Web-based programming with Java

HIBERNATE

ThS. Nguyễn Nghiệm
0913.745.789 - NghiemN@fpt.edu.vn

NỘI DUNG

- Giới Thiệu
- Ánh xạ
  - Tập tin cấu hình
  - Xây dựng lớp thực thể
- Hibernate API
  - Truy vấn
    - Truy vấn thực thể
    - Truy vấn một thực thể theo khóa chính
  - Thao tác dữ liệu
    - Thêm mới
    - Cập nhật
    - Xóa
- Ngôn ngữ HQL

GIỚI THIỆU

- Hibernate là framework làm việc với cơ sở dữ liệu thuộc loại tốt nhất hiện nay.
- Hibernate ánh xạ các lớp Java vào các bảng của CSDL quan hệ bằng XML hoặc Annotation
- Hibernate là tầng trung gian giữa các đối tượng và CSDL để điều khiển các công việc quản lý lưu trữ trạng thái của các đối tượng đó dựa trên cơ sở ánh xạ.

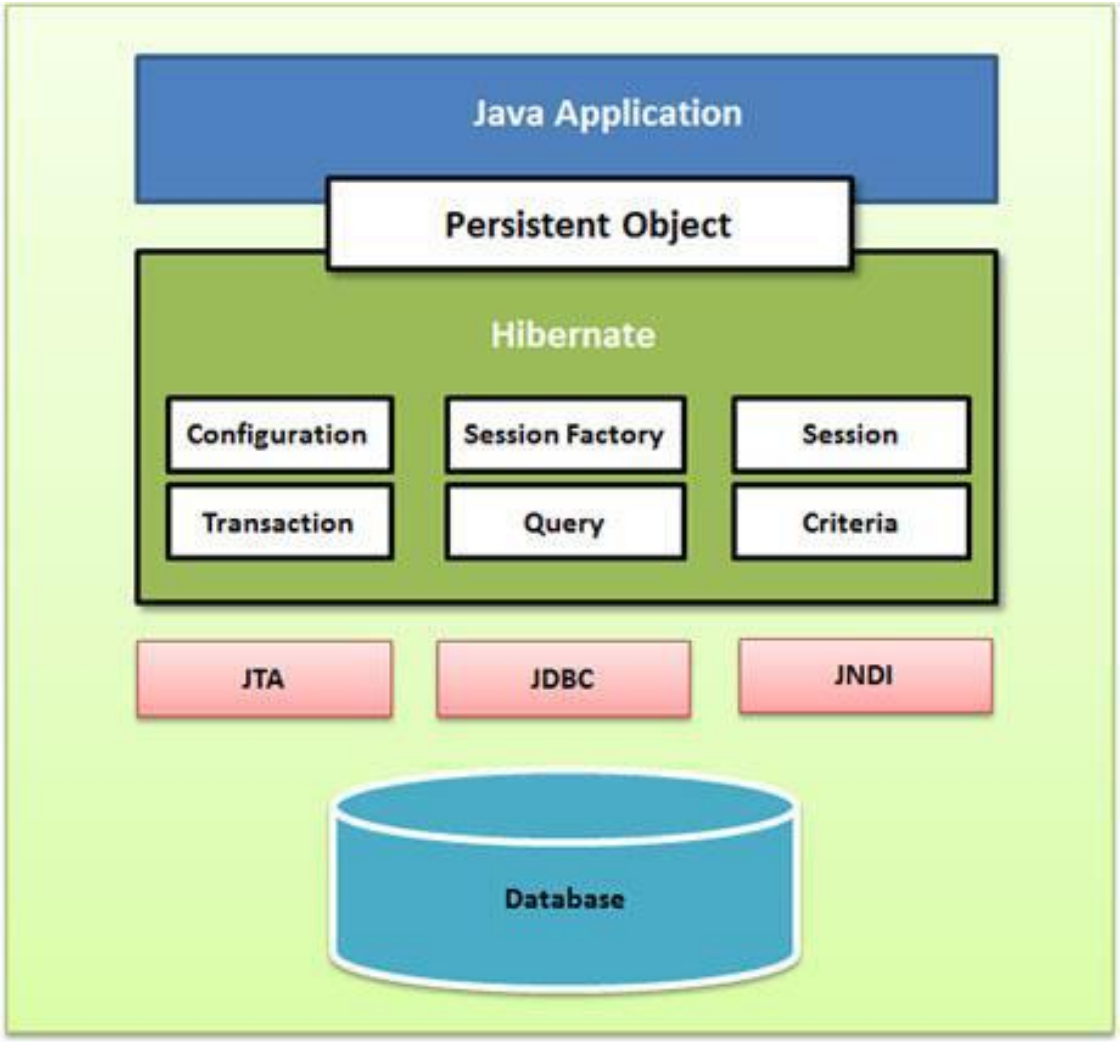
ĐẶC ĐIỂM CỦA HIBERNATE

- Hibernate cung cấp API đơn giản để thao tác và truy vấn các đối tượng trực tiếp từ cơ sở dữ liệu.
- Hibernate trong suốt với SQL - chúng ta chỉ làm việc với các đối tượng.
- Truy vấn các đối tượng kết hợp một cách dễ dàng thông qua mối quan hệ giữa các thực thể.
- Hibernate giúp giảm bớt 95% nhiệm vụ của người lập trình cơ sở dữ liệu.

Hỗ trợ CSDL

- HSQL Database Engine
- DB2/NT
- MySQL
- PostgreSQL
- FrontBase
- Oracle
- **Microsoft SQL Server Database**
- Sybase SQL Server
- Informix Dynamic Server

