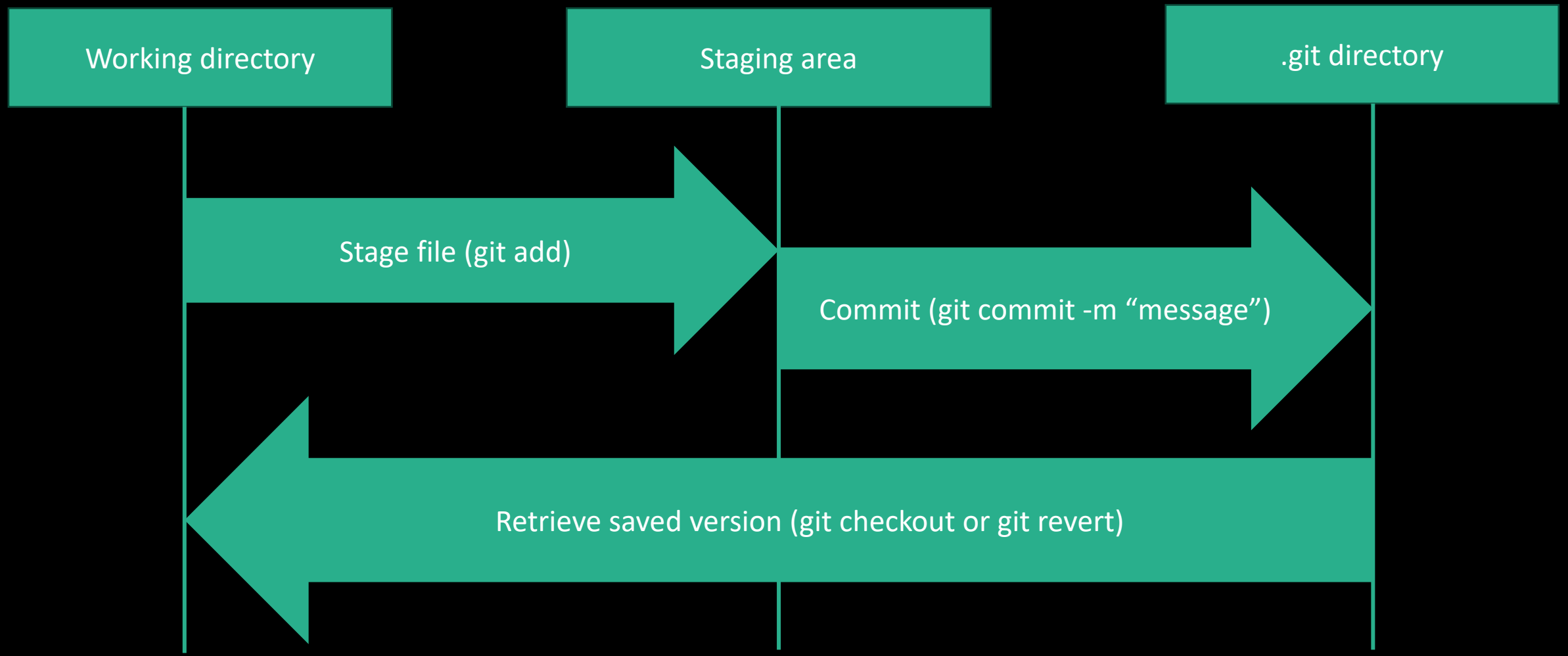
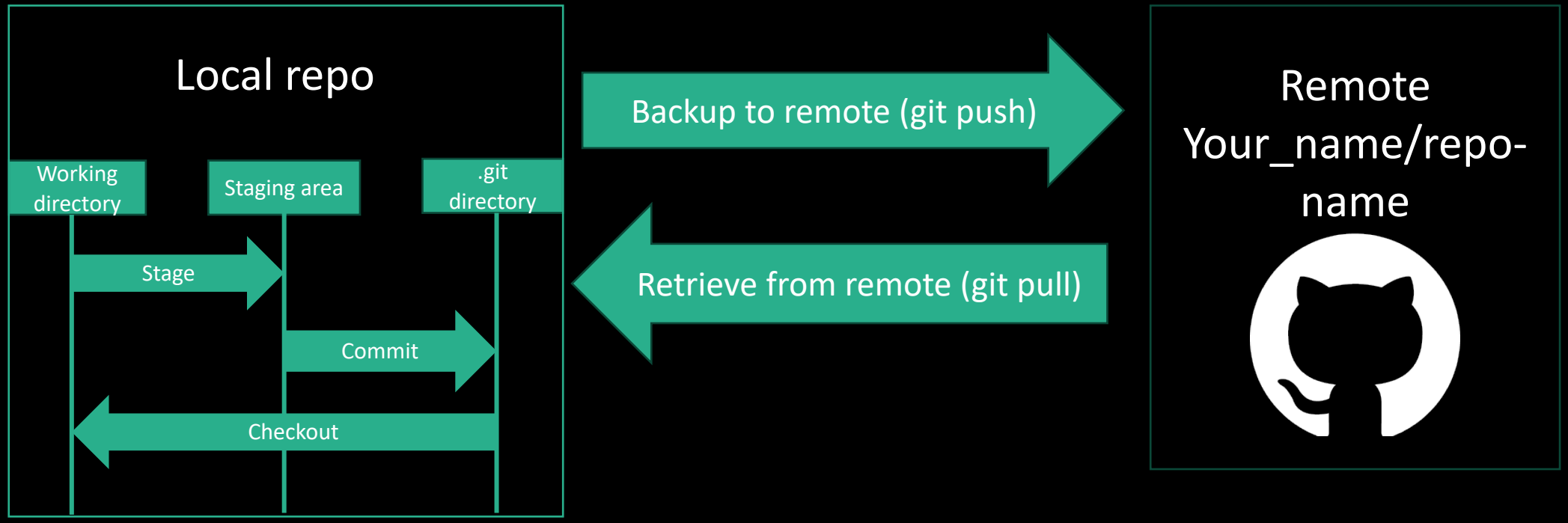
