NetRexx Pipelines: What's New (and Old)

Jeff Hennick

31st International Rexx Language Symposium, Sept. 2020

Everywhere in the Universe: Online

Pipeline

- When the output of one program is the input to a second
- Symbolized by the “pipe character:” |
- Example:

```
pipe disk input file | count words | console
```

- Individual programs called Stages, or Filters
- Some select some records and not others based on content or position in the file
- Some alter the record in some way before passing it on

Quick History

- 1973 Unix, Douglas McIlroy and Douglas McIlroy, Bell Labs
  - Programs with Standard Input and Standard Output
  - Byte stream oriented file notation
  - Single Input and Output Streams
- 1980-1992, CMS, John Hartmann, IBM Denmark
  - No Standard I/O, uses unique builtin programs
  - Record stream oriented
  - Multiple Input and Output Streams

Quick History II

- 1997 NetRexx Pipelines, Ed Tomlinson
  - Port from CMS to NetRexx for Run Anywhere
  - Object stream oriented
    - Primarily NetRexx records
  - Multi stream input & output
  - Stages, both builtin and user, written in NetRexx

Quick History III

- 1979-1982 Rexx, Mike Cowlishaw, IBM
- 1991-1996 Java, James Gosling, Sun Microsystems
- 1996 NetRexx, Mike Cowlishaw, IBM
- 1997-1999 NJPipelines, Ed Tomlinson
- 2011 NetRexx Open Source, RexxLA
- 2011 NJPipelines Open Source, RexxLA
- 2020 Push to make NetRexx Pipelines as compatible with CMS Pipelines as possible.

Pipelines: NetRexx vs CMS

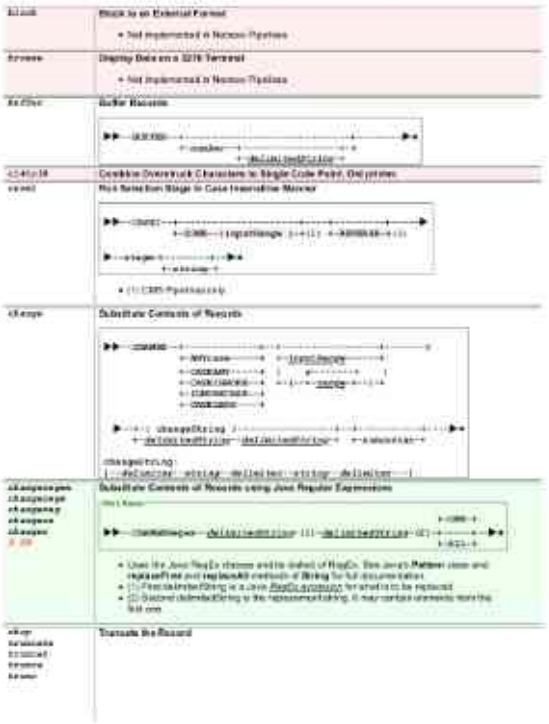
- Very similar
- Different operating environments
  - Quote marks in different places
  - | mark used as system pipe in many NetRexx environments
  - No CP, 3270, APL, etc. Stages In NetRexx
  - Single file system model
  - Default SQL dbms and options differ
- Objects vs only Text Records expands the possibilities
- Java brings new opportunities, e.g. Regular Expressions

Pipelines: NetRexx vs CMS

- Most Stages now work identically – lots of 2020 adds & updates
- Close to 1000 stage tests
  - 132 tests for the Dateconvert stage alone
- Most CMS stages not in NetRexx make no sense outside CMS
  - A few left to port to NetRexx
- Some new stages only in NetRexx
- Documentation – NetRexx has extensive HTML online tables
- SQL works with any Java accessible dbms (dozens)

