99

CONSULTAS SQL

SOLUCIONES PROPUESTAS

ÍNDICE DE CONTENIDO

1. SOLUCIONES CICLISMO

DIAGRAMA CONCEPTUAL

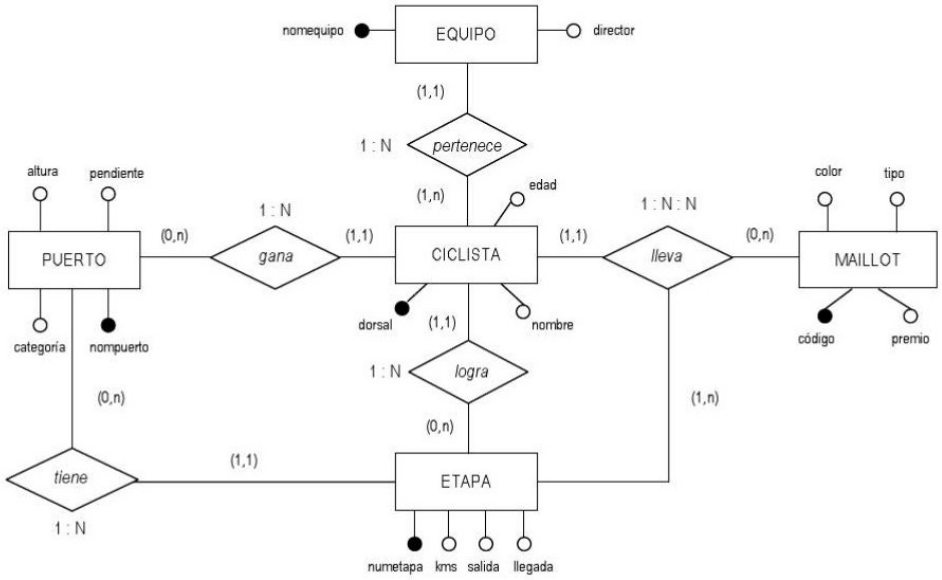
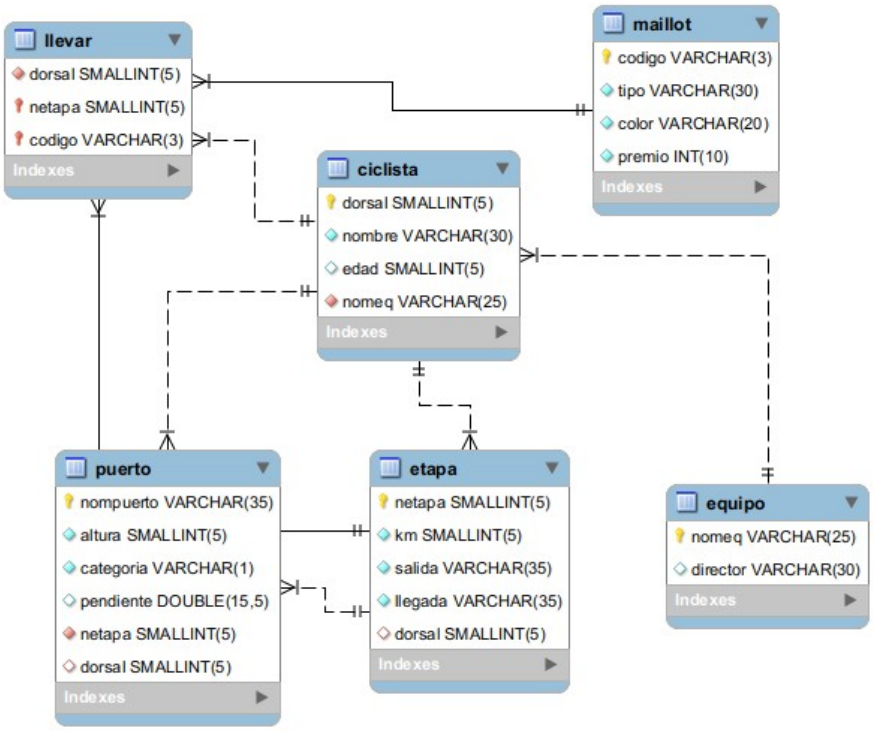


DIAGRAMA FÍSICO



!==========

!CONSULTAS SOBRE UNA SOLA RELACIÓN

!==========

!==========

!CICLISMO #1. Obtener el código, el tipo, el color y el premio de todos los maillots que hay.

!==========

```
SELECT * FROM maillot;
```

!==========

!CICLISMO #2. Obtener el dorsal y el nombre de los ciclistas cuya edad sea menor o igual que 25 años.

!==========

```
SELECT dorsal, nombre
FROM ciclista
WHERE edad <= 25;
```

!==========

!CICLISMO #3. Obtener el nombre y la altura de todos los puertos de categoría 'E' (Especial).

!==========

```
SELECT nompuerto, altura
FROM puerto
WHERE categoria = 'E';
```

!==========

!CICLISMO #4. Obtener el valor del atributo netapa de aquellas etapas con salida y llegada en la misma ciudad.

!==========

```
SELECT netapa
FROM etapa
WHERE salida=llegada;
```

!==========

!CICLISMO #5. ¿Cuántos ciclistas hay?

!==========

```
SELECT COUNT(*) FROM ciclista;
```

```
!============================
!CICLISMO #6. ¿Cuántos ciclistas hay con edad superior a 25 años?.
!============================
SELECT COUNT(*)
FROM ciclista
WHERE edad > 25;

!============================
!CICLISMO #7. ¿Cuántos equipos hay?.
!============================
SELECT COUNT(*)
FROM equipo;

!============================
!CICLISMO #8. Obtener la media de edad de los ciclistas.
!============================
SELECT AVG(edad)
FROM ciclista;

!============================
! Ejercicio nº 9. Obtener la altura mínima y máxima de los puertos de montaña.
!============================
SELECT MIN(altura), MAX(altura)
FROM Puerto;

!================================================
!CONSULTAS SOBRE VARIAS TABLAS
!================================================

!============================
!CICLISMO #10. Obtener el nombre y la categoría de los puertos ganados por ciclistas del equipo 'Banesto'.
!============================
SELECT nompuerto, categoria
FROM ciclista c, puerto p
WHERE c.dorsal=p.dorsal AND c.nomeq like 'Banesto';
```

!==========

```
SELECT nompuerto, categoria
FROM ciclista c, puerto p
WHERE c.dorsal=p.dorsal AND c.nomeq='Banesto';
```

!==========

!CICLISMO #11. Obtener el nombre de cada puerto indicando el número (netapa) y los kilómetros de la etapa en la que se encuentra el puerto.

!==========

```
SELECT nompuerto, e.netapa, km
FROM puerto p, etapa e
WHERE p.netapa=e.netapa;
```

!==========

!CICLISMO #12. Obtener el nombre y el director de los equipos a los que pertenezca algún ciclista mayor de 33 años.

!==========

```
SELECT DISTINCT e.nomeq, director
FROM equipo e, ciclista c
WHERE e.nomeq=c.nomeq AND c.edad >33;
```

!==========

```
SELECT nomeq, director
FROM equipo
WHERE nomeq IN
    (SELECT nomeq
    FROM ciclista
    WHERE edad>33);
```

!==========

!CICLISMO #13. Obtener el nombre de los ciclistas con el color de cada maillot que hayan llevado.

!==========

```
SELECT DISTINCT c.nombre,m.color
FROM maillot m, llevar l, ciclista c
WHERE m.codigo=l.codigo AND c.dorsal=l.dorsal;
```

!============================

!CICLISMO #14. Obtener pares de nombre de ciclista y número de etapa tal que ese ciclista haya ganado esa etapa habiendo llevado el maillot de color 'Amarillo' al menos una vez.