CÁC THÀNH PHẦN HIBERNATE

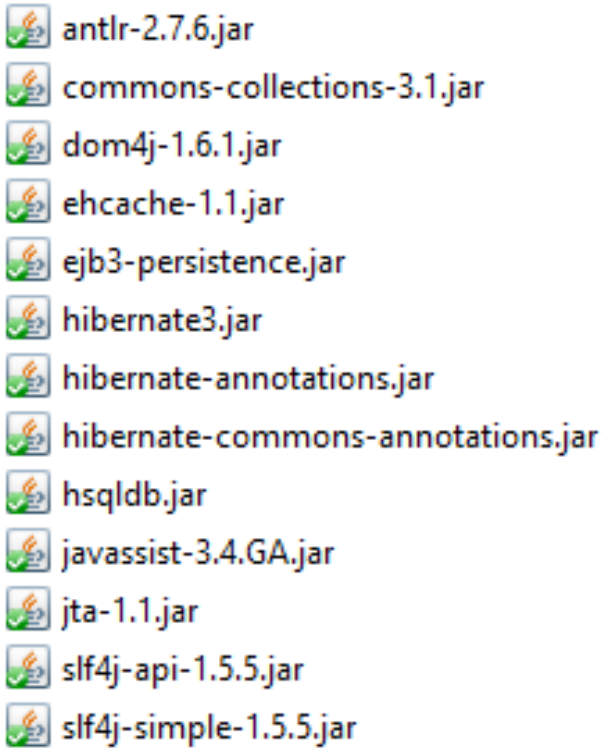


CÁC THÀNH PHẦN HIBERNATE

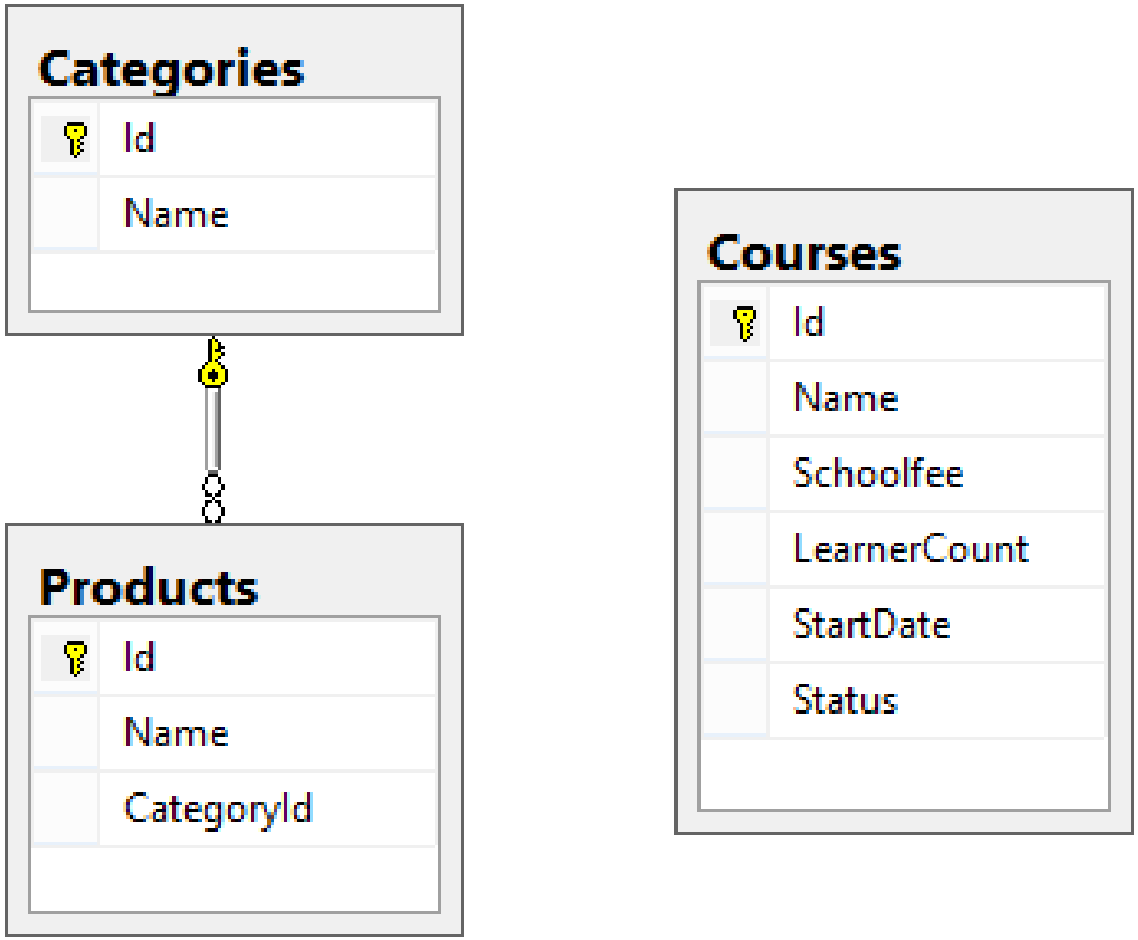
- **Configuration**: được sử dụng để quản lý thông tin cấu hình kết nối đến CSDL và ánh xạ thực thể vào CSDL.
- **SessionFactory**: đối tượng này cho phép sản sinh ra nhiều phiên làm (session) việc khác nhau.
- **Session**: sử dụng để tạo một phiên làm việc với CSDL thông qua Hibernate.
- **Transaction**: sử dụng để điều khiển transaction (none or all)
- **Query**: Đối tượng này được sử dụng để thực hiện truy vấn dữ liệu
- **Criteria**: Đối tượng này được sử dụng để xây dựng câu lệnh truy vấn bằng lập trình thay cho truy vấn bằng câu lệnh HQL hay SQL.

THƯ VIỆN

- Trong số các thư viện trên, **sqljdbc4.jar** được sử dụng để làm việc với SQL Server 2000/2005/2008 của Microsoft.
- Nếu bạn muốn làm việc với một CSDL cụ thể nào đó (Oracle, MySQL, DB2...) thì phải bổ sung thư viện JDBC tương ứng với CSDL đó vào ứng dụng của bạn.



