IEEE P1003.2 Draft 11.2 – September 1991

supplementary groups. (On systems with user and group IDs that are 32-bit integers and with group names with a maximum of 8 bytes per name, 93 supplementary groups plus distinct effective and real group and user IDs could theoretically overflow the 2048-byte {LINE_MAX} text file line limit on the default output case. It would take about 186 supplementary groups to overflow the 2048-byte barrier using `id −G`.) This is not expected to be a problem in practice, but in cases where it is a concern, applications should consider using `fold −s` (see 4.25) before postprocessing the output of `id`.

History of Decisions Made

The 4BSD command `groups` was considered, but was not used as it did not provide the functionality of the `id` utility of the *SVID*. Also, it was thought that it would be easier to modify `id` to provide the additional functionality necessary to systems with multiple groups than to invent another command.

The options `−u`, `−g`, `−n`, and `−r` were added to ease the use of `id` with shell commands substitution. Without these options it is necessary to use some preprocessor such as `sed` to select the desired piece of information. Since output such as that produced by `id -u -n` is wanted frequently, it seemed desirable to add the options.

4.31 `join` — Relational database operator

4.31.1 Synopsis

`join` [`−a` *file_number* | `−v` *file_number*] [`−e` *string*] [`−o` *list*] [`−t` *char*]
    [`−1` *field*] [`−2` *field*] *file1 file2*

Obsolescent version:

`join` [`−a` *file_number*] [`−e` *string*] [`−j` *field*] [`−j1` *field*] [`−j2` *field*]
    [`−o` *list* . . .] [`−t` *char*] *file1 file2*

4.31.2 Description

The `join` utility shall perform an ‘‘equality join’’ on the files *file1* and *file2*. The joined files shall be written to the standard output.

The ‘‘join field’’ is a field in each file on which the files are compared. There shall be one line in the output for each pair of lines in *file1* and *file2* that have identical join fields. The output line by default shall consist of the join field, then the remaining fields from *file1*, then the remaining fields from *file2*. This format can be changed by using the `−o` option (see below). The `−a` option can be used to add unmatched lines to the output. The `−v` option can be used to output only unmatched lines.

By default, the files *file1* and *file2* should be ordered in the collating sequence of `sort −b` (see 4.58) on the fields on which they are to be joined, by default the first in each line. All selected output shall be written in the same collating sequence.

The default input field separators shall be <blank>s. In this case, multiple separators shall count as one field separator, and leading separators shall be ignored. The default output field separator shall be a <space>.

The field separator and collating sequence can be changed by using the `−t` option (see below).

If the input files are not in the appropriate collating sequence, the results are unspecified.

4.31.3 Options

The `join` utility shall conform to the utility argument syntax guidelines described in 2.10.2. The obsolescent version does not follow the utility argument syntax guidelines: the `−j1` and `−j2` options are multicharacter options and the `−o` option takes multiple arguments.

The following options shall be supported by the implementation:

`−a` *file_number*
: Produce a line for each unpairable line in file *file_number*, where *file_number* is `1` or `2`, in addition to the default output. If both `−a 1` and `−a 2` are specified, all unpairable lines shall be output.

`−e` *string*
: Replace empty output fields by string *string*.

`−j` *field*
: (Obsolescent.) Equivalent to: `−1` *field* `−2` *field*

`−j1` *field*
: (Obsolescent.) Equivalent to: `−1` *field*

`−j2` *field*
: (Obsolescent.) Equivalent to: `−2` *field*

`−o` *list*
: Construct the output line to comprise the fields specified in *list*, each element of which has the form *file_number*`.`*field*, where *file_number* is a file number and *field* is a decimal integer field number. The elements of *list* are either comma- or <blank>-separated, as specified in Guideline 8 in 2.10.2. The fields specified by *list* shall be written for all selected output lines. Fields selected by *list* that do not appear in the input shall be treated as empty output fields. (See the `−e` option.) The join field shall not be written unless specifically requested. The *list* shall be a single command line argument. However, as an obsolescent feature, the argument *list* can be multiple arguments on the command line. If this is the case, and if the `−o` option is the last option before *file1*, and if *file1* is of the form *string*.*string*, the results are undefined.

`-t` *char* Use character *char* as a separator, for both input and output. Every appearance of *char* in a line shall be significant. When this option is specified, the collating sequence should be the same as `sort` without the `-b` option.

`-v` *file_number*
Instead of the default output, produce a line only for each unpairable line in *file_number*, where *file_number* is 1 or 2. If both `-v 1` and `-v 2` are specified, all unpairable lines shall be output.

`-1` *field* Join on the *field* th field of file 1. Fields are decimal integers starting with 1.

`-2` *field* Join on the *field* th field of file 2. Fields are decimal integers starting with 1.

4.31.4 Operands

The following operands shall be supported by the implementation:

file1
file2 A pathname of a file to be joined. If either of the *file1* or *file2* operands is –, the standard input is used in its place.

