

---

# **RESOLUCIÓN DE CONSULTAS SQL**

---

PENDIENTE DE REVISIÓN

Luis Egui

**Contenidos**

---

	<b>Página</b>
<b>Problema 1</b>	<b>1</b>
<b>Problema 2</b>	<b>4</b>

### Estilo del código SQL usado

Para el código SQL me gusta que este sea lo más legible posible, y, dado que SQL es un lenguaje declarativo de cuarta generación (4GL)<sup>1</sup>, resulta más cómodo leerlo tal y como se lee un libro. De esta manera:

- La sintaxis SQL, independientemente del Database Management System (DBMS), esta escrita en *lowercase*.
- Los atributos indicados en el operador “*select*” están sujetos al nombre indicado en el modelo relacional dado. Los renombramientos de los mismos o de funciones de agregación están escritos en *lower\_snake\_case*.  
Solo en caso necesario, se indicará: agregando el nombre de la tabla (o su correspondiente renombramiento) delante del atributo. En mi opinión, aumenta la comprensión de los atributos que resultan cabecera de la consulta que se está haciendo.  
En caso de haber bastantes atributos, estos irán sangrados con respecto al operador “*select*”. Por norma general, no suelo superar los 80 caracteres por línea.
- Los nombres de las tablas están escritos en *CamelCase*, sin importar como hayan sido nombradas en el modelo.
- Los “*joins*”, da igual el tipo, se realizarán con sangrado respecto del operador “*from*” para así tener clara la separación de las distintas partes de la consulta y mejorar la comprensión de la misma.  
Lo mismo es aplicable para subconsultas en el operador “*from*” o en el operador “*where*”.

---

<sup>1</sup>4th generation language

**Problema 1**

Dado el modelo relacional:

*Pilotos*(iniciales, nombre, escuderia, pais)

*Circuitos*(nombre, longitud, nVueltas)

*Parrilla*(piloto, circuito, posicion)

*Tiempos*(piloto, circuito, nVuelta, tiempo, paradaBoxes)

a) Realizando la consulta con un producto cartesiano, seria:

```
1  select p1.*
2  from Pilotos as p1, Pilotos as p2
3  where p1.iniciales != p2.iniciales — distinct
4  and p1.pais = p2.pais and p1.escuderia = p2.escuderia;
```

Haciendo uso del *join*, seria:

```
1  select p1.*
2  from Pilotos p1
3      join Pilotos p2 on p1.escuderia = p2.escuderia
4  where p1.pais = p2.pais and p1.iniciales != p2.iniciales;
```

b)

```
1  select distinct escuderia
2  from Pilotos
3      inner join Parrilla on iniciales = piloto
4  where posicion in ('1','2','3');
```

c)

```
1  select circuito, max(tiempo) as tiempo_max_vuelta
2  from Tiempos
3  group by circuito;
```

d)

```
1  select circuito , piloto , count(nVuelta) as vueltas_completadas ,
2      sum(tiempo) as tiempo_total
3  from Tiempos
4  where circuito = "Monza"
5  group by circuito , piloto
6  order by vueltas_completadas desc , tiempo_total asc;
```

e) Usando el operador *limit* de MySQL o PostgreSQL:

```
1  select nombre, longitud
2  from Circuitos
3  order by longitud desc
4  limit 1;
```

Para OracleSQL 12c R1 (12.1):

```
1  select nombre, longitud
2  from Circuitos
3  order by longitud desc
4  fetch first 1 rows only;
```

f)

```
1  select piloto , circuito , sum(tiempo) as tiempo_invertido ,
2      count(nVuelta) as vueltas_totales
3  from Tiempos
4      inner join Circuitos on circuito = nombre
5  group by piloto , circuito
6  having count(vueltas_totales) = nVueltas;
```

- g) Suponemos que *Tiempos(paradaBoxes)* se trata de un *boolean* cuyo valor es *1* cuando el piloto ha realizado la parada en la vuelta indicada en el registro; y que por defecto su valor es *0*.

```
1  select circuito , avg(paradaBoxes) as media_paradas_boxes
2  from Tiempos
3  group by circuito;
```

**Problema 2**

Dado el modelo relacional:

*Persona*(dni, sexo, edad)

*Habla*(dniPersona, idioma, grado)

