1.What is Git and why is it used?

Ans: Git is a dev op tool or an open source tool used for code management .It is a free and open-source version control system used to handle small to very large projects efficiently. Git is used to track any changes made in the source code, enabling multiple developers to work together on non-linear development.

2.Explain the difference between Git pull and Git fetch.

Git	Git fetch
1.It is used to fetch all changes from the remote repository to the local repository without merging into the current working directory	**1.** It bring the copy of all the chances that are made in local repository and merges them into the working repository
2. data is updated in .git directory	2. data is update directly in directory
3. no possibility of merge conflict	**3.** there can be merge conflict
4. Command for Git fetch is git fetch<remote>	**4.** Command for Git Pull is git pull<remote><branch>

3.How do you revert a commit in Git?

STEPS TO REVERT A COMMIT IN GIT:

1. **Use the git log or ref log to find the id of the commit.**
2. **Enter the git revert command.**
3. **Provide an alternative git commit message.**

4.Describe the Git staging area.

In technical terms, the staging area is the middle ground between what you have done to your files (also known as the working directory) and what you had last committed (the HEAD commit). As the name implies, the staging area gives you space to prepare (stage) the changes that will be reflected on the next commit.

5.What is a merge conflict, and how can it be resolved?

A git merge conflict is an event that takes place when Git is unable to automatically resolve differences in code between two commits. Git can merge the changes automatically only if the commits are on different lines or branches.

There are a few steps that could reduce the steps needed to resolve merge conflicts in Git.

Step 1: The easiest way to resolve a conflicted file is to open it and make any necessary changes.

Step 2: After editing the file, we can use the git add a command to stage the new merged content.

Step 3: The final step is to create a new commit with the help of the git commit command.

Step 4: Git will create a new merge commit to finalize the merge.

6.How does Git branching contribute to collaboration?

Git's branching and merging capabilities enable parallel development. Each team member can create their own branch to work on specific features or fixes without disrupting the main project. Once their changes are ready, these branches can be merged back into the main branch, optimizing collaboration and reducing conflicts.

7.What is the purpose of Git rebase ?

rebasing is changing the base of your branch from one commit to another making it appear as if you'd created your branch from a different commit. Internally, Git accomplishes this by creating new commits and applying them to the specified base. It's very important to understand that even though the branch looks the same, it's composed of entirely new commits.

8. Explain the difference between Git clone and Git fork

The difference between git clone and git fork is that git fork is a copy of someone else repository allows you to freely experiment with changes without affecting the original project basically to work on open-source project and Git clone is copy of the repository made on your local machine basically the project you work offline for.

9.How do you delete a branch in Git?

Branches can be deleted in 2 ways

1.Deleting a branch Locally

Delete a branch with git branch -d <branch>.

-d will delete the branch only if it is pushed and merged.

2.Deleting a branch REMOTELY

The command to delete a branch remotely: git push <remote> --delete <branch>.

10.What is a Git hook, and how can it be used?

Git hooks are scripts that run automatically every time a particular event occurs in a Git repository. They let you customize Git's internal behavior and trigger customizable actions at key points in the development life cycle.

Common use cases for Git hooks include encouraging a commit policy, altering the project environment depending on the state of the repository, and implementing continuous integration workflows. But, since scripts are infinitely customizable, you can use Git hooks to automate or optimize virtually any aspect of your development workflow.