!============================

```
SELECT DISTINCT c.nombre, e.netapa
FROM ciclista c, etapa e, llevar l, maillot m
WHERE e.dorsal=c.dorsal AND l.dorsal=c.dorsal AND l.codigo=m.codigo
AND m.color like 'Amarillo';
```

!============================

```
SELECT DISTINCT c.nombre, e.netapa
FROM ciclista c, etapa e, llevar l, maillot m
WHERE e.dorsal=c.dorsal AND l.dorsal=c.dorsal AND l.codigo=m.codigo
AND m.color='Amarillo';
```

!============================

!CICLISMO #15. Obtener el valor del atributo netapa de las etapas que no comienzan en la misma ciudad en que acabó la anterior etapa.

!============================

```
SELECT e2.netapa
FROM etapa e1, etapa e2
WHERE e1.llegada <> e2.salida AND e1.netapa + 1 = e2.netapa;
```

!==

!CONSULTAS CON SUBCONSULTAS

!==

!============================

!CICLISMO #16. Obtener el valor del atributo netapa y la ciudad de salida de aquellas etapas que no tengan puertos de montaña.

!============================

```
SELECT netapa, salida
FROM etapa e
WHERE NOT EXISTS
    (SELECT * FROM puerto p
    WHERE p.netapa=e.netapa );
```

```
SELECT netapa, salida
FROM etapa
WHERE netapa NOT IN
    (SELECT netapa FROM puerto);

!============================
!CICLISMO #17. Obtener la edad media de los ciclistas que han ganado alguna etapa
!============================
SELECT AVG(edad)
FROM ciclista
WHERE dorsal IN
    (SELECT dorsal FROM etapa);

Este cuenta varias veces los que han ganado varias etapas->
SELECT AVG(DISTINCT edad)
FROM ciclista c, etapa e
WHERE c.dorsal= e.dorsal

!============================
!CICLISMO #18. Selecciona el nombre de los puertos con una altura superior a la altura media de todos los puertos.
!============================
SELECT nompuerto
FROM puerto
WHERE altura>
    (SELECT AVG(altura) FROM puerto);

!============================
!CICLISMO #19. Obtener el nombre de la ciudad de salida y de llegada de las etapas donde estén los puertos con mayor pendiente.
!============================
SELECT DISTINCT e.salida, e.llegada
FROM etapa e, puerto p
WHERE e.netapa=p.netapa AND
p.pendiente=
    (SELECT MAX(pendiente) FROM puerto );
```

!==========

!CICLISMO #20. Obtener el dorsal y el nombre de los ciclistas que han ganado los puertos de mayor altura.

!==========

```
SELECT DISTINCT c.dorsal, c.nombre
FROM puerto p, ciclista c
WHERE c.dorsal=p.dorsal AND p.altura=
    (SELECT MAX(altura) FROM puerto);
```

!==========

!CICLISMO #21. Obtener el nombre del ciclista más joven.

!==========

```
SELECT nombre
FROM ciclista
WHERE edad =
    ( SELECT MIN(edad) FROM ciclista );

SELECT nombre
FROM ciclista c
WHERE NOT EXISTS
    (SELECT *
    FROM ciclista c1
    WHERE c1.edad < c.edad);
```

!==========

!CICLISMO #22. Obtener el nombre del ciclista más joven que ha ganado al menos una etapa.

!==========

```
SELECT DISTINCT c.nombre
FROM ciclista c, etapa e
WHERE c.dorsal=e.dorsal AND
c.edad =
    (SELECT MIN(c2.edad) FROM ciclista c2, etapa e2
    WHERE c2.dorsal=e2.dorsal);
```

!============================

!CICLISMO #23. Obtener el nombre de los ciclistas que han ganado más de un puerto.

!============================

```
SELECT nombre
FROM ciclista c
WHERE 1<
    (SELECT COUNT(*) FROM puerto p
    WHERE p.dorsal = c.dorsal );

SELECT DISTINCT nombre
FROM ciclista c, puerto p1, puerto p2
WHERE c.dorsal = p1.dorsal AND c.dorsal=p2.dorsal AND p1.nom_puerto <> p2.nom_puerto
```

!==

!CONSULTAS CON CUANTIFICACIÓN UNIVERSAL

!==

!============================

!CICLISMO #24. Obtener el valor del atributo netapa de aquellas etapas tales que todos los puertos que están en ellas tienen más de 700 metros de altura.!

============================

```
SELECT e.netapa
FROM etapa e
WHERE e.netapa IN
    (SELECT netapa FROM puerto)
AND NOT EXISTS
    (SELECT * FROM puerto p WHERE p.altura <=700 AND e.netapa =p.netapa);

SELECT DISTINCT e.netapa
FROM etapa e, puerto p2
WHERE e.netapa=p2.netapa AND
NOT EXISTS
    (SELECT * FROM puerto p WHERE p.altura <=700 AND e.netapa =p.netapa);

SELECT DISTINCT p2.netapa
FROM puerto p2
```

```
WHERE NOT EXISTS
    (SELECT * FROM puerto p
    WHERE p.altura <=700 AND p2.netapa =p.netapa);

!============================
```

!CICLISMO #25. Obtener el nombre y el director de los equipos tales que todos sus ciclistas son mayores de 25 años.

```
!============================
SELECT e.nomeq, e.director
FROM Equipo e
WHERE e.nomeq IN
    (SELECT nomeq FROM Ciclista c)
AND NOT EXISTS
    (SELECT * FROM Ciclista c
    WHERE c.edad<26 AND c.nomeq=e.nomeq);

SELECT DISTINCT e.nomeq, e.director
FROM Equipo e, Ciclista C2
WHERE e.nomeq = c2.nomeq
AND NOT EXISTS
    (SELECT * FROM Ciclista c
    WHERE c.edad<26 AND c.nomeq=e.nomeq);

!============================
```

!CICLISMO #26. Obtener el dorsal y el nombre de los ciclistas tales que todas las etapas que han ganado tienen más de 170 km (es decir que sólo han ganado etapas de más de 170 km).

```
!============================
SELECT c.dorsal,c.nombre
FROM ciclista c
WHERE c. dorsal IN
    (SELECT dorsal FROM etapa)
AND NOT EXISTS
    (SELECT * FROM Etapa e2
    WHERE e2.km <=170 AND e2.dorsal= c.dorsal);

SELECT DISTINCT c.dorsal,c.nombre
FROM ciclista c, etapa e3
```

```
WHERE c. dorsal= e3.dorsal
AND NOT EXISTS
    (SELECT * FROM Etapa e2
    WHERE e2.km <=170 AND e2.dorsal= c.dorsal);
```

!==========

!CICLISMO #27. Obtener el nombre de los ciclistas que han ganado todos los puertos de una etapa y además han ganado esa misma etapa.

!==========

```
SELECT DISTINCT c.nombre
FROM ciclista c, etapa e
WHERE e.dorsal=c.dorsal AND
NOT EXISTS
    (SELECT * FROM puerto p
    WHERE p.netapa=e.netapa AND c.dorsal <> p.dorsal )
AND EXISTS
    ( SELECT * FROM puerto p
    WHERE p.netapa=e.netapa );

SELECT DISTINCT c.nombre
FROM ciclista c, etapa e, puerto p
WHERE e.dorsal=c.dorsal AND p.netapa=e.netapa
AND NOT EXISTS
    (SELECT * FROM puerto p2
    WHERE p2.netapa=e.netapa AND
    c.dorsal <> p.dorsal );
```

!==========

!CICLISMO #29. Obtener el código y el color de aquellos maillots que sólo han sido llevados por ciclistas de un mismo equipo.

