# Hibernate Criteria examples

Hibernate Criteria API is a more object oriented and elegant alternative to Hibernate Query Language (HQL). It’s always a good solution to an application which has many optional search criteria.

**Example in HQL and Criteria**

Here’s a case study to retrieve a list of **StockDailyRecord**, with optional search criteria – start date, end date and volume, order by date.

**1. HQL example**

In HQL, you need to compare whether this is the first criteria to append the ‘where’ syntax, and format the date to a suitable format. It’s work, but the long codes are ugly, cumbersome and error-prone string concatenation may cause security concern like SQL injection.

**public** **static** List getStockDailtRecord(Date startDate, Date endDate,

Long volume, Session session) {

SimpleDateFormat sdf = **new** SimpleDateFormat("yyyy-MM-dd");

**boolean** isFirst = **true**;

StringBuilder query = **new** StringBuilder("from StockDailyRecord ");

**if** (startDate != **null**) {

**if** (isFirst) {

query.append(" where date >= '" + sdf.format(startDate) + "'");

} **else** {

query.append(" and date >= '" + sdf.format(startDate) + "'");

}

isFirst = **false**;

}

**if** (endDate != **null**) {

**if** (isFirst) {

query.append(" where date <= '" + sdf.format(endDate) + "'");

} **else** {

query.append(" and date <= '" + sdf.format(endDate) + "'");

}

isFirst = **false**;

}

**if** (volume != **null**) {

**if** (isFirst) {

query.append(" where volume >= " + volume);

} **else** {

query.append(" and volume >= " + volume);

}

isFirst = **false**;

}

query.append(" order by date");

Query result = session.createQuery(query.toString());

**return** result.list();

}

## 2. Criteria example

In Criteria, you do not need to compare whether this is the first criteria to append the ‘where’ syntax, nor format the date. The line of code is reduce and everything is handled in a more elegant and object oriented way.

public static List getStockDailyRecordCriteria(Date startDate,Date endDate,

Long volume,Session session){

Criteria criteria = session.createCriteria(StockDailyRecord.class);

**if**(startDate!=**null**){

criteria.add(Expression.ge("date",startDate));

}

**if**(endDate!=**null**){

criteria.add(Expression.le("date",endDate));

}

**if**(volume!=**null**){

criteria.add(Expression.ge("volume",volume));

}

criteria.addOrder(Order.asc("date"));

**return** criteria.list();

}

**Criteria API** Let go through some popular Criteria API functions.

**1. Criteria basic query** Create a criteria object and retrieve all the ‘StockDailyRecord’ records from database.

Criteria criteria = session.createCriteria(StockDailyRecord.class);

**2. Criteria ordering query** The result is sort by ‘date’ in ascending order.

Criteria criteria = session.createCriteria(StockDailyRecord.class) .addOrder( Order.asc("date") );

The result is sort by ‘date’ in descending order.

Criteria criteria = session.createCriteria(StockDailyRecord.class) .addOrder( Order.desc("date") );

**3. Criteria restrictions query**

The Restrictions class provide many methods to do the comparison operation.

##### **Restrictions.eq**

Make sure the valume is equal to 10000.

Criteria criteria = session.createCriteria(StockDailyRecord.**class**)

.add(Restrictions.eq("volume", 10000));

Restrictions.lt, le, gt, ge, Make sure the volume is less than 10000.

Criteria criteria = session.createCriteria(StockDailyRecord.**class**)

.add(Restrictions.lt("volume", 10000));

Make sure the volume is less than or equal to 10000.

Criteria criteria = session.createCriteria(StockDailyRecord.**class**)

.add(Restrictions.le("volume", 10000));

Make sure the volume is great than 10000.

Criteria criteria = session.createCriteria(StockDailyRecord.**class**)

.add(Restrictions.gt("volume", 10000));

Make sure the volume is great than or equal to 10000.

Criteria criteria = session.createCriteria(StockDailyRecord.**class**)

.add(Restrictions.ge("volume", 10000));

Restrictions.like

Make sure the stock name is start with ‘MKYONG’ and follow by any characters.

Criteria criteria = session.createCriteria(StockDailyRecord.**class**)

.add(Restrictions.like("stockName", "MKYONG%"));

Restrictions.between

Make sure the date is between start date and end date.

Criteria criteria = session.createCriteria(StockDailyRecord.**class**)

.add(Restrictions.between("date", startDate, endDate));

Restrictions.isNull, isNotNull

Make sure the volume is null.

Criteria criteria = session.createCriteria(StockDailyRecord.**class**)

.add(Restrictions.isNull("volume"));

Make sure the volume is not null.

Criteria criteria = session.createCriteria(StockDailyRecord.**class**) .add(Restrictions.isNotNull("volume"));

**3. Criteria paging the result**

Criteria provide few functions to make pagination extremely easy. Starting from the 20th record, and retrieve the next 10 records from database.

Criteria criteria = session.createCriteria(StockDailyRecord.**class**);

criteria.setMaxResults(10);

criteria.setFirstResult(20);

**Why not Criteria !?**

The Criteria API do bring some disadvantages.

**1. Performance issue**

You have no way to control the SQL query generated by Hibernate, if the generated query is slow, you are very hard to tune the query, and your database administrator may not like it.

**1. Maintainece issue**

All the SQL queries are scattered through the Java code, when a query went wrong, you may spend time to find the problem query in your application. On the others hand, named queries stored in the Hibernate mapping files are much more easier to maintain.

**Conclusion**

Nothing is perfect, do consider your project needs and use it wisely.