Fpt University

FPT POLYTECHNIC

LẬP TRÌNH JAVA 4

BÀI 8: ONE TO MANY, MANY TO ONE, HIBERNATE QUERY LANGUAGE

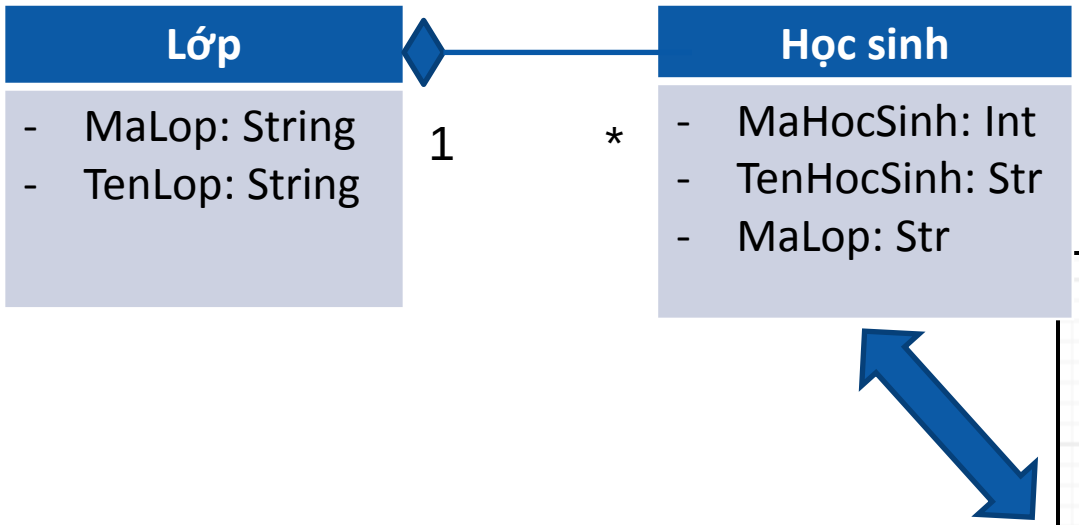
PHẦN 1

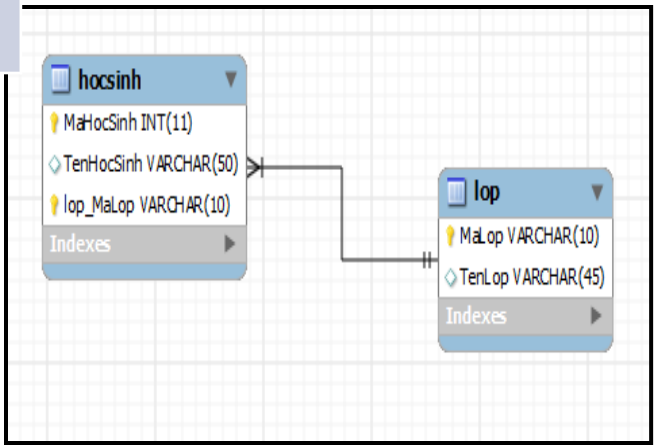
www.poly.edu.vn

Mục tiêu

- Kết thúc bài học này bạn có khả năng
  - Many to one
  - One to many
  - Hibernate Query Language

Mapping Many To One





- Một học sinh thuộc về 1 lớp.
- Một lớp có nhiều học sinh.

LopPoJo

```
1  package pojo;
2
3  public class LopPojo implements java.io.Serializable {
4      private String maLop;
5      private String tenLop;
6  }
   //Các phương thức set, get, constructor
```

Lop.hbm.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE hibernate-mapping PUBLIC
3  "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4  "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
5  <hibernate-mapping>
6      <class name="pojo.LopPojo" table="lop">
7          <id name="maLop" type="string">
8              <column name="MaLop" length="10"/>
9              <generator class="assigned"/>
10         </id>
11         <property name="tenLop" type="string">
12             <column name="TenLop" length="45" />
13         </property>
14     </class>
15 </hibernate-mapping>
```

HocSinhPOJO

```
1  package pojo;
2
3  public class HocSinhPojo implements java.io.Serializable {
4      private int maHocSinh;
5      private String tenHocSinh;
6      private LopPojo lop;

       //Các phương thức get, set, constructor.

   }
```

HocSinh.hbm.xml

```
1  <hibernate-mapping>
2      <class name="pojo.HocSinhPojo" table="hocsinh">
3          <id name="maHocSinh" column="MaHocSinh" type="integer">
4              <generator class="assigned"/>
5          </id>
6          <property name="tenHocSinh" column="TenHocSinh"
7  type="string"/>
8          <many-to-one name="lop" class="pojo.LopPojo"  >
9              <column name="MaLop" />
10         </many-to-one>
11         </class>
12 </hibernate-mapping>
```

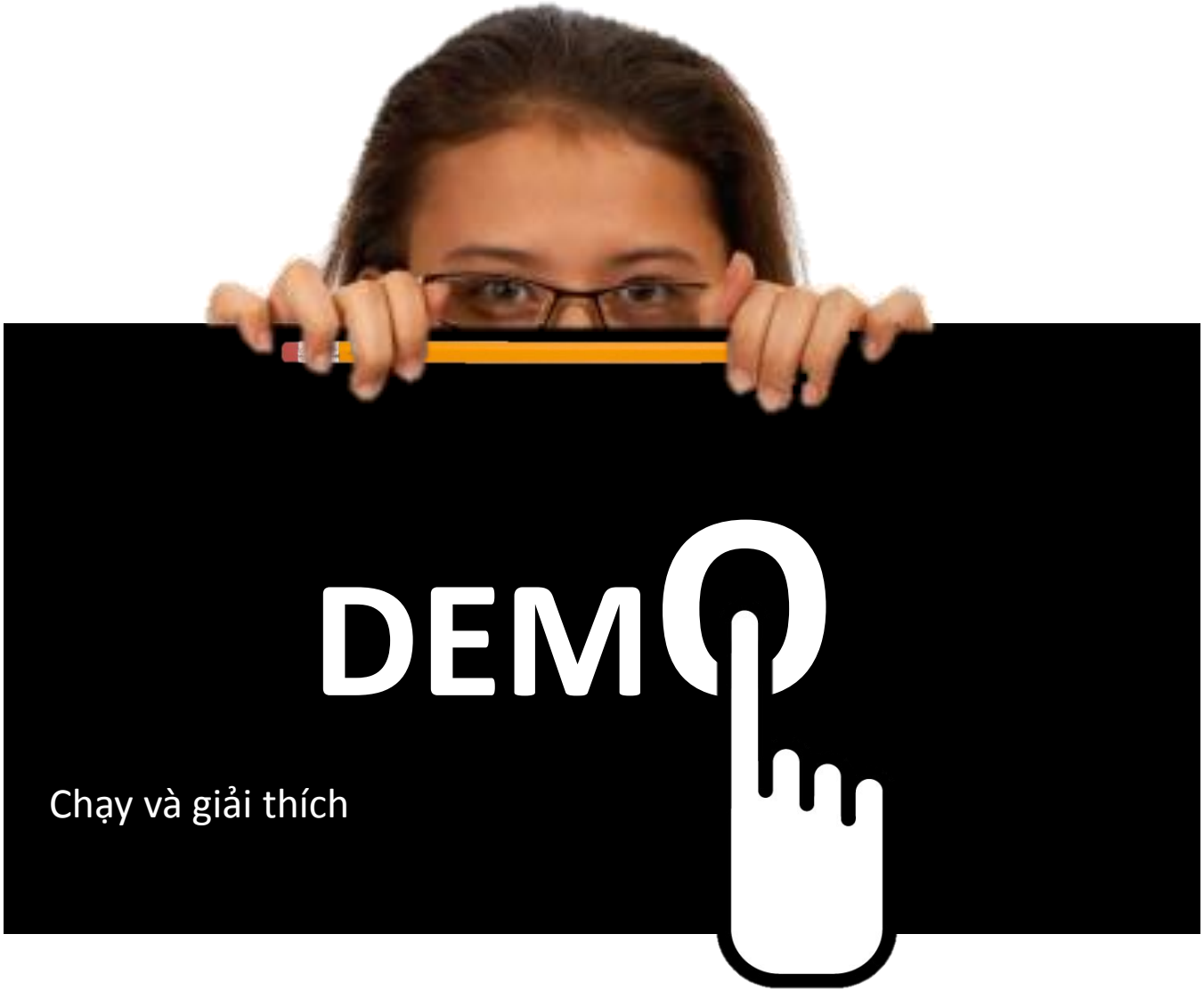
```
<many-to-one
        name="lop” ➔ tên thuộc tính cần mapping
        class="pojo.LopPojo"  > ➔ Tên lớp cần mapping tới
        <column name="MaLop" /> ➔ Tên cột trong table HocSinh
</many-to-one>
```

Lấy thông tin học sinh

