Git useful command :

Load in local a remote branch:

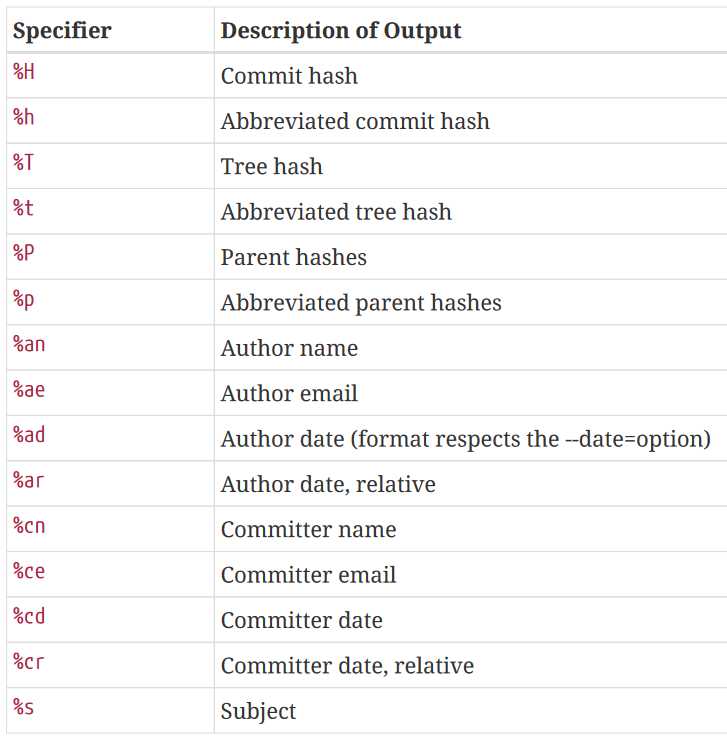
git checkout –track name\_of\_the\_remote\_branch # create a tracking branch

History :

git log --patch -2 🡪 (--patch or -p) Display the differences resulting of committed changes. -2 restrict the displaying of logs to 2.

git log --stat 🡪 summarize the above command line in number of insertions/deletions by file

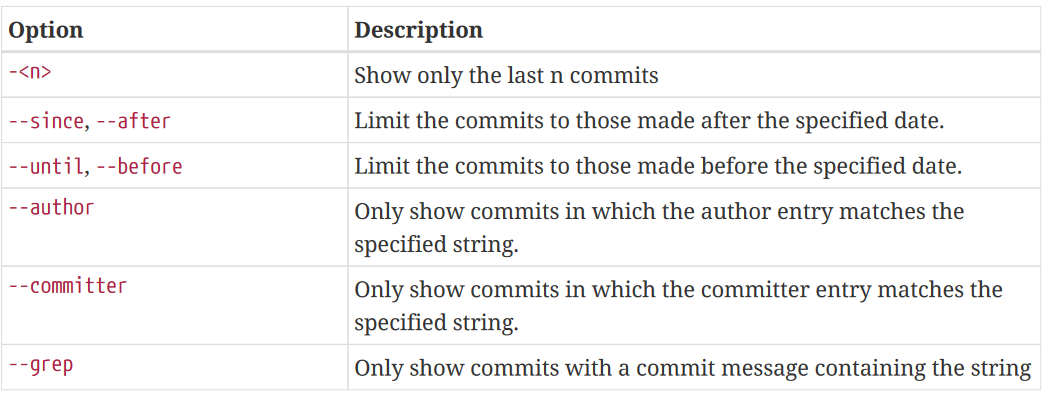
git log --pretty=format:”%h - %an %ad %s” 🡪 to get the some information here the references



git log --pretty=format :“%h %s” --graph 🡪 generate branch diagrams

git log -- *path* 🡪 display filtered log inside *path*

git log with the below arguments





## Compare files from local and remote branch

git fetch remote\_branch

git diff local\_branch remote\_branch – file

## Compare file unstaged file against the last committed files

git difftool HEAD --file\_path

Compare 2 distinct files

git difftool HEAD: path\_file\_1 path\_file\_2

## Observer unstaged files

git ls-files -m ([ref](https://stackoverflow.com/questions/10018533/is-it-possible-to-git-status-only-modified-files))

