Apellidos: González Mahagamage

Nombre: Iván

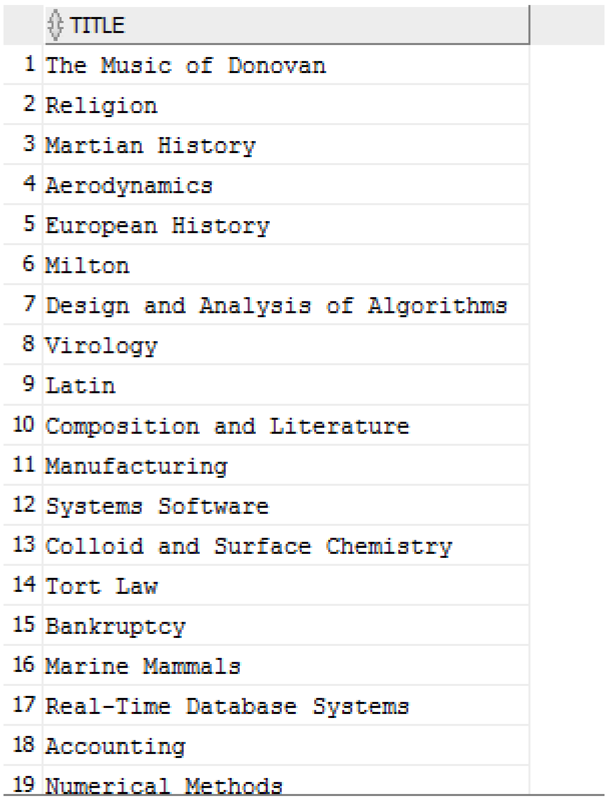
UO: 239795

RI - EJERCICIO CONSULTAS SQL

A. Basic SQL

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

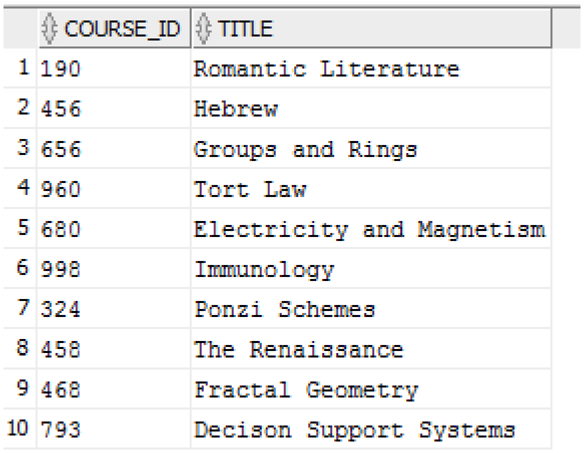
   SELECT DISTINCT c.TITLE
   FROM COURSE c
   WHERE c.CREDITS = 3;



```
   TITLE
 1 The Music of Donovan
 2 Religion
 3 Martian History
 4 Aerodynamics
 5 European History
 6 Milton
 7 Design and Analysis of Algorithms
 8 Virology
 9 Latin
10 Composition and Literature
11 Manufacturing
12 Systems Software
13 Colloid and Surface Chemistry
14 Tort Law
15 Bankruptcy
16 Marine Mammals
17 Real-Time Database Systems
18 Accounting
19 Numerical Methods
```

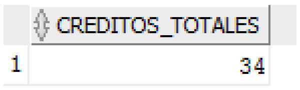
2. For the student with ID 12345 (or any other value), find all course_id and title of all courses registered for by the student and then show the total number of credits for such courses (taken by that student). Don't display the tot_creds value from the student table, you should use SQL aggregation on courses taken by the student.

   (He usado el ID = ‘1000’ para tener algún resultado)
   SELECT c.COURSE_ID, c.TITLE
   FROM COURSE c, STUDENT s
   WHERE c.DEPT_NAME = s.DEPT_NAME
   AND s.ID = '1000';



	COURSE_ID	TITLE
1	190	Romantic Literature
2	456	Hebrew
3	656	Groups and Rings
4	960	Tort Law
5	680	Electricity and Magnetism
6	998	Immunology
7	324	Ponzi Schemes
8	458	The Renaissance
9	468	Fractal Geometry
10	793	Decison Support Systems

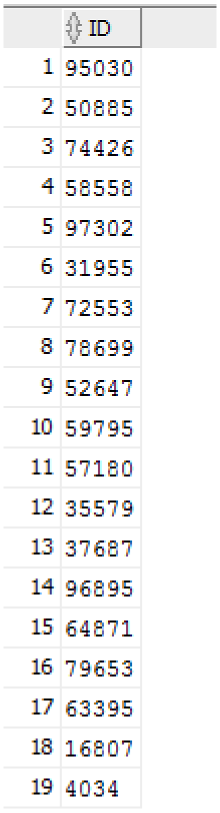
```
SELECT SUM(c.CREDITS) AS Creditos_totales
FROM COURSE c, STUDENT s
WHERE c.DEPT_NAME = s.DEPT_NAME
AND s.ID = '1000'
GROUP BY s.ID;
```



