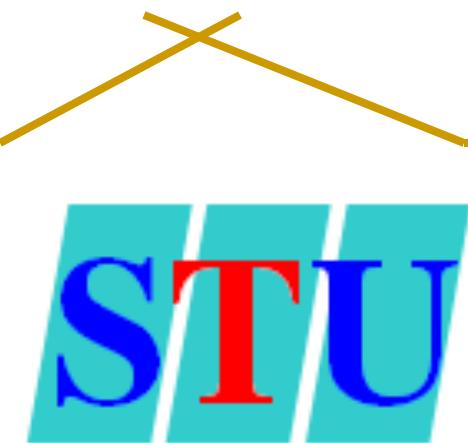


JPA – JPQL

Java Persistence Query Language



Trường Đại học Công nghệ Sài Gòn
Khoa Công nghệ Thông tin

JPA setFirstResult

■ Truy vấn 3 Student

- Query query=em.createQuery("SELECT st FROM Student st");
- query.setFirstResult(2); //start 0
- List stuList=query.getResultList();

setMaxResult

■ Số record max trả về

- Query query=em.createQuery("SELECT st FROM Student st WHERE st.sroll > ?1");
- query.setParameter(1, 100);
- **query.setMaxResults(3);**
- List stuList=query.getResultList();

JPA setParameter ()

■ Có 2 dạng tham số truyền

- Named Parameters:**setParameter(String namedParamter, Object val)**

- **Query query=em.createQuery("SELECT st FROM Student st WHERE st.sroll > :roll");**
 - **query.setParameter("roll", 100);**
 - **List results = query.getResultList();**

- Numbered Parameters:**setParameter(int numberedParamter, Object val)**

- **Query query=em.createQuery("SELECT st FROM Student st WHERE st.sroll > ?1");**
 - **query.setParameter(1, 100);**
 - **List stuList=query.getResultList();**

- **getResultSet:** trả về danh sách kết quả từ câu lệnh truy vấn
 - Query query = em.createQuery("SELECT st FROM Student st WHERE st.sroll > ?1");
 - query.setParameter(1, 101);
 - List results = query.getResultList();
- **getSingleResult(): Trả về một record**
 - Query query = em.createQuery("SELECT st FROM Student st WHERE st.sroll = :roll");
 - query.setParameter("roll", 101);
 - **Student st = (Student) query.getSingleResult();**

executeUpdate()

■ Sử dụng cho câu lệnh update/delete

- Query query = em.createQuery("DELETE FROM Student st WHERE st.sroll= ?1");
- query.setParameter(1, 104);
- **int n_record = query.executeUpdate();**

JPA Native Queries

■ Native Queries : Sử dụng câu lệnh truy vấn thông thường

- Query query=em.createNativeQuery("SELECT sname FROM Student");
- List stList=query.getResultList();

■ Name Queries: Đặt tên queries

- @javax.persistence.NamedQuery(name="studentRecords" ,
query="SELECT st FROM Student st WHERE st.sname= ?1 OR st.scourse= ?2")

- Query query=em.createNamedQuery("studentRecords");
- query.setParameter(1, "vinod");
- query.setParameter(2, "PHD");
- List stList=query.getResultList();

Khai báo NamedQueries

```
@NamedQueries({
```

```
    @NamedQuery(name="allStudentRecords",query  
 ="SELECT st FROM Student st WHERE st.sroll > ?1"),  
    @NamedQuery(name="updateStudentRecords",  
 query="UPDATE Student st SET st.sname= ?1  
 WHERE st.sname= ?2")
```

```
)}
```

Order, GROUP BY

- Query `query=em.createQuery("SELECT st FROM Student st order by st.sname desc");`
- Query `query=em.createQuery("SELECT st FROM Student st group by st.sname");`
- Sub Query
 - Query `query=em.createQuery("SELECT st FROM Student st WHERE st.sroll=(SELECT st.sroll FROM Student st WHERE st.sroll>100 AND st.sroll<102)");`