CSDL MẪU eSTORE



HIBERNATE.CFG.XML

```
<session-factory>
    <property name="hibernate.connection.driver_class">
        com.microsoft.sqlserver.jdbc.SQLServerDriver</property>
    <property name="hibernate.connection.url">
        jdbc:sqlserver://localhost:1433;DatabaseName=eStore;</property>
    <property name="hibernate.connection.username">sa</property>
    <property name="connection.password">songlong</property>
    <property name="connection.pool_size">5</property>
    <property name="show_sql">false</property>
    <property name="hbm2ddl.auto">none</property>
    <property name="hibernate.dialect">
        org.hibernate.dialect.SQLServerDialect</property>
    <!-- Mapping files -->
  <mapping class="estore.entity.Course"/>
  <mapping class="estore.entity.Category"/>
  <mapping class="estore.entity.Product"/>
</session-factory>
```

HIBERNATE.CFG.XML

@name	Mô tả
hibernate.connection.driver_class	JDBC driver
hibernate.connection.url	Chuỗi kết nối đến CSDL
hibernate.connection.username	Mã đăng nhập CSDL
connection.password	Mật khẩu đăng nhập CSDL
connection.pool_size	Số kết nối tối đa trong hồ
show_sql	Xuất câu lệnh SQL khi truy vấn hoặc thao tác dữ liệu
hbm2ddl.auto	Chỉ ra cơ chế tự định nghĩa CSDL, thường dùng ✓ create-drop: tự tạo CSDL ✓ none: làm việc với CSDL đã tồn tại
hibernate.dialect	Chỉ ra phương pháp làm việc với CSDL của Hibernate (xem bảng bên dưới để khai báo cho đúng)

ÁNH XẠ

- Ánh xạ là sự mô tả việc kết hợp giữa
  - Lớp Entity và bảng
  - Các trường và các cột trong bảng
  - Thực thể kết với với Relationship
- Lớp Entity phải thỏa qui ước JavaBean
  - Constructor: phải có hàm tạo mặc định
  - Property: getXyz()/setXyz()

ÁNH XẠ THỰC THỂ

```
@Entity
@Table(name="Courses")
public class Course {
    @Id
    @Column(name="Id")
    String id;
    @Column(name="Name")
    String name;
    @Column(name="Schoolfee")
    Double schoolfee;
    @Column(name="LearnerCount")
    Integer noOfLearners;
    @Column(name="StartDate")
    Date startDate;
    @Column(name="Finished")
    boolean finished;
```

Courses

	Column Name	Condensed Type
🔑	Id	nvarchar(5)
	Name	nvarchar(50)
	Schoolfee	float
	LearnerCount	int
	StartDate	date
	Finished	bit

```
public String getId() {
public void setId(String id) {
```

getXyz()/setXyz()

QUI ƯỚC ÁNH XẠ

- Nếu Entity và Table cùng tên thì có thể bỏ @Table
- Nếu field của Entity và column của Table cùng tên thì có thể bỏ @Column()

```
@Entity
@Table(name="Courses")
public class Course {
    @Id
    String id;
    String name;
    Double schoolfee;
    Integer learnerCount;
    Date startDate;
    boolean finished;
```

Annotation	Mô tả	Ví dụ
@Entity	Chỉ ra class là thực thể	@Entity @Table(name="Courses") public class Course{...}
@Table	Ánh xạ lớp thực thể với bảng	
@Id	Chỉ ra cột khóa chính	@Id @GeneratedValue int id;
@GeneratedValue	Chỉ ra cột tự tăng	
@Column	Ánh xạ thuộc tính với cột	@Column(name = "Name") String name
@Temporal	Chỉ ra kiểu dữ liệu chuyển đổi	@Temporal(TemporalType.DATE) Date startDate
@ManyToOne	Ánh xạ quan hệ nhiều-1	@ManyToOne @JoinColumn(name="CategoryId") Category category;
@JoinColumn	Chỉ ra cột kết nối	
@OneToMany	Ánh xạ quan hệ 1-nhiều	@OneToMany(mappedBy="category") Collection<Product> products;

