Git과 GitHub 고급 (5)

Git 심화 (5): 병합 전략 및 병합 충돌 해결 전략
(Git Rebase 명령 알아보기)

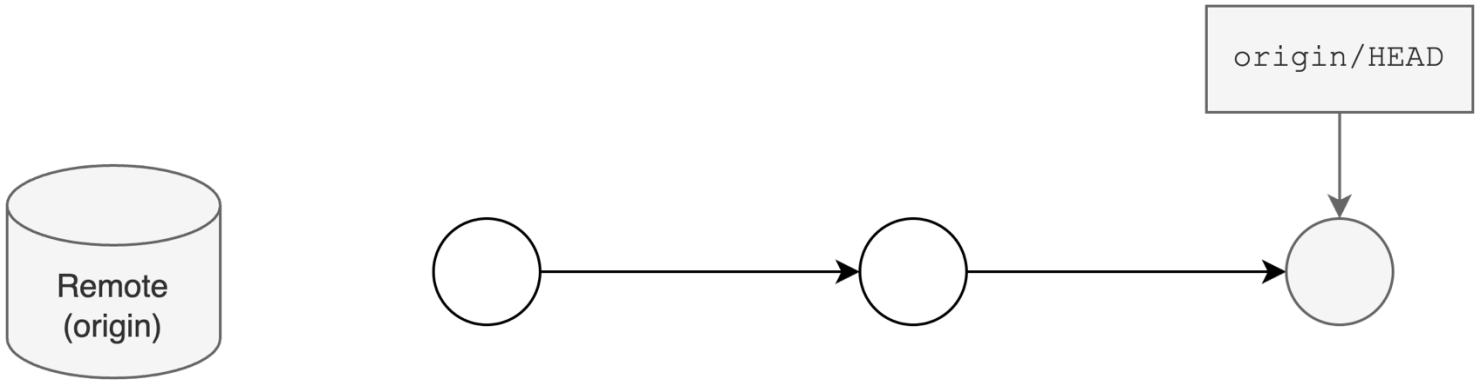
주차별 활동 계획

주차	날짜	주제
1주차	9/25 (수)	Git 기초 (1): Git의 설치, Git과 Git의 Concept 소개
2주차	10/2 (수)	Git 기초 (2): 브랜치 분기와 원격 저장소 & 자주 사용되는 Git 명령어
3주차	10/16 (수)	Git 협업 (3): 브랜치 병합과 GitFlow, 그리고 GitHub Issue
4주차	10/30 (수)	Git 협업 (4): GitHub Pull Request & Code Review
5주차	11/6 (수)	**Git 심화 (5): 병합 전략 및 병합 충돌 해결 전략 (Git Rebase 명령 알아보기)**
6주차	11/13 (수)	Git 심화 (6): Git의 자료구조와 실수로 삭제한 커밋 복구하기
7주차	11/20 (수)	Git 응용 (7): GitHub Actions를 활용한 자동화 워크플로 생성
8주차	11/27 (수)	Git 응용 (8): Git Submodule & Git Hook (로컬에서의 자동화)

