Assignment 1

Due Date: Mar 28, 2025, at 11:59 pm
10% of the final grade

Independently write the following queries in SQL. As they are similar to the Week 3 lab exercises, ensure you fully understand those lab exercises before working on the assignment. If you have questions about the lab exercises, you may discuss them with the tutors. However, please do not discuss the assignment with the tutors.

Submit all queries sequentially in a single **.sql** file. The queries will be executed in batch mode to verify their correctness. This assignment adopts a **black-box assessment**, meaning that each question will receive either full marks or none based on the results executed on a testing database on Duckdb. **So please ensure that duplicated rows are removed and that the ordering of the results is correct.**

1. Using the university schema, write an SQL query to find the name and ID of those Accounting students advised by an instructor in the Physics department. Order results by student ID. [1 marks]

   Answer:

   ```
   select student.ID, student.name
   from student, advisor, instructor
   where student.ID = s_ID
             and instructor.ID= i_ID
             and student.dept_name= 'Accounting'
             and instructor.dept_name= 'Physics'
   order by student.ID;
   ```

   ```
   +-------+---------+
   |  ID   |  name   |
   +-------+---------+
   | 28952 | Kennedy |
   | 63449 | Sellink |
   | 92864 | Rogers  |
   +-------+---------+
   ```

2. Using the university schema, write an SQL query to find the names of those departments whose budget is higher than that of Accounting. List them in alphabetic order. [1 marks]

   Answer:

   ```
   select A.dept_name from department A
   where A.budget > (select budget from department where dept_name = 'Accounting')
   order by A.dept_name;
   ```

   ```
   +-------------+
   |  dept_name  |
   +-------------+
   ```

```
| Astronomy   |
| Athletics   |
| Biology     |
| Cybernetics |
| English     |
| Finance     |
| History     |
| Languages   |
| Math        |
| Mech. Eng.  |
| Physics     |
| Pol. Sci.   |
| Psychology  |
+-------------+
```

3. Using the university schema, use SQL to do the following: 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 student ID. [1 marks]

   Answer:

```
select distinct course_id, ID
from takes
group by ID, course_id having count(*) > 2 order by 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. Using the university schema, write an SQL query to find the names and IDs of those instructors who teach every course taught in his or her department (i.e., every course that appears in the teach relation with the instructor's department name). Order result by name. [1.5 marks]

   Answer:

```
 select I.name, I.ID from instructor as I where not exists (
  select * from (teaches natural join course) as C where C.dept_name = I.dept_name
    and not exists (
      select * from teaches as T
      where T.ID = I.ID and T.course_id = C.course_id
    )
  )
order by name;
```

```
+------------+-------+
|    name    |  ID   |
+------------+-------+
| Atanassov  | 28400 |
| Bertolino  | 97302 |
| DAgostino  | 22591 |
| Gustafsson | 3199  |
| Kean       | 28097 |
| Lent       | 48507 |
| Liley      | 25946 |
| Mahmoud    | 77346 |
| Mingoz     | 6569  |
| Morris     | 36897 |
| Romero     | 43779 |
| Ullman     | 79081 |
| Voronina   | 74420 |
| Wieland    | 19368 |
+------------+-------+
```

5. Using the university schema, write an SQL query to find the name and ID of each History student whose name begins with the letter ‘D’ and who has taken less than five Music courses. Order result by name. [1.5 marks]

Answer:

```
select name, ID from student
where dept_name = 'History' and name like 'D%' and 5 > (
  select count(distinct course_id)
  from course
  where course.dept_name= 'Music' and not exists (
      select *
      from takes
      where takes.ID = student.ID
      and takes.course_id= course.course_id
  )
) order by name;
```

```
+-----------+-------+
|   name    |  ID   |
+-----------+-------+
| Davy      | 3739  |
| Denecker  | 18941 |
| Deshpande | 14023 |
| Douss     | 78552 |
| Dowey     | 53165 |
| Dubink    | 89051 |
| Durrant   | 62487 |
| Duxbury   | 29002 |
+-----------+-------+
```

6. Using the university schema, write an SQL query to find the ID and name of each instructor who has never given an A+ grade in any course she or he has taught. (Instructors who have never taught a course trivially satisfy this condition.) Order result by ID. [2 marks]

Answer:

```
select ID, name from instructor
except
select distinct instructor.ID, instructor.name
from instructor, teaches, takes
where instructor.ID = teaches.ID
        and teaches.course_id = takes.course_id
        and teaches.year = takes.year
        and teaches.semester= takes.semester
        and teaches.sec_id= takes.sec_id
        and takes.grade = 'A+'
order by ID;
```

```
+--------+-------------------+
|   ID   |       name        |
+--------+-------------------+
| 72553 | Yin               |
| 95030 | Arinb             |
| 97302 | Bertolino         |
| 64871 | Gutierrez         |
| 78699 | Pingr             |
| 57180 | Hau               |
| 35579 | Soisalon-Soininen |
| 58558 | Dusserre          |
| 59795 | Desyl             |
| 79653 | Levine            |
| 50885 | Konstantinides    |
| 16807 | Yazdi             |
| 37687 | Arias             |
| 52647 | Bancilhon         |
| 63395 | McKinnon          |
| 74426 | Kenje             |
| 4034  | Murata            |
| 96895 | Mird              |
| 31955 | Moreira           |
+--------+-------------------+
```

7. Using the university schema, write an SQL query to find section(s) with maximum enrollment. The result columns should appear in the order "courseid, secid, year, semester, num". Order result by courseid. (It may be convenient to use the with construct.) [2 marks]

Answer:

```
with enrol(courseid, secid, year, semester, num) as (
        select course_id, sec_id, year, semester, count(*)
        from section natural join takes
        group by (course_id, sec_id, year, semester)
        ),
        maxenrol(value) as (
        select max(e.num) from enrol as e
        )
select courseid, secid, year, semester, num from enrol, maxenrol where num = value
order by course_id;
```

```
+----------+-------+--------+----------+-----+
| courseid | secid |  year  | semester | num |
+----------+-------+--------+----------+-----+
| 192      | 1     | 2002.0 | Fall     | 338 |
| 362      | 1     | 2005.0 | Fall     | 338 |
+----------+-------+--------+----------+-----+
```