Introduction to version control with Git

INTRODUCTION TO VERSION CONTROL WITH GIT

George Boorman
Curriculum Manager, DataCamp

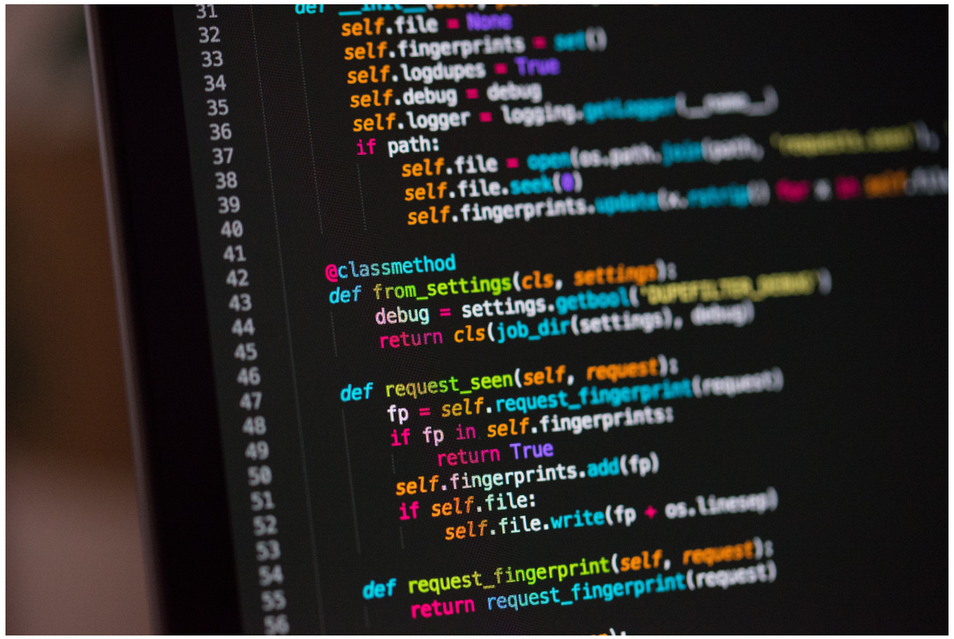
datacamp

What is a version?

1. Contents of a file at a given point in time
2. Metadata (information associated with the file):
   - The author of the file
   - Where it is located
   - The file type
   - When it was last saved

What is version control?

- Version control is a group of systems and processes
  - to manage changes made to documents, programs, and directories
- Version control is useful for anything that:
  - changes over time, or
  - needs to be shared



[1] Image credit: https://unsplash.com/@cdr6934

What is version control?

- Track files in different states
- Simultaneous file development (Continuous Development)
- Combine different versions of files
- Identify a particular version
- Revert changes

Why is version control important?

Why is version control important?

[1] Image credit: https://unsplash.com/@mvdheuvel

Git

- Popular version control system for computer programming and data projects
- Open source
- Scalable

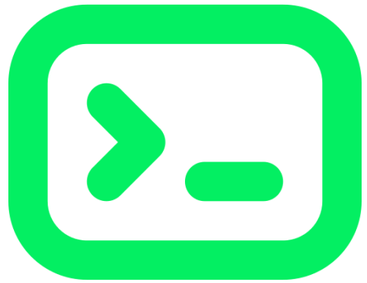
- Git is not GitHub, but
  - it's common to use Git with GitHub

Benefits of Git

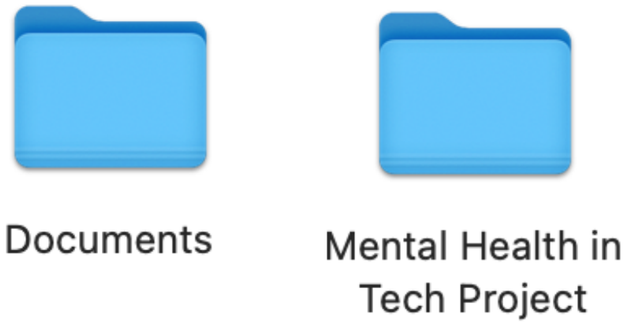
- Git stores everything, so nothing is lost
- Git notifies us when there is conflicting content in files
- Git synchronizes across different people and computers

Using Git

- Git commands are run on the **shell**, also known as the *terminal*
- The shell:
  - is a program for executing commands
  - can be used to easily preview, modify, or inspect files and directories