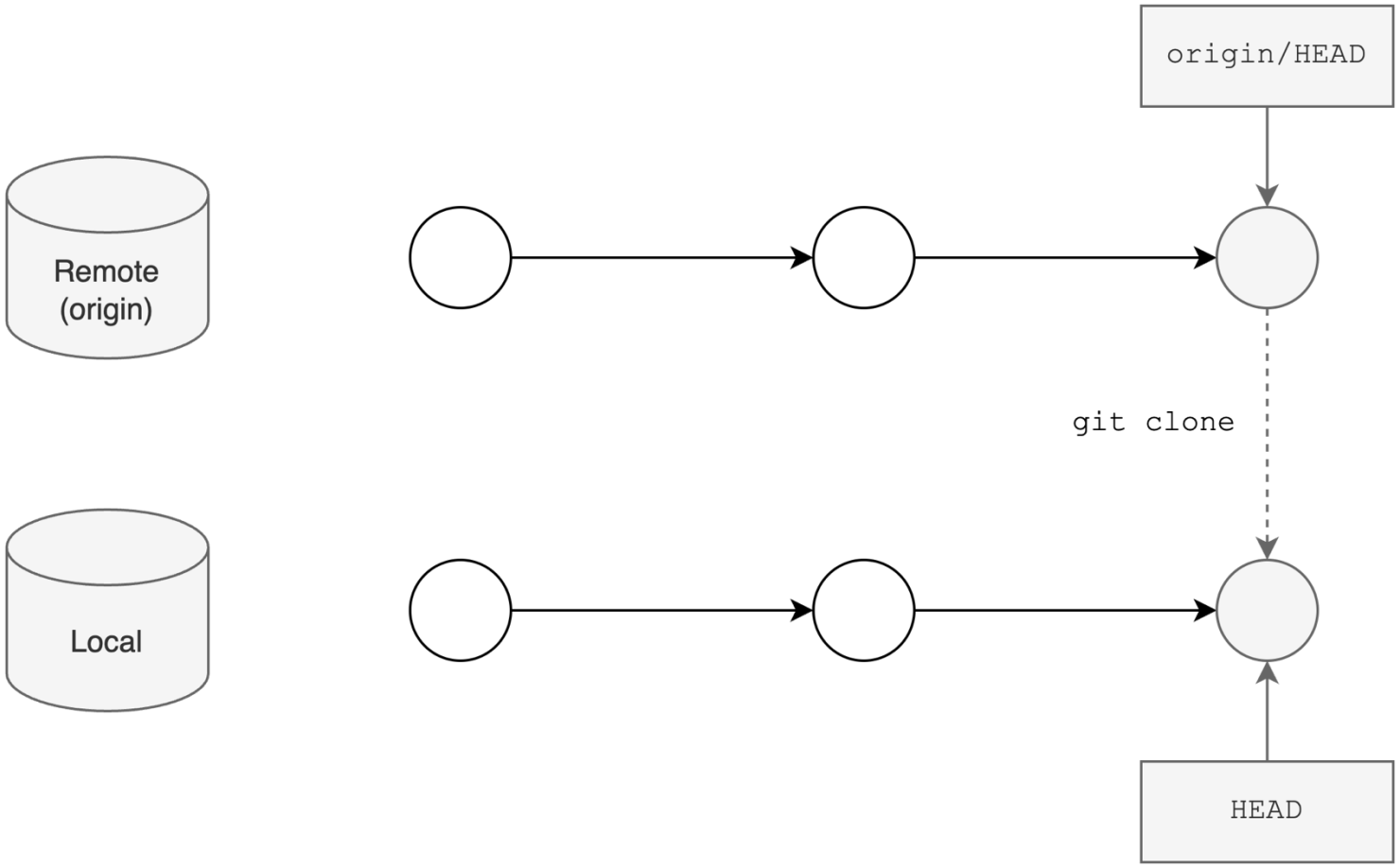
Git Quick Review

Git에 대하여 빠르게 리뷰

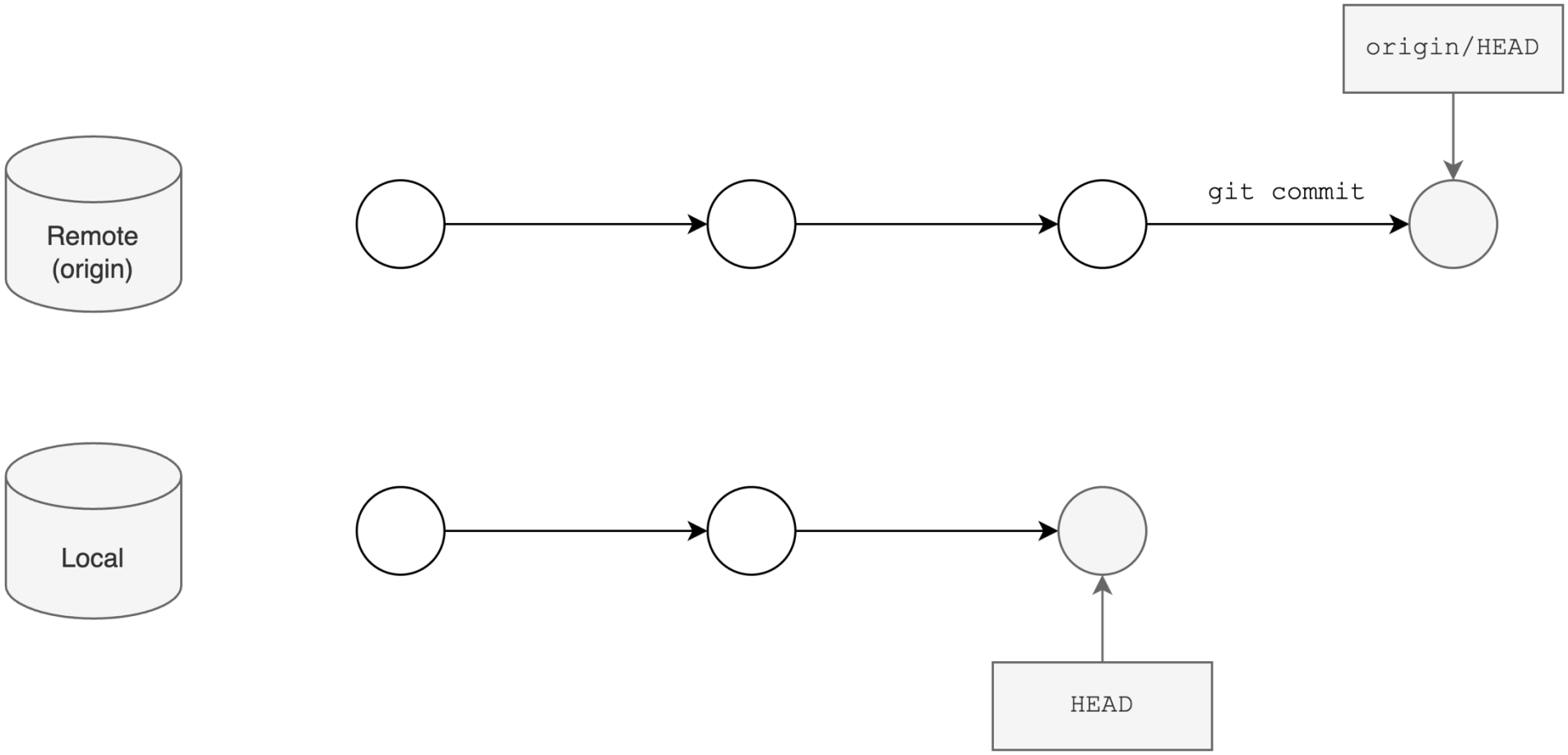
Git



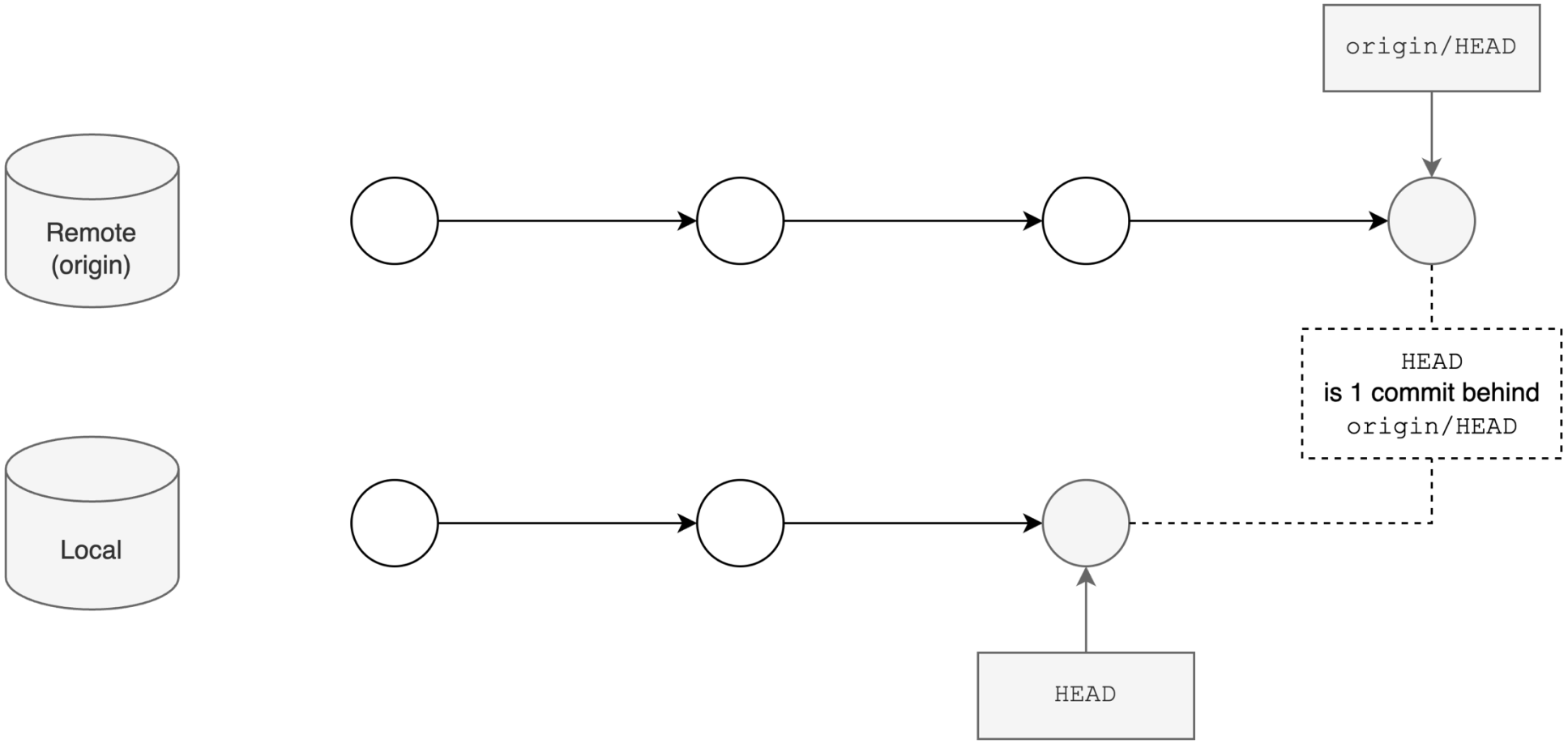
Git



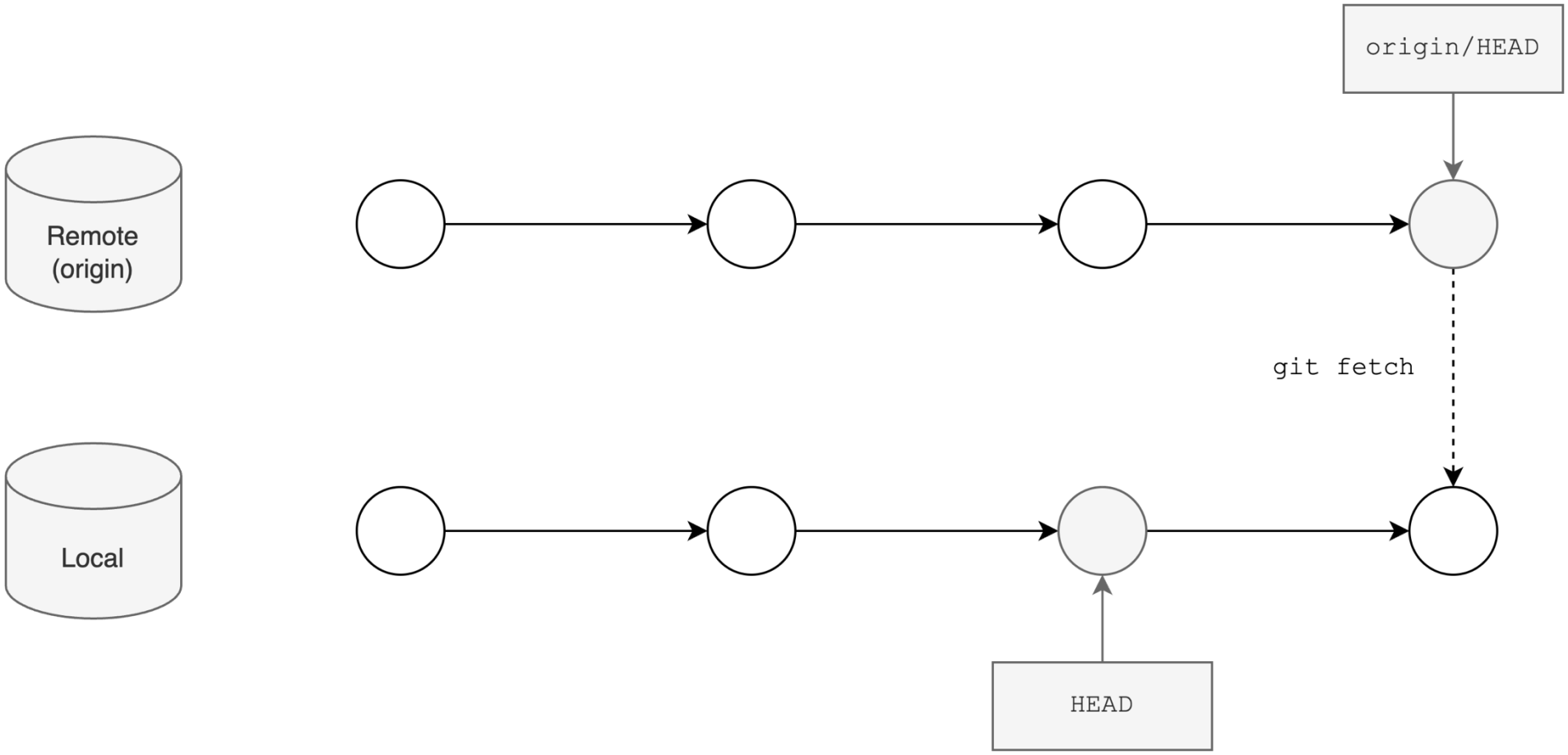
Git



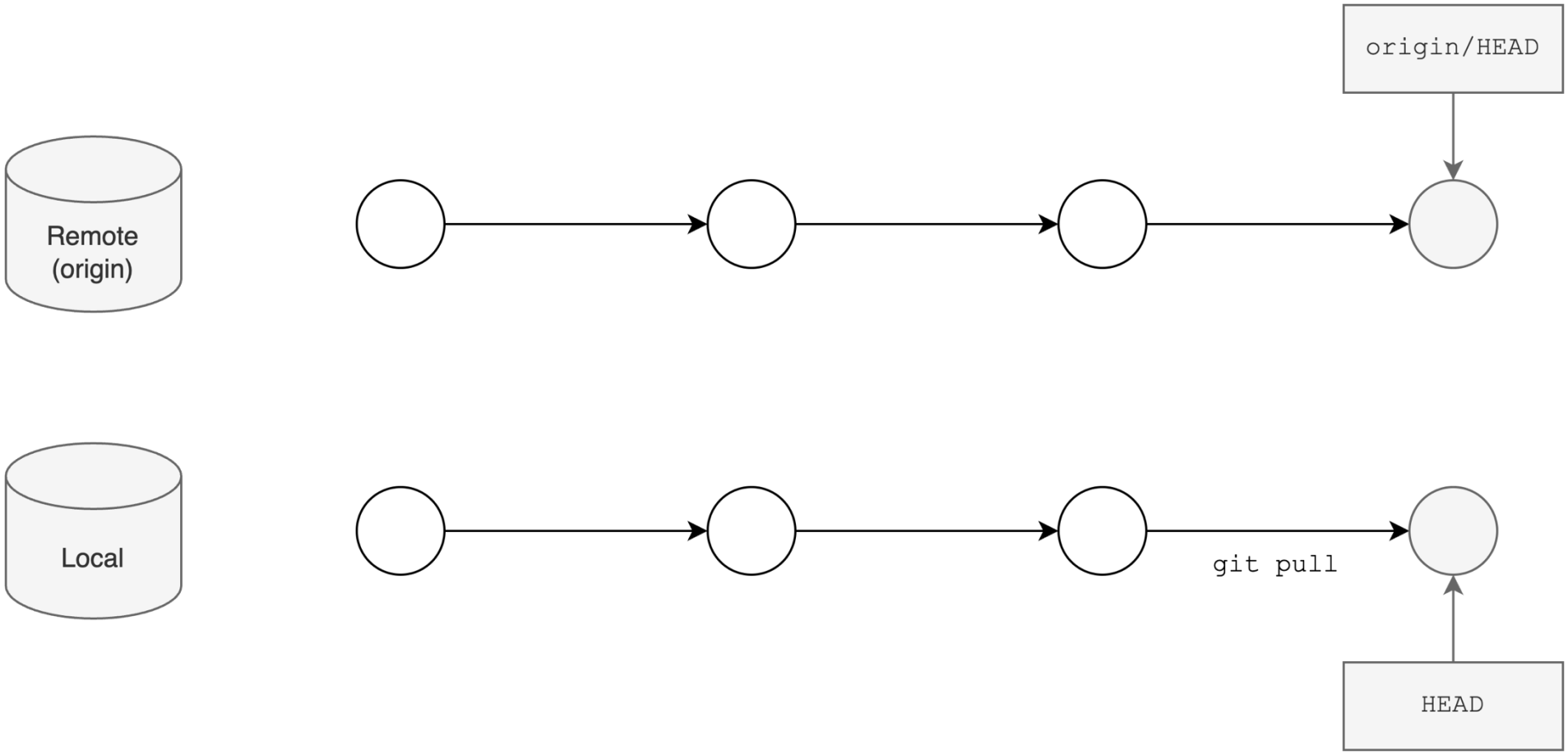
Git


origin/HEAD
Remote
(origin)
HEAD
is 1 commit behind
origin/HEAD
Local
HEAD

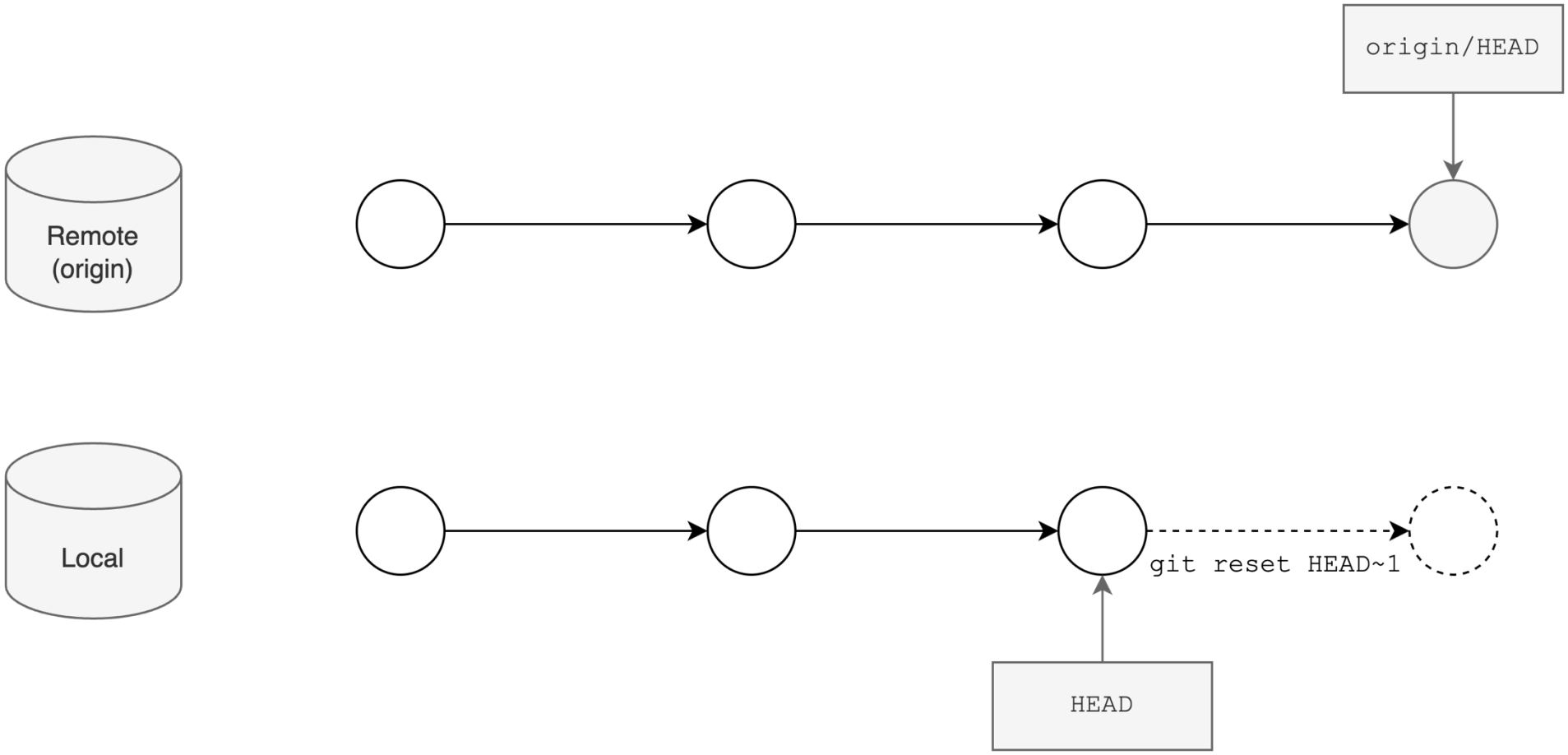
Git



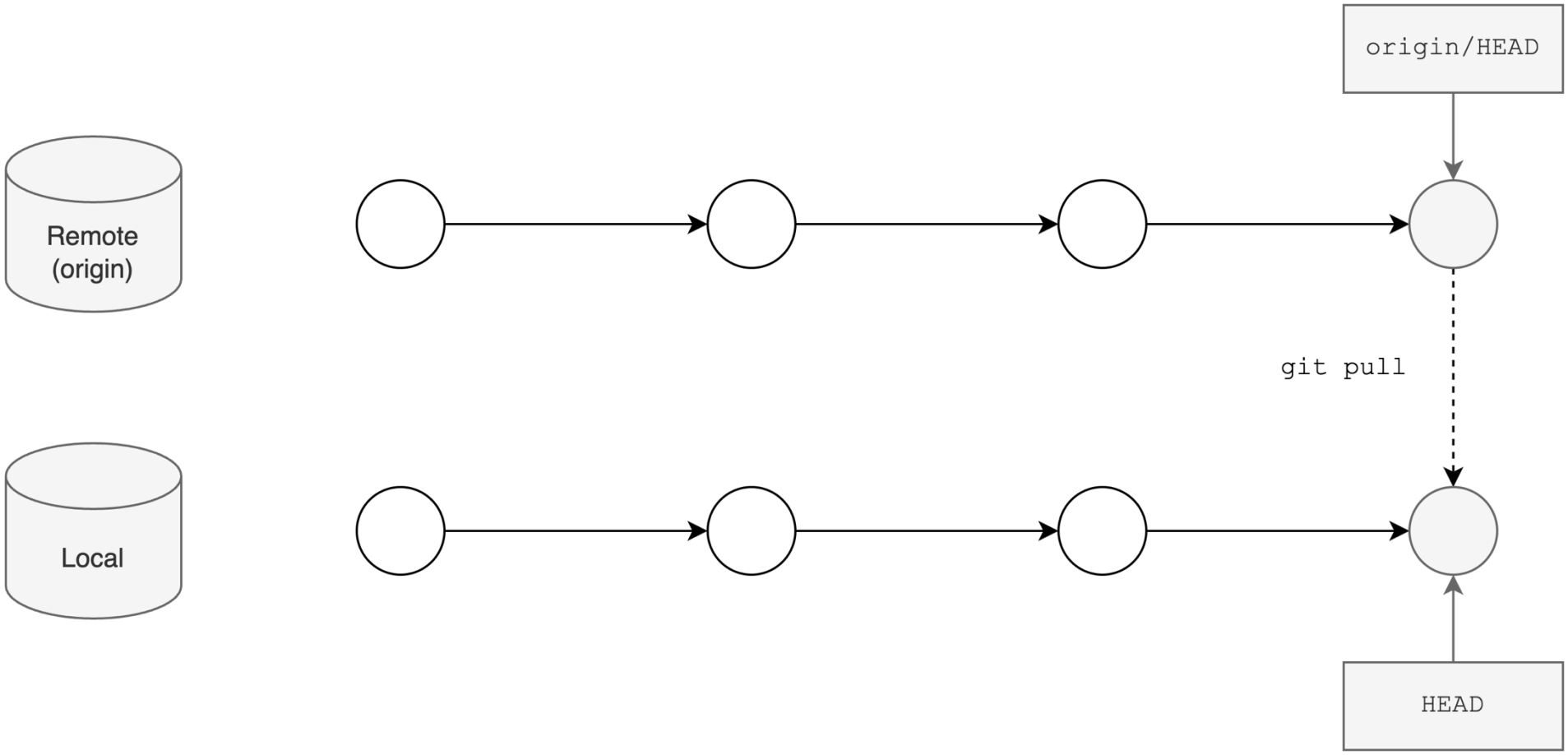
Git



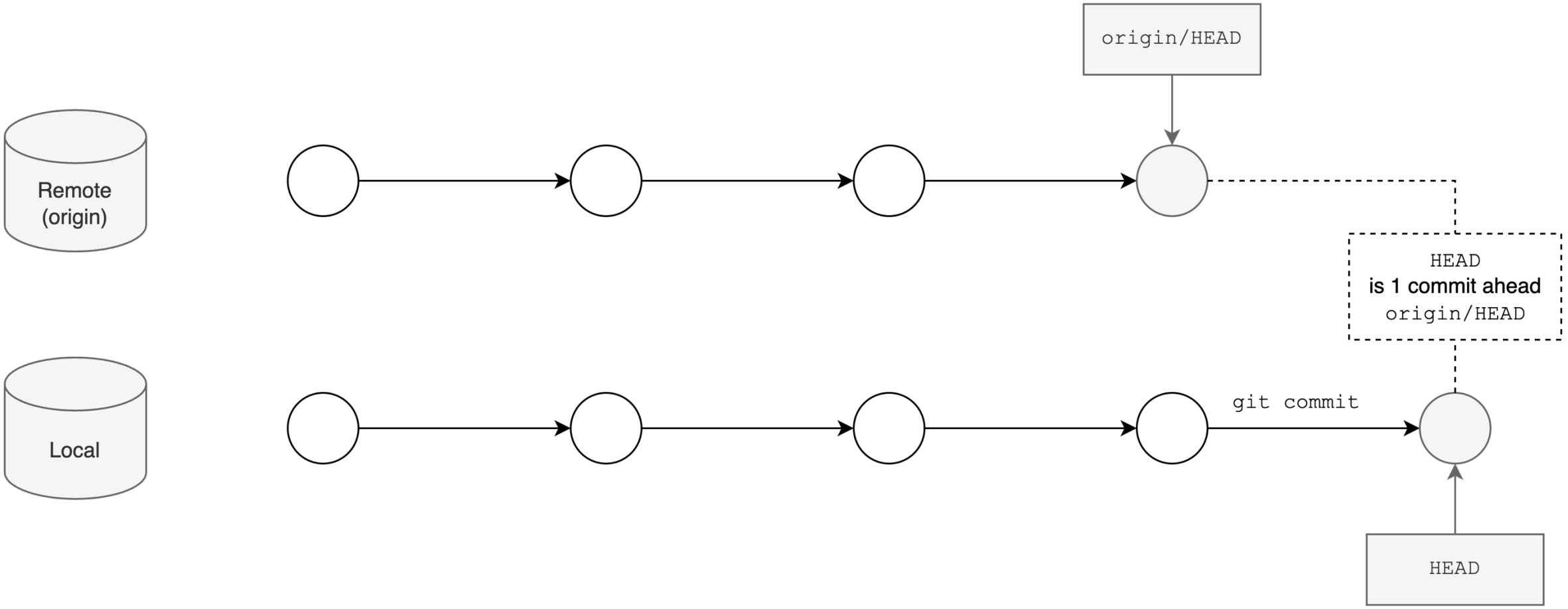
Git



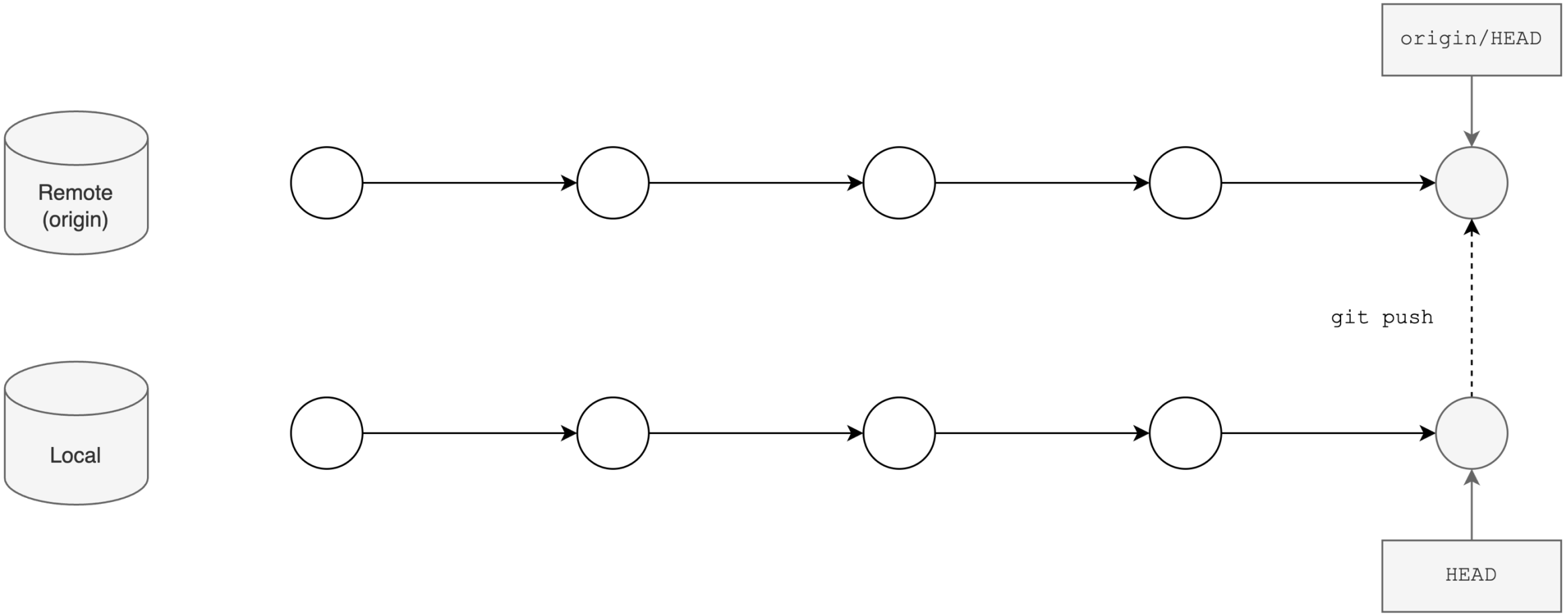
