**Examen enero 2019 preguntas resueltas**

JDBC

**Qué es un driver?**

Es una implementación de diferentes clases e interfaces Java que abstraen las funciones de acceso a una base de datos para que los programas Java puedan acceder a diferentes SGBD a través de SQL.

**Tipos de ResultSet:**

* En función de su navegabilidad e insensibilidad

Desplazable Posicionable Sensible

TYPE\_FORWARD\_ONLY NO NO NO

TYPE\_SCROLL\_INSENSITIVE SI SI NO

TYPE\_SCROLL\_SENSITIVE SI SI SI

* En función de su capacidad para actualizar

CONCUR\_READ\_ONLY NO actualizable

CONCUR\_UPDATABLE SI actualizable

**Transacciones y niveles de aislamiento como en prácticas**

Una transacción es una secuencia de una o más operaciones que reflejan una sola operación en el mundo real. Características de una transacción:

acid

* Atomicidad: o se completan, o no se completan, nunca quedan a medias.
* Consistencia: sólo se guardan datos válidos.
* Isolation (Aislamiento): las transacciones no se afectan las unas a las otras.
* Durabilidad: los datos una vez escritos nunca serán perdidos.

El aislamiento es una propiedad de las transacciones que define cuándo y cómo los cambios producidos por una operación se hacen visibles para las demás operaciones concurrentes.

Niveles de aislamiento:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Lectura sucia | Lectura no repetible | Lectura fantasma |
| Read uncommitted | Si | Si | Si |
| Read committed |  | Si | Si |
| Repeteable read |  |  | Si |
| Serializable |  |  |  |

A menor nivel de aislamiento, el acceso a los datos simultáneamente se facilita, pero ocurren efectos no deseados y viceversa.

**Problemas de una transacción concurrente**

Ya se ha explicado antes, a mayor concurrencia, mayor riesgo de que se produzcan efectos no deseados.

JPA

**Entidad y ValueType. ¿Qué es mutable e inmutable en cada uno de ellos? ¿Sobre qué atributos se debe deben definir los métodos hashCode() y equals() en cada uno de ellos?**

1º Solución: Resuelta en Preguntas RI.pdf pregunta nº8

* Entidades (Entities): Una entidad representa un concepto del

dominio. Se puede asociar a otras entidades. Tiene un ciclo de

vida independiente

* Tipos valor (value type): Value Types representas información

adicional, no conceptos principales. Atributos de una entidad. Su

ciclo de vida depende de la entidad. No puede tener referencias

entrantes

* Los métodos se deben definir sobre la clave natural de las

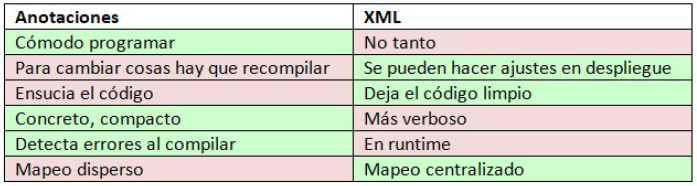
Entidades y sobre todos los atributos en los valueType.

2º Solución: Resuelta en Examen-RI-Febrero-2014(Propuesta solución II) y preguntas interesantes.pdf

* Entidades (Entities): Un entidad representa la existencia de “algo” en el dominio (de la realidad) que tiene identidad propia. Sus propiedades pueden cambiar a lo largo del tiempo pero sigue siendo “ella”. Puede estar asociada con otras entidades Su ciclo de vida es independiente de otras entidades. Debe tener una identidad (lo que en BDD llamaríamos clave primaria). Mutables serían todos aquellos atributos que se pudieran modificar en un futuro y rompieran el encapsulamiento. HashCode y equals redefinido SÓLO sobre los atributos que determinan identidad.
* Tipos valor (value type): Representan un valor, no tienen identidad. Ej: Una moneda de 2€ no importa si es esta o aquella, sólo importa que es de 2€. Su valor es inalterable. Se suelen presentar como atributos de una entidad Su ciclo de vida depende enteramente de la entidad que las posee. Son atributos y son inmutables, no se cambian sus valores (no tienen setters). HashCode y equals redefinido sobre TODOS los atributos.

**Diferencias XML y anotaciones.**

ES. Mapeo de clases.pdf pág 13/99



**Cómo se debe usar el atributo @ID.**

El atributo @id se utiliza para comprobar si dos entidades de la base de datos son la misma

a.getId().equals(b.getId()) Si es true, es la misma entidad.

Cuando estamos codificando las clases del modelo del dominio, deberemos poner la anotación jpa @id en el atributo de la entidad que vaya a ser clave en la tabla de la base de datos.

