**Hibernate – Cascade example (save, update, delete and delete-orphan)**

The “Cascade” keyword is often appear on the collection mapping to manage the state of the collection automatically. In this tutorials, this one-to-many example will be used to demonstrate the cascade effect.

**Cascade save / update example**

In this example, if a ‘Stock’ is saved, all its referenced ‘stockDailyRecords’ should be saved into database as well.

**1. No save-update cascade**

if you want to save the ‘Stock’ and its referenced ‘StockDailyRecord’ into database, you need to save both individually.

Stock stock = **new** Stock();

StockDailyRecord stockDailyRecords = **new** StockDailyRecord();

//set the stock and stockDailyRecords data

stockDailyRecords.setStock(stock);

stock.getStockDailyRecords().add(stockDailyRecords);

session.save(stock);

session.save(stockDailyRecords);

Hibernate:

insert into stock (STOCK\_CODE, STOCK\_NAME)

values (?, ?)

Hibernate:

insert into stock\_daily\_record

(STOCK\_ID, PRICE\_OPEN, PRICE\_CLOSE, PRICE\_CHANGE, VOLUME, DATE)

values (?, ?, ?, ?, ?, ?)

## 2. With save-update cascade

The **cascade=”save-update”** is declared in ‘stockDailyRecords’ to enable the save-update cascade effect.

<!-- Stock.hbm.xml -->

<set name=*"stockDailyRecords"* cascade=*"save-update"* table=*"stock\_daily\_record"* .>

<key>

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

</key>

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

</set>

Stock stock = **new** Stock();

StockDailyRecord stockDailyRecords = **new** StockDailyRecord();

//set the stock and stockDailyRecords data

stockDailyRecords.setStock(stock);

stock.getStockDailyRecords().add(stockDailyRecords);

session.save(stock);

Hibernate: insert into stock (STOCK\_CODE, STOCK\_NAME) values (?, ?)

Hibernate:

insert into stock\_daily\_record

(STOCK\_ID, PRICE\_OPEN, PRICE\_CLOSE, PRICE\_CHANGE, VOLUME, DATE)

values (?, ?, ?, ?, ?, ?)

The code **session.save(stockDailyRecords);** is no longer required, when you save the ‘Stock’, it will “cascade” the save operation to it’s referenced ‘stockDailyRecords’ and save both into database automatically.

**Cascade delete example**

In this example, if a ‘Stock’ is deleted, all its referenced ‘stockDailyRecords’ should be deleted from database as well.

**1. No delete cascade** You need to loop all the ‘stockDailyRecords’ and delete it one by one.

Query q = session.createQuery("from Stock where stockCode = :stockCode ");

q.setParameter("stockCode", "4715");

Stock stock = (Stock)q.list().get(0);

**for** (StockDailyRecord sdr : stock.getStockDailyRecords()){

session.delete(sdr);

}

session.delete(stock);

Hibernate: delete from stock\_daily\_record where DAILY\_RECORD\_ID=?

Hibernate: delete from stock where STOCK\_ID=?

**2. With delete cascade**

The **cascade=”delete”** is declared in ‘stockDailyRecords’ to enable the delete cascade effect. When you delete the ‘Stock’, all its reference ‘stockDailyRecords’ will be deleted automatically.

<!-- Stock.hbm.xml -->

<set name=*"stockDailyRecords"* cascade=*"delete"* table=*"stock\_daily\_record"* .>

<key>

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

</key>

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

</set>

Query q = session.createQuery("from Stock where stockCode = :stockCode ");

q.setParameter("stockCode", "4715");

Stock stock = (Stock)q.list().get(0);

session.delete(stock);

Hibernate: delete from stock\_daily\_record where DAILY\_RECORD\_ID=?

Hibernate: delete from stock where STOCK\_ID=?

**Cascade delete-orphan example**

In above cascade delete option, if you delete a Stock , all its referenced ‘stockDailyRecords’ will be deleted from database as well. How about if you just want to delete two referenced ‘stockDailyRecords’ records? This is called orphan delete, see example…

**1. No delete-orphan cascade**

You need to delete the ‘stockDailyRecords’ one by one.

StockDailyRecord sdr1 = (StockDailyRecord)session.get(StockDailyRecord.**class**,

**new** Integer(56));

StockDailyRecord sdr2 = (StockDailyRecord)session.get(StockDailyRecord.**class**,

**new** Integer(57));

session.delete(sdr1);

session.delete(sdr2);

Hibernate: delete from stock\_daily\_record where DAILY\_RECORD\_ID=?

Hibernate: delete from stock\_daily\_record where DAILY\_RECORD\_ID=?

## 2. With delete-orphan cascade

The **cascade=”delete-orphan”** is declared in ‘stockDailyRecords’ to enable the delete orphan cascade effect. When you save or update the Stock, it will remove those ‘stockDailyRecords’ which already mark as removed.

<!-- Stock.hbm.xml -->

<set name=*"stockDailyRecords"* cascade=*"delete-orphan"* table=*"stock\_daily\_record"*>

<key>

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

</key>

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

</set>

StockDailyRecord sdr1 = (StockDailyRecord)session.get(StockDailyRecord.**class**,

**new** Integer(56));

StockDailyRecord sdr2 = (StockDailyRecord)session.get(StockDailyRecord.**class**,

**new** Integer(57));

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

stock.getStockDailyRecords().remove(sdr1);

stock.getStockDailyRecords().remove(sdr2);

session.saveOrUpdate(stock);

Hibernate: delete from stock\_daily\_record where DAILY\_RECORD\_ID=?

Hibernate: delete from stock\_daily\_record where DAILY\_RECORD\_ID=?

In short, delete-orphan allow parent table to delete few records (delete orphan) in its child table.

**How to enable cascade ?**

The cascade is supported in both XML mapping file and annotation.

**1. XML mapping file**

In XML mapping file, declared the **cascade** keyword in your relationship variable.

<!-- Stock.hbm.xml -->

<set name=*"stockDailyRecords"* cascade=*"save-update, delete"* table=*"stock\_daily\_record"*>

<key>

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

</key>

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

</set>

**2. Annotation**

In annotation, declared the **CascadeType.SAVE\_UPDATE** (save, update) and **CascadeType.REMOVE** (delete) in @Cascade annotation.

//Stock.java

@OneToMany(mappedBy = "stock")

@Cascade({CascadeType.SAVE\_UPDATE, CascadeType.DELETE})

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

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

}