TRUY VẤN THỰC THỂ

Cấu hình

```
Configuration config = new Configuration().configure();
SessionFactory factory = config.buildSessionFactory();
Session session = factory.openSession();
```

Truy vấn

```
String hql = "FROM Course";
Query query = session.createQuery(hql);
List<Course> courses =  query.list();
```

Hiển thị

```
for(Course course : courses){
        String name = course.getName();
        double fee = course.getSchoolfee();
        System.out.println("+ Name of course: " + name);
        System.out.println("+ Schoolfee of course: " + fee);
}
```

TRUY VẤN THỰC THỂ

- Tham số

```
String hql = "FROM Course WHERE name LIKE :name "
                          + " AND schoolfee BETWEEN :min AND :max";
Query query = session.createQuery(hql);
query.setParameter("name", "%asp%");
query.setParameter("min", 5.5);
query.setParameter("max", 200);
List<Course> courses =  query.list();
```

- Phân trang

```
String hql = "FROM Course";
Query query = session.createQuery(hql);
query.setFirstResult(5); // vị trí bắt đầu
query.setMaxResults(10); // số lượng thực thể
List<Course> courses =  query.list();
```

TRUY VẤN THỰC THỂ

- Đặc thù

```
String sql = "SELECT * FROM Courses ORDER BY NewID()";
Query query = session.createSQLQuery(sql);
List<Course> courses =  query.list();
```

- Một số thuộc tính

```
String hql = "SELECT name, schoolfee FROM Course";
Query query = session.createQuery(hql);
List<Object[]> arrays =  query.list();
```

- Một giá trị

```
String hql = "SELECT AVG(schoolfee) FROM Course";
Query query = session.createQuery(hql);
double average = (Double)query.uniqueResult();
```

TRUY VẤN 1 THỰC THỂ

- Sử dụng load()/get()

  Course entity = (Course)session.**load**(Course.class, ‘MVC’)

  Course entity = (Course)session.**get**(Course.class, ‘MVC’)

- Sử dụng refresh()

  Course entity = new Course();

  entity.setId(‘MVC’)

  session.**refresh**(entity);

THAO TÁC THỰC THỂ

```
Transaction transaction = session.beginTransaction();
try{
        ...mã thao tác thực thể...

        /*
         * Chấp nhận kết quả
         */
        transaction.commit();
}
catch (Exception e) {
        /*
         * Hủy bỏ kết quả
         */
        transaction.rollback();

}
```

THÊM THỰC THỂ MỚI

```
Transaction transaction = session.beginTransaction();
try{
        Course entity = new Course();
        entity.setName("Clocks 2012");
        entity.setSchoolfee(123.9);
        entity.setNoOfLearners(25);
        entity.setStartDate(new Date());
        entity.setFinished(false);
        session.save(entity);
        transaction.commit();
}
catch (Exception e) {
        transaction.rollback();

}
```

