



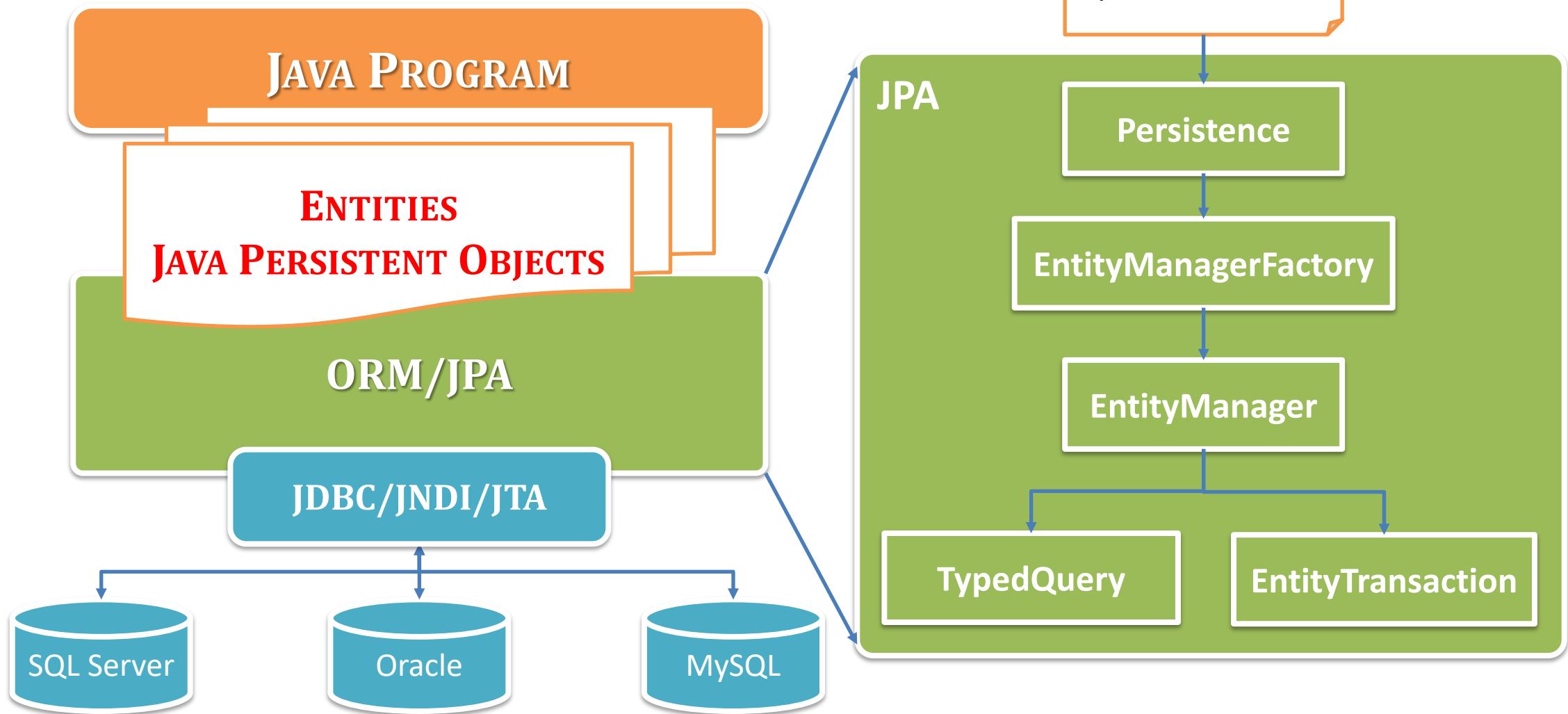
LẬP TRÌNH JPA

LẬP TRÌNH JAVA #4 (P1.2)

- Giới thiệu các thành phần JPA
- EntityManagerFactory
- EntityManager
- UserTransaction
- TypedQuery
- Entity LifeCycle

