Lecture 11: Oracle + Python & Class Project

Wednesday, February 25, 2015

Agenda

- Python Oracle Interface (cx_Oracle)
- Class Project

Getting Started with cx_Oracle

Mac Users:

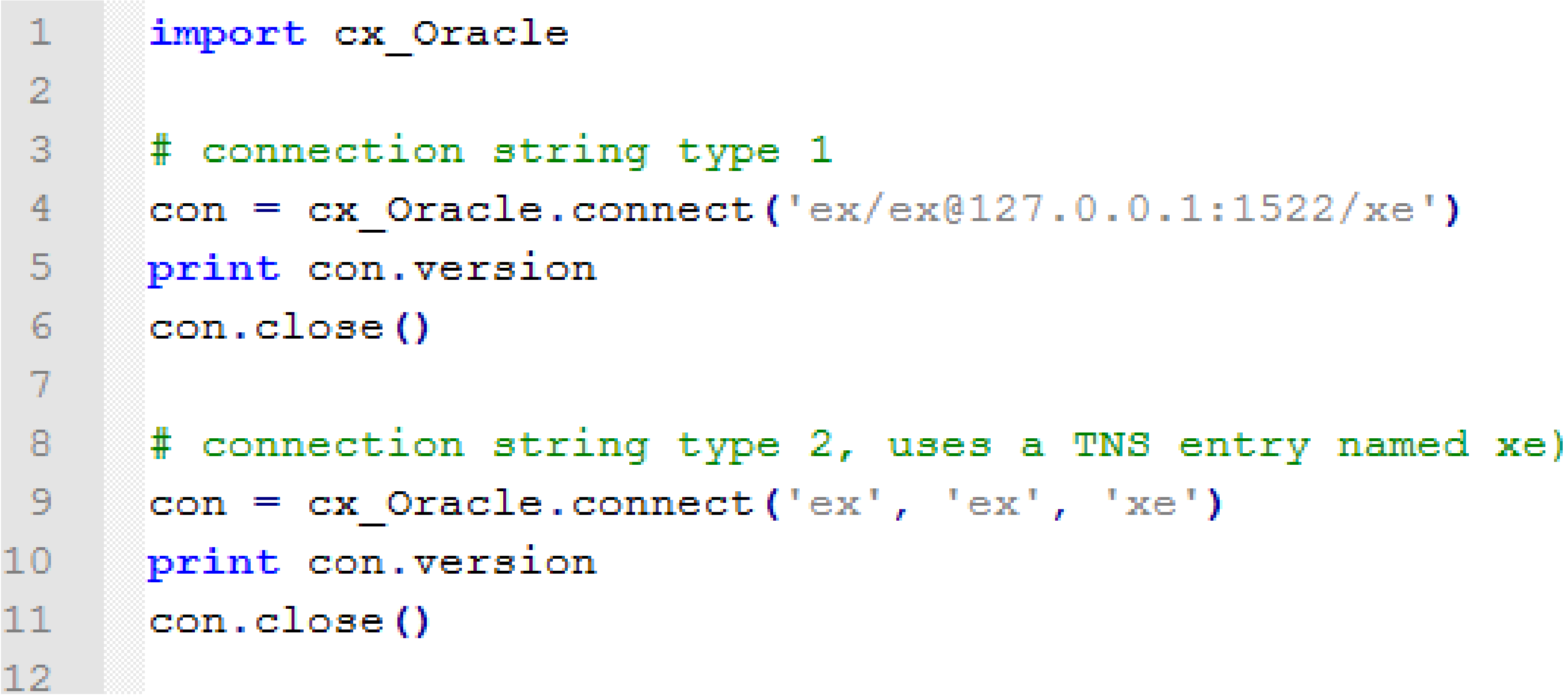
- Option 1: Download new VM that is loaded with cx_Oracle:
  http://www.cs.utexas.edu/~scohen/vm_image/cs327_1.ova

- Option 2: Install and build cx_Oracle from source (requires Oracle Instant client):
  http://www.cs.utexas.edu/~scohen/cx_Oracle/macosx.html

Windows Users:

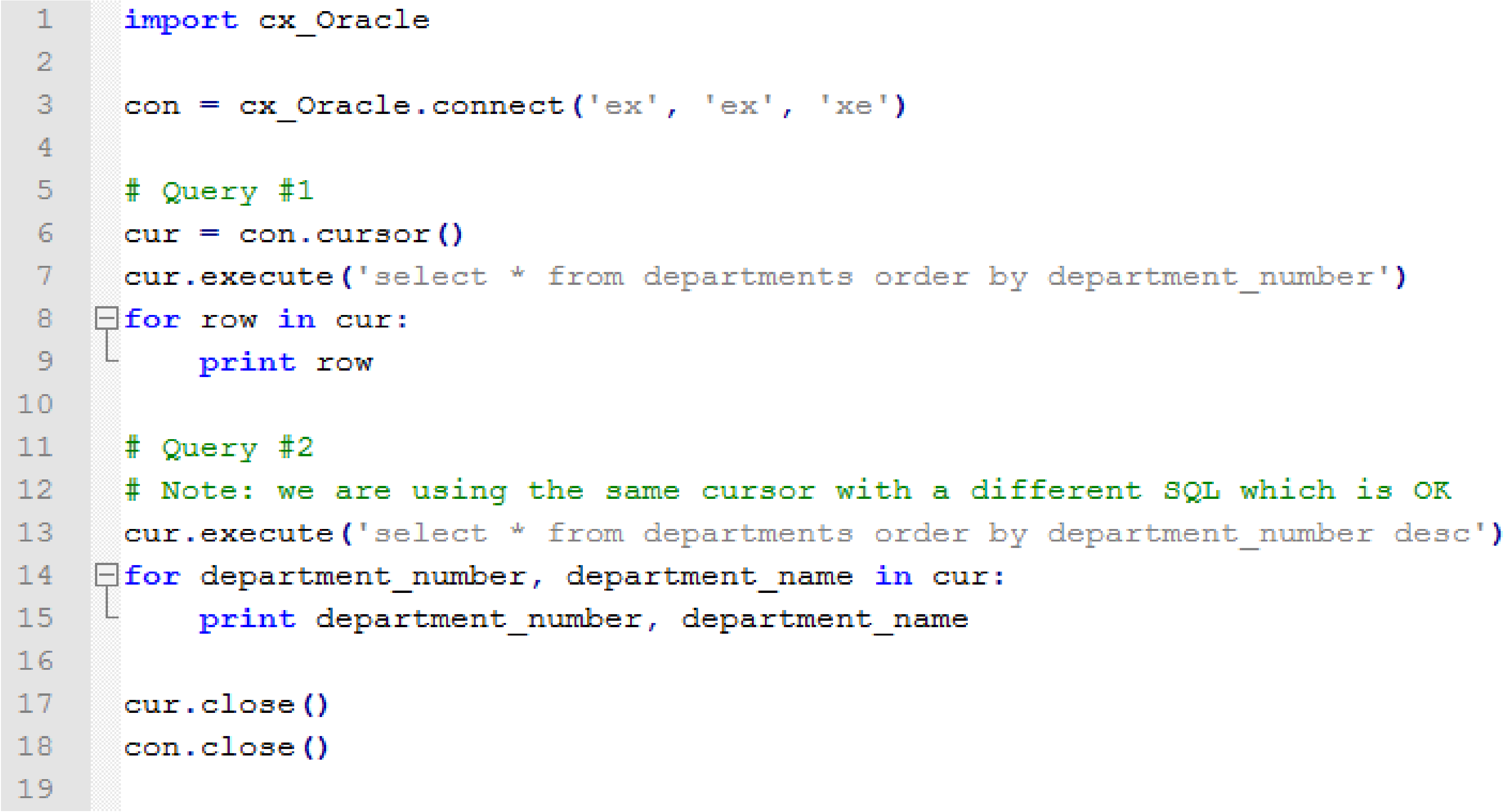
- Install Python 2.7 and then install cx_Oracle (doesn’t require the Oracle Instant client):
  http://www.cs.utexas.edu/~scohen/cx_Oracle/windows.html

