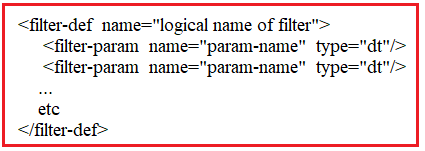
The main intension of Hibernate filter is to filter the sql query results on the basis of particular condition. The hibernate filter can be disabled or enabled for particular hibernate condition.

The filter can be defined using

1. Xml tags.
2. Annotations.

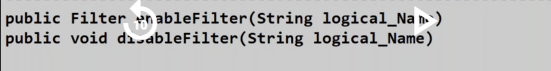
**a.XML tags:-** The fileter can be defined using <filter-def> tag. It is child tag to <hibernate-mapping> tag.



We need to call the filter in <class> tag using <filter> tag.



In Client Application, we can enable or disable the filter by using following methods of “**Session”** Interface.



The client Application can supply the value to filter parameter using following method of “Filter” interface.



Example:-1

**Domain class/POJO Class: Student.java**

package com.hib.domain;

import java.io.Serializable;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.NamedNativeQueries;

import javax.persistence.NamedNativeQuery;

import javax.persistence.Table;

@Entity

@Table(name="stu")

public class Student implements Serializable{

@Id

@Column(name="sid")

private int sid;

@Column(name="sname")

private String sname;

@Column(name="salary")

private float marks;

public int getSid() {

return sid;

}

public void setSid(int sid) {

this.sid = sid;

}

public String getSname() {

return sname;

}

public void setSname(String sname) {

this.sname = sname;

}

public float getMarks() {

return marks;

}

public void setMarks(float marks) {

this.marks = marks;

}

}

**Hibernate Mapping File: Student.hbm.xml**

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

<!DOCTYPE hibernate-mapping PUBLIC

"-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping>

<class name=*"com.hib.domain.Student"* table=*"stu"*>

<id name=*"sid"*/>

<property name=*"sname"*/>

<property name=*"marks"* column=*"salary"*/>

<filter name=*"con"* condition=*"salary != :sal"*/>

</class>

<filter-def name=*"con"*>

<filter-param name=*"sal"* type=*"int"*/>

</filter-def>

</hibernate-mapping>

**HibernateConfigFile:**

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

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name=*"hibernate.connection.driver\_class"*>oracle.jdbc.OracleDriver</property>

<property name=*"hibernate.connection.url"*>jdbc:oracle:thin:@localhost:1521:xe</property>

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

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

<property name=*"hibernate.dialect"*>org.hibernate.dialect.Oracle10gDialect</property>

<mapping resource=*"com/hib/domain/Student.hbm.xml"*/>

</session-factory>

</hibernate-configuration>

**ClientApplication:Test.java**

package com.hib.domain;

import java.sql.SQLException;

import java.util.List;

import java.util.Scanner;

import org.hibernate.Filter;

import org.hibernate.Query;

import org.hibernate.SessionFactory;

import org.hibernate.cfg.Configuration;

import org.hibernate.classic.Session;

public class Test {

public static void main(String[] args)throws SQLException {

try {

Configuration cfg=new Configuration();

cfg.configure("com/hib/domain/hibernate.cfg.xml");

SessionFactory sf=cfg.buildSessionFactory();

Session s=sf.openSession();

Query q=s.createQuery("from Student");

Filter f;

Scanner sv=new Scanner(System.in);

System.out.println("1.Enable Filter\n2.Disable Filter");

System.out.print("Enter the choice:");

int ch=sv.nextInt();

switch(ch){

case 1:

f=s.enableFilter("con");

System.out.println("Filter disabled");

System.out.print("Enter the salary:");

f.setParameter("sal", sv.nextInt());

break;

case 2:

s.disableFilter("con");

System.out.println("Filter disabled");

break;

default:

System.out.println("Worng choice");

}

System.out.println("sid\tsname\tmarks");

System.out.println("=======================");

List<Student>li=q.list();

for(Student y:li) {

System.out.println(y.getSid()+"\t"+y.getSname()+"\t"+y.getMarks());

}

}

catch(Exception e) {

e.printStackTrace();

