MySQL problems with answers

Problem set 1

Write the following queries in SQL, using this schema. The expected output is given for each query when run on `large-university.db` (download this file and run SQLite locally viaMySQL problems
`sqlite3` large-university.db or upload to the browser-based version).

1. Find the names of those departments whose budget is higher than that of Astronomy. List them in alphabetic order.

```
SELECT
    dept_name
FROM
    department
WHERE
    budget > (SELECT
            budget
        FROM
            department
        WHERE
            dept_name = 'Astronomy');

# dept_name
'Athletics'
'Biology'
'Cybernetics'
'Finance'
'History'
'Math'
'Physics'
'Psychology'
```

2. Display a list of all instructors, showing each instructor's ID and the number of sections taught. Make sure to show the number of sections as 0 for instructors who have not taught any section.

```
SELECT
    I.ID, COUNT(T.ID) as number_of_sections
FROM
    instructor AS I
        NATURAL LEFT JOIN
    teaches AS T
```

```
GROUP BY I.ID
ORDER BY number_of_sections;

# ID, number_of_sections
'35579', '0'
'52647', '0'
'50885', '0'
'57180', '0'
'58558', '0'
'59795', '0'
'63395', '0'
'64871', '0'
'72553', '0'
'4034', '0'
'37687', '0'
'74426', '0'
'78699', '0'
'79653', '0'
'31955', '0'
'95030', '0'
'96895', '0'
'16807', '0'
'97302', '0'
'15347', '1'
'73623', '1'
'65931', '1'
'80759', '1'
'90376', '1'
'90643', '1'
'48570', '1'
'25946', '1'
'50330', '1'
'4233', '1'
'42782', '1'
'48507', '1'
'14365', '2'
'63287', '2'
'3335', '2'
'28400', '2'
'81991', '2'
'28097', '2'
'41930', '3'
'19368', '3'
'34175', '3'
'43779', '4'
'95709', '4'
'3199', '4'
'36897', '5'
'77346', '6'
'79081', '6'
```

```
'74420', '6'
'99052', '9'
'6569', '10'
'22591', '13'
```

3. For each student who has retaken a course at least twice (i.e., the student has taken the course at least three times), show the course ID and the student's ID. Please display your results in order of course ID and do not display duplicate rows.

```
select distinct course_id, ID
    from takes
    group by ID, course_id having count(*) > 2
    order by course_id;

# course_id, ID
'362', '16480'
'362', '16969'
'362', '27236'
'362', '39925'
'362', '39978'
'362', '44881'
'362', '49611'
'362', '5414'
'362', '69581'
'362', '9993'
```

4. Find the names of Biology students who have taken at least 3 Accounting courses.

```
select name
from student natural join (
    select ID from takes
    where course_id in ( select course_id from course
        where dept_name = "Accounting")
        group by ID having count(*) > 2
    ) as T where dept_name = "Biology";

SELECT s.name
FROM student s
JOIN takes t ON s.ID = t.ID
JOIN course c ON t.course_id = c.course_id
WHERE s.dept_name = 'Biology'
AND c.dept_name = 'Accounting'
GROUP BY s.ID
HAVING COUNT(*) > 2;

# name
'Michael'
```

```
'Dalton'
'Shoji'
'Wehen'
'Uchiyama'
'Schill'
'Kaminsky'
'Giannoulis'
```

5. Find the sections that had maximum enrollment in Fall 2010.

```sql
SELECT course_id, sec_id
FROM takes
WHERE semester = 'Fall' AND year = 2010
GROUP BY course_id, sec_id
HAVING COUNT(ID) = (
    SELECT MAX(enrollment_count)
    FROM (
        SELECT COUNT(ID) AS enrollment_count
        FROM takes
        WHERE semester = 'Fall' AND year = 2010
        GROUP BY course_id, sec_id
    ) AS subquery
);

select course_id, sec_id
from takes
WHERE semester = 'Fall' AND year = 2010
group by course_id, sec_id
order by count(*) desc
limit 1;

# course_id, sec_id
'867', '2'
```

6. Find student names and the number of law courses taken for students who have taken at least half of the available law courses. (These courses are named things like 'Tort Law' or 'Environmental Law').

```sql
select name, count(*)
from student as st
natural join takes as tt
where tt.course_id in (
    select course_id
    from course
    where title like "%Law%"
)
group by st.ID
```

```
having count(*) > (
    select count(*)/2
    from course
    where title like "%Law%"
);

# name, count(*)
'Nakajima', '4'
'Nikut', '4'
'Hahn-', '4'
'Nanda', '4'
'Schinag', '4'
```

