Script

Log

Filter

```
git log -n <aantal>
git log --since=<datum>
git log --until=<datum>
git log --author=<naam>
git log --grep=<patroon>
git log -p # Toon wat er in een commit veranderd is
git log <bestand>
git log --stat # stats wijziging
git log branch-naam --pretty=format:"%s | %an | %ad" --date=format:"%Y/%m/%d"
```

Sorteren

```
# DATUM ACHTERWAARTS
--sort=committerdate
# AUTEUR
 | sort
# HASH
--sort=hash
# COMMIT
--sort=subject
```

Stats

```
stats() {
    local branch=${1:-HEAD}
    local commit_count contributor_count file_count tag_count first_commit
last_commit local_branches remote_branches untracked_files

    commit_count=$(git rev-list --count "$branch")
    contributor_count=$(git shortlog -sn "$branch" | wc -l)
    file_count=$(git ls-files | wc -l)
    tag_count=$(git tag | wc -l)
    first_commit=$(git log --reverse --format="%ad" "$branch" | head -1)
    last_commit=$(git log -1 --format="%ad" "$branch")
    local_branches=$(git branch | wc -l)
    remote_branches=$(git branch -r | wc -l)
    untracked_files=$(git ls-files --others --exclude-standard | wc -l)

    echo "=== Repository Stats ==="
    echo "Branch: $branch"
    echo "$commit_count commit$( [ "$commit_count" -ne 1 ] && echo "s" ) by
$contributor_count contributor$( [ "$contributor_count" -ne 1 ] && echo "s" )"
    echo "$file_count tracked file$( [ "$file_count" -ne 1 ] && echo "s" )"
    echo "$untracked_files untracked file$( [ "$untracked_files" -ne 1 ] && echo
"s" )"
    echo "$tag_count tag$( [ "$tag_count" -ne 1 ] && echo "s" )"
    echo "Local branches: $local_branches, Remote branches: $remote_branches"
    echo "First commit: $first_commit"
    echo "Last commit:  $last_commit"

    echo ""
    echo "Top contributors:"
    git shortlog -sn "$branch" | head -5

    echo ""
    echo "Commits per month:"
    git log --date=short --pretty=format:"%ad" "$branch" | cut -d- -f1-2 | sort |
uniq -c

    git log --name-only --pretty=format: | grep -v '^$' | sort | uniq -c | sort -
nr | head -10

    git log --format='%an %ad' --date=short | sort -u -k1,1
}
```

UNDO

check of er commits zijn

```
    if [ $(git rev-list --count HEAD) -eq 0 ]; then
        echo "Error: No commits to undo."
```

```
        return 1
    fi
```

Vraag bevestiging

```
    local last_commit_message
    last_commit_message=$(git log -1 --pretty=%s)
    echo "Laatste commit: \"$last_commit_message\""
    read -p "Weet je zeker dat je deze commit wilt ongedaan maken? (j/n): "
confirm
    if [[ "$confirm" != [jJ] ]]; then
        echo "Annulering: commit blijft behouden."
        return 0
    fi
```

Kies type reset

```
    echo "Kies type reset:"
    echo "1) --soft (houdt wijzigingen staged)"
    echo "2) --mixed (houdt wijzigingen ongestaged)"
    echo "3) --hard (verwijdert wijzigingen volledig!)"
    read -p "Optie (1/2/3): " reset_type

    case "$reset_type" in
        1) git reset --soft HEAD~1 ;;
        2) git reset --mixed HEAD~1 ;;
        3) git reset --hard HEAD~1 ;;
        *) echo "Ongeldige optie, annulering."; return 1 ;;
    esac
```

meerder commits undo

```
    read -p "Aantal commits om ongedaan te maken (max $total_commits): " num
    if ! [[ "$num" =~ ^[0-9]+$ ]] || [ "$num" -gt "$total_commits" ] || [ "$num" -
lt 1 ]; then
        echo "❌ Ongeldig aantal."
        return 1
    fi

    git reset --hard HEAD~1 ;;
```

bekijk veranderingen van een commit

```
git show <commit-hash>
```

Ask_Change

```
ask_to_change() {
    local setting="$1"

    # Prompt tonen
    read -p "$setting not set. Do you want to set it now? [Y/n] " answer

    # Standaard is ENTER gelijk aan 'yes'
    answer=${answer:-y}

    # Kleine letters vergelijken
    case "${answer,,}" in
        y|yes)
            return 0   # wil aanpassen
            ;;
        *)
            return 1   # niet aanpassen
            ;;
    esac
}
```

create a repo

```
mkdir git_practice
cd git_practice
git init
```

create README file

```
touch README.md
echo "This is readme file for git_practice project" > README.md
```

cleanup_old_files

```
  echo "Cleaning up old files"
  find ~/.trash -type f -mtime +14 -exec rm -v {} \;
```

Maak trash folder

```
TRASH_DIR="$HOME/.trash"
if [ ! -d "$TRASH_DIR" ]; then
  mkdir "$TRASH_DIR"
  echo "Created trash folder $TRASH_DIR"
fi
```

revert commit

```
revert_commit() {
    local commit_hash="$2"
    if [ -z "$commit_hash" ]; then
        echo "Please provide a commit hash."
        exit 1
    fi
    git revert "$commit_hash"
    echo "Reverted commit '$commit_hash'."
}
```