

Tech Session 3:

Git + SQL

15 September 2021

Agenda

1. Introduction to git & github
2. Set up github access on the server
3. Test out some github workflows
4. SQL Refresher Exercises

git and github overview

What are git and github?

- **git** is version control software - it tracks changes to files/folders
 - Always have the current version
 - Review how a project has changed (how did the code work before?)
 - Collaboration: easy to combine work from multiple teammates
- **github** hosts git online
 - Organize teams and repositories
 - Share repositories over the internet
 - Other extra features beyond git (e.g., “releases”, CI, etc)

"FINAL".doc



FINAL.doc!



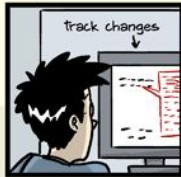
FINAL_rev.2.doc



FINAL_rev.6.COMMENTS.doc



FINAL_rev.8.comments5.
CORRECTIONS.doc



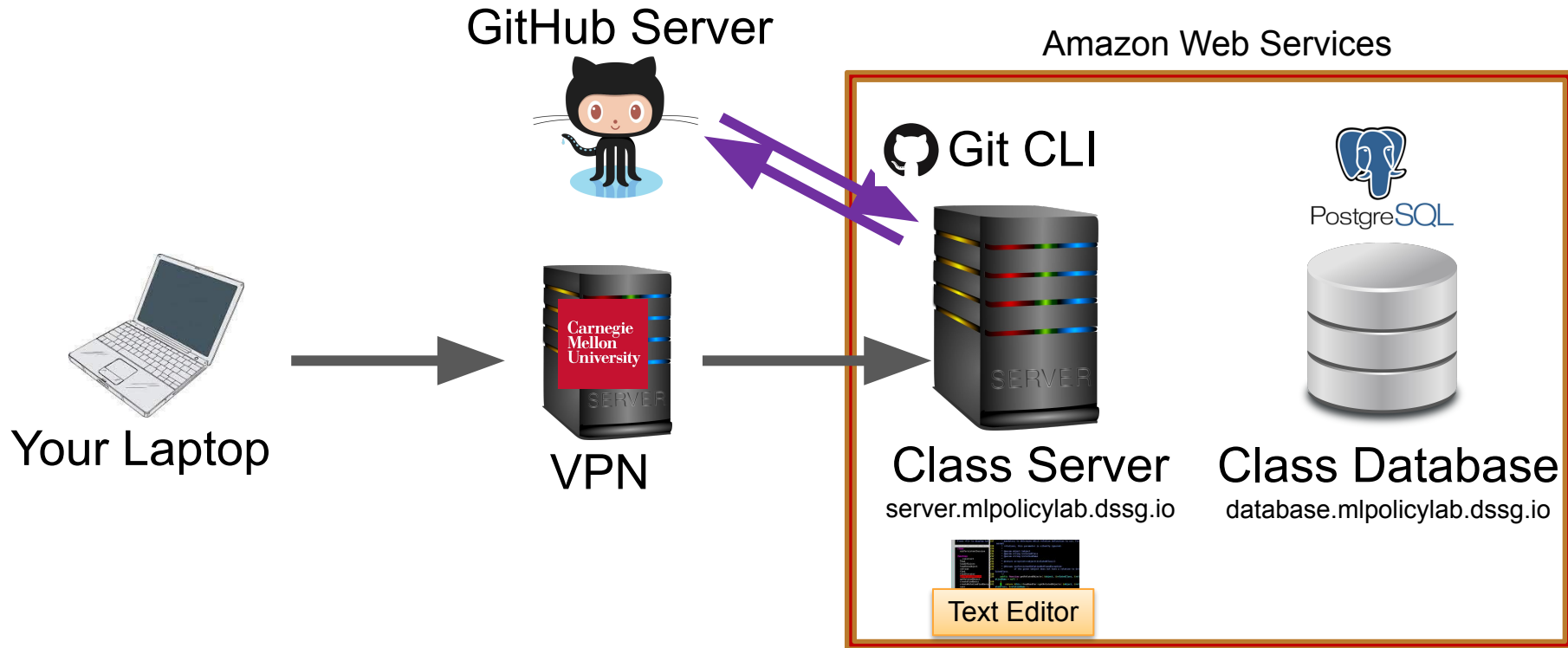
FINAL_rev.18.comments7.
corrections9.MORE.30.doc




FINAL_rev.22.comments49.
corrections.10.#@\$%WHYDID
ICOMETOGRADSCHOOL????.doc

Class Infrastructure Elements:

When edit code with a *remote* text editor



Git basics: Repositories

 **Data Science for Social Good**

[Overview](#) [Repositories](#) [Packages](#) [People](#) [Teams](#) [Projects](#) [Settings](#)

Type ▾

Language ▾

Sort ▾

New repository

mlforpublicpolicylab
Repo for ML for Public Policy Lab course at CMU

machine-learning public-policy

Jupyter Notebook

MIT


24

68

0

0

Updated 23 minutes ago



MLinPractice
Repository for ML in Practice Course at CMU (10-718)

Jupyter Notebook

MIT


2

19

0

0

Updated 2 hours ago



mlinpractice_fall21_hmm

Private

Repository for CMU Class Project Group

Jupyter Notebook


0

0

0

0

Updated 2 days ago



Git basics: Getting a repository

Two ways to get a repository:

1. Create a new one with `git init`:

- Create a folder to store your new repository
- `cd` to it
- Run `git init` inside

2. Download a pre-existing repository with `git clone`:

- `cd` to a folder for storing projects, like `~/projects`
- Run `git clone git@github.com:{username}/{project name}`
- ex: `git clone git@github.com:dssg/test-mlpolicylab-private`
- We'll do this one!

