SQL

Sintaxis Explícita para el JOIN

Considere la siguiente consulta:

```
Select  *
from A, B, C
where A.id=B.id AND B.ref=C.id;
```

El Planificador – programa del SGBD que enumera todos los posibles planes para ejecutar esa consulta – tiene tres opciones para ejecutar el JOIN.

1.- $C \bowtie_{B.ref=C.id} (A \bowtie_{A.id=B.id} B)$.

2.- $A \bowtie_{A.id=B.id} (B \bowtie_{B.ref=C.id} C)$.

3.- $\sigma_{B.ref=C.id} (B \bowtie_{B.id=A.id} (A \times C))$.

Semánticamente Equivalentes: Dan el mismo Resultado

Sintaxis Explícita para el JOIN

Considere la siguiente consulta:

```
Select  *
from A, B, C
where A.id=B.id AND B.ref=C.id;
```

El Planificador – programa del SGBD que enumera todos los posibles planes para ejecutar esa consulta – tiene tres opciones para ejecutar el JOIN.

1.- $C \bowtie_{B.ref=C.id} (A \bowtie_{A.id=B.id} B)$.

2.- $A \bowtie_{A.id=B.id} (B \bowtie_{B.ref=C.id} C)$.

3.- $\sigma_{B.ref=C.id} (B \bowtie_{B.id=A.id} (A \times C))$.

Significativamente Diferentes: En el costo de Ejecución.

Sintaxis Explícita para el JOIN

Una manera de restringir al Planificador en la enumeración de los posibles planes de ejecución es usar una sintaxis explícita para el JOIN.

Entonces se definen tipos de JOIN:

.- LEFT JOIN, RIGTH JOIN.
.- CROSS JOIN, INNER JOIN o JOIN.

Impactan la Optimización de Consultas en el SGBD

EJEMPLO:

```
Select *
From A LEFT JOIN (B JOIN C ON (B.ref = C.id)) ON (A.id = B.id);
```

GRUPOS, PREDICADOS ESPECIALES Y AGREGADOS

Al SQL se le han añadido algunas funciones matemáticas que permiten hacer cálculos a partir de valores de atributos de una relación, o bien, que permitan satisfacer consultas que requieran que el resultado aparezca clasificado por grupos.

Para agrupar se utiliza:

Group by <atributo>

Predicados Especiales:

Having P

Los agregados son:

AVG : para calcular el promedio de los valores que una relación tiene para un determinado atributo.

MIN : devuelve el valor mínimo, entre las tuplas, de la relación para un determinado atributo.

MAX : devuelve el valor máximo, entre las tuplas, de la relación para un determinado atributo.

SUM : devuelve el total de la suma para un determinado atributo en una relación.

COUNT : el valor que devuelve es el número de tuplas que tiene una relación, es decir, indica cuál es la cardinalidad de la relación.

Consulta 10: Dar el saldo promedio de los depósitos hechos en la sucursal “Centro”

```
Select  AVG(saldo)
from Deposito
where nombre-suc = "Centro“;
```

Consulta 11: Dar el saldo promedio de los depósitos clasificados según cada sucursal del banco.

```
Select  nombre-suc, AVG(saldo)
from Deposito
Group by nombre-suc;
```

Consulta 12: Dar el promedio de los depósitos clasificados según cada sucursal del banco siempre y cuando el promedio sea mayor a 5.000

```
Select  nombre-suc, AVG(saldo)
from Deposito
Group by nombre-suc
 Having AVG(saldo)  > 5.000;
```

Consulta 13: ¿Cuántos cuentahabientes tiene el banco?

```
Select  Count(*)
from Cuentahabiente;
```

ELIMINACIÓN Y ACTUALIZACIÓN EN SQL

La sintaxis de las instrucciones para eliminar, insertar o actualizar una relación se consideran expresiones especiales que pueden o no complementarse con restricciones expresadas en una cláusula *where*. Por ejemplo:

- Para eliminar
  Delete from <relación>

- Para insertar
  Insert into <relación> values (<lista de valores>)

- Para actualizar
  Update <relación> Set <expresión de modificación de campo>

Consulta 14: Eliminar todas las piezas almacenadas en Barquisimeto.

```
Delete  from Pieza
where ciudad-alm = "Barquisimeto“;
```

Consulta 15: Eliminar los préstamos que tengan un importe menor a 100.000

```
Delete  from Prestamo
where importe < 100.000;
```

Consulta 16: Eliminar todos los depósitos que haya hecho el cuentahabiente "V-6.714.528" en cualquier sucursal del banco.

```
Delete  from Deposito
where ci-ch = "V-6.714.528“;
```

Consulta 16’: Eliminar todos los depósitos que haya hecho el cuentahabiente "V-6.714.528" en la sucursal “Centro”.

```
Delete  from Deposito
where numDeposito  in
                        (Select numDeposito
                         from Deposito
                         where ci-ch = "V-6.714.528" and
                                  nombre-suc=“Centro”);
```

Consulta 17: Eliminar todos los depósitos de la sucursal “Oriente” siempre y cuando el saldo sea inferior al promedio de saldo depositados en el banco.

```
Delete  from Deposito
where nombre-suc = "Oriente" and
        saldo < (Select AVG(saldo)
                 from Deposito);
```

Consulta 19: Asignar a los cuentahabientes que tienen un préstamo en la sucursal "Centro" un depósito con 200.000

```
Insert into Deposito
select numDeposito, ci-ch,nombre-suc,200.000
from Prestamo
where nombre-suc = "Centro";
```

Consulta 20: Añadir un depósito del cuentahabiente dado.

```
Insert into Deposito
values ("V-10.556.999",444987,"Oriente",1599);
```

Consulta 21: Sumar el 0.5% a cada saldo de todo depósito mayor a 10.000

```
Update Deposito
set  saldo = saldo * 1.005
where saldo > 10.000;
```

Otros conectores de interés:

CONTAINS

EXISTS

NOT EXISTS

CREATE VIEW

Fuentes consultadas:

[1] Silberchatz, Korth. ,
"Fundamentos de Bases de Datos".

[2] Prof. Elsa Liliana Tovar.
Notas de clase compiladas entre 1997-2016.

[3] http://www.1keydata.com/sql/sqlinsert.html