**1.La representación tabular de un conjunto de entidades débil consiste:**

1. en una tabla con una columna por cada atributo de la misma y los atributos de la clave primaria del conjunto de entidades fuerte del que depende aquella
2. en una tabla con una columna por cada atributo de la misma
3. ninguna de las anteriores

**2.Dentro de los lenguajes de consulta procedimentales se encuentran:**

1. el álgebra relacional y el cálculo relacional de tuplas
2. el álgebra relacional y el cálculo relacional de dominios
3. ninguna de las anteriores

**3.Dados los esquemas prestatario(nombre-cliente, número-préstamo) y préstamo(nombre-sucursal, número-préstamo, importe), entonces la relación r 🡨prestatario x préstamo:**

1. se representaría tabularmente mediante cuatro columnas: {nombre-cliente, numero-préstamo, nombre-sucursal, importe}
2. se representaría tabularmente mediante cinco columnas: {nombre-cliente, prestatario.numero-prestatario, nombre-sucursal, préstamo.numero-préstamo, importe}
3. ninguna de las anteriores

**4.Suponga que queremos encontrar los nombres de los clientes que viven en la ciudad “A” entonces la consulta será:**

https://i.gyazo.com/7d4236cf6919fd148e2854fb47640936.png

1. Sí
2. No
3. Ninguna de las anteriores

**5.En el diagrama E-R el esquema de cuenta es (número-cuenta, saldo), entonces ¿es correcta la consulta (∏saldo (cuenta x cuenta))?**

1. Si, siempre
2. No, nunca
3. Depende

**6.La consulta ∏nombre-cliente, préstamo.número-préstamo, importe (prestatario >< préstamo) es equivalente a ∏nombre-cliente, préstamo.número-prestamo.importe (prestatario x préstamo)?**

1. Sí, siempre
2. No, nunca
3. Depende del ejempla...

**7.Suponga que en una determinada BDD al menos un cliente posee un préstamo y una cuenta, entonces la consulta en SQL:**

(**select** nombre-cliente

**from** impostor)

**union**

**(select** nombre-cliente

**from** prestatario**)**

**en general producirá:**

1. Como máximo dos tuplas
2. Como mínimo dos tuplas
3. Ninguna de las anteriores

**8.El álgebra relacional y el cálculo relacional de dominios**

1. tienen igual potencia expresiva si el último de ellos está restringido a expresiones seguras
2. el álgebra relacional es siempre más potente
3. ninguna de las anteriores

**9.La consulta en SQL**

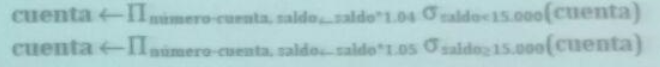
**Select** nombre-sucursal, avg(saldo)

**From** cuenta

**Group by** nombre sucursal

1. Es incorrecta
2. Genera una tabla con una columna
3. Ninguna de las anteriores

**Suponga que quiere actualizar las cuentas pagando intereses del 4% a las cuentas con saldos inferiores a 15000 euros y del 5% a las cuentas con saldos superiores a 15000 euros, entonces, ¿es correcta la secuencia de operaciones?**



1. Sí, siempre
2. No, nunca
3. Ninguna de las anteriores

**11.Los esquemas de las relaciones préstamo y prestatario son, respectivamente, (nombre-sucursal, número-préstamo, importe), (nombre-cliente, número-préstamo) entonces ¿tiene sentido las relaciones préstamo U prestatario y préstamo x prestatario?**

1. La primera sí, pero la segunda no
2. La segunda sí, pero la primera no
3. Ninguna de las anteriores

**12.Se dice que una expresión del cálculo relacional de tuplas es segura si:**

1. el dominio de la expresión no es infinito
2. los valores que aparecen en el resultado de la expresión son valores del dominio de la expresión.
3. Ninguna de las anteriores

**13.En el diseño de BBDD, para modelar una determinada situación, podemos asegurar**

1. Que SIEMPRE es mejor tener un elevado número de tablas con pocos atributos a un número reducido de tablas con muchos atributos
2. Que SIEMPRE es mejor tener un reducido número de talas con muchos atributos a un número elevado de tablas con pocos atributos
3. Ninguna de las anteriores

**14. Una tupla de una determinada relación que no se una con ninguna tupla de otra relación**

1. Nunca se puede dar en un diseño de BBDD
2. En algunos casos se puede dar y llama *tupla colgante*
3. Ninguna de las anteriores

**15. En la representación tabular de la generalización, si la especialización es disjunta y completa**

1. No se crea una tabla para la entidad de nivel más alto sino una tabla para cada entidad de nivel más bajo con sus atributos además de los atributos de la entidad nivel superior.
2. Siempre se cera una tabla para la entidad de nivel más alto.
3. Ninguna de las anteriores.

**16. En general, las consultas en SQL:**

1. Pueden producir tuplas duplicadas ya que eliminarlas es computacionalmente costoso.
2. Nunca producen tuplas duplicadas.
3. Ninguna de las anteriores

**17. En SQL las expresiones algebraicas como *importe\*100* en la siguiente consulta**

**select** *nombre-sucursal, numero-prestamo, importe\*100*

**from** *prestamo*

1. No son admisibles, habría que consultar primero importe y luego multiplicarlo por 100.
2. Son admisibles.
3. Ninguna de las anteriores.

**18. En SQL la operación renombramiento se implementa con “as”, esta operación**

1. Se puede emplear en la cláusula **select** pero no en **from**, es decir, por ejemplo es admisible:

**Select** *nombre-cliente* **as** *cliente*

**From** *prestatario*

pero no es admisible:

**select** *T.cliente*

**from** prestatario **as** T

1. La clausula “as” se puede emplear tanto en el **select** como en el **from**
2. Ninguna de las anteriores.

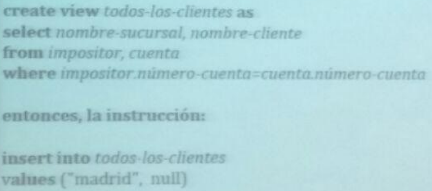
**19. En SQL las instrucciones *delete* y *drop*:**

1. Son equivalentes.
2. Se emplean para borrar información de una BBDD, pero no producen exactamente el mismo resultado.
3. Ninguna de las anteriores.

**20. Suponga que ha definido una vista en una BBDD y que quiere incluir nueva información en dicha vista, en general lo más adecuado es**:

1. Insertar la información directamente en la vista.
2. Insertar la información en las relaciones sobre las que se construye la vista.
3. No es posible insertar nueva información en una vista.

**21. Dada la definición de vista en SQL:**



1. Es incorrecta porque nunca se puede insertar un valor nulo en una tupla.
2. Es incorrecta porque *todos-los-clientes* no es una relación real de la BBDD.
3. Ninguna de las anteriores.

**22. Suponga que queremos combinar la información de dos relaciones en una nueva, pero que algunos atributos no son compartidos por las dos relaciones, entonces, la operación que en general resultará más adecuada es:**

1. El producto cartesiano.
2. La unión.
3. La reunión externa.

**23. La cláusula *select* de SQL es equivalente, en algebra relacional:**

1. A una proyección.
2. A una selección.
3. A una asignación.

**24. En una consulta en SQL, la cláusula *where***

1. Puede formar parte o no de la consulta.
2. Siempre está presente explícitamente.
3. No se emplea en SQL.

**25.En SQL la orden *create* table se emplea**

1. Siempre previamente a cualquier otra, para indicar que vamos a realizar una consulta.
2. No existe.
3. Ninguna de las anteriores.