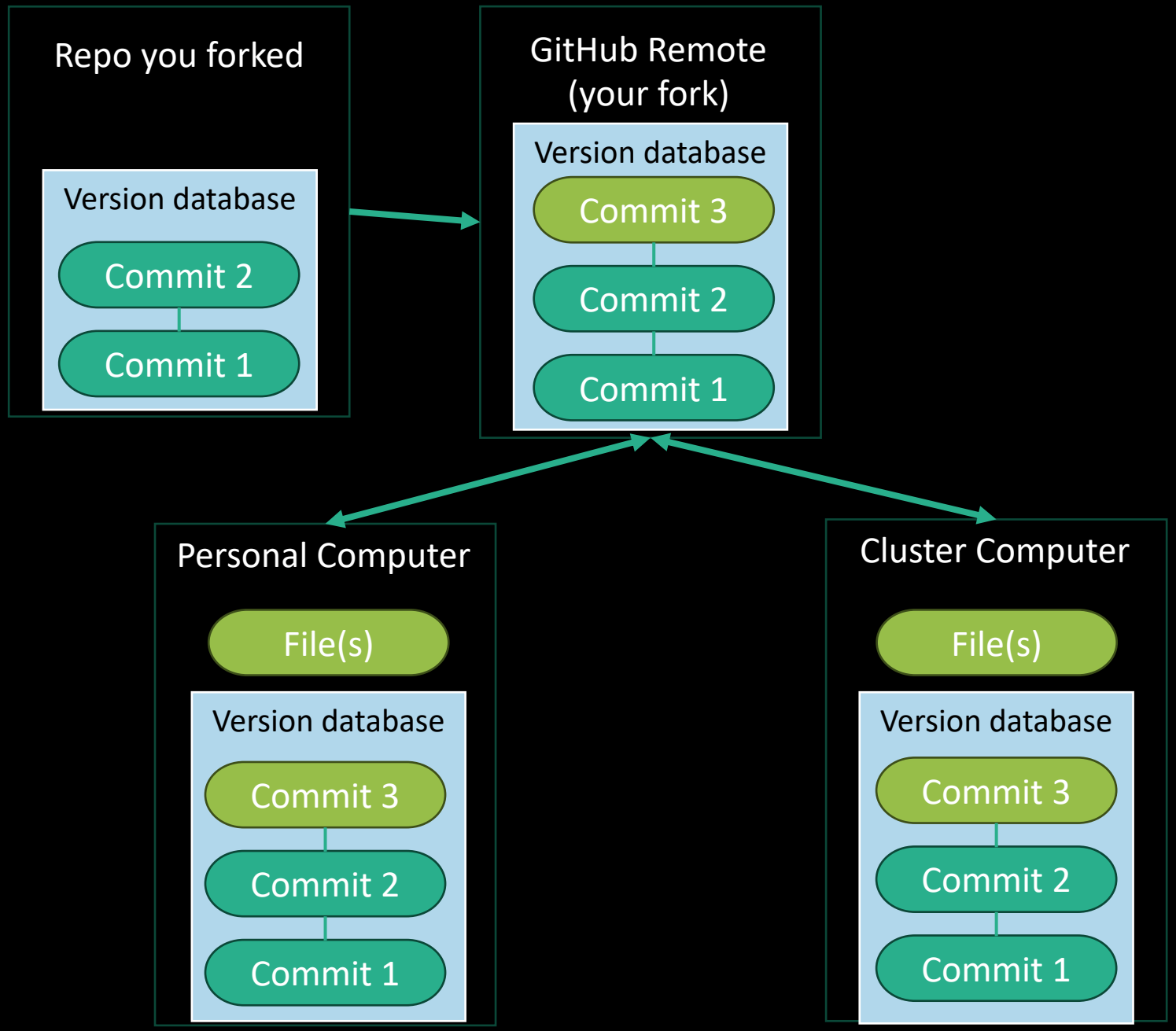
Healthy Habits for Data Science