7. Find the rank and name of the 10 students who earned the most A grades (A-, A, A+). Use alphabetical order by name to break ties. Note: the browser SQLite does not support window functions.

```
select row_number() over () as rnk, P.name from (
    select name, count(*) as cnt
    from student S
    join takes T on S.ID = T.ID
    where T.grade in ("A+", "A", "A-")
    group by T.ID
    ) as P
order by P.cnt desc, P.name
limit 10;

select row_number() over(order by count(*) desc, name) as rnk, name
from student st
natural join takes tt
where tt.grade in ("A+", "A", "A-")
group by ID
order by count(*) desc, name
limit 10;

# rnk, name
'1', 'Lepp'
'2', 'Eller'
'3', 'Masri'
'4', 'Vries'
'5', 'Åström'
'6', 'Gandhi'
'7', 'Greene'
'8', 'Haigh'
'9', 'McCarter'
'10', 'Sanchez'
```

> Here my answer didn't match with the sample because of the different sample database. My answer is based on the latest data, which can be grabbed from the official website :). Nothing to worry!

Problem set 2

1. Find out the ID and salary of the instructors.

```
select ID, salary from instructor;

-- small database
# ID, salary
'10101', '65000.00'
'12121', '90000.00'
'15151', '40000.00'
'22222', '95000.00'
'32343', '60000.00'
'33456', '87000.00'
'45565', '75000.00'
'58583', '62000.00'
'76543', '80000.00'
'76766', '72000.00'
'83821', '92000.00'
'98345', '80000.00'

-- big database
# ID, salary
'14365', '32241.56'
'15347', '72140.88'
'16807', '98333.65'
'19368', '124651.41'
'22591', '59706.49'
'25946', '90891.69'
'28097', '35023.18'
'28400', '84982.92'
'31955', '71351.42'
'3199', '82534.37'
'3335', '80797.83'
'34175', '115469.11'
'35579', '62579.61'
'36897', '43770.36'
'37687', '104563.38'
'4034', '61387.56'
'41930', '50482.03'
'4233', '88791.45'
'42782', '34272.67'
'43779', '79070.08'
'48507', '107978.47'
```

```
'48570', '87549.80'
'50330', '108011.81'
'50885', '32570.50'
'52647', '87958.01'
'57180', '43966.29'
'58558', '66143.25'
'59795', '48803.38'
'63287', '103146.87'
'63395', '94333.99'
'64871', '45310.53'
'6569', '105311.38'
'65931', '79866.95'
'72553', '46397.59'
'73623', '90038.09'
'74420', '121141.99'
'74426', '106554.73'
'77346', '99382.59'
'78699', '59303.62'
'79081', '47307.10'
'79653', '89805.83'
'80759', '45538.32'
'81991', '77036.18'
'90376', '117836.50'
'90643', '57807.09'
'95030', '54805.11'
'95709', '118143.98'
'96895', '119921.41'
'97302', '51647.57'
'99052', '93348.83'
```

2. Find out the ID and salary of the instructor who gets more than $85,000.

```
select ID, salary from instructor
where salary > 85000;

-- small db
# ID, salary
'12121', '90000.00'
'22222', '95000.00'
'33456', '87000.00'
'83821', '92000.00'

-- big db
# ID, salary
'16807', '98333.65'
'19368', '124651.41'
'25946', '90891.69'
'34175', '115469.11'
'37687', '104563.38'
```

```
'4233', '88791.45'
'48507', '107978.47'
'48570', '87549.80'
'50330', '108011.81'
'52647', '87958.01'
'63287', '103146.87'
'63395', '94333.99'
'6569', '105311.38'
'73623', '90038.09'
'74420', '121141.99'
'74426', '106554.73'
'77346', '99382.59'
'79653', '89805.83'
'90376', '117836.50'
'95709', '118143.98'
'96895', '119921.41'
'99052', '93348.83'
```

3. Find out the department names and their budget at the university.

```
select dept_name, budget from department;

-- small db
# dept_name, budget
'Biology', '90000.00'
'Comp. Sci.', '100000.00'
'Elec. Eng.', '85000.00'
'Finance', '120000.00'
'History', '50000.00'
'Music', '80000.00'
'Physics', '70000.00'

-- big db
# dept_name, budget
'Accounting', '441840.92'
'Astronomy', '617253.94'
'Athletics', '734550.70'
'Biology', '647610.55'
'Civil Eng.', '255041.46'
'Comp. Sci.', '106378.69'
'Cybernetics', '794541.46'
'Elec. Eng.', '276527.61'
'English', '611042.66'
'Finance', '866831.75'
'Geology', '406557.93'
'History', '699140.86'
'Languages', '601283.60'
'Marketing', '210627.58'
'Math', '777605.11'
```

```
'Mech. Eng.', '520350.65'
'Physics', '942162.76'
'Pol. Sci.', '573745.09'
'Psychology', '848175.04'
'Statistics', '395051.74'
```

