match, or end of stream
configure	**Set and Query CMS Pipelines Confguration Variables** • Not implemented in Netrexx Pipelines.

console
consol
conso
cons
cons
terminal
termina
termin
termi
term
3.11

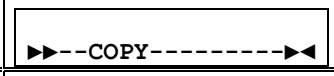
Read or Write the Terminal in Line Mode



- (1) NetRexx only
  On first stage, delimitedString is put out as a prompt
  On other stages, each line is prefixed with delimitedString
  Outout to next stage does NOT include delimitedString
  Either keyword can be used for either stage
- (2) CMS only

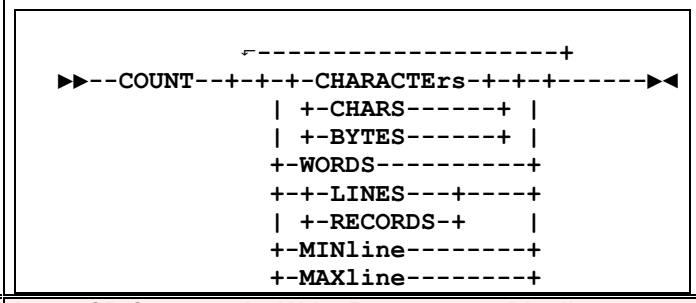
copy

Copy Records, Allowing for a One Record Delay



count

Count Lines, Blank-delimited Words, and Bytes



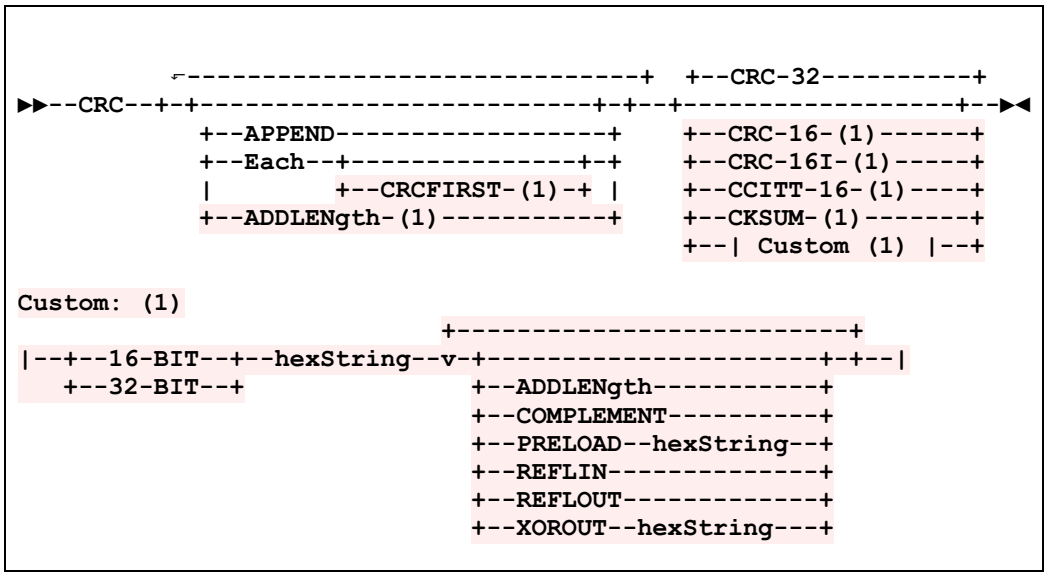
cp

Issue CP Commands, Write Response to Pipeline

- Not implemented in Netrexx Pipelines.

crc
4.05

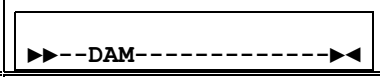
Compute Cyclic Redundancy Code



- (1) Not implemented in Netrexx Pipelines.
- (2) CRC stage uses secondary output, if connected.

dam

Pass Records Once Primed



dateconvert
dateconver
dateconve
dateconv

Convert Date Formats

```
▶▶--DATECONVert--+-----------------+--------▶
                 +--inputRange-(3)-+

    +-SHOrtdate ISOdate-----------------+ +-WINDOW -50----------+
 ▶-+-----------------------------------+-+--------------------+--▶
    |                    +-ISOdate----------+ +-WINDOW-signednumber-+
    +-| Inputformat |-+-+------------------+ +-BASEYEAR-yearnumber-+
    |     +-PREFACE-+ | +-| Outputformat |-+
    +-NOW-+-(5)-----+-+
          +-APPEND--+

                    +--MIDNIGHT--(4)-+
 ▶--+-----------+--+----------------+----------▶◀
     +--TIMEOUT--+  +--NOON--(4)-----+
```

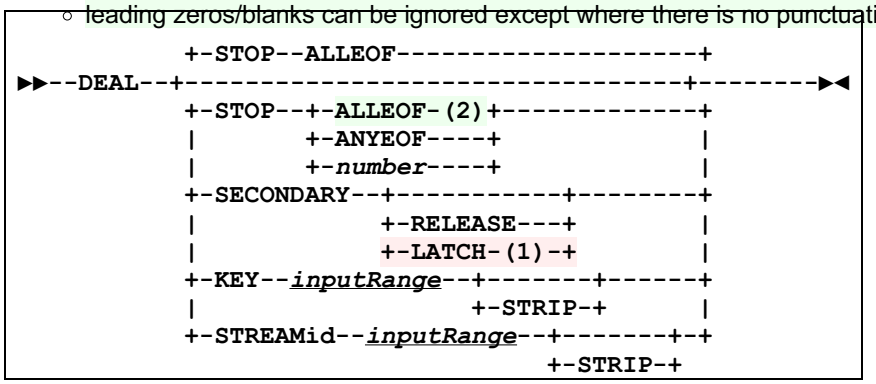
```
Inputformat,       Outputformat:
SHOrtdate      }   mm/dd/yy hh:mm:ss.uuuuuu
USA_SHORT      }
REXX_DATE_U    }

FULldate       }   mm/dd/yyyyyyy hh:mm:ss.uuuuuu
USA            }

ISO_SHORT          yy-mm-dd hh:mm:ss.uuuuuu
ISOdate            yyyyyyy-mm-dd hh:mm:ss.uuuuuu
DB2_SHORT          yy-mm-dd-hh.mm.ss.uuuuuu
DB2                yyyyyyy-mm-dd-hh.mm.ss.uuuuuu
VMDATE  (2)
NORMAL             dd mmm yyyyyyy hh:mm:ss.uuuuuu
CSL_SHORT      }   yy/mm/dd hh:mm:ss.uuuuuu
REXX_DATE_O    }
CSL                yyyyyyy/mm/dd hh:mm:ss.uuuuuu
PIPE_SHORT         yymmddhhmmssuuuuuu
PIPE           }   yyyymmddhhmmssuuuuuu
REXX_DATE_S    }
EURSHORT           dd.mm.yy hh:mm:ss.uuuuuu
EUR                dd.mm.yyyyyyy hh:mm:ss.uuuuuu
JULIAN_SHORT       yy.ddd hh:mm:ss.uuuuuu
JULIAN             yyyyyyy.ddd hh:mm:ss.uuuuuu
TOD_ABSOLUTE }  (2)
TODABS       }  (2)
SCIENTIFIC_ABSOLUTE }  (2)
SCIABS              }  (2)
POSIX              ssssssss
TOD_RELATIVE }  (2)
TODREL       }  (2)
SCIENTIFIC_RELATIVE }  (2)
SCIREL              }  (2)
MET  (2)

The following can be REXX_DATE_x, REXXx, or Rx
REXX_DATE_B  (2)
REXX_DATE_C  (2)
REXX_DATE_D        ddd hh:mm:ss.uuuuuu
REXX_DATE_E        dd/mm/yy hh:mm:ss.uuuuuu
REXX_DATE_E_LONG   dd/mm/yyyyyyy hh:mm:ss.uuuuuu
REXX_DATE_J        yyddd hh:mm:ss.uuuuuu
REXX_DATE_J_LONG   yyyyddd hh:mm:ss.uuuuuu
REXX_DATE_M        mmmmmmmmm (output only)
REXX_DATE_N_SHORT  dd mmm yy hh:mm:ss.uuuuuu
REXX_DATE_N        dd mmm yyyy hh:mm:ss.uuuuuu
REXX_DATE_W        wwwwwwwww (output only)
```

- (1): SPACE is optional here.
- (2) Not implemented in NetRexx Pipelines at this time; mainly mainframe useful only.
- (3): NetRexx Pipelines uses IRange which gives a superset of range options.
- (4): NetRexx Pipelines only. What time to assume if blank time on input.
- (5): NetRexx Pipelines only.
  - Use current local date time.
  - Any Inputrange is ignored.
  - Any output format can be used.
  - PREFACE Write the date record before passing the input to the output.

- APPEND Write the output record after passing the input to the output.
- Bad date records out on secondary. Or error if not connected.
- NetRexx Pipelines is more forgiving on input format:
  - LONG, _SHORT, and normal can all read either 2- or 4-digit years and process per the format;
  - leading zeros/blanks can be ignored except where there is no punctuation.

deal

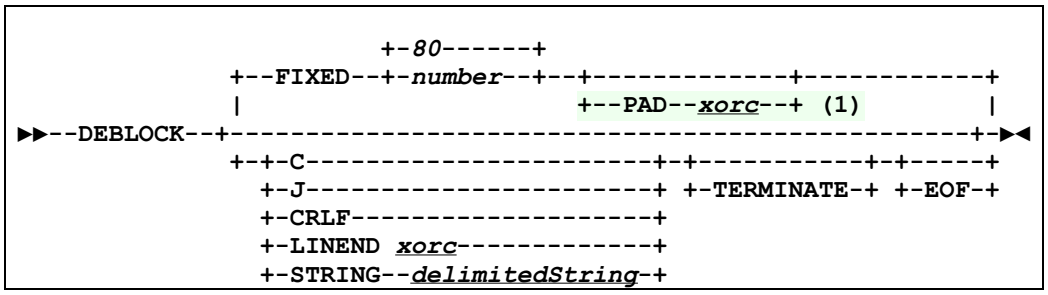
Pass Input Records to Output Streams Round Robin



- (1) Not yet in NetRexx Pipelines
- (2) Not CMS
- Since Java dispatches the stage threads, DEAL may not see a sever immediately, as the severing thread can get multitasked. This can make options like 'ANYEOF' work in unexpected ways.

deblock

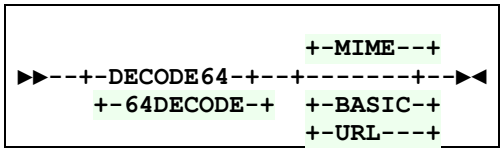
Deblock External Data Formats



- CMS has many more mainframe centric formats that NetRexx Pipelines does not process.
- (1) Not CMS Pipelines

decode64
64decode
3.11

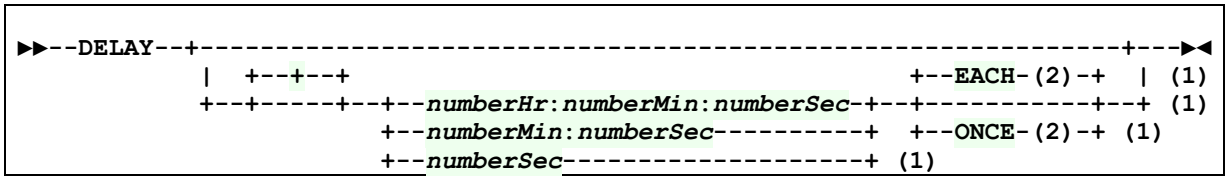
Decode Base-64 Format



- NOTE: CMS is only 64DECODE, and does not have the options; it does MIME.
- BASIC - Output is mapped to a set of characters lying in A-Za-z0-9+/. The encoder does not add any line feed in output, and the decoder rejects any character other than A-Za-z0-9+/.
- URL - Output is mapped to set of characters lying in A-Za-z0-9+_. Output is URL and filename safe.
- MIME - Output is mapped to MIME friendly format. Output is represented in lines of no more than 76 characters each, and uses a carriage return '\r' followed by a linefeed '\n' as the line separator. No line separator is present to the end of the encoded output.
- 3.11: New to NetRexx. Add MIME, BASIC, & URL options.

delay
4.05

Suspend Stream



- (1) Arguments are NetRexx Pipelines only, not CMS. CMS (and NetRexx when there is no argument) reads delays as the first word of each record.
  When arguments are present, they follow the CMS conventions for the delay time in records. The + indicates a duration, no + means time of day. The objects do NOT have the delay as the first word.
  Clock hours are 24h, so 2pm is 14, and are for the next 24 hours if before "now."
  Seconds can have decimal point and milliseconds.
  In relative times, the number of seconds and minutes are not limited to 60; so 120 seconds is the same as 2 minutes.
- (2) Used only for "relative time." EACH, the default, delays before each object; ONCE delays only the first object.
- Uses Java's Thread.sleep() method and may not be exact in fractional seconds.

devinfo

Write Device Information

- Not implemented in Netrexx Pipelines.

dfsort

Interface to DFSORT/CMS

- Not implemented in Netrexx Pipelines.

diage4

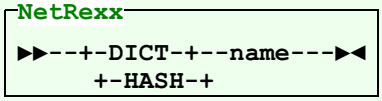
Submit Diagnose E4 Requests

- Not implemented in Netrexx Pipelines.

dict
hash

Read or Write a Dictionary

NetRexx

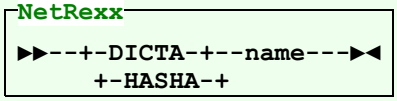


- Pipes for NetRexx only.

dicta
hasha

Write a Dictionary

NetRexx

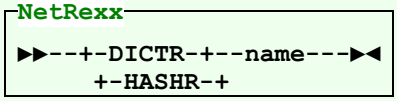


- Pipes for NetRexx only.

dictr
hashr

Read a Dictionary

NetRexx

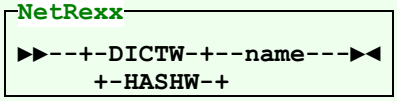


- Pipes for NetRexx only.

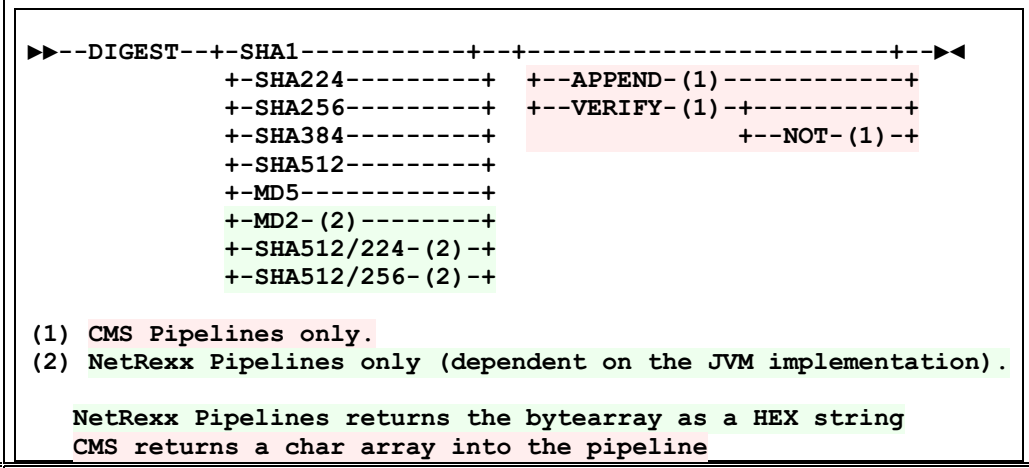
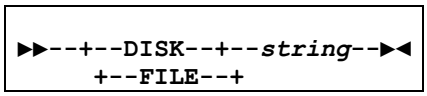
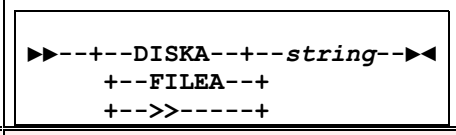
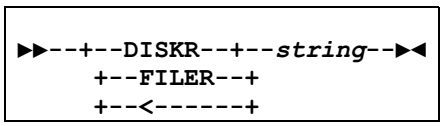
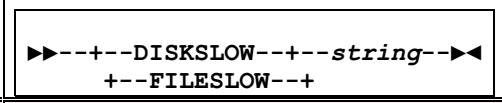
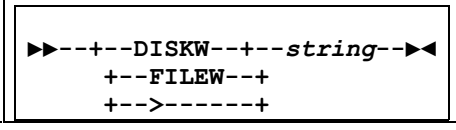
dictw
hashw