```
1   public class Main {
2       public static void main(String[] args) {
3           HocSinhPojo hs = null;
4           SessionFactory ssFac = MyHibernateUtil.getSessionFactory();
5           Session ss = ssFac.openSession();
6           ss.getTransaction().begin();
7           try {
8               hs = (HocSinhPojo)ss.get(HocSinhPojo.class, 1);
9               System.out.println("Tên học sinh: " + hs.getTenHocSinh());
10              System.out.println("Mã lớp: " + hs.getLop().getMaLop());
11              System.out.println("Tên lớp: " + hs.getLop().getTenLop());
12          } catch (HibernateException ex ) {
13              System.out.println(ex.getMessage());
14          }
15          finally
16          {
17              ss.close();
18          }
19      }
20  }
```

Lấy thông tin học sinh khi còn mở Session

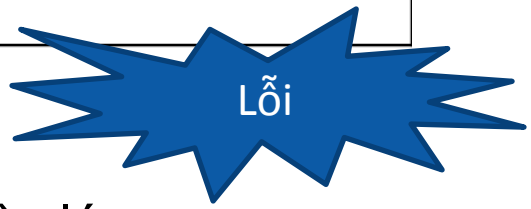

```
Tên học sinh: Hồ Văn Tấn
Mã lớp: 10A
Tên lớp: Lớp 10 ban A
BUILD SUCCESSFUL (total time: 2 seconds)
```



Chạy và giải thích

Lấy thông tin học sinh

```
1   public class Main {
2       public static void main(String[] args) {
3           HocSinhPojo hs = null;
4           SessionFactory ssFac = MyHibernateUtil.getSessionFactory();
5           Session ss = ssFac.openSession();
6           ss.getTransaction().begin();
7           try {
8               hs = (HocSinhPojo)ss.get(HocSinhPojo.class, 1);
9           } catch (HibernateException ex ) {
10              System.out.println(ex.getMessage());
11          }
12          finally
13          {
14              ss.close();
15          }
16          System.out.println("Tên học sinh: " + hs.getTenHocSinh());
17          System.out.println("Mã lớp: " + hs.getLop().getMaLop());
18          System.out.println("Tên lớp: " + hs.getLop().getTenLop());
19      }
20  }
```



Lấy thông tin học sinh sau khi đóng Session
→ chỉ lấy được tên và mã học sinh, không lấy được tên lớp.

Lấy thông tin học sinh

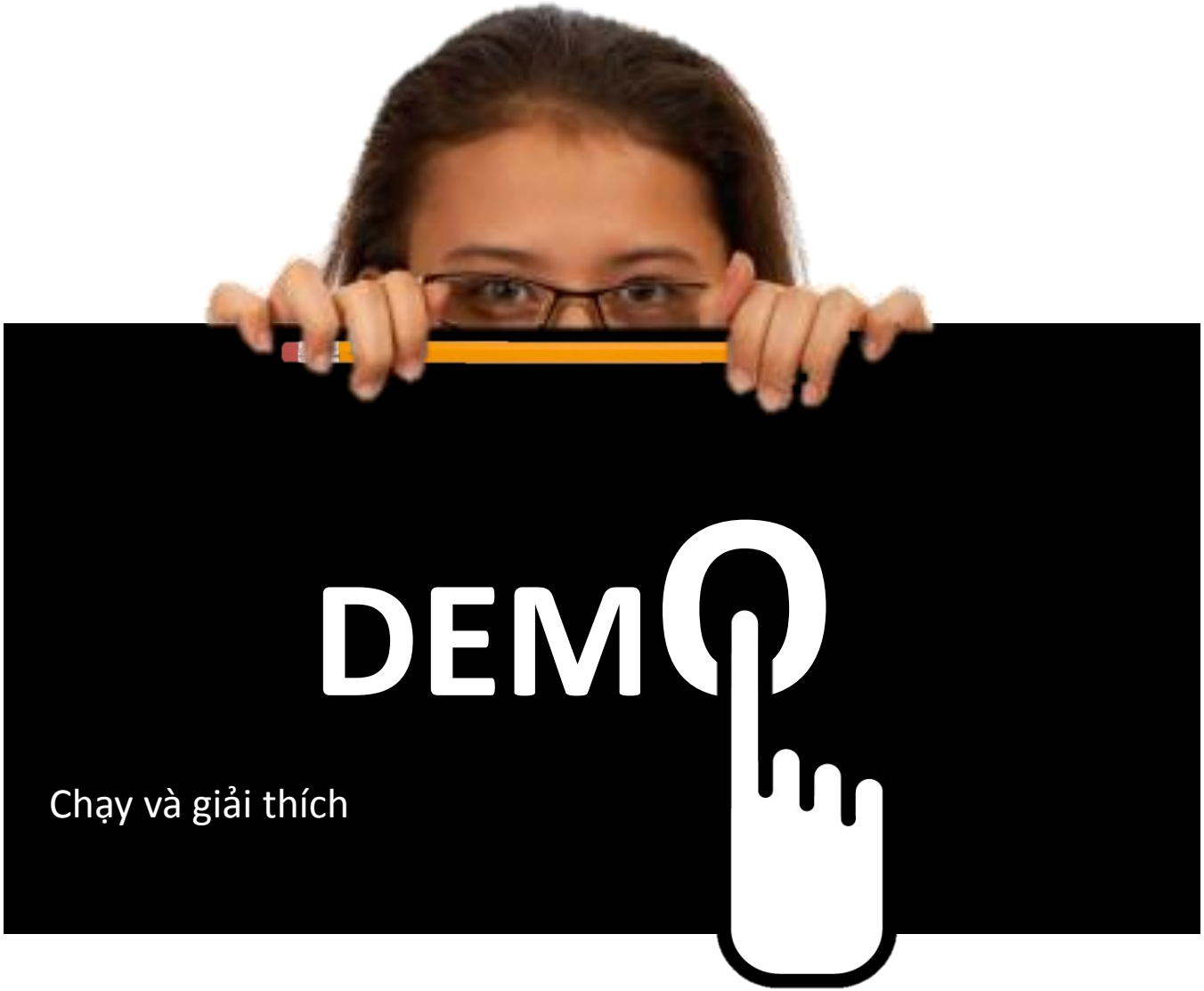
```
Tên học sinh: Hồ Văn Tấn
Mã lớp: 10A
Exception in thread "main" org.hibernate.LazyInitializationException: could not initialize proxy - no Session
        at org.hibernate.proxy.AbstractLazyInitializer.initialize(AbstractLazyInitializer.java:149)
        at org.hibernate.proxy.AbstractLazyInitializer.getImplementation(AbstractLazyInitializer.java:195)
        at org.hibernate.proxy.pojo.javassist.JavassistLazyInitializer.invoke(JavassistLazyInitializer.java:180)
        at pojo.LopPojo_$$_javassist_2.getTenLop(LopPojo_$$_javassist_2.java)
        at qlhs.Main.main(Main.java:31)
Java Result: 1
BUILD SUCCESSFUL (total time: 2 seconds)
```

Lấy thông tin học sinh sau khi đóng Session
→ chỉ lấy được tên và mã học sinh, không lấy được tên lớp.

Lấy thông tin học sinh

