Teniendo en cuenta el siguiente esquema relacional, respecto a la vuelta ciclista:

equipo (<u>equipo,</u> director)

maillot (<u>codigo,</u> tipo, color, premio)

ciclista (<u>dorsal,</u> nombre, edad, *equipo*)

ciclista.equipo → Equipo

etapa (<u>numero</u>, km, salida, llegada, *dorsal*)

etapa.dorsal → ciclista

puerto (<u>nompuerto,</u> altura, categoria, pendiente, *numero*, *dorsal*)

puerto.numero → etapa

puerto.dorsal → ciclista

llevar (<u>dorsal, numero,</u> codigo)

llevar.dorsal → ciclista

llevar.numero → etapa

llevar.codigo → maillot

Realizar las siguientes acciones sobre el esquema anterior:

1.- Obtener el numero de las etapa que tienen algún puerto de montaña, indicando cuántos tiene cada una de ellas.

```
select numero, count(*) "Nº Puertos"
from puerto
group by numero;
```

2.- Obtener los datos de las etapas que no comienzan en la misma ciudad en que acaba la etapa anterior.

```
SELECT E1.*   FROM ETAPA E1, ETAPA E2
 WHERE E1.numero = E2.numero + 1
  AND E1.salida <> E2.llegada;
```

3.- Obtener los datos de los ciclistas que han llevado todos los maillots (no necesariamente en la misma etapa)

```
SELECT C.dorsal, nombre, edad, equipo
  FROM CICLISTA C, LLEVAR L
 WHERE C.dorsal = L.dorsal
 GROUP BY C.dorsal, nombre, edad, equipo
  HAVING COUNT(DISTINCT L.codigo) = (SELECT COUNT(*)
                                     FROM MAILLOT);
```

4.- Todas las etapa que tienen 2 puertos de montaña han pasado a tener los mismos kms que la etapa número 5

```
 update etapa set km = (select km from etapa where numero=5)
where numero in
 (select numero from puerto p
  group by numero
  having count(numero)=2);
```

5- Todas las etapa que tienen 3 0 4 puertos de montaña y mas de 170 km han pasado a tener los mismos kms que la etapa número 5

```
update etapa set km = (select km from etapa where numero=5)
where numero in
 (select e.numero from puerto p, etapa e
  where p.numero=e.numero and km>170
  group by e.numero
  having count(e.numero)in (3,4));
```

6.- Crea una tabla llamada nuevamaillot con los mismo campos que maillot pero que tenga los datos de aquellos maillots que sólo han sido llevados por ciclista de un mismo equipo.

```
create table nuevamaillot as
SELECT M.codigo, M.tipo, M.color, M.premio
FROM LLEVAR L, CICLISTA C, MAILLOT M
WHERE L.dorsal=C.dorsal   AND L.codigo=M.codigo
GROUP BY  M.codigo, M.tipo, M.color, M.premio
HAVING COUNT(DISTINCT C.equipo)=1;
```

7.- Poner el nombre en mayúscula y aumenta un año a los ciclista que han ganado los puertos de mayor alturas

```
update ciclista set nombre=upper(nombre), edad=edad+1
where dorsal in
 (select dorsal from puerto where altura =
   (select max(altura) from puerto) );
```

8.- Da de alta un ciclista con el dorsal 101, tu nombre, la misma edad y equipo del ciclista que ha ganado la etapa 8

```
insert into ciclista
select distinct 101,'tu nombre', edad, equipo from ciclista c, etapa e
where c.dorsal=e.dorsal and e.numero=8;
```

9.- Crea una tabla llamada ganadores que contenga el dorsal y nombre de los ciclistas que han llevado dos o mas maillote en una misma etapa.
La tabla también ha de contener el equipo del ciclista, la etapa y el número de de maillot que han llevado en esa etapa.
Los campos de la nueva tabla han de tener los siguientes nombres:
Dorsal, Nombre, Equipo, Etapa, Nmaillot

```
CREATE TABLE GANADORES (dorsal, nombre, equipo, etapa, nmaillot) AS
SELECT DISTINCT c.dorsal, nombre,  equipo, numero, count(*)
  FROM CICLISTA C, LLEVAR L
 WHERE C.dorsal = L.dorsal
 GROUP BY numero, c.dorsal, nombre, equipo, numero
HAVING COUNT(*) >=2;
```

10.- Borra los maillot que han sido llevados por Melchor Mauri.
Nota: has de poner el borrado en cascada en la tabla llevar que referencia a maillot

```
ALTER TABLE LLEVAR DROP CONSTRAINT FK_LLEVAR_MAI;

ALTER TABLE LLEVAR ADD CONSTRAINT FK_llevar_mai FOREIGN KEY (codigo)
REFERENCES maillot (codigo) on delete cascade;

    delete maillot where codigo in (
select distinct codigo from ciclista c, llevar l
  where nombre='Melchor Mauri' and
  c.dorsal=l.dorsal );
```