Steps followed to maintain a project in Git

1. (Initial Project Set up and Commit with a PR)

* Fork the Project from Organizational GIT, Project will be available in Local GIT (User’s name)- This to be done single time for a Project.
* Also cloning can be done from Org GIT and can be placed in Local Machine.
* Once the Project is imported to Eclipse, New Branch can be created, and the coding can be done
* All the changes in the Local (Eclipse) that are done in a Branch– Commit

git commit -m 'The initial commit of my project'

* Synchronize.
* Switch to Master.
* Merge the Local and select the option (If not fast forward fail)
* Synchronize.
* Push to Local GIT.
* Raise a Pull Request.

1. Merge

* Recent code from GIT is pulled to Local Master.
* Current branch is checked out, then Merge with Master and continue to commit the Staged changes and push the code, Merge conflicts to be resolved before commit

git checkout currentbranch

git merge master

1. Rebasing

* With the command rebase, the changes that were committed on one branch and place them on another one.
* Recent code from GIT is pulled to Local Master.

git pull upstream master

* Current branch is checked out, then rebase Master and continue to commit the Staged changes and push the code.

git checkout currentbranch

git rebase master

Terminologies :-

1. Upstream – It is organization git . In our example it is <https://github.paypal.com/EDSVisualization-R/tmisSQLs>
2. Origin – The place which is forked from upstream. In my case it is

<https://github.paypal.com/akumawat/tmisSQLs>

1. Master - this would be you local master i.e. On your local machine.

Clone the project git clone <HTTP>

After clone get into the get inside the project folder and check

Get remote -v

You will see two lines. i.e. your local is set up with You personal git

Set up with your Upstream

git add remote upstream <Upstream HTTP.>

Everyday when we come to office (Sync you local master with Upstream):-

1. Switch to master : git checkout master
2. Git pull upstream master
3. If there are any changes, then git push origin master.

How to make changes :-

1. Always work on Local branches. Never work on Master branch.
2. Create local branch : git branch <BRANCH\_NAME>
3. Now make whaterver changes you want to do. Git checkout <Branch\_name>
4. Once you are done with your changes -- do “git status” to know what are the changes you made.
5. These changes are all untracked. If you wish to commit these changes then you need to add those changes ie. Track them .
6. To add all the changes – git add .
7. To add a specific file change – git add <FILE\_NAME>
8. Now after adding do git status
9. You can see now the file/files are in green color i.e. files are tracked now.
10. Now we need to commit these changes - git commit -m “Give some meaningful commit name”.

Till now the commit reside on your local branch. We want to push these changes to our Origin Master. Once pushed to origin master, we can then raise a pull request.

1. Switch to master : git checkout master
2. Git pull upstream master
3. If there are any changes, then git push origin master.
4. Git checkout <localbranch>
5. Git rebase master. (hopefully this will be successful) (If not that means some merge issue)
6. If get merge conflict read How to resolve merge conflict para. Other wise, continue.
7. Switch to master - git checkout master.
8. Now merge your master with local branch – git merge <Local branch name>.
9. If get merge conflict read How to resolve merge conflict para. Other wise, continue.
10. Now see git log. You can see that your local branch and master is at same place.
11. Type :q to come out.
12. Now we are in position to push the code to origin
13. Git push origin master
14. Now go to origin master portal and raise a pull request.
15. Someone will review your changes and merge the changes in upstream.
16. Once merged with upstream, there will be an additional commit on upstream.
17. You can do git pull upstream master to get that commit into your local git
18. Now push this merge commit to your origin by git push origin master.

Resolve the Merge conflict.

Below is the example to know merge conflict :

Switched to branch 'comment2'

**➜ Financial\_metric** **git:(comment2)** git rebase master

First, rewinding head to replay your work on top of it...

Applying: commit for merge conflict

Using index info to reconstruct a base tree...

M SQLs/Daily SQLs/Production/Financial\_metric/agg\_tmis\_daily\_credit\_finance\_ins\_v2.sql

Falling back to patching base and 3-way merge...

Auto-merging SQLs/Daily SQLs/Production/Financial\_metric/agg\_tmis\_daily\_credit\_finance\_ins\_v2.sql

CONFLICT (content): Merge conflict in SQLs/Daily SQLs/Production/Financial\_metric/agg\_tmis\_daily\_credit\_finance\_ins\_v2.sql

error: Failed to merge in the changes.

Patch failed at 0001 commit for merge conflict

Use 'git am --show-current-patch' to see the failed patch

Resolve all conflicts manually, mark them as resolved with

"git add/rm <conflicted\_files>", then run "git rebase --continue".

You can instead skip this commit: run "git rebase --skip".

To abort and get back to the state before "git rebase", run "git rebase --abort".

**➜ Financial\_metric** **git:(59d75b5) ✗** git log

**To make changes to the desired file ➜ Financial\_metric** **git:(59d75b5) ✗** vim agg\_tmis\_daily\_credit\_finance\_ins\_v2.sql

**Once done changes add the file**

**➜ Financial\_metric** **git:(59d75b5) ✗** git add agg\_tmis\_daily\_credit\_finance\_ins\_v2.sql

**➜ Financial\_metric** **git:(59d75b5) ✗** git rebase --continue

Applying: commit for merge conflict

**➜ Financial\_metric** **git:(comment2)** git status

On branch comment2

nothing to commit, working tree clean

**➜ Financial\_metric** **git:(comment2)** git log

**Now your local branch is above your master ..**

**You need to merge your master with Local branch ..**

**➜ Financial\_metric** **git:(comment2)** git rebase master

Current branch comment2 is up to date.

**➜ Financial\_metric** **git:(comment2)** git checkout master

Switched to branch 'master'

Your branch is up to date with 'origin/master'.

**➜ Financial\_metric** **git:(master)** git pull upstream master

From https://github.paypal.com/EDSVisualization-R/tmisSQLs

\* branch master -> FETCH\_HEAD

Already up to date.

Current branch master is up to date.

**➜ Financial\_metric** **git:(master)** git merge comment2

Updating 59d75b5..3175f3e

Fast-forward

SQLs/Daily SQLs/Production/Financial\_metric/agg\_tmis\_daily\_credit\_finance\_ins\_v2.sql | 2 +-

1 file changed, 1 insertion(+), 1 deletion(-)

**➜ Financial\_metric** **git:(master)** git log

**➜ Financial\_metric** **git:(master)** git push origin master

Counting objects: 14, done.

Delta compression using up to 8 threads.

Compressing objects: 100% (14/14), done.

Writing objects: 100% (14/14), 1.27 KiB | 1.27 MiB/s, done.

Total 14 (delta 8), reused 0 (delta 0)

remote: Resolving deltas: 100% (8/8), completed with 4 local objects.

To https://github.paypal.com/akumawat/tmisSQLs.git

59d75b5..3175f3e master -> master

**➜ Financial\_metric** **git:(master)**