```
1   public class Main {
2       public static void main(String[] args) {
3           HocSinhPojo hs = null;
4           SessionFactory ssFac = MyHibernateUtil.getSessionFactory();
5           Session ss = ssFac.openSession();
6           ss.getTransaction().begin();
7           try {
8               hs = (HocSinhPojo)ss.get(HocSinhPojo.class, 1);
9               System.out.println("Tên lớp: " + hs.getLop().getTenLop());
10          } catch (HibernateException ex ) {
11              System.out.println(ex.getMessage());
12          }
13          finally
14          {
15              ss.close();
16          }
17          System.out.println("Tên học sinh: " + hs.getTenHocSinh());
18          System.out.println("Mã lớp: " + hs.getLop().getMaLop());
19      }
20  }
```

```
Tên lớp: Lớp 10 ban A
Tên học sinh: Hồ Văn Tấn
Mã lớp: 10A
BUILD SUCCESSFUL (total time: 2 seconds)
```



Chạy và giải thích

Lấy thông tin học sinh

- Nguyên nhân lỗi:
  - Cơ chế Lazy Initialization đang được bật (= true)

→Truy vấn đối tượng HocSinh sẽ không kèm theo truy vấn đối tượng Lop. (chỉ có thể truy vấn được mã lớp mà không truy vấn được tên lớp).

→ Truy vấn đối tượng cha sẽ không kèm theo truy vấn đối tượng con.

MANY TO ONE

- Lazy Initialization & fetch
- Trong Hibernate, Lazy Initialization giúp
  - Tránh các câu truy vấn cơ sở dữ liệu không cần thiết
  - Gia tăng hiệu suất thực thi
  - Lazy mặc định có giá trị là true

Cách 1

- Sau khi có mã lớp, ta dùng làm lấy thông tin lớp theo mã lớp

```
LopDAO.layThongTinLop(int maLop);
```

- Cách 2
  - **Khai báo lazy = false trong Hocsinh.hbm.xml**
- Lazy ="false" truy vấn lớp cha kèm theo truy vấn lớp con.

```
1   <hibernate-mapping>
2     <class name="pojo.HocSinhPojo" table="hocsinh">
3       <id name="maHocSinh" column="MaHocSinh" type="integer">
4         <generator class="assigned"/>
5       </id>
6       <property name="tenHocSinh" column="TenHocSinh" type="string"/>
7       <many-to-one name="lop" class="pojo.LopPojo" lazy="false"  >
8         <column name="MaLop" />
9       </many-to-one>
10      </class>
11  </hibernate-mapping>
```

Cơ chế fetch

- Fetch = "select" sử dụng select để truy vấn lớp con. → sử dụng 2 câu truy vấn select để truy vấn cả lớp cha và con, cách này không hiệu quả vì phải truy xuất tới cơ sở dữ liệu 2 lần.
- Fetch = "join" sử dụng phép kết để gọp truy vấn lớp cha và lớp con trong 1 truy vấn. → hiệu suất cao hơn, sử dụng 1 câu truy vấn.

Cơ chế fetch – sử dụng select

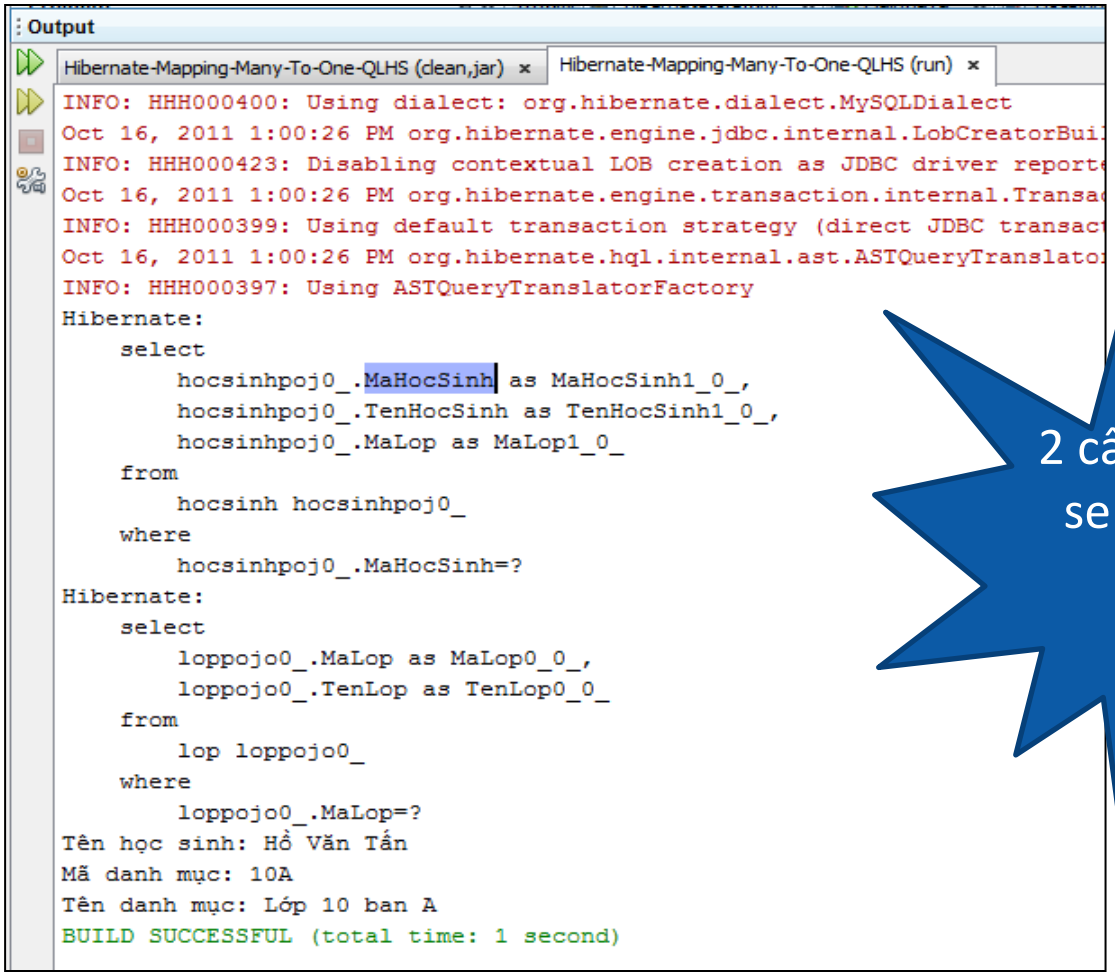
- Hocsinh.hbm.xml