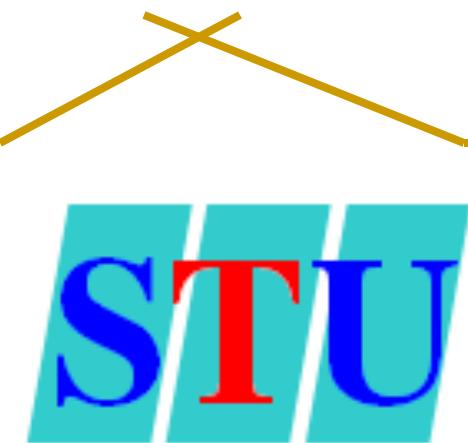
Aggregates

- Avg()
 - Query query=em.createQuery("SELECT p FROM Product p WHERE p.price > (SELECT **AVG**(p.price) FROM p);");
- Sum()
 - Query query=em.createQuery("SELECT **SUM**(p.price) FROM Product p");
- Min()
 - Query query=em.createQuery("SELECT p FROM Product p WHERE p.price=(SELECT **MIN**(p.price) FROM p)");
- Max()
 - Query query=em.createQuery("SELECT p FROM Product p WHERE p.price < (SELECT **MAX**(p.price) FROM p)");
- Count()
 - Query query=em.createQuery("SELECT COUNT(p.itemName) FROM Product p");

JPQL Functions

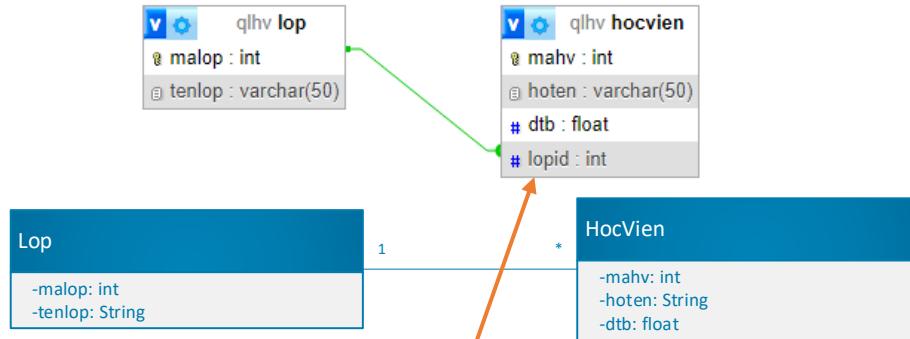
- **Concat()**
 - Query query=em.createQuery("UPDATE Student st SET st.sname=CONCAT(st.sname,' kumar')");
- **SubString()**
 - Query query=em.createQuery("UPDATE Student st SET st.sname=SUBSTRING('vinod kumar',1,5) WHERE st.sname='vinod kumar'");
- **Trim()**
 - Query trimQuery=em.createQuery("UPDATE Student st SET st.sname=TRIM(st.sname)");
- **Lower()**
 - Query trimQuery=em.createQuery("UPDATE Student st SET st.sname=LOWER(st.sname)");
- **Upper()**
- **Length()**
 - Query query=em.createQuery("SELECT st FROM Student st WHERE LENGTH(st.sname)=5");
- **Locate()**
 - Query query=em.createQuery("SELECT st FROM Student st WHERE LOCATE('i',st.sname)=2");
- **Abs()**
- **Sqrt()**
- **Mod()**
 - Query query=em.createQuery("SELECT p FROM Product p WHERE MOD(p.price,10) = 0");