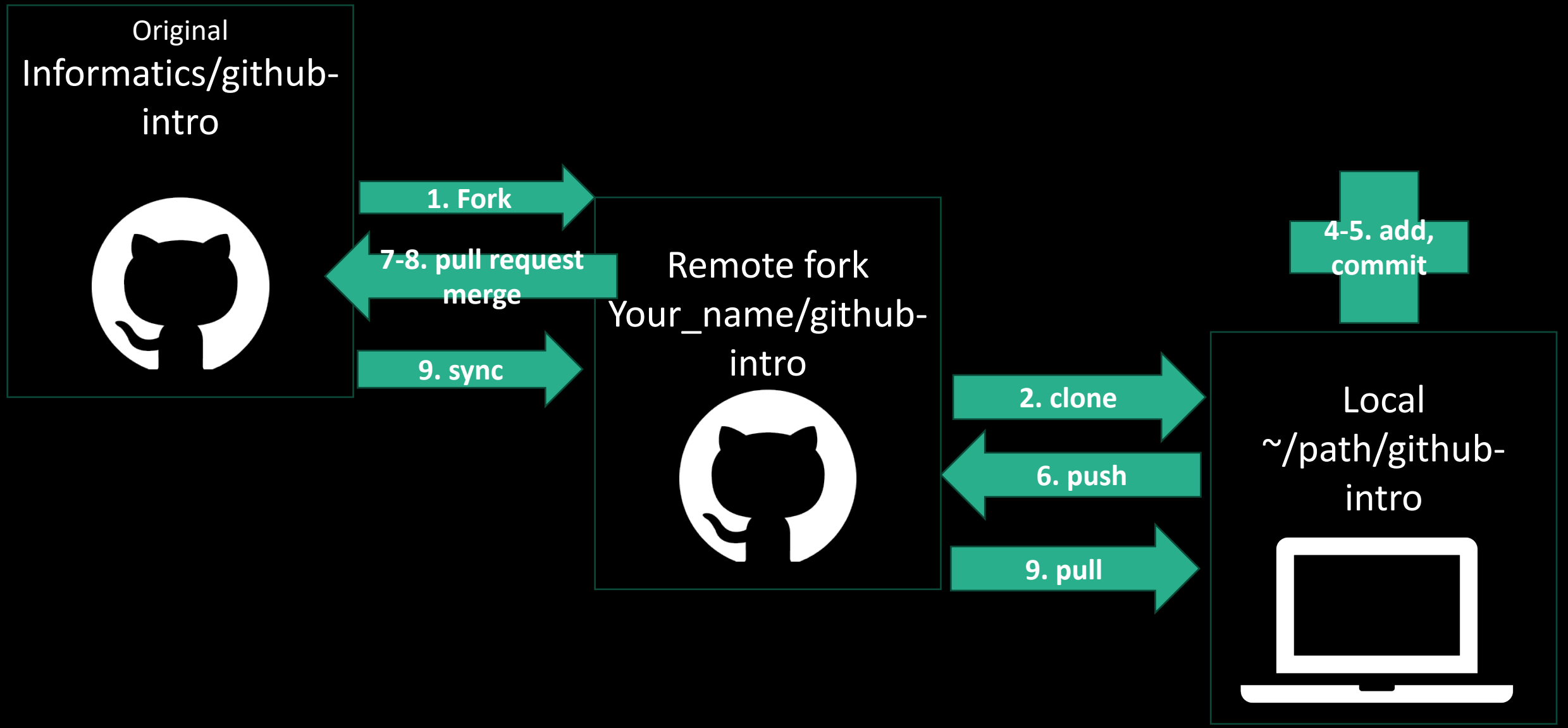
Day 3: Version control with git/GitHub

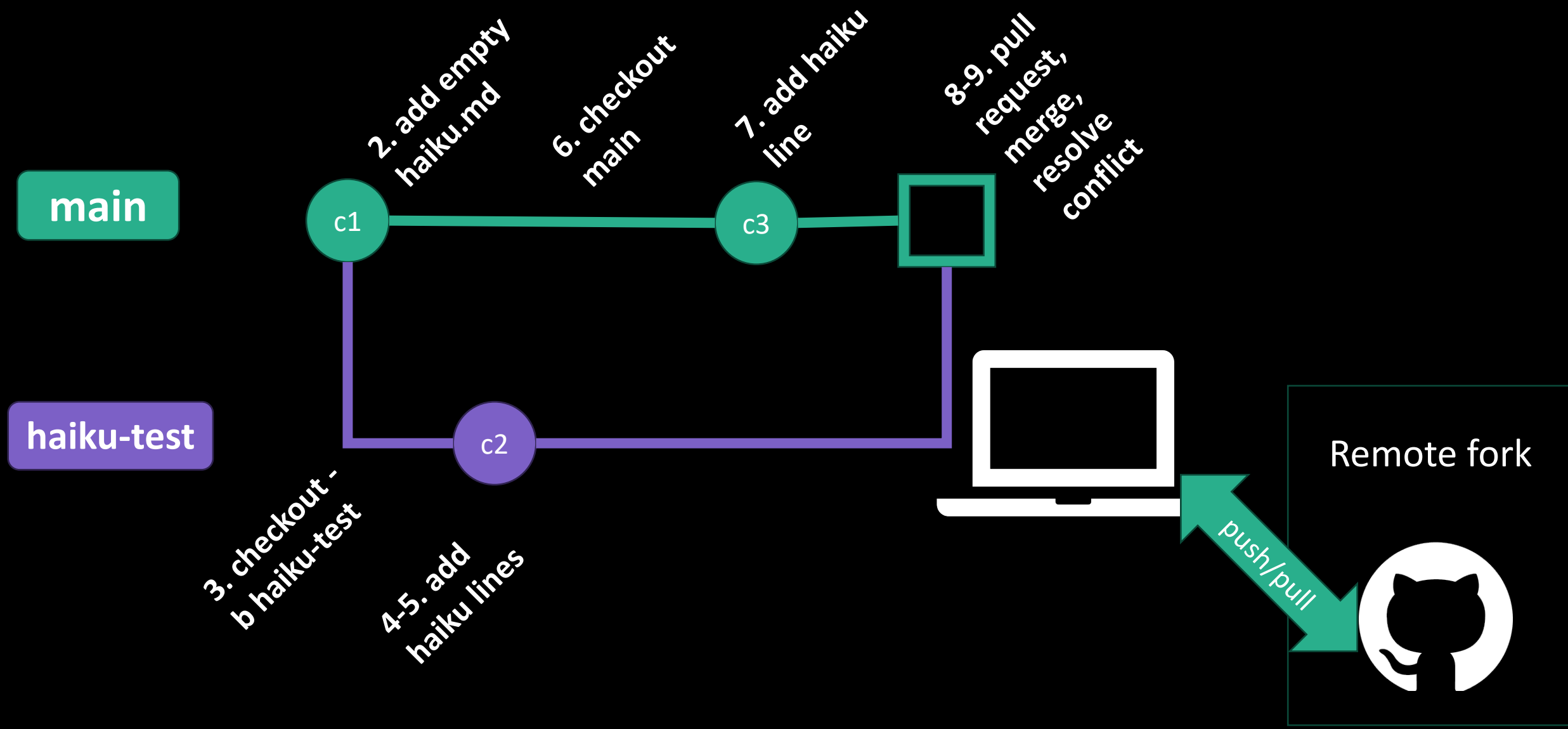

Working directory
Staging area
.git directory
Stage file (git add)
Commit (git commit -m “message”)
Retrieve saved version (git checkout or git revert)


Local repo
Working directory
Staging area
.git directory
Stage
Commit
Checkout
Backup to remote (git push)
Retrieve from remote (git pull)
Remote
Your_name/repo-name


Repo you forked
Version database
Commit 2
Commit 1
GitHub Remote
(your fork)
Version database
Commit 3
Commit 2
Commit 1
Personal Computer
File(s)
Version database
Commit 3
Commit 2
Commit 1
Cluster Computer
File(s)
Version database
Commit 3
Commit 2
Commit 1

https://github.com/harvardinformatics/github-intro


Original
Informatics/github-intro
1. Fork
7-8. pull request merge
9. sync
Remote fork
Your_name/github-intro
2. clone
6. push
9. pull
4-5. add, commit
Local
~/path/github-intro


main
haiku-test
c1
c2
c3
2. add empty haiku.md
6. checkout main
7. add haiku line
8-9. pull request, merge, resolve conflict
3. checkout -b haiku-test
4-5. add haiku lines
push/pull
Remote fork