Git



Git

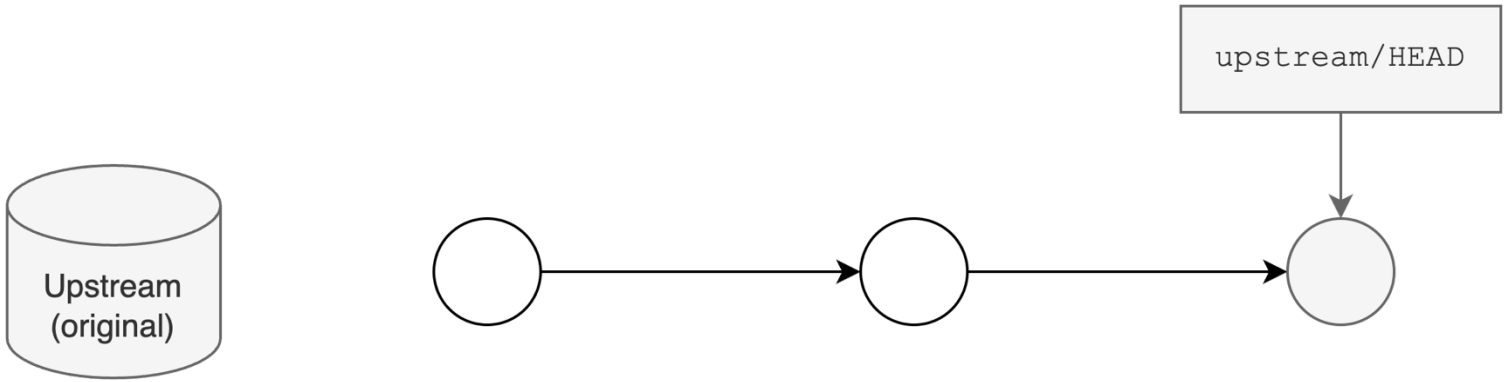

origin/HEAD
Remote
(origin)
HEAD
is 1 commit ahead
origin/HEAD
Local
git commit
HEAD

Git

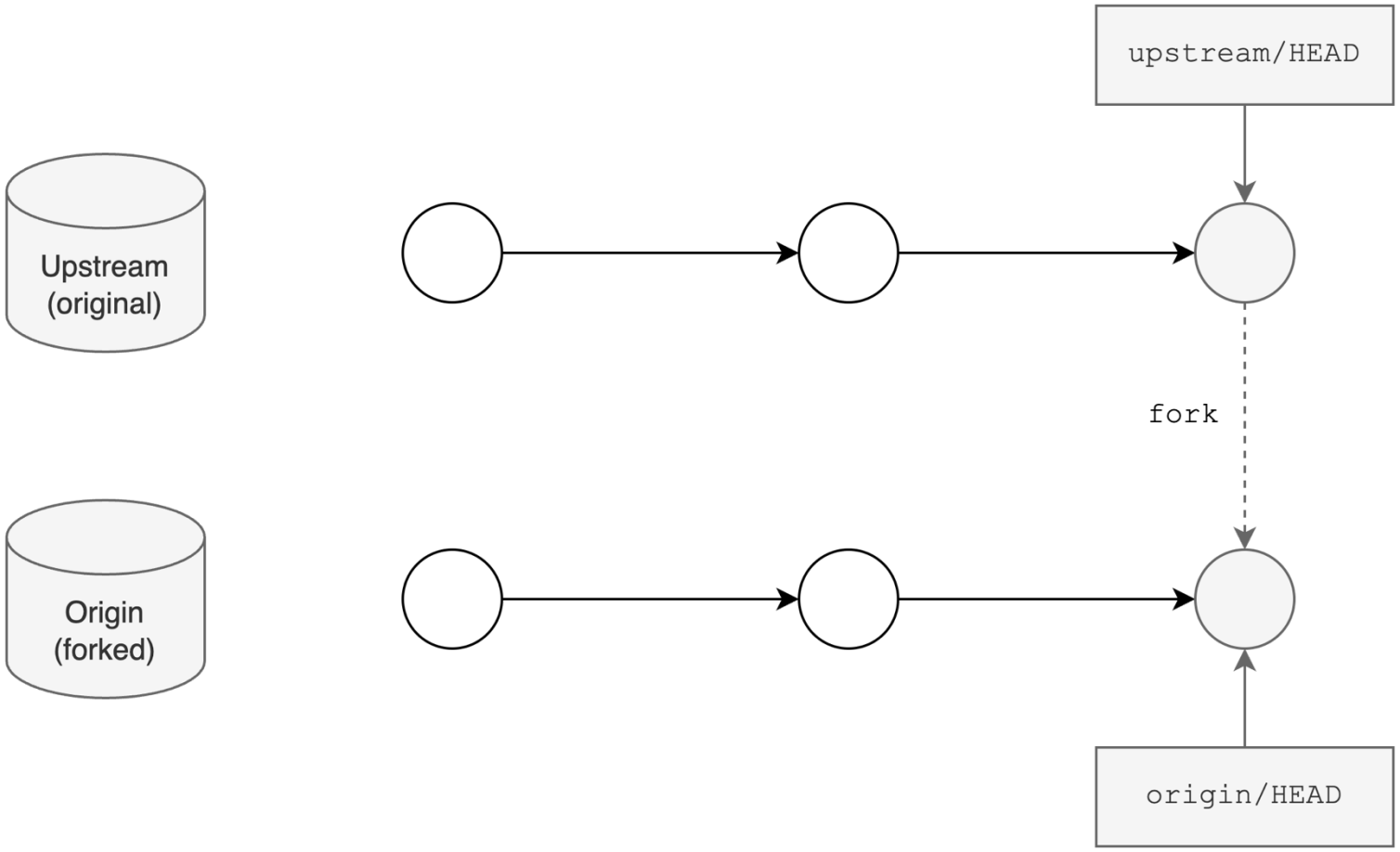


GitHub Quick Review

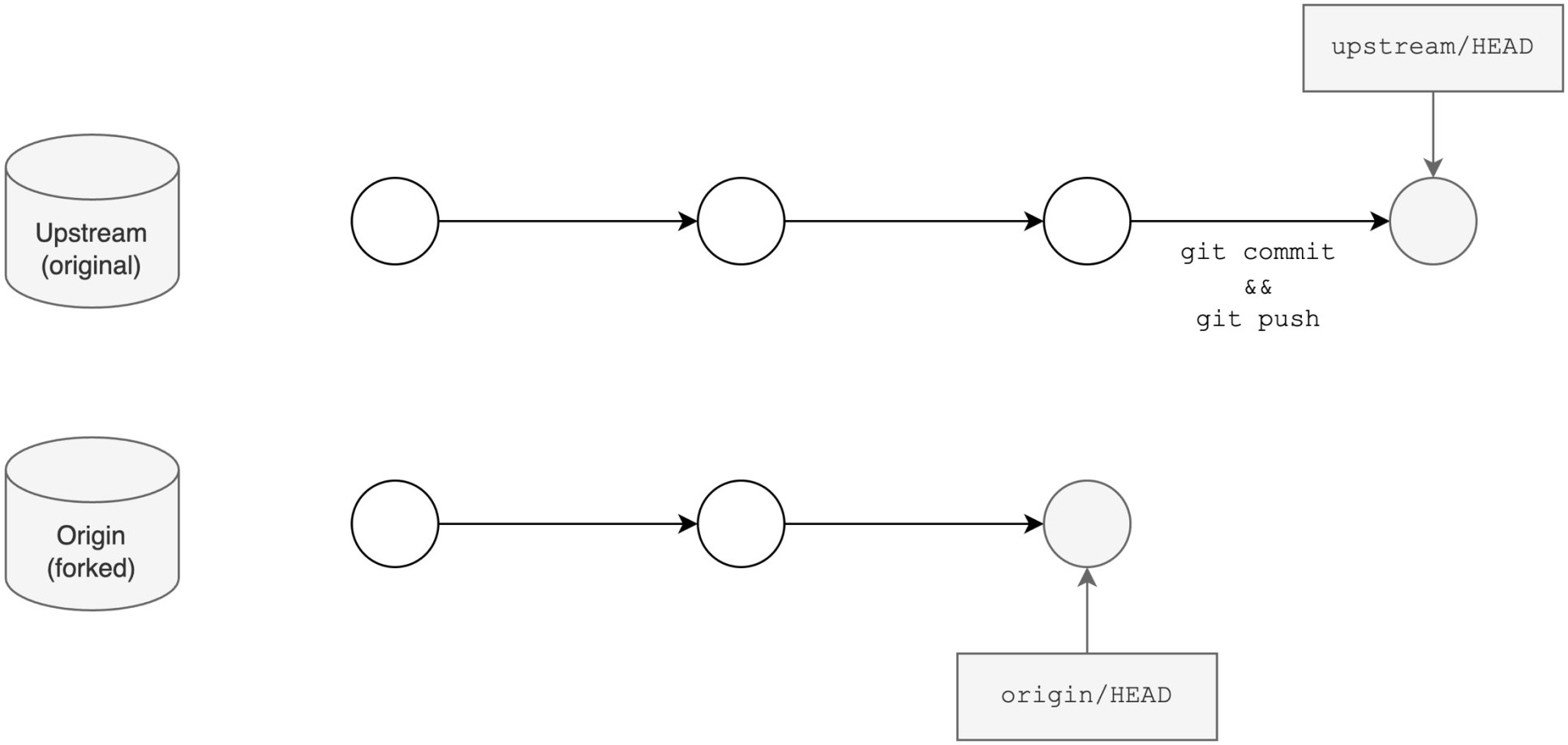
GitHub



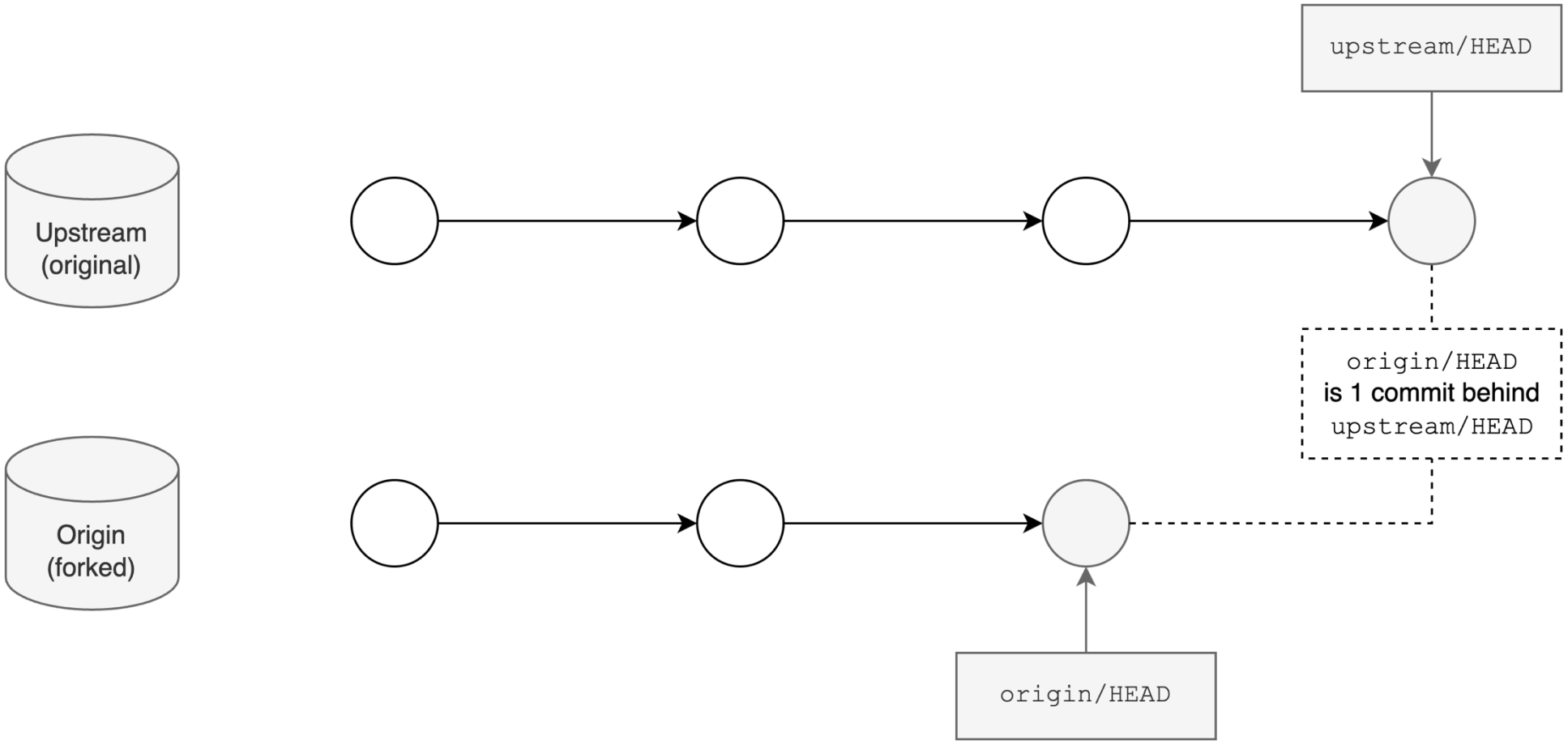
GitHub


upstream/HEAD
Upstream
(original)
fork
Origin
(forked)
origin/HEAD

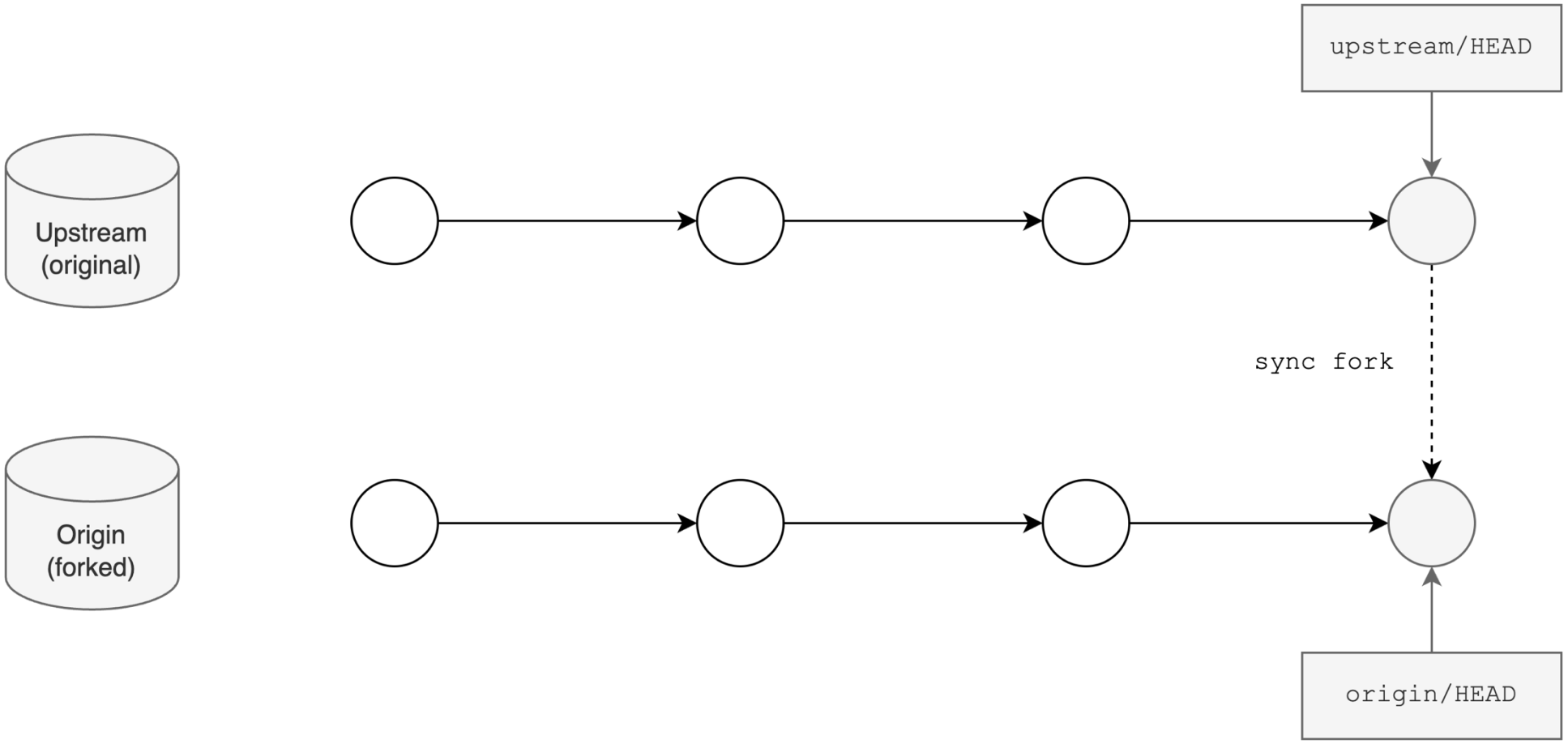
GitHub


upstream/HEAD
Upstream
(original)
git commit
&&
git push
Origin
(forked)
origin/HEAD

