RESOLUCIÓN DE CONSULTAS SQL

PENDIENTE DE REVISIÓN

Luis Egui

Contenidos

Estilo del codigo SQL usado

Para el codigo SQL me gusta que este sea lo mas legible posible, y, dado que SQL es un lenguaje declarativo de cuarta generacion (4GL)[1], resulta mas comodo 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"* estan sujetos al nombre indicado en el modelo relacional dado. Los renombramientos de los mismos o de funciones de agregacion estan escritos en *lower_snake_case*.
  Solo en caso necesario, se indicara: agregando el nombre de la tabla (o su correspondiente renombramiento) delante del atributo. En mi opinion, aumenta la comprension de los atributos que resultan cabecera de la consulta que se esta haciendo.
  En caso de haber bastantes atributos, estos iran sangrados con respecto al operador *"select"*. Por norma general, no suelo superar los 80 caracteres por linea.
- Los nombres de las tablas estan escritos en *CamelCase*, sin importar como hayan sido nombradas en el modelo.
- Los *"joins"*, da igual el tipo, se realizaran con sangrado respecto del operador *"from"* para asi tener clara la separacion de las distintas partes de la consulta y mejorar la comprension de la misma.
  Lo mismo es aplicable para subconsultas en el operador *"from"* o en el operador *"where"*.

[1] 4th generation language

Problema 1

Dado el modelo relacional:

$$Pilotos(\underline{iniciales}, nombre, escuderia, pais)$$
$$Circuitos(\underline{nombre}, longitud, nVueltas)$$
$$Parrilla(\underline{piloto}, \underline{circuito}, posicion)$$
$$Tiempos(\underline{piloto}, \underline{circuito}, \underline{nVuelta}, tiempo, paradaBoxes)$$

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) Haciendo uso de un SQL mas estandar:

```
1  select nombre, longitud
2  from Circuitos
3  where longitud = (
4          select max(longitud)
5          from Circuitos
6          );
```

Para OracleSQL 12c R1 (12.1):

```
1  select nombre, longitud
2  from Circuitos
3  order by longitud desc
4  fetch first row with ties;
```

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:

$$Persona(\underline{dni}, sexo, edad)$$
$$Habla(\underline{dniPersona}, \underline{idioma}, grado)$$
$$SolicitaCita(\underline{dniSolicitante}, \underline{dniSolicitado}, idioma)$$

a)

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

b)

```sql
select avg(edad) as promedio_edad_hispanohablantes
from Habla h1
     inner join Persona on h1.dniPersona = dni
where idioma = 'Espanol' and not exists (
         select idioma
         from Habla h2
         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.

```sql
select dni, sexo, edad, count(idioma) as lenguas_habladas
from Habla
     inner join Persona on dniPersona = dni
group by dni
order by lenguas_habladas desc;
-- 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):

```
select edad, sexo
from (
        select p2.dni, p2.edad, p2.sexo, count(*) as num_citas
        from SolicitaCita
             inner join Persona p1 on dniSolicitante = p1.dni
             inner join Persona p2 on dniSolicitado = p2.dni
        where p1.sexo != p2.sexo
        group by p2.dni, p2.edad, p2.sexo
        order by num_citas desc
        fetch first row with ties
      );
```

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:

```
select distinct h1.idioma -- idiomas que habla el solicitante
from SolicitaCita
     inner join Habla h1 on dniSolicitante = dniPersona
union -- une filas de manera vertical, sin duplicarlas.
select distinct h2.idioma -- idiomas que habla el solicitado
from SolicitaCita
     inner join Habla h2 on dniSolicitado = dniPersona
minus
select distinct idioma -- idiomas pedidos en las citas
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)

```
select distinct h1.dniPersona, h1.grado
from SolicitaCita
     inner join Habla h1 on dniSolicitante = h1.dniPersona
     inner join Habla h2 on dniSolicitado = h2.dniPersona
     inner join Persona on dniSolicitado = dni
where sexo = 'M' and edad < 25
      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;
```

Problema 3

Dado el modelo relacional:

$$Montura(\underline{codigoMontura}, nombre, sexo, edad)$$
$$Monta(\underline{dni}, \underline{codigoMontura}, \underline{idCompeticion}, clasificacion)$$
$$Jinete(\underline{dni}, nombre, apellidos, categoria, sexo, cache)$$
$$Competicion(\underline{id}, nombre, fecha, lugar)$$

a)

```sql
select count(*) as num_competiciones
from Competicion
where fecha = 2014 and lugar = 'Madrid';
```

b)

```sql
select sexo, categoria, avg(cache) as cache_medio
from Jinete
where sexo = 'Mujer'
group by sexo, categoria;
```

c)

```sql
select idCompeticion, nombre, count(distinct dni) as num_participantes
from Monta
     inner join Competicion on id = idCompeticion
group by idCompeticion, nombre
order by num_participantes desc
fetch first row with ties;
```

d)

```sql
select nombre, apellidos, categoria,
       count(distinct idCompeticion) as num_participaciones
from Monta
     inner join Jinete on Monta.dni = Jinete.dni
where sexo = 'Hombre' and clasificacion > 1
group by nombre, apellidos, categoria
having count(num_participaciones) >= 100
order by num_participaciones desc;
```

e)

```
1  select nombre, apellidos, categoria, cache
2  from Jinete, (  select categoria, avg(cache) as media_cache
3                  from Jinete
4                  where sexo = 'Hombre'
5                  group by categoria
6                  ) referencia_masculina
7  where sexo = 'Mujer'
8        and Jinete.categoria = referencia_masculina.categoria
9        and cache > referencia_masculina.media_cache;
```

f)

```
1  select avg(cache)
2  from (
3      select distinct dni, cache
4      from Monta
5           inner join Jinete on Monta.dni = Jinete.dni
6      where clasificacion in ('1','2','3')
7  );
```

Problema 4

Dado el modelo relacional:

$$Competicion(\underline{id}, nombre, fecha, lugar, ambito)$$
$$Participa(\underline{dni}, \underline{idComp}, clasificacion)$$
$$Patinador(\underline{dni}, nombre, apellidos, sexo, fechaNac, pais)$$

a)

```sql
select count(*) as num_comp
from Participa
     inner join Competicion on id = idComp
where lugar = 'Chicago'
      and (fecha >= 2010 and fecha <= 2015)
      and clasificacion is not null; -- si !null => se ha celebrado
```

b)

```sql
select nombre, apellidos
from Participa
     inner join Patinador on Patinador.dni = Participa.dni
     inner join Competicion on id = idComp
where sexo = 'M' and ambito = 2 -- Ambito: (2) Camp. Mundial
      and clasificacion = 1;
```

c)

```sql
select pais, avg(clasificacion) as media_clasif
from Participa
     inner join Patinador on Patinador.dni = Participa.dni
where clasificacion is not null -- si !null => se ha celebrado
      and sexo = 'H'
group by pais
order by media_clasif asc;
```

d) Esta consulta no deja claro si la competicion se ha celebrado o no.

```sql
select id, nombre, count(dni) as num_participantes
from Participa
     inner join Competicion on id = idComp
group by id, nombre
order by num_participantes desc
fetch first row with ties;
```

Haciendo uso de un SQL mas estandar:

```
1   select id, nombre, count(dni) as num_participantes
2   from Participa
3        inner join Competicion on id = idComp
4   group by id, nombre
5   having count(dni) = (
6           select max(num_participantes)
7           from (
8                select idComp, count(dni) as num_participantes
9                from Participa
10               group by idComp
11          )
12      );
```

e) No me queda claro que es lo que pide exactamente.