

Instructions:

1. Electronic submission: Your assignment is **due by 11:00 PM, 3/15**.
2. Accessing MySQL:
 - (a) You may only access MySQL from basalt, therefore write and test your queries on basalt.
 - (b) Create a directory called *hw5*. Write all your queries in *hw4*.
 - (c) Download **culinary.sql**; follow directions provided in lab2 to create the culinary tables in your database on basalt.
 - (d) Each query should be in a separate file named **qi.sql**, where $i = 1, 2, \dots, 6$. You should have **q1.sql, q2.sql, ..., q6.sql** corresponding to each query, by order.
3. Submit instructions:
 - (a) **Queries must be submitted from agate not basalt.**
 - (b) Copy your queries from basalt to agate using ftp, scp, or rsync. Copy files to a directory *hw4* on agate.
 - (c) From directory *hw4*, submit your queries using the command:
`~cs775/submit 5 q*`
 - (d) 5 is the assignment number. If you want to resubmit, then you need to use 5a, 5b, 5c,..... for assignment number (not 4).
 - (e) We have had submission problems in the past. In order to ensure that you get credit for your work, make a tar file of your final submission using the command
`tar -zcpvf hw4.tar q*`
Do not touch *hw5.tar* until you get back your graded assignment. The tar file keeps a dated copy of submitted files in your directory.
4. The TA will be grading your assignment by using the following command:
mysql -user username -password=password dbname < q1.sql > q1.out
where *q1.sql* is the input file and the output result is redirected to *q1.out*. Note that there are two hyphens (-) before -user and -password.
If the username is **xyz** and the password is **zzzzz** the grading command will be:
mysql -user xyz -password=zzzzz xyz < q1.sql > q1.out
for each query. The dbname, xyz, is the same as your username.
Please note that the TA will use a different instance of the database while grading.

5. Late policy: 1 day late: 2 points off, 2 days late: 4 points off; > 2 days late: will not be graded.
6. The relevant reading material is from Chapter 6 and Chapter 7.1.
7. The queries are mostly similar to the queries from previous assignments. I think that SQL queries are often easier than RA queries.
8. To test some of the queries, you may have to add data to the culinary database.

Notes about the database:

- The database stores information about different culinary courses. A course is offered by a school and consists of 1 or more levels. Each level is numbered for a course starting with 1 and increasing by 1. (See culinary.sql file for an example.)
- Staff members are either a chef or an assistant.
- Any staff member can also register to be a student in a class (on dates other than the ones s/he works).
- All queries regarding courses refer to just codes (do not check classdates); all queries regarding offerings refer to code + classdate.

Queries

For each of the following queries, write the SQL statement required to produce the desired output according to the “result schema.”

1. (5 points) **q1**: Retrieve names of all staff members and courses they register for as students, NULL otherwise. Order results in ascending order by name.

Result has schema (ssn, name, code, date).

```

+-----+-----+-----+-----+
| ssn   | name   | code  | classdate |
+-----+-----+-----+-----+
| 5555  | alice  | NULL  | NULL      |
| 1111  | joe    | cs1   | 2016-01-10 |
| 1111  | joe    | cs2   | 2016-01-11 |
| 1111  | joe    | cs3   | 2016-01-12 |
| 1111  | joe    | cs4   | 2016-01-04 |
| 7777  | marcus | NULL  | NULL      |
+-----+-----+-----+-----+
6 rows in set (0.000 sec)

```

2. (5 points) **q2:** For each offering, retrieve name of staff teaching the class, NULL if no staff assigned to the class.

Result has schema (code, date, ssn, name).

```

+-----+-----+-----+-----+
| code | classdate | ssn | name |
+-----+-----+-----+-----+
| cs1 | 2016-01-01 | NULL | NULL |
| cs1 | 2016-01-10 | NULL | NULL |
| cs2 | 2016-01-02 | NULL | NULL |
| cs2 | 2016-01-11 | NULL | NULL |
| cs3 | 2016-01-03 | NULL | NULL |
| cs3 | 2016-01-12 | 7777 | marcus |
| cs4 | 2016-01-04 | NULL | NULL |
| cs4 | 2016-01-13 | NULL | NULL |
| cs5 | 2016-01-05 | NULL | NULL |
| cs5 | 2016-01-14 | NULL | NULL |
| fc1 | 2015-11-19 | NULL | NULL |
| hb1 | 2016-01-08 | 1111 | joe |
| hb1 | 2016-01-17 | NULL | NULL |
| jp1 | 2016-01-06 | 5555 | alice |
| jp1 | 2016-01-15 | 5555 | alice |
| jp2 | 2016-01-07 | NULL | NULL |
| jp2 | 2016-01-16 | NULL | NULL |
| jp2 | 2016-02-16 | NULL | NULL |
| ln1 | 2016-01-09 | 5555 | alice |
| ln1 | 2016-01-18 | 7777 | marcus |
| ln1 | 2016-02-09 | NULL | NULL |
+-----+-----+-----+-----+
21 rows in set (0.001 sec)

```

3. (5 points) **q3:**

Retrieve courses with no non-staff registrations. (That is, the course only has students who are staff or the course has no registrations.)

Result has schema (school, course, level, classdate).

```

+-----+-----+-----+-----+
| school | course | level | classdate |
+-----+-----+-----+-----+
| Charlie's Cooking School | Sausages | 2 | 2016-01-02 |
| Charlie's Cooking School | Sausages | 4 | 2016-01-13 |

```

Jacque's Culinary Adventure	Pastries	2	2016-02-16
Lorenzo's Lean Cuisine	Napoli	1	2016-01-09
Lorenzo's Lean Cuisine	Napoli	1	2016-01-18
Lorenzo's Lean Cuisine	Napoli	1	2016-02-09

6 rows in set (0.000 sec)

4. (5 points) **q4:**

Retrieve students who register for all levels of a course (that is, there is a registration for every level of a course for that student). Order by ssn.

Result has schema (ssn, school, course).

ssn	school	course
2222	Jacque's Culinary Adventure	Pastries
2222	French Academy	Croissants
3333	Hans's Haute Cuisine	Bavarian
3333	Jacque's Culinary Adventure	Pastries
8888	Hans's Haute Cuisine	Bavarian

5 rows in set (0.001 sec)

5. (5 points) **q5:**

Retrieve student pairs where the first student takes at least every course the second takes.

Result has schema (ssn1, name1, ssn2, name2).

ssn1	name1	ssn2	name2
2222	mary	1111	joe
3333	steve	8888	tommy

2 rows in set (0.000 sec)

6. (5 points) **q6:** Retrieve non-staff student pairs where the first student takes none of the courses that the second takes.

Result has schema (ssn, ssn).

ssn	ssn
-----	-----

```
+-----+-----+
| 8888 | 2222 |
| 2222 | 8888 |
+-----+-----+
2 rows in set (0.000 sec)
```