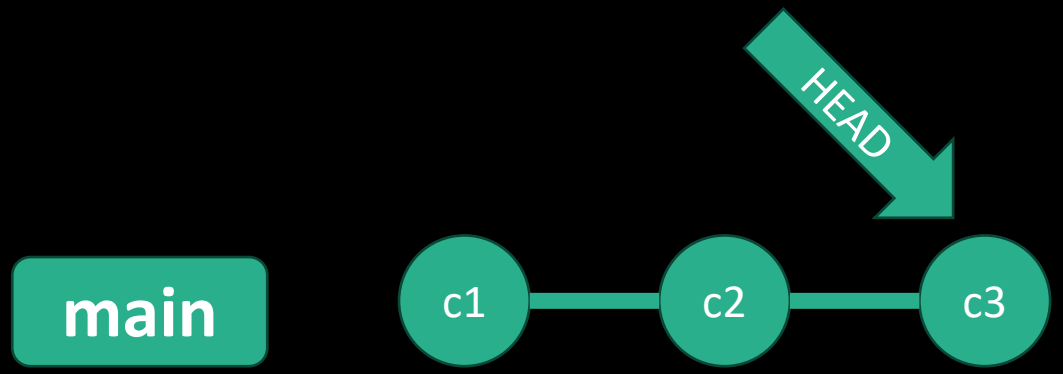
Term	Definition
repository (repo)	a folder tracked by git
fork	copy someone else's GitHub repo into your own account
clone	copy a repo from a remote onto your local computer
add	add file to staging area (start tracking it)
commit	make a snapshot of staging area to repository
push	upload changes to remote repo
remote	a repository hosted on a server (e.g. GitHub)
staging area	intermediate area where files are tracked before they are added to git
pull	fetch changes from a remote and merge into existing branch
branch	an isolated development path that was diverted from the main line at a specific commit
status	check the status of your git repo

Here's what you should use git for:

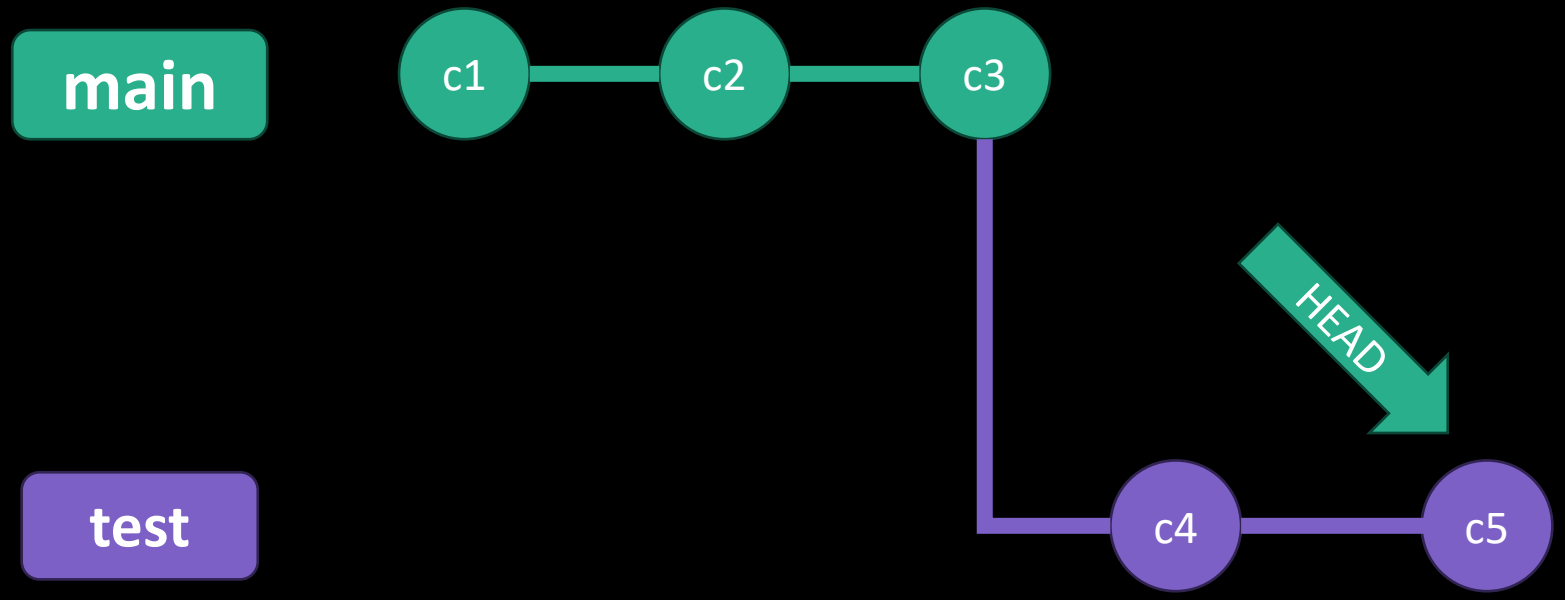
- Scripts
- Notebooks
- READMEs
- Configuration files (yamls, config files)