Getting a GitHub Repo: `git clone`

Class Server

server.mlpolicylab.dssg.io



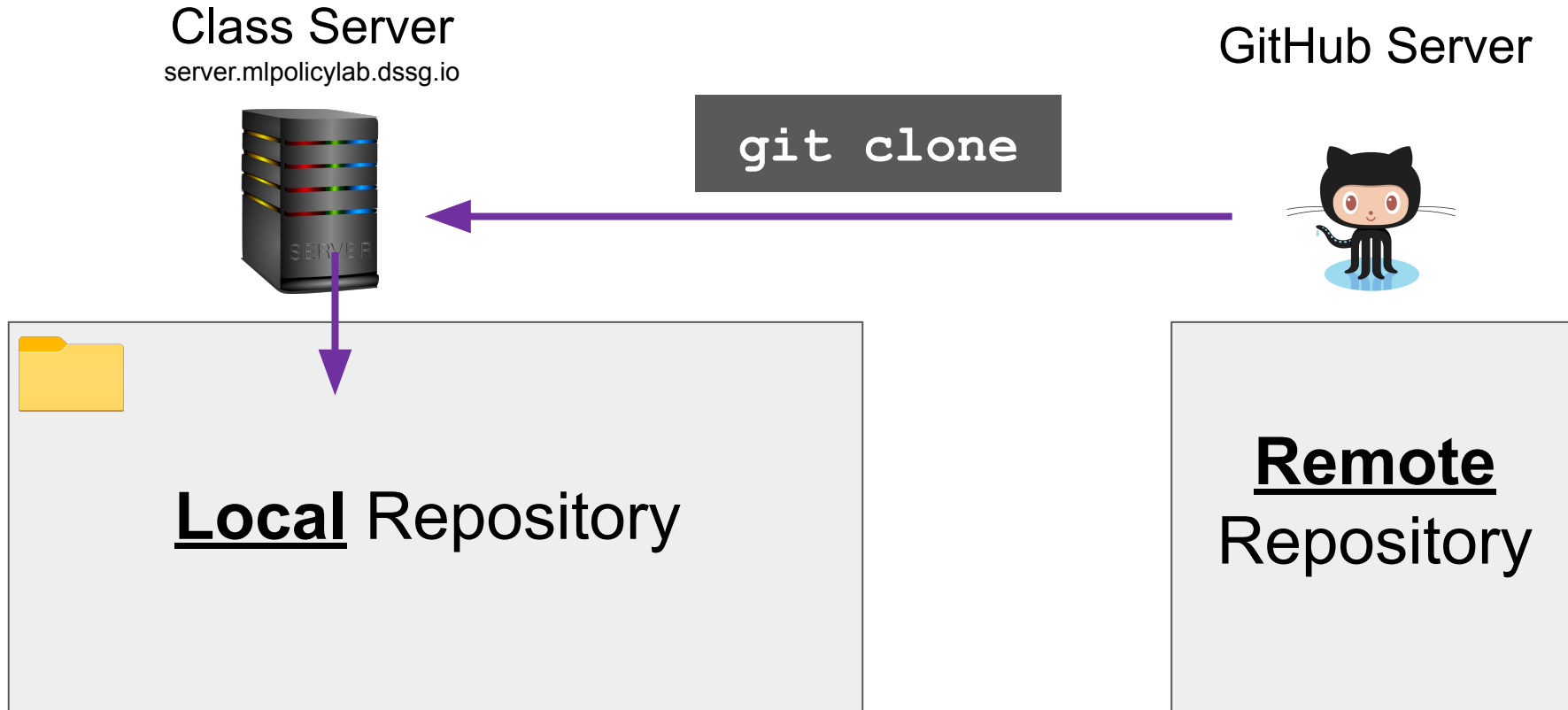
`git clone`

GitHub Server



Remote
Repository

Getting a GitHub Repo: `git clone`



🔗 master ▾

🔗 1 branch

🏷 1 tag

Go to file

Add file ▾

↓ Code ▾



rayidghani Update README.md



01 - Intro and Scoping

Update README.md



02 - Case Studies and Acquiring Data

data discussion slide order



03 - Data Exploration, Analytical For...

updating weekly readmes



04 - Machine Learning Pipelines

updating weekly readmes



05 - Features

updating weekly readmes



06 - Performance and Evaluation Pt 1

updating weekly readmes

12 days ago

12 days ago

Clone with SSH ?

[Use HTTPS](#)

Use a password protected SSH key.

git@github.com:dssg/mlforpublicpoli



Open with GitHub Desktop



Download ZIP

Structure of your local repo

Class Server

server.mlpolicylab.dssg.io



GitHub Server



Local Repository

WORKING

ADDED

aka "index"

COMMITTED

aka "HEAD"

Remote
Repository

Structure of your local repo:

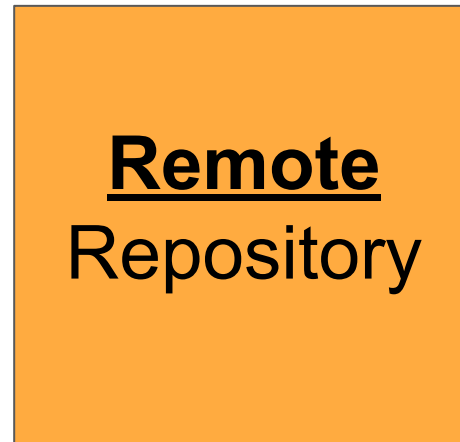
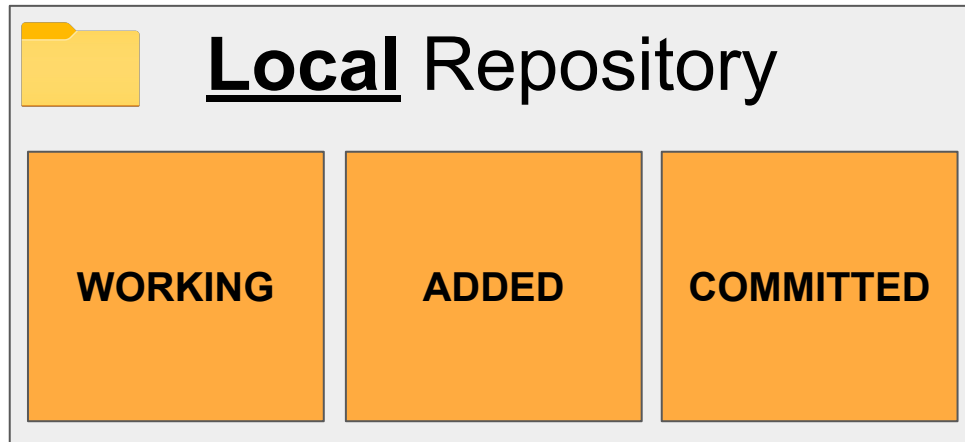
After you clone, these are all the same...

Class Server

server.mlpolicylab.dssg.io



GitHub Server



Git basics: Commits

Commit - A record of the state of a repository at one point in time

- A repository keeps a list of all of your commits
- Look at old commits to review change history
- Compare two commits and see what's changed

Git basics: Commits

 [dssg](#) / [test-mlpolicylab-private](#) Private

 Unwatch ▾

50

 Star 1

 Fork 0

[Code](#)

[Issues](#)

[Pull requests](#)

[Actions](#)

[Projects](#)

[Wiki](#)

[Security](#)

[Insights](#)

[Settings](#)

 master ▾

Commits on Aug 30, 2021

cleanup old test files

 shaycrk committed 13 days ago



[4bef4e7](#)



Commits on Sep 11, 2020

initial git commit

 danschnelbach committed on Sep 11, 2020



[3192312](#)



initial git test upload.

 danschnelbach committed on Sep 11, 2020



[e9a6540](#)



Commits on Sep 10, 2020

Checkpoint

 danielyou95 committed on Sep 10, 2020



[13a017a](#)



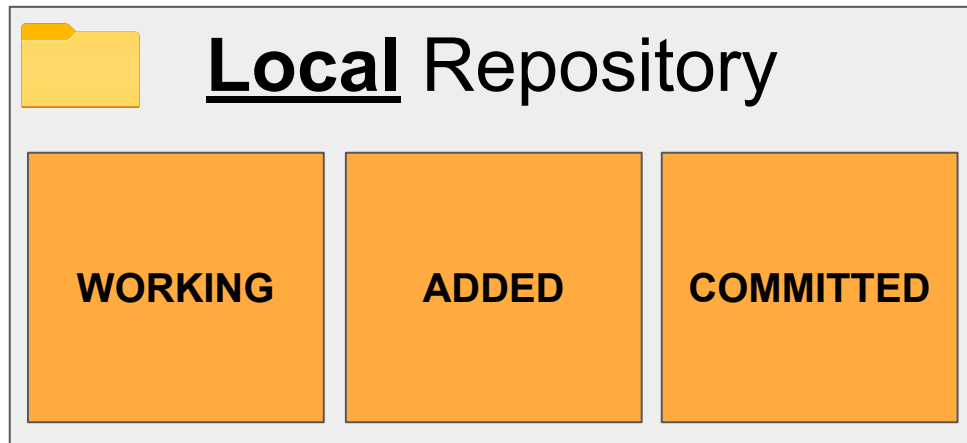
Git Basics: Editing Your Code

Class Server

server.mlpolicylab.dssg.io



GitHub Server



Remote
Repository

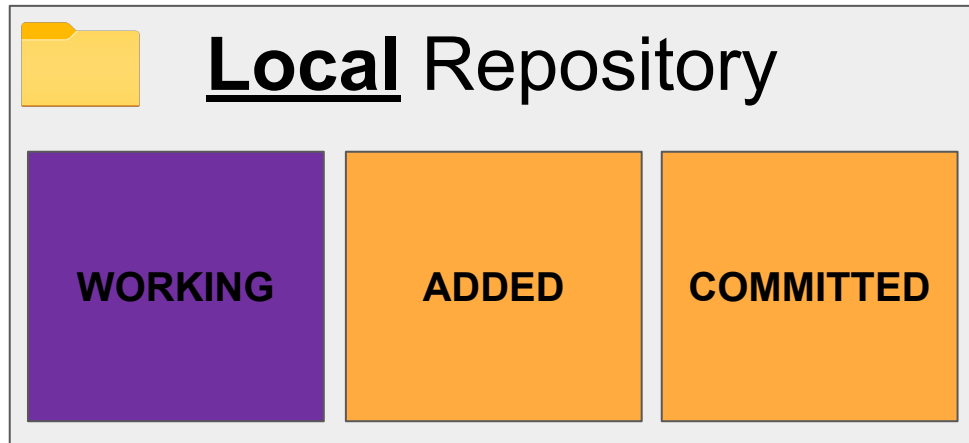
Git Basics: Editing Your Code

Class Server

server.mlpolicylab.dssg.io



GitHub Server



Remote
Repository

Git Basics: Staging for Commit (`git add`)