```
1   <hibernate-mapping>
2     <class name="pojo.HocSinhPojo" table="hocsinh">
3       <id name="maHocSinh" column="MaHocSinh" type="integer">
4         <generator class="assigned"/>
5       </id>
6       <property name="tenHocSinh" column="TenHocSinh" type="string"/>
7       <many-to-one name="lop" class="pojo.LopPojo" lazy="false"  fetch="select">
8         <column name="MaLop" />
9       </many-to-one>
10      </class>
11  </hibernate-mapping>
```

Chú ý: mỗi khi sữa lại file cấu hình xml (cấu hình hibernate, cấu hình mapping, …) Phải clean and built lại project thì thay đổi mới có hiệu lực.

❑ Cơ chế fetch – sử dụng select

Bản chất, các câu truy vấn HQL đều được chuyển về SQL, như hình dưới có 2 câu select được gọi. ⇔ truy xuất CSDL 2 lần



❑ Cơ chế fetch – sử dụng join

Hocsinh.hbm.xml

```
1   <hibernate-mapping>
2     <class name="pojo.HocSinhPojo" table="hocsinh">
3       <id name="maHocSinh" column="MaHocSinh" type="integer">
4         <generator class="assigned"/>
5       </id>
6       <property name="tenHocSinh" column="TenHocSinh" type="string"/>
7       <many-to-one name="lop" class="pojo.LopPojo" lazy="false"  fetch="join">
8         <column name="MaLop" />
9       </many-to-one>
10      </class>
11  </hibernate-mapping>
```

Chú ý: mỗi khi sữa lại file cấu hình xml (cấu hình hibernate, cấu hình mapping, …)
Phải clean and built lại project thì thay đổi mới có hiệu lực.

❑ Cơ chế fetch – sử dụng join

```
1   public class Main {
2       public static void main(String[] args) {
3           HocSinhPojo hs = null;
4           SessionFactory ssFac = MyHibernateUtil.getSessionFactory();
5           Session ss = ssFac.openSession();
6           ss.getTransaction().begin();
7           try {
8              String hql="select hs from HocSinhPojo hs left join fetch hs.lop
9                                  where hs.maHocSinh=:maHocSinh";
10              Query query=ss.createQuery(hql);
11              query.setInteger("maHocSinh", 1);
12              hs = (HocSinhPojo) query.uniqueResult();
13          } catch (HibernateException ex) {
14              System.out.println(ex.getMessage());
15          } finally {
16              ss.close();
17          }
18          System.out.println("Tên hoccsinh: " + hs.getTenHocSinh());
19          System.out.println("Mã danh mục: " + hs.getLop().getMaLop());
20          System.out.println("Tên danh mục: " + hs.getLop().getTenLop());
21      }
22  }
23
```

❑ Cơ chế fetch – sử dụng join

Có 1 câu select được gọi, có sử dụng phép Join ⇔ truy xuất CSDL 2 lần

```
Output - Hibernate-Mapping-Many-To-One-QLHS (run)
INFO: HHH000397: Using ASTQueryTranslatorFactory
Hibernate:
    select
        hocsinhpoj0_.MaHocSinh as MaHocSinh1_0_,
        loppojo1_.MaLop as MaLop0_1_,
        hocsinhpoj0_.TenHocSinh as TenHocSinh1_0_,
        hocsinhpoj0_.MaLop as MaLop1_0_,
        loppojo1_.TenLop as TenLop0_1_
    from
        hocsinh hocsinhpoj0_
    left outer join
        lop loppojo1_
            on hocsinhpoj0_.MaLop=loppojo1_.MaLop
    where
        hocsinhpoj0_.MaHocSinh=?
Tên học sinh: Hồ Văn Tấn
Mã danh mục: 10A
Tên danh mục: Lớp 10 ban A
BUILD SUCCESSFUL (total time: 2 seconds)
```

1 câu truy vấn select được gọi.

Cascade

- Save – update
- Delete

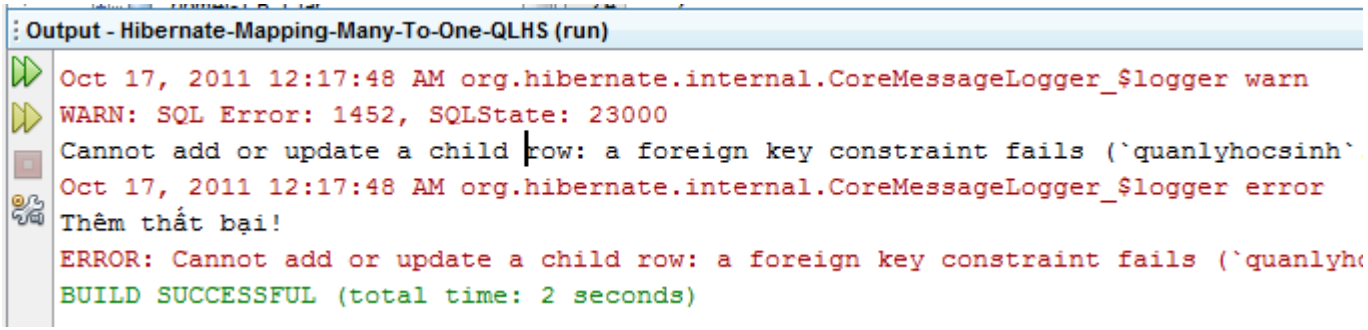
❑ Cascade – None cascade

```
1   <hibernate-mapping>
2       <class name="pojo.HocSinhPojo" table="hocsinh">
3           <id name="maHocSinh" column="MaHocSinh" type="integer">
4               <generator class="assigned"/>
5           </id>
6           <property name="tenHocSinh" column="TenHocSinh" type="string"/>
7           <many-to-one name="lop" class="pojo.LopPojo"
8                        lazy="false" fetch="join" cascade="none">
9               <column name="MaLop" />
10          </many-to-one>
11          </class>
    </hibernate-mapping>
```

Mặc định nếu không khai báo thì cascade=none

❑ Cascade – không dùng update-save

```
1   public static void main(String[] args) {
2           HocSinhPojo hs = new HocSinhPojo(15, "Trần Văn Đạt", null);
3           LopPojo lop = new LopPojo("12E", "Lớp 12 chuyên Hóa");
4
5           hs.setLop(lop);
6           if(HocSinhDAO.themHocSinh(hs))
7           {
8               System.out.println("Thêm thành công!");
9           }
10          else
11              System.out.println("Thêm th?t b?i!");
        }
```



Lớp 12E không tồn tại trong cơ sở dữ liệu.

❑ Cascade – sử dụng update-save

HocSinh.hbm.xml