4. List out the names of the instructors from Computer Science who have more than $70,000.

```
desc instructor;
select name from instructor
where salary > 70000;

-- small db
# name
'Wu'
'Einstein'
'Gold'
'Katz'
'Singh'
'Crick'
'Brandt'
'Kim'

-- big db
# name
'Bawa'
'Yazdi'
'Wieland'
'Liley'
'Atanassov'
'Moreira'
'Gustafsson'
'Bourrier'
'Bondi'
'Arias'
'Luo'
'Romero'
'Lent'
'Sarkar'
'Shuming'
'Bancilhon'
'Jaekel'
'McKinnon'
'Mingoz'
'Pimenta'
'Sullivan'
'Voronina'
```

```
'Kenje'
'Mahmoud'
'Levine'
'Valtchev'
'Bietzk'
'Sakurai'
'Mird'
'Dale'
```

5. For all instructors in the university who have taught some course, find their names and the course ID of
   all courses they taught.

```
select I.name, T.course_id
from instructor I
natural join teaches T
order by I.name;

-- small db
# name, course_id
'Brandt', 'CS-190'
'Brandt', 'CS-190'
'Brandt', 'CS-319'
'Crick', 'BIO-101'
'Crick', 'BIO-301'
'Einstein', 'PHY-101'
'El Said', 'HIS-351'
'Katz', 'CS-101'
'Katz', 'CS-319'
'Kim', 'EE-181'
'Mozart', 'MU-199'
'Srinivasan', 'CS-101'
'Srinivasan', 'CS-315'
'Srinivasan', 'CS-347'
'Wu', 'FIN-201'

-- large db
# name, course_id
'Atanassov', '603'
'Atanassov', '604'
'Bawa', '457'
'Bietzk', '158'
'Bondi', '571'
'Bondi', '274'
'Bondi', '747'
'Bourrier', '960'
'Bourrier', '949'
'Choll', '461'
```

```
'DAgostino', '663'
'DAgostino', '338'
'DAgostino', '338'
'DAgostino', '352'
'DAgostino', '400'
'DAgostino', '400'
'DAgostino', '991'
'DAgostino', '642'
'DAgostino', '599'
'DAgostino', '482'
'DAgostino', '962'
'DAgostino', '972'
'DAgostino', '867'
'Dale', '893'
'Dale', '237'
'Dale', '629'
'Dale', '237'
'Dale', '927'
'Dale', '748'
'Dale', '802'
'Dale', '158'
'Dale', '496'
'Gustafsson', '169'
'Gustafsson', '169'
'Gustafsson', '631'
'Gustafsson', '561'
'Jaekel', '852'
'Jaekel', '334'
'Kean', '366'
'Kean', '808'
'Lembr', '200'
'Lembr', '843'
'Lent', '626'
'Liley', '192'
'Luo', '679'
'Mahmoud', '704'
'Mahmoud', '735'
'Mahmoud', '735'
'Mahmoud', '493'
'Mahmoud', '864'
'Mahmoud', '486'
'Mingoz', '362'
'Mingoz', '527'
'Mingoz', '137'
'Mingoz', '362'
'Mingoz', '426'
'Mingoz', '304'
'Mingoz', '319'
'Mingoz', '445'
'Mingoz', '349'
```

```
'Mingoz', '362'
'Morris', '696'
'Morris', '791'
'Morris', '795'
'Morris', '313'
'Morris', '242'
'Pimenta', '875'
'Queiroz', '559'
'Romero', '105'
'Romero', '489'
'Romero', '105'
'Romero', '476'
'Sakurai', '468'
'Sakurai', '960'
'Sakurai', '258'
'Sakurai', '270'
'Sarkar', '867'
'Shuming', '468'
'Sullivan', '694'
'Tung', '692'
'Tung', '401'
'Tung', '421'
'Ullman ', '408'
'Ullman ', '974'
'Ullman ', '345'
'Ullman ', '200'
'Ullman ', '760'
'Ullman ', '408'
'Valtchev', '702'
'Valtchev', '415'
'Vicentino', '793'
'Voronina', '376'
'Voronina', '239'
'Voronina', '959'
'Voronina', '443'
'Voronina', '612'
'Voronina', '443'
'Wieland', '581'
'Wieland', '591'
'Wieland', '545'
```

