NAME

git−lfs−migrate − Migrate history to or from Git LFS

SYNOPSIS

git lfs migrate *mode* [options] [−−] [branch ...]

DESCRIPTION

Convert files in a Git repository to or from Git LFS pointers, or summarize Git file sizes by file type. The **import** mode converts Git files (i.e., blobs) to Git LFS, while the **export** mode does the reverse, and the **info** mode provides an informational summary which may be useful in deciding which files to import or export.

In all modes, by default **git lfs migrate** operates only on the currently checked−out branch, and only on files (of any size and type) added in commits which do not exist on any remote. Multiple options are available to override these defaults.

When converting files to or from Git LFS, the **git lfs migrate** command will only make changes to your local repository and working copy, never any remotes. This is intentional as the **import** and **export** modes are generally "destructive" in the sense that they rewrite your Git history, changing commits and generating new commit SHAs. (The exception is the "no−rewrite" **import** sub−mode; see [IMPORT (NO REWRITE)] for details.)

You should therefore always first commit or stash any uncommitted work before using the **import** or **export** modes, and then validate the result of the migration before pushing the changes to your remotes, for instance by running the **info** mode and by examining your rewritten commit history.

Once you are satisfied with the changes, you will need to force−push the new Git history of any rewritten branches to all your remotes. This is a step which should be taken with care, since you will be altering the Git history on your remotes.

To examine or modify files in branches other than the currently checked−out one, branch refs may be specified directly, or provided in one or more **−−include−ref** options. They may also be excluded by prefixing them with ^ or providing them in **−−exclude−ref** options. Use the **−−everything** option to specify that all refs should be examined, including all remote refs. See [INCLUDE AND EXCLUDE (REFS)] for details.

For the **info** and **import** modes, all file types are considered by default; while useful in the **info** mode, this is often not desirable when importing, so either filename patterns (pathspecs) or the **−−fixup** option should normally be specified in that case. (At least one include pathspec is required for the **export** mode.) Pathspecs may be defined using the **−−include** and **−−exclude** options (**−I** and **−X** for short), as described in *INCLUDE AND EXCLUDE*.

As typical Git LFS usage depends on tracking specific file types using filename patterns defined in **.gitattributes** files, the **git lfs migrate** command will examine, create, and modify **.gitattributes** files as necessary.

The **import** mode (see *IMPORT*) will convert Git objects of the file types specified (e.g., with **−−include**) to Git LFS pointers, and will add entries for those file types to **.gitattributes** files, creating those files if they do not exist. The result should be as if **git lfs track** commands had been run at the points in your Git history corresponding to where each type of converted file first appears. The exception is if the **−−fixup** option is given, in which case the **import** mode will only examine any existing **.gitattributes** files and then convert Git objects which should be tracked by Git LFS but are not yet.

The **export** mode (see *EXPORT*) works as the reverse operation to the **import** mode, converting any Git LFS pointers that match the file types specified with **−−include**, which must be given at least once. Note that **.gitattributes** entries will not be removed, nor will the files; instead, the **export** mode inserts "do not track" entries similar to those created by the **git lfs untrack** command. The **−−remote** option is available in the **export** mode to specify the remote from which Git LFS objects should be fetched if they do not exist in the local Git LFS object cache; if not provided, **origin** is used by default.

The **info** mode (see *INFO*) summarizes by file type (i.e., by filename extension) the total number and size of files in a repository. Note that like the other two modes, by default the **info** mode operates only on the currently checked−out branch and only on commits which do not exist on any remote, so to get a summary

of the entire repository across all branches, use the **−−everything** option. If objects have already been converted to Git LFS pointers, then by default the size of the referenced objects is totaled and reported separately. You may also choose to ignore them by using **−−pointers=ignore** or to treat the pointers as files by using **−−pointers=no−follow**. (The latter option is akin to how existing Git LFS pointers were handled by the **info** mode in prior versions of Git LFS).

