**Hibernate Cache Support-**

**Why?**

To reduce the number of calls from database and improve the application performance.

**How it works?**

It will fetch only one time data from database and store it on local cache (temporary storage).

For n no. of user retrieve data only once from database and store it to cache and return it to user.

There are **three types** of cache in the hibernate as

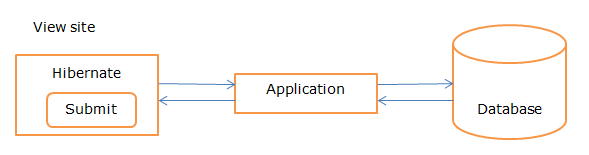
Session level cache

SessionFactory level cache

Query level cache

Problem without cache-

Suppose I have Java study material application or site. It is the constant data, suppose 1 lakh user daily visit to site to read the concepts, so it will hit 1 lakh times to database due to this your application will slow and lot of processing time it will takes. To overcome this issue, we should go for cache supports.



**Session level cache-**

It is called as first level cache.

It is apply for single user or one user only.

It is default level cache.

The first level cache data will not be available to entire application because application can use many session objects.

**When to use?**

Example- Login to Gmail application, if you want to retrieve the inbox mails at first time login. It will load the data from database. If you trying to refresh and if you do not have new mails. The data instead of reading every time from database, it will load from cache itself until doing logout. It will do only one select operation.

Note-

It will fetch data only one time from database and store it on session objects next time when user request some data, so it will retrieve data from session objects.

Program- Using Session

**Student.java**

**package** com.session;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

@Entity

**public** **class** Student {

@Id

**private** **int** id;

**private** String name;

**private** String city;

**private** String mobile;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

**public** String getMobile() {

**return** mobile;

}

**public** **void** setMobile(String mobile) {

**this**.mobile = mobile;

}

}

**Pom.xml**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>session</groupId>

<artifactId>SessionDemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>4.1.4.Final</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.21</version>

</dependency>

</dependencies>

</project>

**hibernate.cfg.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE hibernate-configuration SYSTEM

"classpath://org/hibernate/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name=*"hibernate.connection.driver\_class"*>com.mysql.jdbc.Driver</property>

<property name=*"hibernate.connection.url"*>jdbc:mysql://localhost:3306/test</property>

<property name=*"hibernate.connection.username"*>root</property>

<property name=*"hibernate.connection.password"*>root</property>

<property name=*"hibernate.dialect"*>org.hibernate.dialect.MySQLDialect</property>

<property name=*"hbm2ddl.auto"*>update</property>

<property name=*"show\_sql"*>true</property>

<mapping class=*"com.session.Student"*></mapping>

</session-factory>

</hibernate-configuration>

**Test.java**

**package** com.session;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.cfg.Configuration;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

Configuration configuration = **new** Configuration();

configuration.configure("hibernate.cfg.xml");

SessionFactory sessionFactory = configuration.~~buildSessionFactory~~();

Session session = sessionFactory.openSession();

// we are using two session objects.

Student s1 = (Student) session.get(Student.**class**, 8);

System.***out***.println("First student data is>>");

System.***out***.println("Name>>" + s1.getName());

System.***out***.println("City>>" + s1.getCity());

System.***out***.println("Mobile>>" + s1.getMobile());

Student s2 = (Student) session.get(Student.**class**, 8);

System.***out***.println("Second student data is>>");

System.***out***.println("Name>>" + s2.getName());

System.***out***.println("City>>" + s2.getCity());

System.***out***.println("Mobile>>" + s2.getMobile());

}

}

**Query level cache-**

**Why?**

If I want to execute some query again and again or multiple times (Example select max salary from employee) then you should go for query level cache.

**Student.java**

**package** com.test;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

@Entity

**public** **class** Student {

@Id

**private** **int** id;

**private** String name;

**private** String city;

**private** String mobile;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

**public** String getMobile() {

**return** mobile;

}

**public** **void** setMobile(String mobile) {

**this**.mobile = mobile;

}

}

**Hibernate.cfg.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE hibernate-configuration SYSTEM

"classpath://org/hibernate/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name=*"hibernate.connection.driver\_class"*>com.mysql.jdbc.Driver</property>

<property name=*"hibernate.connection.url"*>jdbc:mysql://localhost:3306/test</property>

<property name=*"hibernate.connection.username"*>root</property>

<property name=*"hibernate.connection.password"*>root</property>

<property name=*"hibernate.dialect"*>org.hibernate.dialect.MySQLDialect</property>

<property name=*"hbm2ddl.auto"*>update</property>

<property name=*"show\_sql"*>true</property>

<mapping class=*"com.test.Student"*></mapping>

</session-factory>

</hibernate-configuration>

**Test.java**

**package** com.demo;

**import** java.util.List;

**import** org.hibernate.Query;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.Transaction;

**import** org.hibernate.cfg.Configuration;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

Configuration configuration = **new** Configuration();

configuration.configure("hibernate.cfg.xml");

