Auxiliar 7 – Bases de datos - Consultas SQL

Enunciado

Sean las siguientes tablas de una base de datos de una corredora de propiedades:

```
Arrendatario(RUT,Nombre,Apellido)
Arrienda(RUT,Id_casa,Deuda)     Ojo: Deuda >=0 (si es 0, no hay deuda)
Telefonos(RUT,Fono)
Dueño(RUT,Nombre,Apellido)
Casa(Id_casa,RUT,Nro,Calle,Comuna)
```

Al respecto, conteste las siguientes preguntas:

1. Los arrendatarios que arriendan la casa ubicada en la calle Carrera nº 1024, Santiago.
2. ¿Cuánto le deben a María Pérez?
3. ¿Cuál es la deuda total para cada dueño?
4. Liste todas las personas de la base de datos
5. Indique los dueños que poseen tres o más casas.
6. Liste los dueños que tengan deudores en todas sus casas.
7. Entregue **estadísticas** sobre los arrendatarios por casa. Entregue:
    1. El promedio.
    2. La varianza.
    3. El máximo.
    4. El mínimo.
    5. La moda.
    6. La mediana.

Solución

Recomendación: evite colocar *selects* en el *from*. Prefiera no anidar de esa manera; muchas veces eso se puede escribir dentro de *where*.

1. Este es el tipo más sencillo de consulta posible.

```
SELECT A.RUT, A.Nombre, A.Apellido
FROM Arrendatario A, Arrienda B, Casa C
WHERE A.RUT=B.RUT AND B.Id_casa=C.Id_casa
  AND C.Calle='Carrera' AND C.Nro='1024' AND C.Comuna='Santiago' ;
```

2. Se supondrá que María Pérez hay una sola.

```
SELECT SUM(A.Deuda)   FROM Arrienda A, Casa B, Dueño C
WHERE A.Id_casa=B.Id_casa AND B.RUT=C.RUT
  AND C.Nombre='María' AND C.Apellido='Pérez' ;
```

3. Aquí es necesario agrupar la información, así la suma se hará dentro de cada grupo indicado. Entregué toda la información necesaria en el SELECT, aunque con el RUT del dueño bastaría (si en el trabajo le piden algo así, entregue todo).

```
SELECT SUM(A.Deuda), C.RUT, C.Nombre, C.Apellido
FROM Arrienda A, Casa B, Dueño C
WHERE A.Id_casa=B.Id_casa AND B.RUT=C.RUT
GROUP BY C.RUT ;
```

4. Las personas de la BD son los arrendatarios y los dueños. Para entregar ambos, hay que realizar una unión. **Nota:** para realizar una unión, **los esquemas deben ser compatibles** (atributos con mismo nombre y dominio). Afortunadamente, éste es el caso.

```
SELECT * FROM Arrendatario UNION SELECT * FROM Dueño ;
```

5. Hay dos maneras de hacer esto: con agregación y sin ésta. El caso sin agregación (menos evidente en general) consiste en hacer un join de tres tablas.

Sin agregación:

```
SELECT A.RUT, A.Nombre, A.Apellido
FROM Dueño A, Casa C1, Casa C2, Casa C3
WHERE A.RUT=C1.RUT AND C1.RUT=C2.RUT AND C2.RUT=C3.RUT
  AND C1.Id_casa<>C2.Id_casa AND C1.Id_casa<>C3.Id_casa
  AND C2.Id_casa<>C3.Id_casa ;
```

Con agregación: en este caso, es necesario utilizar HAVING. HAVING es el WHERE pero para funciones agregadas. En el HAVING sólo pueden aparecer funciones agregadas y constantes.

```
SELECT A.RUT, A.Nombre, A.Apellido
FROM Dueño A, Casa C
WHERE A.RUT=C.RUT
GROUP BY A.RUT
HAVING COUNT(DISTINCT C.Id_casa)>=3 ;
```

6. Jugando con la semántica vemos que un dueño con deudores en todas sus casas equivale a un dueño con deuda en todas sus casas. Y el complemento de eso son los dueños con casas sin deudas.

```
SELECT D.RUT, D.Nombre, D.Apellido
FROM Dueño D, Casa C
WHERE D.RUT=C.RUT
EXCEPT
SELECT D.RUT, D.Nombre, D.Apellido
FROM Dueño D, Casa C, Arrienda A
WHERE D.RUT=C.RUT AND C.Id_casa=A.Id_casa AND A.Deuda>0
```