Documentation

Full



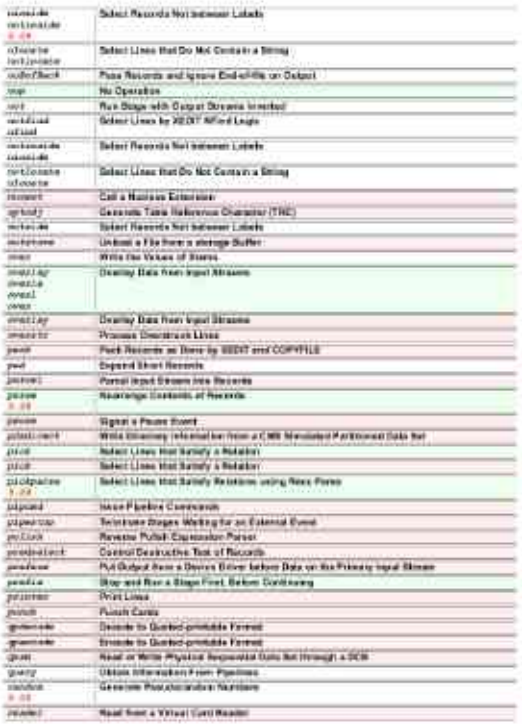
Pink: CMS Only

White: Both NetRexx & CMS

Green: NetRexx only

Index

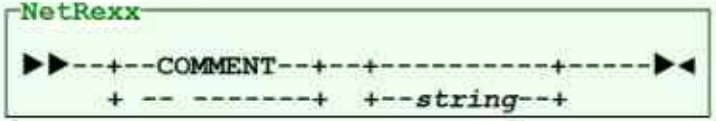
Double click row to open



NetRexx Pipelines exclusive Stages

- Comment (- -) a stage, not a language feature
- Compare – two input streams, report if the same or different
  - Used extensively in our stage tests
- Parse based select and change stages
- Regex based select and change stages

COMMENT

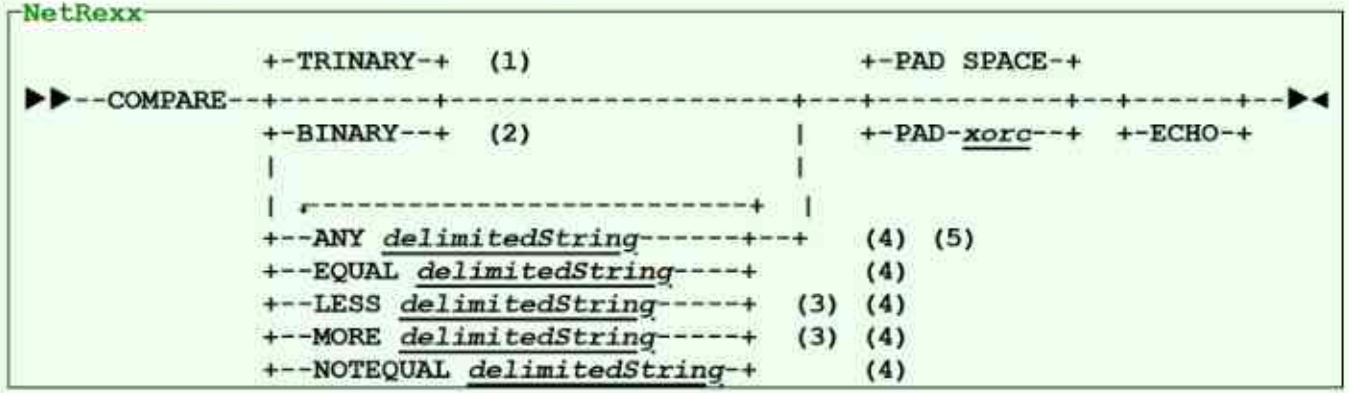
comment -- *3.09*	**Comment stage** NetRexx  • Not in CMS Pipelines; • This is a STAGE, not a programming comment. It must have a SPACE after --. • It must have either a stageEnd or pipeEnd. • If ended with a stageEnd, it passes records through on primary input to output streams. • If ended with a pipeEnd, it does NOT pass records through. • If used before a driver stage, it must have a pipeEnd.