When using the **−−everything** option, take note that it means all refs (local and remote) will be considered, but not necessarily all file types. The **import** and **info** modes consider all file types by default, although the **−−include** and **−−exclude** options constrain this behavior. Also note that after importing across all branches with the **−−everything** option (and then checking to ensure the results are satisfactory!) it may be convenient to update multiple branches on your remotes by using the **−−all** option to **git push**.

Unless the **−−skip−fetch** option is given, **git lfs migrate** always begins by fetching updated lists of refs from all the remotes returned by **git remote**, but as noted above, after making changes to your local Git history while converting objects, it will never automatically push those changes to your remotes.

MODES

- **info** Show information about repository size. See *INFO*.
- **import** Convert Git objects to Git LFS pointers. See *IMPORT* and [IMPORT (NO REWRITE)].
- **export** Convert Git LFS pointers to Git objects. See *EXPORT*.

OPTIONS

−I *paths* **−−include=***paths*
: See *INCLUDE AND EXCLUDE*.

−X *paths* **−−exclude=***paths*
: See *INCLUDE AND EXCLUDE*.

−−include−ref=*refname*
: See [INCLUDE AND EXCLUDE (REFS)].

−−exclude−ref=*refname*
: See [INCLUDE AND EXCLUDE (REFS)].

−−skip−fetch
: Assumes that the known set of remote references is complete, and should not be refreshed when determining the set of "un−pushed" commits to migrate. Has no effect when combined with **−−include−ref** or **−−exclude−ref**.

−−everything
: See [INCLUDE AND EXCLUDE (REFS)].

  Note: Git refs are "case−sensitive" on all platforms in "packed from" (see **git−pack−refs(1)**). On "case−insensitive" file systems, e.g. NTFS on Windows or default APFS on macOS, **git−lfs−migrate(1)** would only migrate the first ref if two or more refs are equal except for upper/lower case letters.

−−yes Assume a yes answer to any prompts, permitting noninteractive use. Currently, the only such prompt is the one asking whether to overwrite (destroy) any working copy changes. Thus, specifying this option may cause data loss if you are not careful.

[branch ...]
: Migrate only the set of branches listed. If not given, **git−lfs−migrate(1)** will migrate the currently checked out branch.

  References beginning with ^ will be excluded, whereas branches that do not begin with ^ will be included.

  If any of **−−include−ref** or **−−exclude−ref** are given, the checked out branch will not be appended, but branches given explicitly will be appended.

INFO

The **info** mode summarizes the sizes of file objects present in the Git history. It supports all the core **migrate** options and these additional ones:

- −−**above=<size>** Only count files whose individual filesize is above the given size. **size** may be specified as a number of bytes, or a number followed by a storage unit, e.g., "1b", "20 MB", "3 TiB", etc.

  If a set of files sharing a common extension has no files in that set whose individual size is above the given −−**above** no files no entry for that set will be shown.

- −−**top=<n>** Only display the top **n** entries, ordered by how many total files match the given pathspec. The default is to show only the top 5 entries. When existing Git LFS objects are found, an extra, separate "LFS Objects" line is output in addition to the top **n** entries, unless the −−**pointers** option is used to change this behavior.

- −−**unit=<unit>** Format the number of bytes in each entry as a quantity of the storage unit provided. Valid units include: * b, kib, mib, gib, tib, pib − for IEC storage units * b, kb, mb, gb, tb, pb − for SI storage units

  If a −−**unit** is not specified, the largest unit that can fit the number of counted bytes as a whole number quantity is chosen.

- −−**pointers=[follow|no−follow|ignore]** Treat existing Git LFS pointers in the history according to one of three alternatives. In the default **follow** case, if any pointers are found, an additional separate "LFS Objects" line item is output which summarizes the total number and size of the Git LFS objects referenced by pointers. In the **ignore** case, any pointers are simply ignored, while the **no−follow** case replicates the behavior of the **info** mode in older Git LFS versions and treats any pointers it finds as if they were regular files, so the output totals only include the contents of the pointers, not the contents of the objects to which they refer.