```
1  <hibernate-mapping>
2      <class name="pojo.HocSinhPojo" table="hocsinh">
3          <id name="maHocSinh" column="MaHocSinh" type="integer">
4              <generator class="assigned"/>
5          </id>
6          <property name="tenHocSinh" column="TenHocSinh" type="string"/>
7          <many-to-one name="lop" class="pojo.LopPojo"
8                       lazy="false" fetch="join" cascade="save-update">
9              <column name="MaLop" />
10         </many-to-one>
11         </class>
   </hibernate-mapping>
```

❑ Cascade – sử dụng update-save

```
1   public static void main(String[] args) {
2           HocSinhPojo hs = new HocSinhPojo(15, "Trần Văn Đạt", null);
3           LopPojo lop = new LopPojo("12E", "Lớp 12 chuyên Hóa");
4
5           hs.setLop(lop);
6           if(HocSinhDAO.themHocSinh(hs))
7           {
8               System.out.println("Thêm thành công!");
9           }
10          else
11              System.out.println("Thêm thất bại!");
        }
```

MaLop	TenLop
10A	Lớp 10 ban A
10B	Lớp 10 ban B
10C	Lớp 10 ban C
10D	Lớp 10 ban D
11A	Lớp 11 ban A
11B	Lớp 11 ban B
12A	Lớp 12 ban A
12B	Lớp 12 ban B
12D	Lớp 12 ban D
12E	Lớp 12 chuyên Hóa
NULL	NULL

MaHocSinh	TenHocSinh	MaLop
1	Hồ Văn Tấn	10A
2	Sử Bá Thuần	10B
3	Văn Kinh Luân	10A
4	Lương Quang Hà	10C
5	Lý Thị Loan	10D
15	Trần Văn Đạt	12E
NULL	NULL	NULL

❑ Cascade – không dùng delete

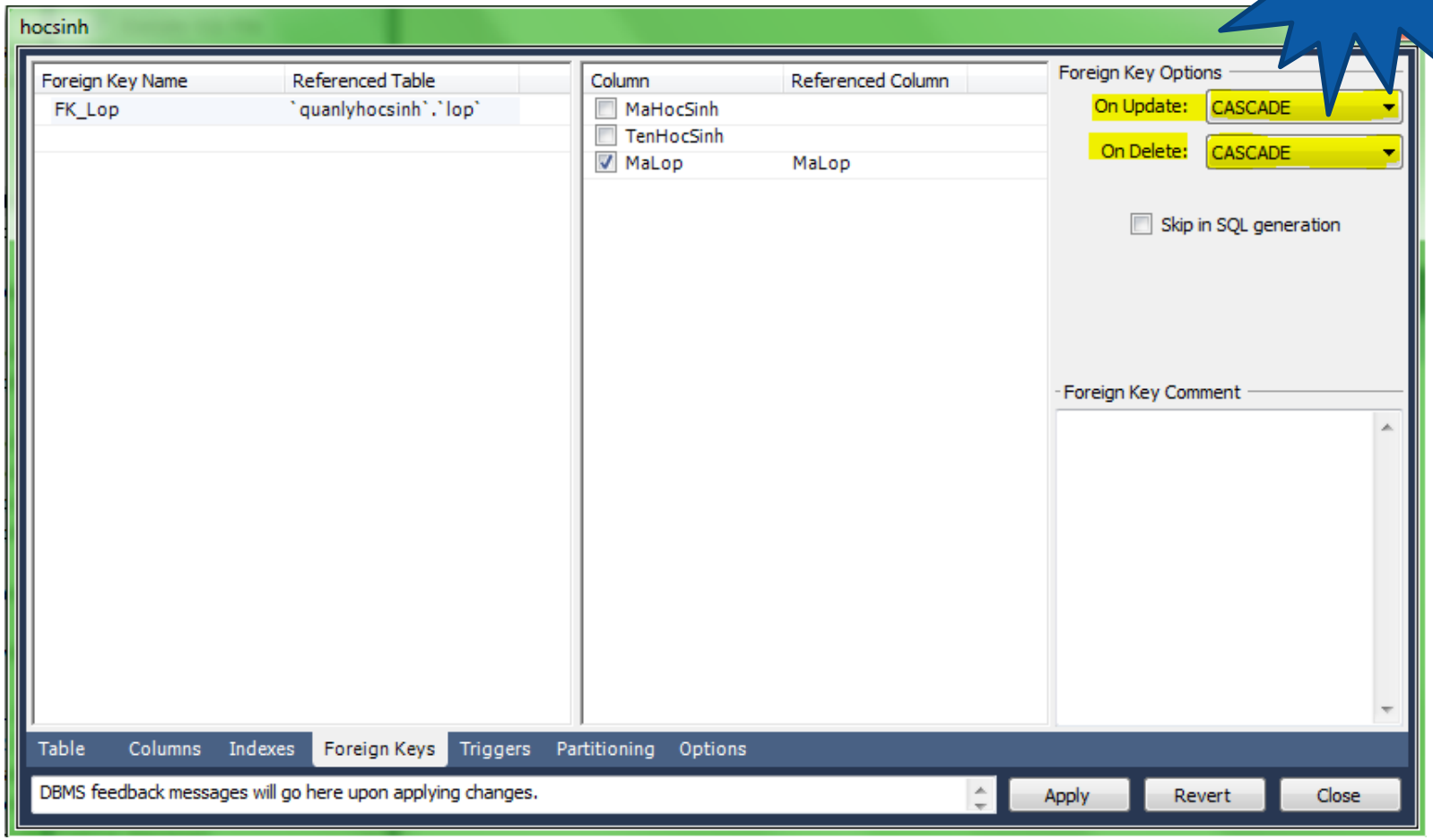
```
1  public class Main {
2      public static void main(String[] args) {
3          if(LopDAO.xoaLop("12E"))
4              System.out.println("Xóa thành công!");
5          else
6              System.out.println("Xóa th?t b?i!");
7      }
8  }
```

```
Output - Hibernate-Mapping-Many-To-One-QLHS (run)
ERROR: Cannot delete or update a parent row: a foreign key constraint fails (`quanlyhoc
Oct 17, 2011 12:54:22 AM org.hibernate.engine.jdbc.batch.internal.AbstractBatchImpl
org.hibernate.exception.ConstraintViolationException: Cannot delete or update a parent
INFO: HHH000010: On release of batch it still contained JDBC statements
BUILD SUCCESSFUL (total time: 2 seconds)
```

LỖI

Không thể xóa lớp 12E→ lỗi tham chiếu khóa ngoại

❑ Cascade – sử dụng delete



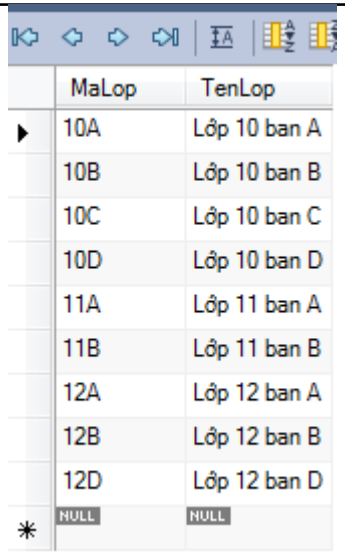
Để dùng Cascade trong hibernate phải cho phép sử dụng cascade trong CSDL

❑ Cascade – sử dụng delete