- Directory = folder



Useful shell commands

```
pwd
```

```
/home/repl/Documents
```

```
ls
```

```
archive    finance.csv    finance_data_clean.csv    finance_data_modified.csv
```

Changing directory

```
cd archive
```

```
pwd
```

```
/home/repl/Documents/archive
```

Editing a file

```
nano finance.csv
```

- Use `nano` to:
  - delete,
  - add,
  - or change contents of a file

- Save changes: `Ctrl + O`
- Exit the text editor: `Ctrl + X`

Editing a file

- `echo` —create or edit a file
- Create a new file `todo.txt`

```
echo "Review for duplicate records" > todo.txt
```

- Add content to existing file `todo.txt`

```
echo "Review for duplicate records" >> todo.txt
```

Checking Git version

```
git --version
```

```
git version 2.17.1
```

Let's practice!

INTRODUCTION TO VERSION CONTROL WITH GIT

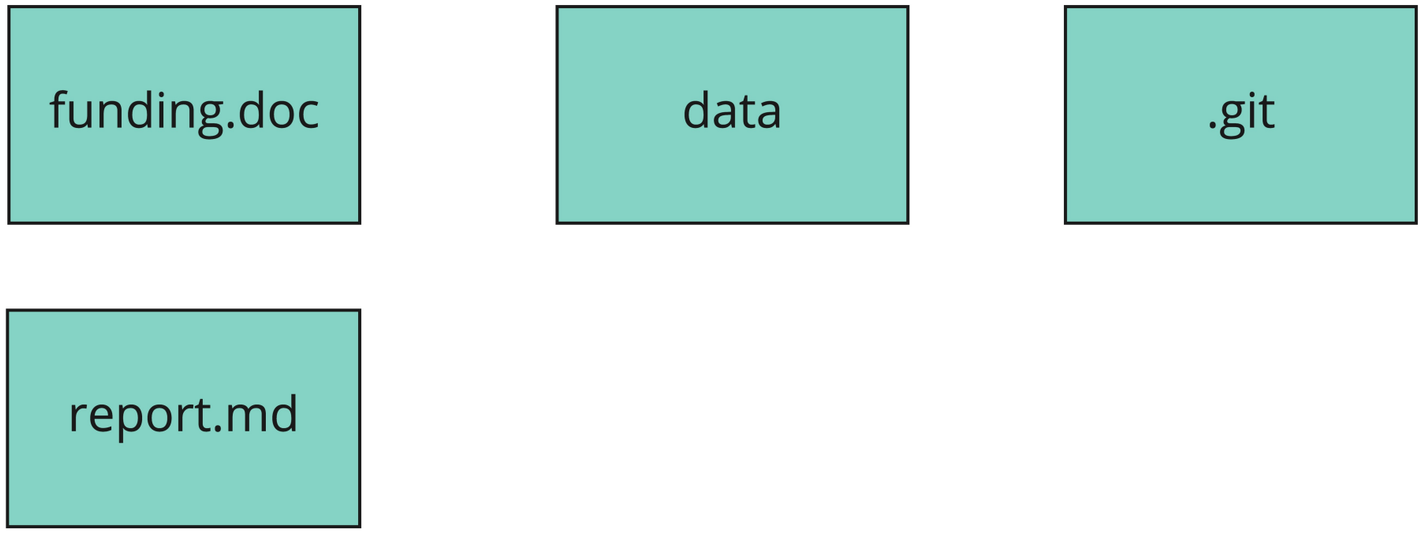
Saving files

INTRODUCTION TO VERSION CONTROL WITH GIT

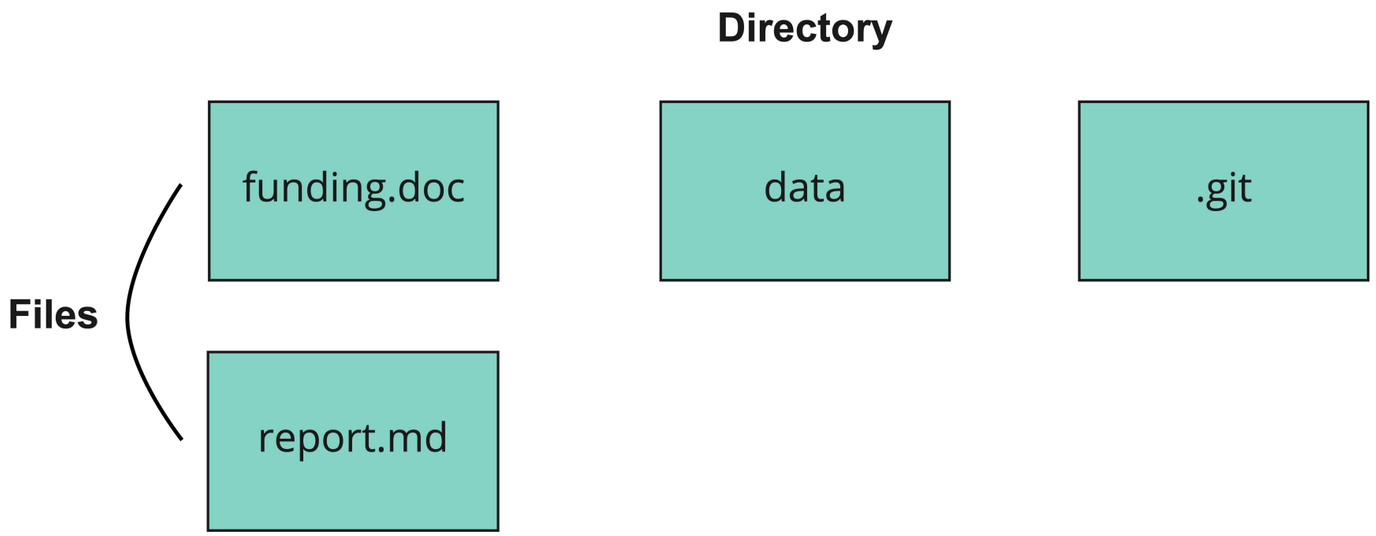
George Boorman
Curriculum Manager, DataCamp

datacamp

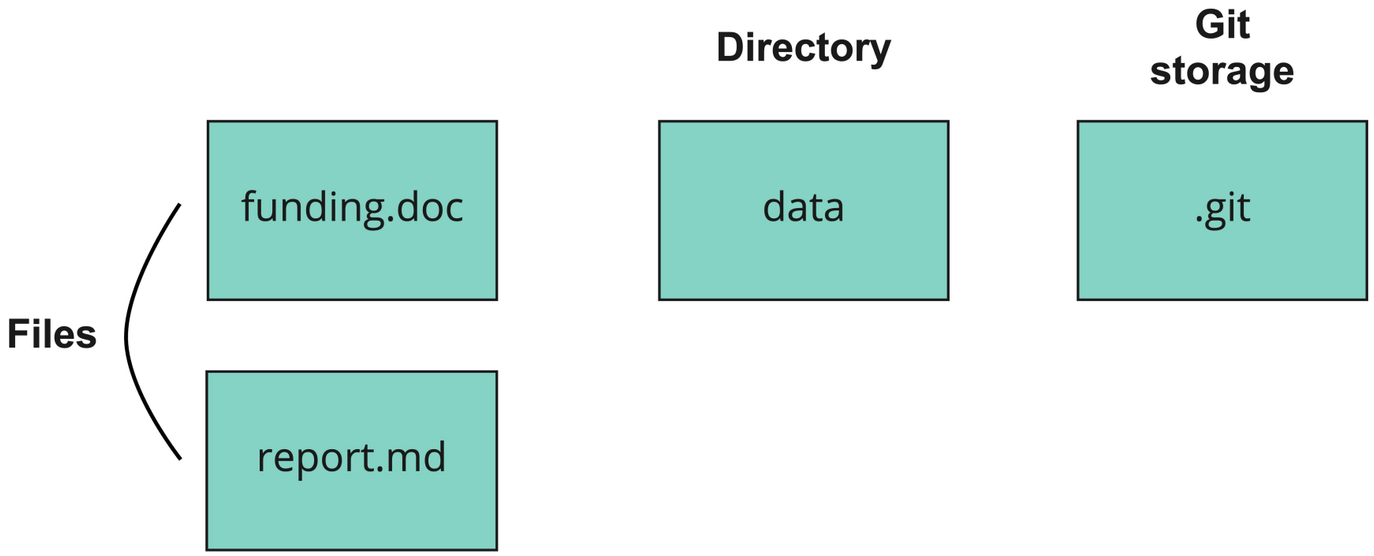
A repository



A repository



A repository



Do not edit `.git` !

