**Texto

Descripción generada automáticamente**

**COMANDOS**

* **PWD** nos muestra el path o ruta de la carpeta en donde nos encontramos ubicados
* **CD** me permite acceder (entrar) a una carpeta en un nivel o varios niveles
* **CD ..** me permite salir de una carpeta en un nivel o varios niveles OJO los dos puntos deben ser separados por un espacio del comando cd
* **LS** me muestra los archivos que contiene una carpeta, puede ser la ubicación actual o una ruta especifica, no muestra los archivos ocultos
* **LS -A** me muestra los archivos que contiene una carpeta, puede ser la ubicación actual o una ruta especifica, incluyendo los archivos ocultos
* **LS -L** me lista los archivos que contiene una carpeta con sus atributos, puede ser la ubicación actual o una ruta especifica, no muestra los archivos ocultos
* **LS -AL** me lista los archivos que contiene una carpeta con sus atributos, puede ser la ubicación actual o una ruta especifica, incluyendo los archivos ocultos
* **CLEAR** limpiar la consola o terminal, o un shorcut **CRTL + L**
* **MKDIR <NOMBRE CARPETA>** nos permite crear una carpeta
* **TOUCH <NOMBRE DEL ARCHIVO>** nos permite crear un archivo
* **CAT <NOMBRE DEL ARCHIVO>** me permite visualizar el contenido del un archivo y lo muestra en el terminal
* **HISTORY** nos muestra un historial de los comandos que hemos utilizado
* **RM <NOMBRE DEL ARCHIVO>** me permite borrar un archivo

**COMANDOS GIT**

**GIT INIT** inicializa el repositorio de git

**GIT ADD:** agrega un archivo al repositorio

**GIT COMMIT -M “mensaje”** Agrega definitivamente los cambios hechos al repositorio, se puede poner un comentario para indicar que cambió

**GIT ADD** . Agrega todos los cambios en todos los archivos al área de staging

**GIT STATUS** Para ver el estado de la BD sí se han hecho cambios, pero no se han agregado, ahí saldrá

**GIT SHOW** Muestra todos los cambios históricos hechos y sus detalles (qué cambió, cuándo y quién los hizo)

**GIT LOG** Muestra la historia completa de un archivo

**GIT PUSH** envía los cambios al repositorio remoto

**GIT PULL** Agrega los cambios del commit más nuevo al repositorio local

**GIT DIFF** muestra los cambios entre 2 commits

**GIT RM:** lo usamos para borrar un archivo que hayamos añadido, para eliminarlo por completo de nuestra rama usamos git rm --cached.

**GIT COMMIT:** se usa para añadir un commit a nuestra rama, también podemos ponerle un -m seguidamente ponemos entre comillas nuestro ensaje.

**GIT COMMIT -am: es para archivos que ya hice add y se modificaron, directamente uso este comando y no add**

**GIT CONFIG:** muestra configuraciones de git también podemos usar --list para mostrar la configuración por defecto de nuestro git y si añadimos --show-origin nos muestra las configuraciones guardadas y su ubicación.

**GIT CONFIG --GLOBAL USER.NAME:** cambia de manera global el nombre del usuario, seguidamente ponemos entre comillas nuestro nombre.

**GIT CONFIG --GLOBAL USER.EMAIL:** cambia de manera global el email del usuario, seguidamente ponemos entre comillas nuestro nombre.

