**Hibernate data filter example – XML and annotation**

Hibernate data filter is an innovative way to filter the retrieve data from database, in a more reusable way and “visibility” rules. The data filter has a unique name, global access and accept parametrized value for the filter rule, you can enable and disabled it in a Hibernate session.

**Hibernate data filter example**

In this example, it defined a data filter and filter the collection data with specified date. The Hibernate data filter can be implemented both in XML mapping file and annotation.

**1. Hibernate data filter in XML mapping file**

Define a data filter with ‘**filter-def**‘ keyword, and accept a date parameter.

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

<filter-param name=*"stockRecordFilterParam"* type=*"date"* />

</filter-def>

##### **XML mapping example**

A XML mapping file example to declare and assign it to collection set.

<hibernate-mapping>

<class name=*"Stock"* table=*"stock"*>

...

<set name=*"stockDailyRecords"* inverse=*"true"* table=*"stock\_daily\_record"*>

<key>

<column name=*"STOCK\_ID"* not-null=*"true"* />

</key>

<one-to-many class=*" StockDailyRecord"* />

<filter name=*"stockRecordFilter"* condition=*"date >= :stockRecordFilterParam"* />

</set>

</class>

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

<filter-param name=*"stockRecordFilterParam"* type=*"date"* />

</filter-def>

</hibernate-mapping>

In **condition=”date >= :stockRecordFilterParam”**, the ‘date’ is a properties belong to ‘StockDailyRecord’.

**2. Hibernate data filter in annotation**

Define a data filter with ‘**@FilterDef**‘ keyword, and accept a date parameter with **@ParamDef**.

@FilterDef(name="stockRecordFilter",

parameters=@ParamDef( name="stockRecordFilterParam", type="date" ) )

##### **Annotation example**

An annotation file example to declare and assign it to collection set.

@Entity

@FilterDef(name="stockRecordFilter",

parameters=@ParamDef( name="stockRecordFilterParam", type="date" ) )

@Table(name = "stock", catalog = "mkyong")

**public** **class** Stock **implements** java.io.Serializable {

...

@OneToMany(fetch = FetchType.LAZY, mappedBy = "stock")

@Filter(

name = "stockRecordFilter",

condition="date >= :stockRecordFilterParam"

)

**public** Set<StockDailyRecord> getStockDailyRecords() {

**return** **this**.stockDailyRecords;

}

In **condition=”date >= :stockRecordFilterParam”**, the ‘date’ is a properties belong to ‘StockDailyRecord’.

**How to enable and disable data filter**

Enable the data filter.

Filter filter = session.enableFilter("stockRecordFilter");

filter.setParameter("stockRecordFilterParam", **new** Date());

Disable the data filter.

session.disableFilter("stockRecordFilter");

**Applying and implementing the date filter**

Here’s a code snippet to show how to applying and implementing the data filter.

Session session = HibernateUtil.getSessionFactory().openSession();

System.out.println("\*\*\*\*\*\* Enabled Filter \*\*\*\*\*\*");

Filter filter = session.enableFilter("stockRecordFilter");

filter.setParameter("stockRecordFilterParam", **new** Date());

Stock stock = (Stock)session.get(Stock.**class**, 2);

Set<StockDailyRecord> sets = stock.getStockDailyRecords();

**for**(StockDailyRecord sdr : sets){

System.out.println(sdr.getDailyRecordId());

System.out.println(sdr.getDate());

}

System.out.println("\*\*\*\*\*\* Disabled Filter \*\*\*\*\*\*");

session.disableFilter("stockRecordFilter");

//clear the loaded instance and get Stock again, for demo only

session.evict(stock);

Stock stock2 = (Stock)session.get(Stock.**class**, 2);

Set<StockDailyRecord> sets2 = stock2.getStockDailyRecords();

**for**(StockDailyRecord sdr : sets2){

System.out.println(sdr.getDailyRecordId());

System.out.println(sdr.getDate());

}

Output

\*\*\*\*\*\* Enabled Filter \*\*\*\*\*\*

58

2010-01-31

\*\*\*\*\*\* Disabled Filter \*\*\*\*\*\*

60

2010-01-02

58

2010-01-31

63

2010-01-23

61

2010-01-03

In this example (both XML and annotation), after the filter is enabled, all its ‘StockDailyRecord’ collection is filter by your parameter date.

P.S filter.setParameter(“stockRecordFilterParam”, new Date());, the current new Date is 2010-01-27.