D8A CHEATSHEET · #042 · GIT FOR DATA

Git, just the part data work actually uses.

D8A.ACADEMY/RESOURCES/GIT-FOR-DATA

THE DAILY LOOP

`git status`	what changed
`git add -p`	stage by chunk
`git commit -m "…"`	say why, not what
`git push`	ship it

BRANCHES

`git switch -c fix-x`	new branch
`git switch main`	go back
`git merge fix-x`	bring it in

UNDO

`git restore f.sql`	discard local edit
`git revert <sha>`	undo, keep history
`git stash`	park it, come back

READING HISTORY

`git log --oneline`	the story
`git diff main`	what would merge
`git blame f.py`	who and when

DATA HYGIENE

`.gitignore first`	before commit 1
`*.csv data/ .env`	never in a repo
`commit the schema`	not the rows

GOTCHAS

`git add .`	← just added a 2 GB csv.
`push --force`	rewrites shared history
`secrets in history`	rotate the key. now

WORTH MEMORIZING

01 The loop you run every day

Four commands cover 90% of real usage. The flag worth learning is -p.

```
git status                -- what changed
git add -p                -- stage hunk by hunk
git commit -m (why)       -- "fix tz bug", not "edits"
git push                  -- ship it

git switch -c (branch)    -- experiments live here
```

→ add -p shows every change before staging it. It is also how you catch the file that should never be committed.

02 Trap: the repo is public

A portfolio repo gets read by recruiters. And by scrapers.

YOU COMMITTED	WHAT HAPPENS	DO INSTEAD
`data/raw.csv, 400 MB`	clone takes minutes, forever	`.gitignore + a sample`
`.env with an API key`	scrapers find it in hours	`rotate it, then clean`
`password in a notebook`	it stays in history	`never in code at all`

→ Deleting the file in a new commit does not remove it from history. Assume any pushed secret is burned.