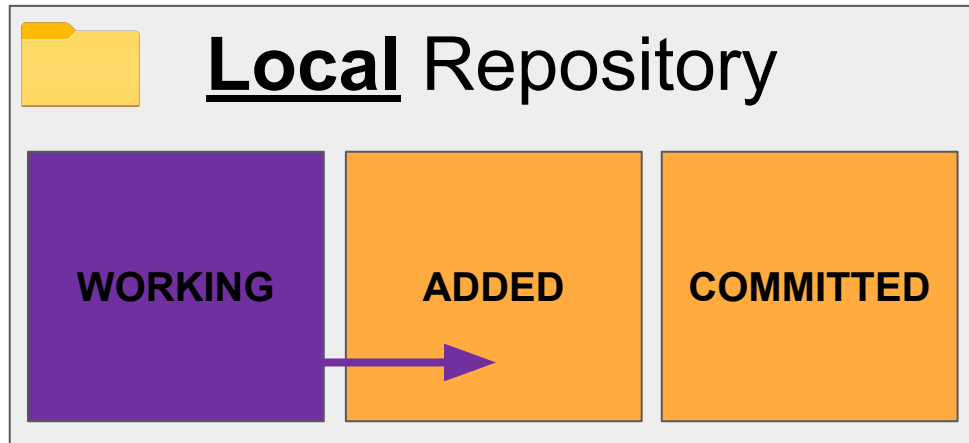
Class Server

server.mlpolicylab.dssg.io



`git add`

GitHub Server



Remote
Repository

Git Basics: Staging for Commit (`git add`)

Class Server

server.mlpolicylab.dssg.io



`git add`

GitHub Server



 **Local** Repository

WORKING

ADDED

COMMITTED

Remote
Repository

Git Basics: Committing Code (`git commit`)

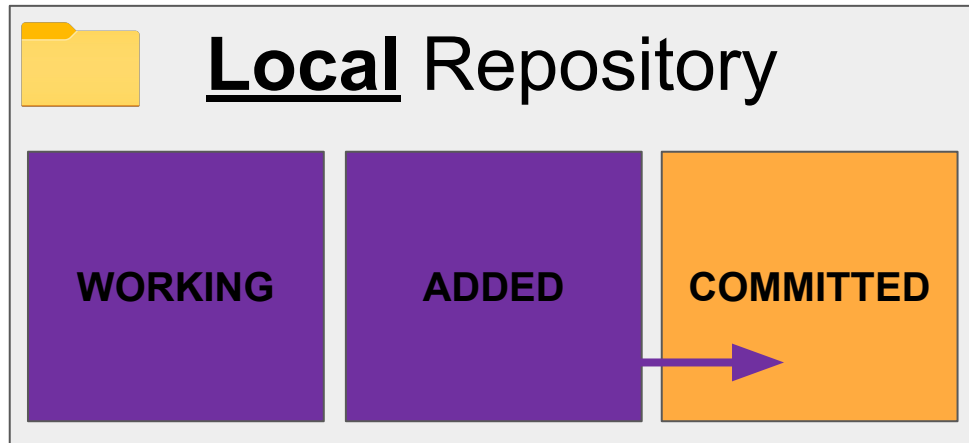
Class Server

server.mlpolicylab.dssg.io



`git commit`

GitHub Server



Remote
Repository

Git Basics: Committing Code (`git commit`)

Class Server

server.mlpolicylab.dssg.io



`git commit`

GitHub Server



Local Repository

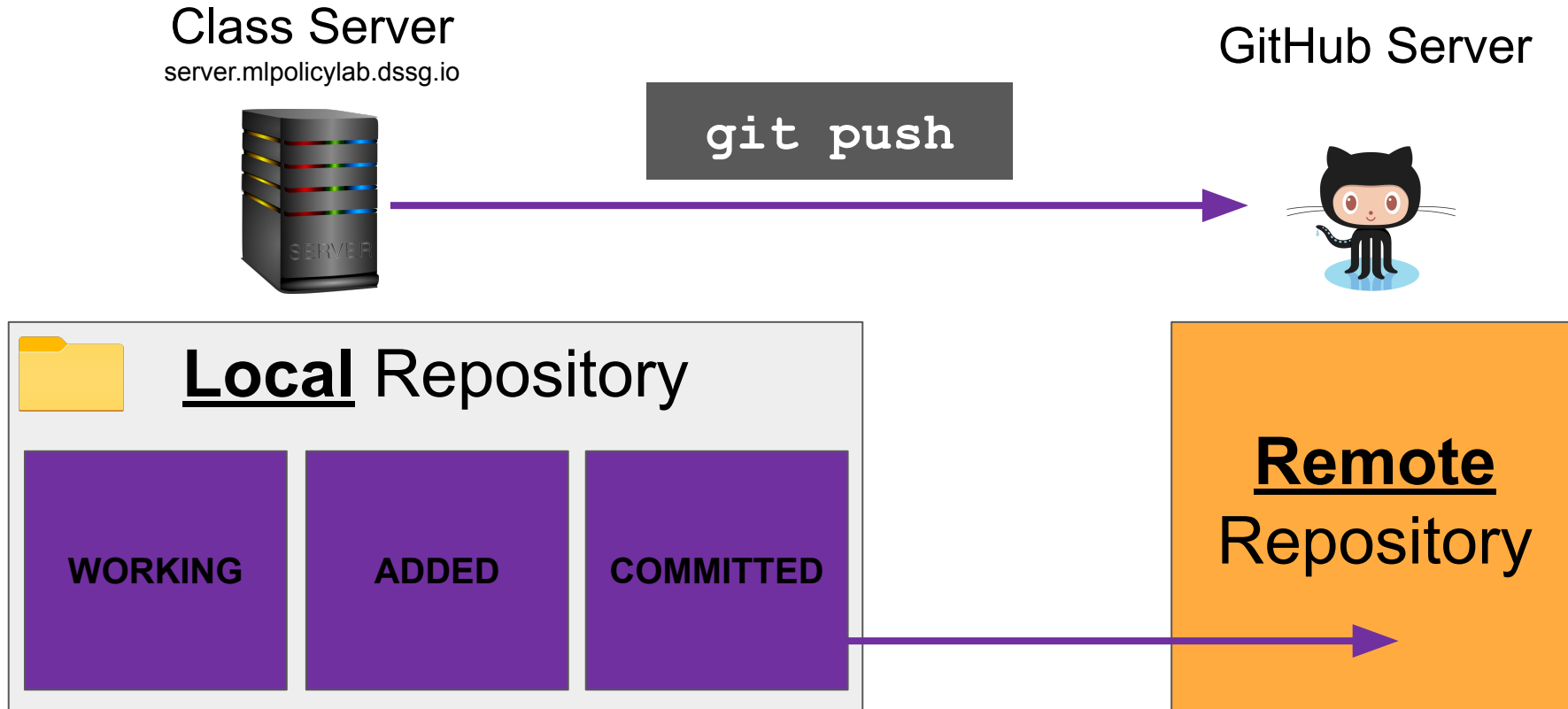
WORKING

ADDED

COMMITTED

Remote
Repository

Git Basics: Updating Remote (`git push`)