JPA - OneToMany



Trường Đại học Công nghệ Sài Gòn
Khoa Công nghệ Thông tin

OneToMany Relationship Mapping



```
@Entity
public class HocVien {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int mahv;
    private String hoten;
    private float dtb;

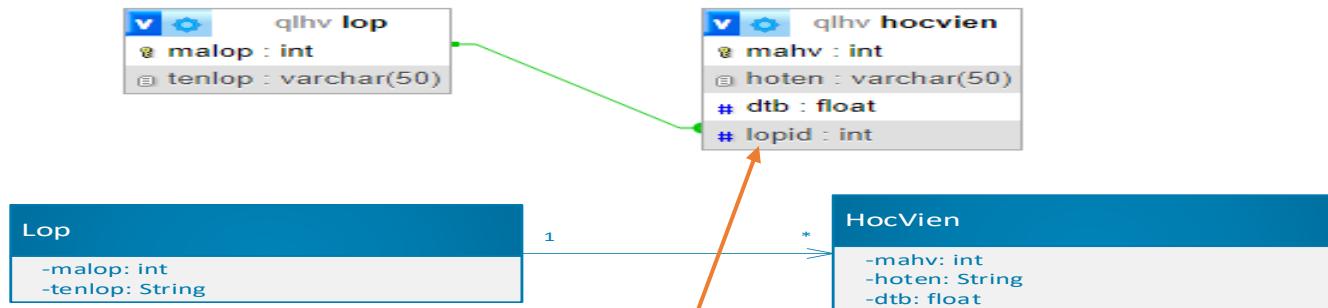
    @ManyToOne
    @JoinColumn(name = "lopid", nullable = false)
    private Lop lop;
}

@Entity
public class Lop {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int malop;
    private String tenlop;

    @OneToMany (mappedBy = "lop")
    private List<HocVien> dshv = new ArrayList<HocVien>();

    public void addHocVien(HocVien hv) {
        dshv.add(hv);
        hv.setLop(this);
    }
    public void removeHocVien(HocVien hv) {
        dshv.remove(hv);
        hv.setLop(null);
    }
}
```

OneToMany Relationship Mapping



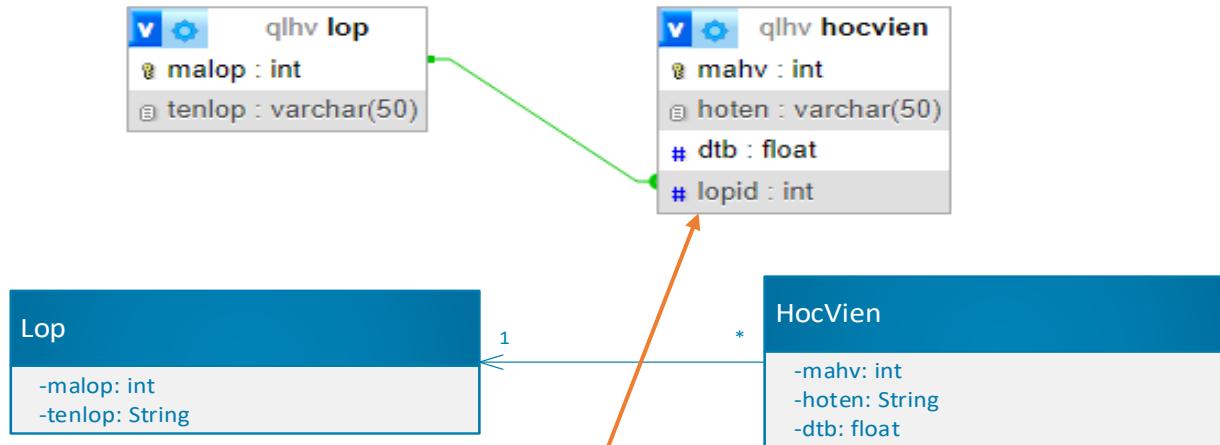
```
@Entity
public class Lop {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int malop;
    private String tenlop;

    @OneToMany
    @JoinColumn(name = "lopid")
    private List<HocVien> dshv = new ArrayList<HocVien>();
```

```
@Entity
public class HocVien {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int mahv;
    private String hoten;
    private float dtb;
```

```
    @OneToMany (cascade = CascadeType.MERGE, fetch = FetchType.LAZY) ???
    @JoinColumn(name = "lopid", nullable = false) ???
```

OneToMany Relationship Mapping



```
@Entity
public class HocVien {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int mahv;
    private String hoten;
    private float dtb;

    @ManyToOne
    @JoinColumn(name = "lopid")
    private Lop lop;
```

```
@Entity
public class Lop {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int malop;
    private String tenlop;
```