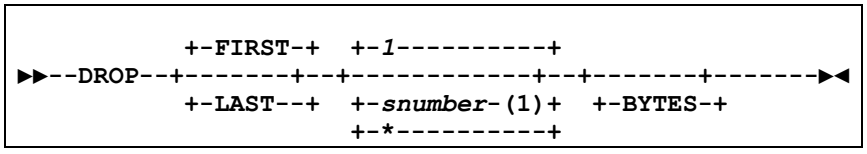
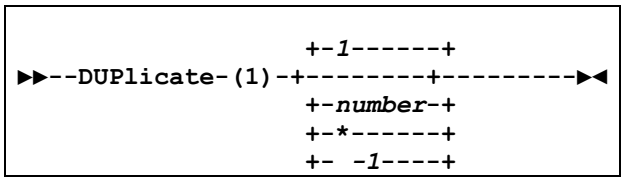
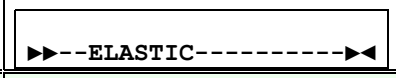
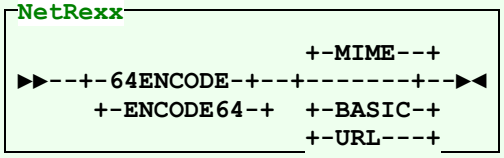
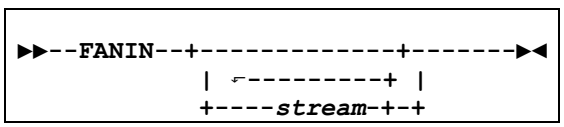
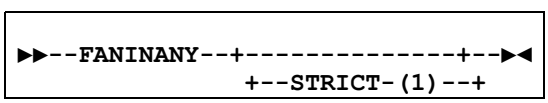
Write a Dictionary

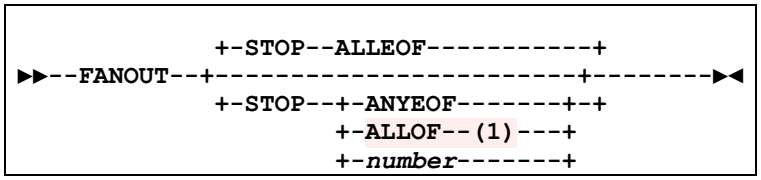
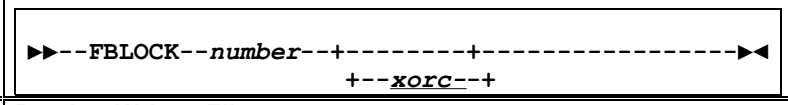
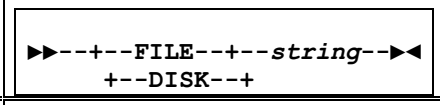
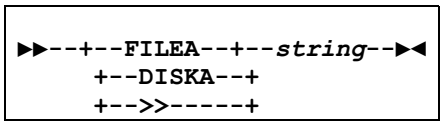
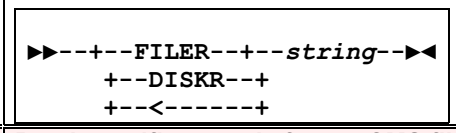
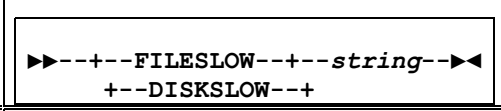
NetRexx

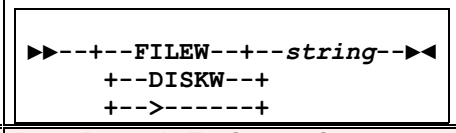
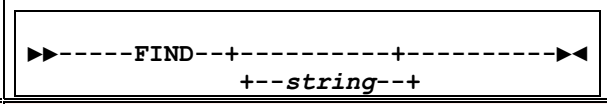
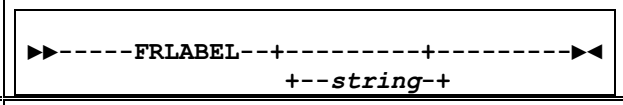
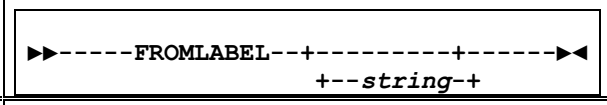
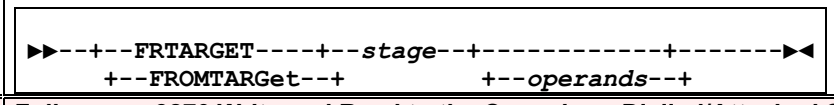
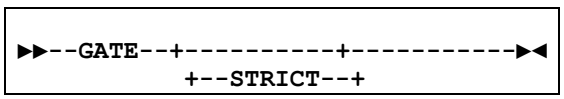


- Pipes for NetRexx only.

digest	**Compute a Message Digest**  **(1) CMS Pipelines only.** **(2) NetRexx Pipelines only (dependent on the JVM implementation).** **NetRexx Pipelines returns the bytearray as a HEX string** **CMS returns a char array into the pipeline**
disk ***file***	**Read a File**  • As in CMS, equivalent to diskr (Pipes for NetRexx Only) or **<**.
diska ***filea*** ***>>***	**Append to or Create a File** 
diskback ***fileback***	**Read a File Backwards** • Not implemented in Netrexx Pipelines.
diskfast ***filefast***	**Read, Create, or Append to a File** • Not implemented in Netrexx Pipelines.
diskid	**Map CMS Reserved Minidisk** • Not implemented in Netrexx Pipelines.
diskr ***filer*** ***<***	**Read a File**  • As in CMS, equivalent to diskr (Pipes for NetRexx Only) or **<**.
diskrandom ***filerandom***	**Random Access a File** • Not implemented in Netrexx Pipelines.
diskslow ***fileslow***	**Read, Create, or Append to a File** 
diskupdate ***fileupdate***	**Replace Records in a File** • Not implemented in Netrexx Pipelines.
diskw ***filew*** ***>***	**Replace or Create a File** 

drop	**Discard Records from the Beginning or the End of the File**  • (1) CMS: must be positive. NetRexx Pipelines: negative reverses FIRST/LAST, so DROP FIRST -3 is the same as DROP LAST 3.
duplicate *duplicat* *duplica* *duplic* *dupli* *dupl* *dup*	**Copy Records**  • (1) CMS is DUPlicat due to 8-character name limitation
elastic	**Buffer Sufficient Records to Prevent Stall** 
encode64 *64encode* *3.11*	**Encode to Base-64 Format** NetRexx  • NOTE: CMS is only 64DECODE, and does not have the options; it does MIME. • BASIC - Output is mapped to a set of characters lying in A-Za-z0-9+/. The encoder does not add any line feed in output, and the decoder rejects any character other than A-Za-z0-9+/. • URL - Output is mapped to set of characters lying in A-Za-z0-9+_. Output is URL and filename safe. • MIME - Output is mapped to MIME friendly format. Output is represented in lines of no more than 76 characters each, and uses a carriage return '\r' followed by a linefeed '\n' as the line separator. No line separator is present to the end of the encoded output. • 3.11: New to NetRexx. Add MIME, BASIC, & URL options.
eofback	**Run an Output Device Driver and Propagate End-of-File Backwards** • Not implemented in Netrexx Pipelines.
escape	**Insert Escape Characters in the Record** • Not implemented in Netrexx Pipelines.
fanin	**Concatenate Streams** 
faninany	**Copy Records from Whichever Input Stream Has One**  • (1) CMS only.
fanintwo	**Pass Records to Primary Output Stream**

fanout	**Copy Records from the Primary Input Stream to All Output Streams**  • (1) CMS only
fanouttwo	**Copy Records from the Primary Input Stream to Both Output Streams**
fbaread	**Read Blocks from a Fixed Block Architecture Drive** • Not implemented in Netrexx Pipelines.
fbawrite	**Write Blocks to a Fixed Block Architecture Drive** • Not implemented in Netrexx Pipelines.
fblock	**Block Data, Spanning Input Records** 
file ***disk***	**Read or Write a File** 
filea ***diska*** ***>>***	**Append to or Create a File** 
fileback ***diskback***	**Read a CMS file backwards** • Not implemented in Netrexx Pipelines.
filedescriptor	**Read or Write an OpenExtensions File that Is Already Open** • Not implemented in Netrexx Pipelines.
filefast ***diskfast***	**Read or write a CMS file** • Not implemented in Netrexx Pipelines.
filer ***file*** ***disk*** ***diskr*** ***<***	**Read a File** 
filerandom ***diskrandom***	**Read specific records from a CMS file**
fileslow ***diskslow***	**Read, Create, or Append to a File** 
filetoken	**Read or Write an SFS File That is Already Open** • Not implemented in Netrexx Pipelines.
fileupdate ***diskupdate***	**Change records in a CMS file**

filew ***diskw*** ***>***	**Replace or Create a File** 
fillup	**Pass Records To Output Streams** • Not implemented in Netrexx Pipelines.
filterpack	**Manage Filter Packages** • Not implemented in Netrexx Pipelines.
find	**Select Lines by XEDIT Find Logic** 
fitting	**Source or Sink for Copipe Data** • Not implemented in Netrexx Pipelines.
fmtfst	**Format a File Status Table (FST) Entry** • Not implemented in Netrexx Pipelines.
frlabel ***fromlabel***	**Select Records from the First One with Leading String** 
fromlabel ***frlabel***	**Select Records from the First One with Leading String** 
frtarget	**Select Records from the First One Selected by Argument Stage** 
fullscreen ***fullscree*** ***fullscre*** ***fullscr***	**Full screen 3270 Write and Read to the Console or Dialled/Attached Screen** • Not implemented in Netrexx Pipelines.
fullscrq	**Write 3270 Device Characteristics** • Not implemented in Netrexx Pipelines.
fullscrq	**Write 3270 Device Characteristics** • Not implemented in Netrexx Pipelines.
fullscrs	**Format 3270 Device Characteristics** • Not implemented in Netrexx Pipelines.
gate	**Pass Records Until Stopped** 
gather	**Copy Records From Input Streams** • Not implemented in Netrexx Pipelines.

gen

Generate a Sequence of Numbers Starting with 1

NetRexx

```
▶▶--GEN--number----▶◀
```

- Not implemented in CMS Pipelines.

getfiles
getfiles
getfile
getfil
getfi
getf
get

Read Files

```
▶▶--GETfiles--------------------▶◀
```

getovers

Write the Contents of Objects

NetRexx

```
▶▶--GETOVERS----▶◀
```

- Input stream 0 should contain rexx objects. The getovers stage will output the index and contents of the stem on stream 0. If output stream 1 is connected, the root is placed there. Any severed streams will cause then stage to exit. Passing a non rexx object will cause the stage to exit with return code 13.
- Pipes for NetRexx only.

getstems

Write the Contents of Members of Stems

NetRexx

```
▶▶--GETSTEMS----▶◀
```

- Input stream 0 should contain rexx objects containing stems. The getstems stage will output the contents of the stem on stream 0. If output stream 1 is connected, the root is placed there. Any severed streams will cause then stage to exit. Passing a non rexx stem object will cause the stage to exit with return code 13.
- Pipes for NetRexx only.

grep
regex
3.09

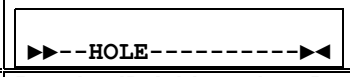
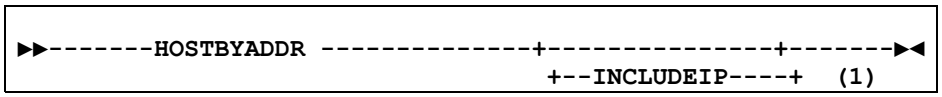
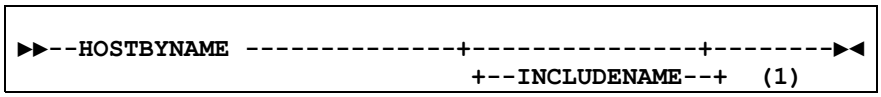
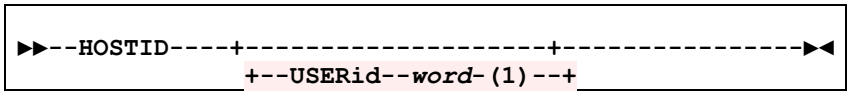
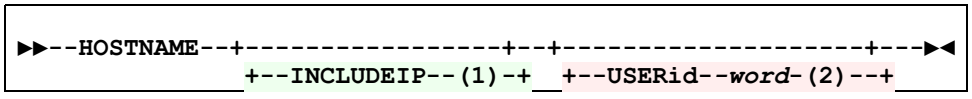
Select Lines by a Regular Expresion

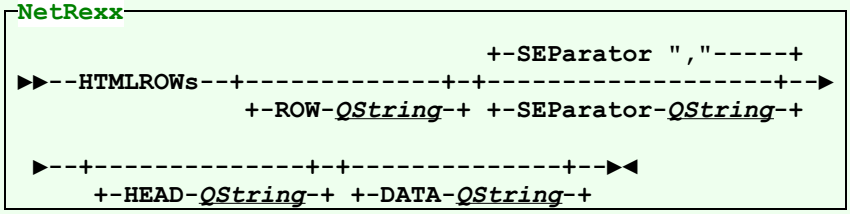
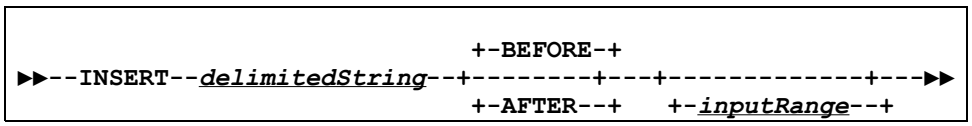
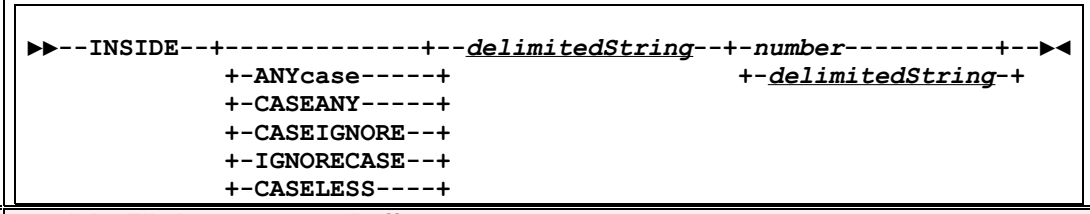
NetRexx

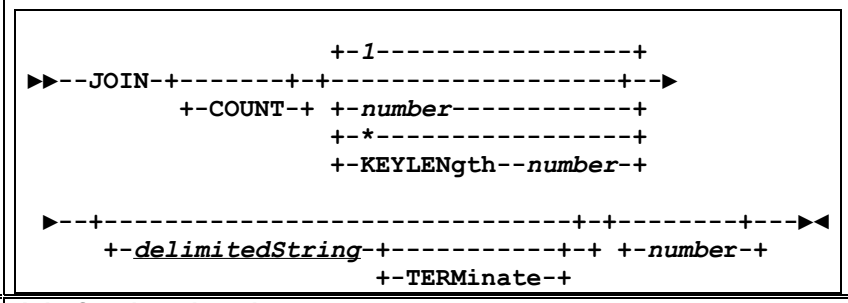
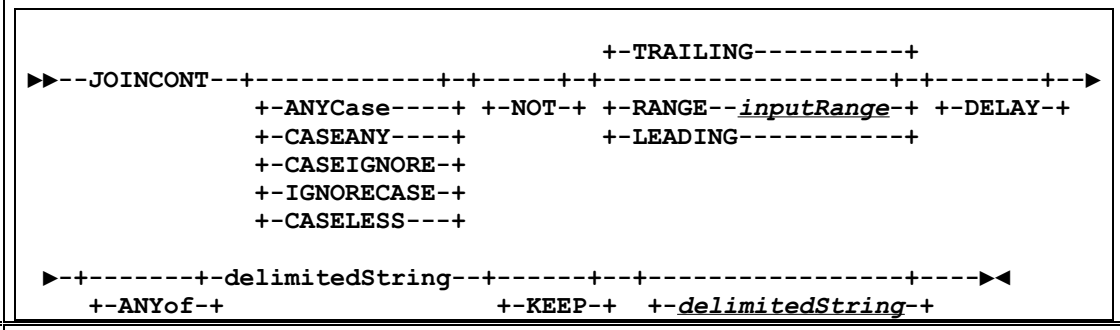
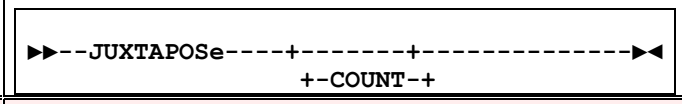
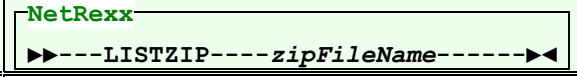
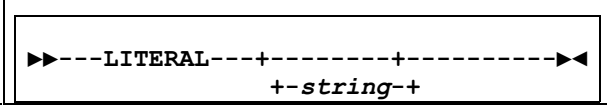
```
▶▶--+--GREP---+--+-------------------------+--regex_Dstring-(1)---▶◀
     +--REGEX--+  +-(--| options_string |--)-+

options_string:
   ↶-----------------------------------+
|--+-+-------------------------------+-+--|
     +-Numbers-----------------------+ (2)
     +-Before-+-1------+-------------+ (3)
     |        +-number-+             |
     +-After-+-1------+--------------+ (3)
     |       +-number-+              |
     +-Context-+-1------+------------+ (4)
     |         +-number-+            |
     +-NOSeparator--------------------+ (5)
     +-Separator-+-/--/------------+--+ (5)
     |           +-delimitedString-+  |
     +-Tertiary-----------------------+ (6)
     +-COUnt--------------------------+ (7)
```