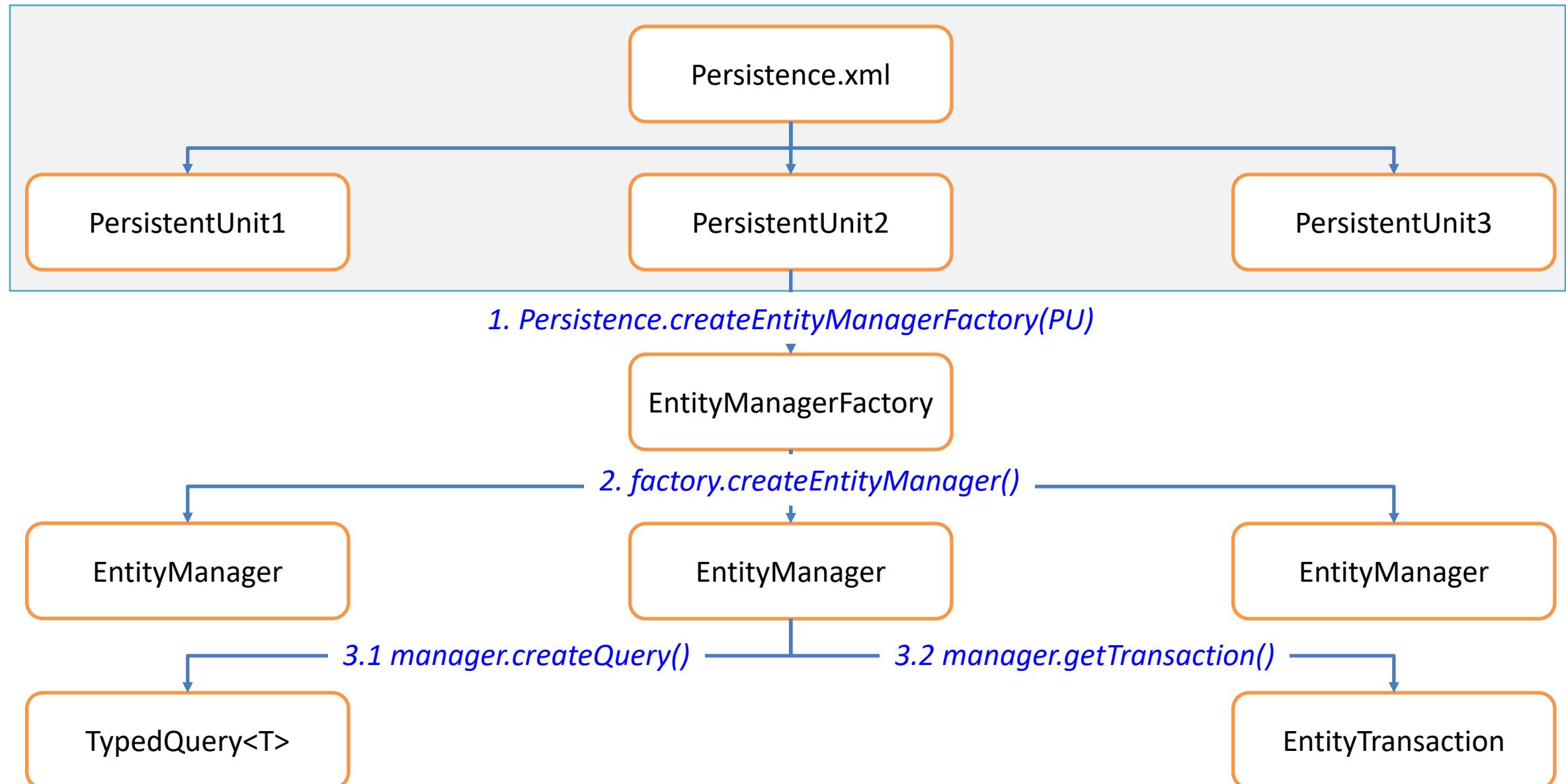


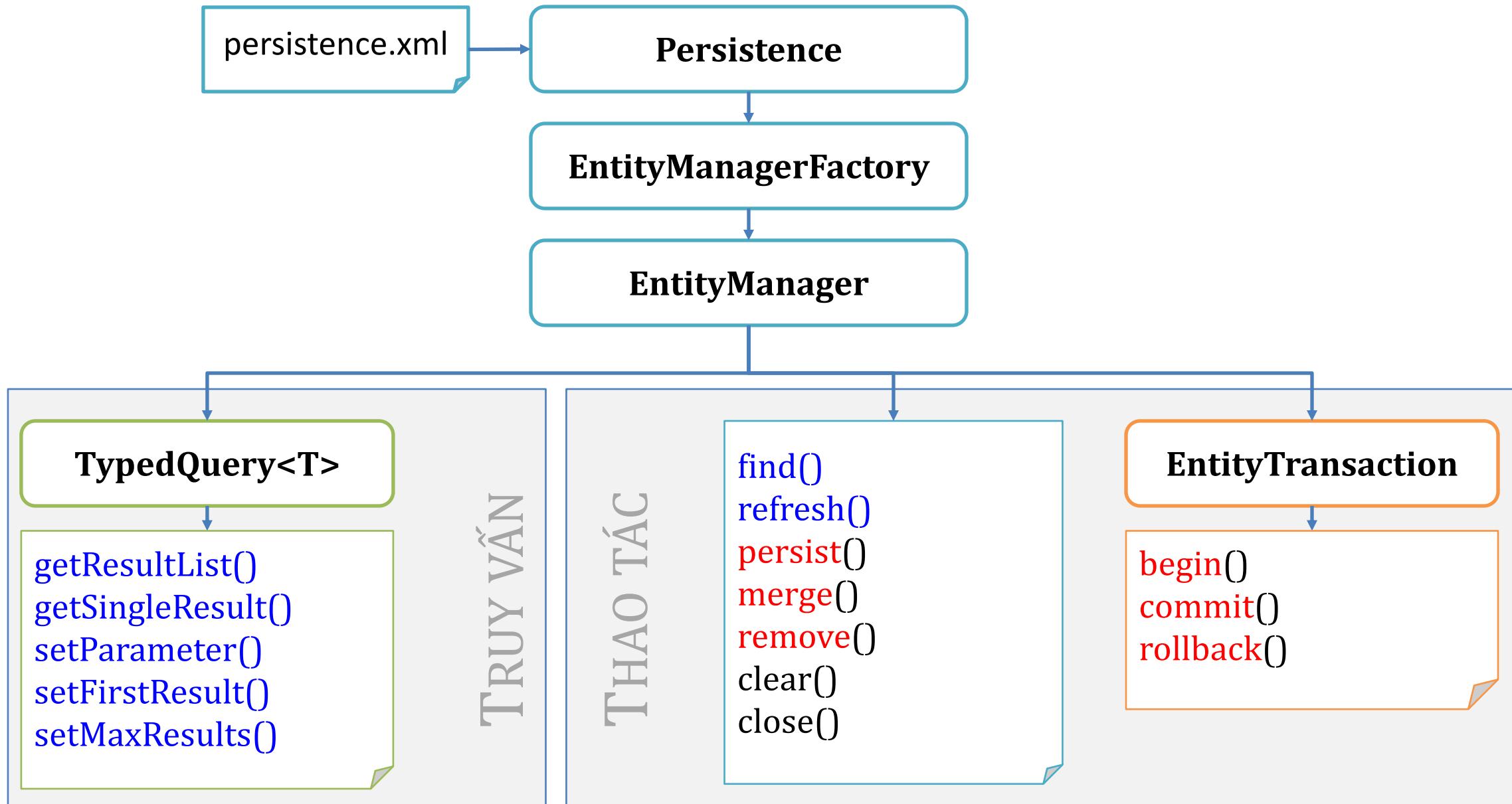
TỔNG QUAN VỀ JPA



- ❑ **persistence.xml:** khai báo kết nối đến CSDL
- ❑ **Persistence:** Nạp persistence.xml và tạo ra đối tượng EntityManagerFactory
- ❑ **EntityManagerFactory:** Đối tượng này cho phép tạo ra EntityManager để bắt đầu lập trình truy vấn và thao tác dữ liệu.
- ❑ **EntityManager:** Cho phép thao tác (thêm, sửa, xóa) và truy vấn 1 thực thể
- ❑ **EntityTransaction:** Dùng để điều khiển transaction các thao tác dữ liệu (thêm, sửa, xóa)
- ❑ **TypedQuery<T>:** Dùng để truy vấn dữ liệu với câu lệnh JPQL

MÔ HÌNH LẬP TRÌNH JPA





REVIEW: JPA – TRUY VẤN DỮ LIỆU

```
EntityManagerFactory factory = Persistence.createEntityManagerFactory("PolyOE");
EntityManager em = factory.createEntityManager();
```

```
// Câu lệnh JPQL truy vấn thực thể
String jpql = "SELECT o FROM User o";
// Tạo Query/TypedQuery<T> để truy vấn
TypedQuery<User> query = em.createQuery(jpql, User.class);
// Truy vấn danh sách thực thể
List<User> list = query.getResultList();
// Truy vấn một thực thể (nếu JPQL đảm bảo chỉ có 1 thực thể)
User entity = query.getSingleResult();

em.close();
```