!==========

```
SELECT DISTINCT m.codigo, m.color
FROM maillot m, llevar l, ciclista c
WHERE c.dorsal=l.dorsal AND m.codigo=l.codigo
AND NOT EXISTS
    (SELECT * FROM llevar l2, ciclista c2
    WHERE c2.dorsal=l2.dorsal AND
    c2.nomeq<>c.nomeq AND l2.codigo=l.codigo);
```

!==========

!CICLISMO #30. Obtener el nombre de aquellos equipos tales que sus ciclistas sólo hayan ganado puertos de 1ª categoría.

!==========

```
SELECT e.nomeq
FROM equipo e
WHERE NOT EXISTS
    (SELECT * FROM ciclista c, puerto p
    WHERE c.dorsal = p.dorsal
    AND p.categoria NOT like '1'
    AND c.nomeq=e.nomeq)
AND EXISTS
    ( SELECT * FROM ciclista c2, puerto p2
    WHERE c2.dorsal =p2.dorsal AND c2.nomeq=e.nomeq);
```

Otra: Obtener el nombre de aquellos equipos tales que sus ciclistas sólo hayan ganado puertos de 1ª categoría.

```
SELECT DISTINCT c.nomeq
FROM ciclista c, puerto p
WHERE c.dorsal = p.dorsal
AND NOT EXISTS
(SELECT * FROM puerto p2, ciclista c2
WHERE p2.dorsal = c2.dorsal AND p2.categoria <> '1' AND c2.nomeq = c.nomeq)
```

!==========

!CONSULTAS AGRUPADAS

!==========

!==========

!CICLISMO #31 Obtener el valor del atributo netapa de aquellas etapas que tienen puertos de montaña indicando cuántos tiene.

!==========

```
SELECT e.netapa, COUNT(*)
FROM etapa e, puerto p
WHERE e.netapa=p.netapa
GROUP BY e.netapa;

SELECT netapa, COUNT(*)
FROM puerto
GROUP BY netapa;
```

!==========

!CICLISMO #32. Obtener el nombre de los equipos que tengan ciclistas indicando cuántos tiene cada uno.

!==========

```
SELECT E.nomeq, COUNT(*)
FROM EQUIPO E left join ciclista c on E.NOMEQ=C.NOMEQ
GROUP BY E.nomeq;
```

!==========

!CICLISMO #33 Obtener el nombre de todos los equipos indicando cuántos ciclistas tiene cada uno.

!==========

```
SELECT nomeq, COUNT(*) FROM ciclista
GROUP BY nomeq
UNION
    SELECT nomeq, 0  FROM equipo
    WHERE nomeq NOT IN (SELECT nomeq FROM ciclista);
```

Otra opción:

```
SELECT e.nomeq, COUNT(dorsal)
FROM equipo e LEFT JOIN ciclista c ON e.nomeq=c.nomeq
GROUP BY e.nomeq;
```

!==========

!CICLISMO #34 Obtener el director y el nombre de los equipos que tengan más de 3 ciclistas y cuya edad media sea inferior o igual a 30 años.

!==========

```
SELECT e.director, e.nomeq
FROM ciclista c, equipo e
WHERE c.nomeq=e.nomeq
GROUP BY e.director, e.nomeq
HAVING COUNT(*) >3 AND AVG(edad) <= 30;
```

!==========

!CICLISMO #35 Obtener el nombre de los ciclistas que pertenezcan a un equipo que tenga más de cinco corredores y que hayan ganado alguna etapa indicando cuántas etapas ha ganado.

!==========

```
SELECT c.nombre, COUNT(*)
FROM ciclista c, etapa e
WHERE c.dorsal=e.dorsal
AND 5<
    ( SELECT COUNT(*) FROM ciclista c2
    WHERE c2.nomeq=c.nomeq )
GROUP BY c.nombre, c.dorsal;
```

!==========

!CICLISMO #36. Obtener el nombre de los equipos y la edad media de sus ciclistas de aquellos equipos que tengan la media de edad máxima de todos los equipos.

!==========

```
SELECT C.nomeq, AVG(c.edad)
FROM ciclista c
GROUP BY c.nomeq
HAVING AVG(c.edad) >= ALL
    (SELECT AVG(d.edad) FROM ciclista d  GROUP BY d.nomeq);
```

```
SELECT C.nomeq, AVG(c.edad)
FROM ciclista c
GROUP BY c.nomeq
HAVING AVG(c.edad) =
    SELECT MAX(edad)  FROM
        (SELECT AVG(d.edad) as edad FROM ciclista d GROUP BY d.nomeq);
```

```
!============================
```

!CICLISMO #37 Obtener el director de los equipos cuyos ciclistas han llevado más días maillots de cualquier tipo. Nota: cada tupla de la relación Llevar indica que un ciclista ha llevado un maillot un día

```
!============================
SELECT e.director
FROM equipo e, ciclista c, llevar l
WHERE e.nomeq = c.nomeq AND c.dorsal = l.dorsal
GROUP BY e.director, e.nomeq
HAVING COUNT(*) >= ALL
    (SELECT COUNT(*)
    FROM ciclista d, llevar m
    WHERE d.dorsal = m.dorsal
    GROUP BY d.nomeq);
```

```
!================================================
!CONSULTAS GENERALES
!================================================
```

```
!============================
```

!CICLISMO #38. Obtener el código y el color del maillot que ha sido llevado por algún ciclista que no ha ganado ninguna etapa.

```
!============================
SELECT DISTINCT m.codigo, m.color
FROM ciclista c, llevar l, maillot m
WHERE c.dorsal=l.dorsal AND m.codigo=l.codigo AND
c.dorsal NOT IN
     (SELECT e.dorsal FROM etapa e);
```

```
SELECT DISTINCT m.codigo, m.color
FROM llevar l, maillot m
WHERE m.codigo=l.codigo AND
l.dorsal NOT IN
    (SELECT e.dorsal FROM etapa e);

SELECT DISTINCT m.codigo, m.color
FROM llevar l, maillot m
WHERE m.codigo=l.codigo AND
NOT EXIST  (SELECT e.dorsal FROM etapa e WHERE e.dorsal = l.dorsal);
```

!==========

!CICLISMO #39. Obtener el valor del atributo netapa, la ciudad de salida y la ciudad de llegada de las etapas de más de 190 km. Y que tengan por lo menos dos puertos.

!==========

```
SELECT e.netapa, e.salida, e.llegada
FROM etapa e
WHERE e.km>190 AND 2<=
    (SELECT COUNT(*) FROM puerto p
    WHERE p.netapa=e.netapa );

SELECT DISTINCT e.netapa, e.salida, e.llegada
FROM etapa e, puerto p, puerto p2
WHERE .e.km>190 AND
e.netapa = p.netapa AND p2.netapa=e.netapa AND p.nompuerto <> p2.nompuerto;

SELECT e.netapa, e.salida, e.llegada
FROM etapa e
WHERE e.km>190 AND e.netapa IN
    (SELECT e2.netapa
    FROM etapa e2, puerto p
    WHERE e2.netapa=p.netapa
    GROUP BY e2.netapa
    HAVING COUNT(*) >1);
```

!==========

!CICLISMO #40. Obtener el dorsal y el nombre de los ciclistas que no han llevado todos los maillots que ha llevado el ciclista de dorsal 20

!==========

```
SELECT c.dorsal, c.nombre
FROM ciclista c
WHERE EXISTS
    (SELECT * FROM llevar l
    WHERE l.dorsal=20 AND
    NOT EXISTS
        (SELECT * FROM llevar l2
        WHERE l2.dorsal=c.dorsal AND
        l2.codigo=l.codigo)
    );
```

Tenemos que encontrar un maillot del 20 que no haya llevado el ciclista C. En la subquery obtengo todos los maillots del ciclista C (SELECT * FROM llevar l2 WHERE l2.dorsal=c.dorsal) y digo que no tiene que existir 1 que haya llevado el 20 (l2.codigo = l.codigo)

!==========

!CICLISMO #41. Obtener el dorsal y el nombre de los ciclistas que han llevado al menos un maillot de los que ha llevado el ciclista de dorsal 20.

