GROUP BY and HAVING: Solutions

Schema

Student(sID, surName, firstName, campus, email, cgpa)  Offering[dept, cNum] ⊆ Course[dept, cNum]
Course(dept, cNum, name, breadth)                      Took[sID] ⊆ Student[sID]
Offering(oID, dept, cNum, term, instructor)            Took[oID] ⊆ Offering[oID]
Took(sID, oID, grade)

Questions

1. Write a query to find the average grade, minimum grade, and maximum grade for each offering.

   Solution:

   ```sql
   select avg(grade), min(grade), max(grade)
   from Took
   group by oid;
   ```

   Output:

<table>
<thead>
<tr>
<th>avg</th>
<th>min</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>59.00</td>
<td>39</td>
<td>98</td>
</tr>
<tr>
<td>60.67</td>
<td>45</td>
<td>75</td>
</tr>
<tr>
<td>70.50</td>
<td>52</td>
<td>89</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75.00</td>
<td>54</td>
<td>96</td>
</tr>
<tr>
<td>78.00</td>
<td>78</td>
<td>78</td>
</tr>
<tr>
<td>83.00</td>
<td>71</td>
<td>89</td>
</tr>
<tr>
<td>(23 rows)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(1 row)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. Which of these queries is legal?

   SELECT surname, sid
   FROM Student, Took
   WHERE Student.sid = Took.sid
   GROUP BY sid;

   SELECT instructor, max(grade),
   count(Took.oid)
   FROM Took, Offering
   WHERE Took.oid = Offering.oid
   GROUP BY instructor;

   SELECT surname, Student.sid
   FROM Student, Took
   WHERE Student.sid = Took.sid
   GROUP BY campus;

   SELECT Course.dept, Course.cnum,
   count(oid), count(instructor)
   FROM Course, Offering
   WHERE Course.dept = Offering.dept and
   Course.cnum = Offering.cnum
   GROUP BY Course.dept, Course.cnum
   ORDER BY count(oid);
Solution: Here's the result of each:

```sql
SELECT surname, sid
FROM Student, Took
WHERE Student.sid = Took.sid
GROUP BY sid;
```

ERROR: column reference "sid" is ambiguous
LINE 1: SELECT surname, sid

```sql
instructor | max | count
------------+-----+-------
Heap        | 82  | 1
Miller      | 91  | 1
Johancsik   | 99  | 3 . . etc.
Mylopoulos  | 96  | 3
Percy       | 98  | 4
Mendel      | 75  | 3
```

(17 rows)

ERROR: column "student.surname" must appear in the GROUP BY clause or be used in an aggregate function
LINE 1: SELECT surname, Student.sid

```sql
department | cnum | count | count
------------+-----+-------+-------
ENV         | 200 | 1     | 1
. . . etc.
CSC         | 263 | 3     | 3
CSC         | 148 | 4     | 4
CSC         | 207 | 4     | 4
CSC         | 343 | 5     | 5
```

(17 rows)

3. Find the sid and minimum grade of each student with an average over 80.

Solution:

```sql
SELECT SID, min(grade)
FROM Took
GROUP BY sID
HAVING AVG(grade) > 80;
```

Output:

```
sid | min
-----+-----
98000 | 54
99999 | 52
(2 rows)
```
4. Find the sid, surname, and average grade of each student, but keep the data only for those students who have taken at least 10 courses.

Solution:

```sql
SELECT Student.sID, surname, avg(grade)
FROM Student, Took
WHERE Student.sID = Took.sID
GROUP BY Student.sID
HAVING count(grade) >= 10;
```

Output:

<table>
<thead>
<tr>
<th>sid</th>
<th>surname</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>98000</td>
<td>Fairgrieve</td>
<td>83.2000000000000000</td>
</tr>
<tr>
<td>99999</td>
<td>Ali</td>
<td>84.5833333333333333</td>
</tr>
<tr>
<td>157</td>
<td>Lakemeyer</td>
<td>75.9333333333333333</td>
</tr>
</tbody>
</table>

(3 rows)

5. For each student who has passed at least 10 courses, report their sid and average grade on the courses that they passed.

Solution:

```sql
SELECT sid, AVG(grade)
FROM took
WHERE grade >= 50
GROUP BY sid
HAVING count(*) >= 10;
```

Output:

<table>
<thead>
<tr>
<th>sid</th>
<th>avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>98000</td>
<td>83.2000000000000000</td>
</tr>
<tr>
<td>99999</td>
<td>84.5833333333333333</td>
</tr>
<tr>
<td>157</td>
<td>75.9333333333333333</td>
</tr>
</tbody>
</table>

(3 rows)

There is a lot going on here. Be sure you are clear on the difference between WHERE and HAVING, and which rows are left at the moment where the HAVING condition is checked for each group.

6. For each student who has passed at least 10 courses, report their sid and average grade on all of their courses.

Solution: Here, because we don’t want a filter applied (only passing grades count) when choosing which students to report on, but we don’t want that filter applied when we compute their average grade. A single query, with a single WHERE clause, can’t accomplish this. Views to the rescue!

```sql
CREATE VIEW Seniors AS
SELECT sid
FROM Took
WHERE grade >= 50
GROUP BY sid
HAVING count(*) >= 10;
```
SELECT Seniors.sid, AVG(grade)  
FROM Seniors, Took  
WHERE seniors.sid = Took.sid  
GROUP BY Seniors.sid;

Output:

| sid   | avg               |
|-------+-------------------|
| 98000 | 83.2000000000000000 |
| 99999 | 84.5833333333333333 |
| 157   | 75.9333333333333333 |

(3 rows)

Notice that the average for student 157 is different than it was in the previous question. This is because that student failed one course, and it now is allowed to pull down the reported average.

7. Which of these queries is legal?

```sql
SELECT dept  
FROM Took, Offering  
WHERE Took.oID = Offering.oID  
GROUP BY dept  
HAVING avg(grade) > 75;
```

```sql
SELECT Took.oID, avg(grade)  
FROM Took, Offering  
WHERE Took.oID = Offering.oID  
GROUP BY Took.oID  
HAVING avg(grade) > 75;
```

```sql
SELECT Took.oID, dept, cNum, avg(grade)  
FROM Took, Offering  
WHERE Took.oID = Offering.oID  
GROUP BY Took.oID  
HAVING avg(grade) > 75;
```

```sql
SELECT oID, avg(grade)  
FROM Took  
GROUP BY sID  
HAVING avg(grade) > 75;
```

Solution: Here’s the result of each:

```sql
depart | avg                   
-------+----------------------
EEB    | 82.0000000000000000
ANT    | 91.0000000000000000
HIS    | 72.0000000000000000
CSC    | 83.0000000000000000
```

(4 rows)

ERROR: column "offering.dept" must appear in the GROUP BY clause or be used in an aggregate function

LINE 1: SELECT Took.oID, dept, cNum, avg(grade)

```sql
SELECT Took.oID, avg(grade)  
FROM Took, Offering  
WHERE Took.oID = Offering.oID  
GROUP BY Took.oID  
HAVING avg(grade) > 75;
```

ERROR: column "took.oID" must appear in the GROUP BY clause or be used in an aggregate function

LINE 1: SELECT oID, avg(grade)