- −−**fixup** Infer −−**include** and −−**exclude** filters on a per−commit basis based on the .gitattributes files in a repository. In practice, this option counts any filepaths which should be tracked by Git LFS according to the repository's .gitattributes file(s), but aren't already pointers. The .gitattributes files are not reported, in contrast to the normal output of the **info** mode. This option is incompatible with explicitly given −−**include**, −−**exclude** filters and with any −−**pointers** setting other than **ignore**, hence −−**fixup** implies −−**pointers=ignore** if it is not explicitly set.

The format of the output shows the filename pattern, the total size of the file objects (excluding those below the −−**above** threshold, if one was defined), and the ratio of the number of files above the threshold to the total number of files; this ratio is also shown as a percentage. For example:

***.gif 93 MB 9480/10504 files(s) 90% *.png 14 MB 1732/1877 files(s) 92%**

By default only the top five entries are shown, but −−**top** allows for more or fewer to be output as desired.

IMPORT

The **import** mode migrates objects present in the Git history to pointer files tracked and stored with Git LFS. It supports all the core **migrate** options and these additional ones:

- −−**verbose** Print the commit oid and filename of migrated files to STDOUT.

- −−**above=<size>** Only migrate files whose individual filesize is above the given size. **size** may be specified as a number of bytes, or a number followed by a storage unit, e.g., "1b", "20 MB", "3 TiB", etc.

- −−**object−map=<path>** Write to **path** a file with the mapping of each rewritten commits. The file format is CSV with this pattern: **OLD−SHA**,**NEW−SHA**

- −−**no−rewrite** Migrate objects to Git LFS in a new commit without rewriting Git history. Please note that when this option is used, the **migrate import** command will expect a different argument list, specialized options will become available, and the core **migrate** options will be ignored. See [IMPORT (NO REWRITE)].

- −−**fixup** Infer −−**include** and −−**exclude** filters on a per−commit basis based on the **.gitattributes** files in a repository. In practice, this option imports any filepaths which should be tracked by Git LFS according to the repository's **.gitattributes** file(s), but aren't already pointers. This option is incompatible with explicitly given −−**include**, −−**exclude** filters.

If −−**no−rewrite** is not provided and −−**include** or −−**exclude** (−**I**, −**X**, respectively) are given, the **.gitattributes** will be modified to include any new filepath patterns as given by those flags.

If −−**no−rewrite** is not provided and neither of those flags are given, the gitattributes will be incrementally modified to include new filepath extensions as they are rewritten in history.

IMPORT (NO REWRITE)

The **import** mode has a special sub−mode enabled by the −−**no−rewrite** flag. This sub−mode will migrate objects to pointers as in the base **import** mode, but will do so in a new commit without rewriting Git history. When using this sub−mode, the base **migrate** options, such as −−**include−ref**, will be ignored, as will those for the base **import** mode. The **migrate** command will also take a different argument list. As a result of these changes, −−**no−rewrite** will only operate on the current branch − any other interested branches must have the generated commit merged in.

The −−**no−rewrite** sub−mode supports the following options and arguments:

- −**m <message>** −−**message=<message>** Specifies a commit message for the newly created commit.
- [file ...] The list of files to import. These files must be tracked by patterns specified in the gitattributes.

If −−**message** is given, the new commit will be created with the provided message. If no message is given, a commit message will be generated based on the file arguments.

EXPORT

The **export** mode migrates Git LFS pointer files present in the Git history out of Git LFS, converting them into their corresponding object files. It supports all the core **migrate** options and these additional ones:

- −−**verbose** Print the commit oid and filename of migrated files to STDOUT.
- −−**object−map=<path>** Write to **path** a file with the mapping of each rewritten commit. The file format is CSV with this pattern: **OLD−SHA**,**NEW−SHA**
- −−**remote=<git−remote>** Download LFS objects from the provided **git−remote** during the export. If not provided, defaults to **origin**.

The **export** mode requires at minimum a pattern provided with the −−**include** argument to specify which files to export. Files matching the −−**include** patterns will be removed from Git LFS, while files matching the −−**exclude** patterns will retain their Git LFS status. The export command will modify the **.gitattributes** to set/unset any filepath patterns as given by those flags.

