Terminal Commands:

1) pwd : shows current working directory
2) ls : shows all the folders and files in the current directory
3) cd archive : (move forward in a folder)
4) git –version : returns git version

5) **Creating a repo**:
   i) git init <name>
   iI) cd <name>
   iii) git status (this shows the modified and untracked file)
6) **Converting a project into a new repo:**
   i) git init
   ii) git status (this shows the modified and untracked file)

[warning : don’t create repo inside another repo, it creates confusions, then there will be 2 .git directories]

7) The Git Workflow

1. **Edit and save files** on your computer.
   - Make changes to your project files as needed.

2. **Add the file(s) to the Git staging area**
   - This step tracks the modifications you’ve made.
   - Use `git add <file>` or `git add .` to stage changes.

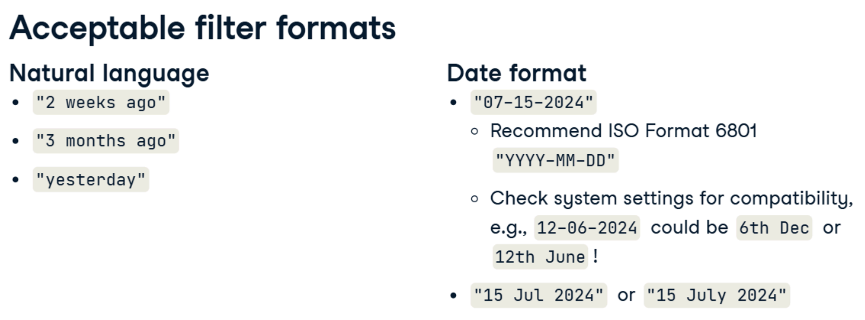
3. **Commit the files**
   - Git takes a snapshot of the staged files at that point in time.
   - Use `git commit -m "Your commit message"` to save the changes.
   - This allows you to compare versions or revert files if needed.

4) USE git push origin main/master
   This uploads your commit to the `main` branch on GitHub.
 (Use the correct branch name if different, like `master` or `dev`.)

8) Git Version History:

i) git log

ii) git log -3 : it will return most recent 3
iii) git log readme.md : look at the specific file commits
iv) git log -3 mental-health.csv : combining both
v) git log –since=’Month Day Year’
vi) git log –since=’Month Day Year’ –until=’Month Day Year’



Acceptable filter formats

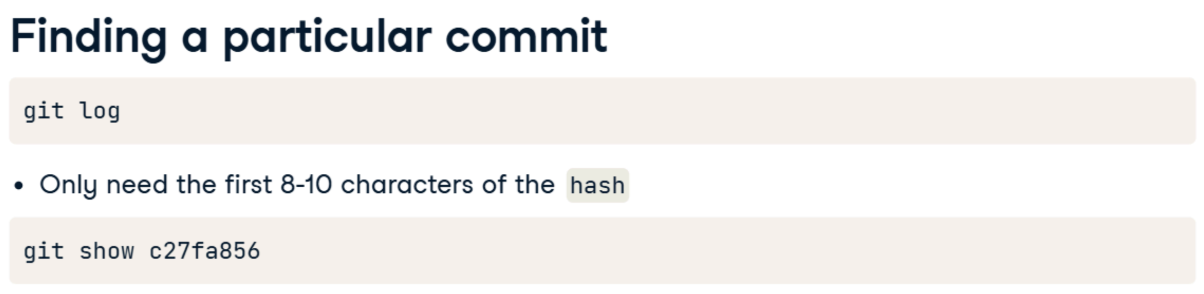
Natural language

- `"2 weeks ago"`
- `"3 months ago"`
- `"yesterday"`

Date format

- `"07-15-2024"`
  - Recommend ISO Format 6801 `"YYYY-MM-DD"`
  - Check system settings for compatibility, e.g., `12-06-2024` could be `6th Dec` or `12th June` !
- `"15 Jul 2024"` or `"15 July 2024"`

vii)