COMPARE diagram

compare

Compare Primary and Secondary Streams, Write the Result

NetRexx



COMPARE notes

- (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 njpipes 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

Example: COMMENT & COMPARE

Test of REVERSE stage

```
-- reverse string  IBM ex 1 p541 ?
   literal Hello, World! |       -- test data |
   reverse |                     -- stage under test |
   c1: compare ,
    any ~# 1 reverse string IBM ex 1 p541~ ,
    equal ~*OK*~ ,
    notequal ~**FAIL**  at rec \\c col \\b.~ ,
    less ~Less:\nActual:\n\\p\nExpected:\n\\s\n~ ,
    more ~More:\nActual:\n\\p\nExpected:\n\\s\n~ |
   o: ?
   literal !dlroW ,olleH  |      -- expected output |
   c1: ?
```

```
PS C:\Users\Jeff\documents\pipe tests> java reverse_tests1
# 1 reverse string IBM ex 1 p541 *OK*
# 2 reverse string leading spaces *OK*
# 3 reverse string trailing spaces *OK*
# 4 reverse original test *OK*
PS C:\Users\Jeff\documents\pipe tests>
```

Locate Records on Content

- CMS & NetRexx Pipelines have FIND and LOCATE, based on Xedit
- Also PICK that allows logical comparisons of the text
  - NetRexx has only a subset of options, for now
- Some other stages, too
- NetRexx adds
  - PICKPARSE – uses Parse notation
  - GREP – uses Regular expressions notation

PICKPARSE Stage

pickparse
3.09

Select Lines that Satisfy Relations using Rexx Parse

NetRexx

- 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 logic_template_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.
- The parse_template and logic_template(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.

GREP (or REGEX) Stage

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.
-

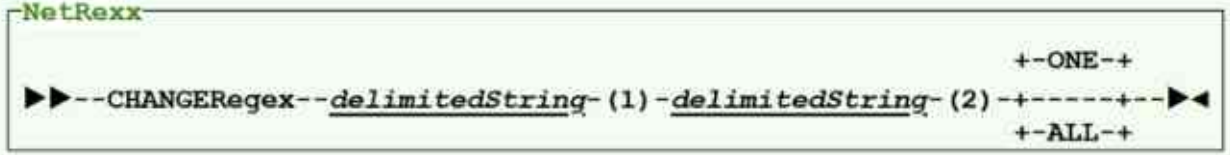
Change Record’s Content

- CMS & NetRexx have Spec, Change, and other stages for changing a record’s content
- NetRexx adds
  - Parse – uses Parse notation
  - Changeregex – uses Regular Expression notation

PARSE Stage

parse *3.09*	**Rearrange Contents of Records** NetRexx ▶▶--PARSE--*parse_template_Dstring*---*output_template_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 $$n. • NetRexx Pipelines only. • . • Example: `parse / 2 $1 +1/ /The second letter is "$1". $$1 won't be changed./`

CHANGEREGEX Stage

changeregex ***changerege*** ***changereg*** ***changere*** ***changer*** *3.09*	**Substitute Contents of Records using Java Regular Expressions**  • 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 delimitedString is a Java *RegEx expresion* for what is to be replaced. • (2) Second delimitedString is the replacement string. It may contain elements from the first one.

IP Address Stages Added

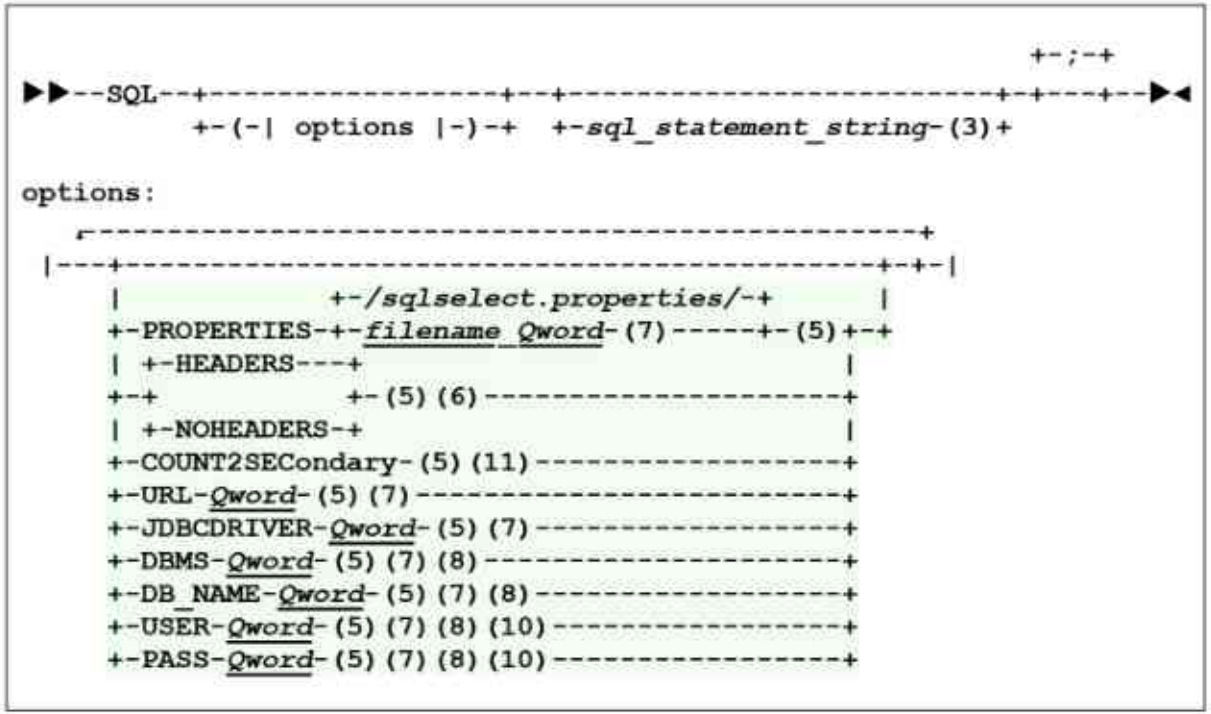
- HostByAddress
- HostByName
- HostId
- HostName