	CREDITOS_TOTALES
1	34

3. Display the IDs and names 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")

```
SELECT i.ID
FROM INSTRUCTOR i
WHERE i.ID NOT IN (
        SELECT t.ID
        FROM TEACHES t
);
```

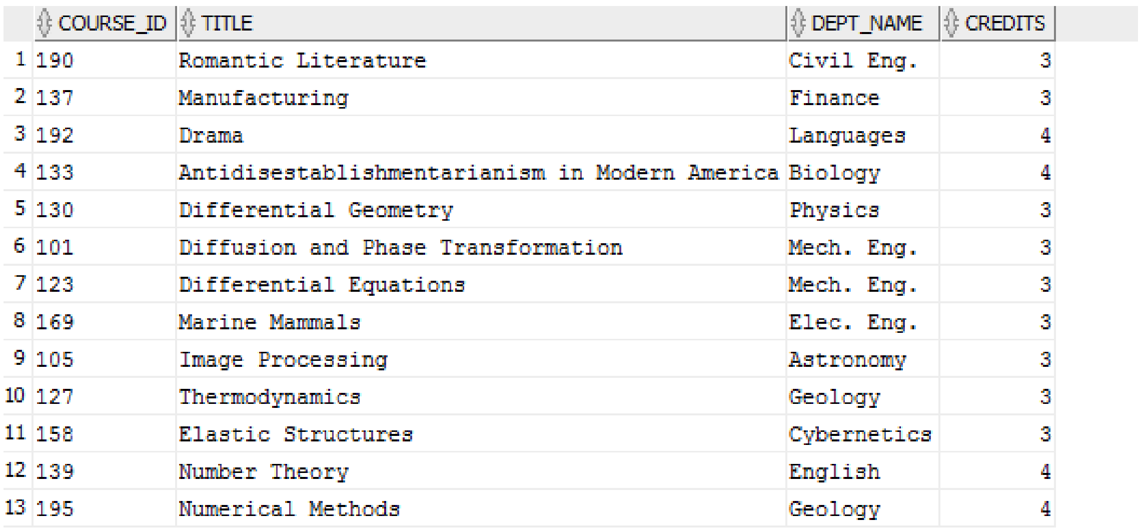


	ID
1	95030
2	50885
3	74426
4	58558
5	97302
6	31955
7	72553
8	78699
9	52647
10	59795
11	57180
12	35579
13	37687
14	96895
15	64871
16	79653
17	63395
18	16807
19	4034

B. Intermediate SQL

1. Find all courses whose identifier starts with the string "CS-1"
   He tenido que cambiar el string porque con la VERSIÓN" LARGA" DE LA BASE DE DATOS no hay ninguno que empieza por "CS-1".

   ```
   SELECT DISTINCT *
   FROM COURSE c
   WHERE c.COURSE_ID LIKE 'CS-1%';
   ```



	COURSE_ID	TITLE	DEPT_NAME	CREDITS
1	190	Romantic Literature	Civil Eng.	3
2	137	Manufacturing	Finance	3
3	192	Drama	Languages	4
4	133	Antidisestablishmentarianism in Modern America	Biology	4
5	130	Differential Geometry	Physics	3
6	101	Diffusion and Phase Transformation	Mech. Eng.	3
7	123	Differential Equations	Mech. Eng.	3
8	169	Marine Mammals	Elec. Eng.	3
9	105	Image Processing	Astronomy	3
10	127	Thermodynamics	Geology	3
11	158	Elastic Structures	Cybernetics	3
12	139	Number Theory	English	4
13	195	Numerical Methods	Geology	4