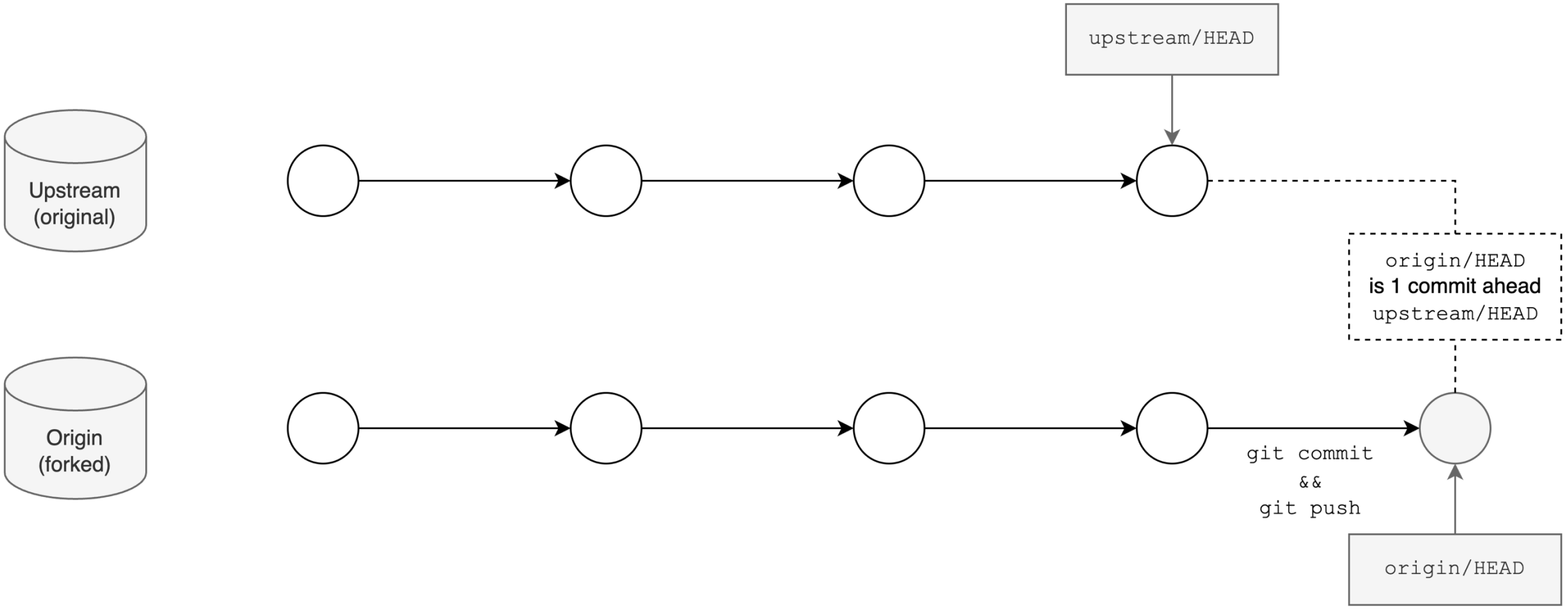
GitHub


upstream/HEAD
Upstream
(original)
origin/HEAD
is 1 commit behind
upstream/HEAD
Origin
(forked)
origin/HEAD

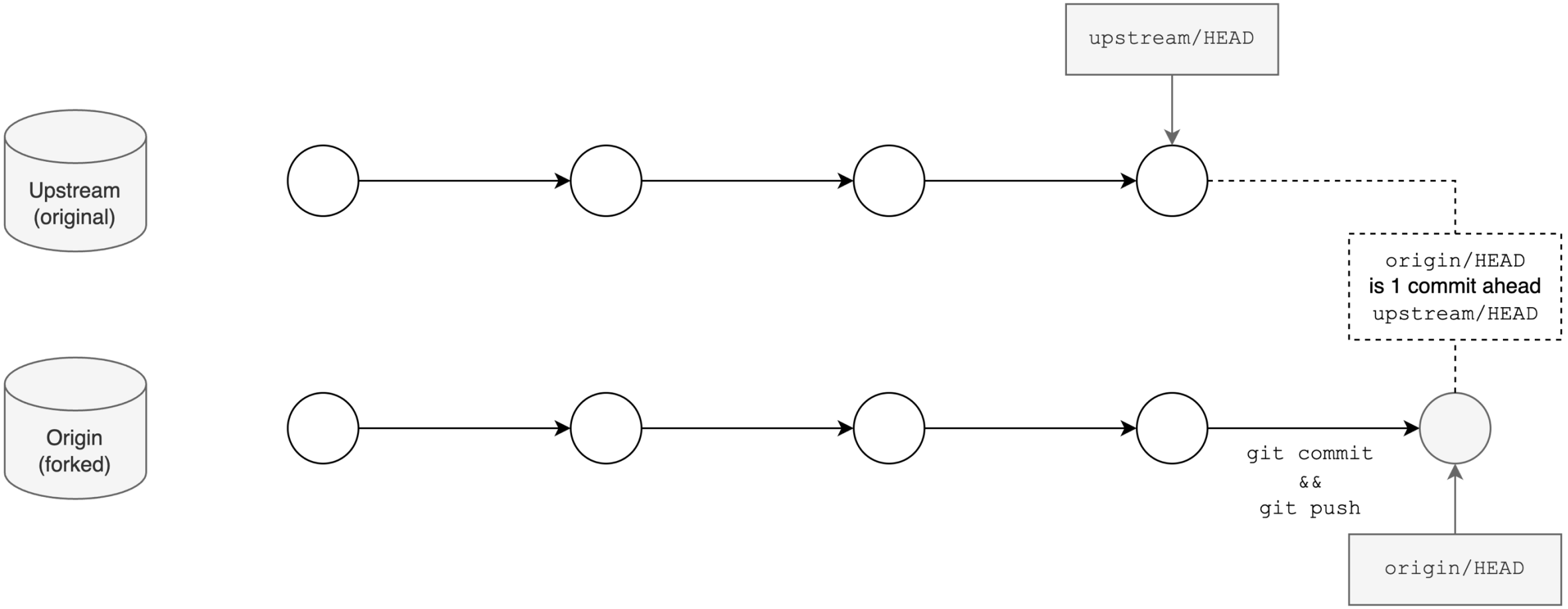
GitHub


upstream/HEAD
Upstream
(original)
sync fork
Origin
(forked)
origin/HEAD

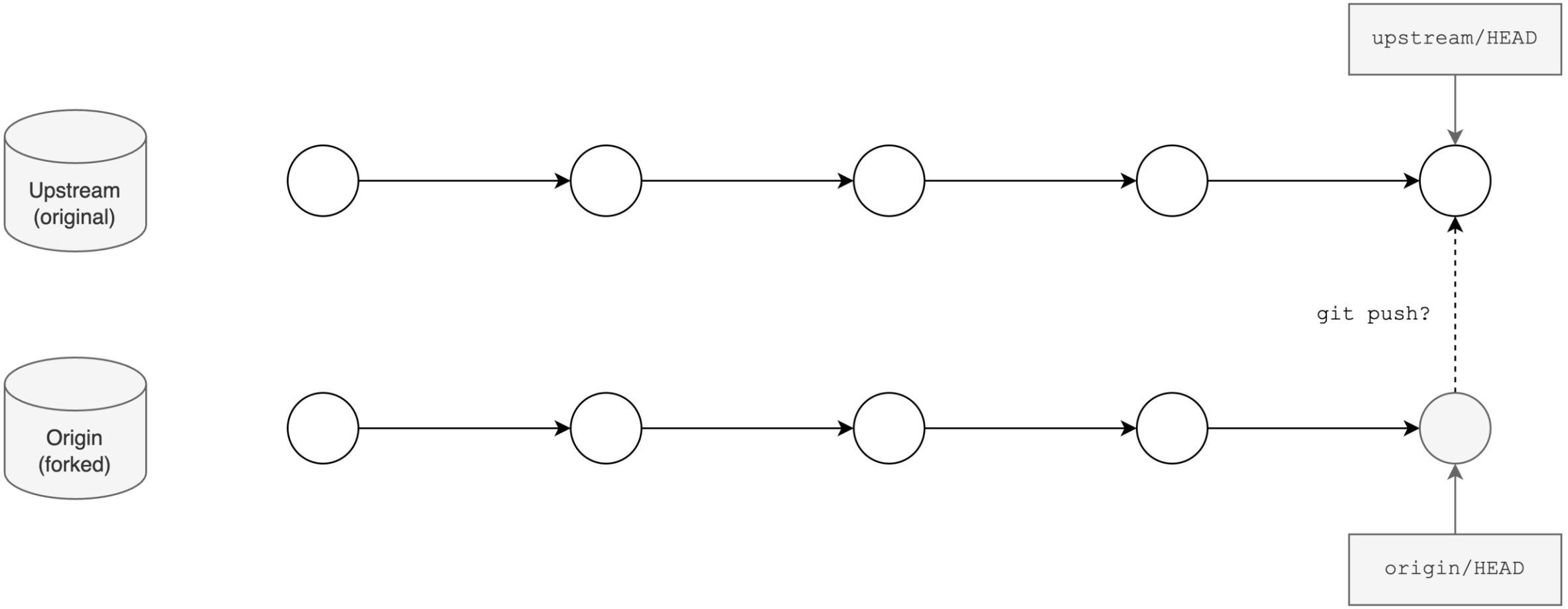
GitHub


upstream/HEAD
Upstream
(original)
origin/HEAD
is 1 commit ahead
upstream/HEAD
Origin
(forked)
git commit
&&
git push
origin/HEAD

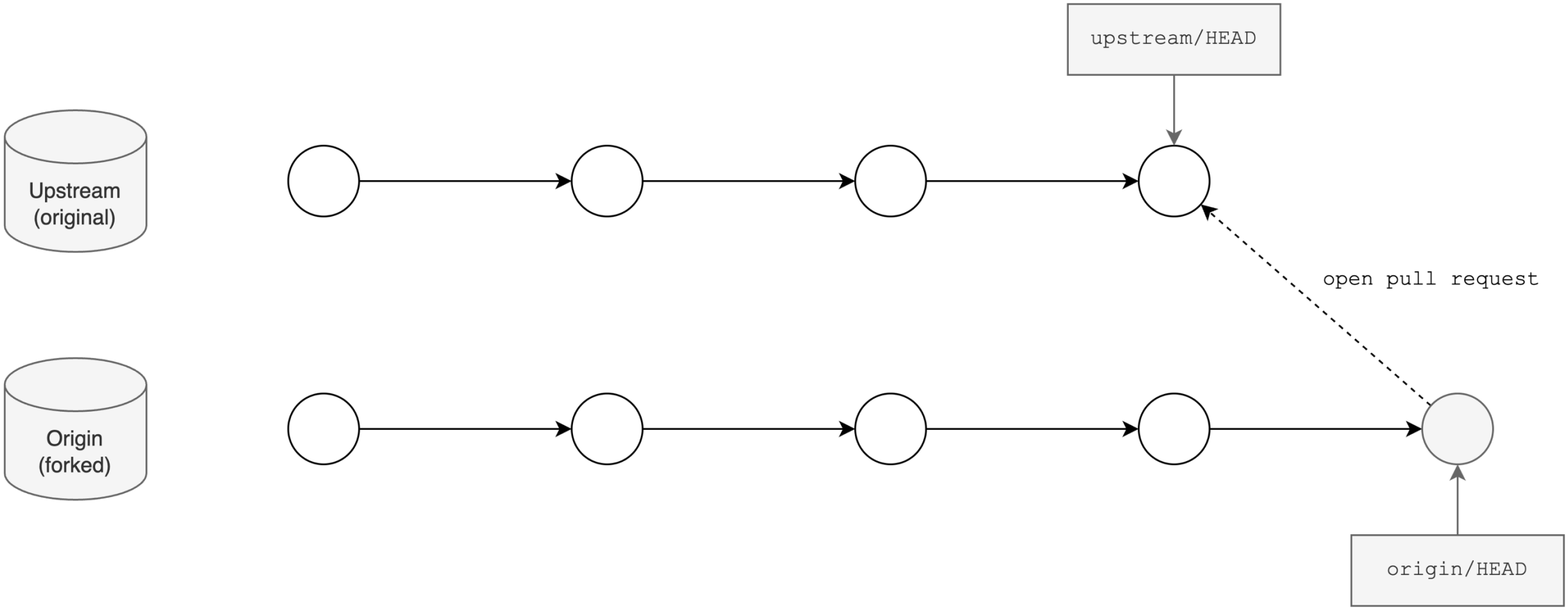
GitHub


upstream/HEAD
Upstream
(original)
origin/HEAD
is 1 commit ahead
upstream/HEAD
Origin
(forked)
git commit
&&
git push
origin/HEAD