Finding a particular commit

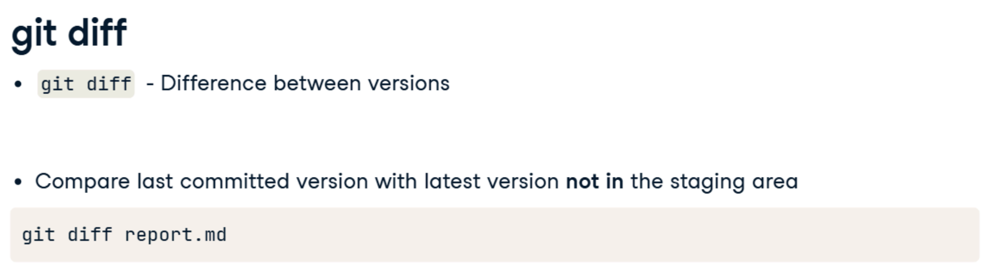
```
git log
```

- Only need the first 8-10 characters of the `hash`

```
git show c27fa856
```

9) Comparing Versions:

i)



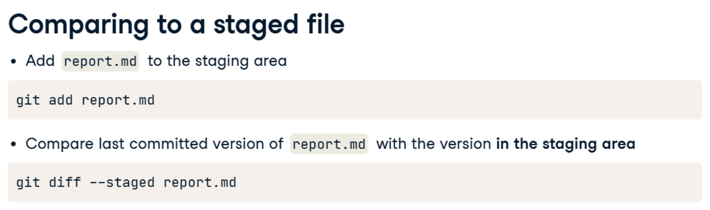
git diff

- `git diff` - Difference between versions

- Compare last committed version with latest version **not in** the staging area

```
git diff report.md
```

ii)



Comparing to a staged file

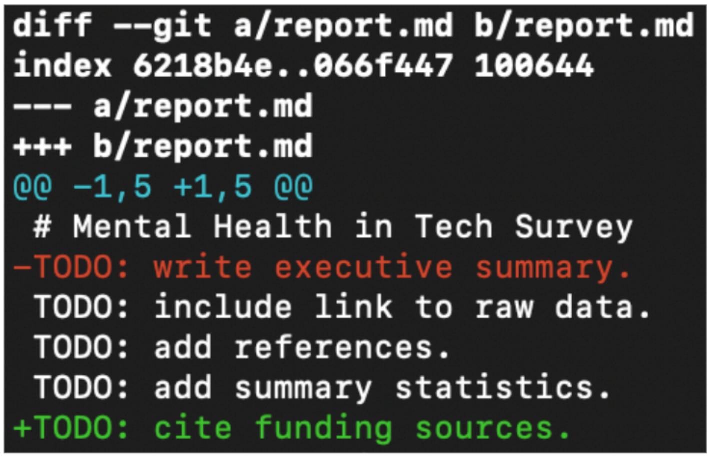
- Add `report.md` to the staging area

```
git add report.md
```

- Compare last committed version of `report.md` with the version **in the staging area**

```
git diff --staged report.md
```

Comparing to a staged file



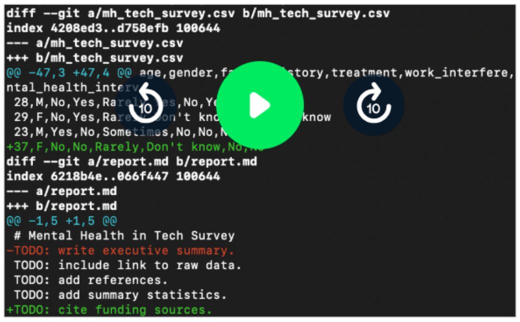
```
diff --git a/report.md b/report.md
index 6218b4e..066f447 100644
--- a/report.md
+++ b/report.md
@@ -1,5 +1,5 @@
 # Mental Health in Tech Survey
-TODO: write executive summary.
 TODO: include link to raw data.
 TODO: add references.
 TODO: add summary statistics.
+TODO: cite funding sources.
```

iii)

