Week 4/5:

Do some backend calculations of how much we can pull per day given 5TB of storage (free). (Also calculate how much it would cost per month, and per year)

5TB*0.05 = 5*1024GB*0.05 = 256GB per month for free
256GB/30 days = 8.53GB per day depending on the number of days
5*1024GB*$ 0.01 per GB/month = $51.2 per month
$51.2 * 12 months = $614.4 per year

Exercise 2: Start understanding the code (especially the sql part). Remember the whole idea of enforcing some kind of limit on retrievals? Lets start brainstorming ideas on how to accomplish this, and we can talk over on friday our plan of execution. This requires you to:

1. Study the code, understand the mechanism
2. Partition the Archives? Vaults? Into individual packets to download? Or maybe specific byte download? (Assume we start with 2 TB of data)-> ramp up to 7TB in 3 years.

If a user wants to open a file from archive, entire archive has to be downloaded as we cannot access individual files from glaicer due to delayed-access and the inability to do so. The best way would be to make the archive the minimal unit for download. Therefore if the data retrieval is around 1GB per request, archive would be most ideal with 1GB

3. Add in a config parser to be able to change size of total + size of downloads per day?
   http://docs.python.org/2/library/configparser.htm
4. Develop some kind of stopping mechanism, or warning mechanism that you have reached a daily retrieval rate.

Daily retrieval rate is determined by the total storage used by a user on Glacier. This changes regularly as a user uploads more data, so the program would not be able to do so in exact manner. Therefore it would be wise to set to limit the download before the free allowance is reached. Warning mechamism would have to be made using cache data. Cache needs to keep track of how much data is stored and how much data has been retrieve on which date.

5. Maybe develop some kind of queuing system? <- think of ways to do this, maybe using sql database?

Queueing system would require some automation, since each download has to be done in every few hours. The program will not most likely be turned on by automation, so it would have to be on for long time. The system should work in two ways: cost-efficient and time. One has to set priority of cost or time and the machine will calculate estimated time and cost.

6. Any other ideas/suggestions? Things that I am missing.

7. Look for any amazon credits you can use (like free money!), I believe there's like an education credit? or something like that. browse around.

http://aws.amazon.com/grants/

Research grant is chosen 4 times a year and the project has to be chosen for the grant for the program to use amazon glacier at discounted rate (i believe). There seems to be no other guaranteed education discount.

You can try to code some of it (just to get the hang of certain tools/utils), but I rather we discuss on Friday our plan of attack on how we're going to accomplish this goal before we start coding.