Guillaume Minette de Saint Martin

Advanced data Technologies

Lab 3

1. **(Easy) How many different grades each student has received?**

   select g_student, count(g_grade)

   from grades

   group by g_student;

```
--1.   (Easy) How many different grades each student has received?
select g_student, count(g_grade)
from grades
group by g_student;
```

Query Result

SQL | All Rows Fetched: 7 in 0,069 seconds

	G_STUDENT	COUNT(G_GRADE)
1	6	6
2	2	6
3	4	5
4	5	5
5	8	6
6	3	5
7	7	5

First group by students in grades tabele then count the number of grades per student.

2. **(Easy) Find all grades that have been received in first ten days of the month.**

*select **

from grades

where g_date < '2010.06.11';

```
--2.   (Easy) Find all grades that have been received in first ten days of the month.
select *
from grades
where g_date < '2010.06.11';
```

Query Result

SQL | All Rows Fetched: 20 in 0,008 seconds

	G_ID	G_COURSE	G_STUDENT	G_GRADE	G_DATE
1	20	5	2	7	2010.06.07
2	21	5	3	8	2010.06.07
3	22	5	4	7	2010.06.07
4	23	5	5	9	2010.06.07
5	24	5	6	2	2010.06.07
6	25	5	7	8	2010.06.07
7	26	5	8	9	2010.06.07
8	27	6	2	4	2010.06.01
9	28	6	3	8	2010.06.01
10	29	6	4	6	2010.06.01
11	30	6	5	8	2010.06.01
12	31	6	6	10	2010.06.01
13	32	6	7	10	2010.06.01
14	33	6	8	10	2010.06.01
15	34	7	2	8	2010.06.03
16	35	7	3	8	2010.06.03
17	36	7	4	8	2010.06.03
18	37	7	5	8	2010.06.03
19	38	7	6	8	2010.06.03
20	39	7	8	9	2010.06.03

Select only grades for which data is before the 11th of the month.

3. **(Easy) Sort courses according to length of their names.**

*select **

from courses

order by length(c_title);

```
--3.  (Easy) Sort courses according to length of their names.
select *
from courses
order by length(c_title);
```

Query Result

SQL | All Rows Fetched: 6 in 0,052 seconds

	C_ID	C_TITLE	C_TEACHER	C_C_POINTS
1	6	AI in Business	2	4
2	3	Systems Theory	3	4
3	5	Grid Networking	5	2
4	4	Knowledge Engineering	4	4
5	7	Enterprise Architecture	6	2
6	2	Advanced Data Technologies	2	4

First get length of their names then use it to order results.

4. **(Medium) Find how many full months ago has each grade been received.**

select floor(months_between(sysdate,g_date))

from grades;

```
--4.  (Medium) Find how many full months ago has each grade been received.
select floor(months_between(sysdate,g_date))
from grades;
```

Query Result

SQL | All Rows Fetched: 38 in 0,006 seconds

	FLOOR(MONTHS_BETWEEN(SYSDATE,G_DATE))
1	63
2	63
3	64
4	64
5	64
6	64
7	64
8	64
9	64
10	63

First get the current date then compute the diffenrence between this current date and the date of the grade and take only the integer part of the result to have full months.

5. **(Medium) Find average marks of all students and round them**

a) *Till the tenth of the grade.*

```
select g_student, round(avg(g_grade),1)

from grades

group by g_student;
```

```
--5.  (Medium) Find average marks of all students and round them
 --a.  Till the tenth of the grade.
 select g_student, round(avg(g_grade),1)
 from grades
 group by g_student;
```

Query Result

SQL | All Rows Fetched: 7 in 0,016 seconds

	G_STUDENT	ROUND(AVG(G_GRADE),1)
1	6	6,7
2	2	7
3	4	7,8
4	5	6,2
5	8	9
6	3	7
7	7	7,4

First group by students in grades table then conpute their mark average and round it with only one digit.

b) *To the nearest integer smaller than the average mark, for example from 7.9 to 7.*

```
select g_student, floor(avg(g_grade))

from grades

group by g_student;
```

```
--b.  To the nearest integer smaller than the average mark, for example from 7.9 to 7.
select g_student, floor(avg(g_grade))
from grades
group by g_student;
```

Query Result

SQL | All Rows Fetched: 7 in 0,009 seconds

	G_STUDENT	FLOOR(AVG(G_GRADE))
1	6	6
2	2	7
3	4	7
4	5	6
5	8	9
6	3	7
7	7	7

First group by students in grades table then conpute their mark average and take the integer part of the result.

c) *To the nearest integer grater than the average mark, for example from 7.05 to 8.*

select g_student, ceil(avg(g_grade))

from grades

group by g_student;

```
--c.  To the nearest integer grater than the average mark, for example from 7.05 to 8.
select g_student, ceil(avg(g_grade))
from grades
group by g_student;
```

Query Result

SQL | All Rows Fetched: 7 in 0,012 seconds

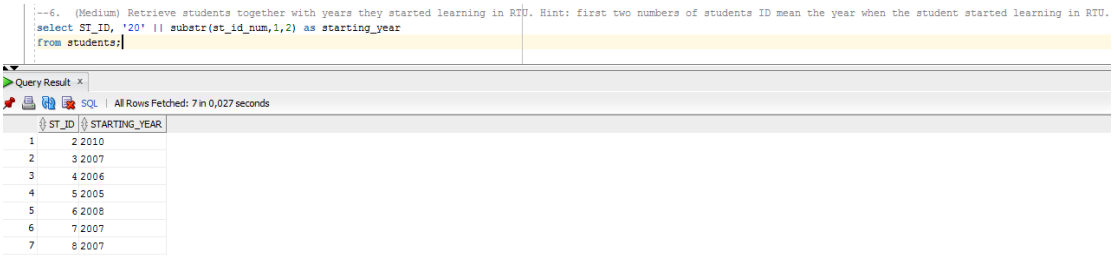
	G_STUDENT	CEIL(AVG(G_GRADE))
1	6	7
2	2	7
3	4	8
4	5	7
5	8	9
6	3	7
7	7	8

First group by students in grades table then conpute their mark average and take the closest bigest integer.