!==========

```
SELECT DISTINCT c.dorsal, c.nombre
FROM ciclista c, llevar l, llevar l2
WHERE c.dorsal<>20 AND l.dorsal=c.dorsal AND l2.dorsal=20 AND l2.codigo=l.codigo;

SELECT DISTINCT c.dorsal, c.nombre
FROM ciclista c, llevar l
WHERE c.dorsal<>20 AND l.dorsal=c.dorsal
AND l.codigo IN
    (SELECT l2.codigo FROM llevar l2 WHERE l2.dorsal=20);
```

!==========

!CICLISMO #42. Obtener el dorsal y el nombre de los ciclistas que no han llevado ningún maillot de los que ha llevado el ciclista de dorsal 20.

!==========

```
SELECT Dorsal, Nombre
FROM ciclista WHERE Dorsal NOT IN (SELECT DISTINCT c.dorsal
                                   FROM ciclista c, llevar l, llevar l2
                                   WHERE c.dorsal<>20 AND l.dorsal=c.dorsal
                                   AND l2.dorsal=20 AND l2.codigo=l.codigo);

SELECT DISTINCT Dorsal, Nombre
FROM ciclista WHERE NOT EXISTS (SELECT DISTINCT c.dorsal
                                FROM ciclista c, llevar l, llevar l2
                                WHERE c.dorsal<>20 AND l.dorsal=c.dorsal
                                AND l2.dorsal=20 AND l2.codigo=l.codigo
                                AND ciclista.dorsal=c.dorsal);
```

!==========

!CICLISMO #45. Obtener el dorsal y el nombre del ciclista que ha llevado durante más kilómetros un mismo maillot e indicar también el color de dicho maillot.

!==========

```
   DORSAL     NOMBRE                           COLOR
---------- -------------------------------- -------
        20 Alfonso Gutiérrez                Verde
1 fila seleccionada.
```

¿Qué DATOS necesitamos?

1. **Necesitamos el nombre del ciclista y el color del maillot por lo que, necesariamente, tendremos que leer de las tablas CICLISTA y MAILLOT y, para saber qué ciclista llevó cada maillot necesitamos la tabla LLEVAR.**

Paso 1: Construimos la SELECT principal que nos servirá como punto de partida.

Podemos usar solo alias para ir más rápido. También ayuda hacer un boceto de qué devolverá:

```
SELECT C.dorsal, C.nombre, M.color
FROM C, LL, M
WHERE C=LL AND LL=M
```

Dorsal	Nombre	Color
25	Pedro García	Verde
31	Ana López	Amarillo
46	Juan García	Verde

Paso 2: Nos piden la suma de los kilómetros en los que ha llevado cada maillot.

Esto requiere añadir la tabla ETAPA en el FROM, añadir una función de agregado (SUM) y agrupar como mínimo por TODOS los campos del SELECT. Además, es conveniente incluir todas las claves primarias de las tablas indicadas en la SELECT por si hubieran varias filas con esos campos repetidos (dos ciclistas llamados "Pedro García"... por ejemplo... o dos maillots con el mismo color "Verde" y diferente código).

Podemos usar solo alias para ir más rápido. También ayuda hacer un boceto de qué devolverá:

```
SELECT C.dorsal, C.nombre, M.color, SUM(E.km)
FROM C, LL, M, E
WHERE C=LL AND LL=M AND M=E
GROUP BY C.dorsal, C.nombre, M.codigo, M.color
```

Dorsal	Nombre	Color	SUM(km)
25	Pedro García	Verde	20+30=50
31	Ana López	Amarillo	10+50=60
46	**Juan García**	**Verde**	**30+40=70**

Paso 3: Ya podemos ver el resultado que queremos obtener, solo nos queda dar un paso más para obtener el resultado con un mayor mayor que todos en la columna agregada.

Sería muy tentador hacer algo como MAX(SUM(km), pero concatenar funciones de agregado no está permitido en casi ningún SGBD.

En su lugar, tenemos dos opciones:

A) La más rápida y sencilla: usando LIMIT.

El LIMIT no es estándar y solo está disponible en MySQL. El resto de SGBD hay alternativas similares como TOP que se coloca junto al SELECT en POSTGRES.

Podemos usar solo alias para ir más rápido. También ayuda hacer un boceto de qué devolverá:

```
SELECT C.dorsal, C.nombre, M.color, SUM(E.km)
FROM C, LL, M, E
WHERE C=LL AND LL=M AND M=E
GROUP BY  C.dorsal, C.nombre, M.codigo, M.color
ORDER BY SUM(E.km) DESC
LIMIT 1;
```

Dorsal	Nombre	Color	SUM(km)
46	**Juan García**	**Verde**	**30+40=70**

B) La más compleja pero estándar (vale en cualquier SGDB): anidar esta consulta en otra consulta

Obtendremos la fila del resultado anterior que sea >= que TODOS esos mismos resultados.

La subconsulta solo puede devolver un valor para poder usar el cuantificador universal ALL. En este caso, eliminamos los campos del SELECT no agregados y mantenemos el GROUP BY y el FROM para que se pueda calcular la suma de manera esperada.

Algo como:

```
SELECT A,B,C, SUM(W)
FROM/WHERE <varias tablas>
GROUP BY A,B,C, D
HAVING SUM(W) >= ALL (
        SELECT A,B,C, SUM(W)
        FROM <varias tablas>
        GROUP BY A,B,C, D
)
```

```
SELECT C.dorsal, C.nombre, M.color, SUM(E.km)
FROM C, LL, M, E WHERE C=LL AND LL=M AND M=E
GROUP BY  C.dorsal, C.nombre, M.codigo, M.color
HAVING SUM(E2.km) >= ALL (
        SELECT A,B,C, SUM(E2.km)
        FROM <mismas tablas con otros alias>
        GROUP BY <mismos campos>
)
```

Dorsal	Nombre	Color	SUM(km)
46	**Juan García**	**Verde**	**30+40=70**

Paso 3: Solo nos queda adaptar la salida a lo que nos piden, que incluye únicamente el dorsal, el nombre y el color.

A) La más rápida y sencilla: usando LIMIT.

Tenemos que pasar de lo de la izquierda a lo de la derecha.

```
SELECT C.dorsal, C.nombre, M.color, SUM(E.km)
FROM C, LL, M, E
WHERE C=LL AND LL=M AND M=E
GROUP BY  C.dorsal, C.nombre, M.codigo, M.color
ORDER BY SUM(E.km) DESC
LIMIT 1;
```

Dorsal	Nombre	Color	SUM(km)
46	**Juan García**	**Verde**	**30+40=70**

```
SELECT C.dorsal, C.nombre, M.color, SUM(E.km)
FROM C, LL, M, E
WHERE C=LL AND LL=M AND M=E
GROUP BY  C.dorsal, C.nombre, M.codigo, M.color
ORDER BY SUM(E.km) DESC
LIMIT 1;
```

Dorsal	Nombre	Color	~~SUM(km)~~
46	**Juan García**	**Verde**	~~**30+40=70**~~

B) La más compleja pero estándar (vale en cualquier SGDB): anidar esta consulta en otra consulta

Tenemos que pasar de lo de la izquierda a lo de la derecha.

