Roll Back

1. git revert <commit-hash>

- **Effect**:
  - This creates a **new commit** that undoes the changes introduced by the specified commit.
  - The original commit and its history remain intact.
  - Safe for shared branches since history is preserved and no force-push is required.
  - **Use Case**: When you want to undo a commit but keep a clear history of what was undone.

2. git reset --soft <commit-hash>

- **Effect**:
  - Moves the `HEAD` pointer to the specified commit.
  - Changes made in commits after the target commit are **kept staged** (in the index).
  - Does not touch your working directory or the content of the files.
  - **Use Case**: When you want to uncommit but keep changes ready to be re-committed.

3. git reset --hard <commit-hash>

- **Effect**:
  - Moves the `HEAD` pointer to the specified commit.
  - Discards all changes made in commits after the target commit.
  - **Working directory and staging area are wiped clean**, matching the state of the target commit.
  - **Use Case**: When you want to completely discard commits and their changes.

git reflog for recovery