Partiendo del script cfgs.sql, crea la base de datos correspondiente a los alumnos matriculados en algunos de los módulos de 1º y 2º curso de cfgs y sus correspondientes notas.

A la hora de ejecutar el script, no olvidarse antes de crear la base de datos y hacerla activa.

```
create database cfgs;
use cfgs;
```

OJO-→ al importar la base de datos con la conversión del juego de caracteres pueden aparecer caracteres ilegibles.

1. Generar el diagrama ER a través de la ingeniería inversa.
2. Añadir una nueva tabla a la base de datos, hacerlo manualmente con créate table:

   Profesores (dni_prof, nombre, apellido1, apellido2, direc, cod_asig)

   La clave primaria será el dni_prof y la clave foránea será cod_asig que apunta a la clave primaria cod de la tabla asignaturas.
3. Añadir la clave foránea con la instrucción alter
4. Insertar 3 profesores distintos que dan distintas asignaturas.
5. Crea una vista que muestre el nombre del alumno, nombre del módulo que cursa y notas de esos módulos.
6. Crear un índice para el campo apenom de la tabla alumnos. (con créate index o con alter y add index) ALTER TABLE cfgs.alumnos ADD INDEX ape (APENOM ASC);
7. Insertar un nuevo alumno con tus datos personales.
8. Insertar un 5, en la tabla notas, en la asignatura de FOL de aquellos alumnos que no tengan ninguna nota en ninguna asignatura. (será el alumno insertado en el paso anterior). Hacerlo con insert …select y haciendo la subconsulta.

```
Insert into notas(DNI,COD,NOTA)
select alumnos.DNI, asignaturas.COD, 5 from alumnos,
asignaturas where alumnos.DNI not in (select DNI from notas)
and asignaturas.nombre='FOL';
```

9. Subir la nota un Punto a todas las asignaturas de aquellos alumnos cuya nota media sea inferior a 6. Se puede hacer un paso previo y crear una vista para sacar la media.

```
create view media as
select dni, avg(nota) as medias from notas group by dni;

update notas set nota = nota+1
where dni in (select dni from media where medias<6);
```

10. Incrementa en 1 punto todas las notas de todos los alumnos en la asignatura FOL

    update notas n natural join asignaturas a set nota=nota+1 where a.nombre='FOL';
    update notas n inner join asignaturas a on n.cod =a.cod set nota=nota+1 where a.nombre='FOL';
    update notas n, asignaturas a set nota=nota+1 where a.nombre='FOL';
    update notas n, asignaturas a set nota=nota+1 where n.cod =a.cod and a.nombre='FOL';

11. Borra las notas del alumno cuyo DNI sea 12344345, hacerlo con una transacción comprobar que ha sido efectiva y luego hacer roolback. Poner las capturas que todo ha salido bien.

delete from notas where notas.dni=12344345

delete from notas where notas.dni=(select alumnos.dni from alumnos where apenom='alcalde garcía, elena') and cod= (select cod from asignaturas where nombre='análisis');

12.

13. Inserta un nuevo registro en la tabla notas para la alumna Díaz Fernández, María y para la asignatura Prog. Leng. Estr. con una nota de 6. (hacerlo con el insert ... select)

    insert into notas (dni, cod, nota) select alumnos.dni, asignaturas.cod, 6 from asignaturas, alumnos where asignaturas.nombre='Prog. Leng. Estr.' and alumnos.APENOM='Díaz Fernández, María' ;

14. Diseña un procedimiento al que pasamos como parámetro de entrada el nombre de uno de los módulos existentes en la BD y visualice el nombre de los alumnos que lo han cursado junto con su nota. Contralar si el módulo que se pasa existe o no. Al final del listado debe aparecer el nº de suspensos y el número de aprobados. Asimismo, deben aparecer al final los nombres y las notas de los alumnos que tengan la nota más alta y la más baja.

```
delimiter //

create procedure mostraralumnos(in parametro varchar(30))
begin
if not exists (select cod from asignaturas where nombre collate
latin1_spanish_ci=parametro collate latin1_spanish_ci) then
select 'Este módulo no existe';
else
select alumnos.APENOM, notas.nota from alumnos natural join notas natural join
asignaturas
where asignaturas.nombre collate latin1_spanish_ci=parametro collate
latin1_spanish_ci;

end if;

end
//
delimiter ;

call mostraralumnos('FOL');
```

```
delimiter //
create procedure actividad13 (in nombre_mod varchar(25))
begin
if exists (select NOMBRE from asignaturas where asignaturas.NOMBRE=nombre_mod collate latin1_spanish_ci)
then
select alumnos.APENOM, notas.NOTA from
alumnos natural join notas natural join asignaturas
where asignaturas.NOMBRE=nombre_mod collate latin1_spanish_ci;
select count(NOTA) as aprobados from notas natural join asignaturas where NOTA>=5 and asignaturas.NOMBRE=nombre_mod collate latin1_spanish_ci;
select count(NOTA) as suspensos from notas natural join asignaturas where NOTA<5 and asignaturas.NOMBRE=nombre_mod collate latin1_spanish_ci;
select APENOM, NOTA as mejor from alumnos natural join notas natural join asignaturas where NOTA =(select max(NOTA) from notas natural join asignaturas where asignaturas.NOMBRE=nombre_mod collate latin1_spanish_ci) and asignaturas.NOMBRE=nombre_mod collate latin1_spanish_ci;
select APENOM, NOTA as peor from alumnos natural join notas natural join asignaturas where NOTA =(select min(NOTA) from notas natural join asignaturas where asignaturas.NOMBRE=nombre_mod collate latin1_spanish_ci)and asignaturas.NOMBRE=nombre_mod collate latin1_spanish_ci;
else
select 'no existe';
end if;
end //

delimiter ;

call actividad13 ('FOL');
```

otra forma con unión all:

```
delimiter //
create procedure ejercicio13(in nom varchar(25))
begin
if exists(select asignaturas.NOMBRE from asignaturas where
asignaturas.NOMBRE = nom) then
select APENOM, NOTA from notas inner join alumnos on alumnos.DNI =
notas.DNI
where COD = (select COD from asignaturas where asignaturas.NOMBRE =
nom)
UNION ALL
select 'Alumnos aprobados:', count(*) from alumnos inner join notas
on alumnos.DNI = notas.DNI
where COD = (select COD from asignaturas where asignaturas.NOMBRE =
nom) and notas.NOTA >= 5
UNION ALL
select 'Alumnos suspendidos:', count(*) from alumnos inner join notas
on alumnos.DNI = notas.DNI
where COD = (select COD from asignaturas where asignaturas.NOMBRE =
nom) and notas.NOTA < 5
UNION ALL
select 'Nota más alta:', ''
UNION ALL
(select APENOM, NOTA from notas inner join alumnos on alumnos.DNI =
notas.DNI
where COD = (select COD from asignaturas where asignaturas.NOMBRE =
nom)
order by NOTA desc limit 1)
UNION ALL
select 'Nota más baja:', ''
UNION ALL
(select APENOM, NOTA from notas inner join alumnos on alumnos.DNI =
notas.DNI
where COD = (select COD from asignaturas where asignaturas.NOMBRE =
nom)
order by NOTA asc limit 1);
else
select "[ERROR] El módulo indicado no existe.";
end if;

end;
//
delimiter ;

call ejercicio13('ENTORNOS GRÁFICOS');

call ejercicio13('modulonoexistente');
```