**To Do on Chair Project:**

1. Add a feature to FacultyMember that is inactive\_starting\_year=AcademicYear…., which could be empty, but if it’s not, then stop showing this person after a certain academic year in drop downs, etc. That way can keep the person in the db, but not have them show up everywhere ☺. This requires migrating the db, etc.
2. Put in safety feature to stop one chair from editing another dept’s stuff!!!
3. Add in something to specify whether a course offering is the first or second half of a semester. This could be something with three properties in the course offering model (first half, second half, both), and then on the displays, it could have a button to change among the three options, or something. Requested by Donna Downs. Maybe should consult with her a bit to see what specific functionality she would like.
4. **Speed it up!!!!** The Faculty Load Summary page loads pretty slowly for big depts (like BSCEP). Could look at it with the Django debug toolbar, to see why. There is some stuff that Tom has talked about that would allow for caching, speeding up complicated queries (where you chase down foreign keys, etc.) This would probably help. In general, you package the various things together, and then run the query, which is faster than doing one query and doing a bunch of logic stuff in python to throw out much of the data. Apparently django has some documentation on this in the query section.
5. **Add office hours, meetings…**.
6. **Donna Downs’ request. Add a flag for faculty members to say if they are adjunct/non-adjunct; put the adjuncts together on one load sheet. Actually, the property is already there, so would just need to use it. Could make a page (accessible from the Profile page and the Faculty to Display page, say) where this can be set up with checkboxes. And maybe something on the page where it goes to dump out the load sheets, so that you would have the option of grouping people together as adjuncts…? Actually, maybe that’s all this is needed?!? Put another checkbox on the page before it dumps the load sheets to say if the person should be regarded as an adjunct. Hmm…!**
7. **Add CRNs to course offerings.**
8. Create a sandbox year that allows you to play around with different scenarios?
   1. Should it have the ability to copy things over from that year into the actual current year?
   2. In Profile, make it so that you can either show or not show the sandbox year. Uh-oh…that’s going to require a change to the model!!!
9. For courses that are offered spring and fall, the faculty load summary seems to list the SPRING ones first, and then the FALL ones. WHY?!? (See EDU150L for an example.)
10. Export feature:
    1. Fix redirect after downloading the file. Not sure how to do this at the moment…. From Tom:
       1. See the "checked" answer here: <http://stackoverflow.com/questions/20827665/django-redirect-after-file-download>, which references an informative page that features a solution: <http://johnculviner.com/jquery-file-download-plugin-for-ajax-like-feature-rich-file-downloads/>
11. Add capacities to rooms.
12. Superuser:
    1. Oops….crn was attached to Courses instead of to CourseOfferings….ruh-roh….
13. Add new field for actual course enrollments?
14. Add things for profs’ schedules (office hours, committee assignments, etc.)
15. Possibilities for data export:
    1. Weekly schedule (for Barb…for hanging on our doors) – I guess this is actually a Word document. What if this could be dumped out in pdf?
       1. Would need to add a new field for office hours.
16. Faculty Load Summary page:
    1. Under Meeting Times > Update Class Schedule, make it possible to delete all entries and immediately go to New Class Schedule.
    2. Add # students as a column?

**Nice AJAX/django reference:**

http://www.tangowithdjango.com/book/chapters/ajax.html

**Migrating instructions:**

Dr. Kiers,

I just saw this and if you have already solved the problem disregard this email.

Here are the steps I would use:

1. If you are not using south, use south. I think it will work with your version of python. You can install it with pip install south.

2. If you just installed south, make sure you have done an initial migration.

3. Add the field to the model with a default value that you can remove later in a later migration.

4. python manage.py schemamigration --auto <appname>

5. It should reply with something saying that it created the migration and you can apply it with migrate.

6. python manage.py migrate <appname>

7. The migration should have taken place and it will try to load the initial data which will fail because the schema has changed.

8. run the backup.sh script again to dump it.

9. Try to migrate again. (when using south, always used migrate instead of syncdb)

The above steps should also work for changing a foreign key to a many to many field. If you are not or can not use south then I have no idea what you would do.

I hope this helps.