*SolicitaCita*(dniSolicitante, dniSolicitado, idioma)

a)

```
1  select idioma, count(*) as no_hablantes
2  from Habla
3  where grado = 3
4  group by idioma
5  order by no_hablantes desc
6  fetch first row with ties;
```

b)

```
1  select avg(edad) as promedio_edad_hispanohablantes
2  from Habla h1
3       inner join Persona on h1.dniPersona = dni
4  where idioma = 'Espanol' and not exists (
5       select idioma
6       from Habla h2
7       where h1.dniPersona = h2.dniPersona and h2.idioma != 'Espanol');
```

c) Esta consulta no nos devuelve una unica fila, sino que para cada persona; nos devuelve el numero de idiomas que habla. AL estar ordenadas de mayor numero de idiomas hablados, a menor: obtendremos en primera posicion la persona que mas idiomas habla.

```
1  select dni, sexo, edad, count(idioma) as lenguas_habladas
2  from Habla
3       inner join Persona on dniPersona = dni
4  group by dni
5  order by lenguas_habladas desc;
6  — lenguas_habladas son filas unicas ya que idioma es unique.
```

En MySQL y PostgreSQL es sencillo poder limitar el numero de filas obtenidas en la anterior consulta, de la siguiente manera:

```
1  select dni, sexo, edad, count(idioma) as lenguas_habladas
2  from Habla
3       inner join Persona on dniPersona = dni
4  group by dni
5  order by lenguas_habladas desc
6  limit 1;
```

Para Oracle SQL 12c R1 (12.1):

```
1  select sexo, edad
2  from (
3       select dni, sexo, edad, count(idioma) as lenguas_habladas
4       from Habla
5            inner join Persona on dniPersona = dni
6       group by dni, sexo, edad
7       order by lenguas_habladas desc
8       fetch first row with ties
9  );
```

d) De manera trivial:

```
1  select p_req.dni, count(*) as num_citas
2  from SolicitaCita
3       inner join Persona p on dniSolicitante = p.dni
4       inner join Persona p_req on dniSolicitado = p_req.dni
5  where p.sexo != p_req.sexo
6  group by p_req.dni
7  having count(*) = (
8       select max(num_citas)
9       from (
10            select p2.dni, count(*) as num_citas
11            from SolicitaCita
12                 inner join Persona p1 on dniSolicitante = p1.dni
13                 inner join Persona p2 on dniSolicitado = p2.dni
14                 where p1.sexo != p2.sexo
15                 group by p2.dni
16            )
17       );
```



Para Oracle 12c R1 (12.1):

```
1  select edad, sexo
2  from (
3      select p2.dni, p2.edad, p2.sexo, count(*) as num_citas
4      from SolicitaCita
5           inner join Persona p1 on dniSolicitante = p1.dni
6           inner join Persona p2 on dniSolicitado = p2.dni
7      where p1.sexo != p2.sexo
8      group by p2.dni, p2.edad, p2.sexo
9      order by num_citas desc
10     fetch first row with ties
11 );
```

- e) Recordamos que piden los idiomas hablados por el solicitante/solicitado pero que no han sido pedidos en la cita. Por lo que la consulta seria:

```
1  select distinct h1.idioma — idiomas que habla el solicitante
2  from SolicitaCita
3       inner join Habla h1 on dniSolicitante = dniPersona
4  union — une filas de manera vertical, sin duplicarlas.
5  select distinct h2.idioma — idiomas que habla el solicitado
6  from SolicitaCita
7       inner join Habla h2 on dniSolicitado = dniPersona
8  minus
9  select distinct idioma — idiomas pedidos en las citas
10 from SolicitaCita;
```

Tambien es posible realizar la consulta haciendo uso del *left join* sobre la tabla *SolicitaCita*. En cuanto a cual seria la consulta mas optima, seria necesario realizar una prueba dado que, aunque la operacion *minus* es pobre en optimizacion, esta se realiza sobre valores ya filtrados anteriormente.

f)

```
1  select distinct h1.dniPersona, h1.grado
2  from SolicitaCita
3       inner join Habla h1 on dniSolicitante = h1.dniPersona
4       inner join Habla h2 on dniSolicitado = h2.dniPersona
5       inner join Persona on dniSolicitado = dni
6  where sexo = 'M' and edad < 25
7       and h1.idioma = 'Ingles' and h2.idioma = 'Ingles';
```

g)

```
1  select dni_no_solicitado, sexo, edad,
2         count(idioma) as num_idiomas_hablados
3  from (
4         select dni as dni_no_solicitado
5         from Persona
6         minus — personas nunca solicitadas
7         select distinct dniSolicitado
8         from SolicitaCita
9         )
10 — puede que exista la persona pero no hable ningun idioma
11 left join Habla on dni_no_solicitado = dniPersona
12 inner join Persona p on dni_no_solicitado = p.dni
13 group by dni_no_solicitado, sexo, edad
14 order by num_idiomas_hablados desc;
```