```
<hibernate-mapping>
    <class name="pojo.HocSinhPojo" table="hocsinh">
        <id name="maHocSinh" column="MaHocSinh" type="integer">
            <generator class="assigned"/>
        </id>
        <property name="tenHocSinh" column="TenHocSinh" type="string"/>
        <many-to-one name="lop" class="pojo.LopPojo"
                     lazy="false" fetch="join" cascade="save-update, delete">
            <column name="MaLop" />
        </many-to-one>
        </class>
</hibernate-mapping>
```

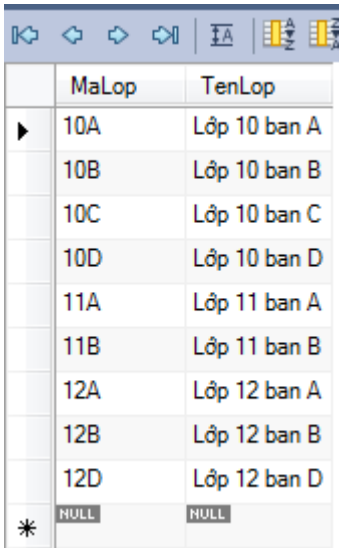
❑ Cascade – sử dụng delete

```
1  public class Main {
2      public static void main(String[] args) {
3          if(LopDAO.xoaLop("12E"))
4              System.out.println("Xóa thành công!");
5          else
6              System.out.println("Xóa thất bại!");
7      }
8  }
```



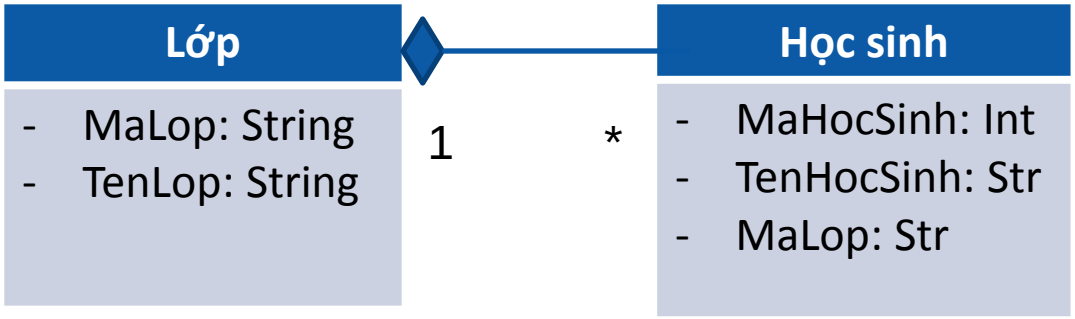
MaLop	TenLop
10A	Lớp 10 ban A
10B	Lớp 10 ban B
10C	Lớp 10 ban C
10D	Lớp 10 ban D
11A	Lớp 11 ban A
11B	Lớp 11 ban B
12A	Lớp 12 ban A
12B	Lớp 12 ban B
12D	Lớp 12 ban D
NULL	NULL

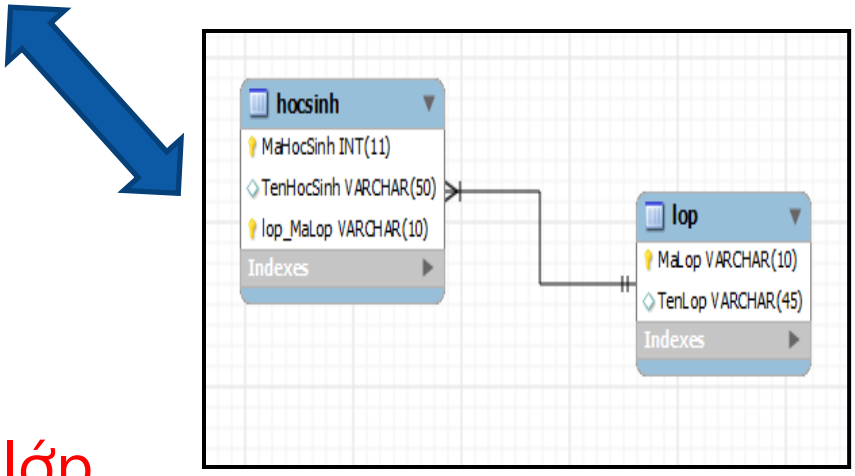

```
Xóa thành công!
BUILD SUCCESSFUL (total time: 1 second)
```



MaLop	TenLop
10A	Lớp 10 ban A
10B	Lớp 10 ban B
10C	Lớp 10 ban C
10D	Lớp 10 ban D
11A	Lớp 11 ban A
11B	Lớp 11 ban B
12A	Lớp 12 ban A
12B	Lớp 12 ban B
12D	Lớp 12 ban D
NULL	NULL

ONE TO MANY





- Một học sinh thuộc về 1 lớp.
- Một lớp có nhiều học sinh.

❑ LopPOJO

```
1   package pojo;
2
3   import java.util.HashSet;
4   import java.util.Set;
5
6   public class LopPojo implements java.io.Serializable {
7       private String maLop;
8       private String tenLop;
9       private Set<HocSinhPojo> danhSachHocSinh = new HashSet<HocSinhPojo>(0);
10
11      // Các phương thức get, set, construction
12
13  }
```

Lop.hbm.xml

```
1   <hibernate-mapping>
2       <class name="pojo.LopPojo" table="lop">
3           <id name="maLop" type="string">
4               <column name="MaLop" length="10"/>
5               <generator class="assigned"/>
6           </id>
7           <property name="tenLop" type="string">
8               <column name="TenLop" length="45" />
9           </property>
10          <set name="danhSachHocSinh" lazy="false" fetch="select">
11              <key>
12                  <column name="MaLop"/>
13              </key>
14              <one-to-many class="pojo.HocSinhPojo" />
15          </set>
        </class>
    </hibernate-mapping>
```

HocSinhPOJO

```
1  package pojo;
2
3  public class HocSinhPojo implements java.io.Serializable {
4      private int maHocSinh;
5      private String tenHocSinh;
6      private LopPojo lop;

       //Các phương thức get, set, constructor.