6. Find the names of all instructors whose salary is greater than at least one instructor in the Biology
   department.

```sql
select name
from instructor
where dept_name = "Biology"
```

```
and salary > ( select min(S.salary) from (
    select salary
    from instructor
    where dept_name = "Biology"
) as S );

-- big db
# name
'Valtchev'
```

7. Find the advisor of the student with ID 12345

```
select * from advisor
where s_ID = 12345;

-- small db
# s_ID, i_ID
'12345', '10101'
```

8. Find the average salary of all instructors.

```
select avg(I.salary) average_salary from
(select salary from instructor) as I;

-- small db
# average_salary
'74833.333333'

-- big db
# average_salary
'77600.188200'
```

9. Find the names of all departments whose building name includes the substring 'Watson'.

```
select dept_name from department
where building like "%Watson%";

-- small db
# dept_name
'Biology'
'Physics'
```

10. Find the names of instructors with salary amounts between $90,000 and $100,000.

```
select name from instructor
where salary between 90000 and 100000;
```

```
-- small db
# name
'Wu'
'Einstein'
'Brandt'

-- large db
# name
'Yazdi'
'Liley'
'McKinnon'
'Sullivan'
'Mahmoud'
'Dale'
```

11. Find the instructor names and the courses they taught for all instructors in the Biology department who
    have taught some course.

```
select name, course_id
from instructor
natural left join teaches
where dept_name = "Biology";

-- small db
# name, course_id
'Crick', 'BIO-101'
'Crick', 'BIO-301'

-- large db
# name, course_id
'Queiroz', '559'
'Valtchev', '415'
'Valtchev', '702'
```

12. Find the courses taught in Fall-2009 semester.

```
select course_id from teaches
where semester = "Fall" and year = 2009;

-- big db
# course_id
'105'
'237'
'242'
'304'
```

```
'334'
'486'
'960'
```

13. Find the set of all courses taught either in Fall-2009 or in Spring-2010.

```
select course_id from teaches
where (semester = "Fall" and year = 2009) or (semester = "Spring" and year
= 2010);

( select course_id from teaches
where semester = "Fall" and year = 2009 )
union ( select course_id from teaches
where semester = "Spring" and year = 2010 );

-- big db
# course_id
'105'
'237'
'242'
'270'
'304'
'334'
'443'
'486'
'493'
'679'
'692'
'735'
'960'
```

14. Find the set of all courses taught in the Fall-2009 as well as in Spring-2010.

```
select course_id from teaches
where (semester = "Fall" and year = 2009) and (semester = "Spring" and
year = 2010);

( select course_id from teaches
where semester = "Fall" and year = 2009 )
intersect ( select course_id from teaches
where semester = "Spring" and year = 2010 );
```

15. Find all courses taught in the Fall-2009 semester but not in the Spring-2010 semester.

```
select course_id from teaches
where (semester = "Fall" and year = 2009) and not (semester = "Spring" and
year = 2010);
```

```
( select course_id from teaches
where semester = "Fall" and year = 2009 )
except ( select course_id from teaches
where semester = "Spring" and year = 2010 );

-- big db
# course_id
'105'
'237'
'242'
'304'
'334'
'486'
'960'
```

16. Find all instructors who appear in the instructor relation with null values for salary.

```
select * from instructor
where salary = NULL;
```

17. Find the average salary of instructors in the Finance department.

```
select avg(T.salary) from
( select salary from instructor
where dept_name = "Finance" ) as T;

-- small db
# avg(T.salary)
'85000.000000'

-- big db
# avg(T.salary)
'105311.380000'
```

18. Find the total number of instructors who teach a course in the Spring-2010 semester.

```
select  count(T.id) from
( select id from instructor
    natural join teaches
    where semester = "Spring"
    and year = 2010 ) as T ;

-- big db
# count(T.id)
'6'
```