Git Basics: Updating Remote (`git push`)

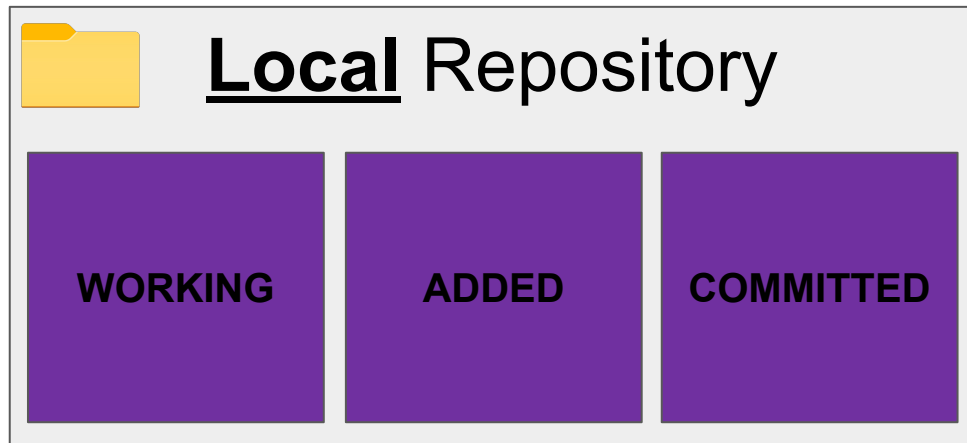
Class Server

server.mlpolicylab.dssg.io



`git push`

GitHub Server

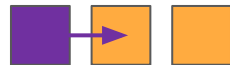


Remote
Repository

Quick note on “add” vs “commit”

When **creating a new file**, you will always need to do:

```
git add my_new_file
```



```
git commit
```



When only **modifying existing files**, you can stage and commit in one step via:

```
git commit -a
```



Git Basics: When Your Teammate Pushes Updates...

Class Server

server.mlpolicylab.dssg.io



GitHub Server



Local Repository

WORKING

ADDED

COMMITTED

Remote
Repository

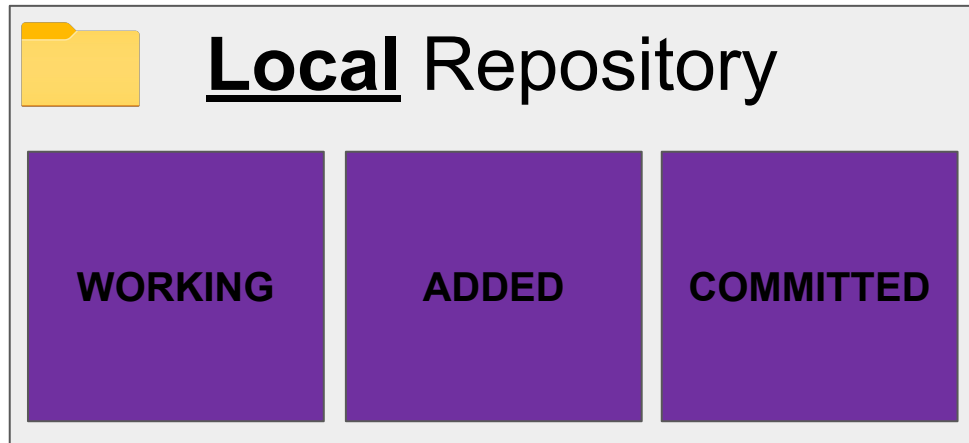
Git Basics: When Your Teammate Pushes Updates...

Class Server

server.mlpolicylab.dssg.io



GitHub Server



**Remote
Repository**

Git Basics: Getting Updates (`git pull`)

Class Server

server.mlpolicylab.dssg.io



GitHub Server



`git pull`



Local Repository

WORKING

ADDED

COMMITTED

Remote
Repository



Git basics: Making commits

1. Stage files:

- Staging collects the files you intend to update in the repository
- In your repository, run: `git add <some_file>` for each file to be updated
- Example: `git add test.py`

2. Once you've staged the desired files, commit your changes:

- Committing your changes adds them to your repository
- In your repository, run: `git commit -m "<some commit message>"`
- Commit messages are short messages that summarize a commit.
Ex: `git commit -m "Update docstrings in feature engineering"`

(if interested: [commit message guidelines](#))

Git basics: sharing your changes

`git pull <remote> <branch>`: Update your local repository (and your code) to match a remote repository.

- We `pull` our changes from GitHub
- Might cause a `merge conflict` - read about that [here](#)
- Do this before you push!
- Example: `git pull origin master`

`git push <remote> <branch>`: Add your changes to a repository on a remote machine

- We use GitHub, so `push` sends our changes to GitHub's servers
- Your new commits are added to the remote repository
- Example: `git push origin master`

Git: Common workflow

When you start working:

The first time, clone an existing repo: `git clone`

Every time, get changes since last time: `git pull`

Add new files: `git add` or make changes to existing files

Make a local checkpoint: `git commit`

Push to the remote repository: first `git pull`, resolve any conflicts, then `git push`

More advanced cheatsheet is <https://gist.github.com/jedmao/5053440>

.gitignore: Telling git what not to track (e.g., secrets)

cfa-getcalfresh / .gitignore



shaycrk model workflow diagrams

 History

 2 contributors



9 lines (9 sloc) | 132 Bytes

...

```
1 all_apps_062120.csv
2 *.gitattributes
3 *.ipynb_checkpoints
4 __pycache__
5 db_config.yaml
6 secret_config.yaml
7 /data/*
8 *scratch.py
9 .DS_Store
```

More Advanced Git Topics (not for today...)

- Branching, Merging, Pull Requests
- Dealing with Conflicts
- Rebasing
- Resetting History
- Forks
- CI, Releases, Deployment

When in doubt...



When in doubt...