Usamos solo los alias para simplificar.

```
SELECT A,B,C, SUM(W)
FROM <varias tablas>
GROUP BY A,B,C, D
HAVING SUM(W) >= ALL (
        SELECT SUM(W)
        FROM <varias tablas>
        GROUP BY A,B,C, D
)
```

Dorsal	Nombre	Color	SUM(km)
46	**Juan García**	**Verde**	**30+40=70**

```
SELECT A,B,C, SUM(W)
FROM <varias tablas>
GROUP BY A,B,C, D
HAVING SUM(W) >= ALL (
        SELECT SUM(W)
        FROM <varias tablas>
        GROUP BY A,B,C, D
)
```

Dorsal	Nombre	Color	~~SUM(km)~~
46	**Juan García**	**Verde**	~~**30+40=70**~~

Paso 4: Eliminamos las tablas innecesarias.

A) La más rápida y sencilla: usando LIMIT.

```
SELECT C.nombre, C.dorsal, M.color
FROM ciclista C, maillot M, etapa E, llevar LL
WHERE C.dorsal=LL.dorsal AND LL.codigo=M.codigo AND LL.netapa=E.netapa
GROUP BY C.nombre, C.dorsal, M.color
ORDER BY SUM(E.km) DESC
LIMIT 1;
```

B) La más compleja pero estándar (vale en cualquier SGDB): anidar esta consulta en otra consulta

```
SELECT C.dorsal, C.nombre, M.color
FROM ciclista C, llevar LL, etapa E, maillot M
WHERE E.netapa = LL.netapa AND C.dorsal = LL.dorsal AND M.codigo=LL.codigo
GROUP BY C.dorsal, C.nombre, M.codigo, M.color
HAVING SUM(E.km) >= ALL
        (SELECT SUM(E2.km)
        FROM ~~ciclista C~~, llevar LL2, etapa E2, ~~maillot M~~
        WHERE E2.netapa = LL2.netapa
        GROUP BY LL2.dorsal, LL2.codigo);
```

```
SELECT C.dorsal, C.nombre, M.color
FROM ciclista C, llevar LL, etapa E, maillot M
WHERE E.netapa = LL.netapa AND C.dorsal = LL.dorsal AND M.codigo=LL.codigo
GROUP BY C.dorsal, C.nombre, M.codigo, M.color
HAVING SUM(E.km) >= ALL
        (SELECT SUM(E2.km)
        FROM llevar LL2, etapa E2
        WHERE E2.netapa = LL2.netapa
        GROUP BY LL2.dorsal, LL2.codigo);
```

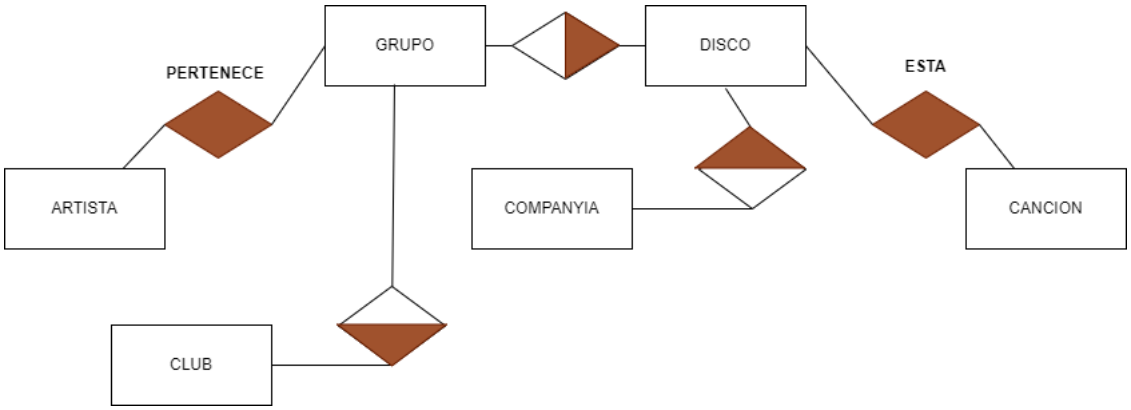
```
!============================
!CICLISMO #47 Obtener el valor del atributo netapa y los km de las etapas que tienen puertos de montaña
!============================
```

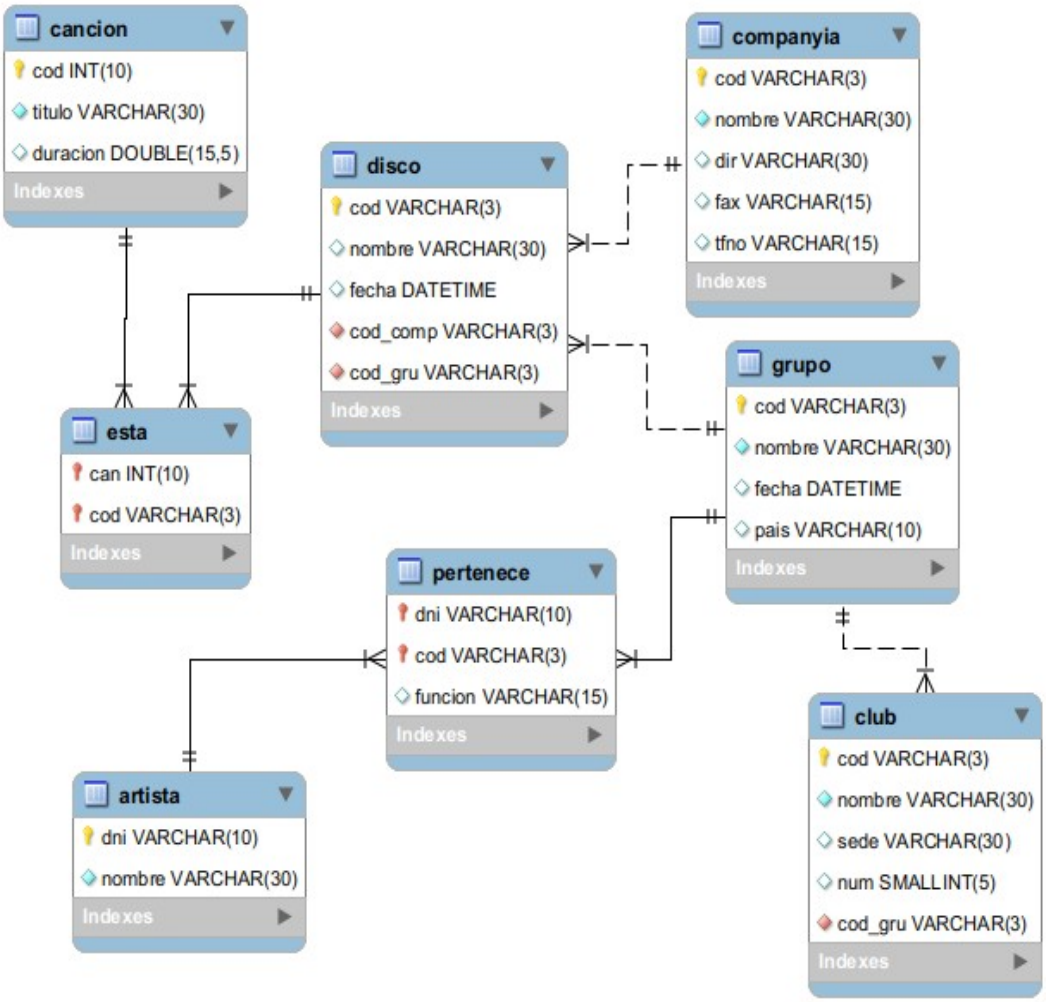
Hay muchas maneras de obtener el mismo resultado.

```
SELECT e.netapa, e.km
FROM etapa e, puerto p
WHERE e.netapa=p.netapa
GROUP BY e.netapa, e.km;

SELECT DISTINCT e.netapa, km
FROM etapa e, puerto p
WHERE e.netapa=p.netapa;

SELECT e.netapa, e.km
FROM etapa e
WHERE e.netapa in (SELECT p.netapa FROM puerto p);

SELECT E.netapa, E.km
FROM etapa E
WHERE EXISTS
    (SELECT P.nompuerto
    FROM puerto P
    WHERE P.netapa=E.netapa);
```

Importante: Recuerda que el DISTINCT, al igual que el *, pueden ser la mejor opción en muchos casos, pero no los uses a la ligera, ya que suelen ser síntoma de MALA PRAXIS.