19. Find the average salary in each department.

```
select dept_name, avg(salary)
from department
natural join instructor
group by dept_name;

-- small db
# dept_name, avg(salary)
'Biology', '72000.000000'
'Comp. Sci.', '77333.333333'
'Elec. Eng.', '80000.000000'
'Finance', '85000.000000'
'History', '61000.000000'
'Music', '40000.000000'
'Physics', '91000.000000'

-- big db
# dept_name, avg(salary)
'Accounting', '48716.592500'
'Athletics', '77098.198000'
'Pol. Sci.', '100053.073333'
'Psychology', '61143.050000'
'Languages', '57421.856667'
'English', '72089.050000'
'Statistics', '67795.441667'
'Elec. Eng.', '74162.740000'
'Comp. Sci.', '98133.470000'
'Marketing', '84097.437500'
'Astronomy', '79070.080000'
'Mech. Eng.', '79813.020000'
'Physics', '114576.900000'
'Cybernetics', '96346.567500'
'Finance', '105311.380000'
'Geology', '99382.590000'
'Biology', '61287.250000'
```

20. Find the number of instructors in each department who teach a course in the Spring-2010 semester.

```
select dept_name, count(ID)
from department
natural join instructor
where ID in (
    select ID from teaches
    where semester = "Spring"
    and year = 2010
)
```

```
group by dept_name;

--big db
# dept_name, count(ID)
'Athletics', '1'
'English', '2'
'Physics', '1'
'Geology', '1'
```

21. List out the departments where the average salary of the instructors is more than $42,000.

```
select dept_name
from department D
where ( select avg(salary) from (
    select salary
    from instructor I
    where D.dept_name = I.dept_name
) as T ) > 42000;

select distinct dept_name
from instructor
group by dept_name
having avg(salary) > 42000;

-- small db
# dept_name
'Biology'
'Comp. Sci.'
'Elec. Eng.'
'Finance'
'History'
'Physics'

-- big db
# dept_name
'Accounting'
'Astronomy'
'Athletics'
'Biology'
'Comp. Sci.'
'Cybernetics'
'Elec. Eng.'
'English'
'Finance'
'Geology'
'Languages'
'Marketing'
```

```
'Mech. Eng.'
'Physics'
'Pol. Sci.'
'Psychology'
'Statistics'
```

22. For each course section offered in 2009, find the average total credits (tot cred) of all students enrolled
in the section, if the section had at least 2 students.

```
select course_id, sec_id, avg(tot_cred)
from takes
natural join student
where year = 2009
group by sec_id, course_id
having count(*) > 1;

-- big db
# course_id, sec_id, avg(tot_cred)
'105', '1', '68.3578'
'237', '2', '65.6656'
'242', '1', '64.4576'
'304', '1', '64.9023'
'334', '1', '62.8806'
'486', '1', '64.8980'
'604', '1', '65.7233'
'960', '1', '66.0847'
'972', '1', '65.2607'
```

23. Find all the courses taught in both the Fall-2009 and Spring-2010 semesters.

```
select course_id from teaches
where (semester = "Fall" and year = 2009) and (semester = "Spring" and
year = 2010);

( select course_id from teaches
where semester = "Fall" and year = 2009 )
intersect ( select course_id from teaches
where semester = "Spring" and year = 2010 );
```

24. Find all the courses taught in the Fall-2009 semester but not in the Spring-2010 semester.

```
select course_id from teaches
where (semester = "Fall" and year = 2009) and not (semester = "Spring" and
year = 2010);
```

```
( select course_id from teaches
where semester = "Fall" and year = 2009 )
except ( select course_id from teaches
where semester = "Spring" and year = 2010 );

-- big db
# course_id
'105'
'237'
'242'
'304'
'334'
'486'
'960'
```

25. Select the names of instructors whose names are neither 'Mozart' nor 'Einstein'.

```
select name from instructor
where name not in ("Mozart", "Einstein");

-- small db
# name
'Srinivasan'
'Wu'
'El Said'
'Gold'
'Katz'
'Califieri'
'Singh'
'Crick'
'Brandt'
'Kim'
```

26. Find the total number of (distinct) students who have taken course sections taught by the instructor
with ID 110011.

```
select count(T.ID)
from takes T
natural join section S
where "110011" in (
    select ID
    from teaches ST
    where T.course_id = ST.course_id
    and T.sec_id = ST.sec_id
    and T.semester = ST.semester
```

```
    and T.year = ST.year
);

SELECT COUNT(DISTINCT t.ID) AS total_students
FROM takes t
JOIN teaches te ON t.course_id = te.course_id
               AND t.sec_id = te.sec_id
               AND t.semester = te.semester
               AND t.year = te.year
WHERE te.ID = '110011';
```