   }
```

HocSinh.hbm.xml

```
1   <hibernate-mapping>
2       <class name="pojo.HocSinhPojo" table="hocsinh">
3           <id name="maHocSinh" column="MaHocSinh" type="integer">
4               <generator class="assigned"/>
5           </id>
6           <property name="tenHocSinh" column="TenHocSinh" type="string"/>
7           <many-to-one name="lop" class="pojo.LopPojo"  >
8               <column name="MaLop" />
9           </many-to-one>
10          </class>
11  </hibernate-mapping>
12
```

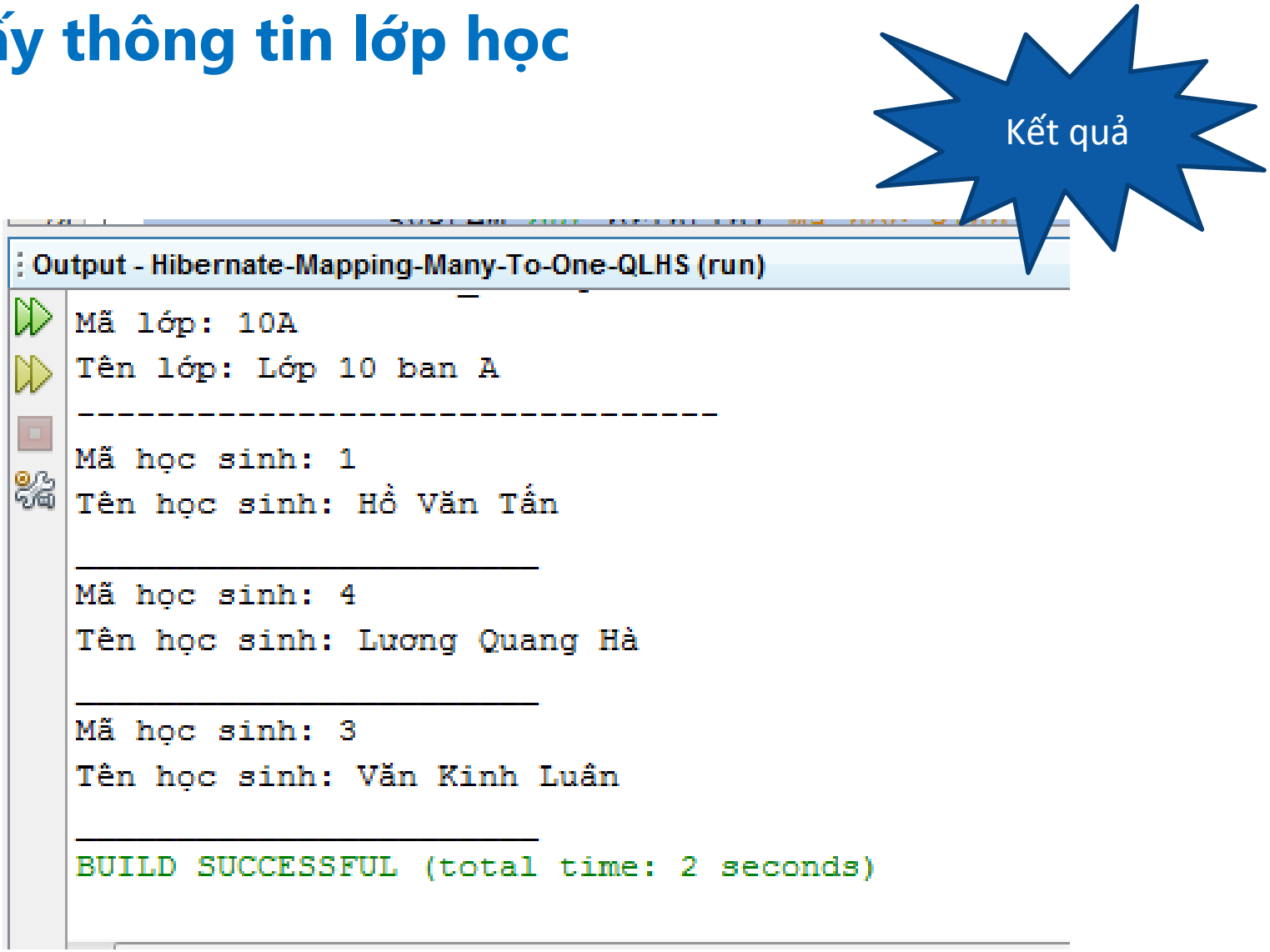
❑ DAO: Lấy thông tin lớp học

```
1   public static LopPojo layThongTinLop(String maLop) {
2           LopPojo lop = null;
3           SessionFactory ssFac = MyHibernateUtil.getSessionFactory();
4           Session ss = ssFac.getCurrentSession();
5           Transaction trans = ss.getTransaction();
6           trans.begin();
7           try {
8               lop = (LopPojo)ss.get(LopPojo.class, maLop);
9               trans.commit();
10          } catch (HibernateException ex ) {
11              System.out.println(ex.getMessage());
12          }
13          return lop;
14      }
```

❑ Lấy thông tin lớp học

```
1  public class Main {
2      public static void main(String[] args) {
3          LopPojo lop = LopDAO.layThongTinLop("10A");
4          System.out.println("Mã lớp: " + lop.getMaLop());
5          System.out.println("Tên lớp: " + lop.getTenLop());
6          System.out.println("-------------------------------");
7          Iterator<HocSinhPojo> dsHocSinh =
8                                  lop.getDanhSachHocSinh().iterator();
9
10         while(dsHocSinh.hasNext())
11         {
12             HocSinhPojo hs = dsHocSinh.next();
13             System.out.println("Mã học sinh: " + hs.getMaHocSinh());
14             System.out.println("Tên học sinh: " +
15                                                 hs.getTenHocSinh());
16             System.out.println("_________________________");
17         }
18     }
19 }
20
```

❑ Lấy thông tin lớp học



Fpt University

FPT POLYTECHNIC

LẬP TRÌNH JAVA 4

BÀI 8: ONE TO MANY, MANY TO ONE, HIBERNATE QUERY LANGUAGE

PHẦN 2

www.poly.edu.vn

HIBERNATE QUERY LANGUAGE

- Hibernate Query Language (HQL)
- HQL - from HQL - select
- HQL - aggregate function
- HQL - where
- HQL - Expression
- HQL - order by
- HQL - group by & having
- HQL - sub query

❑ JDBC - SQL

- JDBC sử dụng các câu lệnh SQL để truy vấn dữ liệu và các thao tác cập nhật như thêm, xóa, sửa trên bảng dữ liệu.
- Để thao tác tốt các câu lệnh SQL cần quan tâm đến các bảng, các dòng, các cột và mối quan hệ giữa các bảng và đặc biệt là hệ quản trị cơ sở dữ liệu đang làm việc.
- Kết quả trả về của câu lệnh truy vấn là danh sách các dòng dữ liệu.

Hibernate - HQL

- Hibernate cung cấp các API cho phép thực hiện tác thao tác cập nhật như thêm, xóa, sửa.
- Hibernate cung cấp ngôn ngữ truy vấn rất mạnh được gọi là Hibernate Query Language (HQL).
- HQL độc lập hệ quản trị cơ sở dữ liệu và được Hibernate thông dịch sang SQL tương ứng trong quá trình thực thi.
- HQL là ngôn ngữ truy vấn theo hướng đối tượng. Kết quả truy vấn là đối tượng
- Hibernate sử dụng các lớp đối tượng và các thuộc tính thay cho các bảng và các cột.

HQL - Phân biệt hoa thường

- HQL không phân biệt thường hoa ngoại trừ
  - Tên các lớp đối tượng
  - Các thuộc tính trong lớp đối tượng
- Ví dụ 2 câu truy vấn giống nhau
  - **Select s from Sach s <-> SELECT s FROM Sach s**
  - **Select s From Sach s <-> SELECT s FROM Sach s**
- Ví dụ 2 câu truy vấn khác nhau
  - **select s from sach s <-> SELECT s FROM Sach s**
  - **Select s From SACH s <-> SELECT s FROM Sach s**

HQL - Mệnh đề from

- Lấy tất cả cá đối tượng danh mục