SessionFactory sessionFactory = configuration.~~buildSessionFactory~~();

Session session = sessionFactory.openSession();

System.***out***.println("first query>>");

Query query = session.createQuery("from Student");

query.setCacheable(**true**);

List<Student> students = query.list();

**for** (Student student : students) {

System.***out***.println("ID=" + student.getId());

System.***out***.println("City=" + student.getCity());

System.***out***.println("Mobile=" + student.getMobile());

System.***out***.println("Name=" + student.getMobile());

}

session.close();

sessionFactory.close();

System.***out***.println("Record retrieved successfully.");

}

}

**Pom.xml**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>hibernate</groupId>

<artifactId>HibernateSample</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>4.1.4.Final</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.21</version>

</dependency>

</dependencies>

</project>

Output-

first query>>

Hibernate: select student0\_.id as id0\_, student0\_.city as city0\_, student0\_.mobile as mobile0\_, student0\_.name as name0\_ from Student student0\_

ID=1

City=pune

Mobile=9887972688

Name=9887972688

ID=8

City=pune

Mobile=7575878795

Name=7575878795

Feb 12, 2021 3:07:56 PM org.hibernate.service.jdbc.connections.internal.DriverManagerConnectionProviderImpl stop

INFO: HHH000030: Cleaning up connection pool [jdbc:mysql://localhost:3306/test]

Record retrieved successfully.

**SessionFactory level cache-**

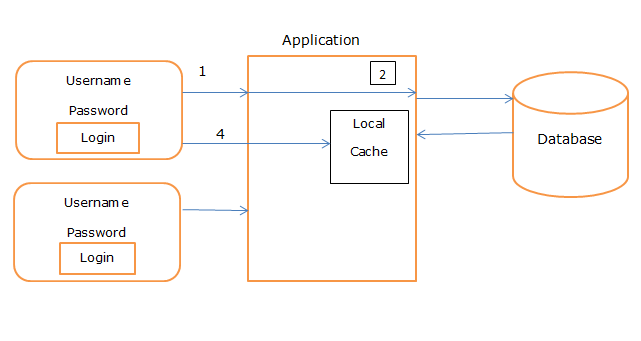
It is called as second level cache.

It will apply for all the users.

Program-2 Using SessionFactory

**Why?**

Problem with session level cache



Here admin1 and admin2 are the trying to access the same data, if we are using session, it will create the two session objects for some data or common data but if we use sessionfactory cache, data will fetch from database only one(data store under the sessionfactory) and data will available to any user.

**When?**

It is useful if you have multiple session objects from sessionfactory.

Program- using SessionFactory

**Student.java**

**package** com.session;

**import** javax.persistence.Cacheable;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

@Entity

**public** **class** Student {

@Id

**private** **int** id;

**private** String name;

**private** String city;

**private** String mobile;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

**public** String getMobile() {

**return** mobile;

}

**public** **void** setMobile(String mobile) {

**this**.mobile = mobile;

}

}

**Hibernate.cfg.xml**

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<!DOCTYPE hibernate-configuration SYSTEM

"classpath://org/hibernate/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name=*"hibernate.connection.driver\_class"*>com.mysql.jdbc.Driver</property>

<property name=*"hibernate.connection.url"*>jdbc:mysql://localhost:3306/test</property>

<property name=*"hibernate.connection.username"*>root</property>

<property name=*"hibernate.connection.password"*>root</property>

<property name=*"hibernate.dialect"*>org.hibernate.dialect.MySQL57Dialect</property>

<property name=*"hbm2ddl.auto"*>update</property>

<property name=*"show\_sql"*>true</property>

<property name=*"cache.use\_Second\_level\_cache"*>true</property>

<property name=*"hibernate.cache.provider\_class"*>

org.hibernate.cache.EhCacheProvider

</property>

<property name=*"net.sf.ehcache.configuration-ResourceName"*>ehcache.xml</property>

<mapping class=*"com.hibernate.Student"*></mapping>

</session-factory>

</hibernate-configuration>

Ehcache.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<ehcache xmlns:xsi=*"https://www.w3.org/2001/XMLSchema-instance"*