Comparing multiple staged files

- Compare **all staged files** to versions in the last commit

```
git diff --staged
```



```
diff --git a/mh_tech_survey.csv b/mh_tech_survey.csv
index 4208ed3..d758efb 100644
--- a/mh_tech_survey.csv
+++ b/mh_tech_survey.csv
@@ -47,3 +47,4 @@ age,gender,f[illegible]story,treatment,work_interfere,
ntal_health_interv[illegible]
 28,M,No,Yes,Rarely[illegible]es,No,Ye[illegible]
 29,F,No,Yes,Rare[illegible]on't kno[illegible]know
 23,M,Yes,No,Sometimes,No,No,N[illegible]
+37,F,No,No,Rarely,Don't know,No,[illegible]
diff --git a/report.md b/report.md
index 6218b4e..066f447 100644
--- a/report.md
+++ b/report.md
@@ -1,5 +1,5 @@
 # Mental Health in Tech Survey
-TODO: write executive summary.
 TODO: include link to raw data.
 TODO: add references.
 TODO: add summary statistics.
+TODO: cite funding sources.
```

iv)

Comparing two commits

- Find the commit hashes

```
git log
```

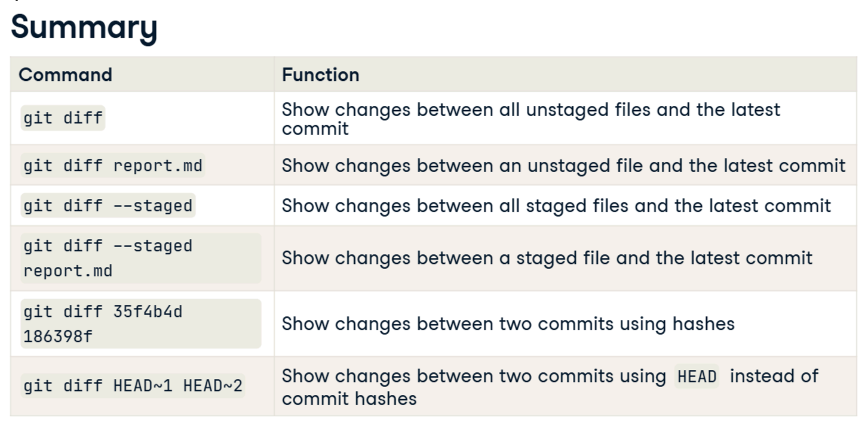
- Compare them

```
git diff 35f4b4d 186398f
```

- What changed **from** first hash **to** second hash
  - Put most recent hash second
- State in latest commit = `HEAD`
- Compare second most recent with the most recent commit

```
git diff HEAD~1 HEAD
```

v)

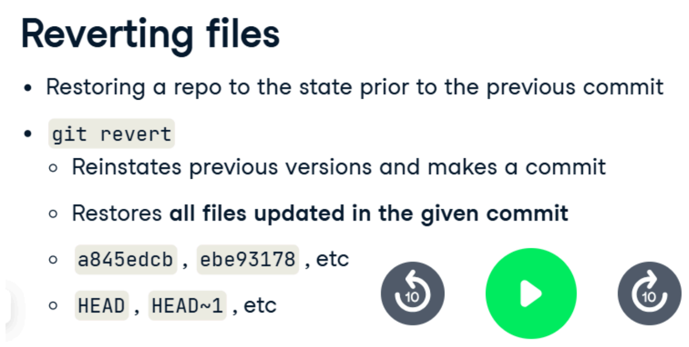


Summary

Command	Function
`git diff`	Show changes between all unstaged files and the latest commit
`git diff report.md`	Show changes between an unstaged file and the latest commit
`git diff --staged`	Show changes between all staged files and the latest commit
`git diff --staged report.md`	Show changes between a staged file and the latest commit
`git diff 35f4b4d 186398f`	Show changes between two commits using hashes
`git diff HEAD~1 HEAD~2`	Show changes between two commits using `HEAD` instead of commit hashes

9) Gir REVERT:

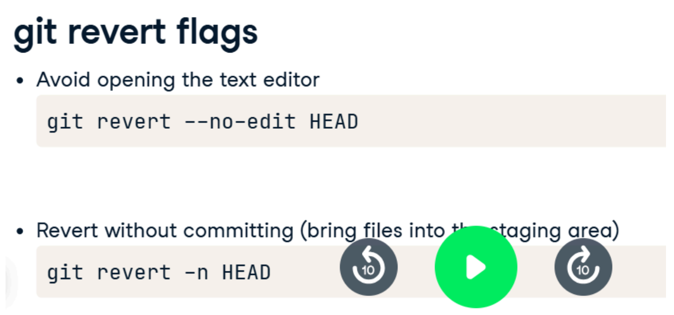
i)



Reverting files

- Restoring a repo to the state prior to the previous commit
- `git revert`
  - Reinstates previous versions and makes a commit
  - Restores **all files updated in the given commit**
  - `a845edcb` , `ebe93178` , etc
  - `HEAD` , `HEAD~1` , etc

ii)



git revert flags

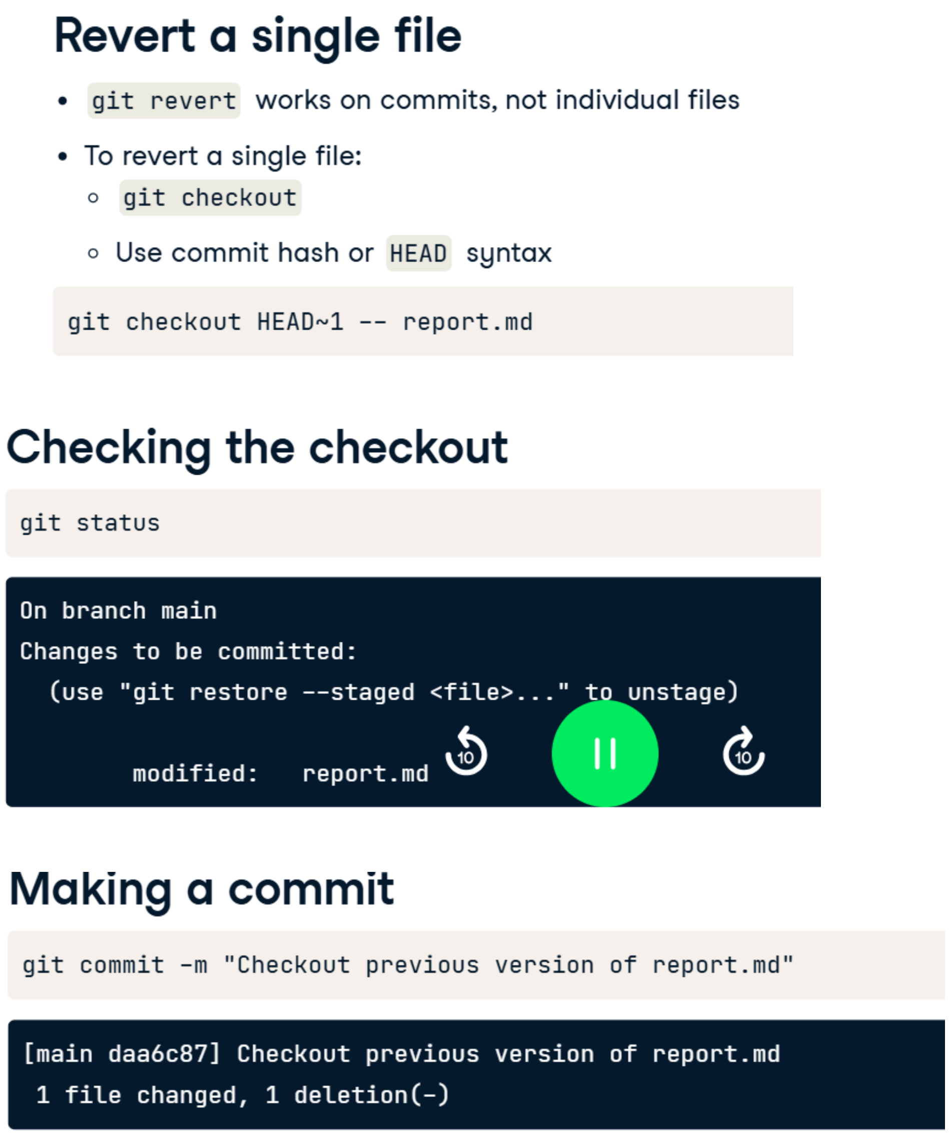
- Avoid opening the text editor

```
git revert --no-edit HEAD
```