Otra manera consiste en exigir que la deuda de cada casa del dueño sea positiva. En este caso, una consulta anidada exigiendo igualdad sobre ALL basta.

```
SELECT D.RUT, D.Nombre, D.Apellido
FROM Dueño D, Casa C
WHERE D.RUT=C.RUT
  AND 0 = ALL (SELECT A.Deuda
               FROM Arrienda A
               WHERE C.Id_casa=A.Id_casa)
  OR NOT EXISTS (SELECT *
               FROM Arrienda A
               WHERE C.Id_casa=A.Id_casa) ;
```

Esto es equivalente a lo anterior: que no exista (NOT EXISTS) una casa con deuda para este dueño. De hecho, es más eficiente.

```
SELECT D.RUT, D.Nombre, D.Apellido
FROM Dueño D, Casa C
WHERE D.RUT=C.RUT
  AND NOT EXISTS (SELECT *
                  FROM Arrienda A
                  WHERE C.Id_casa=A.Id_casa AND A.Deuda>0) ;
```

7. En esta sección veremos cómo calcular estadísticas con SQL estándar.

El promedio de arrendatarios por casa: la manera correcta de hacerlo es considerar todos los arrendatarios -que efectivamente arrienden- y todas las casas. Así se contarán las casas con 0 arrendatarios, que deberían ser consideras en el promedio.

```
SELECT DISTINCT COUNT(DISTINCT A.RUT)/COUNT(DISTINCT B.Id_casa)
FROM Arrienda A, Casa B ;
```

El máximo requiere una consulta anidada sencilla. Hay que contar el número de arrendatarios por cada grupo y exigir que sea mayor a los de los demás grupos. **Esta consulta tiene un error. ¿Cuál es? ¿Cómo lo corrige?**

```
SELECT COUNT(A.RUT)
FROM Arrienda A
GROUP BY A.Id_casa
HAVING COUNT(A.RUT) >= ALL(SELECT COUNT(B.RUT)
                           FROM Arrienda B
                           GROUP BY B.Id_casa) ;
```

El mínimo es análogo. **Esta consulta tiene un error. ¿Cuál es?[1] ¿Cómo lo corrige?**

```
SELECT COUNT(A.RUT)
FROM Arrienda A
GROUP BY A.Id_casa
HAVING COUNT(A.RUT) <= ALL(SELECT COUNT(B.RUT)
                           FROM Arrienda B
                           GROUP BY B.Id_casa) ;
```

1 Hint: ¿hay casas sin arrendatarios?

Ahora necesito la siguiente consulta: el número de arrendatarios por casa. Se advierte que la práctica de realizar consultas anidadas en el FROM se debe evitar si se puede usar IN, NOT IN, EXISTS, ALL, etc.

```
(*) = SELECT Id_casa AS ID, COUNT(DISTINCT RUT) AS N
      FROM Arrienda
      GROUP BY Id_casa
      UNION
      SELECT Id_casa AS ID, 0 AS N
      FROM Casa
      WHERE Id_casa NOT IN (SELECT X.Id_casa
                            FROM Arrienda X) ;
```

La varianza es la desviación de la media. Puesto que dos operaciones agregadas son necesarias para resolver esto, forzosamente necesitamos realizar una consulta previa e incluirla en el FROM.

Fórmula: $Var(X)=E(X^2)-E^2(X)$

```
SELECT SUM(N*N)/COUNT(ID) - AVG(N)*AVG(N)
FROM (*);
```

La moda es la frecuencia más repetida. La frecuencia es el COUNT, por ende aplico la comparación en el HAVING.

```
SELECT X.N
FROM (*) X
GROUP BY X.N
HAVING COUNT(DISTINCT X.ID) >= ALL (SELECT COUNT(DISTINCT Y.ID)
                                    FROM (*) Y
                                    GROUP BY Y.N) ;
```

La mediana es la frecuencia central. **Se supondrá que es única.**

Propuesto: ¿Cómo se puede hacer en los casos con, por ejemplo, 4 candidatos? Algo como: 0 0 0 0 1 2 3 3 4 4 4 4 5 5 6 7 8 9 9 11. No es mucho más complicado que la consulta siguiente puesto que hay que hacer una pequeña observación para contestar la pregunta.

```
SELECT DISTICT X.N
FROM (*) X, (*) Y, (*) Z
WHERE X.N > Y.N AND X.N < Z.N
GROUP BY X.N
HAVING COUNT(DISTINCT Y.ID) = COUNT(DISTINCT Z.ID) ;
```