2. SOLUCIONES MÚSICA





```
!================================================
!CONSULTAS SOBRE UNA SOLA RELACIÓN
!================================================

!============================
!MÚSICA #1. ¿Cuántos discos hay?
!============================
SELECT COUNT(*) FROM disco ;

!============================
!MÚSICA #2. Selecciona el nombre de los grupos que no sean de España.
!============================
SELECT nombre FROM grupo
WHERE pais <> 'España' ;

!============================
!MÚSICA #3. Obtener el título de las canciones con más de 5 minutos de duración.
!============================
SELECT titulo FROM cancion
WHERE duracion > 5.0 ;

!============================
!MÚSICA #4. Según los datos en la base de datos, obtener la lista de las distintas funciones que se pueden realizar en un grupo.
!============================
SELECT DISTINCT funcion FROM pertenece ;

!============================
!MÚSICA #5. Selecciona el nombre y la sede de los clubes de fans con más de 500 socios.
!============================
SELECT nombre,sede FROM club
WHERE num>500 ;
```

!==========

!Consultas con varias tablas

!==========

!==========

!MÚSICA #6. Obtener el nombre y la sede de cada club de fans de grupos de España así como el nombre del grupo al que admiran.

!==========

```
SELECT C.nombre, C.sede, G.nombre FROM club C, grupo G
WHERE C.cod_gru = G.cod AND G.pais = 'España' ;
```

!==========

!MÚSICA #7. Obtener el nombre de los artistas que pertenezcan a un grupo de España.

!==========

```
SELECT DISTINCT A.nombre
FROM artista A, pertenece P, grupo G
WHERE P.dni = A.dni AND P.cod = G.cod AND G.pais = 'España';
```

!==========

!MÚSICA #8. Obtener el nombre de los discos que contienen alguna canción que dure más de 5 minutos.

!==========

```
SELECT DISTINCT D.nombre
FROM disco D, cancion C, esta E
WHERE E.can = C.cod AND E.cod = D.cod AND C.duracion > 5.00;

SELECT D.nombre FROM disco D
WHERE D.cod in (SELECT e.cod
FROM cancion C, esta E
WHERE C.duracion > 5.00 AND E.can = C.cod) ;
```

!==========

!MÚSICA #9. Obtener los nombres de las canciones que dan nombre al disco en el que aparecen.

!==========

```
SELECT DISTINCT C.titulo
FROM disco D, cancion C, esta E
WHERE D.nombre = C.titulo AND E.can = C.cod AND E.cod = D.cod ;
```

!============================

!MÚSICA #10. Obtener los nombres de compañías y direcciones postales de aquellas compañías que han grabado algún disco que empiece por 'A'.

!============================

```
SELECT DISTINCT C.nombre, C.dir FROM disco D, companyia C
WHERE D.cod_comp = C.cod AND D.nombre LIKE 'A%';
```

!==

!Consultas con subconsultas

!==

!============================

!MÚSICA #11 . Obtener el nombre de los discos del grupo más viejo.

!============================

```
SELECT d.nombre FROM disco d, grupo g
WHERE g.cod=d.cod_gru AND g.fecha=
      ( SELECT MIN(fecha) FROM grupo ) ;
```

!============================

!MÚSICA #12. Obtener el nombre de los discos grabados por grupos con club de fans con más de 5000 personas.

!============================

```
SELECT d.nombre FROM disco d, grupo g WHERE g.cod=d.cod_gru AND g.cod IN
( SELECT f.cod_gru FROM club f WHERE f.NUM>5000) ;

SELECT d.nombre FROM disco d WHERE d.cod_gru IN
( SELECT f.cod_gru FROM club f WHERE f.NUM>5000) ;

SELECT DISTINCT(d.nombre)
FROM disco d, club cl
WHERE d.cod_grup = cl.cod_gru AND cl.NUM > 5000 ;

SELECT DISTINCT(d.nombre)
FROM disco d, grupo g, club cl
WHERE d.cod_grup = g.cod AND g.cod = cl.cod_gru AND cl.NUM > 5000 ;
```

!============================

!MÚSICA #13. Obtener el nombre de los clubes con mayor número de fans indicando ese número.

!============================

```
SELECT nombre, num FROM club
WHERE num =
    (SELECT MAX(C.NUM) FROM club C) ;
```

!============================

!MÚSICA #14. Obtener el título de las canciones de mayor duración indicando la duración.

!============================

```
SELECT C.titulo, C.duracion FROM cancion C
WHERE C.duracion =
    (SELECT MAX(D.duracion) FROM cancion D) ;
```

!==

! Consultas con cuantificador universal

!==

!============================

!MÚSICA #15. Obtener el nombre de las compañías discográficas que no han trabajado con grupos españoles.

!============================

```
SELECT C.nombre FROM companyia C
WHERE NOT EXISTS
    (SELECT * FROM disco D, grupo G
    WHERE D.cod_gru = G.cod AND G.pais = 'España' AND C.cod = D.cod_comp) ;
```

!==========

!MÚSICA #16. Obtener el nombre de las compañías discográficas que sólo han trabajado con grupos españoles.

!==========

```
SELECT C.nombre FROM companyia C WHERE
NOT EXISTS
    (SELECT *
    FROM disco D, grupo G
    WHERE D.cod_gru = G.cod AND G.pais <> 'España' AND C.cod = D.cod_comp)
    AND EXISTS
        (SELECT *
        FROM disco D, grupo G
        WHERE D.cod_gru = G.cod AND G.pais = 'España' AND C.cod = D.cod_comp) ;

SELECT DISTINCT C.nombre
FROM companyia C, disco D WHERE
c.cod = d.cod_comp AND NOT EXISTS
    (SELECT *
    FROM disco D2, grupo G
    WHERE D2.cod_gru = G.cod AND G.pais <> 'España' AND C.cod = D2.cod_comp)
```

!==========

!MÚSICA #17. Obtener el nombre y la dirección de aquellas compañías discográficas que han grabado todos los discos de algún grupo.

!==========

```
SELECT DISTINCT C.nombre, C.dir
FROM companyia C, disco D
WHERE D.cod_comp = C.cod AND NOT EXISTS
    (SELECT * FROM disco E
    WHERE E.cod_gru = D.cod_gru AND E.cod_comp <>
    D.cod_comp);
```

!==========
! Consultas agrupadas
!==========

!==========
!MÚSICA #18. Obtener el nombre de los grupos que sean de España y la suma de sus fans.
!==========

```
SELECT g.nombre, SUM(cl.NUM)
FROM grupo g, club cl
WHERE cl.cod_gru=g.cod AND g.pais like 'España'
GROUP BY g.cod, g.nombre ;

SELECT g.nombre, SUM(cl.NUM)
FROM grupo g, club cl
WHERE cl.cod_gru=g.cod AND g.pais = 'España'
GROUP BY g.cod, g.nombre ;
```