Staging and committing

- Saving a draft
  - **Staging area**
- Save files/update the repo
  - **Commit changes**

[1] Image credits: https://unsplash.com/@brandomakesbranding; https://unsplash.com/@almapapi

Staging and committing

Staging area

Making a commit

Accessing the .git directory

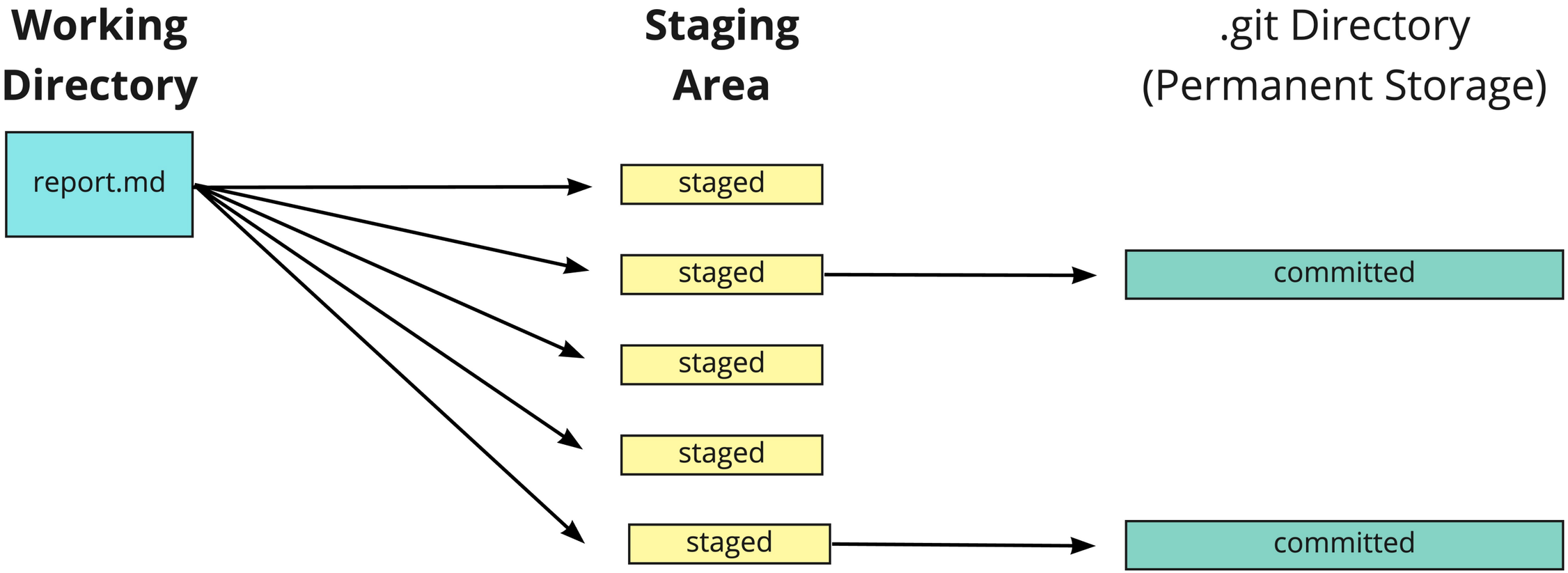
```
ls
```

```
data    report.md
```

```
ls -a
```

```
.      .DS_Store    data
..     .git         report.md
```

Making changes to files



Git workflow

- Modify a file
- Save the draft
- Commit the updated file
- Repeat

Modifying a file

```
nano report.md
```

```
# Mental Health in Tech Survey
TODO: write executive summary.
TODO: include link to raw data.
```

Save using `Ctrl + O` and `Ctrl + X`

Saving a file

- Adding a single file

```
git add report.md
```

- Adding all modified files

```
git add .
```

- `.` = all files and directories in current location

Making a commit

```
git commit -m "Updating TODO list in report.md"
```

- Log message is useful for reference
- Best practice = short and concise

Check the status of files

```
git status
```

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

```
git commit -m "New TODO in report.md"
```

Let's practice!

INTRODUCTION TO VERSION CONTROL WITH GIT

Comparing files

INTRODUCTION TO VERSION CONTROL WITH GIT

George Boorman
Curriculum Manager, DataCamp

datacamp

Why compare files?