27. Find the ID and names of all instructors whose salary is greater than at least one instructor in the History
department.

```
select ID, name
from instructor
where salary > ( select min(T.salary) from (
    select salary
    from instructor
    where dept_name = "History"
) as T );

-- small db
# ID, name
'10101', 'Srinivasan'
'12121', 'Wu'
'22222', 'Einstein'
'33456', 'Gold'
'45565', 'Katz'
'58583', 'Califieri'
'76543', 'Singh'
'76766', 'Crick'
'83821', 'Brandt'
'98345', 'Kim'
```

28. Find the names of all instructors that have a salary value greater than that of each instructor in the
Biology department.

```
select name
from instructor
where salary > ( select max(T.salary) from (
    select salary
    from instructor
    where dept_name = "Biology"
) as T );
```

```
-- small db
# name
'Wu'
'Einstein'
'Gold'
'Katz'
'Singh'
'Brandt'
'Kim'

-- big db
# name
'Yazdi'
'Wieland'
'Liley'
'Atanassov'
'Gustafsson'
'Bourrier'
'Bondi'
'Arias'
'Luo'
'Romero'
'Lent'
'Sarkar'
'Shuming'
'Bancilhon'
'Jaekel'
'McKinnon'
'Mingoz'
'Pimenta'
'Sullivan'
'Voronina'
'Kenje'
'Mahmoud'
'Levine'
'Bietzk'
'Sakurai'
'Mird'
'Dale'
```

29. Find the departments that have the highest average salary.

```sql
select dept_name
from instructor
group by dept_name
order by avg(salary) desc
limit 1;
```

```
-- small db
# dept_name
'Physics'

-- small db
# dept_name
'Physics'
```

30. Find all courses taught in both the Fall 2009 semester and in the Spring-2010 semester.

```
select course_id from teaches
where (semester = "Fall" and year = 2009) and (semester = "Spring" and
year = 2010);

( select course_id from teaches
where semester = "Fall" and year = 2009 )
intersect ( select course_id from teaches
where semester = "Spring" and year = 2010 );
```

31. Find all students who have taken all the courses offered in the Biology department.

```
with cnt as (
    select count(course_id) as ct
    from course
    where dept_name = "Biology"
)
select ID
from takes
where course_id in (
    select course_id
    from course
    where dept_name = "Biology"
)
group by ID
having count(*) = (
    select ct from cnt
);
```

32. Find all courses that were offered at most once in 2009.

```
select course_id
from takes
where year = 2009
group by course_id
having count(*) = 1;
```

33. Find all courses that were offered at least twice in 2009.

```
select course_id
from takes
where year = 2009
group by course_id
having count(*) > 1;
```

34. Find the average instructors' salaries of those departments where the average salary is greater than
$42,000.

```
select dept_name
from department D
where ( select avg(salary) from (
    select salary
    from instructor I
    where D.dept_name = I.dept_name
) as T ) > 42000;

select distinct dept_name
from instructor
group by dept_name
having avg(salary) > 42000;

-- small db
# dept_name
'Biology'
'Comp. Sci.'
'Elec. Eng.'
'Finance'
'History'
'Physics'

-- big db
# dept_name
'Accounting'
'Astronomy'
'Athletics'
'Biology'
'Comp. Sci.'
'Cybernetics'
'Elec. Eng.'
'English'
'Finance'
'Geology'
'Languages'
'Marketing'
```

```
'Mech. Eng.'
'Physics'
'Pol. Sci.'
'Psychology'
'Statistics'
```

35. Find the maximum across all departments of the total salary at each department.

```sql
select sum(salary)
from department
natural join instructor
group by dept_name
order by sum(salary) desc
limit 1;

SELECT MAX(total_salary) AS max_total_salary
FROM (
    SELECT dept_name, SUM(salary) AS total_salary
    FROM instructor
    GROUP BY dept_name
) AS dept_salaries;

-- small db
# max_total_salary
'232000.00'

-- big db
# max_total_salary
'406772.65'
```

36. List all departments along with the number of instructors in each department.

```sql
select dept_name, count(ID)
from department
natural left join instructor
group by dept_name;

-- small data
# dept_name, count(ID)
'Biology', '1'
'Comp. Sci.', '3'
'Elec. Eng.', '1'
'Finance', '2'
'History', '2'
'Music', '1'
'Physics', '2'

-- big data
```

```
# dept_name, count(ID)
'Accounting', '4'
'Astronomy', '1'
'Athletics', '5'
'Biology', '2'
'Civil Eng.', '0'
'Comp. Sci.', '2'
'Cybernetics', '4'
'Elec. Eng.', '4'
'English', '4'
'Finance', '1'
'Geology', '1'
'History', '0'
'Languages', '3'
'Marketing', '4'
'Math', '0'
'Mech. Eng.', '2'
'Physics', '2'
'Pol. Sci.', '3'
'Psychology', '2'
'Statistics', '6'
```