- NetRexx Pipelines only.
- Records matching the RegEx are put out on primary output.
- Records not matching are put out on secondary, if connected, or discarded.
- .
- (1) Regex_string is a Java RegEx expresion. Null string passes all records.
- (2) Records are prefaced with records number, 10 characters, right justified.
- (3) Number of records put out after a matching record.
- (4) Number of records put out before and after a matching record.
- (5) Inserted before a group of "before records" or the found record with "after records."
- (6) Send all matching records (no numbers) to tertiary output stream, if connected.
- (7) Only a count of matches is put out on the primary output stream. (Other options probably should not be used with this.)

hash ***dict***	**Read or Write a Dictionary** NetRexx ▶▶--+-HASH-+--name---▶◀ +-DICT-+ • Pipes for NetRexx only.
hasha ***dicta***	**Write a Dictionary** NetRexx ▶▶--+-HASHA-+--name---▶◀ +-DICTA-+ • Pipes for NetRexx only.
hashr ***dictr***	**Read a Dictionary** NetRexx ▶▶--+-HASHR-+--name---▶◀ +-DICTR-+ • Pipes for NetRexx only.
hashw ***dictw***	**Write a Dictionary** NetRexx ▶▶--+-HASHW-+--name---▶◀ +-DICTW-+ • Pipes for NetRexx only.
help ***ahelp*** ***?***	**Display Help for Pipelines** ▶▶--+--HELP---+--------+---------------------------------+--▶◀ +--AHELP--+ +--*word*--------------------------+ +--?-(3)--+ | +--BUILTINS-(1)---+ | +--MENU-(1)-+-----------------+--+ | +--COMMANDS-(1)---+ | | +--HOST-(1)-------+ | | +--MESSAGES-(1)---+ | | +--OTHER-(1)------+ | | +--SYNTAX-(1)-----+ | +--MSG-(1)--number--------------+ +--number-(1)--------------------+ +--SQL-(1)--string---------------+ +--SQLCODE-(1)--+----------+-----+ +--number--+ (1) CMS Pipelines only. Not yet in NetRexx Pielines. (2) If primary output is connected, lines are propagated, otherwise they are sent to the console by "say." (3) ? is the default pipeEnd character. Here it is useful only when a different pipeEnd is defined.
hfs ***bfs***	**Read or Append File in the Hierarchical File System** • Not implemented in Netrexx Pipelines.
hfsdirectory ***hfsdir*** ***bfsdirectory*** ***bfsdir***	**Read Contents of a Directory in a Hierarchical File System** • Not implemented in Netrexx Pipelines.
hfsquery ***hfsq*** ***bfsquery*** ***bfsq***	**Write Information Obtained from OpenExtensions into the Pipeline** • Not implemented in Netrexx Pipelines.
hfsreplace ***hfsrep*** ***bfsreplace*** ***bfsrep***	**Replace the Contents of a File in the Hierarchical File System** • Not implemented in Netrexx Pipelines.

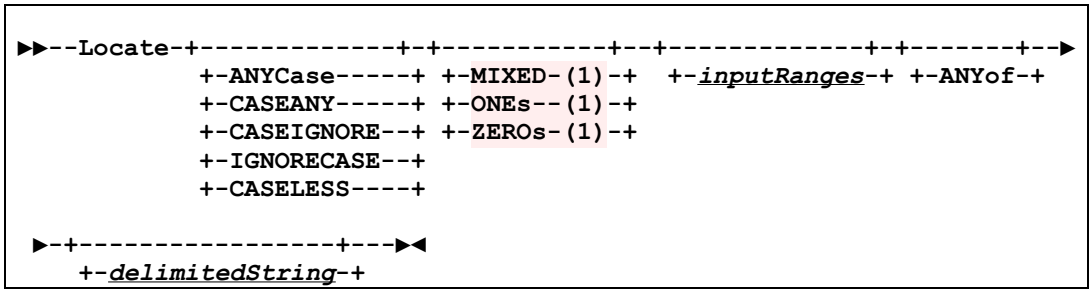
hfsstate ***hfsstat*** ***bfsstate*** ***bfsstat***	**Obtain Information about Files in the Hierarchical File System** • Not implemented in Netrexx Pipelines.
hfsxecute ***hfsx*** ***bfsxecute*** ***bfsx***	**Issue OpenExtensions Requests** • Not implemented in Netrexx Pipelines.
hlasm	**Interface to High Level Assembler** • Not implemented in Netrexx Pipelines.
hlasmerr	**Extract Assembler Error Messages from the SYSADATA File** • Not implemented in Netrexx Pipelines.
hole	**Destroy Data** 
hostbyaddr ***3.09***	**Resolve IP Address into Domain and Host Name**  • (1) Optional parameter not present in VM/CMS version • INCLUDEIP - Also include the IP address along with the hostname. Output: <hostname>/<ip address> Example: **dns.google/8.8.8.8** • Known issues: The underlying Java method getByName/getHostName does not appear to handle IPv6 addresses in any known and consistent manner. Could be related to a host configuration issue but googling shows odd and inconsistent results for getting around this.
hostbyname ***3.09***	**Resolve a Domain Name into an IP Address**  • (1) Optional parameter not present in CMS Pipelines • Arguments: INCLUDENAME - Also include the name of the host on output. • Output: <hostname>/<ip address> Example: **dns.google/8.8.8.8**
hostid ***3.09***	**Write TCP/IP Default IP Address**  • (1) The USERid option available under CMS Pipelines is not applicable and is ignored in NetRexx Pipelines
hostname ***3.09***	**Write TCP/IP Host Name**  • (1) Optional parameter not present in VM/CMS version • (2) The USERid option available under CMS is not applicable and is ignored in NetRexx Pipelines • Arguments: INCLUDEIP - include the IP address of the system in the response in the form <hostname>/<ip address>

htmlrows ***htmlrow*** ***3.11***	**Convert rows to HTML format** NetRexx  • HTMLROWs reads rows from its primary input stream and writes them to its primary output stream, altering them to have the proper HTML tags for TABLE ROWS. • I.e., it converts abc,mnop,xyz into <tr><td>abc</td><td>mnop</td><td>xyz</td></tr> • The SEPARATOR QStirng, by default the comma character, can be specified. • There are options to put additional data inside the tags. This could be used for class or style tag options, for example. ◦ ROW QString : puts its information into the <tr>-tags ◦ DATA QString : puts its information into the <td>-tags ◦ HEAD QString : puts its information into the <th>-tags (1) • QString is a quoted string of characters. The quote marks may be either single or double, but must match. If there are no spaces in the string, the quote marks are optional. • (1) If there is a HEAD option, the first row read has <th>-tags instead of <td>-tags. It must have a QString of at least "". Succeeding rows have the standard <td>-tags.
httpsplit	**Split HTTP Data Stream** • Not implemented in Netrexx Pipelines.
iebcopy	**Process IEBCOPY Data Format** • Not implemented in Netrexx Pipelines.
if	**Process Records Conditionally** • Not implemented in Netrexx Pipelines.
immcmd	**Write the Argument String from Immediate Commands** • Not implemented in Netrexx Pipelines.
insert	**Insert String in Records**  • insert a string into a record before or after the record content. Will be much more efficient than specs especially if the input is a Byte[]
inside	**Select Records between Labels** 
instore	**Load the File into a storage Buffer** • Not implemented in Netrexx Pipelines.
ip2socka	**Build sockaddr_in Structure** • Not implemented in Netrexx Pipelines.

ispf	**Access ISPF Tables** • Not implemented in Netrexx Pipelines.
jeremy	**Write Pipeline Status to the Pipeline** • Not implemented in Netrexx Pipelines.
join	**Combine Records** 
joincont	**Join Continuation Lines** 
juxtapose	**Preface Record with Marker** 
ldrtbls	**Resolve a Name from the CMS Loader Tables** • Not implemented in Netrexx Pipelines.
listcat	**Obtain Data Set Names** • Not implemented in Netrexx Pipelines.
listdsi	**Obtain Information about Data Sets** • Not implemented in Netrexx Pipelines.
listispf	**Read Directory of a Partitioned Data Set into the Pipeline** • Not implemented in Netrexx Pipelines.
listpds	**Read Directory of a Partitioned Data Set into the Pipeline** • Not implemented in Netrexx Pipelines.
listzip	**List the Files in a Zipped File** 
literal	**Write the Argument String** 

locate
locat
loca
loc
lo
l

Select Lines that Contain a String



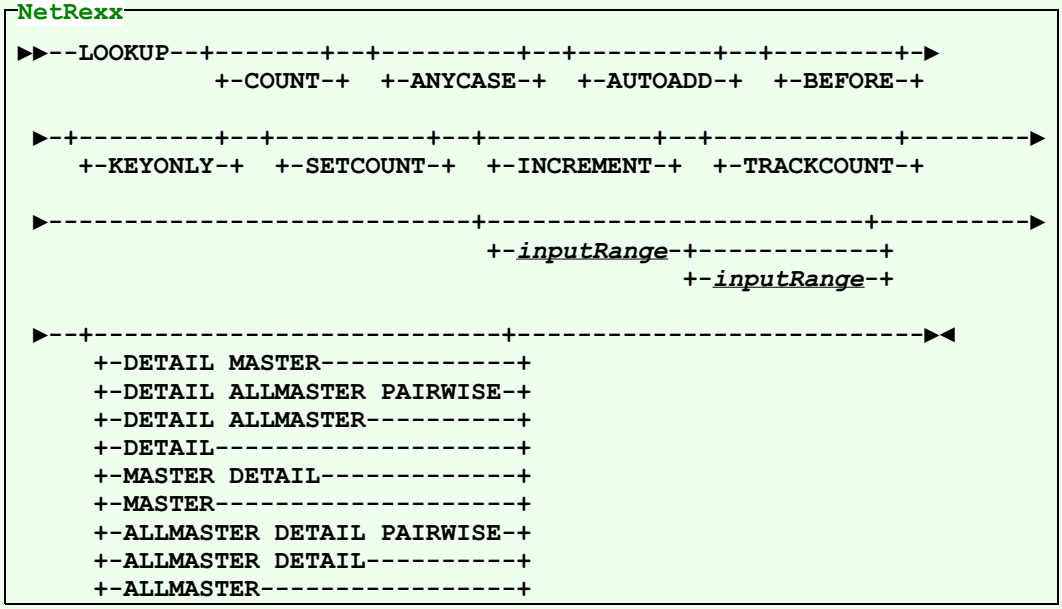
(1) Not in NetRexx Pipelines, yet.

- [2] IBM documentation has this as "LOCATE" rather than "Locate". But the abbreviations work in both systems.

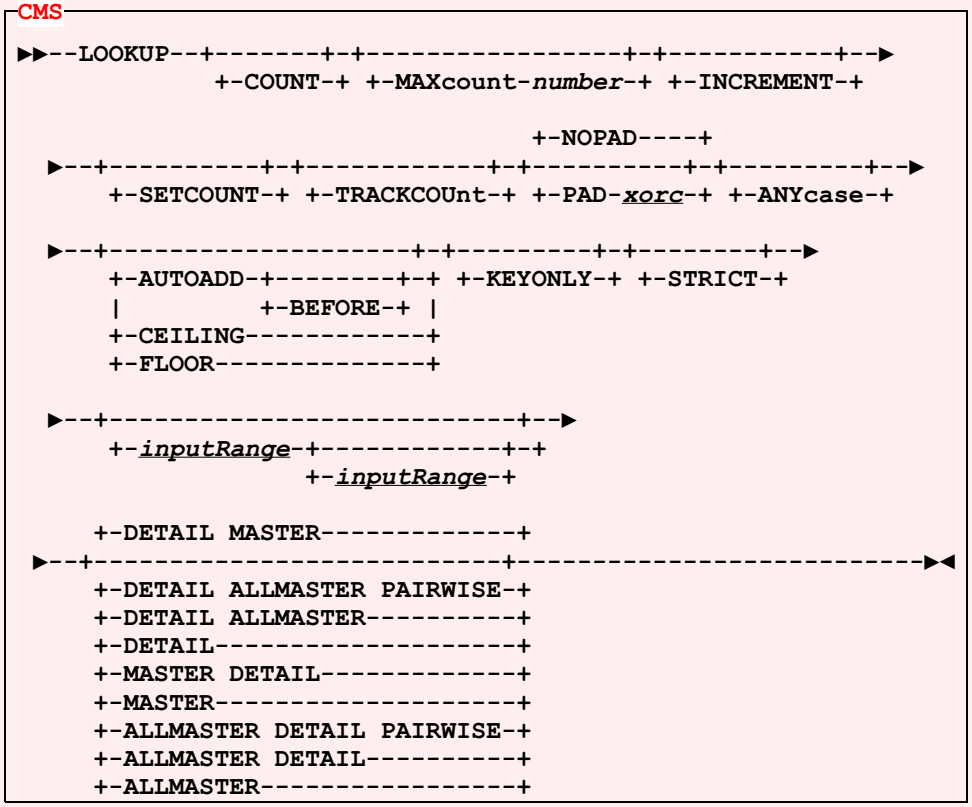
lookup

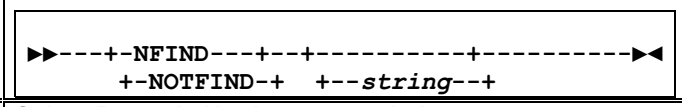
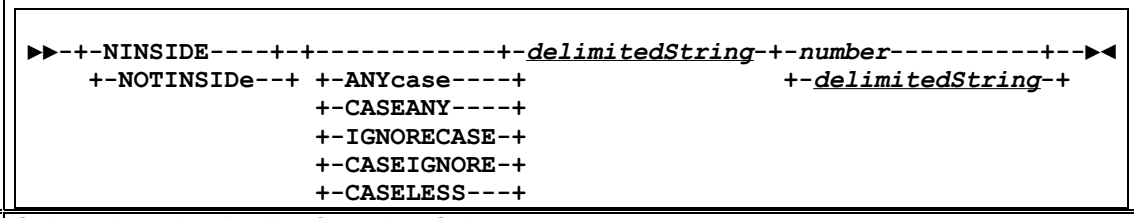
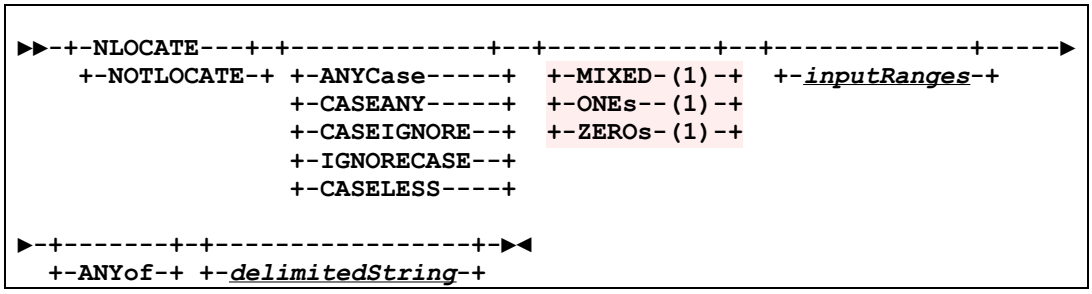
Find Records in a Reference Using a Key Field

NetRexx



- in stream 0 are detail records
- in stream 1 are master records
- in stream 2 adds to masters
- in stream 3 delete from masters
-
- out stream 0 are matched records
- out stream 1 are unmatched detail records
- out stream 2 are unmatched or counted master records
- out stream 3 deleted masters
- out stream 4 duplicate masters
- out stream 5 unmatched master deletes
-
- lookup does not consider an unconnected output stream an error. It does proprogate EOFs from output streams.