Sample DB Connections



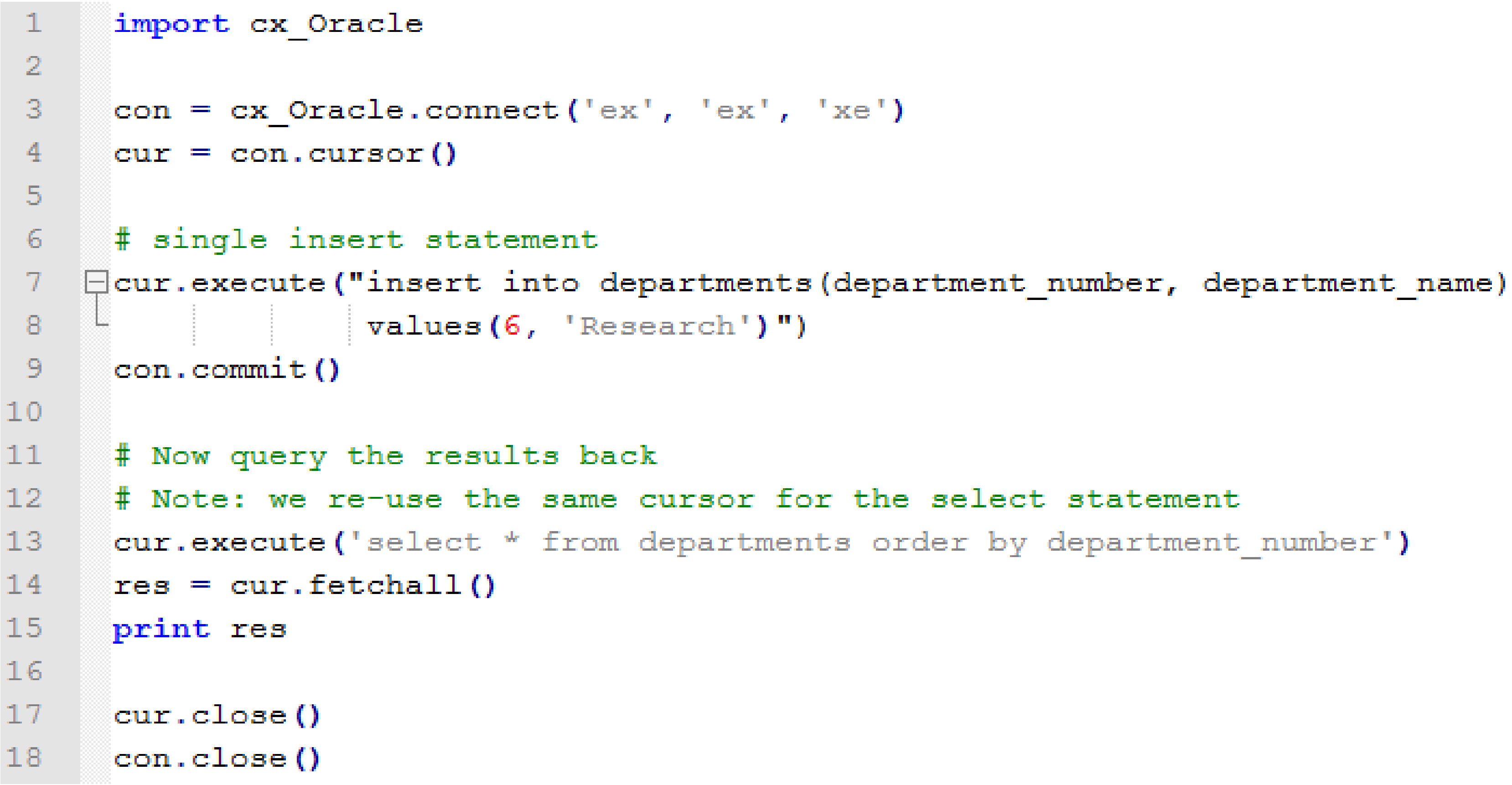
```
import cx_Oracle

# connection string type 1
con = cx_Oracle.connect('ex/ex@127.0.0.1:1522/xe')
print con.version
con.close()

# connection string type 2, uses a TNS entry named xe)
con = cx_Oracle.connect('ex', 'ex', 'xe')
print con.version
con.close()