Problem set 3

1. Find the titles of courses in the Comp. Sci. department that have 3 credits.

```
select title
from course
where dept_name = "Comp. Sci."
and credits = 3;

-- small db
# title
'Robotics'
'Image Processing'
'Database System Concepts'

-- large db
# title
'International Finance'
'Computability Theory'
'Japanese'
```

2. Find the IDs of all students who were taught by an instructor named Einstein; make sure there are no duplicates in the result.

```
select takes_table.ID
from takes takes_table
join teaches teaches_table
    on takes_table.course_id = teaches_table.course_id
    and takes_table.sec_id = teaches_table.sec_id
    and takes_table.semester = teaches_table.semester
    and takes_table.year = teaches_table.year
where teaches_table.ID = ( select ID from instructor
where name = "Einstein" ) ;

-- small db
# ID
'44553'
```

3. Find the ID and name of each student who has taken at least one Comp. Sci. course; make sure there are no duplicate names in the result.

```
select distinct ID, name
from student
natural join takes
where takes.course_id in (
    select course_id
    from course
    where dept_name = "Comp. Sci."
);

-- small db
# ID, name
'00128', 'Zhang'
'12345', 'Shankar'
'45678', 'Levy'
'54321', 'Williams'
'76543', 'Brown'
'98765', 'Bourikas'

-- big db
-- big output
```

4. Find the course id, section id, and building for each section of a Biology course.

```
select course_id, sec_id, building
from section
natural join course
where dept_name = "Biology";

-- small db
# course_id, sec_id, building
```

```
'BIO-101', '1', 'Painter'
'BIO-301', '1', 'Painter'

-- big db
# course_id, sec_id, building
'415', '1', 'Lamberton'
'559', '1', 'Lamberton'
'702', '1', 'Saucon'
```

5. Output instructor names sorted by the ratio of their salary to their department's budget (in ascending order).

```
select name
from instructor
natural left join department
order by (salary/ budget) asc;

-- small db
# name
'Mozart'
'Srinivasan'
'Singh'
'Wu'
'Katz'
'Crick'
'Brandt'
'Kim'
'El Said'
'Califieri'
'Gold'
'Einstein'

-- big data
# name
'Konstantinides'
'Kean'
'Tung'
'Queiroz'
'DAgostino'
'Lembr'
'Soisalon-Soininen'
'Yin'
'Desyl'
'Murata'
'Bawa'
'Bertolino'
'Hau'
'Pimenta'
```

```
'Ullman '
'Shuming'
'Gutierrez'
'Dale'
'McKinnon'
'Valtchev'
'Mingoz'
'Vicentino'
'Romero'
'Voronina'
'Yazdi'
'Arinb'
'Jaekel'
'Luo'
'Choll'
'Bietzk'
'Pingr'
'Liley'
'Sarkar'
'Bancilhon'
'Moreira'
'Sakurai'
'Lent'
'Morris'
'Atanassov'
'Wieland'
'Mahmoud'
'Arias'
'Gustafsson'
'Dusserre'
'Levine'
'Sullivan'
'Kenje'
'Mird'
'Bourrier'
'Bondi'
```