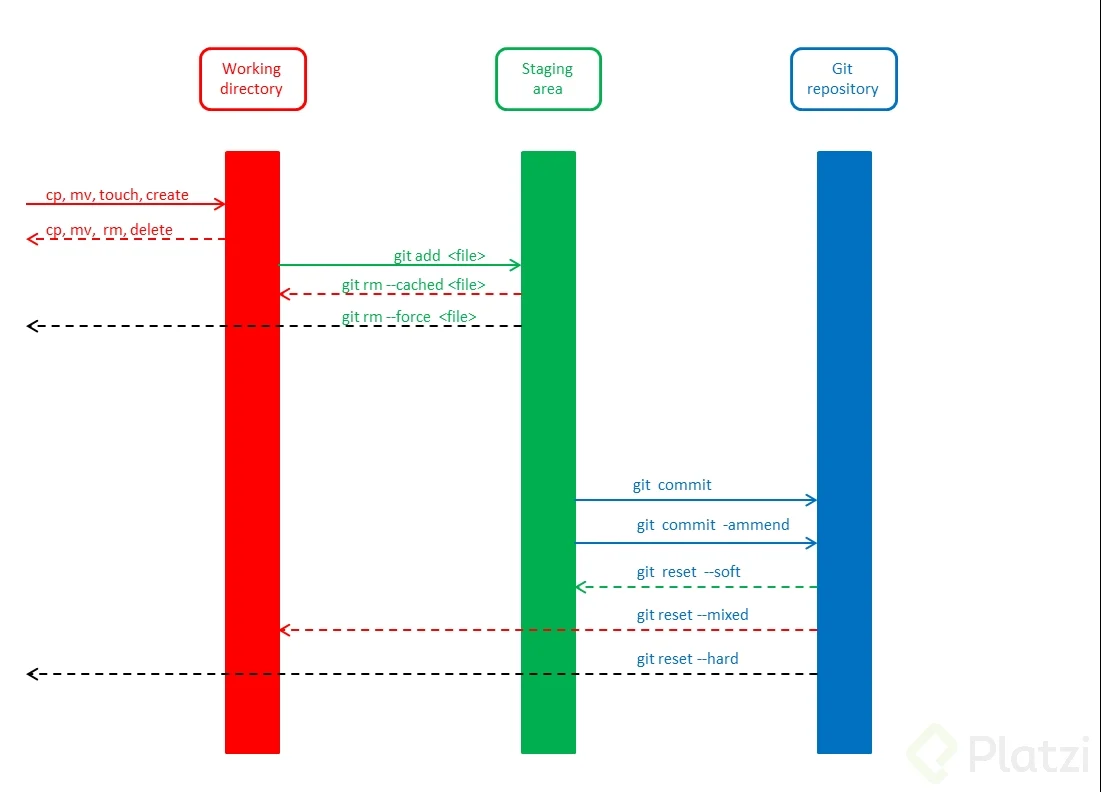
**GIT RESET --SOFT [SHA 1]:** elimina los cambios hasta el staging area

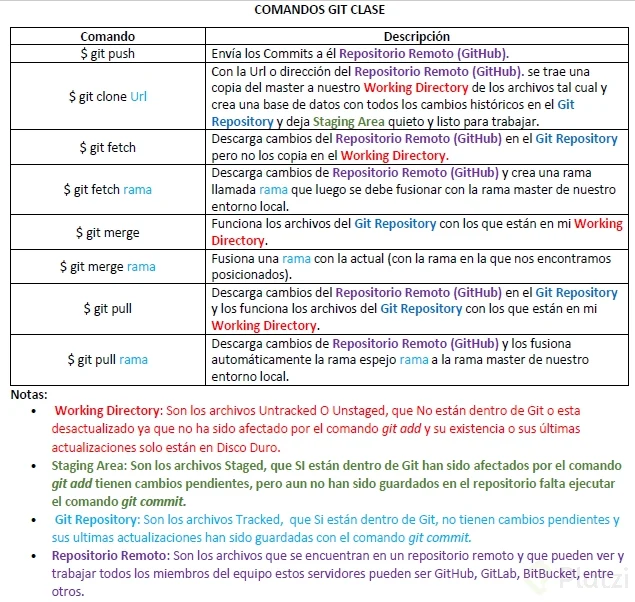
**GIT RESET --MIXED [SHA 1]:** elimina los cambios hasta el working area