```

Sample Queries



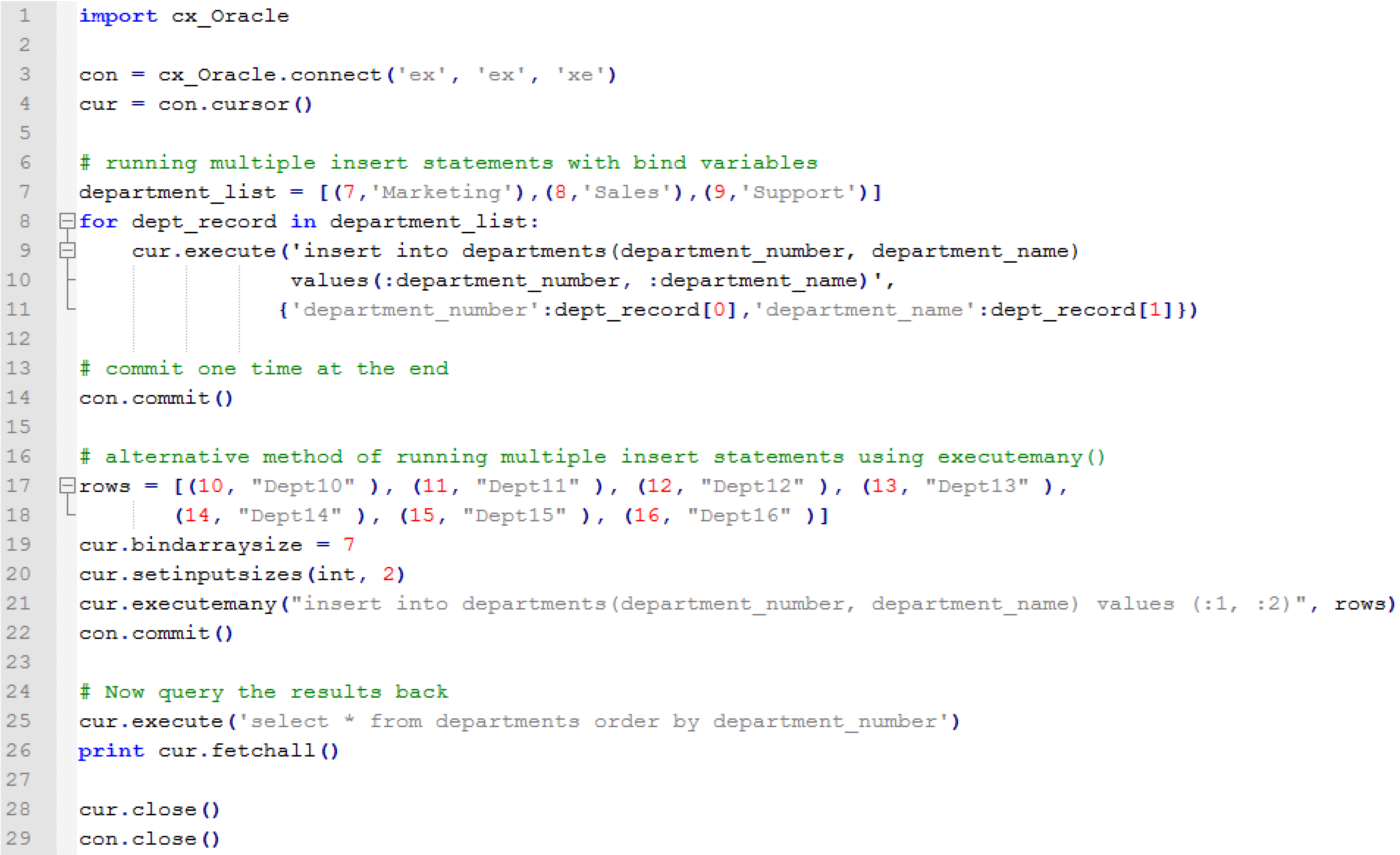
```python
import cx_Oracle

con = cx_Oracle.connect('ex', 'ex', 'xe')

# Query #1
cur = con.cursor()
cur.execute('select * from departments order by department_number')
for row in cur:
    print row

# Query #2
# Note: we are using the same cursor with a different SQL which is OK
cur.execute('select * from departments order by department_number desc')
for department_number, department_name in cur:
    print department_number, department_name

cur.close()
con.close()