6. Output instructor names and buildings for each building an instructor has taught in. Include instructor names who have not taught any classes (the building name should be NULL in this case).

```sql
select name, building
from instructor
natural left join teaches
natural left join section;

-- small db
# name, building
```

```
'Srinivasan', 'Packard'
'Srinivasan', 'Watson'
'Srinivasan', 'Taylor'
'Wu', 'Packard'
'Mozart', 'Packard'
'Einstein', 'Watson'
'El Said', 'Painter'
'Gold', NULL
'Katz', 'Packard'
'Katz', 'Watson'
'Califieri', NULL
'Singh', NULL
'Crick', 'Painter'
'Crick', 'Painter'
'Brandt', 'Taylor'
'Brandt', 'Taylor'
'Brandt', 'Taylor'
'Kim', 'Taylor'

-- big db
# name, building
'Lembr', 'Saucon'
'Lembr', 'Fairchild'
'Bawa', 'Saucon'
'Yazdi', NULL
'Wieland', 'Saucon'
'Wieland', 'Alumni'
'Wieland', 'Saucon'
'DAgostino', 'Fairchild'
'DAgostino', 'Stabler'
'DAgostino', 'Lambeau'
'DAgostino', 'Lambeau'
'DAgostino', 'Main'
'DAgostino', 'Whitman'
'DAgostino', 'Chandler'
'DAgostino', 'Saucon'
'DAgostino', 'Fairchild'
'DAgostino', 'Taylor'
'DAgostino', 'Nassau'
'DAgostino', 'Taylor'
'DAgostino', 'Lamberton'
'Liley', 'Polya'
'Kean', 'Saucon'
'Kean', 'Polya'
'Atanassov', 'Taylor'
'Atanassov', 'Bronfman'
'Moreira', NULL
'Gustafsson', 'Gates'
'Gustafsson', 'Drown'
'Gustafsson', 'Main'
```

```
'Gustafsson', 'Taylor'
'Bourrier', 'Saucon'
'Bourrier', 'Lamberton'
'Bondi', 'Main'
'Bondi', 'Power'
'Bondi', 'Gates'
'Soisalon-Soininen', NULL
'Morris', 'Fairchild'
'Morris', 'Chandler'
'Morris', 'Saucon'
'Morris', 'Polya'
'Morris', 'Lamberton'
'Arias', NULL
'Murata', NULL
'Tung', 'Saucon'
'Tung', 'Gates'
'Tung', 'Taylor'
'Luo', 'Saucon'
'Vicentino', 'Nassau'
'Romero', 'Chandler'
'Romero', 'Taylor'
'Romero', 'Drown'
'Romero', 'Lamberton'
'Lent', 'Lamberton'
'Sarkar', 'Lamberton'
'Shuming', 'Power'
'Konstantinides', NULL
'Bancilhon', NULL
'Hau', NULL
'Dusserre', NULL
'Desyl', NULL
'Jaekel', 'Taylor'
'Jaekel', 'Gates'
'McKinnon', NULL
'Gutierrez', NULL
'Mingoz', 'Fairchild'
'Mingoz', 'Lamberton'
'Mingoz', 'Rathbone'
'Mingoz', 'Saucon'
'Mingoz', 'Lamberton'
'Mingoz', 'Alumni'
'Mingoz', 'Bronfman'
'Mingoz', 'Lamberton'
'Mingoz', 'Alumni'
'Mingoz', 'Saucon'
'Pimenta', 'Power'
'Yin', NULL
'Sullivan', 'Alumni'
'Voronina', 'Taylor'
'Voronina', 'Power'
```

```
'Voronina', 'Whitman'
'Voronina', 'Gates'
'Voronina', 'Lamberton'
'Voronina', 'Saucon'
'Kenje', NULL
'Mahmoud', 'Whitman'
'Mahmoud', 'Lamberton'
'Mahmoud', 'Taylor'
'Mahmoud', 'Drown'
'Mahmoud', 'Taylor'
'Mahmoud', 'Power'
'Pingr', NULL
'Ullman ', 'Chandler'
'Ullman ', 'Taylor'
'Ullman ', 'Taylor'
'Ullman ', 'Taylor'
'Ullman ', 'Garfield'
'Ullman ', 'Polya'
'Levine', NULL
'Queiroz', 'Lamberton'
'Valtchev', 'Lamberton'
'Valtchev', 'Saucon'
'Bietzk', 'Whitman'
'Choll', 'Main'
'Arinb', NULL
'Sakurai', 'Main'
'Sakurai', 'Power'
'Sakurai', 'Lambeau'
'Sakurai', 'Power'
'Mird', NULL
'Bertolino', NULL
'Dale', 'Taylor'
'Dale', 'Power'
'Dale', 'Fairchild'
'Dale', 'Taylor'
'Dale', 'Stabler'
'Dale', 'Saucon'
'Dale', 'Saucon'
'Dale', 'Fairchild'
'Dale', 'Saucon'
```

7. ~~Find the names of those departments whose budget is higher than that of Astronomy. List them in alphabetic order.~~
8. ~~Output instructor names and buildings for each building an instructor has taught in. Include instructor names who have not taught any classes (the building name should be NULL in this case).~~

9. ~~For each student who has retaken a course at least twice (i.e., the student has taken the course at least three times), show the course ID and the student's ID. Please display your results in order of course ID and do not display duplicate rows.~~
10. ~~Find the names of Biology students who have taken at least 3 Accounting courses~~
11. ~~Find the rank and name of the 10 students who earned the most A grades (A-, A, A+). Use alphabetical order by name to break ties. Note: the browser SQLite does not support window functions.~~