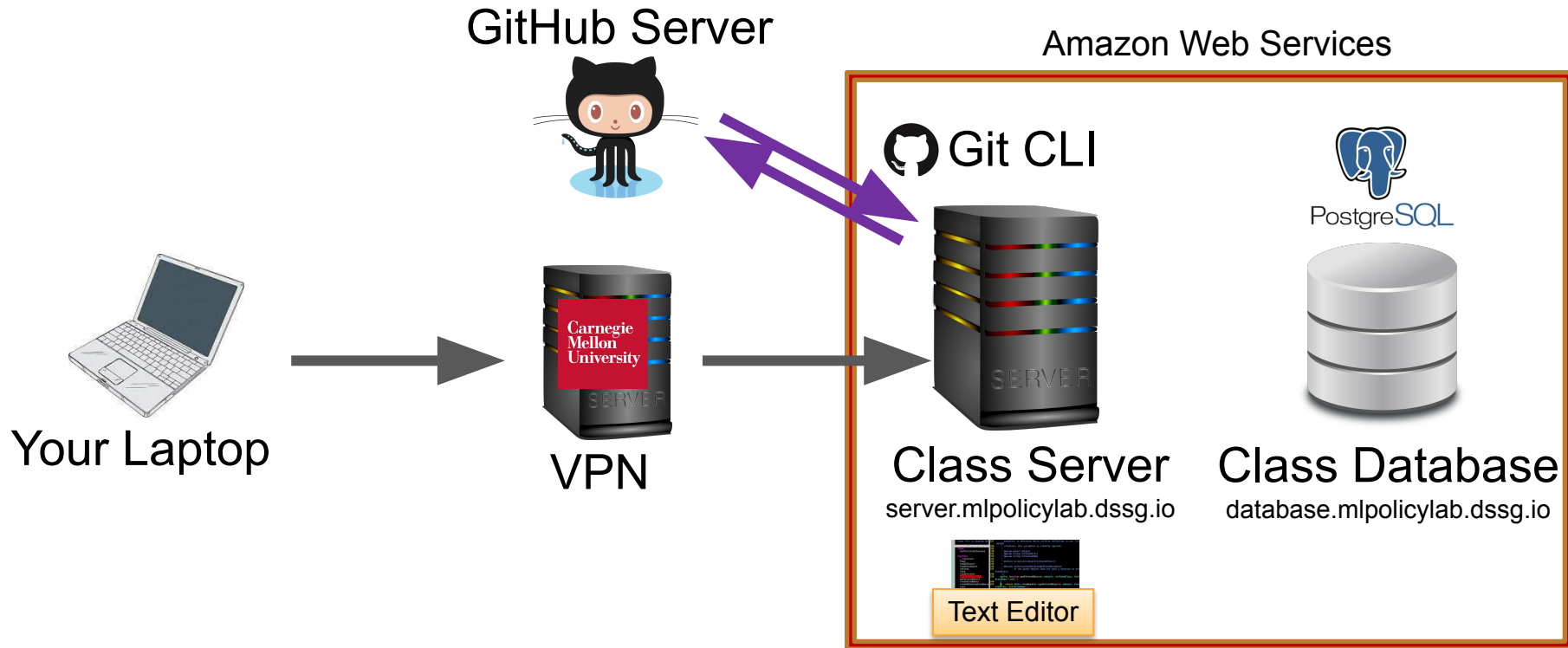


If that doesn't fix it, `git.txt` contains the phone number of a friend of mine who understands git. Just wait through a few minutes of 'It's really pretty simple, just think of branches as...' and eventually you'll learn the commands that will fix everything.

git and github set up

Class Infrastructure Elements:

When edit code with a *remote* text editor



Step 1: ssh to the server

```
ssh -i /path/to/key {andrew_id}@server.mlpolicylab.dssg.io
```

Step 2: generate a new private key

On the server:

```
ssh-keygen -t ed25519 -C "{andrew_id}@andrew.cmu.edu"
```

Accept the default location. You can enter a passphrase if you want (but may want to leave it blank for faster git workflows).

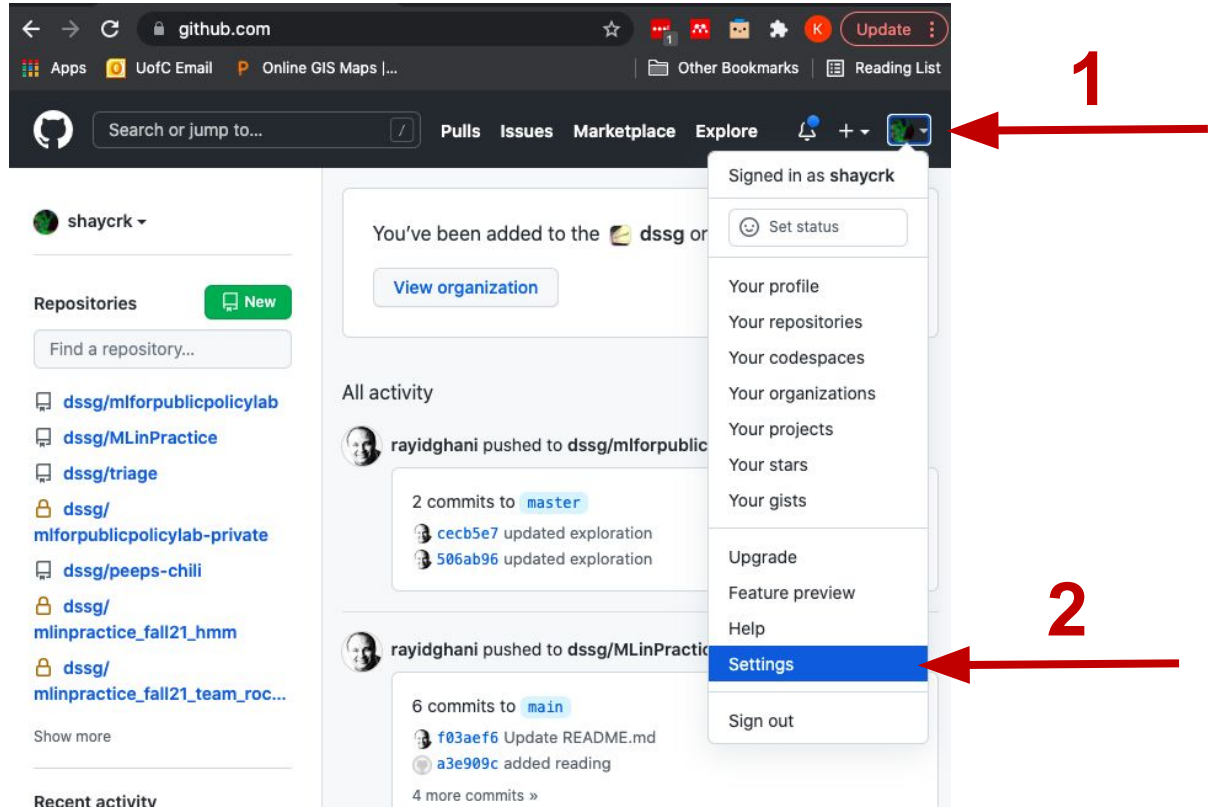
Step 3: add your new public key to github (via web)

On the server:

```
cat ~/.ssh/id_ed25519.pub
```

Copy the output of this -- we'll load it as a new key in github

Step 3: add your new public key to github (via web)



The screenshot shows the GitHub web interface. At the top, the navigation bar includes the GitHub logo, a search bar, and links for Pulls, Issues, Marketplace, and Explore. A red arrow labeled '1' points to the user profile icon in the top right corner. A dropdown menu is open, showing the user is signed in as 'shaycrk'. The menu includes options like 'Set status', 'Your profile', 'Your repositories', 'Your codespaces', 'Your organizations', 'Your projects', 'Your stars', 'Your gists', 'Upgrade', 'Feature preview', 'Help', 'Settings', and 'Sign out'. A red arrow labeled '2' points to the 'Settings' option in the dropdown menu.

github.com

Apps UofC Email Online GIS Maps ... Other Bookmarks Reading List

Search or jump to... Pulls Issues Marketplace Explore

Signed in as shaycrk

You've been added to the dssg or

View organization

Repositories

Find a repository...

dssg/mlforpublicpolicylab

dssg/MLinPractice

dssg/triage

dssg/mlforpublicpolicylab-private

dssg/peeps-chili

dssg/mlinpractice_fall21_hmm

dssg/mlinpractice_fall21_team_roc...

Show more

Recent activity

All activity

rayidghani pushed to dssg/mlforpublic

2 commits to master

cecb5e7 updated exploration

506ab96 updated exploration

rayidghani pushed to dssg/MLinPractice

6 commits to main

f03aef6 Update README.md

a3e909c added reading

4 more commits »

Step 3: add your new public key to github (via web)

The screenshot shows the GitHub interface for user Kit Rodolfa. The left sidebar contains a list of settings: Account settings, Profile, Account, Appearance, Account security, Billing & plans, Security log, Security & analysis, Sponsorship log, Emails, Notifications, Scheduled reminders, SSH and GPG keys, and Repositories. A red arrow labeled '1' points to the 'SSH and GPG keys' option. The main content area is titled 'SSH keys' and includes a 'New SSH key' button in the top right corner, which is highlighted by a red arrow labeled '2'. Below the title, there is a list of existing SSH keys: 'MacBook Air Key' and 'CMU Rayid Desktop Key'. Each key entry shows its SHA256 fingerprint, a 'Delete' button, and its usage status. At the bottom of the SSH keys section, there is a link to a guide on generating SSH keys. Below the SSH keys section, the 'GPG keys' section is visible, featuring a 'New GPG key' button and a message stating that there are no GPG keys associated with the account.