**Consulta sobre matrícula alumno y asignatura (todos los alumnos con nota mayor de 5 y matriculados en una asignatura con un código).**

select a

from alumnos a

where a.id in(

select m.idAlumno

from matriculas m

where m.nota>=5

and m.idAsignatura=?

)

**Qué consulta está mal y por qué:**

select a.vehiculo.cliente

from Averia a

where a.factura.fecha = '12/12/2014'

select f.averias.vehiculo.cliente

from Factura f

where f.fecha = '12/12/2014'

La segunda consulta está mal porque sólo se puede encadenar de esa manera cuando la relacción de cardinalidad es de muchos a uno, o de uno a uno.

Por ejemplo, de WorkOrder a Vehicle se puede porque una WorkOrder puede tener un solo vehículo mientras que un vechículo puede tener varias WorkOrder, en ese caso si se puede acceder directamente al Vehicle de una WorkOrder (w.vehicle) porque estamos ante una relacción de muchos (WorkOrder) a uno (vehicle) y vamos desde workorder hasta vehicle.

Mientras que una factura tiene varias WorkOrder, por eso no se puede hacer de esa manera, porque estamos ante una relación de uno (factura) a muchos (WorkOrder).

Recuperación de información

**Definir consulta.**

Una consulta es la traducción de una necesidad de información.

**Ventaja stemming (estematización)**

Reducción de palabras a su raíz (que no su lema). P.ej Universidad, universitarios, universitarias, universitaria, universitario.

Ventajas:

* Reduce el número de términos que conforman el lenguaje de indexación
* Aglutina términos que están relacionados semánticamente

Problema principal:

* Overstemming: Términos no relacionados entre sí pueden reducirse al mismo stem (p.ej. universidad y universo, libro y librar)

**Idf.**

El número de documentos de la colección en que aparece un término a la inversa: inverse document frequency.

A mayor número de documentos que contienen un término, menor es la importancia del mismo y viceversa.

**¿Se puede hacer un estudio de la exhaustividad de una búsqueda en la web?**

No, se podría hacer en un entorno controlado donde ya sepamos cuántos documentos son relevantes de antemano (Si ya sabemos que hay 10 relevantes, y la búsqueda retorna 15 documentos de los cuales 10 son relevantes, la precisión no es del 100% pero la exhaustividad es la máxima puesto que retorna todos los documentos relevantes). En la web el número de resultados es prácticamente infinito, por lo que aunque retorne 10 documentos y los 10 sean relevantes, sí, la precisión es del 100%, pero la exhaustividad no se puede calcular porque no se sabe cuantos documentos relevantes existen realmente. (Dejamos a parte consultas de el número de la lotería y la cura de enfermedades que no tienen cura y ese tipo de consultas que se sabe ya que no hay documentos relevantes).

**Diferencias recuperación de información tradicional a en la web.**

La recuperación en la Web es muy distinta:

* Cantidad de documentos es muchísimo mayor.
* Mucha mayor heterogeneidad
* Entorno adversarial
* Es preciso explotar la estructura de hiperenlaces (p.ej. PageRank)
* Puede explotarse el comportamiento agregado de los usuarios (búsquedas recomendadas, similares, learning to rank)

NO-SQL

**Explicar brevemente el teorema CAP.**

El teorema CAP o teorema Brewer en NoSql, dice que en sistemas distribuidos es imposible garantizar a la vez: consistencia, disponibilidad y tolerancia a particiones.

* AP: garantizan disponibilidad y tolerancia a particiones, pero no la consistencia, al menos de forma total.
* CP: garantizan consistencia y tolerancia a particiones. Para lograr la consistencia y replicar los datos a través de los nodos, sacrifican la disponibilidad.
* CA: garantizan consistencia y disponibilidad, pero tienen problemas con la tolerancia a particiones. Este problema lo suelen gestionar replicando los datos.

**Modelos de datos NoSql.**

a. Clave-Valor (Key-Value Stores)

b. Documental (Document Databases)

c. Almacenamiento en columnas (Column-Family Stores)

d. Grafo (Graph Databases)

**Consistencia eventual.**

Datos consistentes pasado un tiempo. Se permite inconsistencia en la replicación, pero eventualmente todos los nodos estarán actualizados.

**Quorums.**

Son el número de nodos en los que hay que escribir un dato antes de indicarle al cliente que su dato ya está grabado. Después de eso, los datos se siguen propagando por las distintas réplicas.

En la lectura es el número de nodos que hay que contactar para leer un dato (y que devuelvan el mismo valor) antes de devolver al cliente el valor pedido.

Según se escoja entre R y W, no hace falta contactar con el mismo número de nodos para asegurar que cuando un cliente pide leer un dato, es seguro que el dato que se devuelve es correcto.

**Consulta que devuelva películas que no tienen director.**

MATCH (m:Movie) WHERE NOT ((m) <-[:DIRECTED]())