INCLUDE AND EXCLUDE

You can specify that **git lfs migrate** should only convert files whose pathspec matches the −−**include** glob patterns and does not match the −−**exclude** glob patterns, either to reduce total migration time or to only migrate part of your repo. Multiple patterns may be given using commas as delimiters.

Pattern matching is done so as to be functionally equivalent to the pattern matching format of **.gitattributes**. In addition to simple file extension matches (e.g., ***.gif**) patterns may also specify directory paths, in which case the **path/**** format may be used to match recursively.

INCLUDE AND EXCLUDE (REFS)

You can specify that **git lfs migrate** should only convert files added in commits reachable from certain references, namely those defined using one or more −−**include−ref** options, and should ignore files in commits reachable from references defined in −−**exclude−ref** options.

D−−−E−−−F / \ A−−−B−−−−−−C refs/heads/my−feature \ \ \ refs/heads/main \ refs/remotes/origin/main

In the above configuration, the following commits are reachable by each ref:

refs/heads/main: C, B, A refs/heads/my−feature: F, E, D, B, A refs/remote/origin/main: A

The following **git lfs migrate** options would, therefore, include commits F, E, D, C, and B, but exclude commit A:

−−include−ref=refs/heads/my−feature −−include−ref=refs/heads/main −−exclude−ref=refs/remotes/origin/main

The presence of flag **−−everything** indicates that all local and remote references should be migrated.

EXAMPLES

Migrate unpushed commits

A common use case for the migrate command is to convert large Git objects to LFS before pushing your commits. By default, it only scans commits that don't exist on any remote, so long as the repository is non−bare.

First, run **git lfs migrate info** to list the file types taking up the most space in your repository:

$ git lfs migrate info migrate: Fetching remote refs: ..., done migrate: Sorting commits: ..., done migrate: Examining commits: 100% (1/1), done *.mp3 284 MB 1/1 files(s) 100% *.pdf 42 MB 8/8 files(s) 100% *.psd 9.8 MB 15/15 files(s) 100% *.ipynb 6.9 MB 6/6 files(s) 100% *.csv 5.8 MB 2/2 files(s) 100%

Now, you can run **git lfs migrate import** to convert some file types to LFS:

$ git lfs migrate import −−include="*.mp3,*.psd" migrate: Fetching remote refs: ..., done migrate: Sorting commits: ..., done migrate: Rewriting commits: 100% (1/1), done main d2b959babd099fe70da1c1512e2475e8a24de163 −> 136e706bf1ae79643915c134e17a6c933fd53c61 migrate: Updating refs: ..., done

If after conversion you find that some files in your working directory have been replaced with Git LFS pointers, this is normal, and the working copies of these files can be repopulated with their full expected contents by using **git lfs checkout**.

Migrate local history

You can also migrate the entire history of your repository:

‘‘‘ # Check for large files and existing Git LFS objects in your local main branch $ git lfs migrate info −−include−ref=main$ git lfs migrate info −−everything# and listing the top 100 or fewer results $ git lfs migrate info −−everything −−pointers=ignore −−top=100 ‘‘‘

The same flags will work in **import** mode:

‘‘‘ # Convert all zip files in your main branch $ git lfs migrate import −−include−ref=main −−include="*.zip"$ git lfs migrate import −−everything −−include="*.zip"$ git lfs migrate import −−everything −−above=100Kb ‘‘‘

Note: This will require a force−push to any existing Git remotes. Using the **−−all** option when force−pushing may be convenient if many refs were updated, e.g., after importing to Git LFS with the **−−everything** option.

Migrate without rewriting local history

You can also migrate files without modifying the existing history of your repository. Note that in the examples below, files in subdirectories are not included because they are not explicitly specified.

Without a specified commit message:

$ git lfs migrate import −−no−rewrite test.zip *.mp3 *.psd

With a specified commit message:

$ git lfs migrate import −−no−rewrite \ −m "Import test.zip, .mp3, .psd files in root of repo" \ test.zip

***.mp3 *.psd**

SEE ALSO

git−lfs−checkout(1), git−lfs−track(1), git−lfs−untrack(1), gitattributes(5).

Part of the git−lfs(1) suite.