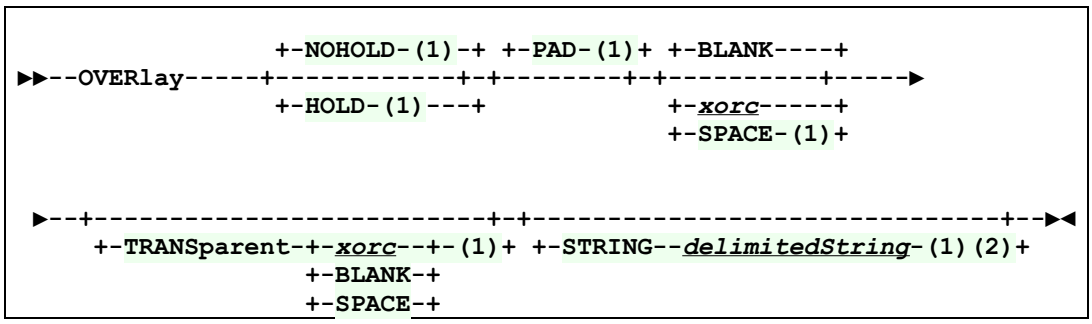
lookup	**Find Records in a Reference Using a Key Field** CMS  • in stream 0 are detail records • in stream 1 are master records • in stream 2 adds to masters • in stream 3 delete from masters • • out stream 0 are matched records • out stream 1 are unmatched detail records • out stream 2 are unmatched or counted master records • out stream 3 deleted masters • out stream 4 duplicate masters • out stream 5 unmatched master deletes • • lookup does not consider an unconnected output stream an error. It does proprogate EOFs from output streams.
maclib	**Generate a Macro Library from Stacked Members in a COPY File** • Not implemented in Netrexx Pipelines.
mapmdisk	**Map Minidisks Into Data spaces** • Not implemented in Netrexx Pipelines.
mctoasa	**Convert CCW Operation Codes to ASA Carriage Control** • Not implemented in Netrexx Pipelines.

mdiskblk	**Read or Write Minidisk Blocks** • Not implemented in Netrexx Pipelines.
mdskrandom ***mdskrand***	**Random Access a CMS File on a Mode** • Not implemented in Netrexx Pipelines.
mdskslow	**Read, Append to, or Create a CMS File on a Mode** • Not implemented in Netrexx Pipelines.
mdskupdate ***mdskupda***	**Replace Records in a File on a Mode** • Not implemented in Netrexx Pipelines.
members ***member***	**Extract Members from a Partitioned Data Set** • Not implemented in Netrexx Pipelines.
merge	**Merge Streams** • Not implemented in Netrexx Pipelines.
mqsc	**Issue Commands to a WebSphere MQ Queue Manager** • Not implemented in Netrexx Pipelines.
nfind ***notfind***	**Select Lines by XEDIT NFind Logic** 
ninside ***notinsid*** ***notinside*** ***3.09***	**Select Records Not between Labels** 
nlocate ***notlocate***	**Select Lines that Do Not Contain a String**  • (1) Not in NetRexx Pipelines, yet.
noEofBack	**Pass Records and Ignore End-of-file on Output**

nop	**No Operation** NetRexx ▶▶--NOP---------▶◀ • Pipes for NetRexx only.
not	**Run Stage with Output Streams Inverted** ▶▶--NOT--*stage*--+------------+-------▶◀ +--*operands*--+
notfind *nfind*	**Select Lines by XEDIT NFind Logic** ▶▶---+-NOTFIND-+--+----------+---------▶◀ +-NFIND---+ +--*string*--+
notinside *notinsid* *ninside*	**Select Records Not between Labels** ▶▶-+-NOTINSIDe--+-+------------+-*delimitedString*-+-*number*----------+--▶◀ +-NINSIDE----+ +-ANYcase----+ +-*delimitedString*-+ +-CASEANY----+ +-IGNORECASE-+ +-CASEIGNORE-+ +-CASELESS---+
notlocate *nlocate*	**Select Lines that Do Not Contain a String** ▶▶-+-NOTLOCATE-+-+-------------+--+-----------+--+-------------+-----▶ +-NLOCATE---+ +-ANYCase-----+ +-MIXED-(1)-+ +-*inputRanges*-+ +-CASEANY-----+ +-ONEs--(1)-+ +-CASEIGNORE--+ +-ZEROs-(1)-+ +-IGNORECASE--+ +-CASELESS----+ ▶-+-------+-+-----------------+-▶◀ +-ANYof-+ +-*delimitedString*-+ • (1) Not in NetRexx Pipelines, yet.
nucext	**Call a Nucleus Extension** • Not implemented in Netrexx Pipelines.
optcdj	**Generate Table Reference Character (TRC)** • Not implemented in Netrexx Pipelines.
outside	**Select Records Not between Labels** ▶▶--OUTSIDE-+------------+-*delimitedString*-+-*number*----------+--▶◀ +-ANYcase----+ +-*delimitedString*-+ +-CASEANY----+ +-CASEIGNORE-+ +-IGNORECASE-+ +-CASELESS---+
outstore	**Unload a File from a storage Buffer** • Not implemented in Netrexx Pipelines.
over	**Write the Values of Stems** • Obsolete. Now use varover. over is now an alias for overlay..

overlay
overla
overl
over

Overlay Data from Input Streams



- HOLD keeps the last record from each stream, except primary, and uses it if the stream ends.
- TRANSPARENT means that character can be different from the PAD character.
  If omitted, it is the same as PAD character.
- dstream can be used instead of a non-primary stream.
- (1) NetRexx Pipelines only
- (2) same as highest (+1) stream; implies HOLD

overstr

Process Overstruck Lines

- Not implemented in Netrexx Pipelines.

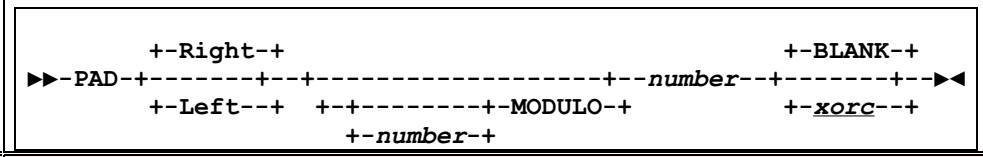
pack

Pack Records as Done by XEDIT and COPYFILE

- Not implemented in Netrexx Pipelines.

pad

Expand Short Records



parcel

Parcel Input Stream Into Records

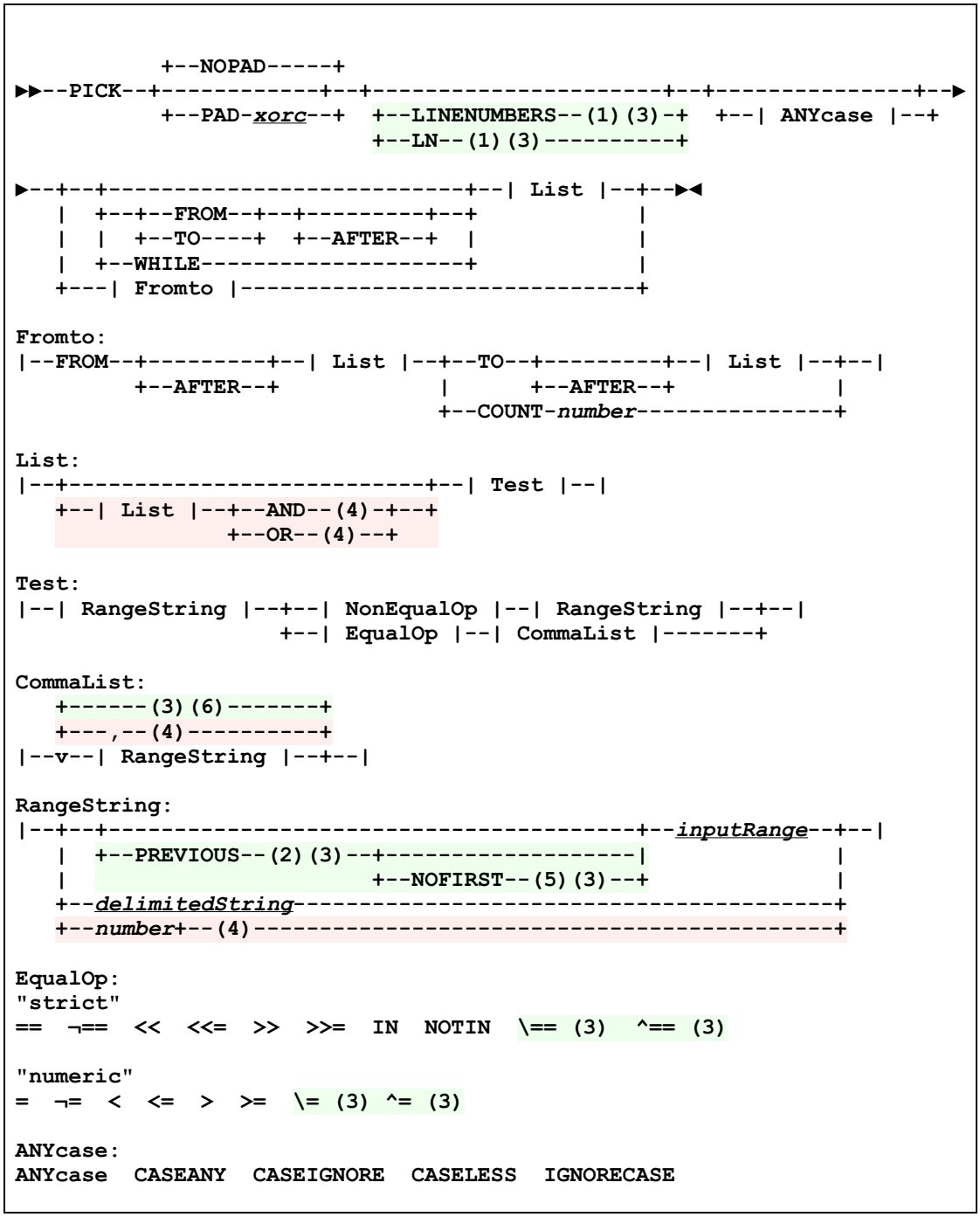
- Not implemented in Netrexx Pipelines.

parse
4.05
	Rearrange Contents of Records
	NetRexx
	►►--PARSE--*parse_template_Dstring*--►
	►--+---+---►
	+--+--NETREXX--+--*NetRexx_statement_Dstring*--+
	+--NR-------+
	►--+-------------------------+--►
	+*output_template_Dstring*--+
	►--+---------------------------------------+----►◄
	+--FINALLY--*NetRexx_statement_Dstring*--+
	• Records are parsed via the parse_template_delimited_string.
	• Variables are named _n, where n is 1 to 9.
	• The values of the variables are put into the output_template_delimited_string replacing _n.
	• For a literal _n that won't be changed, use __n.
	• The two NetRexx_statement_Dstrings are single statements, or multiple statements separated by ";"s.
	◦ The _n variables can be used and changed.
	◦ The string **\n** will split the string into separate ouput records.
	◦ The special indexed REXX variable COUNTER[] is also available in these Dstrings. This is specific to a PARSE stage, but persists between records. All the indexed values are initiated to 0. Both indexes and values can be strings.
	◦ This is powerful and has the possibility of doing damage to your pipe.
	You have been warned!
	• If the NR NetRexx_statement_Dstring returns a value, it is used as the output instead of the optional output_template_Dstring.
	• The FINALLY's statement_Dstring is executed after the last input record has been processed. The value returned is put out as an "extra" output record.
	• (As of 4.05) Variable names of "$n" are depreciated, and can not be used with NETREXX or FINALLY options.
	• NetRexx Pipelines only.
	• .
	• Examples:
	◦ `parse / 2 _1 +1/ /The second letter is "_1". __1 won't be changed./`
	◦ `parse /2 _1 +1/ NR /counter[1]=counter[1]+1; _9=counter[1]/ /_9/ FINALLY /return "Count:" _9/`
	◦ `PARSE /_1 2 _2 +1 _3/ ,`
	`NR /if _2.datatype('L') then counter['c'] = counter['c'] + 1; _2 = _2.upper/ ,`
	`/_1_2_3/ ,`
	`FINALLY /return counter['c'] 'Changed to upper'/`

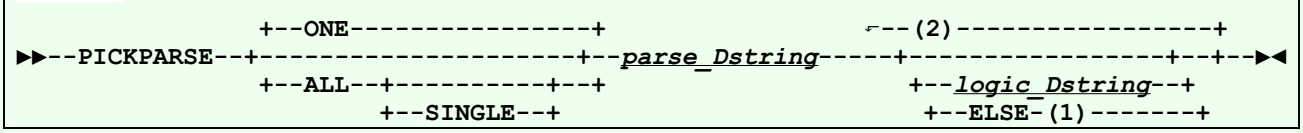
pause	**Signal a Pause Event** • Not implemented in Netrexx Pipelines.
pdsdirect ***pds***	**Write Directory Information from a CMS Simulated Partitioned Data Set** • Not implemented in Netrexx Pipelines.

pick
4.05

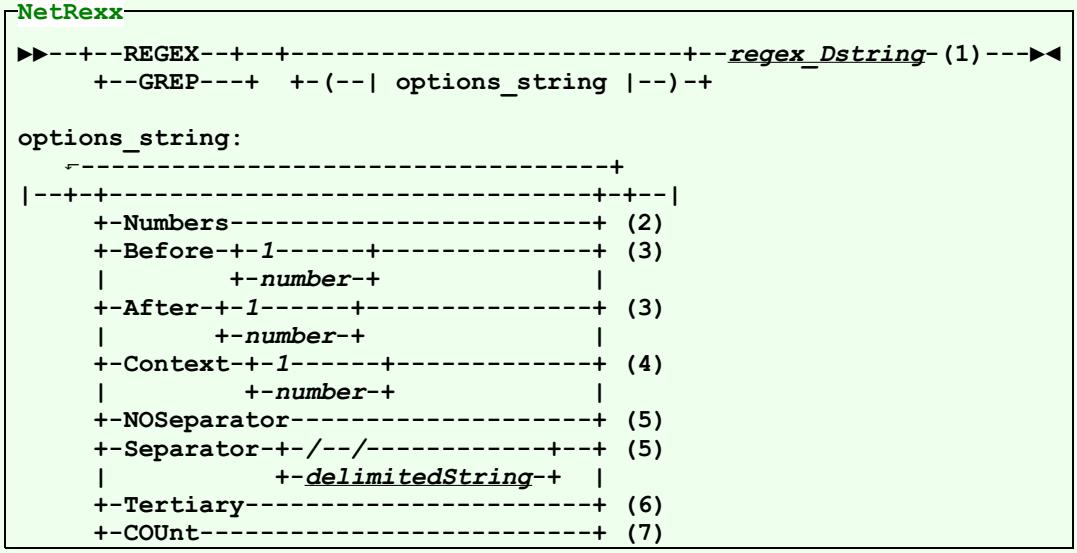
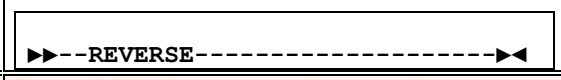
Select Lines that Satisfy a Relation



- (1) NetRexx only. Inserts the original record number followed by a SPACE at the beginning of each output record.
- (2) NetRexx only. Uses the data from the previous record. Before the first record, this is Rexx "".
- (3) NetRexx Pipelines only. Not yet in CMS Pipelines.
- (4) CMS Pipelines only. Not yet in NetRexx Pipelines.
- (5) NetRexx Only. Uses first record data for first record instead of previous "".
- (6) CMS uses ",", NetRexx does not. CMS limits to right side, NetRexx allows on the left, too; also with any comparison op. NetRexx concats the several ranges.

pickparse 3.09	**Select Lines that Satisfy Relations using Rexx Parse** **NetRexx**  • Records are parsed via the parse_delimited_string. • Variables are named $n, where n is 1 to 9. • The values of the variables are put into the logic_delimited_string replacing $n and evaluated. If TRUE, the record is put out on the stream numbered by the dstring's position. • The stream for a Dstring of ELSE is used if no previous logic Dstring is TRUE. • If there is no specific ELSE, there is an implied one at the end; if that stream is not connected, the record is discarded. • If ONE then the record is put out on, at most, one stream: the first one matched. • If ALL then the record is put out on all streams matched. • If SINGLE then the records are all put out on the primary output stream. • The parse_delimited_string and logic_delimited_string(s) follow normal NetRexx rules. • (1) Implied ELSE after last specified dstring. • (2) Up to 10 logic_Dstrings may be specified to go to up to 11 ouput streams (including an implied ELSE). • Not implemented in CMS Pipelines. Pickparse permits selecting records by a NetRexx logical expression, using parts of the record selected by a Rexx PARSE template. A simple example has two delimited strings, a Rexx template and a logical expression: `pickparse / . . $3 . 50 $5 +5 / / $3 < $5 /` The parse template selects the 3rd word, and the 5 characters starting in column 50. the variable names are a dollar sign and a digit. Then those variables can be used in the logic expression. When run, and records matching the logic expression are written to the primary output stream, others to the secondary. If either stream is not connected, the corresponding records are discarded. There can be multiple logic expressions, each in its own delimited string. Parenthetical expressions may be used. Records are matched to each in turn. Any records matching are written to that output stream, if connected. With the option ONE, the default, each record is written to one output stream: the first one it matches. With the option ALL, the matching goes on and a record could be written to multiple output streams. There is an implicit or explicit ELSE as the last logic expression. Records that have not matched any of the previous expressions match this and are written or discarded depending on if the stream is connected or not. The parse template can define up to 9 separate zones, $1 to $9. The variables $_n are also available for the logic expressions; they are the values from the previous record. Initially these are "". There can be up to 10 output streams defined, and up to 9 logic expressions plus ELSE.
pipcmd	**Issue Pipeline Commands** • Not implemented in Netrexx Pipelines.
pipestop	**Terminate Stages Waiting for an External Event** • Not implemented in Netrexx Pipelines.
polish	**Reverse Polish Expression Parser** • Not implemented in Netrexx Pipelines.
predselect predsel	**Control Destructive Test of Records** • Not implemented in Netrexx Pipelines.
preface	**Put Output from a Device Driver before Data on the Primary Input Stream** • Not implemented in Netrexx Pipelines.

prefix	**Stop and Run a Stage First, Before Continuing** NetRexx ▶▶--PREFIX---*string*-------------▶◀ • Blocks its primary input and excutes stage supplied as an argument. The output from this stage are put to the primary output stream. When its compete the primary input is shorted. • Not implemented in CMS Pipelines.
printmc	**Print Lines** • Not implemented in Netrexx Pipelines.
punch	**Punch Cards** • Not implemented in Netrexx Pipelines.
qpdecode	**Decode to Quoted-printable Format** • Not implemented in Netrexx Pipelines.
qpencode	**Encode to Quoted-printable Format** • Not implemented in Netrexx Pipelines.
qsam	**Read or Write Physical Sequential Data Set through a DCB** • Not implemented in Netrexx Pipelines.
qsort	**Quick Order Records on Whole Length** NetRexx ▶▶--QSORT------▶◀ • This sort routine is very basic. It uses sortRexx class, which implements the sortClass interface. To sort objects of classes other than Rexx requires that you implement another sortClass with a name begining with 'sort'. • Not implemented in CMS Pipelines.
query	**Obtain Information From Pipelines** +-VERSION------+ ▶▶--Query--+--------------+----------▶◀ +-LEVEL--------+ +-SOURCE-(1)---+ +-MSGLEVEL-(2)-+ +-MSGLIST-(2)--+ • (1) Not CMS • (2) Not NetRexx Pipelines
random ***3.09***	**Generate Pseudorandom Numbers** ▶▶--RANDOM--+-----------------------------------+----▶◀ | +-*----------+ +-*------------+ | +-+-*max_number*-+--+--------------+-+ +-*seed_snumber*-+ • NetRexx Pipelines will be a different sequence than CMS gives with the same seed.
reader	**Read from a Virtual Card Reader** • Not implemented in Netrexx Pipelines.
readpds	**Read Members from a Partitioned Data Set** • Not implemented in Netrexx Pipelines.