4.31.5 External Influences

4.31.5.1 Standard Input

The standard input shall be used only if the *file1* or *file2* operand is –. See Input Files.

4.31.5.2 Input Files

The input files shall be text files.

4.31.5.3 Environment Variables

The following environment variables shall affect the execution of `join`:

LANG This variable shall determine the locale to use for the locale categories when both **LC_ALL** and the corresponding environment variable (beginning with **LC_**) do not specify a locale. See 2.6.

LC_ALL This variable shall determine the locale to be used to override any values for locale categories specified by the settings of **LANG** or any environment variables beginning with **LC_**.

LC_COLLATE This variable shall determine the collating sequence `join` expects to have been used when the input files were sorted.

LC_CTYPE This variable shall determine the locale for the interpretation of sequences of bytes of text data as characters (e.g., single- versus multibyte characters in arguments and input files).

LC_MESSAGES This variable shall determine the language in which messages should be written.

4.31.5.4 Asynchronous Events

Default.

4.31.6 External Effects

4.31.6.1 Standard Output

The `join` utility output shall be a concatenation of selected character fields. When the `−o` option is not specified, the output shall be:

`"%s%s%s\n"`, *<join field>*, *<other file1 fields>*, *<other file2 fields>*

If the join field is not the first field in either file, the *<other file fields>* are:

<fields preceding join field>, *<fields following join field>*

When the `−o` option is specified, the output format shall be:

`"%s\n"`, *<concatenation of fields>*

where the concatenation of fields is described by the `−o` option, above.

For either format, each field (except the last) shall be written with its trailing separator character. If the separator is the default (`<blank>`s), a single `<space>` character shall be written after each field (except the last).

4.31.6.2 Standard Error

Used only for diagnostic messages.

4.31.6.3 Output Files

None.

4.31.7 Extended Description

None.

4.31.8 Exit Status

The `join` utility shall exit with one of the following values:

0	All input files were output successfully.
>0	An error occurred.

4.31.9 Consequences of Errors

Default.

4.31.10 Rationale. *(This subclause is not a part of P1003.2)*

Examples, Usage

Pathnames consisting of numeric digits should not be specified directly following the `−o` list.

The developers of the standard believed that `join` should operate as documented in the *SVID* and BSD, not as historically implemented. Historical implementations do not behave as documented in these areas:

(1) Most implementations of `join` require using the `−o` option when using the `−e` option.

(2) Most implementations do not parse the `−o` option as documented, and parse the elements as separate *argv* items, until the item is not of the form *file_number* .*field* . This behavior is permitted as an obsolescent usage of the utility. To ensure maximum portability, *file1* should not be of the form *string* .*string* . A suitable alternative to guarantee portability would be to put the `−−` flag before any *file1* operand.

The obsolescent `−j`, `−j1`, and `−j2` options have been described to show how they have been used in historical implementations. Earlier drafts showed `−j` *file_number field* , but a space was never allowed before the *file_number* and two option arguments were never intended.

History of Decisions Made

The ability to specify *file2* as − is not historical practice; it was added for completeness.

As a result of a balloting comment, the `−v` option was added to the nonobsolescent version. This option was felt necessary because it permitted the writing of *only* those lines that do not match on the join field, as opposed to the `−a` option, which prints both lines that do and do not match. This additional facility is parallel with the `-v` option of `grep`.

4.32 `kill` — Terminate or signal processes

4.32.1 Synopsis

`kill -s` *signal_name pid* ...

`kill -l` [*exit_status*]

Obsolescent Versions :

`kill` [−*signal_name*] *pid* ...

`kill` [−*signal_number*] *pid* ...

4.32.2 Description

The `kill` utility shall send a signal to the process(es) specified by each *pid* operand.

For each *pid* operand, the `kill` utility shall perform actions equivalent to the POSIX.1 {8} *kill* () function called with the following arguments:

(1) The value of the *pid* operand shall be used as the *pid* argument.

(2) The *sig* argument is the value specified by the `-s` option, −*signal_number* option, or the −*signal_name* option, or by SIGTERM , if none of these options is specified.

4.32.3 Options

The `kill` utility shall conform to the utility argument syntax guidelines described in 2.10.2, except that in the obsolescent form, the −*signal_number* and −*signal_name* options are usually more than a single character.

The following options shall be supported by the implementation:

`-l` (The letter ell.) Write all values of *signal_name* supported by the implementation, if no operand is given. If an *exit_status* operand is given and it is a value of the `?` shell special parameter (see 3.5.2 and `wait` in 4.70) corresponding to a process that was terminated by a signal, the *signal_name* corresponding to the signal that terminated the process shall be written. If an *exit_status* operand is given and it is the unsigned decimal integer value of a signal number, the *signal_name* (the POSIX.1 {8}-defined symbolic constant name without the `SIG` prefix) corresponding to that signal shall be written. Otherwise, the results are unspecified.