!==========
!MÚSICA #19 Obtener para cada grupo con más de dos componentes el nombre y el número de componentes del grupo.
!==========

```
SELECT G.nombre, COUNT(P.dni)
FROM grupo G, pertenece P
WHERE P.cod = G.cod
GROUP BY G.cod ,G.nombre
HAVING COUNT(P.dni) > 2 ;

SELECT G.nombre, COUNT(*)
FROM grupo G, pertenece P
WHERE P.cod = G.cod
GROUP BY G.cod ,G.nombre
HAVING COUNT(*) > 2 ;
```

```
!============================
!MÚSICA #20 Obtener el número de discos de cada grupo.
!============================
SELECT g.nombre, COUNT(d.cod)
FROM disco d, grupo g
WHERE g.cod=d.cod_gru
GROUP BY g.cod, g.nombre ;

Esta no mostraría los grupos que tienen 0 discos.
SELECT g.nombre, COUNT(d.cod)
FROM grupo g LEFT JOIN disco d ON g.cod=d.cod_gru
GROUP BY g.cod, g.nombre ;

!================================================
! Otras Consultas (union, join)
!================================================

!============================
!MÚSICA #21. Obtener el número de canciones que ha grabado cada compañía discográfica y su dirección.
!============================
SELECT C.nombre, COUNT (DISTINCT E.can), C.dir
FROM companyia C, disco D, esta E
WHERE C.cod = D.cod_comp AND E.cod = D.cod
GROUP BY C.cod, C.nombre, C.dir
UNION
    SELECT C.nombre, 0, C.dir FROM companyia C
    WHERE NOT EXISTS
        (SELECT E.can FROM disco D, esta E
        WHERE C.cod = D.cod_comp AND E.cod = D.cod) ;

SELECT C.nombre, COUNT (DISTINCT E.can), C.dir
FROM companyia C LEFT JOIN disco D ON C.cod = D.cod_comp
LEFT JOIN esta E ON E.cod = D.cod
GROUP BY C.cod, C.nombre, C.dir
```

!==========

!Consultas generales

!==========

!==========

!MÚSICA #22 Obtener los nombre de los artistas de grupos con clubes de fans de más de 500 personas y que el grupo sea de Inglaterra

!==========

```
SELECT DISTINCT a.nombre
FROM artista a, grupo g, club cl, pertenece p
WHERE g.pais like 'Inglaterra' AND p.cod=g.cod AND a.dni=p.dni AND cl.cod_gru=g.cod AND cl.NUM >500 ;
```

!==========

!MÚSICA #23 Obtener el título de las canciones de todos los discos del grupo U2.

!==========

```
SELECT DISTINCT c.titulo
FROM disco d, cancion c, grupo g, esta e
WHERE c.cod=e.can AND e.cod=d.cod AND g.cod=d.cod_gru AND g.nombre like 'U2' ;
```

!==========

!MÚSICA #25 Obtener el nombre de los artistas que pertenecen a más de un grupo.

!==========

Solución algo enrevesada, pero válida consistente en usar una tabla de manera reflexiva:

```
SELECT DISTINCT A.nombre
FROM artista A, pertenece P, pertenece Q
WHERE P.dni = A.dni AND Q.dni = A.dni
AND P.cod <> Q.cod;
```

Otra alternativa más sencilla:

```
SELECT A.nombre
FROM artista A
WHERE 1<
        (SELECT COUNT(*)
        FROM pertenece P
        WHERE P.dni = A.dni);
```

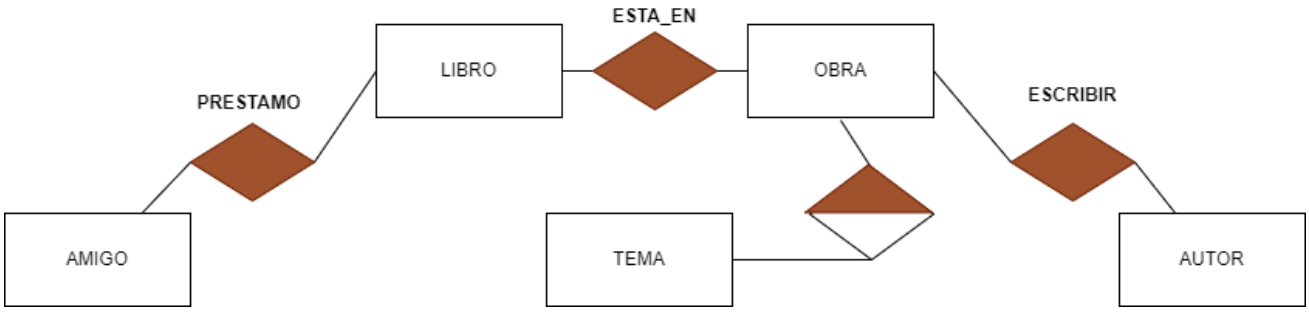
Si usamos agrupaciones: **¡CUIDADO! PUEDE HABER VARIOS ARTISTAS CON EL MISMO NOMBRE POR LO QUE DEBO AGRUPAR POR LA CLAVE PRIMARIA**

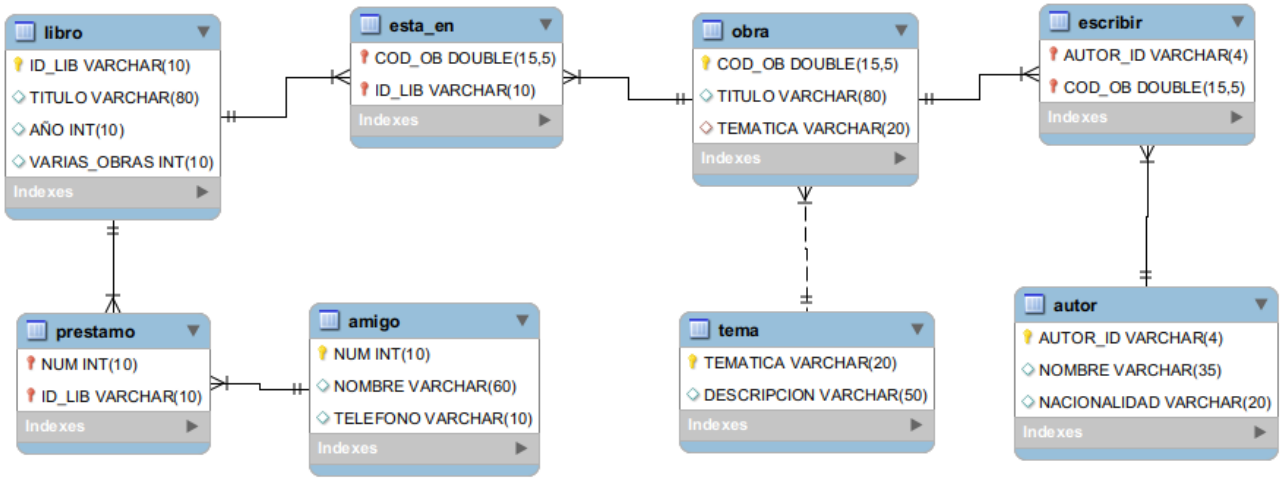
```
SELECT A.nombre
FROM artista A, pertenece P
WHERE P.dni = A.dni
GROUP BY A.dni, A.nombre
HAVING COUNT(P.cod)>1;
```

```
SELECT A.nombre
FROM artista A INNER JOIN pertenece P ON P.dni = A.dni
GROUP BY A.dni, A.nombre
HAVING COUNT(P.cod)>1;
```

Importante: Recuerda que todos los campos no agregados que aparecen en el SELECT deben estar en el GROUP BY y que, es recomendable incluir siempre alguna PK en el GROUP BY

3. SOLUCIONES BIBLIOTECA





```
!============================
!BIBLIOTECA #1 ¿Cuántos libros hay de los que se conozca el año de adquisición?
!============================
SELECT COUNT(*) as lib_año FROM libro
WHERE año is NOT NULL;

SELECT COUNT(año) as lib_año FROM libro;

!============================
```