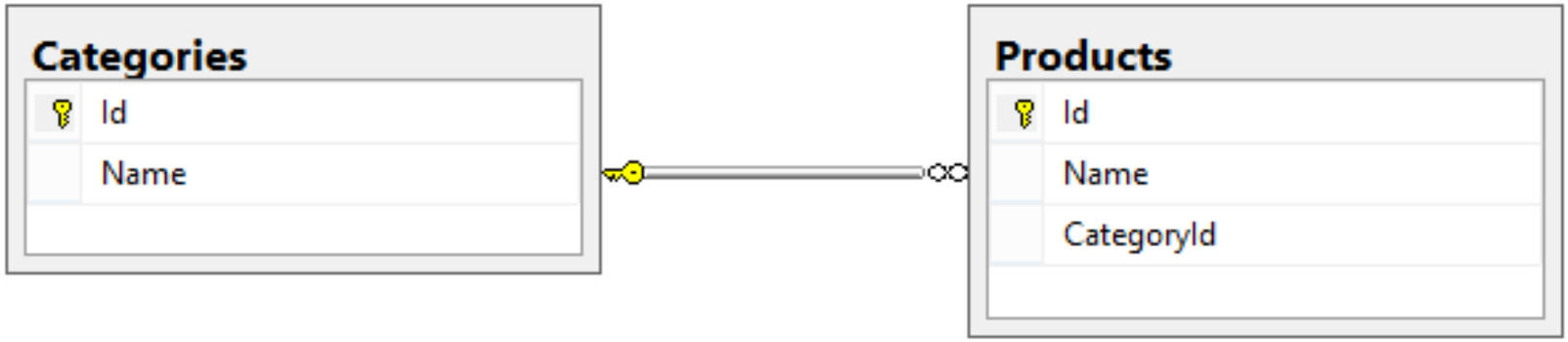
CẬP NHẬT THỰC THỂ

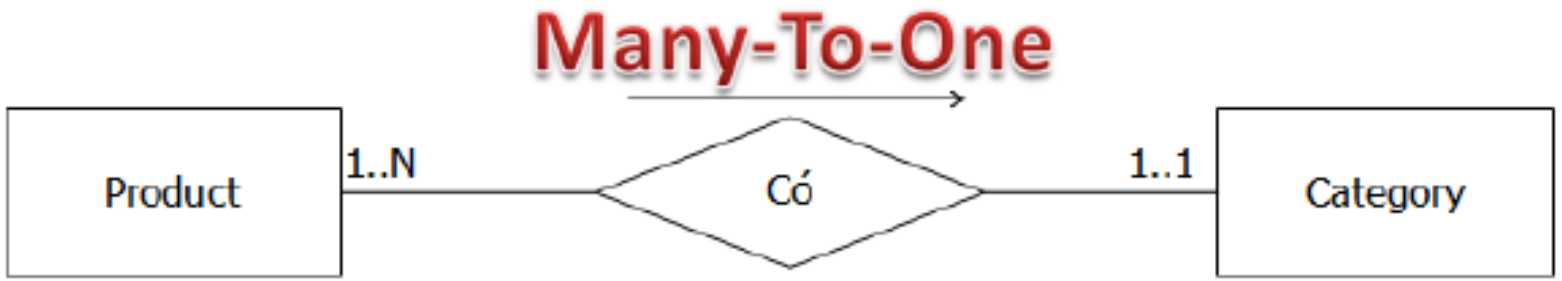
```
Transaction transaction = session.beginTransaction();
try{
        Course entity = (Course) session.load(Course.class, 1);
        entity.setNoOfLearners(38);
        entity.setStartDate(new Date());
        session.update(entity);
       transaction.commit();
}
catch (Exception e) {
       transaction.rollback();

}
```

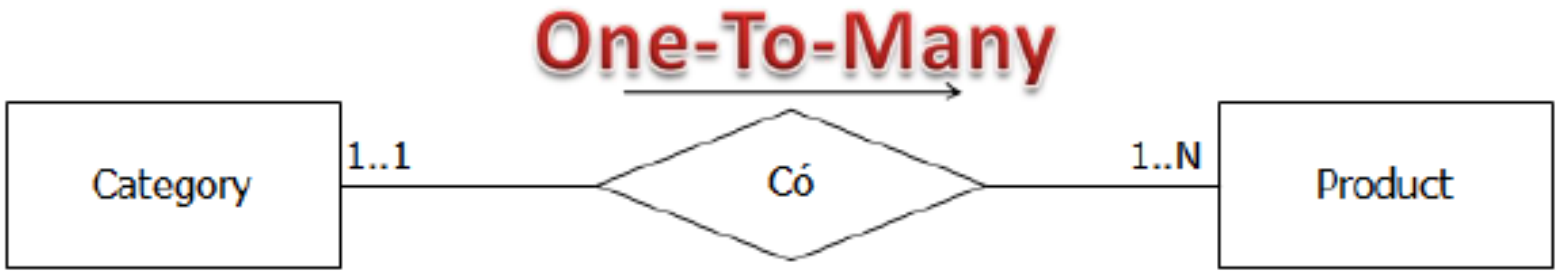
XÓA THỰC THỂ

```
Transaction transaction = session.beginTransaction();
try{
        Course entity = new Course();
        entity.setId(1);
        session.delete(entity);
        transaction.commit();
}
catch (Exception e) {
        transaction.rollback();

}
```