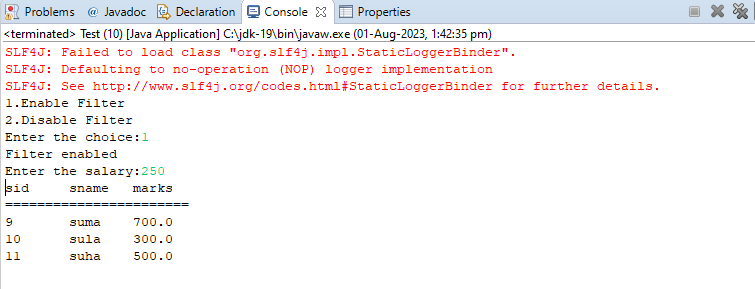
}

}

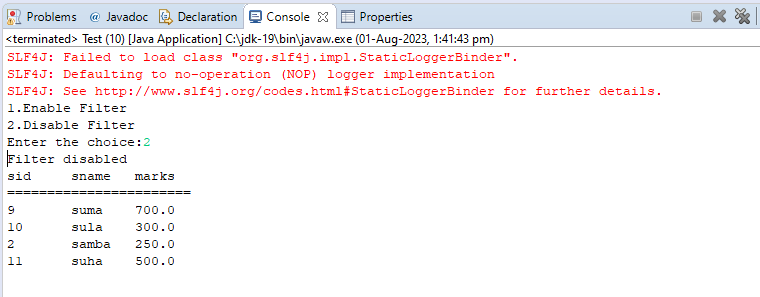
}

**Output:-**

**Run-1**

****

**Run-2**

****

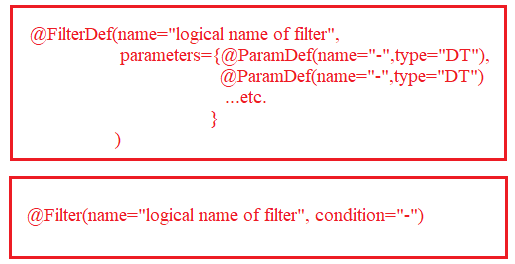
**B. Annatotaions:-**

**1.@FilterDef:-** Before you can use a filter, you need to define it. You do that using Hibernate’s @FilterDef annotation, which you can apply on the class or package level. The name attribute is its only mandatory attribute. Each filter definition requires a name that’s unique within your persistence unit. You will use this name when you apply a filter definition to an entity class or attribute.

In addition to the name, you can also define an array of parameters.

[2**.@Filter:-**](mailto:2.@Filter:-)

Syntax:



Example:-

**POJO class:-Student.java**

**package** com.hib.domain;

**import** java.io.Serializable;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

**import** javax.persistence.Table;

**import** org.hibernate.annotations.Filter;

**import** org.hibernate.annotations.FilterDef;

**import** org.hibernate.annotations.ParamDef;

@Entity

@Table(name="stu")

@FilterDef(name="con",parameters=@ParamDef(name="sal",type="int"))

@Filter(name="con" , condition="salary > :sal")

**public** **class** Student **implements** Serializable{

@Id

@Column(name="sid")

**private** **int** sid;

@Column(name="sname")

**private** String sname;

@Column(name="salary")

**private** **float** marks;

**public** **int** getSid() {

**return** sid;

}

**public** **void** setSid(**int** sid) {

**this**.sid = sid;

}

**public** String getSname() {

**return** sname;

}

**public** **void** setSname(String sname) {

**this**.sname = sname;

}

**public** **float** getMarks() {

**return** marks;

}

**public** **void** setMarks(**float** marks) {

**this**.marks = marks;

}

}

**HIBERNATE CONFIGURATION FILE:**

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

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name=*"hibernate.connection.driver\_class"*>oracle.jdbc.OracleDriver</property>

<property name=*"hibernate.connection.url"*>jdbc:oracle:thin:@localhost:1521:xe</property>

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

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

<property name=*"hibernate.dialect"*>org.hibernate.dialect.Oracle10gDialect</property>

<mapping class=*"com.hib.domain.Student"*/>

</session-factory>

</hibernate-configuration>

**ClientApplication:Test.java**

package com.hib.domain;

import java.sql.SQLException;

import java.util.List;

import org.hibernate.Filter;

import org.hibernate.Query;

import org.hibernate.SessionFactory;

import org.hibernate.cfg.Configuration;

import org.hibernate.classic.Session;

public class Test {

public static void main(String[] args)throws SQLException {

try {

Configuration cfg=new Configuration();

cfg.configure("com/hib/domain/hibernate.cfg.xml");

SessionFactory sf=cfg.buildSessionFactory();

Session s=sf.openSession();

Query q=s.createQuery("from Student");

Filter f=s.enableFilter("con");

f.setParameter("sal", 250);

System.out.println("sid\tsname\tmarks");

System.out.println("=======================");

List<Student>li=q.list();

for(Student y:li) {

System.out.println(y.getSid()+"\t"+y.getSname()+"\t"+y.getMarks());

}

}

catch(Exception e) {

e.printStackTrace();

}

}

}

Output:-