Here's what you should NOT use git for:

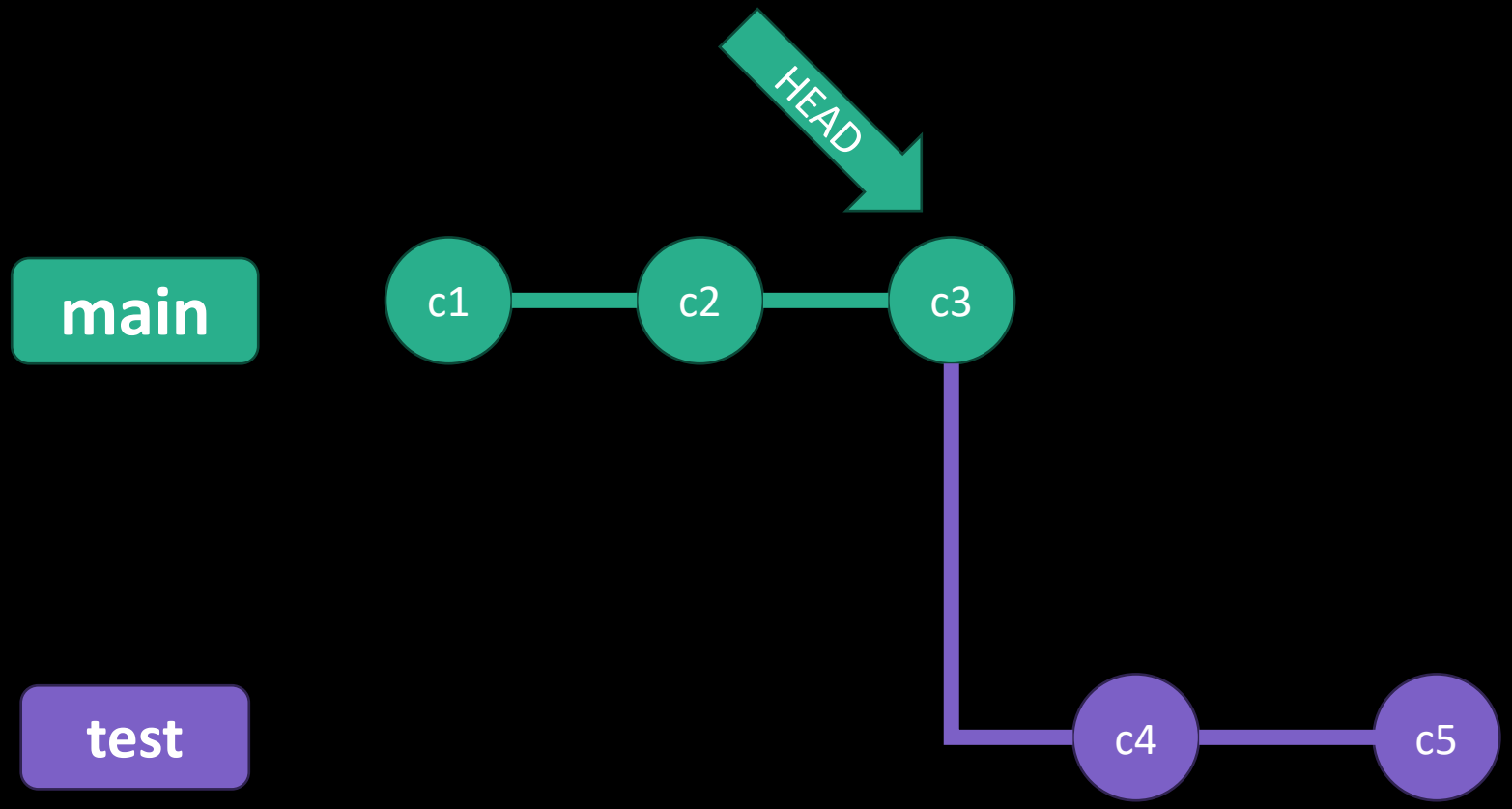
- Data
- Software (conda envs, containers)
- Non-plaintext files (pdfs, images, videos, binary files)
- Large files
- Passwords and sensitive information (!!!)
- Log files (slurm logs, error logs, etc)
- Junk files (temp, OS-generated junk)