- Revert without committing (bring files into th staging area)

```
git revert -n HEAD
```

iii)



Revert a single file

- `git revert` works on commits, not individual files
- To revert a single file:
  - `git checkout`
  - Use commit hash or `HEAD` syntax

```
git checkout HEAD~1 -- report.md
```

Checking the checkout

```
git status
```

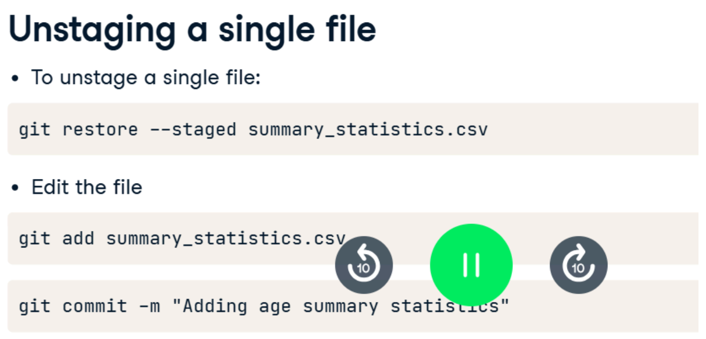
```
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)

        modified:   report.md
```

Making a commit

```
git commit -m "Checkout previous version of report.md"
```

```
[main daa6c87] Checkout previous version of report.md
 1 file changed, 1 deletion(-)
```

iv) Unstaging a file



Unstaging a single file

- To unstage a single file:

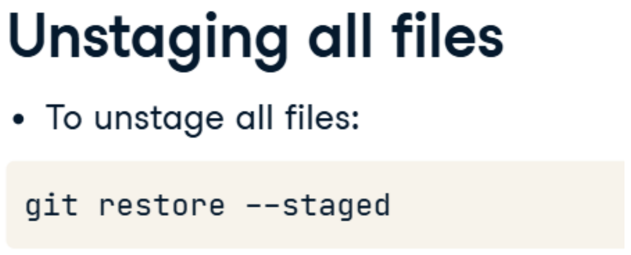
```
git restore --staged summary_statistics.csv
```

- Edit the file

```
git add summary_statistics.csv
```

```
git commit -m "Adding age summary statistics"
```

v) Unstaging all files

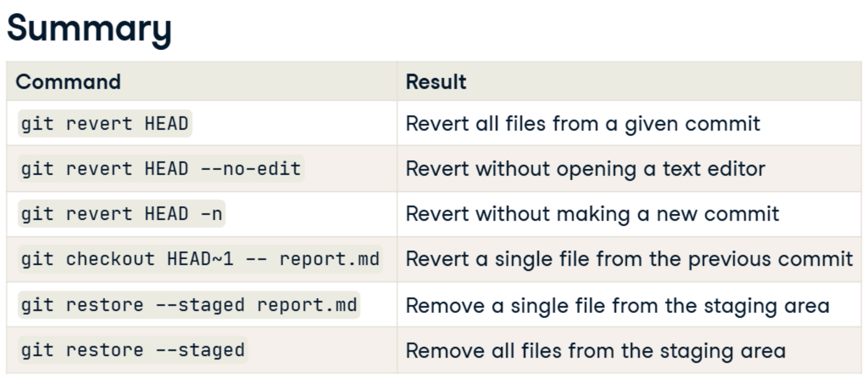


Unstaging all files

- To unstage all files:

```
git restore --staged
```

vi) Summary



Summary

Command	Result
`git revert HEAD`	Revert all files from a given commit
`git revert HEAD --no-edit`	Revert without opening a text editor
`git revert HEAD -n`	Revert without making a new commit
`git checkout HEAD~1 -- report.md`	Revert a single file from the previous commit
`git restore --staged report.md`	Remove a single file from the staging area
`git restore --staged`	Remove all files from the staging area