- By CollinK

SQL Stage

sql
3.09

Interface to SQL



SQL Example 1: SQLite

```
pipe (ct)
literal ,
  drop table if exists person; ,
  create table person (id integer, name string); ,
  insert into person values(1, 'leo-rexx'); ,
  insert into person values(2, 'yui-rexx'); ,
  select * from person |
split ; |
sql |
cons ?
```

Rows changed can be sent to secondary stream

```
PS C:\Users\Jeff\documents\pipe tests> java ct
0
0
1
1
ID--NAME--
1   leo-rexx
2   yui-rexx
```

From SELECT
Header row is optional

SQL Example 2: Manual Pages

```
pipe (make_stages_page)
  literal ,
    select * from stages_head;,
    select * from stage_scripts; ,
    select * from stages_style; ,
    select '</head><body>'; ,
    select * from stages_top; ,
    select '<table>'; ,
    select * from stages_table; ,
    select '</table></body></html>' |
  split ; |
  sql (dbms sqlite db_name c:\\Users\\Jeff\\NetRexx-Code\\documentation\\njpipes\\stages.db) |
  > stages.html ?
```

What Top Stages / Options Are Missing?

- Help
  - Need SQLite on the system (or port to another dbms)
  - Need to be able to find the database file
- Spec
  - Many of the recently added options
- Over – changed definition
  - What was over is now varover
  - Over is now a synonym for overlay, as in CMS

Stage Template 1

```
-- [stage_name].nrx NJPipe Stage
/*
 *  Copyright (C) [year]  [author]
 *  Distributed under the ICU 1.8.1 License with NO WARRANTIES of ANY kind.
 *  See LICENSE for the license and information on using, copying, modifying,
 *  and distributing this program.
 */
/*
 [date] New. [author]
*/
/** [stage_name]
   >>--[STAGENAME]----[OPTIONS]---------------------><
*/
```

Stage Template 2

```
options nostrictcase nostrictargs nostrictsignal
--package org.netrexx.njpipes.stages   -- keep commented out if not in the "stages" directory

import org.netrexx.njpipes.pipes.

class [stage_name] extends stage [uses DString, IRange]

method run()  -- this is the method the Pipeline scheudler calls
  /*
    Process commandline options and other set up
    IRange class, getKeyWord() method, and DString class can all help here.
  */
```

Stage Template 3

```
loop label Main forever
  record_in = [Rexx] peekto()
  /*       record_out = modify(record_in) and/or selected = [1 or 0]  */
  [if selected then] output(record_out)
  readto() -- clears the input and tells the previous stage all is OK here
catch stageError
  rc = rc()
end Main
Exit(rc*(rc<>12))  -- 12 is End-of-Data and OK
```

Contact

Questions / Feedback / Ideas

NetRexx Pipelines: What's New (and Old)

Jeff Hennick

Jeff@Jeff-H.com

31st International Rexx Language Symposium, Sept. 2020