Kit Rodolfa
Your personal account [Switch to another account](#) [Go to your personal profile](#)

SSH keys [New SSH key](#)

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

MacBook Air Key
SHA256: VUQc66U5Ryfe64NtJtrSFky0uIAW4K5NGXIJJLZJG
s
SSH Added on Jan 31, 2019
Last used within the last week — Read/write [Delete](#)

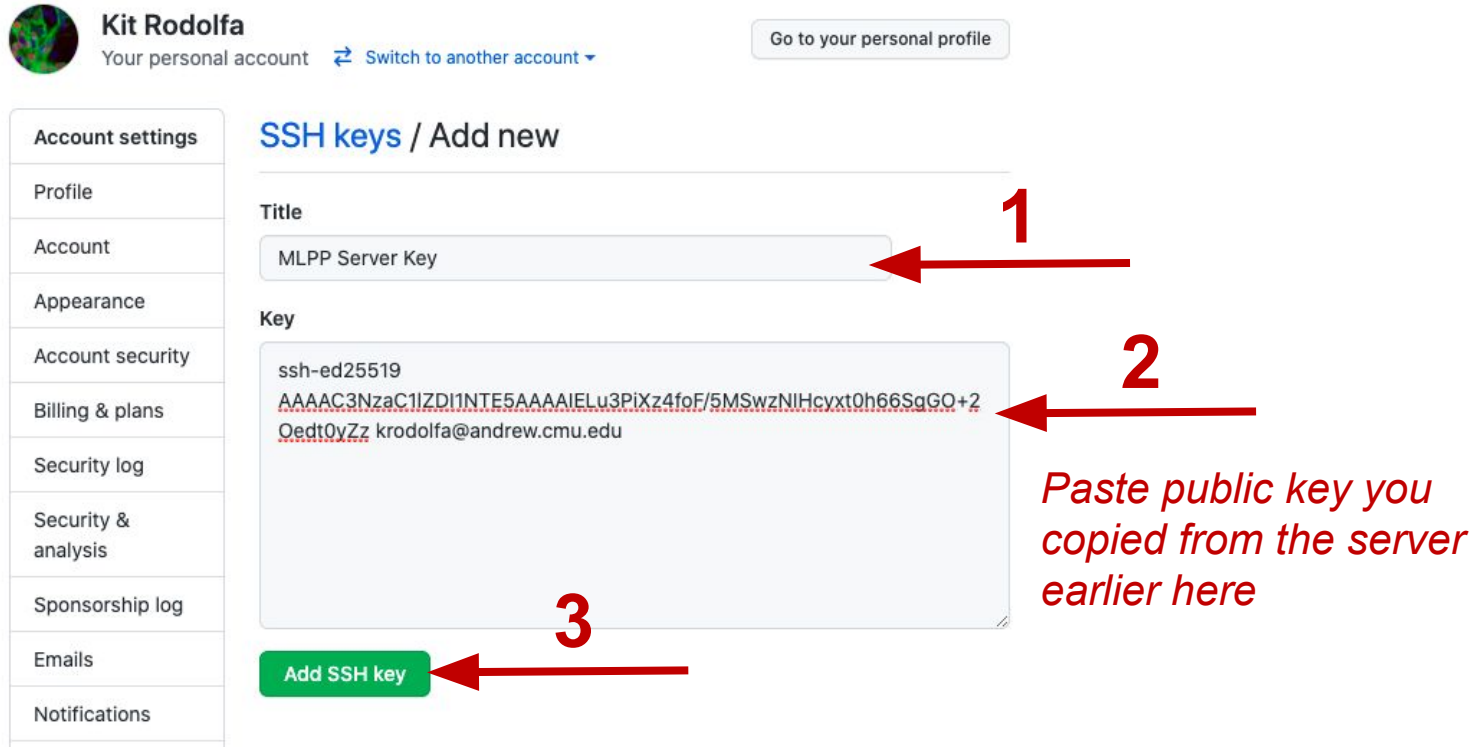
CMU Rayid Desktop Key
SHA256: Xb8z/d0mRE3sAg8meCiPeGi0zIasZUzdfME47khN3B
8
SSH Added on Sep 12, 2019
Last used within the last 2 months — Read/write [Delete](#)

Check out our guide to [generating SSH keys](#) or troubleshoot [common SSH problems](#).

GPG keys [New GPG key](#)

There are no GPG keys associated with your account.
Learn how to [generate a GPG key and add it to your account](#).

Step 3: add your new public key to github (via web)



The screenshot shows the GitHub interface for user Kit Rodolfa. On the left is a sidebar with account settings. The main area is titled 'SSH keys / Add new'. It contains a 'Title' input field with the text 'MLPP Server Key', a 'Key' text area containing an SSH public key, and a green 'Add SSH key' button at the bottom. Three red arrows with numbers 1, 2, and 3 point to these elements respectively. A red text note is positioned to the right of the key text area.

Kit Rodolfa
Your personal account [Switch to another account](#) [Go to your personal profile](#)

Account settings

- Profile
- Account
- Appearance
- Account security
- Billing & plans
- Security log
- Security & analysis
- Sponsorship log
- Emails
- Notifications

SSH keys / Add new

Title

MLPP Server Key

Key

```
ssh-ed25519  
AAAAC3NzaC1lZDI1NTE5AAAAIElu3PiXz4foF/5MSwzNIHcyxt0h66SgGO+2  
Qedt0yZz krodolfa@andrew.cmu.edu
```

Add SSH key

1

2

Paste public key you copied from the server earlier here

3

Step 4: check that you can talk to github

On the server:

```
ssh git@github.com
```

If it's your first time connecting, type “yes” to verify you want to.

If all is working, you should get a message acknowledging your github username and confirming you successfully connected

Step 4: check that you can talk to github

Did everyone get the
“successfully connected” message?

Step 5: configure the git client

On the server:

```
git config --global user.name "Your name here"
```

```
git config --global user.email "your_email@example.com"
```

```
git config --global color.ui true
```

```
git config --global push.default current
```

git and github workflow example

Clone the test repo

On the class server:

```
cd ~
```

```
git clone git@github.com:dssg/test-mlpolicylab-private.git
```

change into the repo directory:

```
cd test-mlpolicylab-private
```



Write some “code”

Create a new file, with some text:

```
echo “{some message for your classmates}” > {your andrew_id}_f21.txt
```

Check to see that git recognizes your new file:

```
git status
```

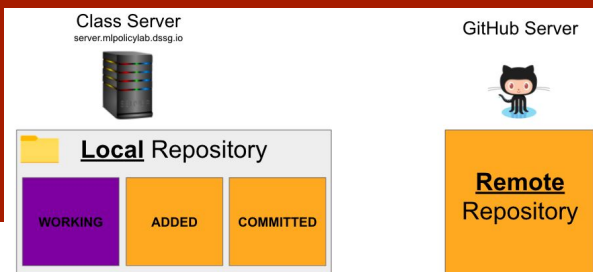
Should look something like:

```
adunmore@ip-10-0-1-213:~/test-mlpolicylab-private$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    adunmore_f20.txt

nothing added to commit but untracked files present (use "git add" to track)
```



Stage your new file

Use `git add` to stage the file for committing:

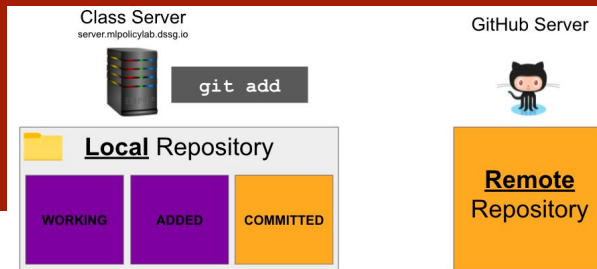
```
git add {andrew_id}_f20.txt
```

Check `git status` to see what's happened:

```
adunmore@ip-10-0-1-213:~/test-mlpolicylab-private$ git add adunmore_f20.txt
adunmore@ip-10-0-1-213:~/test-mlpolicylab-private$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

    new file:   adunmore_f20.txt
```



Commit your change

Use `git commit` to commit your change:

```
git commit -m "{a helpful commit message}"
```

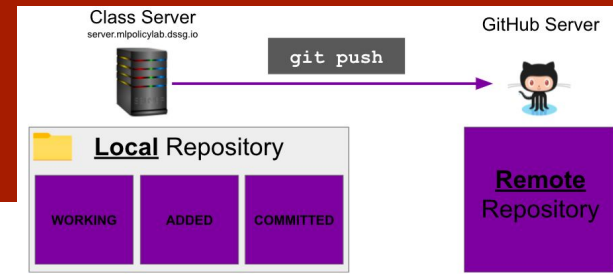


Some Checks...