6. **(Medium) Retrieve students together with years they started learning in RTU. Hint: first two numbers of students ID mean the year when the student started learning in RTU.**

select ST_ID, '20' || substr(st_id_num,1,2) as starting_year

from students;



First create the string with the common part of the answers: 20 then add the 2 first numbers of student ID and name it as starting_year.

7. **(Hard) Do the same as in the previous query, but sort the results according to the years students entered RTU.**

select ST_ID, '20' || substr(st_id_num,1,2) as starting_year

from students

order by starting_year;

```
--7.  (Hard) Do the same as in the previous query, but sort the results according to the years students entered RTU.
select ST_ID, '20' || substr(st_id_num,1,2) as starting_year
from students
order by starting_year;
```

Query Result

SQL | All Rows Fetched: 7 in 0,009 seconds

	ST_ID	STARTING_YEAR
1	5	2005
2	4	2006
3	3	2007
4	8	2007
5	7	2007
6	6	2008
7	2	2010

First create the string with the common part of the answers: 20 then add the 2 first numbers of student ID and name it as starting_year and order results according to this starting year.

8. **(Hard) Calculate the average marks of each teacher and each course. Do it in the same query!**

select T_ID, C_ID, avg(G_GRADE)

from teachers, courses, grades

where G_COURSE=C_ID and C_TEACHER=t_id group by cube(T_ID,C_ID)

having((grouping(T_ID)=1 or grouping(C_ID)=1) and (grouping(T_ID)=0 or grouping(C_ID)=0));

```
--8.  (Hard) Calculate the average marks of each teacher and each course. Do it in the same query!
select T_ID, C_ID, avg(G_GRADE)
from teachers, courses, grades
where G_COURSE=C_ID and C_TEACHER=t_id group by cube(T_ID,C_ID)
having(( grouping(T_ID)=1 or grouping(C_ID)=1) and (grouping(T_ID)=0 or grouping(C_ID)=0));
```

Query Result

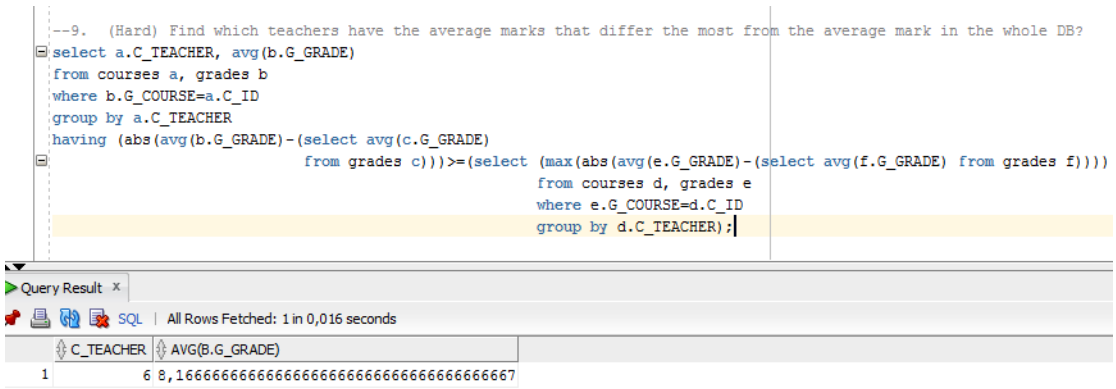
SQL | All Rows Fetched: 11 in 0,058 seconds

	T_ID	C_ID	AVG(G_GRADE)
1	(null)	6	8
2	(null)	2	6,8
3	(null)	4	7
4	(null)	5	7,1428571428571428571428571428571428571 4
5	(null)	3	6,7142857142857142857142857142857142857 1
6	(null)	7	8,1666666666666666666666666666666666666 7
7	6	(null)	8,1666666666666666666666666666666666666 7
8	2	(null)	7,5
9	4	(null)	7
10	5	(null)	7,1428571428571428571428571428571428571 4
11	3	(null)	6,7142857142857142857142857142857142857 1

First join tabeles teachers, grades and courses then group by cube to have the average marks and display only these that are related to one course or one teacher only.

9. **(Hard) Find which teachers have the average marks that differ the most from the average mark in the whole DB?**

```
select a.C_TEACHER, avg(b.G_GRADE)

from courses a, grades b

where b.G_COURSE=a.C_ID

group by a.C_TEACHER

having (abs(avg(b.G_GRADE)-(select avg(c.G_GRADE)

                    from grades c)))>=(select (max(abs(avg(e.G_GRADE)-
(select avg(f.G_GRADE) from grades f))))

                                from courses d, grades e

                                where e.G_COURSE=d.C_ID

                                group by d.C_TEACHER);
```



First join tables courses and grade then group by teacher to compute average but display only these that have the diference of average mark greater or equal to the maximum of the average marks diferences in the whole table.