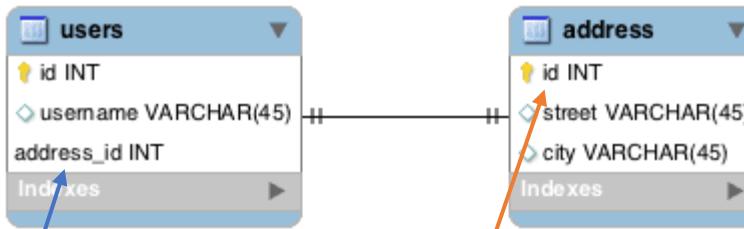
JPA - OneToOne

OneToOne Relationship Mapping

3 cách tiếp cận:

- Dùng **Foreign Key**
- Dùng **Shared Primary Key**
- Dùng **Join Table**

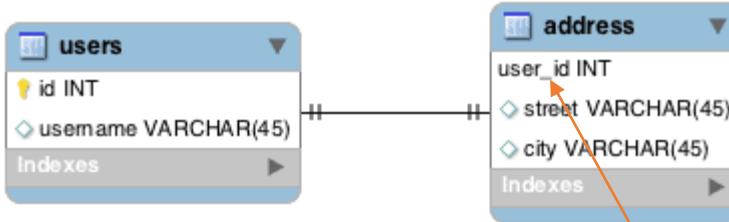
Dùng Foreign Key



```
@Entity  
@Table(name = "users")  
public class User {  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    @Column(name = "id")  
    private Long id;  
    //...  
  
    @OneToOne(cascade = CascadeType.ALL)  
    @JoinColumn(name = "address_id", referencedColumnName = "id")  
    private Address address;  
  
    // ... getters and setters  
}
```

```
@Entity  
@Table(name = "address")  
public class Address {  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    @Column(name = "id")  
    private Long id;  
    //...  
  
    @OneToOne(mappedBy = "address")  
    private User user;  
  
    //... getters and setters  
}
```

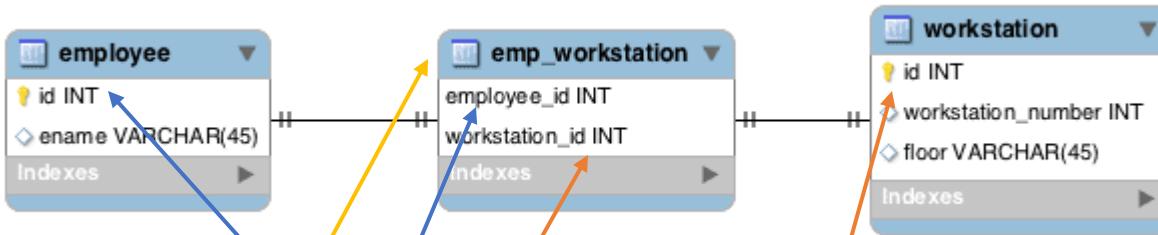
Dùng Shared Primary Key



```
@Entity  
@Table(name = "users")  
public class User {  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    @Column(name = "id")  
    private Long id;  
  
    //...  
  
    @OneToOne(mappedBy = "user", cascade = CascadeType.ALL)  
    @PrimaryKeyJoinColumn  
    private Address address;  
  
    //... getters and setters  
}
```

```
@Entity  
@Table(name = "address")  
public class Address {  
  
    @Id  
    @Column(name = "user_id")  
    private Long id;  
  
    //...  
  
    @OneToOne  
    @MapsId  
    @JoinColumn(name = "user_id")  
    private User user;  
  
    //... getters and setters  
}
```

Dùng Join Table



```
@Entity
@Table(name = "employee")
public class Employee {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "id")
    private Long id;

    //...

    @OneToOne(cascade = CascadeType.ALL)
    @JoinTable(name = "emp_workstation",
        joinColumns =
            { @JoinColumn(name = "employee_id", referencedColumnName = "id") },
        inverseJoinColumns =
            { @JoinColumn(name = "workstation_id", referencedColumnName = "id") })
    private WorkStation workStation;

    //... getters and setters
}
```

```
@Entity
@Table(name = "workstation")
public class WorkStation {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "id")
    private Long id;

    //...

    @OneToOne(mappedBy = "workStation")
    private Employee employee;

    //... getters and setters
}
```