Intermediate GIT Interview Questions

1. What has to be run to squash multiple commits (last N) into a single commit?

Squashing multiple commits to a single one overwrites the history which is why it is recommended to be done using full caution. This step can be done by running the command: git rebase -i HEAD~{{N}} where {{N}} represents the number of commits needed to be squashed.

2. How would you recover a branch that has already pushed changes in the central repository but has been accidentally deleted from every team member's local machines?

We can recover this by checking out the latest commit of this branch in the reflog and then checking it out as a new branch.

3. Can you tell something about git reflog?

This command tracks every single change made in the repository references (that can be branches or tags) and also maintains the branches/tags log history that was either created locally or checked out. Reference logs such as the commit snapshot of when the branch was created or cloned, checked-out, renamed, or any commits made on the branch are maintained by Git and listed by the 'reflog' command.

- This recovery of the branch is only possible when the branch was either created locally or checked-out from a remote repository in your local repository for Git to store its reference history logs.
- This command should be executed in the repository that had the lost branch.

4. What consists of a commit object?

A commit object consists of the following components:

- A set of files that represents the state of a project at a given point in time.
- Reference to parent commit objects.
- A 40 character string termed as SHA-1 name uniquely identifies the commit object.

5. Explain the levels in git config and how can you configure values using them?

- In order to make git work, it uses a set of configurations that are pre-defined by default by means of configuration files (or config files). We can change the default behavior of git by just modifying these files which are basically text files. In order to do this, it is important to understand how git identifies these files. It does so by following the below steps:

  - Firstly, git searches for the config values in the system-wide gitconfig file stored in <<installation_path>>/etc/gitconfig file that has settings defined and applied to **every user** of the system and all their repos.
    - In case you want git to search from this particular file and read/write on it, we can pass the option --system to git config command.

  - Next, git searches for the ~/.gitconfig file or ~/.config/git/config that has the scope specific to the user.
    - Git can be made to read/ write from this file specifically bypassing --global to the git config command.

  - Lastly, git searches for the config values in the git directory of the local repository that we are currently working on.
    - These config values are specific to that particular repository alone and can be accessed by passing --local to the git config command.This is the default config file that gets accessed and modified upon in case we do not specify any levels.

6. What is a detached HEAD and what causes this and how to avoid this?

Detached HEAD indicates that the currently checked-out repository is not a local branch. This can be caused by the following scenarios:

- When a branch is a read-only branch and we try to create a commit to that branch, then the commits can be termed as “free-floating” commits not connected to any branch. They would be in a detached state.
- When we checkout a tag or a specific commit and then we try to perform a new commit, then again the commits would not be connected to any branch. When we now try to checkout a branch, these new commits would be automatically placed at the top.

  In order to ensure that detached state doesn't happen, =instead of checking out commit/tag, we can create a branch emanating from that commit and then we can switch to that newly created branch by using the command: git checkout -b <<new_branch_name>>. This ensures that a new branch is checkout out and not a commit/tag thereby ensuring that a detached state wouldn't happen.

7. What does git annotate command do?

- This command annotates each line within the given file with information from the commit which introduced that change. This command can also optionally annotate from a given revision.
- Syntax: git annotate [<options>] <file> [<revision>]
- You can get to learn more about this command from the official git documentation here.

8. What is the difference between git stash apply vs git stash pop command?

- git stash pop command throws away the specified stash (topmost stash by default) after applying it.
- git stash apply command leaves the stash in the stash list for future reuse. In case we wanted to remove it from the list, we can use the git stash drop command.

git stash pop = git stash apply + git stash drop

9. What do the git diff and git status commands do?

e changes **between commits, wor**	e difference **between the working directory and index** that is essen ; git in depth.

- git diff works in a similar fashion to git status with the only difference of showing the differences between commits and also between the working directory and index.

10. Why is it considered to be easy to work on Git?

With the help of git, developers have gained many advantages in terms of performing the development process faster and in a more efficient manner. Some of the main features of git which has made it easier to work are:

- **Branching Capabilities:**

  - Due to its sophisticated branching capabilities, developers can easily work on multiple branches for the different features of the project.
  - It also has an easier merge option along with an efficient work-flow feature diagram for tracking it.

- **Distributed manner of development:**

  - Git is a distributed system and due to this nature, it became easier to trace and locate data if it's lost from the main server.
  - In this system, the developer gets a repository file that is present on the server. Along with this file, a copy of this is also stored in the developer’s system which is called a local repository.

- Due to this, the scalability of the project gets drastically improved.

- **Pull requests feature:**

  - This feature helps in easier interaction amongst the developers of a team to coordinate merge-operations.
  - It keeps a proper track of the changes done by developers to the code.

- **Effective release cycle:**

  - Due to the presence of a wide variety of features, git helps to increase the speed of the release cycle and helps to improve the project workflow in an efficient manner.

11. How will you create a git repository?

- Have git installed in your system.
- Then in order to create a git repository, create a folder for the project and then run git init.
- Doing this will create a .git file in the project folder which indicates that the repository has been created.