!BIBLIOTECA #2 ¿Cuántos libros tienen más de una obra? Resolver este ejercicio utilizando el atributo num_obras y sin utilizarlo.

```
!============================
SELECT COUNT(*) as Más_1_ob FROM libro
WHERE num_obras >1;

SELECT COUNT(DISTINCT e.ID_LIB) FROM esta_en E, esta_en E2
WHERE E.ID_LIB=E2.ID_LIB AND E.COD_OB <> E2.COD_OB;

SELECT COUNT(*)
FROM (SELECT COUNT(*)
FROM esta_en GROUP BY ID_LIB
HAVING COUNT(COD_OB)>1);

SELECT COUNT(*) FROM libro l
WHERE (SELECT COUNT(*)
FROM esta_en e
WHERE e.ID_LIB=l.ID_LIB)>1;

Esta está mal porque para cada lib da un cont
SELECT COUNT(*)
FROM esta_en GROUP BY ID_LIB HAVING COUNT(*)>1;
```

```
!============================
!BIBLIOTECA #3 ¿Cuántos autores hay en la base de datos de los que no se tiene ninguna obra?
!============================
SELECT COUNT(*) Sin_obra FROM autor
WHERE AUTOR_ID NOT IN (SELECT AUTOR_ID
FROM escribir);

SELECT COUNT(*) Sin_obra
FROM autor a LEFT JOIN escribir e on a.AUTOR_ID=e.author_id WHERE e.AUTOR_ID is NULL;

!============================
!BIBLIOTECA #4 Obtener el nombre de esos autores.
!============================
SELECT nombre
FROM autor
WHERE AUTOR_ID NOT IN (SELECT AUTOR_ID FROM escribir);

SELECT a.nombre FROM autor a
WHERE NOT EXISTS (SELECT *
FROM escribir e
WHERE a.AUTOR_ID= e.AUTOR_ID);

!============================
!BIBLIOTECA #5 Obtener el título de las obras escritas sólo por un autor si éste es de
nacionalidad “Francesa” indicando también el nombre del autor.
!============================
SELECT titulo, nombre FROM obra, autor
WHERE COD_OB in (SELECT COD_OB
FROM escribir GROUP BY COD_OB HAVING COUNT(*)=1)
AND (COD_OB, AUTOR_ID) in (SELECT COD_OB, AUTOR_ID FROM escribir e) AND nacionalidad
='Francesa';

SELECT titulo, nombre FROM obra o, autor a, escribir e1
WHERE o.COD_OB=e1.COD_OB AND e1.AUTOR_ID=a.AUTOR_ID AND
a.nacionalidad=”Francesa”
AND NOT EXISTS (SELECT * FROM escribir e WHERE e.COD_OB=e1.COD_OB AND
e.author_id<>e1.author_id);
```

!==========

!BIBLIOTECA #6 Obtener el título y el identificador de los libros que tengan título y más de dos obras, indicando el número de obras.

!==========

```
SELECT l.ID_LIB, l.titulo, COUNT(*) FROM libro l, esta_en e
WHERE l.titulo is NOT NULL AND e.id_Lib=l.ID_LIB GROUP BY l.ID_LIB, l.titulo
HAVING COUNT(*) >2;

SELECT ID_LIB, titulo, num_obras FROM libro l
WHERE l.titulo is NOT NULL AND num_obras >2;
```

!==========

!BIBLIOTECA #7 Obtener el nombre de los autores de nacionalidad “Española” que han escrito dos o más obras.

!==========

```
SELECT nombre
FROM autor a
WHERE nacionalidad ='Española' AND
2<= (SELECT COUNT(*) FROM escribir e WHERE a.AUTOR_ID= e.AUTOR_ID);

SELECT nombre FROM autor a
WHERE a.nacionalidad ='Española' AND
(SELECT COUNT(*) FROM escribir e
WHERE a.AUTOR_ID= e.AUTOR_ID) >= 2;
```

!==========

!BIBLIOTECA #8 Obtener el nombre de los autores de nacionalidad “Española” que tienen obras en dos o más libros.

!==========

```
SELECT nombre
FROM autor a
WHERE nacionalidad ='Española' AND
2<= (SELECT COUNT(DISTINCT ID_LIB) FROM escribir e, esta_en ee WHERE a.AUTOR_ID= e.AUTOR_ID AND e.COD_OB=ee.COD_OB);

SELECT nombre FROM autor a
WHERE nacionalidad ='Española' AND
(SELECT COUNT(DISTINCT ID_LIB) FROM escribir e, esta_en ee
WHERE a.AUTOR_ID= e.AUTOR_ID AND e.COD_OB=ee.COD_OB) >=2;

SELECT nombre FROM autor a
WHERE nacionalidad ='Española' AND
2<= (SELECT COUNT(*) FROM escribir e, esta_en ee
WHERE a.AUTOR_ID= e.AUTOR_ID AND e.COD_OB=ee.COD_OB GROUP BY ee.ID_LIB);
```

!BIBLIOTECA #9 Obtener el título y el código de las obras que tengan más de un autor.

```
SELECT COD_OB, titulo FROM obra o
WHERE 1 < (SELECT COUNT(*) FROM escribir e WHERE o.COD_OB=e.COD_OB);

SELECT COD_OB, titulo FROM obra o
WHERE (SELECT COUNT(*) FROM escribir e WHERE o.COD_OB=e.COD_OB) > 1;

SELECT COD_OB, titulo FROM obra o, escribir e WHERE o.COD_OB=e.COD_OB GROUP BY COD_OB, titulo HAVING COUNT(*)>1
```

!==========

!BIBLIOTECA #10 Obtener el título y el identificador de los libros que tengan título y que contengan sólo una obra.

!==========

```
SELECT l.ID_LIB, l.titulo FROM libro l
WHERE titulo is NOT NULL AND
1= (SELECT COUNT(*) FROM esta_en e WHERE l.ID_LIB=e.ID_LIB);

SELECT titulo FROM libro l
WHERE titulo is NOT NULL AND
(SELECT COUNT(*) FROM esta_en e WHERE l.ID_LIB=e.ID_LIB) = 1;
```

!==========

!BIBLIOTECA #11 Como se concluye del resultado de la consulta anterior, los libros con una sola obra no tienen título propio. Asumiendo en este caso que su título es el de la obra que contienen, obtener la lista de todos los títulos de libros que hay en la base de datos tengan las obras que tengan título.

!==========

```
SELECT titulo FROM libro
WHERE titulo is NOT NULL
UNION
SELECT titulo
FROM obra o, esta_en e WHERE o.COD_OB =e.COD_OB
AND 1 = (SELECT COUNT(*) FROM esta_en e1
WHERE e.ID_LIB=e1.ID_LIB);

SELECT titulo as titulo FROM libro
WHERE titulo is NOT NULL
UNION
SELECT o.titulo as titulo
FROM obra o, esta_en e, libro l
WHERE o.COD_OB =e.COD_OB AND l.ID_LIB=e.ID_LIB AND l.titulo is NULL
```

```
!============================
!BIBLIOTECA #12 Obtener el nombre del autor (o autores) que más obras han escrito?
!============================
SELECT nombre
FROM autor a, escribir e WHERE a.AUTOR_ID=e.AUTOR_ID GROUP BY a.AUTOR_ID, nombre
HAVING COUNT(*) >= ALL (SELECT COUNT(*) FROM escribir GROUP BY AUTOR_ID);

!============================
!BIBLIOTECA #13 Obtener la nacionalidad (o nacionalidades) menos frecuentes.
!============================
SELECT nacionalidad
FROM autor
GROUP BY nacionalidad
HAVING COUNT(*) <= ALL (SELECT COUNT(*) FROM autor GROUP BY nacionalidad);
```