```

Sample Transaction



```
import cx_Oracle

con = cx_Oracle.connect('ex', 'ex', 'xe')
cur = con.cursor()

# single insert statement
cur.execute("insert into departments(department_number, department_name)
             values(6, 'Research')")
con.commit()

# Now query the results back
# Note: we re-use the same cursor for the select statement
cur.execute('select * from departments order by department_number')
res = cur.fetchall()
print res

cur.close()
con.close()
```

Sample Bulk Transaction



```python
import cx_Oracle

con = cx_Oracle.connect('ex', 'ex', 'xe')
cur = con.cursor()

# running multiple insert statements with bind variables
department_list = [(7,'Marketing'),(8,'Sales'),(9,'Support')]
for dept_record in department_list:
    cur.execute('insert into departments(department_number, department_name)
                values(:department_number, :department_name)',
               {'department_number':dept_record[0],'department_name':dept_record[1]})

# commit one time at the end
con.commit()

# alternative method of running multiple insert statements using executemany()
rows = [(10, "Dept10" ), (11, "Dept11" ), (12, "Dept12" ), (13, "Dept13" ),
       (14, "Dept14" ), (15, "Dept15" ), (16, "Dept16" )]
cur.bindarraysize = 7
cur.setinputsizes(int, 2)
cur.executemany("insert into departments(department_number, department_name) values (:1, :2)", rows)
con.commit()

# Now query the results back
cur.execute('select * from departments order by department_number')
print cur.fetchall()

cur.close()
con.close()
```

cx_Oracle.Cursor Methods

- Cursors let you execute SQL and also return a resultset when applicable
- Cursors are equivalent to Statement + ResultSet in Java
- Obtain a cursor object by calling Connection.cursor()
- 3 SQL Processing Phases:

1-Parse (optional)

Cursor.parse([statement])

2-Execute

Cursor.execute(statement, [parameters], **keywords)

Cursor.executemany(statement, parameters)

3-Fetch (optional)

Cursor.fetchall()

Cursor.fetchmany([rows_no])

Tips

- Get started early, especially if new to Python!
- Read cx_Oracle documentation carefully:

  http://cx-oracle.readthedocs.org

- Check out sample code in cx_Oracle-doc/samples
- Go through a couple tutorials:

  1- Using Python With Oracle Database 11g:

  http://www.oracle.com/technetwork/articles/dsl/python-091105.html

  2- The Mastering Oracle+Python Series:

  http://www.oracle.com/technetwork/articles/dsl/prez-python-queries-101587.html

The Project

- What is it
- Choosing a project
- What is expected
- Phase 1 is mandatory, Phase 2 is optional

Project Ideas

- **Yelp Dataset Challenge:**

  http://www.yelp.com/dataset_challenge/

  -Get this year's Challenge dataset and load it into Oracle

  -Run some interesting analytics queries

  -Submit project to Yelp by June 30th, 2015

- **Product Inventory Database:**

  - Support regional warehouses

  - Track availability of items by region

  - Can use the OM schema as starting point

Note: You are not restricted to using Yelp's dataset. There are other datasets available on Freebase (http://www.freebase.com/).

Project Checkpoints

C1. Form Groups and send me an email. Due on Friday 02/27

C2. Project Proposal. Due on Wednesday 03/04

C3. ER Diagram. Due on Monday 03/09

C4. Sample Queries. Due on Wednesday 03/11

C5. Class Presentation. During week of 03/30

C6. Project Submission (code and final report). Due on Friday 04/03

Project Proposal

- Should be about **1 page** in length.
- Suggested content:

  -title and group members

  -short description of the project

  -list any interesting issues or unanswered questions

  -expected responsibilities/deliverables for each group member

  -important: tools and datasets you are planning to use

Project Presentation

- **10 minutes** per project: 7 minutes presentation plus 3 minutes for questions.
- Suggested content:

  -describe the problem

  -describe your approach

  -give short demo

  -discuss unexpected issues or problems

  -discuss possible extensions

Final Project Submission

- A one page report on how the project was implemented and how it works internally
- A brief description of the code that has been written
- A brief description of the experiments you ran to verify the solution
- End-user documentation (instructions and examples on how somebody can use this project)
- Submit all code including dataset and test cases

Next Class

- Review HW #3
- Discuss database transactions
- Work on project proposal