1. What does `git status` say now?
2. Can you see your new file on the remote repo?

<https://github.com/dssg/test-mlpolicylab-private>

Pulling & Pushing



First, **pull** the class repository to make sure you're up to date:

```
git pull origin master
```

Then, **push** your change to the class repository to share it with your classmates:

```
git push origin master
```

Some Checks...

Can you see your new file on the remote repo now?

<https://github.com/dssg/test-mlpolicylab-private>

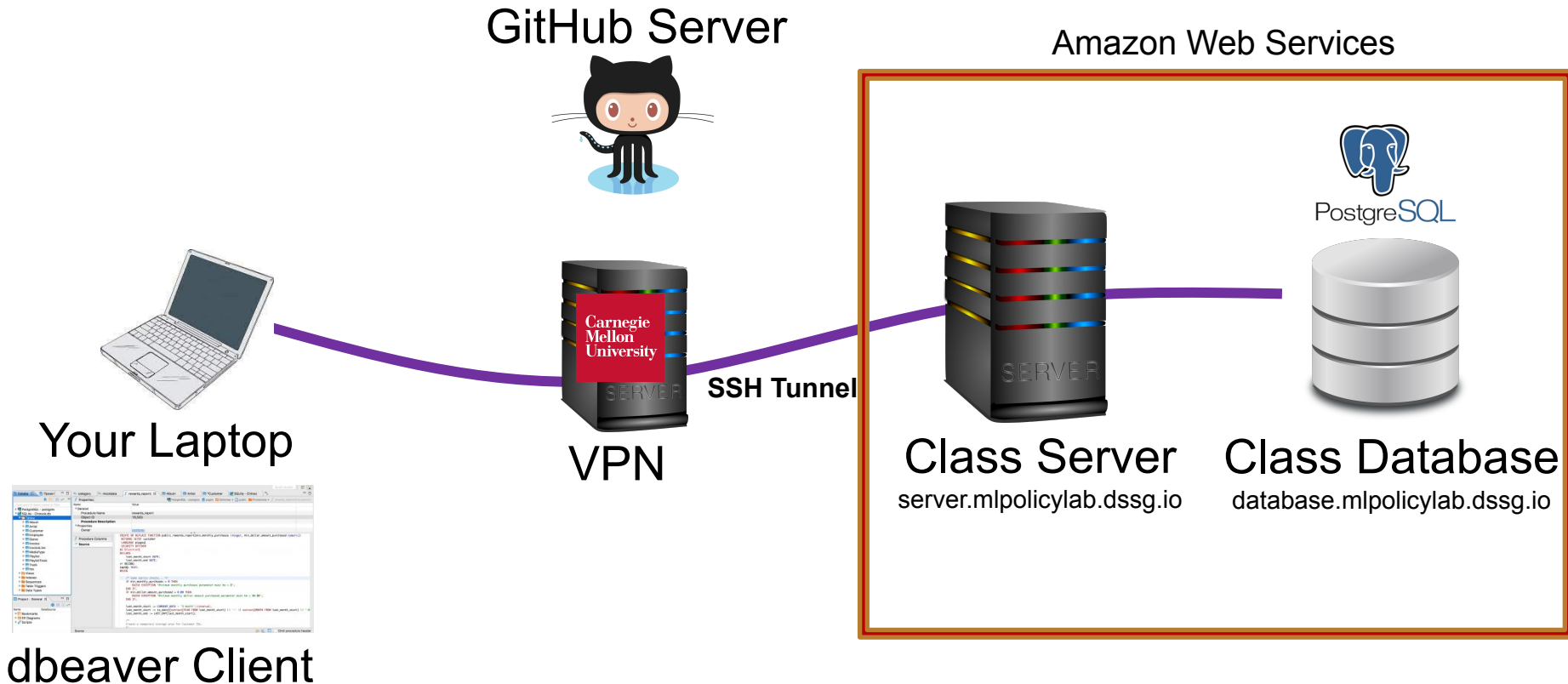
SQL Refresher Exercises

Plan for Today

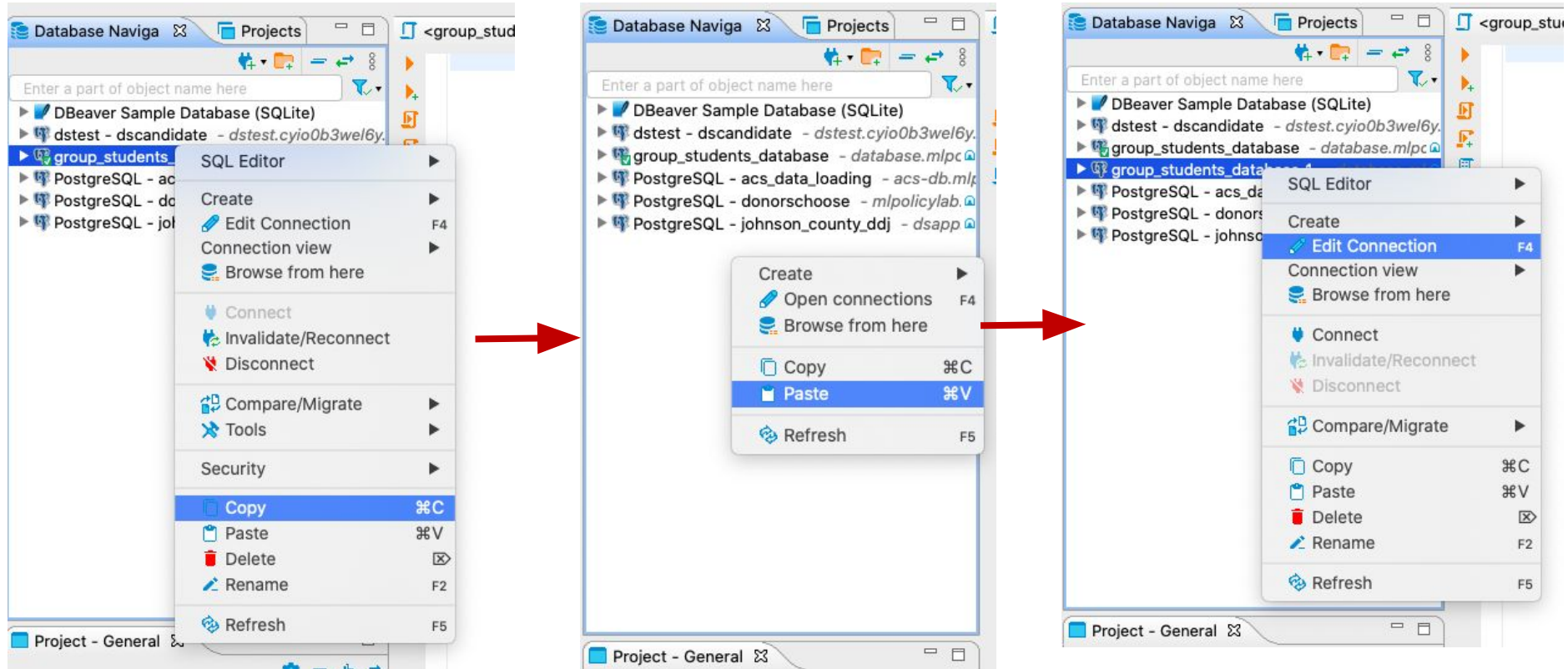
- We'll run through a few SQL exercises by way of a refresher from your previous SQL course
- In a few weeks, we'll cover a couple more advanced topics using the same dataset:
 - Working with SQL via python
 - Advanced SQL queries: CTEs, Window Functions, Temporary Tables
 - Query Optimization and Indices

Class Infrastructure Elements:

When you use dbeaver to connect to the database



Copy/Paste Your Existing Connection to Duplicate



Connection settings

PostgreSQL connection settings



▼ Connection settings

Initialization
Shell Comm
Client ident
Transaction

General
Metadata

Errors and tim

► Data editor

► SQL Editor

Main

PostgreSQL

Driver properties

SSH

Proxy

SSL

Server

Host: database.mlpolicylab.dssg.io

Port: 5432

Database: db_donorschoose_example

Authentication

Authentication: Database Native

Username: postgres

Password:

☒ Save password local

Advanced

Session role:

Local Client:

PostgreSQL 13.2

You can use variables in connection parameters.

