Tarea No. 12. Ciclo de vida de Hibernate

Barrera Pérez Carlos Tonatihu
Profesor: José Asunción Enríquez Zárate
Web Application Development
Grupo: 3CM9

17 de marzo de 2019

Índice

1. Introducción

Hibernate es todo un mundo en cuanto al desarrollo de aplicaciones que interactúen con una fuente de datos mediante un ORM por lo que entender el ciclo de vida de Hibernate es clave en su uso, es por esto que en esta tarea se explicara en que consiste este tema.

2. Desarrollo

El ciclo de vida de Hibernate muestra el como los objetos persistentes son manejados. Hibernate guarda, actualiza y borra registros en las tablas de la base de datos con respecto al valor del objeto ya que su clase persistente está asociada con la tabla de la base de datos. Como se puede observar en la figura 1 Hibernate presenta los siguientes estados.

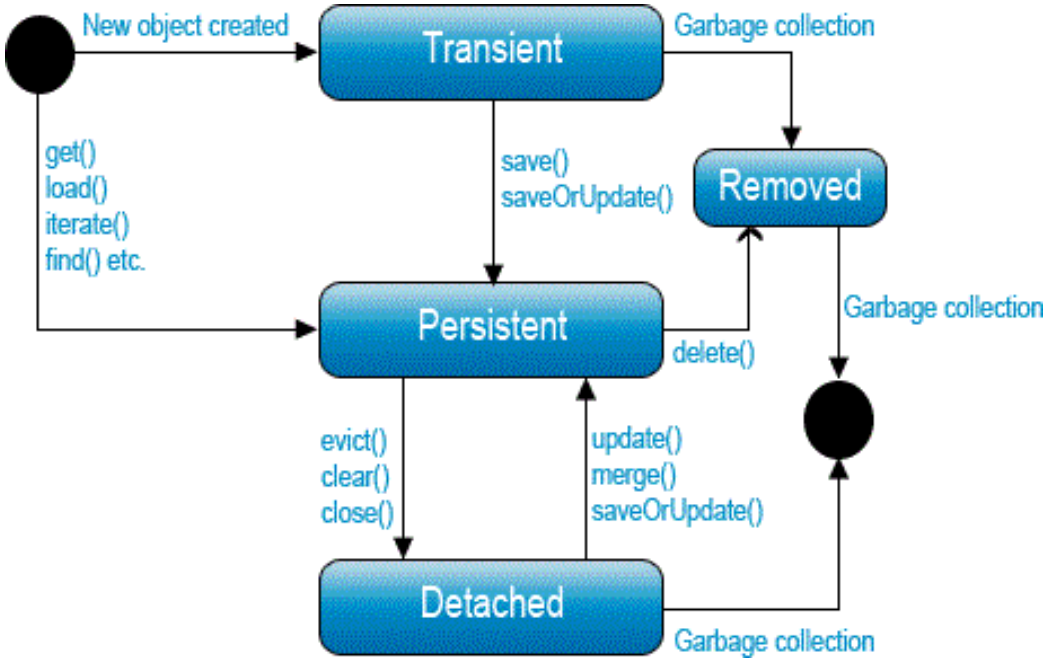

Figura 1: Ciclo de vida de Hibernate

2.1. Transient

Cuando creamos un objeto de nuestra clase POJO podemos decir que se encuentra en el estado trasient. En este estado no se encuentra relacionado

con ninguna tabla de la base de datos, por lo que cualquier modificación no afecta a la base de datos. Si ya no es referenciado por algún objeto pasa al garbage collection. Por ejemplo.

```
1 Empleado empleado = new Empleado // Objeto es estado
    transient
```

2.2. Persistent

Cuando se guarda un objeto transient entra en estado persistent. En este estado tiene una fila de alguna tabla de base de datos asociada con una llave primaria. es inicializado por el manejador de persistencia cuando se llama el método save(). Si ya existe en la base de datos se puede recuperar. Y todos los cambios hechos en este estado son guardados automáticamente.

Se puede obtener un objeto persistente al utilizar alguno de estos metodos en una sesión de Hibernate.

- save()
- update()
- saveOrUpdate()
- lock()
- merge()

Un ejemplo de esto es el siguiente código.

```
1
2 Session session = Session.getCurrentSession();
3
4 Empleado empleado = new Empleado(); // Estado transient
5 session.saveOrUpdate(empleado); // Estado persistent
6
7 empleado.setNombre("Carlos"); // Modificacion es guardada
8
9 session.getTransaction().commit(); // Commit a la
    transaccion
```

2.3. Detached

En este estado el objeto persistente continua existiendo después del cierre de la sesión activa. Se puede decir que un objeto detached se mantiene despues de que termina una transacción por lo que continua representando una fila valida de la base de datos.

Los cambios hechos en este tipo de objetos no son guardados en la base de datos, para entrar en este estado se utiliza el método evict() o en la sesión se utiliza se utiliza el método clear() o close(). Para volver a hacer un objeto persistente se pueden llamar a los siguientes métodos.

- update()
- merge()
- saveOrUpdate()
- lock()

2.4. Removed

Cuando el objeto persistente es borrado de la base de datos pasa el estado de removido, para hacer esto se utiliza el método delete(). Un ejemplo de esto es lo siguiente.

```
1 Session sesion = Session.getCurrentSession();
2 Empleado empleado = sesion.get(Empleado.class, 2);
3 sesion.delete(empleado);
4 empleado.setName("Juan"); // Esto ya no tiene efecto
5 sesion.getTransaction.commit();
```

3. Conclusiones

Hibernate tiene una gran cantidad de conceptos clave para poder utilizarlo correctamente, este es uno de los más fundamentales debido a que al momento de estar trabajando con entidades se pueden presentar bastantes problemas y si no entendemos el ciclo de vida pude volver la resolución de estos problemas algo más complicado de lo que seria en un principio.