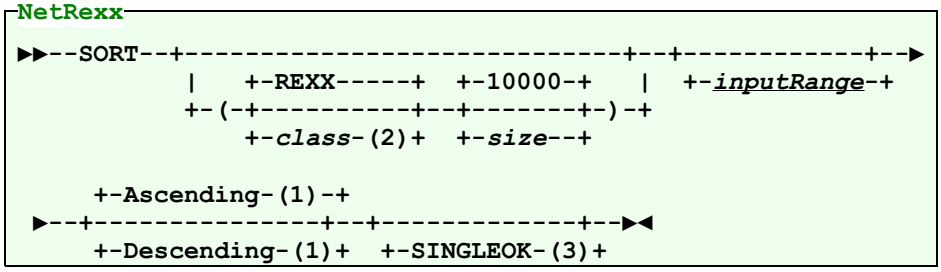
regex grep 3.09	**Select Lines by a Regular Expresion** NetRexx  • NetRexx Pipelines only. • Records matching the RegEx are put out on primary output. • Records not matching are put out on secondary, if connected, or discarded. • . • (1) Regex_string is a Java RegEx expresion. Null string passes all records. • (2) Records are prefaced with records number, 10 characters, right justified. • (3) Number of records put out after a matching record. • (4) Number of records put out before and after a matching record. • (5) Inserted before a group of "before records" or the found record with "after records." • (6) Send all matching records (no numbers) to tertiary output stream, if connected. • (7) Only a count of matches is put out on the primary output stream. (Other options probably should not be used with this.)
retab	**Replace Runs of Blanks with Tabulate Characters** • Not implemented in Netrexx Pipelines.
reverse	**Reverse Contents of Records** 
rexx	**Run a REXX Program to Process Data** • Not implemented in Netrexx Pipelines.
rexxvars	**Retrieve Variables from a REXX or CLIST Variable Pool** • Not implemented in Netrexx Pipelines.
runpipe	**Issue Pipelines, Intercepting Messages** • Not implemented in Netrexx Pipelines.
scm	**Align REXX Comments** • Not implemented in Netrexx Pipelines.
sec2greg	**Convert Seconds Since Epoch to Gregorian Timestamp** • Not implemented in Netrexx Pipelines.

serialize	**Convert Objects to/from a Single Text String** NetRexx ▶▶--SERIALIZE--+----------------+----▶◀ +--classname-(1)-+ • (1) classname if class is specified deserialize input to objects of this type - otherwise serialize input objects. • (2) Pipes for NetRexx only. • (3) For some reason readObject does not like more than one object network in its stream. Block multiple objects. • (4) See examples/serialize_tests01.njp
sfsback	**Read an SFS File Backwards** • Not implemented in Netrexx Pipelines.
sfsdirectory	**List Files in an SFS Directory** • Not implemented in Netrexx Pipelines.
sfsrandom	**Random Access an SFS File** • Not implemented in Netrexx Pipelines.
sfsupdate	**Replace Records in an SFS File** • Not implemented in Netrexx Pipelines.
snake ***3.09***	**Build Multicolumn Page Layout** ▶▶--SNAKE--*number_cols*--+---+--▶◀ +--*number_rows*--+----------------------------+--+ +--*page_seperator_DString*-(1)-+ • (1) NetRexx Pipelines only. Appears first, last, and between pages. Avoid \ as escape terms maybe added in the future. \n for newline is OK. Your system may require \\n .
socka2ip	**Format sockaddr_in Structure** • Not implemented in Netrexx Pipelines.

sort

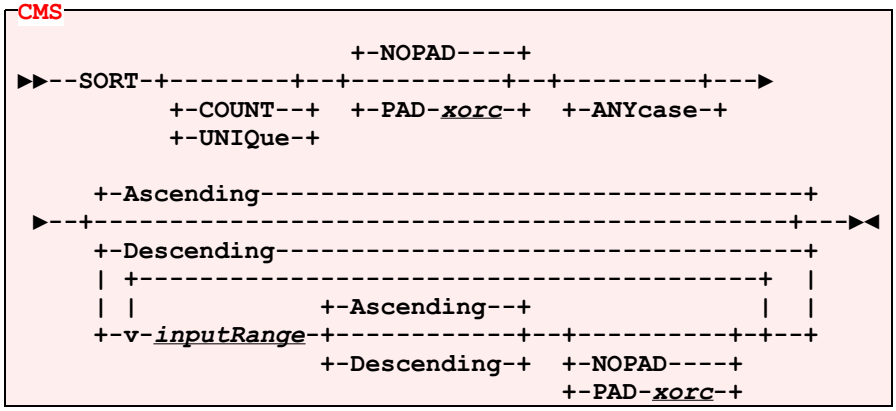
Order Records

NetRexx



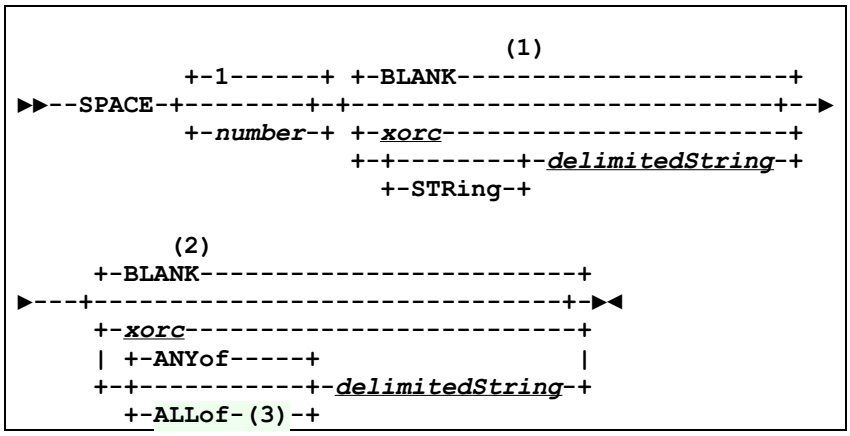
- (1) May come before inputRange, for backwards compatability.
- (2) Requires that you implement another sortClass with a name begining with 'sort'
- (3) Suppresses error message if only one record to sort for Rexx objects.
- Uses *sortClass* class as Interface Class for Generic Sort Objects and *sortRexx* class to Sort Rexx Text Objects

CMS



space
3.09

Space Words Like REXX

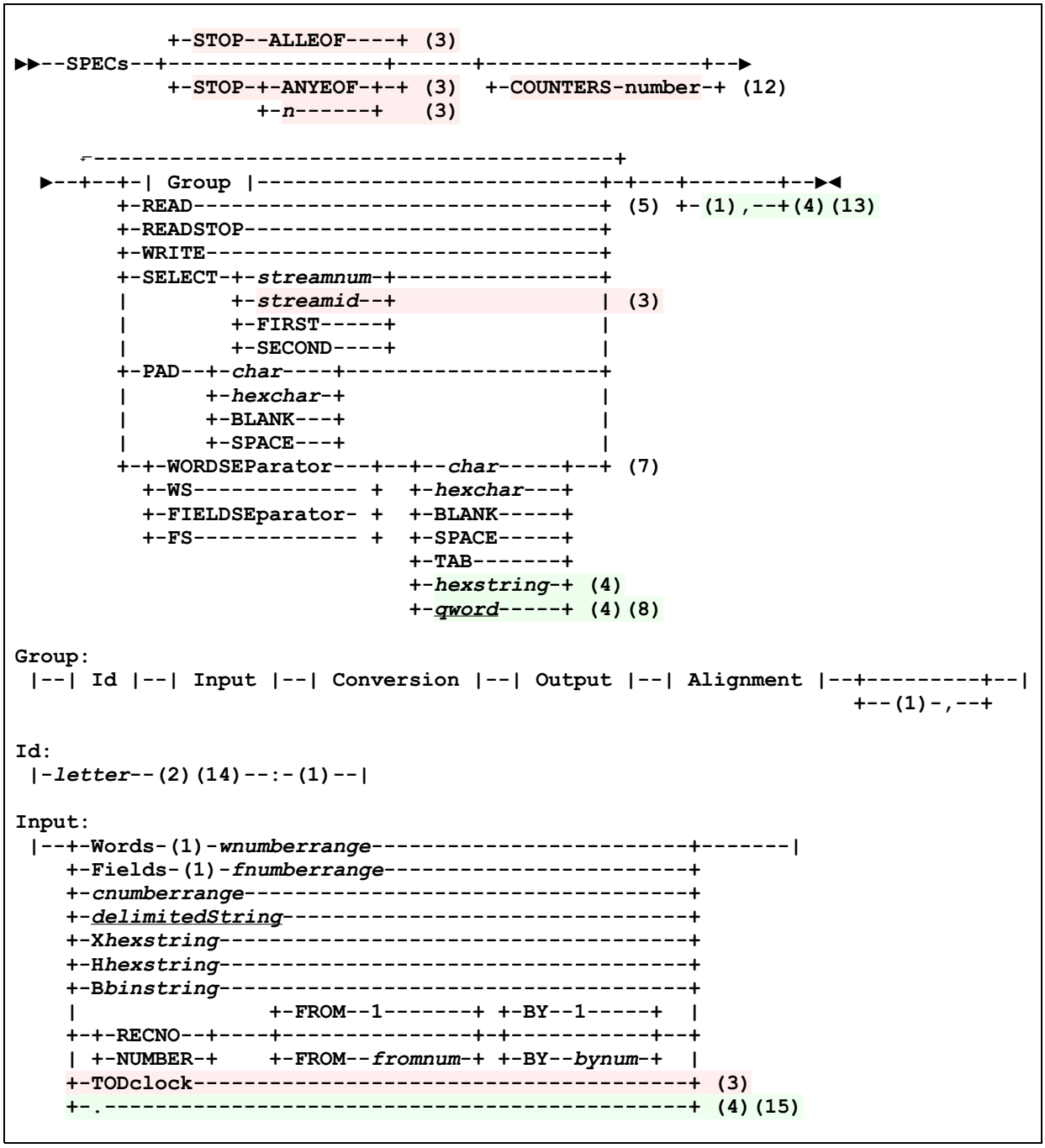


- (0) The order is the reverse of CHANGE!
- (1) the replacement char/string
- (2) the char/chars that will be stripped and replaced
- (3) NetRexx Pipelines only, not CMS. The dstring is treated as a single unit for stripping or replacing

spec
specs
4.05
Fields and

Rearrange Contents of Records

Separators NetRexx and COUNTERS comma separators



```
Conversion:
 |-+-------+--+------------------------------+------|
   +-STRIP-+  +-B2C--------------------------+
              +-B2D--------------------------+ (4)
              +-B2X--------------------------+ (4)
              +-C2B--------------------------+
              +-C2D--------------------------+
              +-C2F--------------------------+ (3)
              +-C2I--------------------------+ (3)
              +-C2P-+------------+-----------+ (3)
              |     +-(2)(scale)-+           | (3)
              +-C2V--------------------------+ (3)
              +-C2X--------------------------+
              +-D2C--------------------------+
              +-D2X--------------------------+ (4)
              +-F2C--------------------------+ (3)
              +-I2C--------------------------+ (3)
              +-P2C-+------------+-----------+ (3)
              |     +-(2)(scale)-+           | (3)
              +-V2C--------------------------+ (3)
              +-X2B--------------------------+ (4)
              +-X2C--------------------------+
              +-X2D--------------------------+ (4)
              +-F2T--------------------------+ (3)
              +-LOWER------------------------+ (4)
              +-UPPER------------------------+ (4)
              +-STRING-----------------------+ (4)(11)
              +-+-NETREXX-+-delimitedString--+ (4)(10)
                +-NR------+                   (4)

Output:
 |--+-Next-+-------+-------+--|
    |      +-(2).n-+       |
    +-NEXTWord-+-+-------+-+
    +-NWord----+ +-(2).n-+ |
    +-columnrange----------+
    +-.--------------------+

Alignment:
    +-Left---+
 |--+--------+--|
    +-Center-+
    +-Centre-+
    +-Right--+

Ranges (cnumberrange, fnumberrange, wnumberrange):
 |--+-snumber-+--(2)--+------------------------+--|
    +---*-----+       +--.-----(2)----number----+
                      +- - -+--(2)--+-snumber-+-+
                      +--;--+       +---*-----+
```

- (1) Blanks are optional in this position.
- (2) Blanks are not allowed here.
- (3) CMS only. Not yet implemented in NetRexx Pipelines
- (4) NetRexx Pipelines only. Not yet implemented in CMS
- (5) NetRexx Pipelines only. READ is giving the same output as READSTOP when the streams are different length.
- (6) This senses if it is the first stage, but comment stages will fool it into not producing any output.
- (7) CMS Pipelines, without documenting it, places this right by default NetRexx Pipelines follows the documentation and places this left by default. Specify the alignment you want to override these defaults.
- (8) A *qword* is an optionally quoted word, with single or double marks. If it contains spaces or begins with a quote mark, it must be quoted. It can not start with a space (the quote mark will be considered a single character, and rest gibberish). If is unquoted and an even number of hexadecimal characters, it will be used as a hexchar or hexstring.
- (9) CMS has a mini-programming language built in. It uses Field Identifiers and Control Breaks, Counters, and Structured Data.
  NetRexx does not yet have any of these features.
- (10) The delimited string is any valid NetRexx code. [**Yes, you can get in trouble!**] It is put into a method and executed for each record. The selected input data is in the variable DATA. The returned string is output. The variable array COUNTER[] is available for your use. Unlike CMS, COUNTER is a full NetRexx variable object of type Rexx. Each COUNTER is initially 0, but can hold any NetRexx value, including strings. COUNTERs are persistent for the life of the stage and are shared across all NETREXX converters in a stage. Index can be

number or string. Fields identified by a fieldid are accessed as field["id"] when the quotes are required, the id is a single letter, case is respected.

- (11) The data is processed by the toString() method.
- (12) CMS Only. NetRexx ignores COUNTERS n; it has an unlimited number. See the NETREXX converter(10).
- (13) NetRexx Only. A comma may be used to separate groups for readability. It is ignored by the system.
- (14) Letter is a single character, a-zA-Z. Case is respected. BLANK is not permitted between letter and : . The resulting data is available in the NETREXX code as field["letter"].
- (15) NetRexx Only. A "." for input is "0.0", no data is selected. **data** is "".

Examples:

CMS Pipelines has built into the SPEC stage its own programming language. It is Rexx-like-but-not-quite-Rexx. For NetRexx Pipelines we have built in the worlds best scripting language: NetRexx. This gives all the power, but with a somewhat different syntax, of the CMS version. It is incorporated into the "Conversion" phase, with the key word NETREXX (or NR) and a delimitedString containing the NetRexx source. In running, this code is encapsulated in a method. The data selected in the Input phase is available as the variable **data**. And whatever is returned is passed to the Output phase of the stage. (As a convenience, if the last statement is not RETURN, the statement "**return data**" is automatically added.)

