数据库系统实验

Lab of MySQL

数据科学与计算机学院　17大数据与人工智能

17341015　陈鸿峥

一、 Accessing the Database

需要先下载DDL-MySQL.sql和smallRelationsInsertFile.sql两个文件，然后用

```
mysql -u root -p
```

进入MySQL界面，然后

```
mysql> CREATE DATABASE school;
mysql> use school;
mysql> source DDL-MySQL.sql
mysql> source smallRelationsInsertFile.sql
```

即可进行实验。

二、 Basic SQL

问题 1. Find the names of all the instructors from Biology department

解答.

```
select name
from instructor
where dept_name = 'Biology';
```

问题 2. Find the names of courses in Computer science department which have 3 credits

解答.

```
select title
from course
where credits = 3;
```

问题 3. For the student with ID 12345 (or any other value), show all course_id and title of all courses registered for by the student.

解答.

```
select course_id, title
from takes natural join course
where ID = 12345;
```

问题 4. As above, but show the total number of credits for such courses (taken by that student). Don't display the tot_creds value from the student table, you should use SQL aggregation on courses taken by the student.

解答.

```
select sum(credits)
from takes natural join course
where ID = 12345;
```

问题 5. As above, but display the total credits for each of the students, along with the ID of the student; don't bother about the name of the student. (Don't bother about students who have not registered for any course, they can be omitted)

解答.

```
select ID, sum(credits)
from takes natural join course
group by ID;
```

问题 6. Find the names of all students who have taken any Comp. Sci. course ever (there should be no duplicate names)

解答.

```
select distinct S.name
from student as S,
  (select * from takes natural join course) as C
where S.ID = C.ID and C.dept_name = 'Comp. Sci.';
```

问题 7. Display the IDs of all instructors who have never taught a couse (Notesad1) Oracle uses the keyword minus in place of except; (2) interpret "taught" as "taught or is scheduled to teach")

解答.

```
-- (without except!)
select ID
from instructor
where ID not in
  (select ID
    from instructor natural join teaches);
```

问题 8. As above, but display the names of the instructors also, not just the IDs.

解答.

```
select name, ID
from instructor
where ID not in
  (select ID
    from instructor natural join teaches);
```

问题 9. You need to create a movie database. Create three tables, one for actors(AID, name), one for movies(MID, title) and one for actor_role(MID, AID, rolename). Use appropriate data types for each of the attributes, and add appropriate primary/foreign key constraints.

解答.

```
create table actors
  (AID    varchar(20),
   name   varchar(50),
   primary key (AID));

create table movies
  (MID    varchar(20),
   title  varchar(50),
   primary key (MID));

create table actor_role
  (MID    varchar(20),
   AID    varchar(20),
   rolename varchar(30),
   primary key (MID,AID,rolename),
   foreign key (MID) references movies(MID),
   foreign key (AID) references actors(AID));
```

问题 10. Insert data to the above tables (approx 3 to 6 rows in each table), including data for actor "Charlie Chaplin", and for yourself (using your roll number as ID).

解答.

```
delete from actor_role;
delete from actors;
delete from movies;
insert into actors values ('01','Charlie Chaplin');
insert into movies values ('M1','Modern Times');
insert into actor_role values ('M1','01','Worker');
insert into movies values ('M2','The Great Dictator');
insert into actor_role values ('M2','01','Adenoid');
insert into actor_role values ('M2','01','Barber');
insert into movies values ('M3','City Lights');
insert into actor_role values ('M3','01','Tramp');
insert into actors values ('02','Leslie Cheung');
insert into movies values ('M4','Farewell My Concubine');
insert into actor_role values ('M4','02','Dieyi Cheng');
insert into actors values ('03','Tom Hanks');
insert into movies values ('M5','Forrest Gump');
insert into actor_role values ('M5','03','Gump');
insert into actors values ('04','No-movie Actor');
```

问题 11. Write a query to list all movies in which actor "Charlie Chaplin" has acted, along with the number of roles he had in that movie.

解答.

```
select name, title, count(rolename)
from actor_role natural join movies natural join actors
where name = 'Charlie Chaplin'
group by MID;
```

问题 12. Write a query to list all actors who have not acted in any movie

解答.

```
drop view no_movie_actor;
create view no_movie_actor as
select name
from actors
where name not in (select distinct name
         from actor_role natural join actors);
select * from no_movie_actor;
```

问题 13. List names of actors, along with titles of movies they have acted in. If they have not acted in any movie, show the movie title as null. (Do not use SQL outerjoin syntax here, write it from scratch.)