12. Tell me something about git stash?

Git stash can be used in cases where we need to switch in between branches and at the same time not wanting to lose edits in the current branch. Running the git stash command basically pushes the current working directory state and index to the stack for future use and thereby providing a clean working directory for other tasks.

13. What is the command used to delete a branch?

- To delete a branch we can simply use the command git branch –d [head].
- To delete a branch locally, we can simply run the command: git branch -d <local_branch_name>
- To delete a branch remotely, run the command: git push origin --delete <remote_branch_name>
- Deleting a branching scenario occurs for multiple reasons. One such reason is to get rid of the feature branches once it has been merged into the development branch.

14. What differentiates between the commands git remote and git clone?

git remote command creates an entry in git config that specifies a name for a particular URL. Whereas git clone creates a new git repository by copying an existing one located at the URL.

15. What does git stash apply command do?

- git stash apply command is used for bringing the works back to the working directory from the stack where the changes were stashed using git stash command.
- This helps the developers to resume their work where they had last left their work before switching to other branches.

16. Differentiate between git pull and git fetch.

l pulls new changes from the currently working b central repository.	l is also used for a similar purpose but it follows a tw mits and changes from desired branch and stores th local repository. to be reflected in the current / target branch, git fet t merge command.

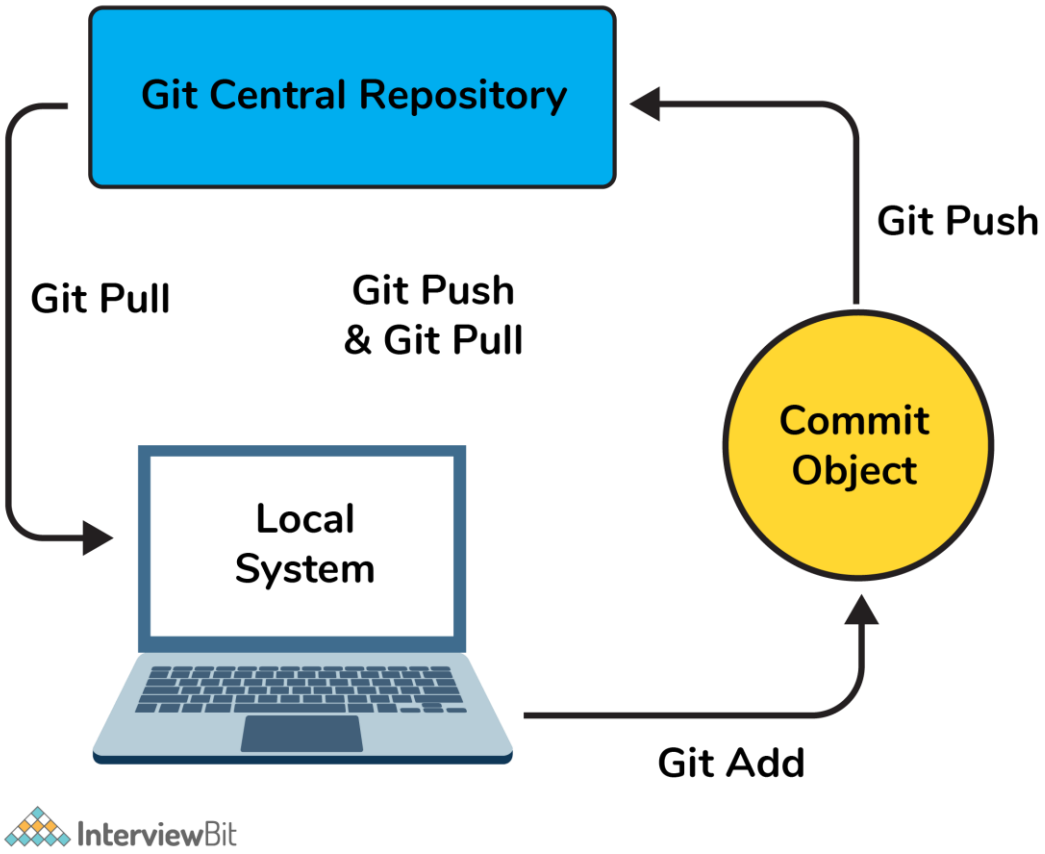
git pull = git fetch + git merge

17. Can you give differences between “pull request” and “branch”?

done when there is a need to put a developer’s change into another	thing but a separate version of

18. Why do we not call git “pull request” as “push request”?

- “Push request” is termed so because it is done when the target repository requests us to push our changes to it.
- “Pull request” is named as such due to the fact that the repo requests the target repository to grab (or pull) the changes from it.



19. Can you tell the difference between Git and GitHub?

Git	GitHUB
This is a distributed version control system **installed on local machines** which allow developers to keep track of commit histories and collaborative work.	This is a **cloud-based source code repository** developed by using git.
This is maintained by “The Linux Foundation”.	This was acquired by “Microsoft”
SVN, Mercurial, etc are the competitors.	GitLab, Atlassian BitBucket, etc are the competitors.

- GitHub provides a variety of services like forking, user management, etc along with providing a central repository for collaborative work.