So all of these pipes work the same (giving **cba**):

- pipe "literal abc | spec 1-* NR /return data.reverse/ 1 | cons"
- pipe "literal abc | spec 1-* NR @rev = data.reverse; return rev@ 1 | cons"
- pipe "literal abc | spec 1-* NR %data = data.reverse% 1 | cons"
- pipe "literal abc | spec 1-* NR /data = data.reverse/ 1 | cons"

Using COUNTERS:

- pipe "literal abc | spec 1-* NR /counter[7]=counter[7]+1;ret=data.reverse counter[7];return ret/ 1 | cons"
  gives **cba 1**

Using Field Ids (and "." for output position). In this case the variable data is not used at all:

- pipe "literal abc | spec 1-* . 1 NR @rev = field['T'].reverse; return rev@ 1 | cons"
- pipe "literal abc | spec T:1-* . t:1 NR @rev = field['T'].reverse; return rev@ 1 | cons"

"T:" and "t:" are different.

Using comma to separate groups:

- pipe "literal abc | spec T:1-* 6-16, 1.2 NR /rev = field['T'].reverse; return rev/ 1.4 | cons"

gives **cba**

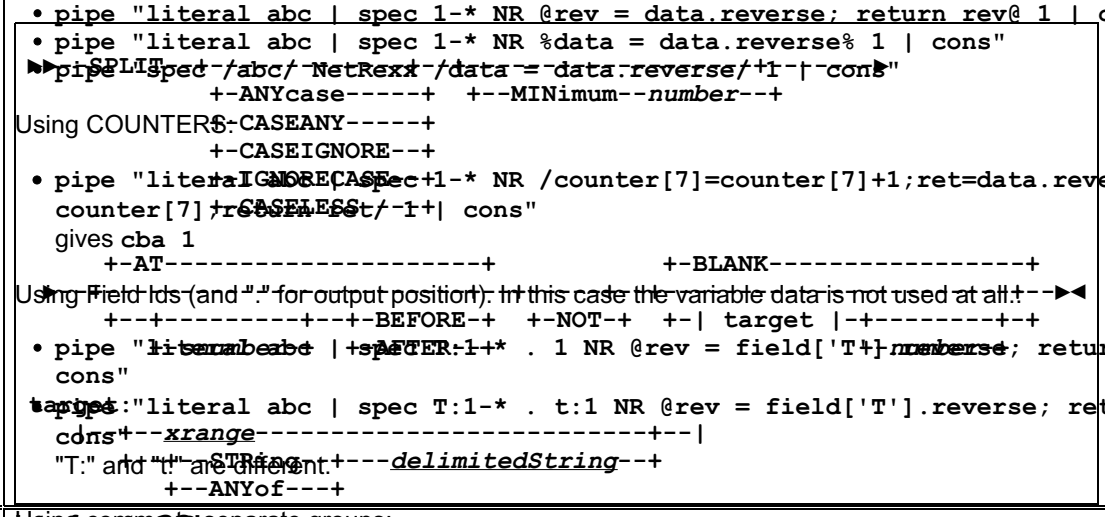
spill

Spill Long Lines at Word Boundaries

- Not implemented in Netrexx Pipelines.

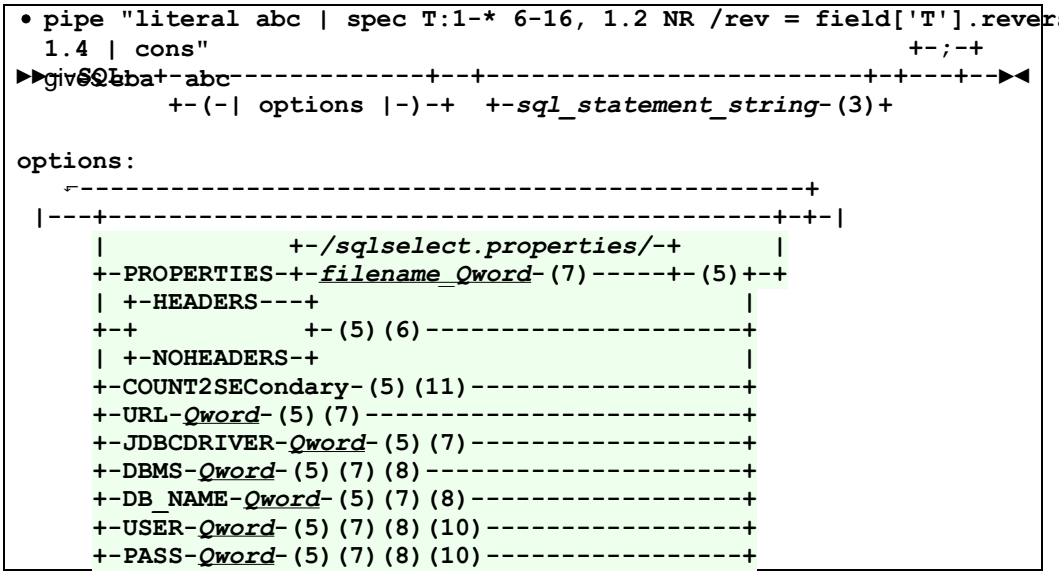
split

Split Records Relative to a Target



sql
3.09

Interface to SQL



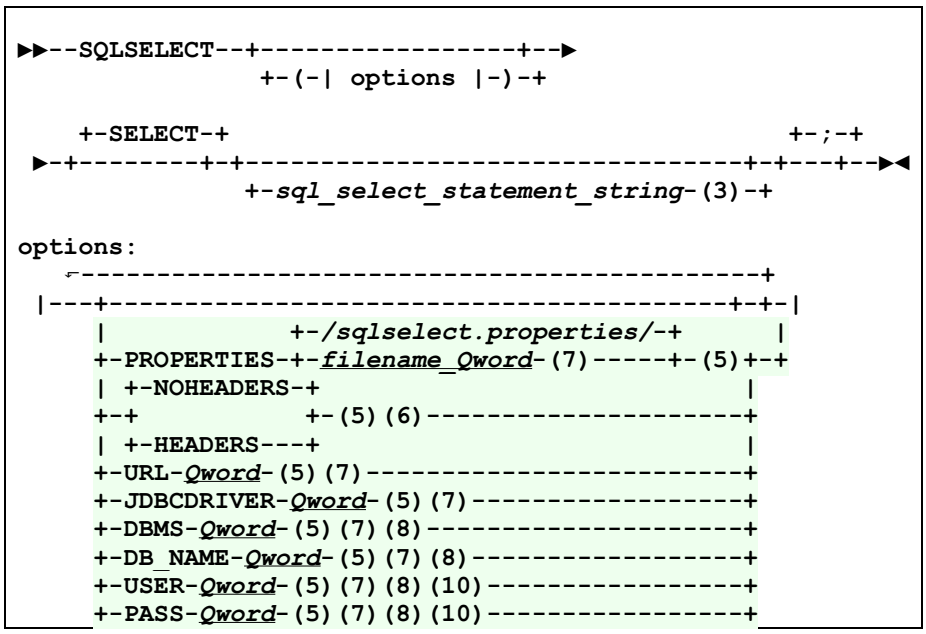
- uses jdbc to select from any jdbc enabled dbms
- properties file (sqlselect.properties default) is read from the secondary input stream to find jdbcdriver name, url, user, pass
- sample properties file:
-

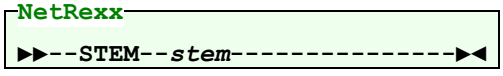
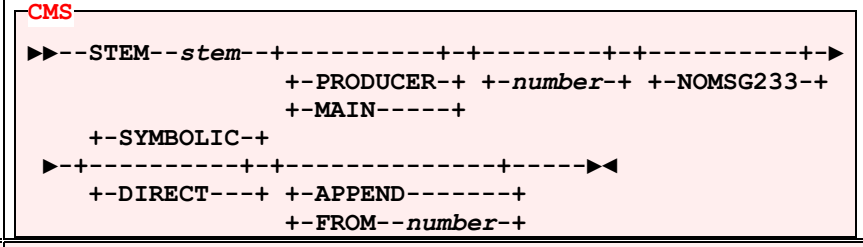
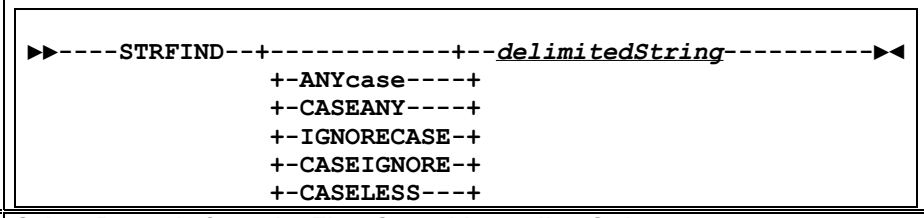
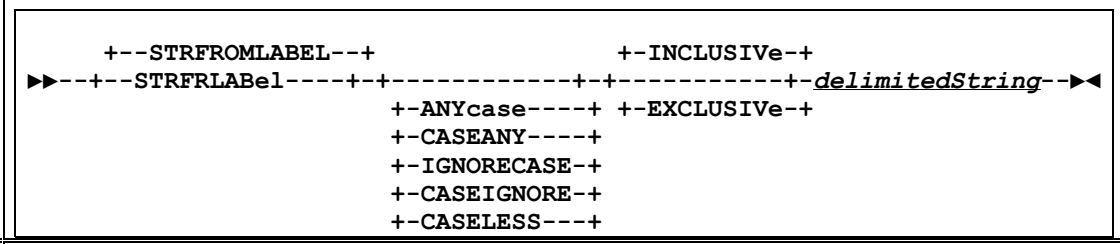
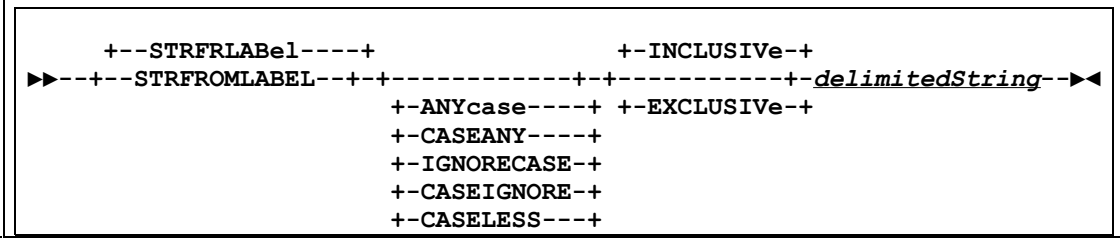
```
#JDBC driver name
#Tue Feb 03 23:29:43 GMT+01:00 1998
jdbcdriver=com.imaginary.sql.msql.MsqlDriver
url=jdbc:msql://localhost:1114/TESTDB
# the following are not needed for some DBMS, ex: SQLite
```

user=db_user_name
pass=password_for_db

-
- if this file is not found default (compiled in) values are used
- (1) when using a sql select * (all columns) from the commandline, quote the query as in
  java pipes.compiler (query) "sql select * from dept | console"
- (2) the netrexx/jdbc combination is extremely case sensitive for column and table names
- (3) this sql_select_string executed, then statements are read from the primary input stream.
  this is optional in NetRexx Pipelines only.
- (4) CMS does not use the stream input
- (5) NetRexx Pipelines only
- (6) CMS Pipelines is implyed HEADERS only.
- (7) A Qword is an optionally quoted word. If it contains spaces, it must be quoted.
- (8) EXPERIMENTAL Subject to change. DBMS is the kind of database, e.g. SQLite. DB_name is the file name.
  These are used in place of URL and JDBCDRIVER. SQLite is the only one tested as of 8/15/20.
- (9) the SQLSELECT stage uses HEADERS as the default.
- (10) USER & PASS are needed for some DBMSs and not others, ex. SQLite.
- (11) the count or other output from non-select statements goes to the secondary output stream if connected, or is discarded. Otherwise it goes to the primary.
-
- Priority order for URL, JDBCDRIVER and DBMS, DB_NAME (first one found rules):
  1. option in the SQL command string
  2. from secondary input stream
  3. from "sql.properties" file or from file specified by PROPERTIES option
  4. Builtin

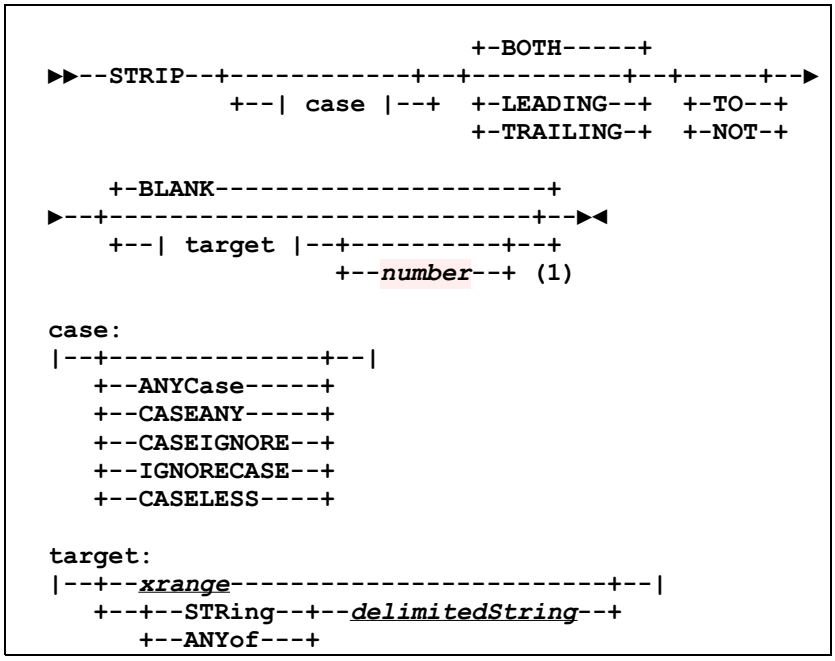
sqlcodes	**Write the last 11 SQL Codes Received** • Not implemented in Netrexx Pipelines.

sqlselect	**Query a Database and Format Result**  • (1) when using a sqlselect * (all columns) from the commandline, quote the query as in java pipes.compiler (query) "sqlselect * from dept | console" • (2) the netrexx/jdbc combination is extremely case sensitive for column and table names • (3) if no sql_select_string is specified, it is read from the primary input stream. this is optional in NetRexx Pipelines only. CMS does not use the stream input. • (4) a maximum of only one record is ever read from the primary input stream. • (5) NetRexx Pipelines only • (6) CMS Pipelines is implied HEADERS only. • (7) A Qword is an optionally quoted word. If it contains spaces, it must be quoted. • (8) EXPERIMENTAL Subject to change. DBMS is the kind of database, e.g. SQLite. DB_name is the file name. These are used in place of URL and JDBCDRIVER. SQLite is the only one tested as of 8/15/20. • (9) the SQL stage uses NOHEADERS as the default. • (10) USER & PASS are needed for some DBMSs and not others, ex. SQLite. • Priority order for URL, JDBCDRIVER, DBMS, DB_NAME, USER, & PASS (first one found rules): 1. option in the SQL command string 2. from secondary input stream 3. from "sqlselect.properties" file or from file specified by PROPERTIES option 4. Builtin
stack	**Read or Write the Program Stack** • Not implemented in Netrexx Pipelines.
starmon	**Write Records from the *MONITOR System Service** • Not implemented in Netrexx Pipelines.
starmsg	**Write Lines from a CP System Service** • Not implemented in Netrexx Pipelines.
starsys	**Write Lines from a Two-way CP System Service** • Not implemented in Netrexx Pipelines.
state	**Provide Information about CMS Files** • Not implemented in Netrexx Pipelines.
state	**Verify that Data Set Exists** • Not implemented in Netrexx Pipelines.
statew	**Provide Information about Writable CMS Files** • Not implemented in Netrexx Pipelines.

stem	**Retrieve or Set Variables in a REXX or CLIST Variable Pool** NetRexx  CMS 
stfle	**Store Facilities List** • Not implemented in Netrexx Pipelines.
storage	**Read or Write Virtual Machine Storage** • Not implemented in Netrexx Pipelines.
strasmfind	**Select Statements from an Assembler File as XEDIT Find** • Not implemented in Netrexx Pipelines.
strasmnfind	**Select Statements from an Assembler File as XEDIT NFind** • Not implemented in Netrexx Pipelines.
strfind	**Select Lines by XEDIT Find Logic** 
strfrlabel *strfrlabe* *strfrlab* *strfromlabel*	**Select Records from the First One with Leading String** 
strfromlabel *strfrlabel* *strfrlabe* *strfrlab*	**Select Records from the First One with Leading String** 

strip

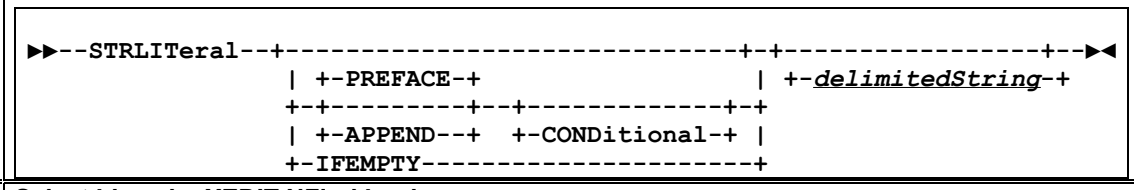
Remove Leading or Trailing Characters



- (1) Not implemented in Netrexx Pipelines.