Driver name: PostgreSQL

Edit Driver Settings

Test Connection ...

Cancel

OK

For this session we'll connect to:

db_donorschoose_example

Dataset

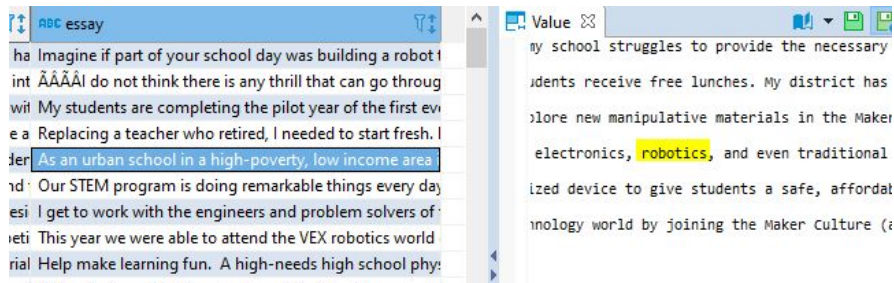
- We will use a Kaggle dataset to play around and perform some basic analysis for a warm up with SQL commands.
- The dataset details are available [here](#).
- donorschoose.org, a non-profit organisation provides a crowdfunding platform for public school teachers to get funds for their class projects directly from individuals.

Dataset

There are four tables under the donorschoose schema:

- 1) **projects** - Contains information about each project.
- 2) **donations** - Contains information about the donations to each project.
- 3) **essays** - Contains project text posted by the teachers.
- 4) **resources** - Contains info about the resources requested for each project.

Q1 -- Select projects that are related to `robotics` from the `essays` table based on the short description provided by teachers.

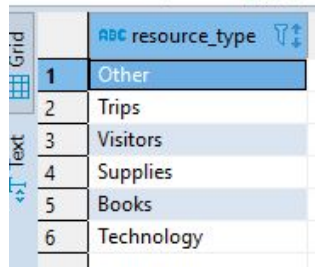


Q2 -- Find the first and last project post dates available in the dataset.

```
1 SELECT MAX(date_posted) as last_date, MIN(date_posted) as first_date
```

	ABC last_date	ABC first_date
1	2014-05-12	2002-09-13

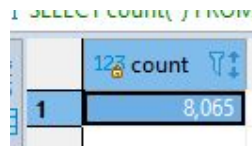
Q3 -- Find all categories of resource-types that can enable a project.



The screenshot shows a database query result in a table format. The table has two columns: an index column and a 'resource_type' column. The index column contains values 1 through 6. The 'resource_type' column contains the following values: Other, Trips, Visitors, Supplies, Books, and Technology. The table is titled 'ABC resource_type'.

	ABC resource_type
1	Other
2	Trips
3	Visitors
4	Supplies
5	Books
6	Technology

Q4 -- Find the total number of project submissions for the month of April in the year 2014.



The screenshot shows a database query result in a table format. The table has two columns: an index column and a 'count' column. The index column contains the value 1. The 'count' column contains the value 8,065. The table is titled '123 count'.

	123 count
1	8,065

Q5 --What is the cumulative donation from the state of Pennsylvania ?

	ABC donor_state	123 sum
1	PA	3,145,306.8800000006

Q6 -- Show state-wise cumulative donation in descending order.

	ABC donor_state	123 total
9	FL	5,904,958
10	NC	5,672,861
11	OK	4,787,662
12	IN	3,738,491
13	MA	3,475,936
14	VA	3,335,513
15	PA	3,145,306
16	MD	3,093,103
17	GA	3,092,347
18	CT	2,416,763
19	AZ	2,335,497
20	DC	2,250,185
21	MI	2,026,364
22	TN	1,950,049
23	CO	1,931,521
24	SC	1,911,372
25	OH	1,543,416
26	MO	1,509,681
27	NV	1,442,208

Q7 -- Find the distribution percentage of different payment methods for donations from Pennsylvania.

	ABC donor_state	ABC payment_method	123 transactions	123 total_transactions	123 pct
1	PA	paypal	6,910	52,451	13.1742006825
2	PA	no_cash_received	18,875	52,451	35.9859678557
3	PA	double_your_impact_match	323	52,451	0.6158128539
4	PA	creditcard	23,549	52,451	44.8971420945
5	PA	check	500	52,451	0.9532706717
6	PA	amazon	2,294	52,451	4.3736058416

Q8 -- For all the projects submitted in the dataset find the total financial aid obtained.

	ABC title	123 overall_donation
17	Tuned in!	423.34
18	Reading and Writing in Color	363.6
19	You Ought to Be in Pictures:" Film as Personal Memoir	350.59
20	Re-String Please!	277.38
21	Ready - Set - Respond!	150
22	Give My First Graders the Gift of Reading!	559.46
23	Play Time! Encourage Reading Skills through Literacy Cer	35
24	HP Powerful Printing!	332.91
25	Concentrate, Collaborate and Learn	[NULL]
26	Butterfly Preserve	937.4
27	Living Pictures: Comprehension through Drama	10
28	Mastering More Than Facts!	1,090.8
29	Independent Reading Center	45
30	Literature Sleuths	5
31	Second Language Advantage	[NULL]
32	Getting Fit and Having Fun With Pedometers in P.E.	[NULL]

SOLUTIONS

A1

```
select title, essay from donorschoose.essays  
where essay like '%robot%'
```

A2

```
select  
    max(date_posted) as last_date, min(date_posted) as first_date  
from  
    donorschoose.projects
```

A3

```
select
    distinct(resource_type)
from
    donorschoose.projects
where
    resource_type is not null
```

A4

```
SELECT
    count(*)
FROM
    donorschoose.projects
WHERE
    date_posted >= '2014-04-01'
    AND date_posted <= '2014-04-30'
```

A5

```
select
    donor_state, sum(donation_total)
from
    donorschoose.donations
group by
    donor_state
having
    donor_state = 'PA'
```

A6

```
select
    donor_state, trunc(sum(donation_total)) as total
from
    donorschoose.donations
group by
    donor_state
order by
    total desc
```

A7

```
select
    dd.donor_state, dd.payment_method, dd.transactions, sub.total_transactions, (100.0 *
    dd.transactions) / sub.total_transactions as Pct
from (
    select donor_state, payment_method, count(payment_method) as transactions
    from donorschoose.donations
    group by donor_state, payment_method
    having donor_state = 'PA'
) dd
join (
    select donor_state, count(*) as total_transactions
    from donorschoose.donations
    group by donor_state
    having donor_state = 'PA'
) sub
on
    dd.donor_state = sub.donor_state
```

A8

```
SELECT
    title, overall_donation
FROM
    (
        SELECT
            e.projectid as projid, title
        FROM
            donorschoose.essays e
        JOIN
            donorschoose.projects p on e.projectid = p.projectid
    ) t
LEFT JOIN
    (
        select projectid, sum(donation_total) as overall_donation
        from
            donorschoose.donations
        group by projectid
    ) d
ON
    t.projid = d.projectid
```