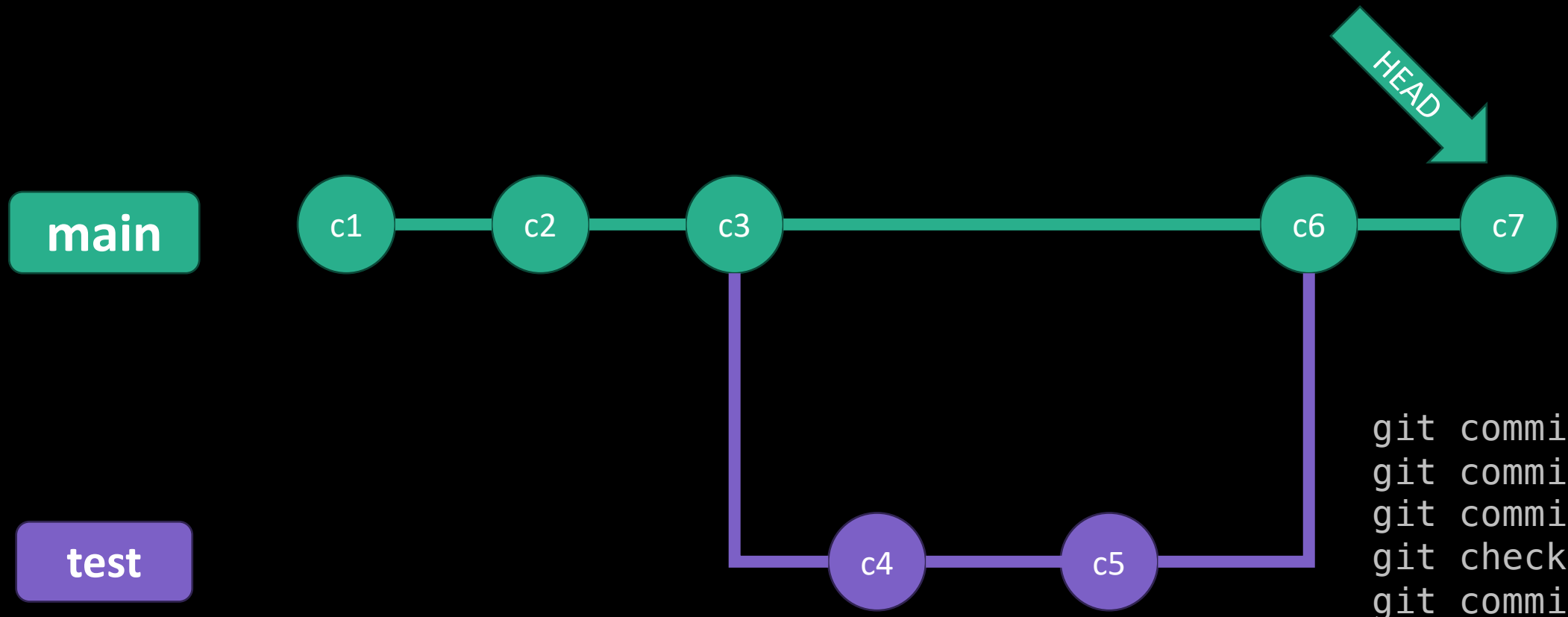

```
git commit
git commit
git commit
```




```
git commit
git commit
git commit
git checkout -b test
git commit
git commit
```




```
git commit
git commit
git commit
git checkout -b test
git commit
git commit
git checkout main
```

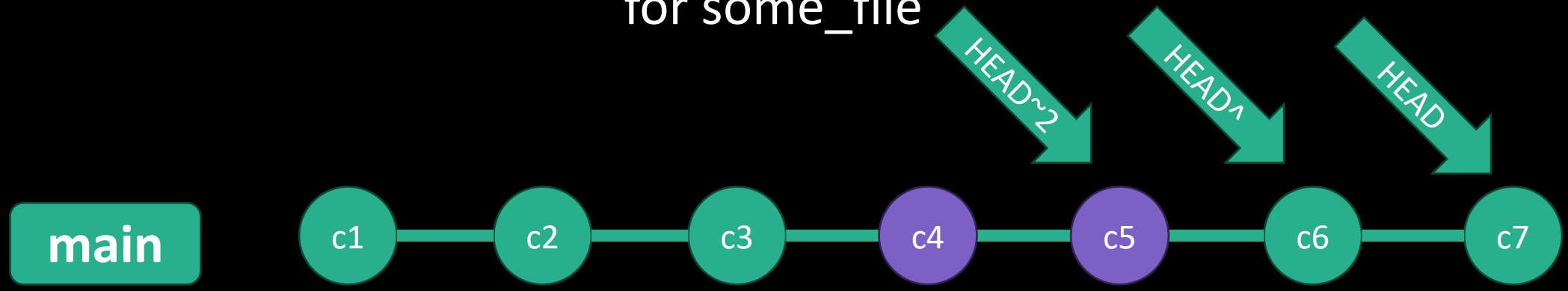



```
git commit
git commit
git commit
git checkout -b test
git commit
git commit
git checkout main
git merge test
git commit
```

git checkout

1. git checkout -b new-branch
2. git checkout d510fh01
3. git checkout HEAD^
4. git checkout main
5. git checkout d510fh01 some_file

