03b_joins

Queries with more than 1 table

University DB

```
set search_path to university;
```

1. *Cartesian product*: match every row in the first table with every row in the second

```
select *
from offering,
     instructor;
```

2. *Join*: like the cartesian product, but only keep the interesting matching rows
   - here, only keep the corresponding instructor ids (column `iid`)

```
select *
from offering,
     instructor
where offering.iid = instructor.iid;
```

3. more modern way to write the same query: use an `inner join`

```
select *
from offering
         inner join instructor on offering.iid = instructor.iid;

select *
from offering as o
         inner join instructor as i on o.iid = i.iid;


select semester, year, section, i.name as instructor_name, c.name as course_name
from offering as o
         inner join instructor as i on o.iid = i.iid
        inner join course c on c.cid = o.cid;
```

4. (Almost) the same query with a `natural join`

   - differences: only 1 `iid` column, and columns in a different order
   - not recommend: the behavior of a natural join can be unpredictable

```
select *
from offering
         natural join instructor;

-- not work
select *
from offering
         natural join instructor
         natural join course;
```

5. Get the instructor ids and names of instructors teaching in the winter 2020 semester
   - must specify which of the 2 `iid` columns we want, even though there are equal

```
select instructor.iid, name
from offering
         inner join instructor on offering.iid = instructor.iid
where semester = 'W'
  and year = 2020;
```

6. Use `distinct` to remove duplicates

```
select distinct instructor.iid, name
from offering
         inner join instructor on offering.iid = instructor.iid
where semester = 'W'
  and year = 2020;
```

7. Get the course codes and names for courses offered in the winter 2020 semester

```
select distinct course.code, course.name
from course
         inner join offering on course.cid = offering.cid
where semester = 'W'
  and year = 2020;
```

8. Get the course codes and names for courses offered in the winter 2020 semester, along with the instructors names
   - first attempt: why this doesn't work?

   ```
   select code, course.name, instructor.name
   from offering
            natural join instructor
            natural join course
   where semester = 'W'
     and year = 2020;
   ```

- second attempt

```
select distinct code, course.name, instructor.name
from offering
         inner join instructor on offering.iid = instructor.iid
         inner join course on offering.cid = course.cid
where semester = 'W'
  and year = 2020;
```

9. What about the offerings without an instructor?
   - The `iid` in offering is allowed to be null, but not the `cid`
   - if `iid` is null in offering, it will not match anything from instructor
   - *outer joins*: keep the rows that don't match

```
select o.oid, o.iid, i.iid
from offering as o
         inner join instructor as i on o.iid = i.iid;

select o.oid, o.iid, i.iid
from offering o
         left outer join instructor i on o.iid = i.iid;

select o.oid, o.iid, i.iid
from offering o
         right outer join instructor i on o.iid = i.iid;

select o.oid, o.iid, i.iid
from offering o
         full outer join instructor i on o.iid = i.iid;
```

10. Get the course ids for courses offered in the year 2020, along with the instructors names

```
select distinct cid, instructor.name as instructor_name
from offering
         left join instructor on offering.iid = instructor.iid
where year = 2020;
```

11. Get the course codes and names for courses offered in the year 2020, along with the semester and the instructors names

```
select distinct code, course.name as course_name, semester, instructor.name as instructor_na
from (offering left join instructor on offering.iid = instructor.iid)
         right join course on offering.cid = course.cid
where year = 2020;
```

12. Find offerings without an instructor

```
select *
from offering
where iid is null;
```

13. Find students not enrolled in any course

```
select s.*
from student s
         left join enrollment e on s.sid = e.sid
where oid is null;
```

14. Find courses which have never been offered

```
select c.*
from course c
         left join offering o on c.cid = o.cid
where oid is null;
```

15. Find offerings in which no students are enrolled in

```
select o.*
from offering o
         left join enrollment e on o.oid = e.oid
where e.oid is null;
```