xsi:noNamespaceSchemaLocation=*"ehcache.xsd"* updateCheck=*"true"*

monitoring=*"autodetect"* dynamicConfig=*"true"*>

<diskStore path=*"java.io.tmpdir/ehcache"* />

<defaultCache maxEntriesLocalHeap=*"10000"* eternal=*"false"*

timeToIdleSeconds=*"120"* timeToLiveSeconds=*"120"*

diskSpoolBufferSizeMB=*"30"* maxEntriesLocalDisk=*"10000000"*

diskExpiryThreadIntervalSeconds=*"120"* memoryStoreEvictionPolicy=*"LRU"*

statistics=*"true"*>

<persistence strategy=*"localTempSwap"* />

</defaultCache>

<cache name=*"employee"* maxEntriesLocalHeap=*"10000"*

eternal=*"false"* timeToIdleSeconds=*"5"* timeToLiveSeconds=*"10"*>

<persistence strategy=*"localTempSwap"* />

</cache>

<cache name=*"org.hibernate.cache.internal.StandardQueryCache"*

maxEntriesLocalHeap=*"5"* eternal=*"false"* timeToLiveSeconds=*"120"*>

<persistence strategy=*"localTempSwap"* />

</cache>

<cache name=*"org.hibernate.cache.spi.UpdateTimestampsCache"*

maxEntriesLocalHeap=*"5000"* eternal=*"true"*>

<persistence strategy=*"localTempSwap"* />

</cache>

</ehcache>

**package** com.session;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.cfg.Configuration;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

Configuration configuration = **new** Configuration();

configuration.configure("hibernate.cfg.xml");

SessionFactory sessionFactory = configuration.~~buildSessionFactory~~();

Session session=sessionFactory.openSession();

Student s1 = (Student) session.load(Student.**class**, 1);

System.***out***.println("First student data is>>");

System.***out***.println("Name>>" + s1.getName());

System.***out***.println("City>>" + s1.getCity());

System.***out***.println("Mobile>>" + s1.getMobile());

System.***out***.println("second level cache>>");

Session session1=sessionFactory.openSession();

Student s3 = (Student) session1.load(Student.**class**, 2);

System.***out***.println("Student data is>>");

System.***out***.println("Name>>" + s3.getName());

System.***out***.println("City>>" + s3.getCity());

System.***out***.println("Mobile>>" + s3.getMobile());

}

}

**Pom.xml**

<project xmlns=*"http://maven.apache.org/POM/4.0.0"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>session</groupId>

<artifactId>SessionDemo</artifactId>

<version>0.0.1-SNAPSHOT</version>

<dependencies>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>4.1.4.Final</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.21</version>

</dependency>

<dependency>

<groupId>net.sf.ehcache</groupId>

<artifactId>ehcache-core</artifactId>

<version>2.6.9</version>

</dependency>

<!-- Hibernate EHCache API -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-ehcache</artifactId>

<version>5.2.16.Final</version>

</dependency>

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-simple</artifactId>

<version>1.7.5</version>

</dependency>

</dependencies>

</project>

**Hibernate object states-**

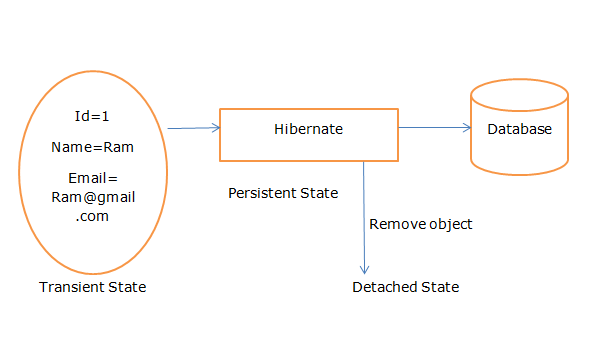
It is called as life cycle of POJO class objects.

There are three types of objects in the hibernate as-

Transient state

Persistent state

Detached state



Transient State-

Simple object or plain object or before going too attached with hibernate called as transient state.

Persistent State-

A plain java old object attached with hibernates called as persistent state.

Detached State-

A process of deleting object from hibernates called as detached state.

**How to write the POJO class?**

POJO class needs to contain the required number of property which are storing into table.

POJO class in which default constructor is mandatory.

For every property, setter and getter need to be provided.

Setter and getter access should be public.

Note- there are several names for POJO class such as formbean, valueobject, etc

Program-

**Student.Java**

**package** com.test;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

@Entity

**public** **class** Student {

@Id

**private** **int** id;

**private** String name;

**private** String city;

**private** String mobile;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getCity() {

**return** city;

}

**public** **void** setCity(String city) {

**this**.city = city;

}

**public** String getMobile() {

**return** mobile;

}

**public** **void** setMobile(String mobile) {

**this**.mobile = mobile;

}

}

**Test.java**

**package** com.test;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.cfg.Configuration;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

Student student= **new** Student();

student.setName("Ram");

student.setCity("Pune");

student.setMobile("8888978585");

//student object is transient

Configuration configuration= **new** Configuration();

configuration.configure("hibernate.cfg.xml");

SessionFactory sessionFactory=configuration.~~buildSessionFactory~~();

Session session=sessionFactory.openSession();

session.save(student);

//student object is persistent

session.beginTransaction().commit();

session.evict(student);

/\*student object is detached or student object is removed from

\*persistent state.

\* \*/

}

}