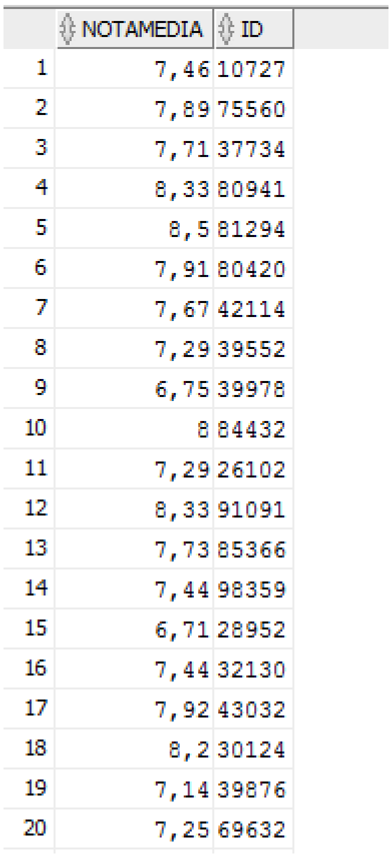
2. Some of you may have noticed that the tot_creds value for students did not match the credits from courses they have taken. Write and execute query to update tot_creds based on the credits passed, to bring the database back to consistency. (This query is provided in the book/slides.)

```
UPDATE STUDENT s set s.TOT_CRED =
        (SELECT SUM(c.credits) AS TOTAL
        FROM STUDENT s2, TAKES t, COURSE c
        WHERE s.ID = t.ID
        AND t.COURSE_ID = c.COURSE_ID
        AND s.id = s2.ID);
```

C. Advanced SQL

1. Grades are mapped to a grade point as follows: A:10, B:8, C:6, D:4 and F:0. Create a table to store these mappings, and write a query to find the average grade of each student, using this table. Make sure students who have not got a non-null grade in any course are displayed with an average of null.

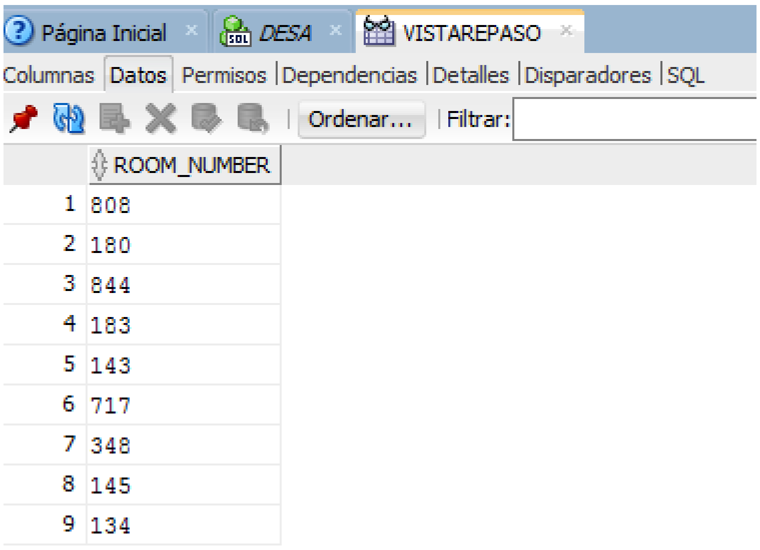
```
create table points(
        letra VARCHAR(2) NOT NULL,
        numero NUMERIC(2) NOT NULL,
        CONSTRAINT points_pk PRIMARY KEY (letra, numero)
);

insert into points values('A', 10);
insert into points values('A-', 9);
insert into points values('B+', 8.5);
insert into points values('B', 8);
insert into points values('B-', 7);
insert into points values('C+', 6.5);
insert into points values('C', 6);
insert into points values('C-', 5);
insert into points values('D', 4);
insert into points values('F', 0);
SELECT ROUND(AVG(p.NUMERO),2) AS notaMedia, t.ID
FROM takes t, points p
WHERE t.GRADE = p.LETRA
AND t.Grade != 'F'
GROUP BY t.ID;
```



	NOTAMEDIA	ID
1	7,46	10727
2	7,89	75560
3	7,71	37734
4	8,33	80941
5	8,5	81294
6	7,91	80420
7	7,67	42114
8	7,29	39552
9	6,75	39978
10	8	84432
11	7,29	26102
12	8,33	91091
13	7,73	85366
14	7,44	98359
15	6,71	28952
16	7,44	32130
17	7,92	43032
18	8,2	30124
19	7,14	39876
20	7,25	69632

2. Find all rooms that have been assigned to more than one section at the same time. Display the rooms along with the assigned sections; I suggest you use a with clause or a view to simplify this query.

```
CREATE VIEW vistaRepaso AS
        SELECT DISTINCT c.ROOM_NUMBER
        FROM CLASSROOM c, SECTION s1, SECTION s2, TIME_SLOT t1,
        TIME_SLOT t2
        WHERE c.BUILDING = s1.BUILDING
        AND c.ROOM_NUMBER = s1.ROOM_NUMBER
        AND c.BUILDING = s2.BUILDING
        AND c.ROOM_NUMBER = s2.ROOM_NUMBER
        AND s1.COURSE_ID != s2.COURSE_ID
        AND s1.TIME_SLOT_ID = s2.TIME_SLOT_ID;
```



	ROOM_NUMBER
1	808
2	180
3	844
4	183
5	143
6	717
7	348
8	145
9	134