## Read the version of a given commit/branch :

git rev-parse branch\_name # get the hash code of the branch/comit

git cat-file -p <sha1>:./file.tex > wherever.tex ([ref](https://stackoverflow.com/questions/14995506/how-to-get-a-copy-of-an-older-version-of-a-file-in-a-git-repository))

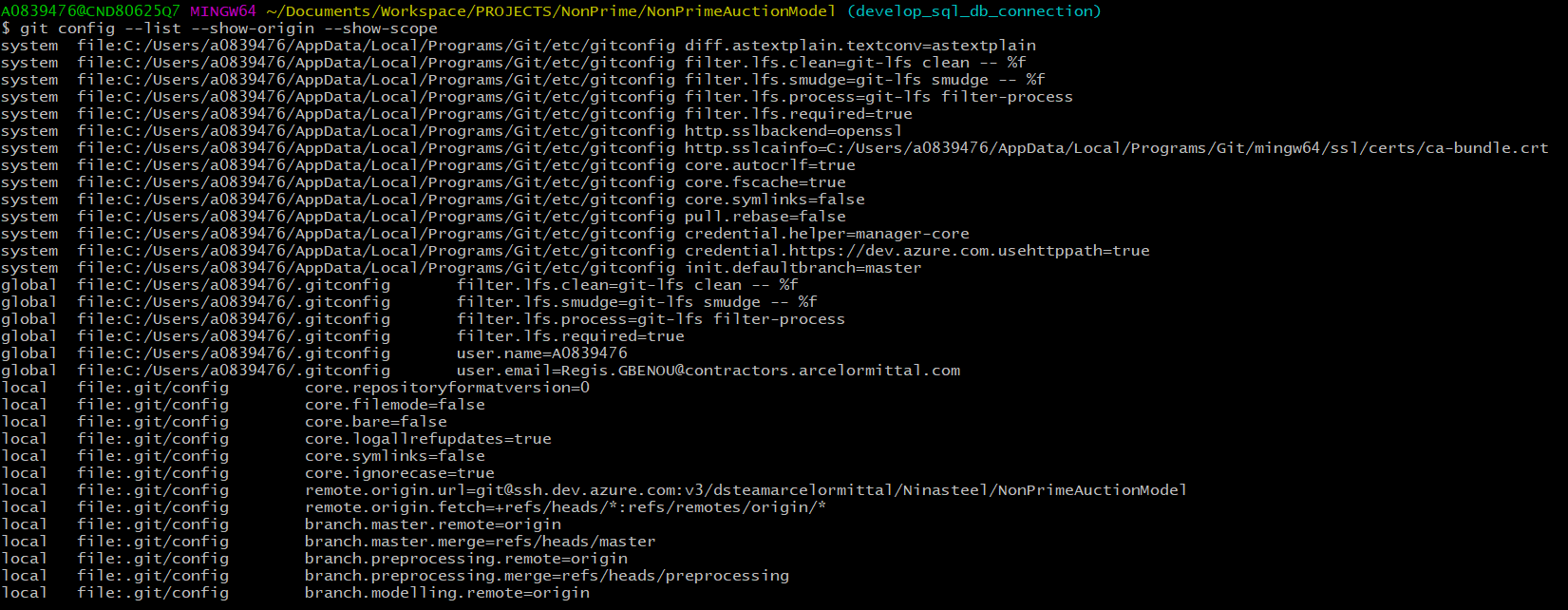
## List of modified files between 2 branches/commits

git diff --name-only SHA1 SHA2 ([ref](https://stackoverflow.com/questions/1552340/how-to-list-only-the-names-of-files-that-changed-between-two-commits))