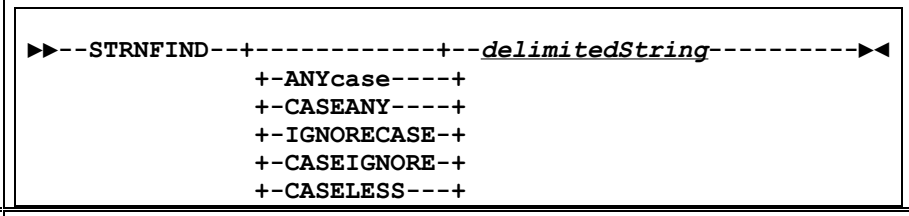
strliteral
strlitera
strliter
strlite
strlit

Write the Argument String



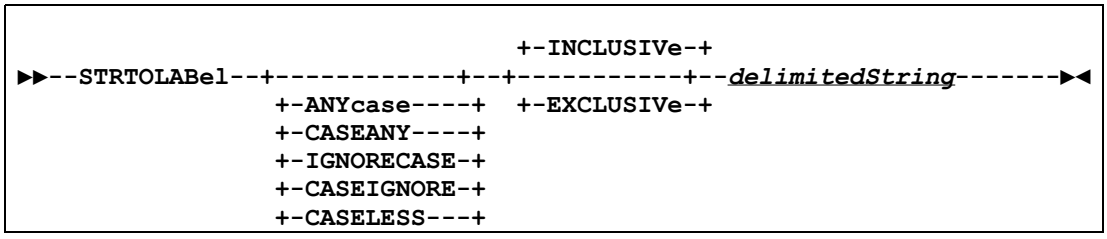
strnfind

Select Lines by XEDIT NFind Logic



strtolabel
strtolabe
strtolab

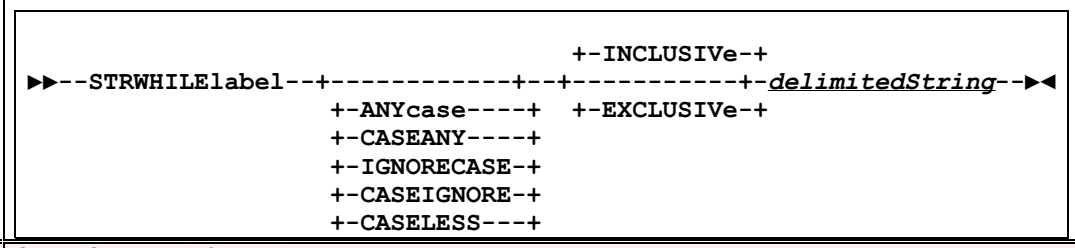
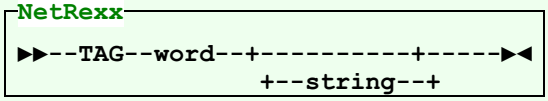
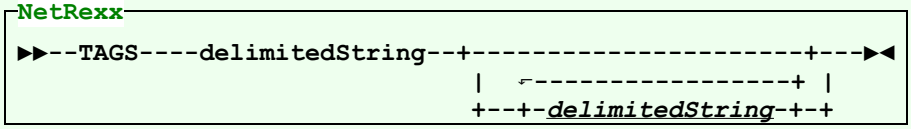
Select Records to the First One with Leading String

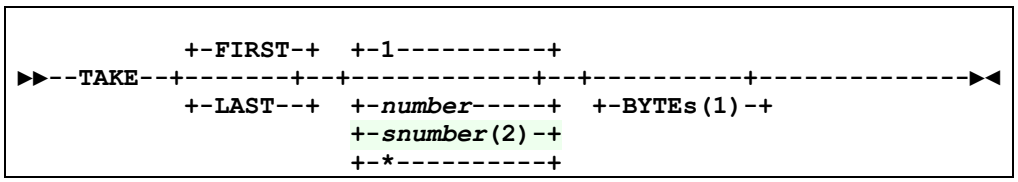


structure
struct

Manage Structure Defnitions

- Not implemented in Netrexx Pipelines.

strwhilelable ***strwhilelabl*** ***strwhilelab*** ***strwhilela*** ***strwhilel*** ***strwhile*** *3.09*	**Select Run of Records with Leading String** 
stsi	**Store System Information** • Not implemented in Netrexx Pipelines.
subcom	**Issue Commands to a Subcommand Environment** • Not implemented in Netrexx Pipelines.
substring ***substr***	**Write substring of record** • Not implemented in Netrexx Pipelines.
synchronise ***sync*** ***synchronize***	**Synchronise Records on Multiple Streams** • Not implemented in Netrexx Pipelines.
synchronize ***sync*** ***synchronise***	**Synchronise Records on Multiple Streams** • Not implemented in Netrexx Pipelines.
sysdsn	**Test whether Data Set Exists** • Not implemented in Netrexx Pipelines.
sysout	**Write System Output Data Set** • Not implemented in Netrexx Pipelines.
sysvar	**Write System Variables to the Pipeline** • Not implemented in Netrexx Pipelines.
tag *3.11*	**Surrounds Input Records with a HTML tag and its End Tag** NetRexx  • Outputs a record: <word string>, then passes through all records on its primary input, and finally a record: </word>.
tags *3.11*	**Surrounds Input Records with HTML tags and their End Tags** NetRexx  • Outputs a record for each delimitedString: <delimitedString>, then passes through all records on its primary input, and finally a record for each, in reverse order: </first_word_of_delimitedString>. • Any delimitedString may be a single word.

take	**Select Records from the Beginning or End of the File**  • (1) CMS must be BYTES • (2) Not CMS; NetRexx Pipelines: minus reverses first/last
tape	**Read or Write Tapes** • Not implemented in Netrexx Pipelines.
tcpchsum	**Compute One's complement Checksum of a Message** • Not implemented in Netrexx Pipelines.

tcpclient

Connect to a TCP/IP Server and Exchange Data

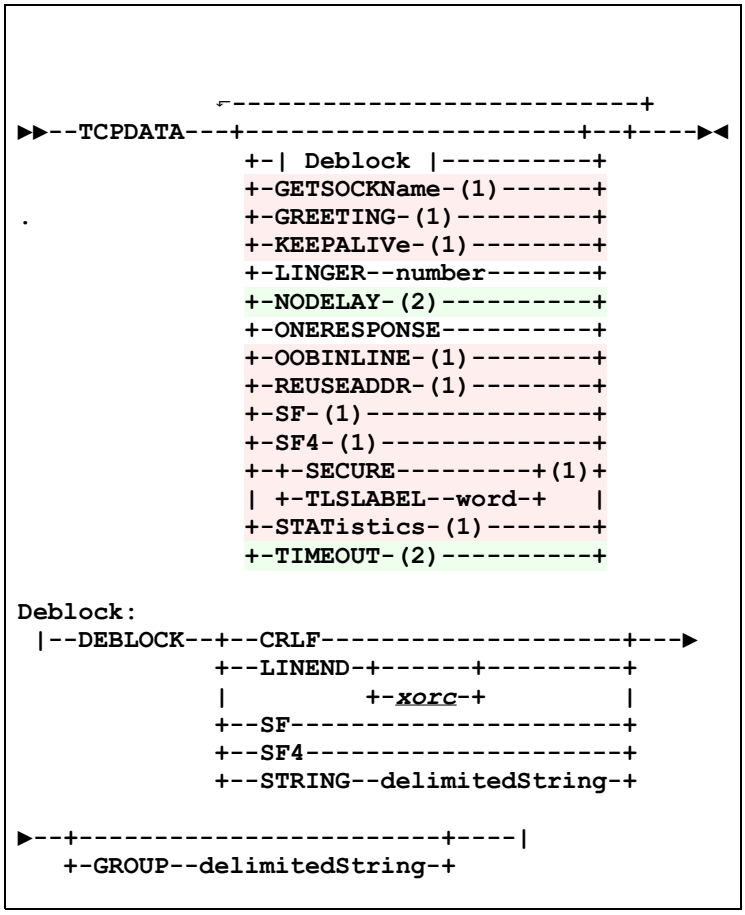
- (1) CMS Pipelines Only.
- (2) NetRexx Pipelines Only.

The options implemented are similar to the CMS definition.

- linger - wait a bit before terminating the last read (units SECONDS)
- timeout - wait this long before timing reads out (units MS)
- deblock - If deblock is omitted a copy stage is used.
- group - similar to CMS. A delimited string containing a stage is expected. You can use a run of stages, but its is dangerous since you to know the stage sep character being used...
- greeting - expect a greeting message and discard it
- nodelay - use the nodelay option
- keepalive - enable keep alive socket option
- oneresponse - synchronize cmds/replys

tcpdata

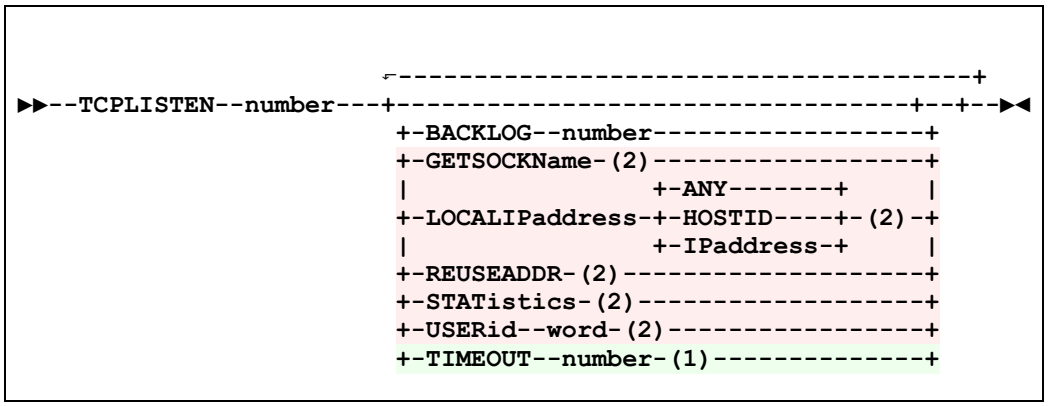
Read from and Write to a TCP/IP Socket



- Simple tcpdata implementation.
- (1) CMS Pipelines Only
- (2) NetRexx Pipelines Only
  - linger - wait a bit before terminating the last read (units SECONDS)
  - timeout - wait this long before timing reads out (units MS)
  - deblock - If deblock is ommited a copy stage is used.
  - group - similiar to cms. A delimited string containing a stage is expected. You can use a run of stages, but its is dangerous since you to know the stage sep character being used...
  - nodelay - use the nodelay option
  - oneresponse - synchronize requests/replies

tcplisten

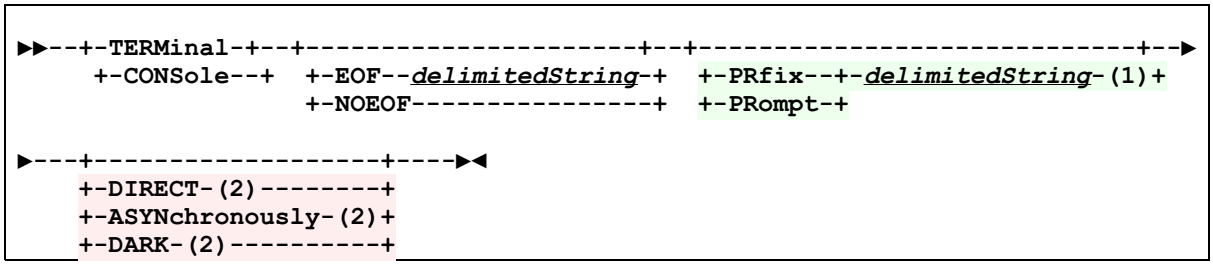
Listen on a TCP Port



- (1) NetRexx Pipelines only.
- (2) CMS Pipelines only.
- Simple tcplisten implementation. You can only supply the port and a timeout value. Its ignored unless tcplisten's output stream has been severed, in which case tcplisten terminates.
- If input stream 0 is connected, tcplisten does a peekto before calling the accept method. The object is consumed after the output of the socket object returns.

terminal
termina
termina
termin
termi
term
console
consol
conso
cons
cons
3.11

Read or Write the Terminal in Line Mode



- (1) NetRexx only
  On first stage, delimitedString is put out as a prompt
  On other stages, each line is prefixed with delimitedString
  Outout to next stage does NOT include delimitedString
  Either keyword can be used for either stage
- (2) CMS only

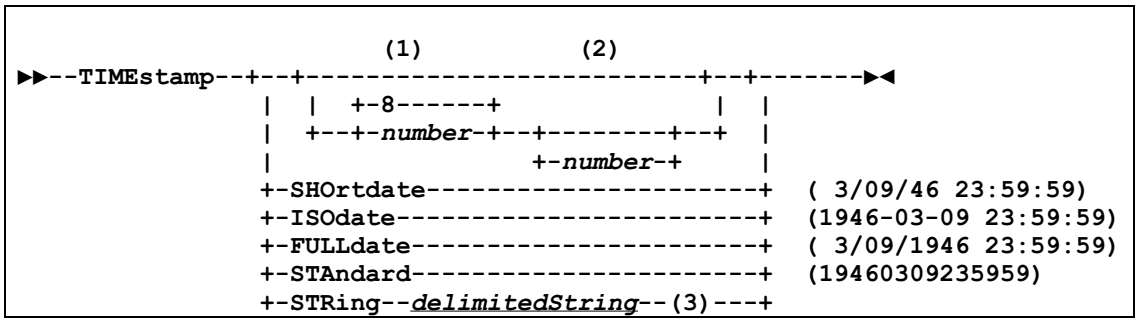
threeway

Split record three ways

- Not implemented in Netrexx Pipelines.

timestamp
timestam
timesta
timest
times
time

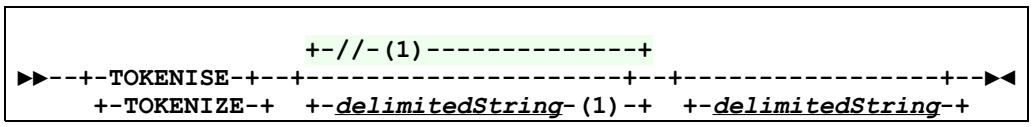
Prefix the Date and Time to Records



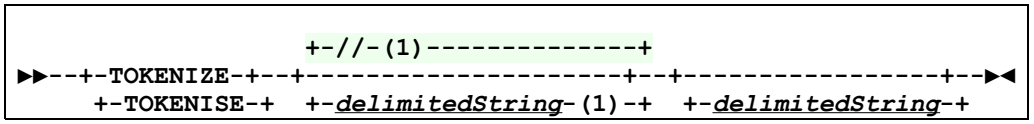
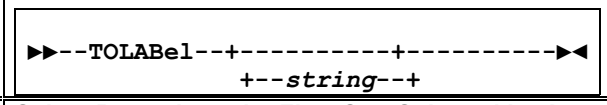
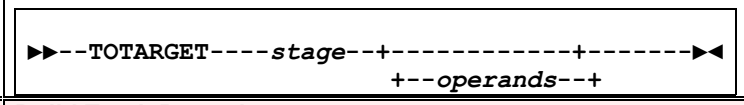
- (1) First character, from right, to include, <= 16
- (2) Count of characters to include. <= 16 - (1). Default = (1)
- (3)
  - %% A single %.
  - %Y Four digits year including century (0000-9999).
  - %y Two-digit year of century (00-99).
  - %m Two-digit month (01-12).
  - %n Two-digit month with initial zero changed to blank (1-12).
  - %d Two-digit day of month (01-31).
  - %e Two-digit day of month with initial zero changed to blank (1-31).
  - %j Julian day of year (001-366).
  - %H Hour, 24-hour clock (00-23).
  - %k Hour, 24-hour clock with leading zero blank (0-23).
  - %M Minute (00-59).
  - %S Second (00-60).
  - %F Equivalent to %Y-%m-%d (the ISO 8601 date format).
  - %T Short for %H:%M:%S.
  - %t Tens and hundredth of a second (00-99).

tokenise
tokenize

Tokenise Records

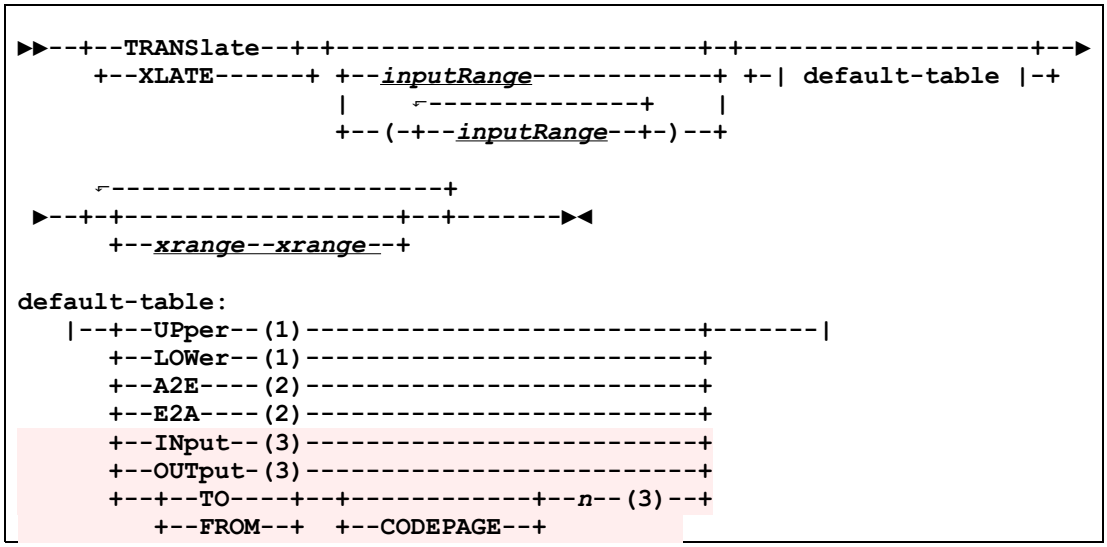


- (1) In CMS Pipelines, the first delimited string is required. In NetRexx Pipelines, it defaults to // if no second string.

tokenize ***tokenise***	**Tokenise Records**  • (1) In CMS Pipelines, the first delimited string is required. In NetRexx Pipelines, it defaults to // if no second string.
tolabel ***tolabe*** ***tolab***	**Select Records to the First One with Leading String** 
totarget	**Select Records to the First One Selected by Argument Stage** 
trackblock	**Build Track Record** • Not implemented in Netrexx Pipelines.
trackdeblock	**Deblock Track** • Not implemented in Netrexx Pipelines.
trackread	**Read Full Tracks from ECKD Device** • Not implemented in Netrexx Pipelines.
tracksquish	**Squish Tracks** • Not implemented in Netrexx Pipelines.
trackverify	**Verify Track Format** • Not implemented in Netrexx Pipelines.
trackwrite	**Write Full Tracks to ECKD Device** • Not implemented in Netrexx Pipelines.
trackxpand	**Unsquish Tracks** • Not implemented in Netrexx Pipelines.

translate
translat
transla
transl
trans
xlate

Transliterate Contents of Records



Notes:

- (1) UTF-16 (ASCII) in NJPipes, probably EBCIDIC in CMS.
- (2) In Netrexx Pipelines. EBCDIC to ASCII or ASCII to EBCDIC. Maybe in CMS, the documentation is unclear.
- (3) Not yet in NetRexx Pipelines
- [4] NetRexx Pipelines only: The secondary input stream is not yet supported.

