Moving on From SQL

Slowly

From last class

- From each department, find the names of the people earning the 2 highest salaries

```
with aaa as (select distinct dept_name from instructor),
     bbb as (select aaa.dept_name as deptt, salary as sall
                    from aaa join lateral (select dept_name, salary from instructor as sii
                                           where sii.dept_name=aaa.dept_name
                                           order by salary desc
                                           limit 2) as lsi
                                  on lsi.dept_name=aaa.dept_name
                    order by aaa.dept_name asc, lsi.salary desc)

select id, name, dept_name, salary
from instructor as i1, bbb
where i1.dept_name=bbb.deptt and i1.salary=bbb.sall
order by dept_name asc, salary desc;
```

Distinct!!! Why???

A brief intro to writing HTML

And the topics of the next several weeks

- By far, the most common way of interacting with DBs is through the web.
- So as a practical user of DBs you need to know something of how HTML and the web works
- Discussion of local forwarding was first instance of doing this.

From Lab

```
-- select one random use sorting and limit.
select *
from launch
order by random()
limit 1;



--in two different ways find all launches whose apogee was higher than the apogee of any launch in 1957.
    -- way 1: using the "all" operator
    -- way 2: using max
    -- compare the time required for each query ... which is faster, why?
    -- to turn on timing: \timing on

select apogee, date
from launch
where apogee > all ( select apogee
                     from launch
                     where date_part('year', date)=1957);

select apogee, date
from launch
where apogee > ( select max(apogee)
                 from launch
                 where date_part('year', date)=1957);
```

SQL -- using set operators

- Sakila
  - suppose there existed a table that contained only the first N rows of the film_category table.
    - e.g.,   select * from film_category limit N;
    - Get a list of the films that are NOT mentioned in this table
      - use [outer] join and set operators
        - Union [all], intersect, except

with fc as (select * from film_category limit 10) (select * from film left outer join fc on film.film_id=fc.film_id) except (select * from film join fc on film.film_id=fc.film_id);