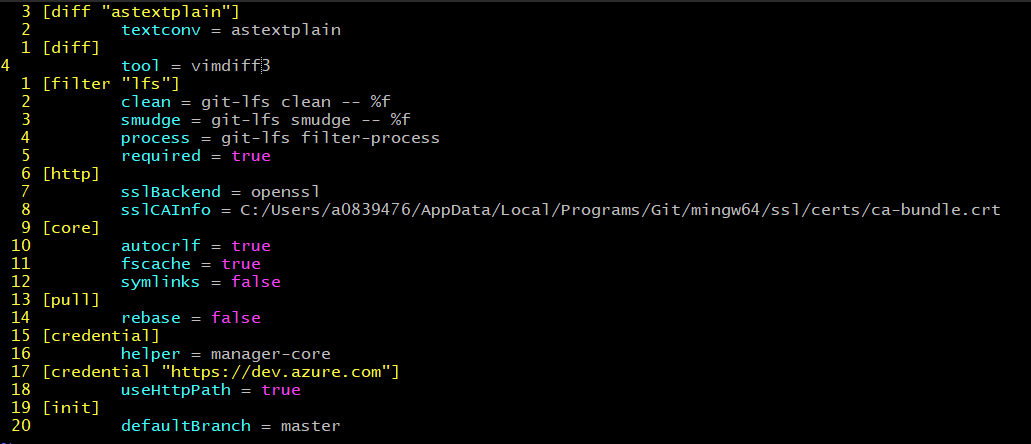
Fixing merge conflicts

([ref](https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/addressing-merge-conflicts/resolving-a-merge-conflict-using-the-command-line))

Know where config elements are stored:



Choose the git diff tool



Use difftool

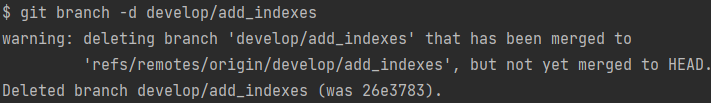


Remove added file before committing

git reset <file\_name>

Remove branch

git branch -d branch\_to\_delete # remove branch only if it is fully merged



Means that the we deleted locally develop/add\_indexees but we did not delete the origin version

git push origin –delete branch\_name

Rename Branch

git branch -m new\_name # on the branch that we want to rename

git branch -m old\_name new\_name # from another branch that the one to rename

git push origin :old\_name new\_name # delete the old\_name remote branch an push new\_name

git push origin -u new\_name # reset the upstream branch for the new\_name local branch

Upload a given branch

git push –set-upstream origin dev

Stashing

git stash

git stash list # to access to the stash list

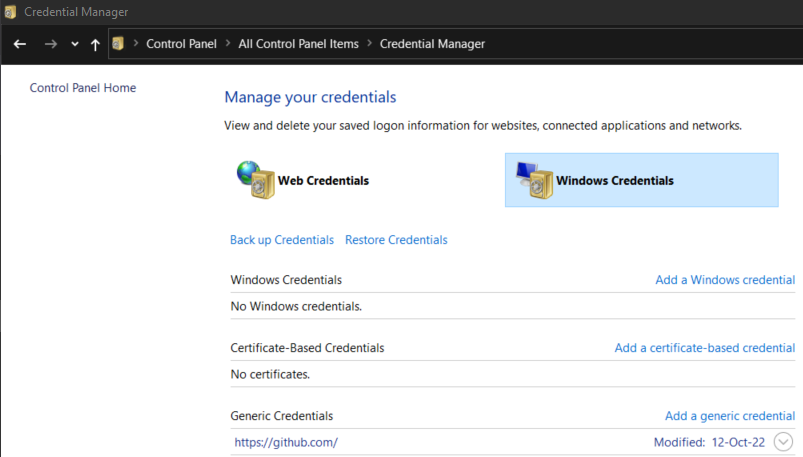
git stash apply # apply the more recent stash (stash@{0})

git stash apply stash@{2} # apply the third more recent stash

git stash drop stash@{1} # remove the second more recent stash

Git Credential

Windows:



Then choose a Generic Credential and edit it then enter Token