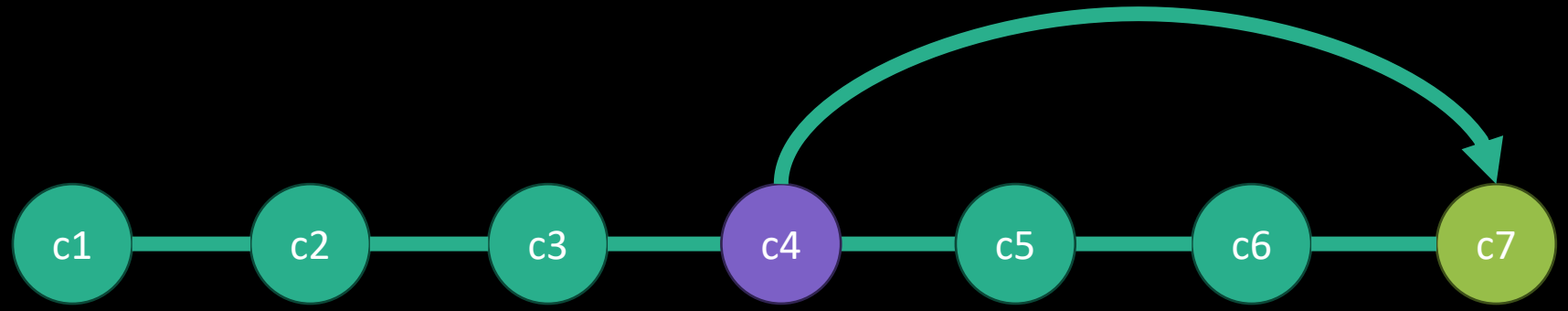
1. make a new branch and point HEAD to it
2. point head to commit d510fh01
3. points head to previous commit
4. points head to latest commit of branch main
5. points head to commit d510fh01 but only for some_file



git revert

1. git revert d510fh01
2. git revert HEAD

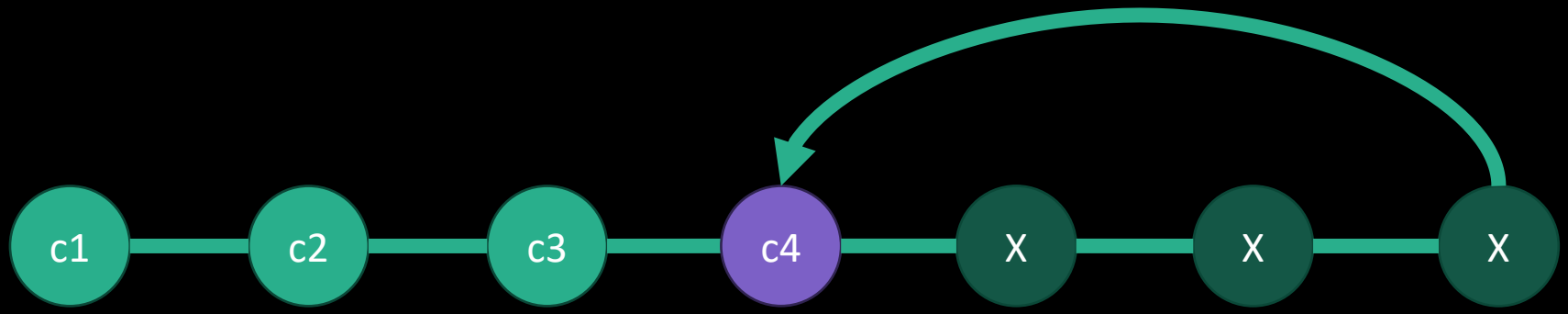
1. make a new commit that reverts commit d510fh01
2. reverts the latest commit



git reset

1. git reset d510fh01
2. git reset HEAD
3. git reset --hard d510fh01

1. undo commits since d510fh01, does not change your working directory
2. unstage everything to previous commit
3. undo commits since d510fh01, including your working directory



advanced git vocab/concepts

Term	Definition
HEAD	Pointer to current location in your repository, typicall latest commit
checkout	Switch your working directory to a different commit or branch, by moving the HEAD
revert	Undo a specific commit and make a new commit
reset	Rewrite commit history since a specific commit
diff	See differences between staged/commited and working directory file
log	see all commit history
reflog	see all git command history (that made changes)

Main git takeaways

- Commits are NOT automatic, commit often
- Always pull and keep your local up to date with your remote
- NEVER add sensitive information to git
- Everything is recorded and almost everything can be undone, so don’t be afraid
- Do your development on a separate branch