THỰC THỂ KẾT HỢP







ÁNH XẠ THỰC THỂ KẾT HỢP

```
@Entity
@Table(name="Categories")
public class Category {
        @Id
        @GeneratedValue
        Integer id;
        String name;

        @OneToMany(mappedBy="category")
        Collection<Product> products;
```

@ManyToOne

@OneToMany

```
@Entity
@Table(name="Products")
public class Product {
        @Id
        @GeneratedValue
        Integer id;
        String name;
        Double unitPrice;
        Date productDate;
        String image;
        Boolean available;

        @ManyToOne()
        @JoinColumn(name="CategoryId")
        Category category;
```

NGÔN NGỮ HQL

- FROM – Tất cả thực thể
  - FROM Course
  - FROM Course **as c**
  - FROM Course **c**
- WHERE – Lọc theo điều kiện
  - FROM Course WHERE name **LIKE** ‘Nguyễn%’
  - FROM Course WHERE schoolfee **BETWEEN** 100 **AND** 250
  - FROM Product WHERE description **IS NOT NULL**
  - FROM Product p WHERE p.category.id **IN** (1, 3, 5, 7)
- SELECT – Một số thuộc tính
  - SELECT **name, schoolfee** FROM Course
  - SELECT c.name, c.schoolfee FROM Course c

NGÔN NGỮ HQL

- Sắp xếp
  - FROM Course **ORDER BY** startDate **DESC**, schoolfee
- Thống kế
  - SELECT **AVG**(unitPrice), **MAX**(discount) FROM Product p **GROUP BY** p.category
  - SELECT AVG(unitPrice), MAX(discount) FROM Product p GROUP BY p.category **HAVING** AVG(unitPrice) > 100

NGÔN NGỮ HQL

- Toán tử
  - Số học:+, -, *, /
  - So sánh:=, >=, <=, <>, !=
  - Logic:AND, OR, NOT
- Đặc biệt:
  - NOT
  - IN,
  - BETWEEN
  - IS NULL
  - LIKE
  - IS EMPTY

NGÔN NGỮ HQL

- Tổng hợpsốliệu
  - avg(...), sum(...), min(...), max(...)
  - count(*)
  - count(...), count(distinct ...), count(all...)
- Xử lý chuỗi
  - concat(...,...): ghépchuỗi
  - substring(): lấy chuỗi con
  - trim(): cắt bỏ ký tự trắng 2 đầu chuỗi
  - lower(): chuyển in thường
  - upper(): chuyển in hoa
  - length(): lấy đội dài chuỗi
  - locate(): tìm vị trí chuỗi con

NGÔN NGỮ HQL

- Hàm xử lý thời gian
  - current_date(): lấy ngày, tháng năm
  - current_time(): lấy giờ, phút và giây
  - current_timestamp(): lấy ngày giờ
  - second(...): lấy giây
  - minute(...): lấy phút
  - hour(...): lấy giờ trong ngày
  - day(...): lấy ngày trong tháng
  - month(...): lấy tháng
  - year(...): lấy năm

NGÔN NGỮ HQL

- Các hàm khác
  - abs(): lấy giá trị tuyệt đối
  - sqrt(): tính căn bậc 2
  - str(): chuyển số/ngày sang chuỗi
  - cast(... as ...): ép kiểu

TÓM TẮT

- Giới Thiệu
- Ánh xạ
  - Tập tin cấu hình
  - Xây dựng lớp thực thể
- Hibernate API
  - Truy vấn
    - Truy vấn thực thể
    - Truy vấn một thực thể theo khóa chính
  - Thao tác dữ liệu
    - Thêm mới
    - Cập nhật
    - Xóa
- Ngôn ngữ HQL