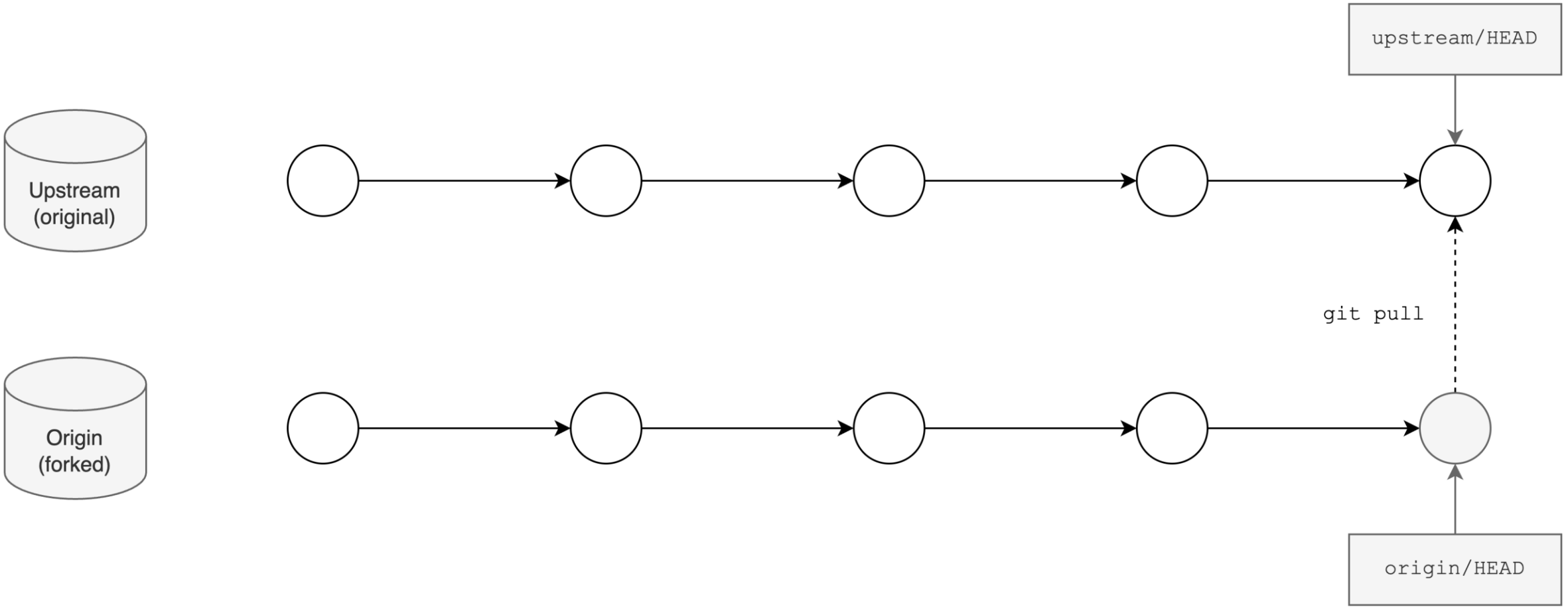
GitHub


upstream/HEAD
Upstream
(original)
git push?
Origin
(forked)
origin/HEAD

GitHub


upstream/HEAD
Upstream
(original)
open pull request
Origin
(forked)
origin/HEAD

GitHub

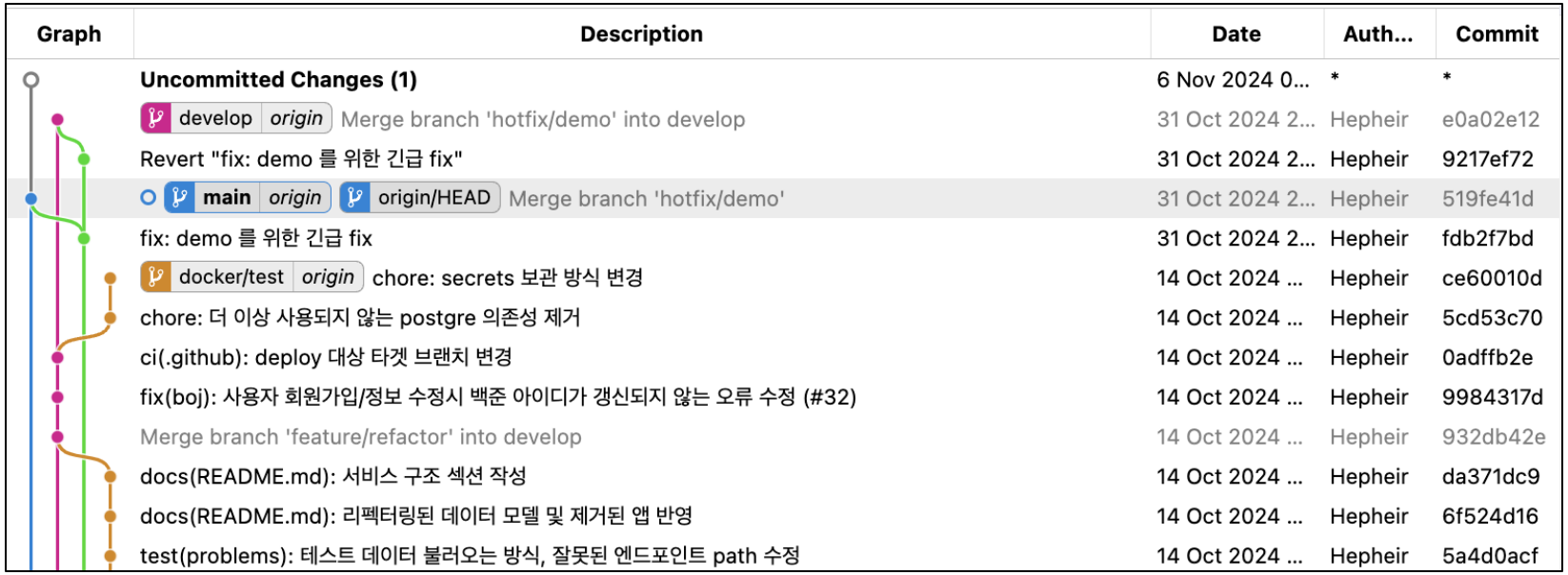

upstream/HEAD
Upstream
(original)
git pull
Origin
(forked)
origin/HEAD

Git Merge

Merge에 대해서 빠르게 알아보자

Merge

- Merge의 장점
  - 병합된 브랜치가 삭제되어 사라진다해도 히스토리 그래프 상에는 다른 분기로 표기
    - 어떤 브랜치에서 어떤 커밋이 어떻게 병합 되었는지 자세히 알 수 있음.



Merge

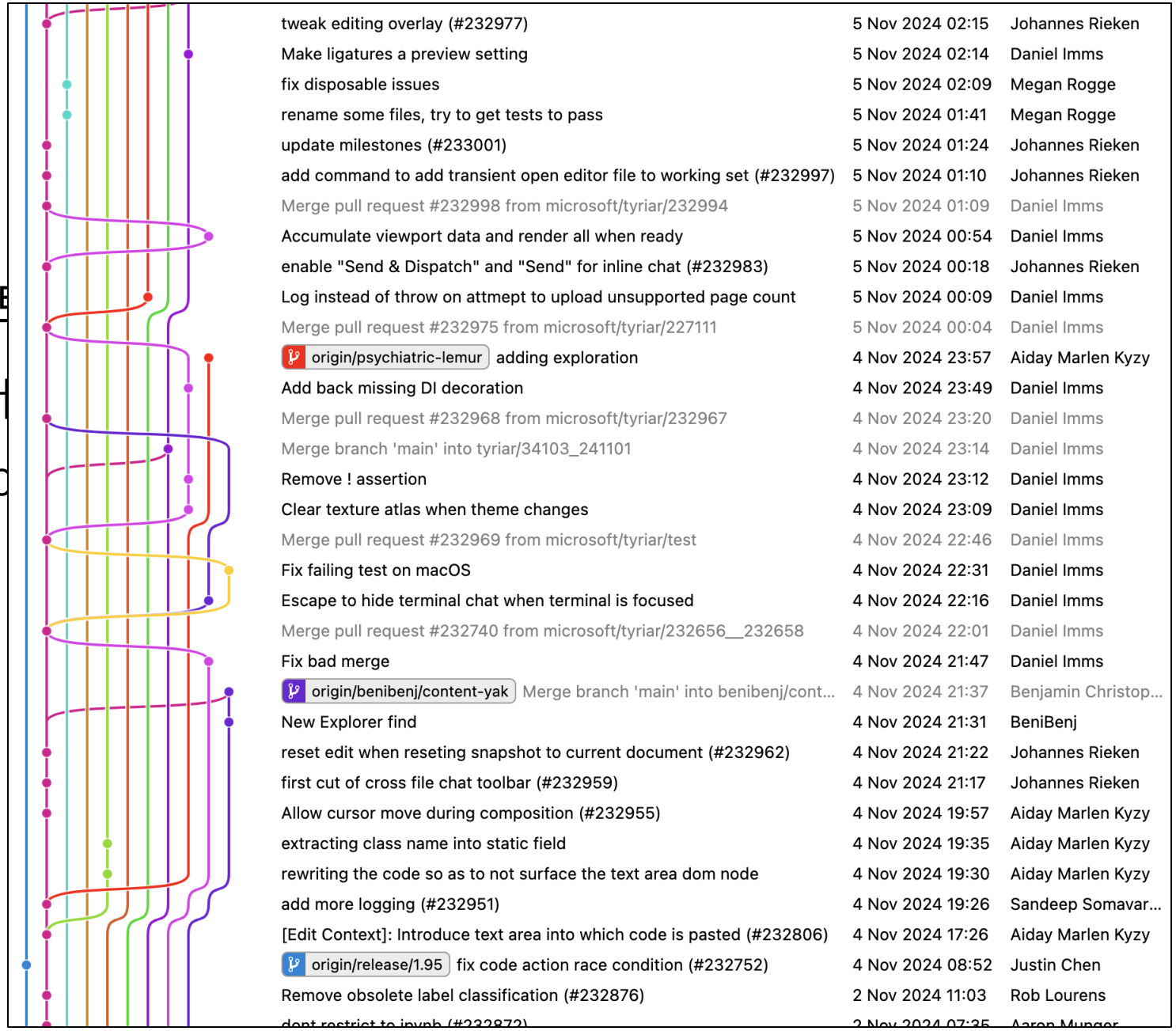
- 커밋 히스토리 (Commit History)
  - 원칙적으로 하나의 커밋은 의미있는 하나의 변경사항을 의미.
  - 커밋들이 모여서 시간 순으로 정렬된 것을 Commit History 라고 함.
  - 커밋 히스토리가 중요한 이유
    - 버그가 발생했을 경우
      - 버그를 유발하는 코드와 관련된 커밋만 찾아서 코드의 변경사항을 빠르게 확인 가능
    - 래거시 코드를 수정해야 할 때
      - 당시의 개발자가 어떤 의도로 코드를 고쳤는지 기록해 놓은 커밋 외에 의지할 곳이 없을 수도 있음.

Merge

- 커밋 히스토리 (Commit History)
  - 실무에서는 오타 수정과 같은 사소한 커밋을 하는 경우도 많다.
  - 너무 많아지면...

Merge

- 커밋 히스
  - 실무에서
  - 너무 많



tweak editing overlay (#232977)	5 Nov 2024 02:15	Johannes Rieken
Make ligatures a preview setting	5 Nov 2024 02:14	Daniel Imms
fix disposable issues	5 Nov 2024 02:09	Megan Rogge
rename some files, try to get tests to pass	5 Nov 2024 01:41	Megan Rogge
update milestones (#233001)	5 Nov 2024 01:24	Johannes Rieken
add command to add transient open editor file to working set (#232997)	5 Nov 2024 01:10	Johannes Rieken
Merge pull request #232998 from microsoft/tyriar/232994	5 Nov 2024 01:09	Daniel Imms
Accumulate viewport data and render all when ready	5 Nov 2024 00:54	Daniel Imms
enable "Send & Dispatch" and "Send" for inline chat (#232983)	5 Nov 2024 00:18	Johannes Rieken
Log instead of throw on attmept to upload unsupported page count	5 Nov 2024 00:09	Daniel Imms
Merge pull request #232975 from microsoft/tyriar/227111	5 Nov 2024 00:04	Daniel Imms
origin/psychiatric-lemur adding exploration	4 Nov 2024 23:57	Aiday Marlen Kyzy
Add back missing DI decoration	4 Nov 2024 23:49	Daniel Imms
Merge pull request #232968 from microsoft/tyriar/232967	4 Nov 2024 23:20	Daniel Imms
Merge branch 'main' into tyriar/34103_241101	4 Nov 2024 23:14	Daniel Imms
Remove ! assertion	4 Nov 2024 23:12	Daniel Imms
Clear texture atlas when theme changes	4 Nov 2024 23:09	Daniel Imms
Merge pull request #232969 from microsoft/tyriar/test	4 Nov 2024 22:46	Daniel Imms
Fix failing test on macOS	4 Nov 2024 22:31	Daniel Imms
Escape to hide terminal chat when terminal is focused	4 Nov 2024 22:16	Daniel Imms
Merge pull request #232740 from microsoft/tyriar/232656__232658	4 Nov 2024 22:01	Daniel Imms
Fix bad merge	4 Nov 2024 21:47	Daniel Imms
origin/benibenj/content-yak Merge branch 'main' into benibenj/cont...	4 Nov 2024 21:37	Benjamin Christop...
New Explorer find	4 Nov 2024 21:31	BeniBenj
reset edit when reseting snapshot to current document (#232962)	4 Nov 2024 21:22	Johannes Rieken
first cut of cross file chat toolbar (#232959)	4 Nov 2024 21:17	Johannes Rieken
Allow cursor move during composition (#232955)	4 Nov 2024 19:57	Aiday Marlen Kyzy
extracting class name into static field	4 Nov 2024 19:35	Aiday Marlen Kyzy
rewriting the code so as to not surface the text area dom node	4 Nov 2024 19:30	Aiday Marlen Kyzy
add more logging (#232951)	4 Nov 2024 19:26	Sandeep Somavar...
[Edit Context]: Introduce text area into which code is pasted (#232806)	4 Nov 2024 17:26	Aiday Marlen Kyzy
origin/release/1.95 fix code action race condition (#232752)	4 Nov 2024 08:52	Justin Chen
Remove obsolete label classification (#232876)	2 Nov 2024 11:03	Rob Lourens
dont restrict to ipynb (#232872)	2 Nov 2024 07:35	Aaron Munger

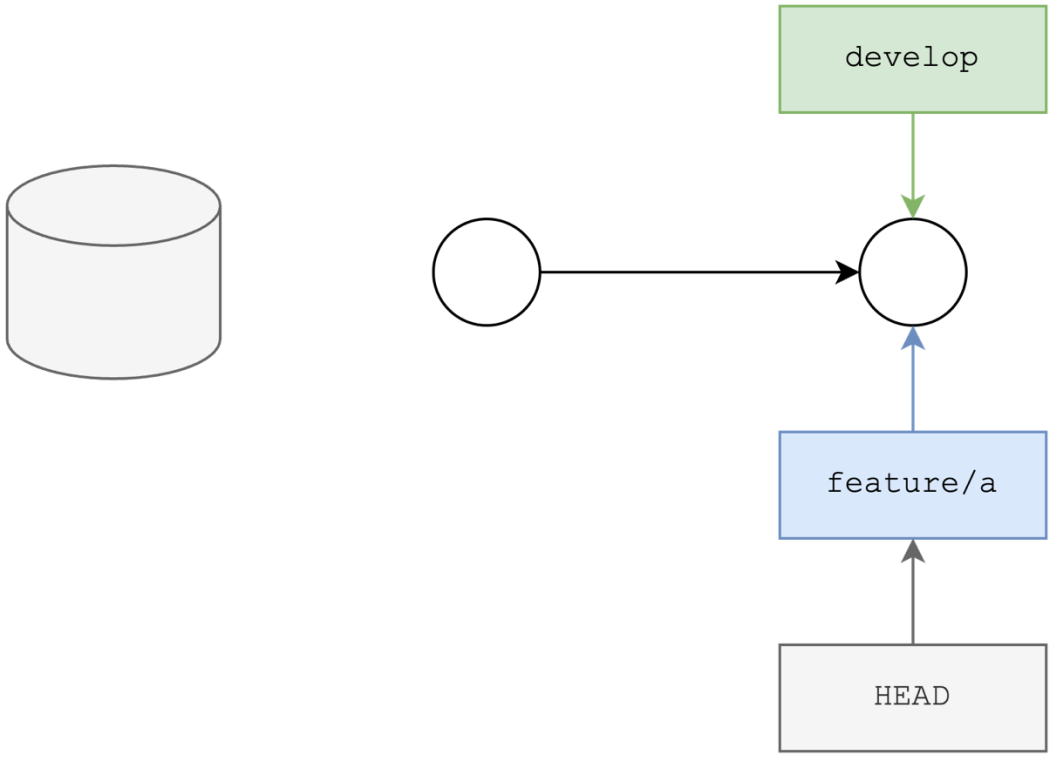
Git Merge의 전략

- Git 에서 한 브랜치에서 다른 브랜치로 합치는 전략에는 두 가지가 있다.

