**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.

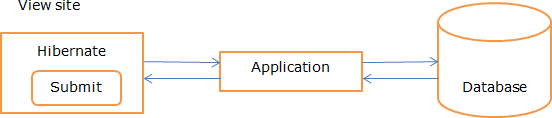
There are three types of cache in the hibernate as Session level cache

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 file

<project xmlns=[*"http://maven.apache.org/POM/4.0.0"*](http://maven.apache.org/POM/4.0.0)xmlns:xsi=[*"htt*](http://www.w3.org/2001/XMLSchema-instance)*p*[*://www.w3.org/2001/XMLSchema-instance"*](http://www.w3.org/2001/XMLSchema-instance)

xsi:schemaLocation=[*"http://ma*](http://maven.apache.org/POM/4.0.0)*v*[*en.apache.org/POM/4.0.0*](http://maven.apache.org/POM/4.0.0)[*http://maven.apache.org/xsd/maven-4.0.0.xsd"*](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.Driv er</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>

Client.java

**package** com.session;

**import** org.hibernate.Session;

# **import** org.hibernate.SessionFactory;

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

**public class** Client {

**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**,10);

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());

Output-

Hibernate: select student0\_.id as id0\_0\_, student0\_.city as city0\_0\_, student0\_.mobile as mobile0\_0\_, student0\_.name as name0\_0\_ from Student student0\_ where student0\_.id=?

First student data is>> Name>>ashok Email>>pune Address>>7575878795

Hibernate: select student0\_.id as id0\_0\_, student0\_.city as city0\_0\_, student0\_.mobile as mobile0\_0\_, student0\_.name as name0\_0\_ from Student student0\_ where student0\_.id=?

Second student data is>> Name>>rohit Email>>mumbai Address>>8989898989

Query level cache- Why?

If I want to execute some query again and again or multiple times 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.Driv er</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.test; import java.util.\*;

import org.hibernate.Query; import org.hibernate.Session;

import org.hibernate.SessionFactory; importorg.hibernate.cfg.Configuration; public class Test {

public static void main(String[] args) {

Configuration cfg = new Configuration(); cfg.configure("hibernate.cfg.xml");

SessionFactory sessionFactory = cfg.buildSessionFactory();

Session session = sessionFactory.openSession();

// pass the class name and id for updating record 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"*](http://maven.apache.org/POM/4.0.0)xmlns:xsi=[*"htt*](http://www.w3.org/2001/XMLSchema-instance)*p*[*://www.w3.org/2001/XMLSchema-instance"*](http://www.w3.org/2001/XMLSchema-instance)xsi:schemaLocation=[*"http://ma*](http://maven.apache.org/POM/4.0.0)*v*[*en.apache.org/POM/4.0.0*](http://maven.apache.org/POM/4.0.0)

[*http://maven.apache.org/xsd/maven-4.0.0.xsd"*](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

org.hibernate.service.jdbc.connections.internal.DriverManagerConnectionProviderI mpl stop

INFO: HHH000030: Cleaning up connection pool [jdbc:mysql://localhost:3306/test] Record retrieved successfully.