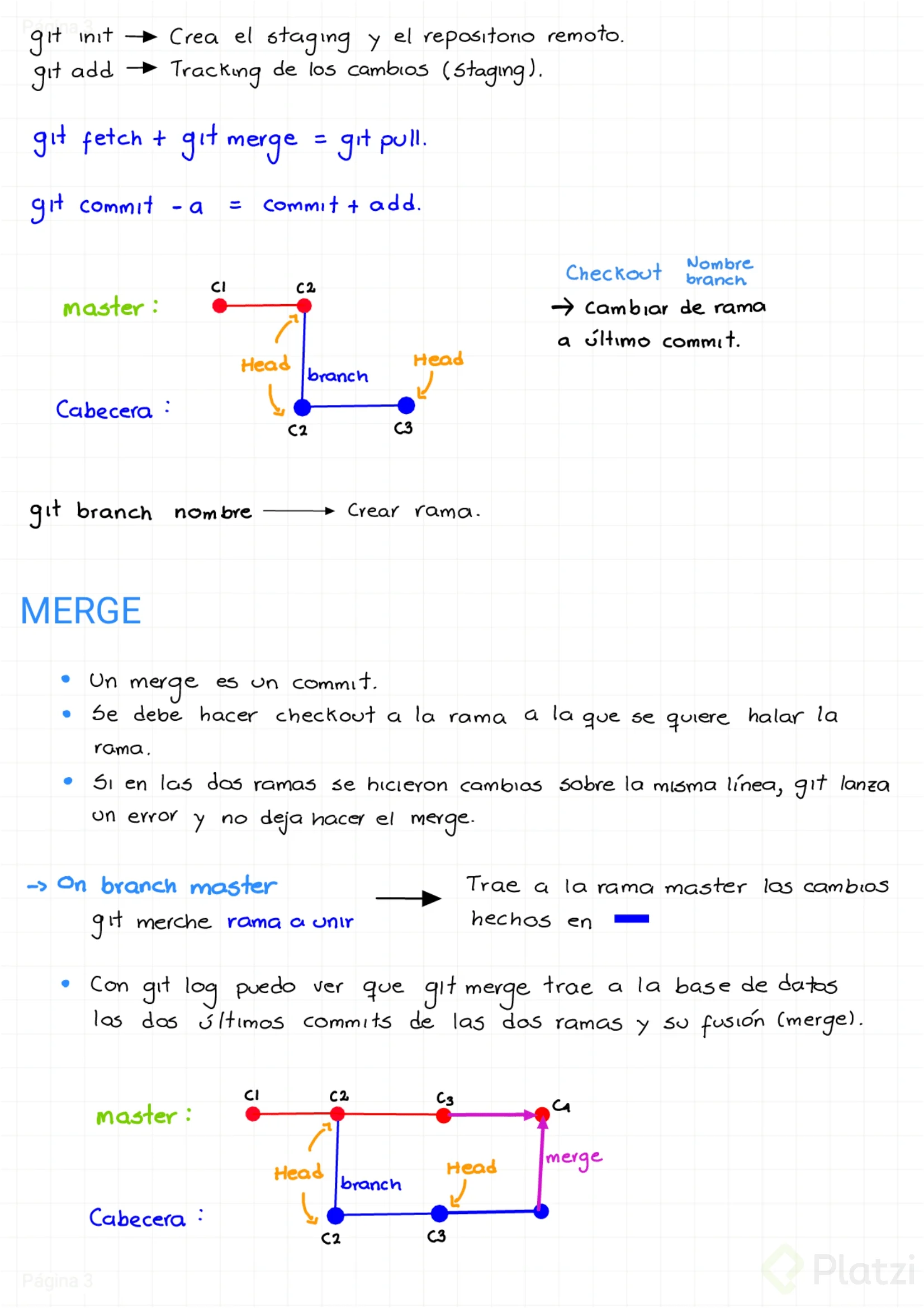
**GIT RESET --HARD [SHA 1]:** regresa hasta el commit del [SHA-1]

**GIT CHECKOUT -NOMBRE-** moverse entre ramas

**GIT BRANCH -NOMBRE- :** crea una rama

****

****

****

* **git remote add origin (url)**
* **git remote (muestra el origen)**
* **git remote -v(es verval)**
* **git pull origin master --allow-unrelated-histories (Fuerza la union de las diferentes historias)**
* **git pull origin master (Descarga cambios)**
* **git push (origin) (master) (Sube Cambios)**