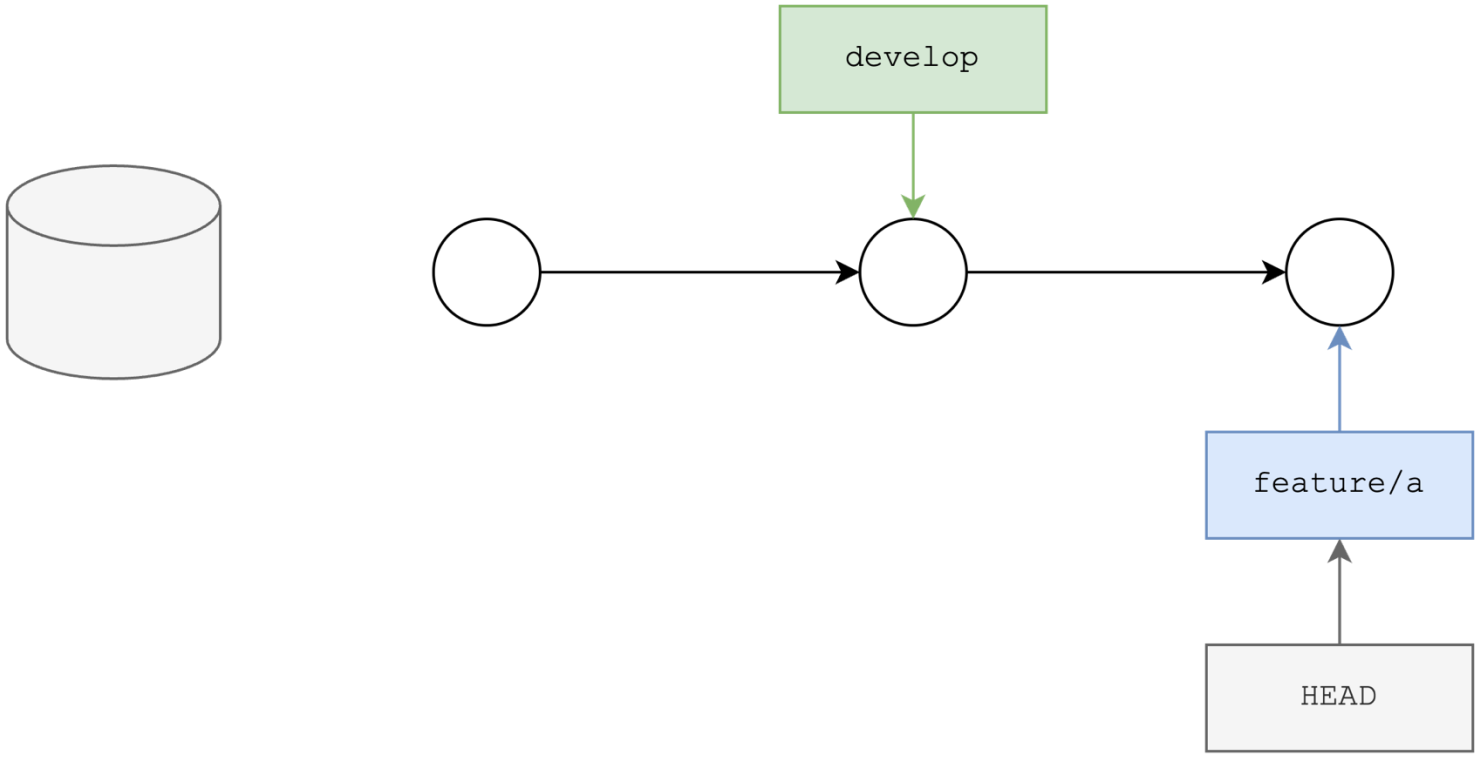
Fast forward

3-way merge

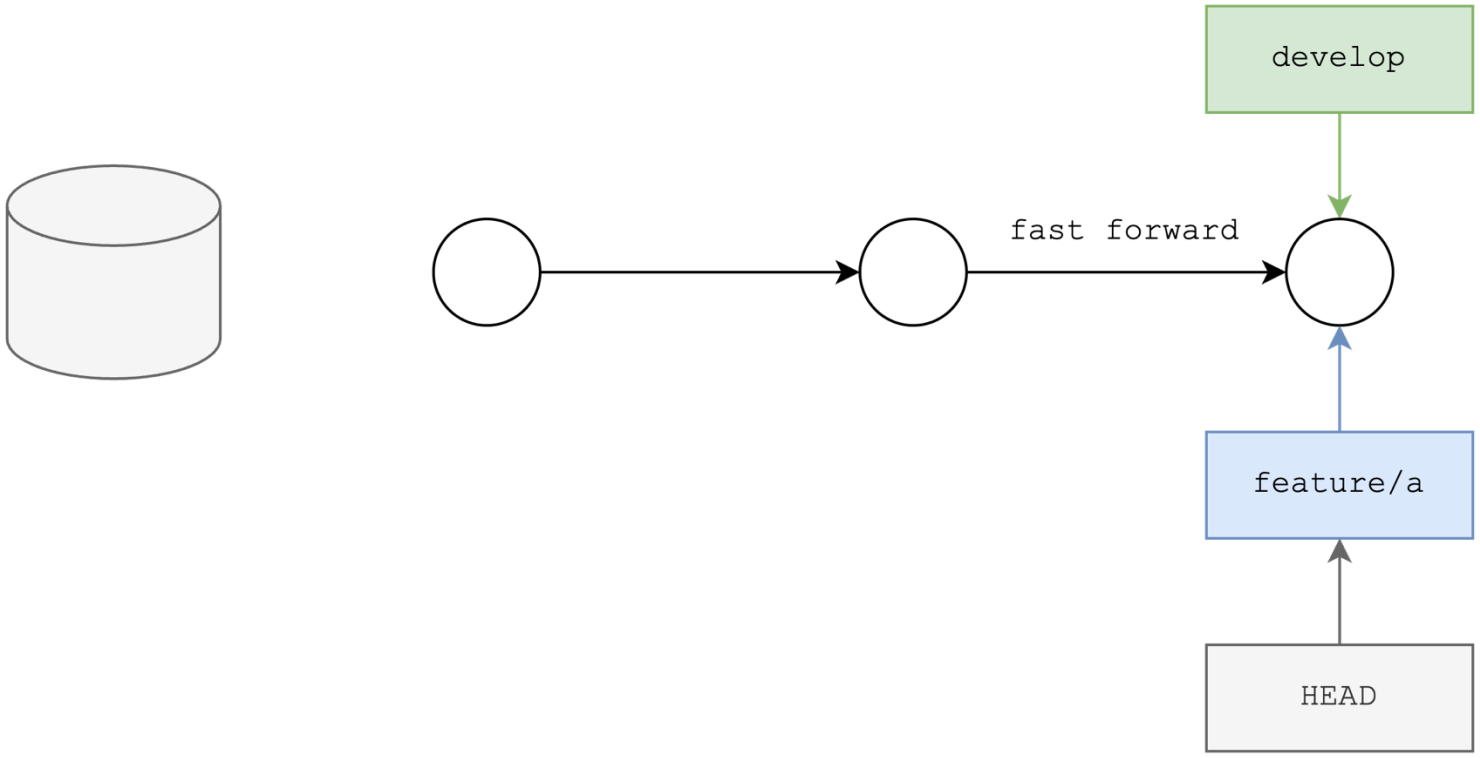
Fast forward



Fast forward



Fast forward

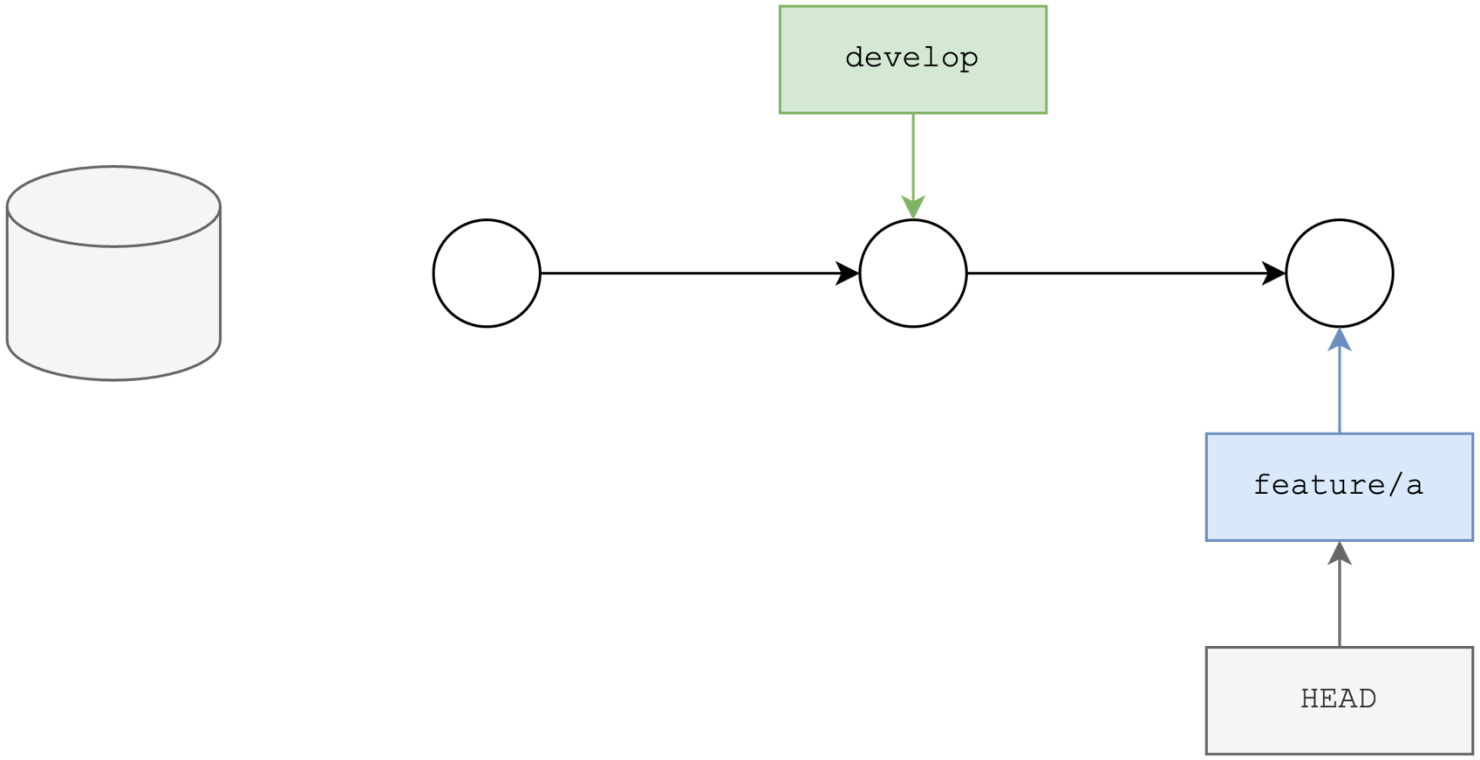


Fast forward

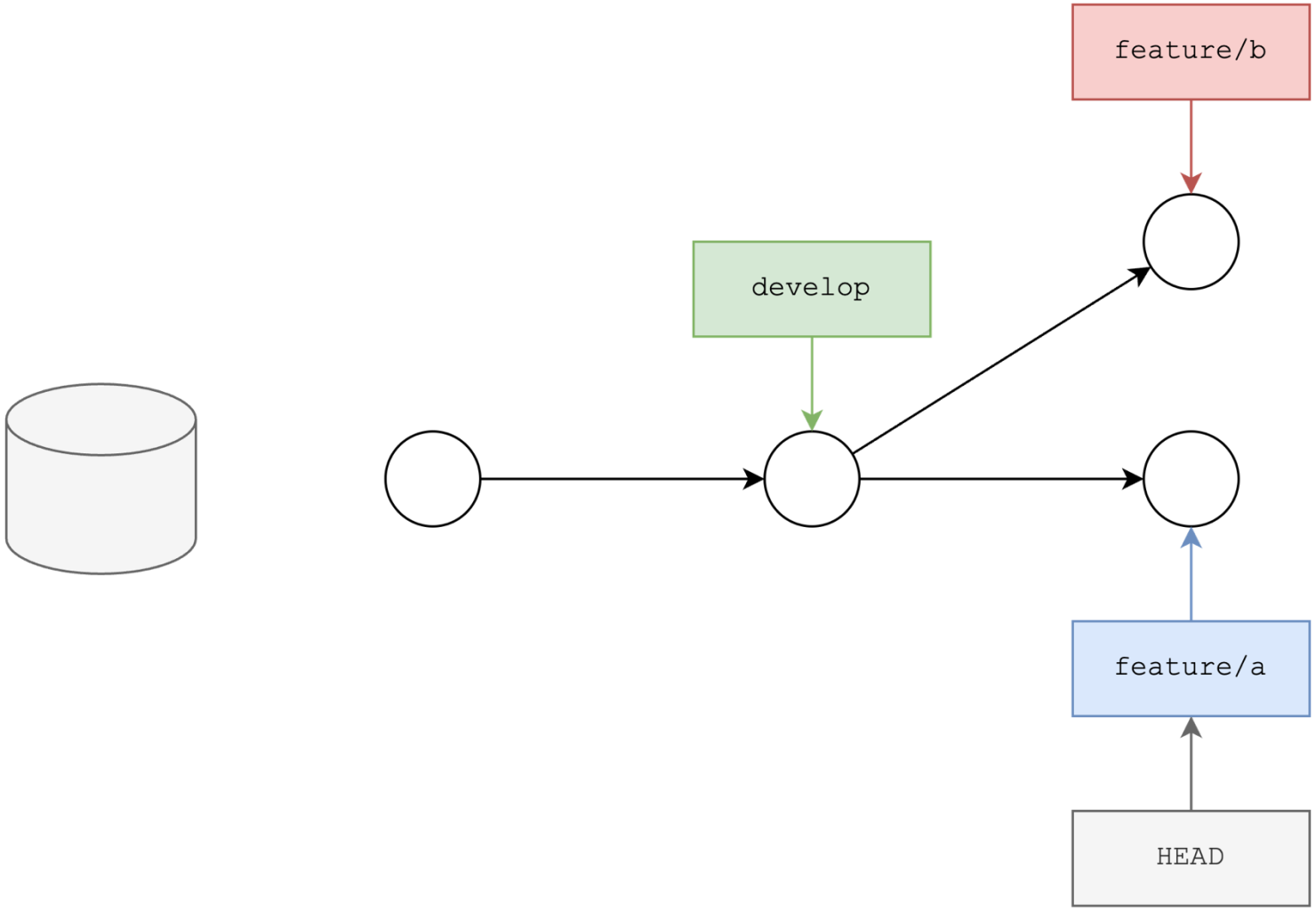
- 서로 다른 상태를 병합하는 것이 아닌, Destination 브랜치를 Source 브랜치 위치로 이동만 해도 되는 상태에서 사용.

- Merge를 위한 커밋이 생성되지 않음.