[1] Image credit: https://unsplash.com/@mluotio83

Comparing a single file

```
nano report.md
```

Updating the file

```
git add .
```

```
git commit -m "Adding tasks for references and summary statistics in report.md"
```

Updating the file again

```
nano report.md
```

```
# Mental Health in Tech Survey
TODO: include link to raw data.
TODO: add references.
TODO: add summary statistics.
TODO: cite funding sources.
```

```
git diff report.md
```

Comparing an unstaged file with the last commit

```
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.
```

Comparing an unstaged file with the last commit

Line changes

```
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.
```

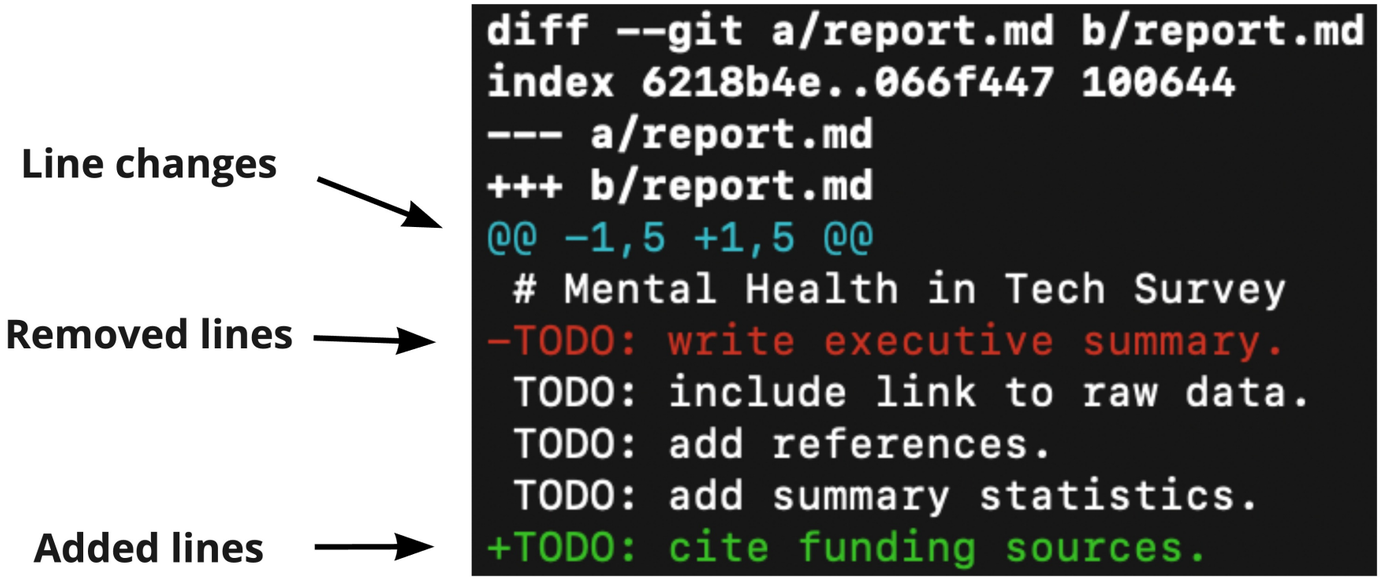
Comparing an unstaged file with the last commit

Line changes

Removed lines

```
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.
```

Comparing an unstaged file with the last commit



Comparing a staged file with the last commit

```
git add report.md
```

```
git diff -r HEAD report.md
```

- `git diff -r` won't work if it isn't followed by `HEAD`

Comparing a staged file with the last commit

```
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.
```

Comparing multiple staged files with the last commit

```
cd data
```

```
nano mh_tech_survey.csv
```

```
git add mh_tech_survey.csv
```

Comparing multiple staged files with the last commit

```
git diff -r HEAD
```

```
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,family_history,treatment,work_interfere,
ntal_health_interv
 28,M,No,Yes,Rarely,Yes,No,Yes
 29,F,No,Yes,Rarely,Don't know,No,Don't know
 23,M,Yes,No,Sometimes,No,No,No
+37,F,No,No,Rarely,Don't know,No,No
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.
```

Recap

- Compare an unstaged file with the last committed version:
  - `git diff filename`
- Compare a staged file with the last committed version:
  - `git diff -r HEAD filename`
- Compare all staged files with the last committed versions:
  - `git diff -r HEAD`

Let's practice!

INTRODUCTION TO VERSION CONTROL WITH GIT