|  |  |
| --- | --- |
| **Save()** | **Persist()** |
| Return generated identity value and is Serializable | Return type is void |
| Uses same generators to generate id | Does not use same generator |
| It is only supported by Hibernate | It is also supported by JPA |
| **Get()** | **Load()** |
| Eager loading | Perform lazy loading |
| Does not generate proxy object | Generates proxy object |
| Involves 1 real object in loading | Involves 1 real and 1 proxy object in loading |
| Use when we have to check whether record is there or not | We should use this method only when we know record is there otherwise throws ObjectNotFoundException |
| Useful for **guaranteed utilization** of the object once it is loaded. | Useful for **delayed for non-guaranteed utilization** of record once it is loaded. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Deleting object**

1. By using session.delete(obj)
2. By loading the object and then use above method.

**Updating the record**

1. **Using session.update()**
2. **Load object then update**
3. **Setting object values in transaction (synchronization process)**

**Approaches**

1. **Using creating object, setting all values and using session.update(object) method** : here we have to create and set all values(values to be updates and not updated also.) We have to provide new values and old values whoch do not require any updation.

Limitations:

Here no provision for checking the record whether it exist or not.

We need to remember the old values and new values in order to update an object. If we are not setting values which don’t require any updation, by default null values will be store.

1. **Loading then updation**

Her we have to load object then call update method. Here no need to set unnecessary values.

1. **Doing updation in transaction (Synchronization process)**

|  |  |  |
| --- | --- | --- |
| **Update** | **saveOrUpdate** | **Merge** |
| Performs object/record update operation | Performs save/insert or update operation on record | Same |
| Does not check availability of object | Check availability of an object | Same |
| Generates update query | Generates select query and insert/update query | Same |
| Do not return any object representing updated record | Same | Returns the object that has been updated |
| Cannot perform record merging | Cannot perform record merging | **Performs record merging** |
|  |  |  |
|  |  |  |

**Cache**

**It is memory to store the data temporary to reduce network trip between client application and server.**

**In hibernate, cache resides at client application by holding domain class object.**

**Types of cache:**

L1 level cache/L1 cache

L2 level cache/L2 cache

L1 cache is associated with session object

L2 cache associated with session factory object and it is available across all sessions.

L1 cache is created with session object and destroyed when session is closed. We no need to do it explicitly.

**Flow:**

When session loads data, it searched the data in L1 cache, if it is not available, it will get it from DB and keep it into L1 and uses L1 cache for multiple requests.

**States of domain class object**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Transient** | not associated with session | * **Doesn’t contain id value** * **Doesn’t represent DB record** * **No associated with session** |
| **Persist** | associated with session | * **Contains id value** * **Associated with session** * **Represents DB records** * **Maintains synchronization** |
| **Detached** | previously associated but currently not associated. | * **Contains id values** * **Doesn’t represents record** * **Does not maintain synchronization** |