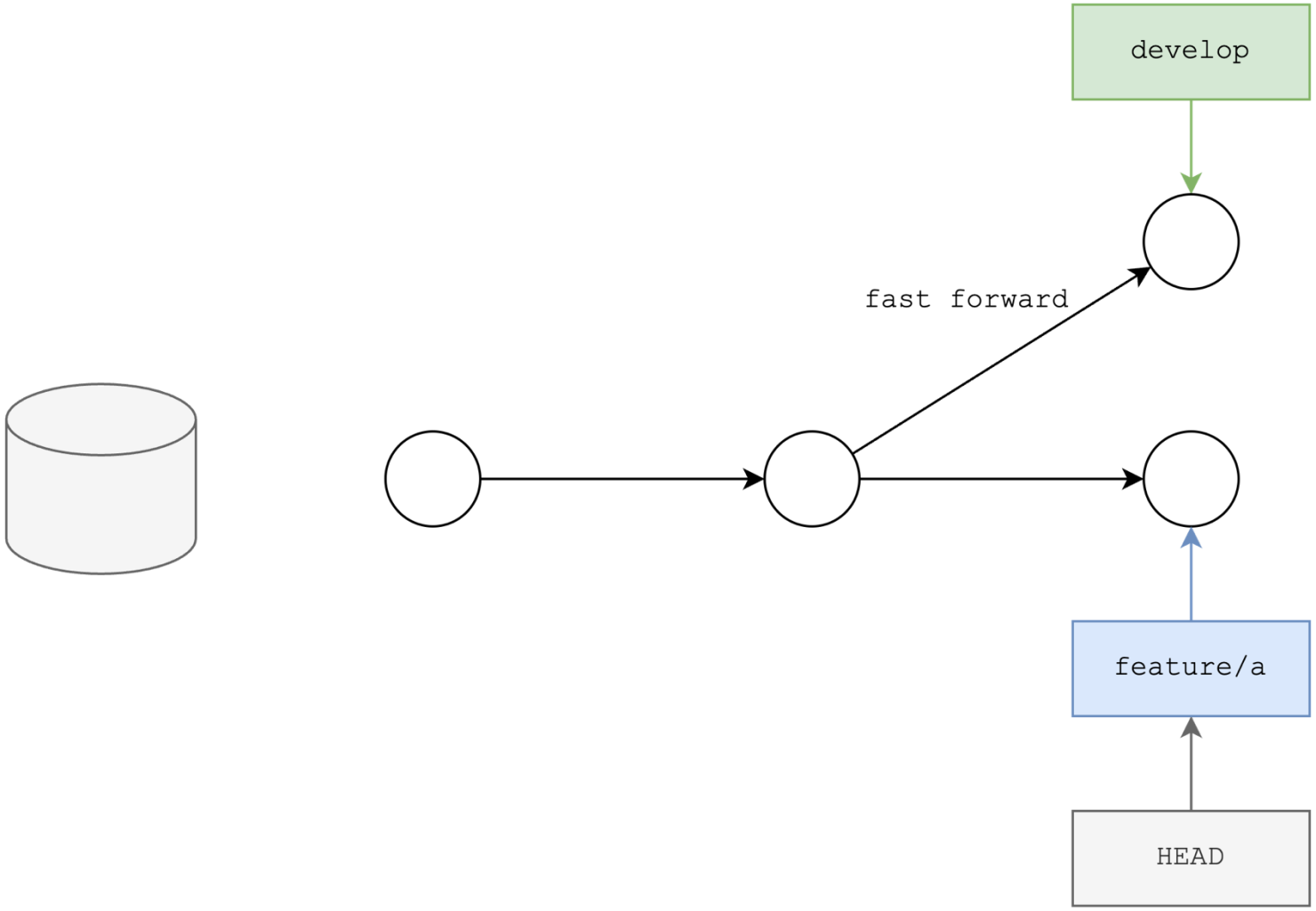
3-way merge



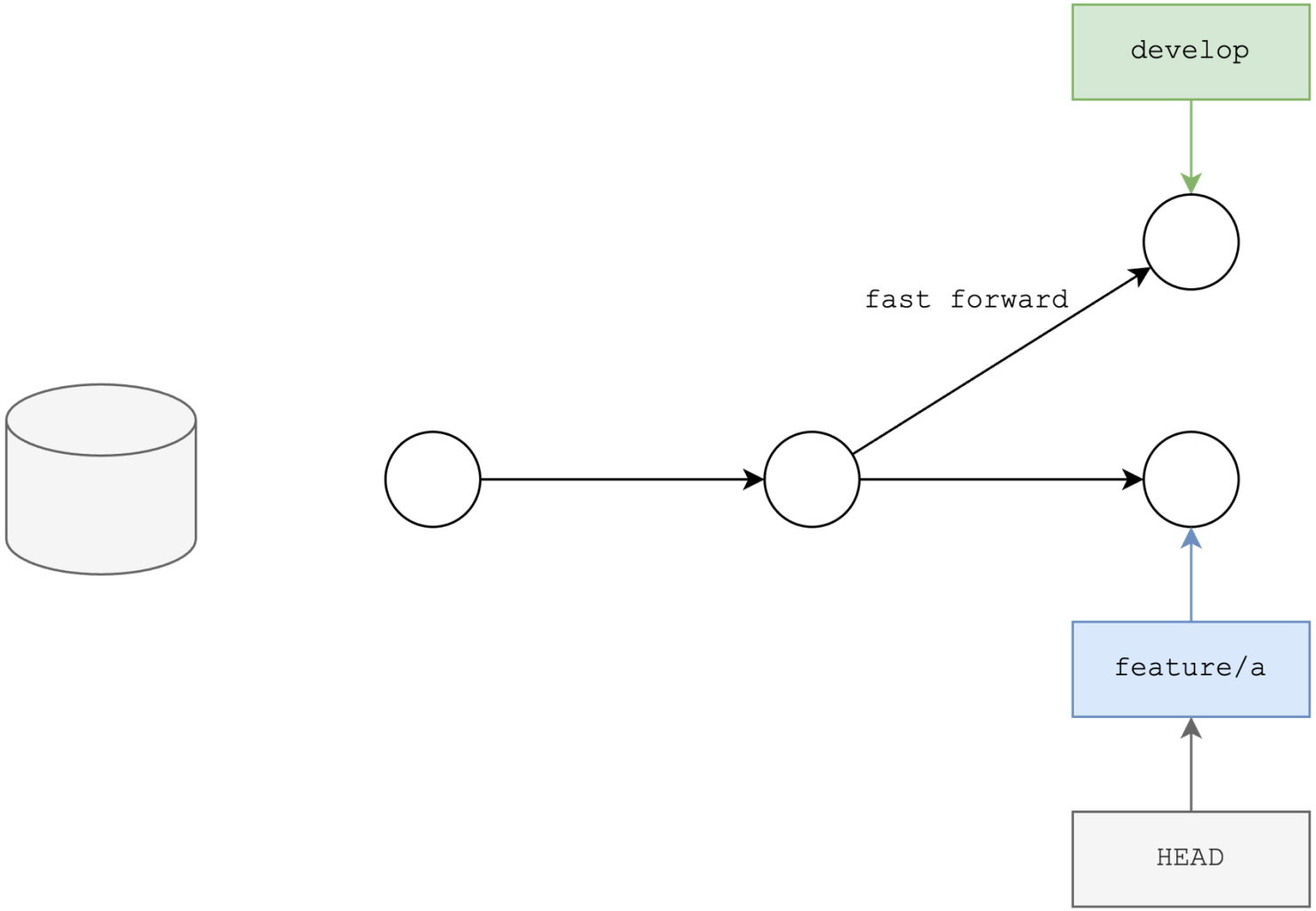
3-way merge



3-way merge



3-way merge

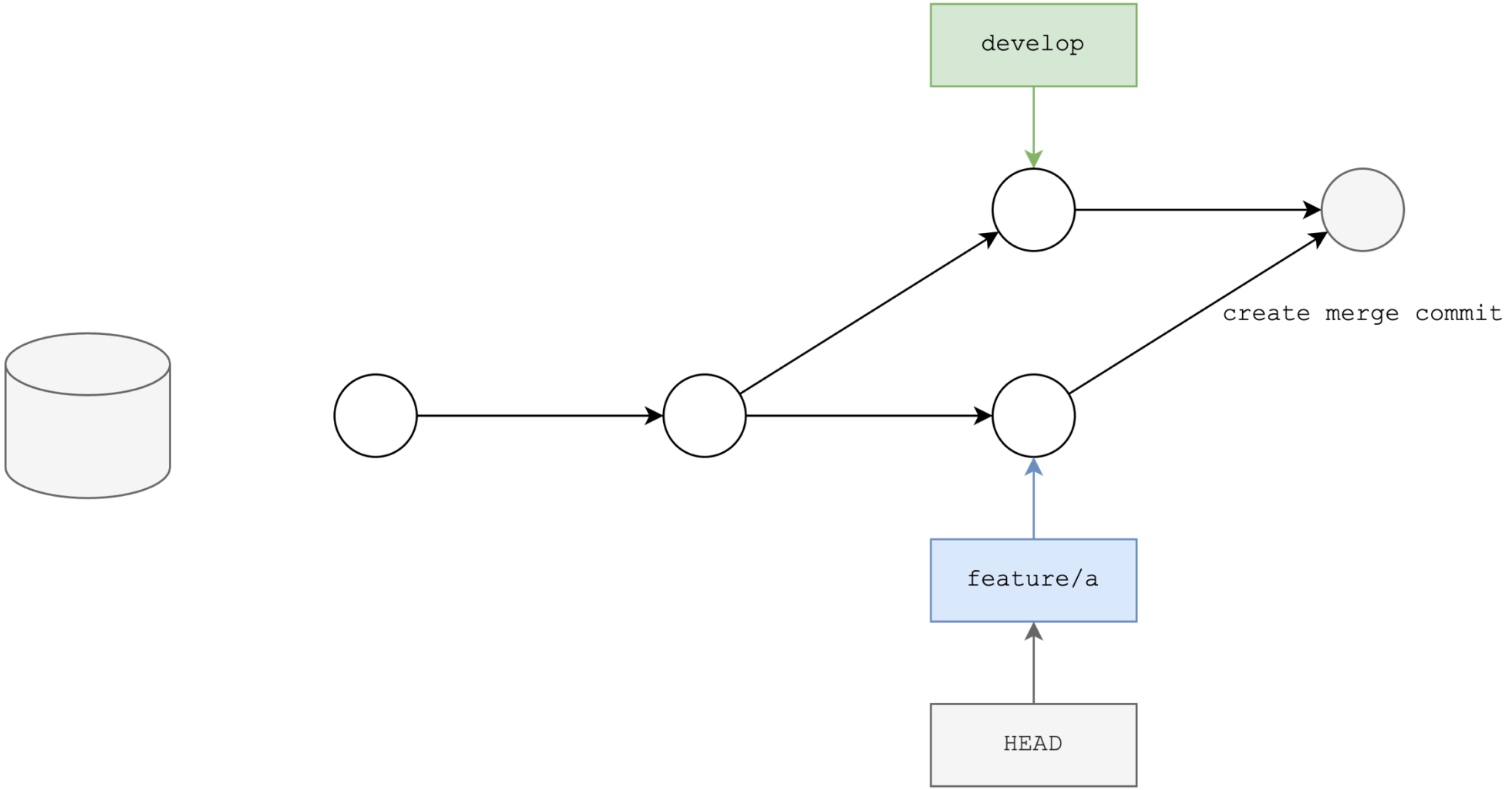


Fast-forward 불가능

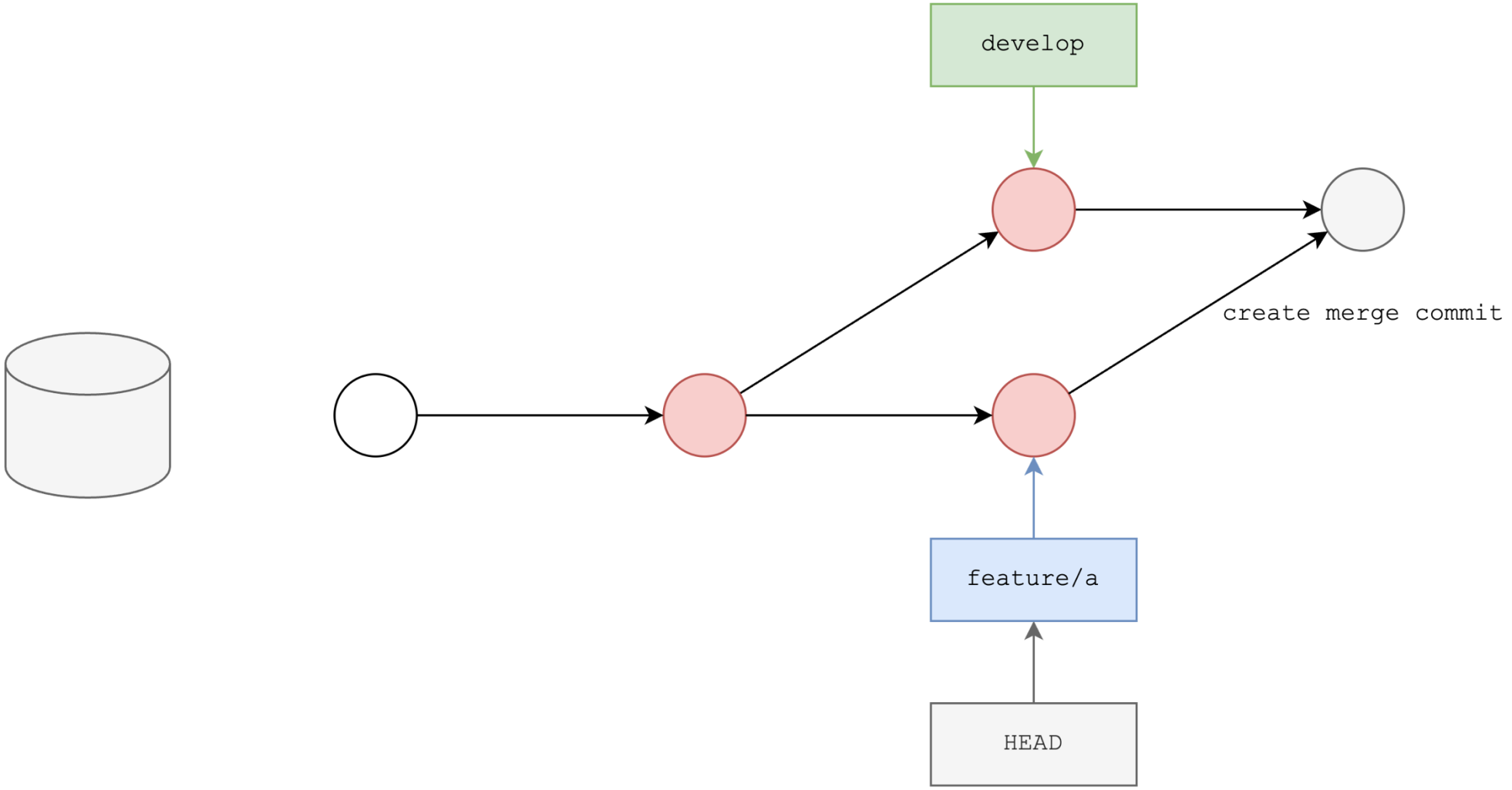
3-way merge

- 3가지를 기준으로 커밋을 병합
  1. 내가 있는 브랜치의 커밋 (Current)
  2. 남의 브랜치 커밋 (In-coming)
  3. 두 브랜치의 공통 조상이 되는 커밋 (Base)

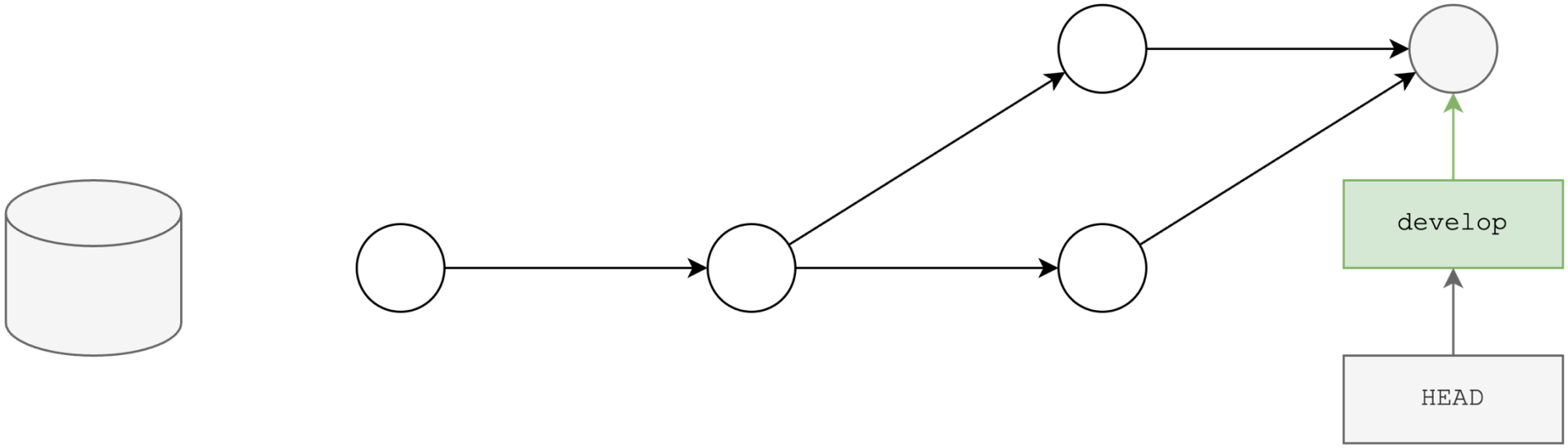
3-way merge

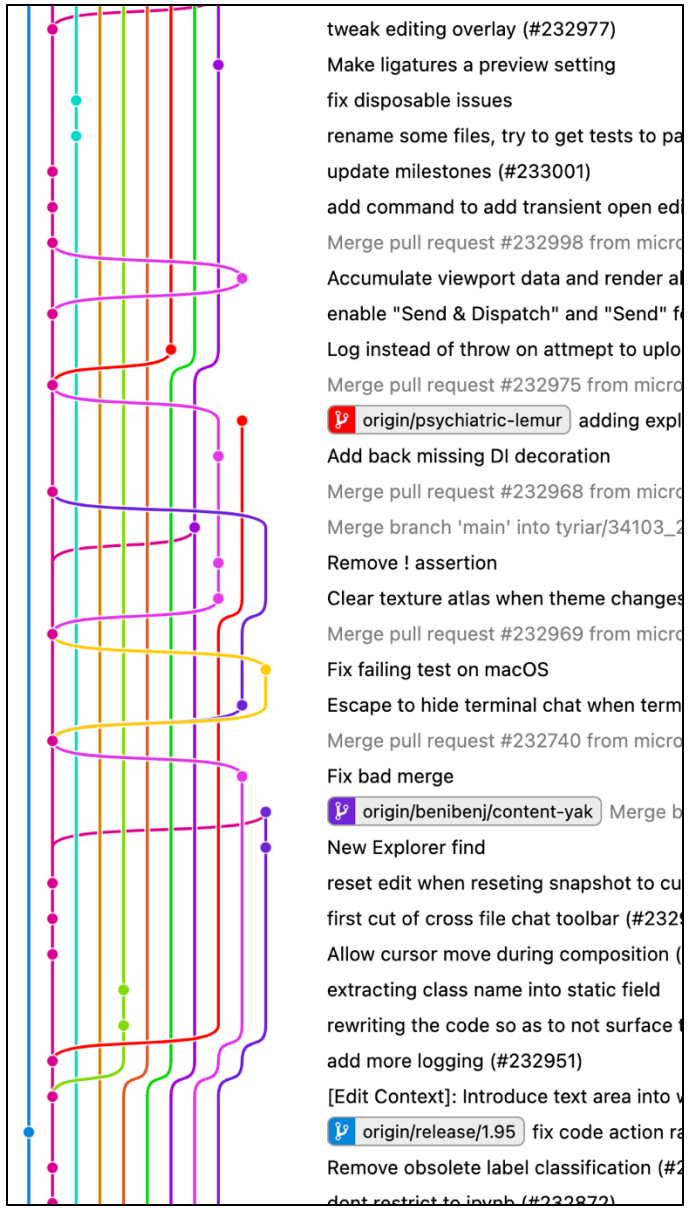


3-way merge



3-way merge





- 협업 실무에서는 3-way merge를 자주보게 될 것.
- 굉장히 정보량이 많아 추적이 어려운...