解答.

```
select *
from ((select name, title
    from actors, movies, actor_role
    where actors.AID = actor_role.AID and movies.MID = actor_role.MID)
  union
  (select name as name, null as title
    from no_movie_actor))
  as result;
```

三、 Intermediate SQL

Using the university schema that you have write the following queries. In some cases you need to insert extra data to show the effect of a particular feature – this is indicated with the question. You should then show not only the query, but also the insert statements to add the required extra data.

问题 1. Find the maximum and minimum enrollment across all sections, considering only sections that had some enrollment, don't worry about those that had no students taking that section

解答.

```
select max(enrollment), min(enrollment)
from (select sec_id, semester, year, count(distinct id) as enrollment
  from takes
  group by sec_id, semester, year) as T;
```

问题 2. Find all sections that had the maximum enrollment (along with the enrollment), using a subquery.

解答.

```
with T(sec_id, semester, year, enrollment) as
  (select sec_id, semester, year, count(distinct id)
    from takes
    group by sec_id, semester, year)
select T.sec_id, T.semester, T.year, T.enrollment
from T, (select max(enrollment) as maxE, min(enrollment)
  from T) as res
where T.enrollment = res.maxE;
```

问题 3. As in in Q1, but now also include sections with no students taking them; the enrollment for such sections should be treated as 0. Do this in two different ways (and create require data for testing)

(a) Using a scalar subquery

(b) Using aggregation on a left outer join (use the SQL natural left outer join syntax)

解答.

```
-- Test data (need to preserve the constraints)
delete from course
  where course_id = 'CS-001';
delete from section
  where sec_id = '1' and semester = 'Fall' and year = '2010';
insert into course(course_id)
  values ('CS-001');
insert into section(course_id, sec_id, semester, year)
  values ('CS-001','1','Fall','2010');
-- Q3.1
select distinct sec_id, semester, year,
  (select count(distinct id)
    from takes
    where (takes.sec_id, takes.semester, takes.year) = (section.sec_id, section.
        ↪ semester, section.year)) as cnt
from section;
-- Q3.2
select distinct sec_id, semester, year, ifnull(cnt,0)
from section left outer join
  (select sec_id, semester, year, count(distinct id) as cnt
  from takes
  group by sec_id, semester, year) as T
using (sec_id, semester, year);
```

问题 4. Find all courses whose identifier starts with the string "CS-1"

解答.

```
select course_id, title
from section natural join course
where course_id like 'CS-1%';
```

问题 5. Find instructors who have taught all the above courses

(a) Using the "not exists ... except ..." structure

(b) Using matching of counts which we covered in class (don't forget the distinct clause!).

解答.

```
-- Q5.1
select distinct ID, name
from (select * from teaches natural join instructor) as T
where not exists (select cs_course.course_id
       from (select course_id
           from course
           where course_id like 'CS-1%') as cs_course
       where cs_course.course_id not in
         (select course_id
           from (select * from teaches natural join instructor) as U
           where U.name = T.name));

-- Q5.2
with U(course_id) as -- all course_id like 'CS-1%'
  (select distinct course_id
    from teaches natural join instructor
    where course_id like 'CS-1%')
select distinct ID, name
from (select * from teaches natural join instructor) as T
where ((select count(distinct course_id)
    from teaches natural join instructor
    where name = T.name and course_id like 'CS-1%')
  = (select count(course_id) from U));
```

问题 6. Insert each instructor as a student, with tot_creds = 0, in the same department

解答.

```
insert into student
  select ID, name, dept_name, '0'
  from instructor;
```

问题 7. Now delete all the newly added "students" above (note: already existing students who happened to have tot_creds = 0 should not get deleted)

解答.

```
delete from student
  where (ID, name, dept_name) in
    (select (ID, name, dept_name)
      from instructor);
```

问题 8. Some of you may have noticed that the tot_creds value for students did not match the credits from courses they have taken. Write and execute query to update tot_creds based on the credits passed, to bring the database back to consistency. (This query is provided in the book/slides.)

解答.

```
update student as S
set tot_cred = (
  select sum(credits)
  from takes natural join course
  where S.ID = takes.ID and takes.grade is not null);
```

问题 9. Update the salary of each instructor to 10000 times the number of course sections they have taught.

解答.

```
update instructor as I
set salary = 10000 * (
  select count(distinct sec_id, semester, year)
  from teaches as T
  where I.ID = T.ID);
```

问题 10. Create your own query: define what you want to do in English, then write the query in SQL. Make it as difficult as you wish, the harder the better.

解答.

```
-- Number of courses taught by instuctors of CS department registered by the
    ↪ students
select count(cs_course.course_id)
from ((select distinct course_id
    from instructor natural join teaches
    where dept_name = 'Comp. Sci.') as cs_course
  inner join
  (select distinct course_id from takes) as all_course
  on cs_course.course_id = all_course.course_id);
```