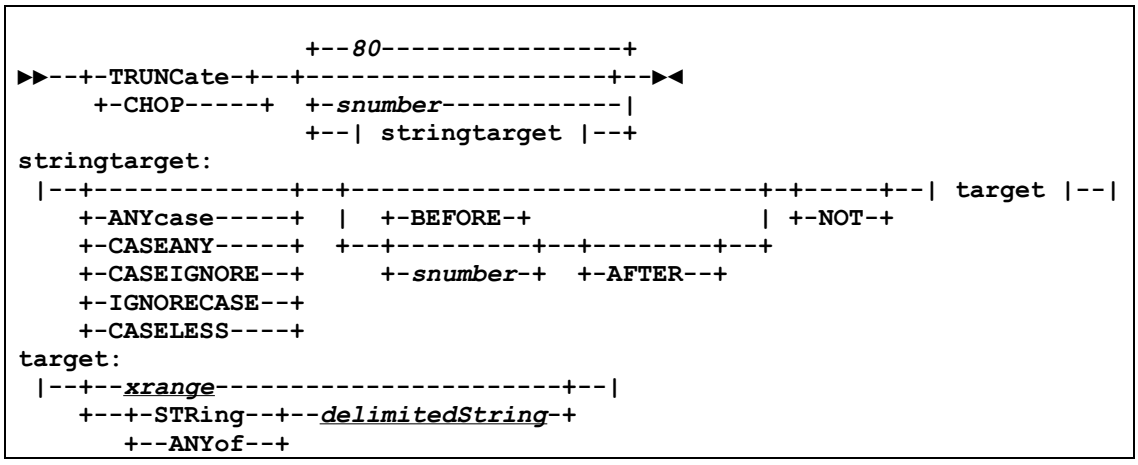
trfread

Read a Trace File

- Not implemented in Netrexx Pipelines.

truncate
truncat
trunca
trunc
chop

Truncate the Record



tso

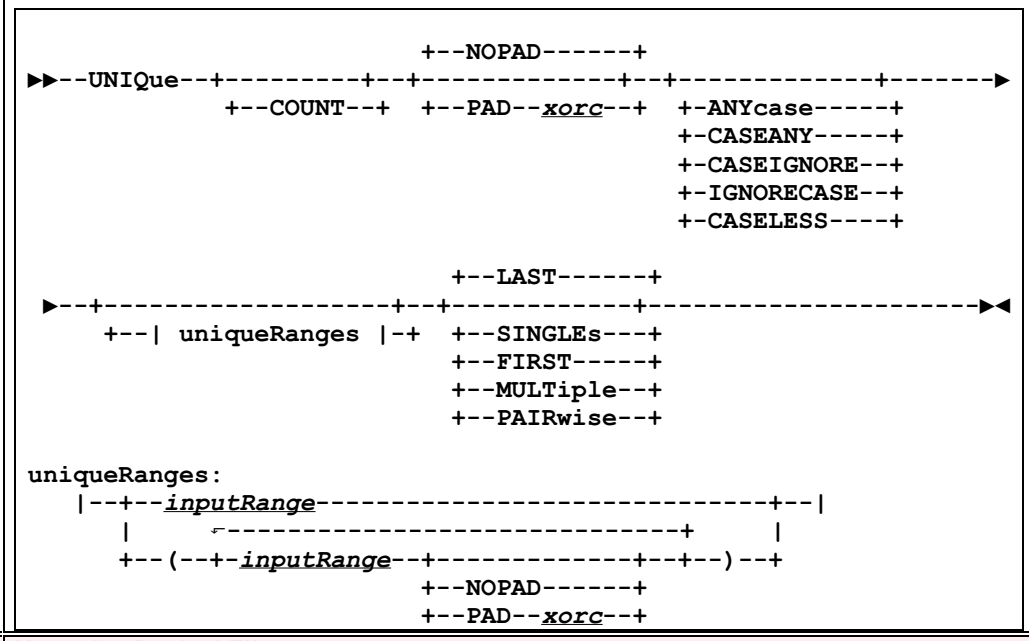
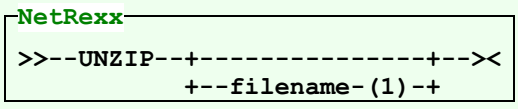
Issue TSO Commands, Write Response to Pipeline

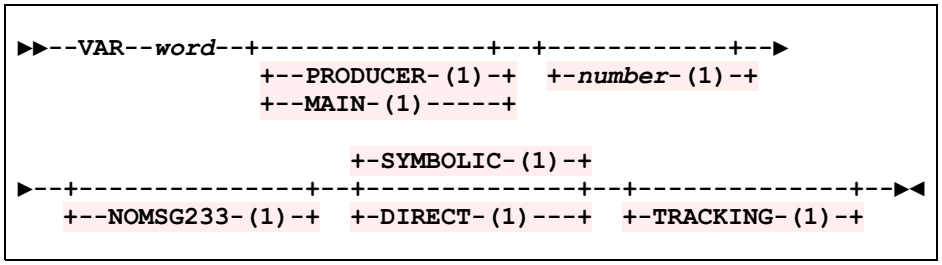
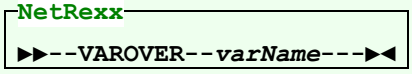
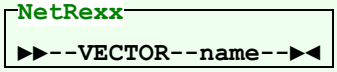
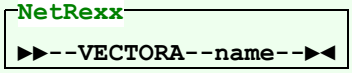
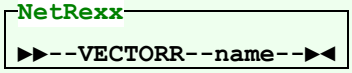
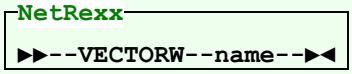
- Not implemented in Netrexx Pipelines.

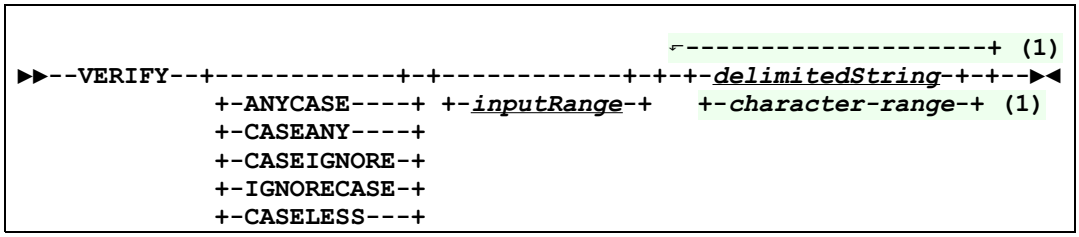
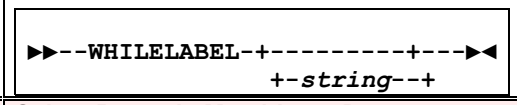
udp

Read and Write an UDP Port

- Not implemented in Netrexx Pipelines.

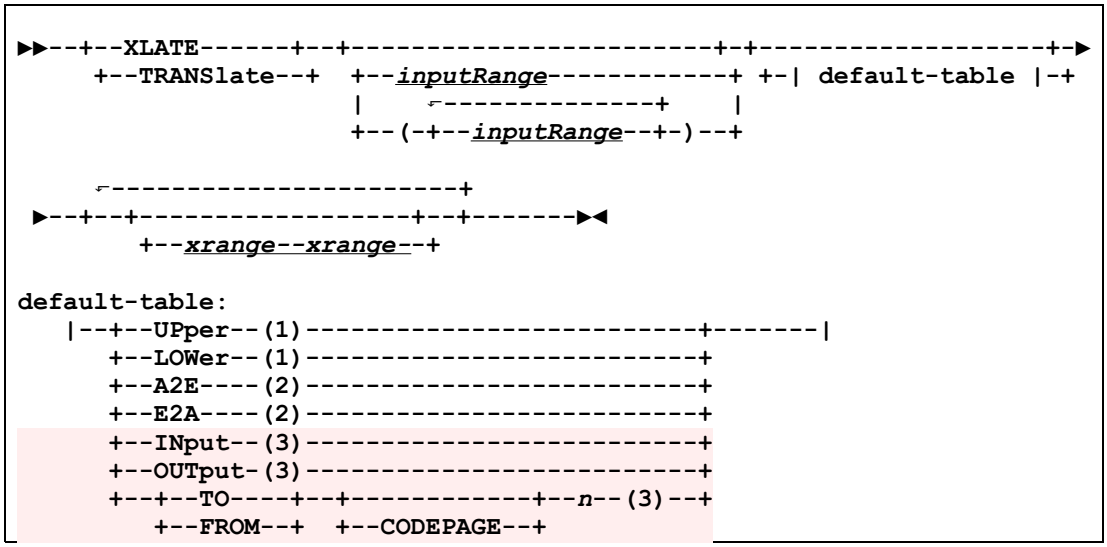
unique ***uniqu*** ***uniq***	**Discard or Retain Duplicate Lines** 
unpack	**Unpack a Packed File** • Not implemented in Netrexx Pipelines.
untab	**Replace Tabulate Characters with Blanks** • Not implemented in Netrexx Pipelines.
unzip	**Extract Files From a ZIP Archive**  Notes: • (1) If filename is not specified, it is read from the primary input stream. Succeeding input objects are ignored. • The extracted file names are passed to the primary output stream. • Any existing files will be replaced. • This is NetRexx Pipelines only.
update	**Apply an Update File** • Not implemented in Netrexx Pipelines.
urldeblock	**Process Universal Resource Locator** • Not implemented in Netrexx Pipelines.
uro	**Write Unit Record Output** • Not implemented in Netrexx Pipelines.
utf	**Convert between UTF-8, UTF-16, and UTF-32** • Not implemented in Netrexx Pipelines.

var	**Retrieve or Set a Variable in a REXX or CLIST Variable Pool**  • In NetRexx Pipelines, this can only read vars. It must be the first stage in a pipe. • (1) CMS Pipelines only.
vardrop	**Drop Variables in a REXX Variable Pool** • Not implemented in Netrexx Pipelines.
varfetch	**Fetch Variables in a REXX or CLIST Variable Pool** • Not implemented in Netrexx Pipelines.
varload	**Set Variables in a REXX or CLIST Variable Pool** • Not implemented in Netrexx Pipelines.
varover *3.09*	**Write the Values of Stems**  • NetRexx Pipelines only; not CMS Pipelines • If the secondary output stream is connected, the root is passed on it.
varset	**Set Variables in a REXX or CLIST Variable Pool** • Not implemented in Netrexx Pipelines.
vchar	**Recode Characters to Different Length** • Not implemented in Netrexx Pipelines.
vector	**Read or Write an Array of Vectors**  • Pipes for NetRexx only.
vectora	**Add to an Array of Vectors**  • Pipes for NetRexx only.
vectorr	**Read From an Array of Vectors**  • Pipes for NetRexx only.
vectorw	**Write to an Array of Vectors**  • Pipes for NetRexx only.

verify *3.09*	**Verify that Record Contains only Specified Characters**  • (1) NetRexx Pipelines only • (1) *character-range* is *xorc*-*xorc* • (1) Examples: A-Z 0-9 c-g a4-ba; 16-bit Unicode characters or hex numbers • (1) Any number greater than zero, any order, of delimitdStrings and character-ranges are allowed.
vmc	**Write VMCF Reply** • Not implemented in Netrexx Pipelines.
vmcdata	**Receive, Reply, or Reject a Send or Send/receive Request** • Not implemented in Netrexx Pipelines.
vmclient	**Send VMCF Requests** • Not implemented in Netrexx Pipelines.
vmclisten	**Listen for VMCF Requests** • Not implemented in Netrexx Pipelines.
waitdev	**Wait for an Interrupt from a Device** • Not implemented in Netrexx Pipelines.
warp	**Pipeline Wormhole** • Not implemented in Netrexx Pipelines.
warplist	**List Wormholes** • Not implemented in Netrexx Pipelines.
whilelabel *3.09*	**Select Run of Records with Leading String** 
wildcard	**Select Records Matching a Pattern** • Not implemented in Netrexx Pipelines.
writepds	**Store Members into a Partitioned Data Set** • Not implemented in Netrexx Pipelines.
xab	**Read or Write External Attribute Buffers** • Not implemented in Netrexx Pipelines.
xedit	**Read or Write a File in the XEDIT Ring** • Not implemented in Netrexx Pipelines.

xlate
translate
translat
transla
transl
trans

Transliterate Contents of Records



Notes:

- (1) UTF-16 (ASCII) in NJPipes, probably EBCIDIC in CMS.
- (2) In Netrexx Pipelines. EBCDIC to ASCII or ASCII to EBCDIC. Maybe in CMS, the documentation is unclear.
- (3) Not yet in NetRexx Pipelines
- [4] NetRexx Pipelines only: The secondary input stream is not yet supported.

xmsg

Issue XEDIT Messages

- Not implemented in Netrexx Pipelines.

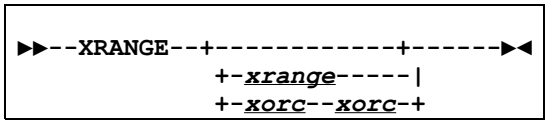
xpndhi

Expand Highlighting to Space between Words

- Not implemented in Netrexx Pipelines.

xrange
3.09

Write a Range of Characters



- NetRexx uses UTF-16 (ASCII) and CMS uses EBCDIC

zip	**Add Files To a new ZIP Archive** NetRexx ▶▶--ZIP--+-----------+--▶◀ +--name-(1)-+ • (1) name is the zip file name. If not provided, the first entry on the primary input stream is used. • (1) If no extension is provided in name, ".ZIP" is added. • (1) Any existing file is replaced. • Subsquent records on the primary input stream are filenames to be added to the Zip archive file. • File names added passed out on primary out stream. • NetRexx Pipelines only.
zone	**Run Selection Stage on Subset of Input Record** ▶▶--ZONE--+--------------+--+---------+--+-----------+--▶ +--*inputRange*--+ +--CASEI--+ +--REVERSE--+ ▶--*stage*--+------------+---▶◀ +--*operands*--+
_rexx	**Cast Input and/or Output of a Stage to Type Rexx** NetRexx +--BOTH------+ ▶▶--_REXX--+------------+--*stage*--+------------+---▶◀ +-+-IN-+-----+ +--*operands*--+ | +-TO-+ | +-+-OUT--+---+ +-FROM-+
_string	**Cast Input and/or Output of a Stage to Type String** NetRexx +--BOTH------+ ▶▶--_STRING--+------------+--*stage*--+------------+---▶◀ +-+-IN-+-----+ +--*operands*--+ | +-TO-+ | +-+-OUT--+---+ +-FROM-+