```
-from DanhMuc
-select dm from DanhMuc dm
-select dm from DanhMuc as dm
```

- Lấy tất cả đối tượng sách

```
-from Sach
-select s from Sach s
-select s from Sach as s
```

HQL - Mệnh đề from - lấy tất cả đối tượng

```
String hql = "from Sach";
Query query = session.createQuery(hql);
List<Sach> ds=query.list();
```

HQL - Mệnh đề from - phân trang

```
1 String hql = "from Sach";
2 Query query = session.createQuery(hql);
3 query.setFirstResult(3);
4 query.setMaxResults(5);
5 List<Sach> ds=query.list();
```

- Lấy từ vị trí thứ n (tính từ 0) setFirstResult (int n)
- lấy tối đa m đối tượng setMaxResults (int m)
- Tương tự trong MYSQL LIMIT N, M

HQL - select

```
1  String hql = "select s.danhMuc from Sach s";
2  Query query = session.createQuery(hql);
3  List<DanhMuc> ds=query.list();
```

```
1  String hql = "select s.danhMuc.tenDanhMuc from Sach
2  s"; Query query = session.createQuery(hql);
3  List<String> ds=query.list();
```

HQL - select

```
1  String h ="select s.maSach, s.tenSach, s.danhMuc from Sach s";
2  Query query = session.createQuery(h);
3  List<Object[]> ds=query.list();
4  for(int i=0; i<ds.size(); i++){
5       Object[] objs=ds.get(i);
6       String maSach=(String)objs[0];
7       String tenSach=(String)objs[1];
8       DanhMuc dm=(DanhMuc)objs[2];
9  }
```

```
1  String hql ="select distinct s.danhMuc.tenDanhMuc from
2  Sach s";
3  Query query = session.createQuery(hql);
4  List<String> ds=query.list();
```

HQL - select

```
1   package pojo;
2
3   public class MyClass {
4       private String maSach; private String
5       tenSach; private DanhMuc danhMuc;
6
7       public MyClass() {
8       }
9
10      public MyClass(String maSach, String tenSach,
11                                          DanhMuc danhMuc) {
12          this.maSach = maSach;
13          this.tenSach =
14          tenSach; this.danhMuc
15          = danhMuc;
16      }
17      //Getters & Setters
18  }
```

HIBERNATE QUERY LANGUAGE

HQL - select

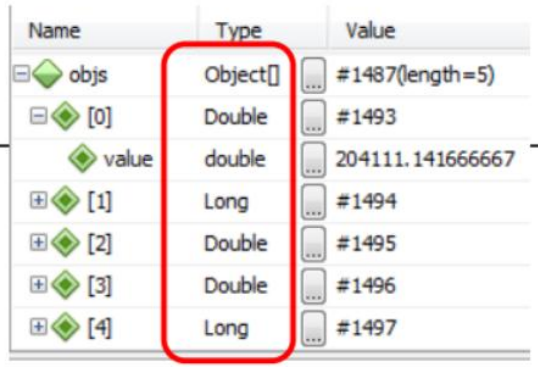
```
1  String hql =
2      "select new pojo.MyClass (s.maSach, s.tenSach, s.danhMuc)
3       from Sach s";
4  Query query = session.createQuery(hql);
5  List<MyClass> ds=query.list();
6  for(int i=0; i<ds.size(); i++){
7      MyClass my=ds.get(i);
8      String maSach=my.getMaSach();
9      String tenSach=my.getTenSach();
       DanhMuc dm=my.getDanhMuc();
   }
```

HQL - Aggregate functions

- avg
- min
- max
- count
- sum

HQL - Aggregate functions

```
1  String hql =          " select avg(s.giaBan) as GiaTrungBinh,"
2                      + " count(*) as SoLuongDauSach,"
3                      + " min(s.giaBan) as GiaBanThapNhat,"
4                      + " max(s.giaBan) as GiaBanCaoNhat, "
5                      + " sum(s.soLuong) as TongSoLuongSach "
6                      + " from Sach s";
7  Query query = session.createQuery(hql);
8  Object[] objs = (Object[]) query.uniqueResult();
9  double giaTrungBinh=(Double)objs[0];
10 long soLuongDauSach=(Long)objs[1];
11 double giaBanThapNhat=(Double)objs[2];
12 double giaBanCaoNhat=(Double)objs[3];
13 long tongSoLuongSach=(Long)objs[4];
```



HQL - where

```
1  String tenSach="Java";
2  String hql = "from Sach s where s.tenSach like :tenSach";
3  Query query = session.createQuery(hql);
4  query.setString("tenSach", "%"+tenSach+"%");
5  ArrayList<Sach> ds=query.list();
```

```
1  String tenDanhMuc="Java";
2  String giaBan=20000;
3  String hql="from Sach s
4              where s.danhMuc.tenDanhMuc=:tenDanhMuc
5                    and s.giaBan>:giaBan"
6  Query query = session.createQuery(hql);
7  query.setString("tenDanhMuc", tenDanhMuc);
8  query.setString("giaBan", giaBan);
9  ArrayList<Sach> ds=query.list();
```

HQL – Expression

- + - * /
- =, >=, <=, <>, !=, like
- and, or, not
- ( )
- in, not in, between, is null, is not null, is empty, is not empty, member of, not member of
- case ... when ... then ... else ... end

HQL – Expression

- concat(...,...)
- current_date(),current_time(),current_timestamp()
- second(...), minute(...), hour(...) ,day(...), month(...), year(...)
- substring(), trim(), lower(), upper(), length(), locate(), abs(), sqrt(), bit_length(), mod()
- coalesce() , nullif()
- str(), ...

HQL - order by

```
1  String tenSach=M3ava";
2
3  String hql = ” from Sach s”;
4  hql=hql +” where s.tenSach like :tenSach”;
5  hql=hql +” order by s.tenSach desc”;
6
7  Query query = session.createQuery(hql);
8  query.setString(MtenSachM, ‘‘%"+tenSach+M%M);
9  ArrayList<Sach> ds = query.list();
```

HQL - group by & having

```
1    String  hql  =  “select s.danhMuc, sum(s.soLuong)";
2    hql  =  hql  +  “from Sach s group by s.danhMuc”;
3    hql = hql +     “having sum(s.soLuong)>100”;
4
5
6    Query  query  =  session.createQuery(hql);
7    ArrayList<Object[]> ds = query.list();
```

HQL - subquery

```
1  String hql  = " select s";
2  hql = hql   + " from Sach s";
3  hql = hql   + " where s.giaBan > ";
4  hql = hql   + "    (";
5  hql = hql   + "       select avg(s.giaBan)";
6  hql = hql   + "       from Sach s";
7  hql = hql   + "    )";
8  Query query = session.createQuery(hql);
9  ArrayList<Sach> ds = query.list();
```

TỔNG KẾT

- Many to one
- One to many
- Hibernate Query Language


TIN HỌC VĂN PHÒNG QUỐC TẾ THEO CHUẨN ICDL
Syllabus 5 - Office 2007
LUẬT KINH TẾ
LUẬT KINH TẾ
KỸ THUẬT SOẠN THẢO VÀ TRÌNH BÀY VĂN BẢN
TIN HỌC VĂN PHÒNG QUỐC TẾ THEO CHUẨN ICDL
Syllabus 5 - Office 2007
Cảm ơn