1) --ff
$ git merge {병합할 브랜치 명}
merge를 진행할 때 아무 옵션을 주지 않는 경우이며, 위에서 살펴보았던 Merge 방법과 동일하다.
현 브랜치와 병합할 브랜치가 Fast-forward 관계이면 Fast-forward 병합을 진행하며, 그렇지 않은 경우는 Merge 커밋을 생성하여 3 way-merge를 진행한다.

2) --no-ff
$ git merge --no-ff {병합할 브랜치 명}
현재 브랜치와 병합 대상의 관계가 Fast-forward관계 여부와 상관없이
Merge 커밋을 생성하여 병합한다.

3) --ff-only
$ git merge --ff-only {병합할 브랜치 명}
현재 브랜치와 병합 대상의 관계가 Fast-forward인 경우에만 병합을 진행한다.
Merge 커밋 생성되지 않는다.

4) --squash
$ git merge --squash {병합할 브랜치 명}

Git Rebase

Rebase에 대해서 빠르게 알아보자

Git Rebase란?

- Git 에서 한 브랜치에서 다른 브랜치로 합치는 방법으로는 두 가지가 있다.

Merge

Rebase

Git Rebase란?

- Git 에서 한 브랜치에서 다른 브랜치로 합치는 방법으로는 두 가지가 있다.

Merge

3-way merge

fast-forward

Rebase

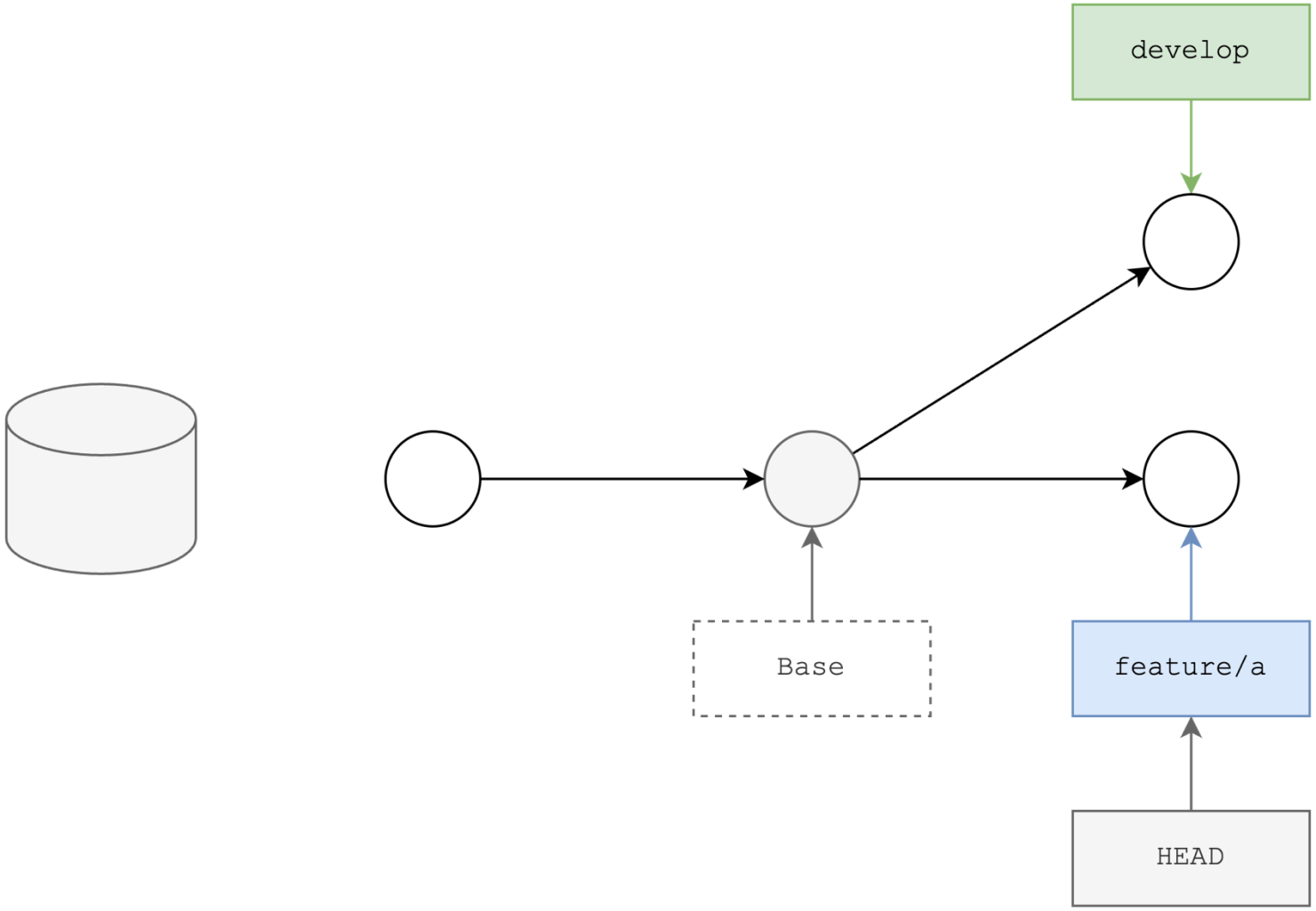
???

Git Rebase란?

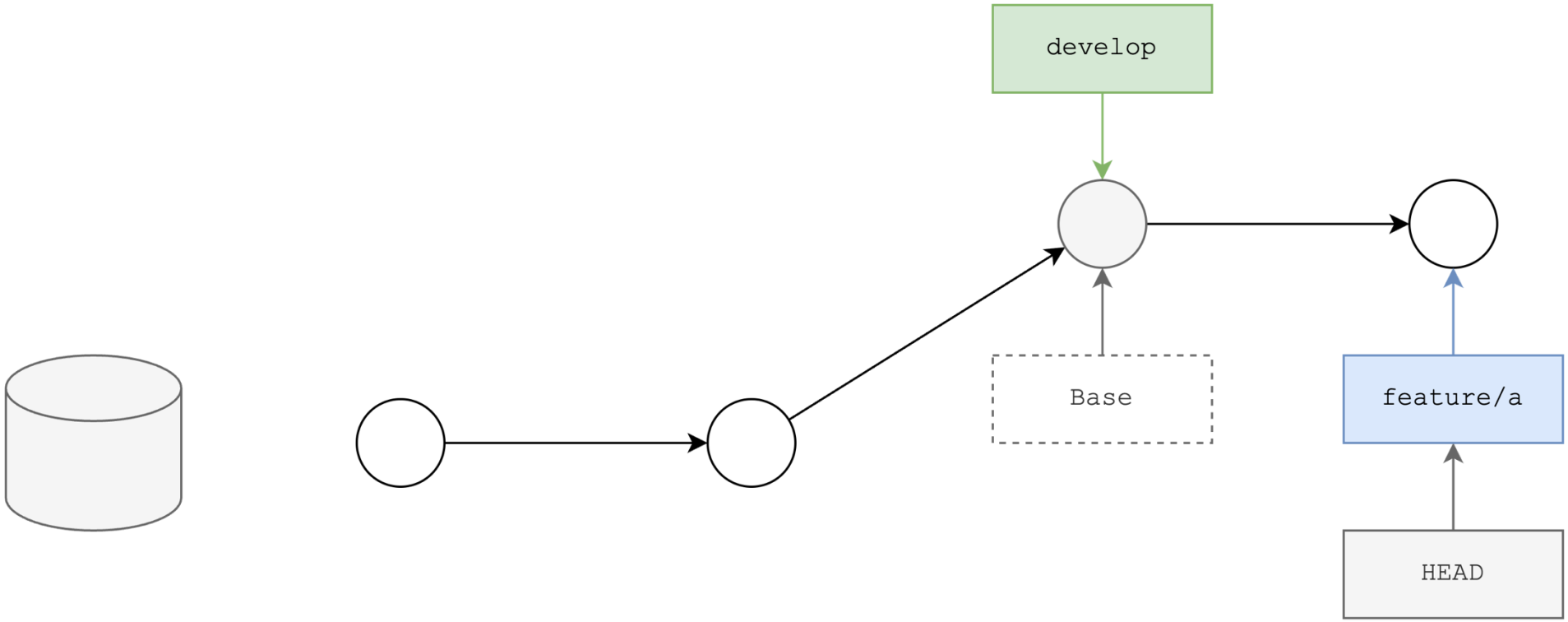
- Re(다시) + Base(기반, 여기서는 공통조상) (..을 지정하다)

- 3-Way Merge 에서 공통조상(Base)을 다른 브랜치로 재설정한다.

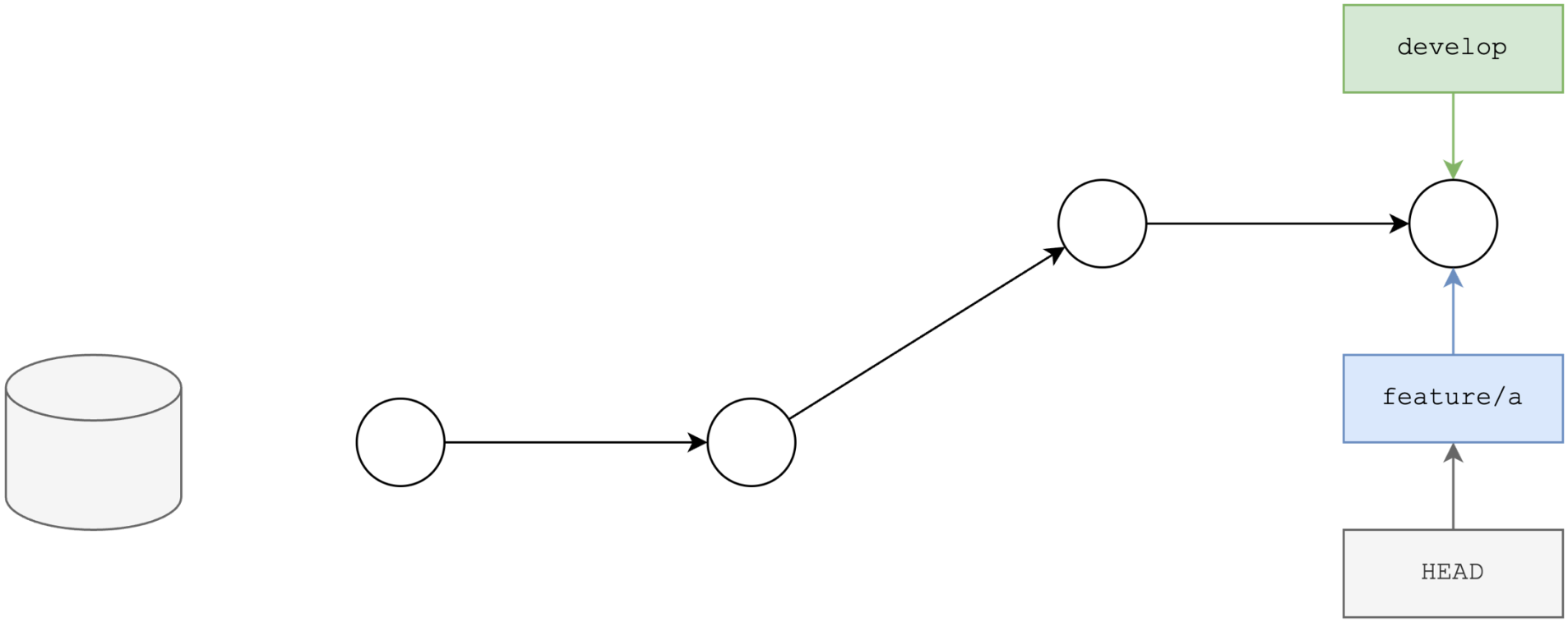
Rebase



Rebase



Rebase



VS Code as Git Editor

VS Code를 기본 git 편집기로 설정하기

https://stackoverflow.com/questions/30024353/how-to-use-visual-studio-code-as-default-editor-for-git

(공통) Rebase 실습

- 요구사항:
  - Main 브랜치에 Develop 브랜치를 Merge 해보자.

- 조건:
  - Merge 후, 커밋 히스토리에 가지가 갈라진 것이 있으면 안된다.

- 실습 레포지토리
  - https://github.com/smu-oss/w4-practice-01

Git Rebase의 세부 명령 조합

- Rebase에는 더 다양한 기능이 존재한다.

- 상호작용(Interactive) 옵션을 통해서 사용가능하다.
  - Squash
  - Edit
  - Rename
  - Drop

```
git rebase -i <시작커밋해쉬 | 브랜치명>
```

Git 프로 꿀팁

- git commit 에 --amend 명령을 붙여서 직전 커밋을 수행할 수 있다.
- 커밋 메시지의 가장 끝에 “Co-Author-By: 이름 <이메일>” 을 추가하면 공동 커밋 작성자로 여러 명을 등록 할 수 있다.

```
git rebase -i <시작커밋해쉬 | 브랜치명>
```

Git Rebase 실습

팀전 (최대 2인)

문제 Pool

- [Lv.1] 빠른 오타 발견 (1pt.)
- ~~[Lv.1] 불필요한 녀석이 숨어있다~~ (2pt.)
- [Lv.2] 커밋이 너무 많다 (3pt.)
- [Lv.2] 화가 난 공동 작업자 (3pt.)
- ~~[Lv.2] 근본 없는 녀석~~ (3pt.)
- ~~[Lv.3] 대규모 Merge Attack을 당하다~~ (5pt.)
- ~~[Lv.3] Case-Insensitive~~ (6pt.)
- [Lv.3] 앗차차, 너무 늦은 오타 발견 (6pt.)

[Lv.1] 빠른 오타 발견 (1pt.)

- 요구사항
  - 레포지토리의 얼굴 README에 대문짝만한 [!!!오타!!!] 가 발견되었다.
  - 오타를 제거해달라는 마켓팅 팀의 요청이 들어왔으니 오타를 처리하자.

- 조건
  - 오직 오타는 "[!!!오타!!!]" 와 정확히 글자가 일치하는 것만을 오타라고 한다.
  - README.adoc 에서 "[!!!오타!!!]"를 제거하면 오타를 고친 것으로 본다.
  - git commit 명령만을 사용하여 수정한다.
    - git commit 의 옵션을 사용하는 것은 허용
    - 단 , 다른 명령어는 사용하면 안된다.

- 실습 레포지토리
  - https://github.com/smu-oss/w4-mission-we-have-typos

[Lv.2] 커밋이 너무 많다 (3pt.)

- 요구사항
  - 커밋이 너무 많아 커밋 수를 줄이려고 한다.
  - 연속으로 배치된 동일한 메시지의 커밋들을 서로 합칠 것.

- 조건
  - 커밋메시지가 다른 커밋끼리 합쳐서는 안된다.
  - 커밋들을 합친 후에도 전체 변경사항은 동일해야 한다.

- 실습 레포지토리
  - https://github.com/smu-oss/w4-mission-too-many-commits

[Lv.2] 화가 난 공동 작성자 (3pt.)

- 요구사항
  - 내 근처에 앉은 팀원이 자신도 실습에 참여했으니 기여자로 등록해달라고 한다.
  - 커밋의 저자로 해당 팀원을 추가해주자

- 조건
  - 커밋의 수와 히스토리 상에서의 순서는 유지되어야 한다.
  - 오직 커밋 메시지만을 변경하여 저자를 추가하자.

- 실습 레포지토리
  - https://github.com/smu-oss/w4-practice-01

[Lv.3] 앗차차, 너무 늦은 오타 발견 (6pt.)

- 요구사항
  - 커밋 히스토리 곳곳에 [!!!오타!!!] 가 발견되었다.
  - 모든 오타를 제거해달라는 마켓팅 팀의 요청이 들어왔으니 모든 오타를 처리하자.

- 조건
  - 오직 오타는 "[!!!오타!!!]" 와 정확히 글자가 일치하는 것만을 오타라고 한다.
  - 모든 파일에서 에서 "[!!!오타!!!]"를 제거하면 오타를 고친 것으로 본다.
  - git rebase 와 git commit --amend 명령만을 사용하여 작업을 수행한다.

- 실습 레포지토리
  - https://github.com/smu-oss/w4-mission-we-have-typos