JPQL là câu lệnh truy vấn đối tượng (*các thành phần bên trong là Entity Class và Property chứ không phải là Table và Column*)

TypedQuery<T>

- getResultList(): List<T>
- getSingleResult(): T

Hãy căn cứ mệnh đề SELECT để xác định T

- ❑ Tham số có thể là tên hoặc vị trí
 - ❖ Tên thì phải bắt đầu bởi dấu hai chấm (:)
 - ❖ Vị trí thì phải bắt đầu bởi dấu hỏi (?)
- ❑ Sử dụng setParameter() để cung cấp dữ liệu cho tham số

```
String jpql = "SELECT o FROM User o WHERE o.admin=:role";
TypedQuery<User> query = em.createQuery(jpql, User.class);
query.setParameter("role", true); // cung cấp giá trị cho tham số :role
List<User> list = query.getResultList();
```

```
String jpql = "SELECT o FROM User o WHERE o.admin=?0";
TypedQuery<User> query = em.createQuery(jpql, User.class);
query.setParameter(0, true); // cung cấp giá trị cho dấu ? đầu tiên
List<User> list = query.getResultList();
```

Tham số

- setParameter(String, Object)
- setParameter(int, Object)



DEMOSTATION

Truy vấn có tham số: tìm kiếm users

```
String jpql = "SELECT o FROM User o";
TypedQuery<User> query = em.createQuery(jpql, User.class);
query.setFirstResult(10); // vị trí bắt đầu
query.setMaxResults(8); // số lượng thực thể tối đa
List<User> list = query.getResultList();
```

Phân trang

- setFirstResult(int)
- setMaxResults(int)

❑ Cập phương thức **setFirstResult()** và **setMaxResults()** cho phép truy xuất một phân đoạn thực thể. Trong đó:

- ❖ **setFirstResult()** chỉ định vị trí bắt đầu
- ❖ **setMaxResults()** chỉ định số thực thể tối đa

❑ Áp dụng kỹ thuật này để truy vấn phân trang. Giả sử **pageNo** là vị trí trang, **pageSize** là số thực thể mỗi trang, để truy vấn 1 trang:

- ❖ **setFirstResult(pageNo*pageSize)**
- ❖ **setMaxResults(pageSize)**



DEMOSTATION

Truy vấn phân trang: phân trang users

REVIEW: LẬP TRÌNH THAO TÁC DỮ LIỆU

// 1. Nạp persistence.xml và tạo EntityManagerFactory

```
EntityManagerFactory factory = Persistence.createEntityManagerFactory("PolyOE");
```

// 2. Tạo EntityManager chuẩn bị lập trình CSDL

```
EntityManager em = factory.createEntityManager();
```

```
<persistence-unit name="PolyOE">
```

// 3.1. Lập trình thao tác dữ liệu

```
em.getTransaction().begin();
try {
    em.persist(entity); // em.merge(entity); | em.remove(entity);
    em.getTransaction().commit();
} catch (Exception e) {
    em.getTransaction().rollback();
}
```

// 3.2 Lập trình truy vấn một thực thể theo khóa chính

```
User user = em.find(User.class, "NghiemN");
```

// 4. Đóng EntityManager và kết thúc lập trình CSDL

```
em.close();
```

Lập trình thao tác dữ liệu cần được điều khiển Transaction.

Transaction = “**None or All**” (được ăn cả, ngã về không)

- ❑ Khẩu hiệu “None or All (Được ăn cả, ngã về không)” là khẩu hiệu của một transaction.
- ❑ Khi thực hiện các lệnh thao tác (insert, update, delete) sẽ làm thay đổi dữ liệu. Thỉnh thoảng chúng ta xem thực hiện một khối lệnh như một lệnh đơn (phải đúng hết mới chấp nhận, chỉ cần sai 1 lệnh nào đó thì phải hủy bỏ những lệnh đã thực hiện trước đó trong khối lệnh)
- ❑ Ví dụ: Tạo một đơn hàng bạn phải insert vào order 1 bản ghi và insert vào order_details nhiều bản ghi. Đơn hàng chỉ thành công khi nhiều lệnh insert nêu trên phải đúng toàn bộ.

```
em.getTransaction().begin();
try {
    Lệnh thao tác 1
    Lệnh thao tác 2
    ...
    em.getTransaction().commit();
} catch (Exception e) {
    em.getTransaction().rollback();
}
```

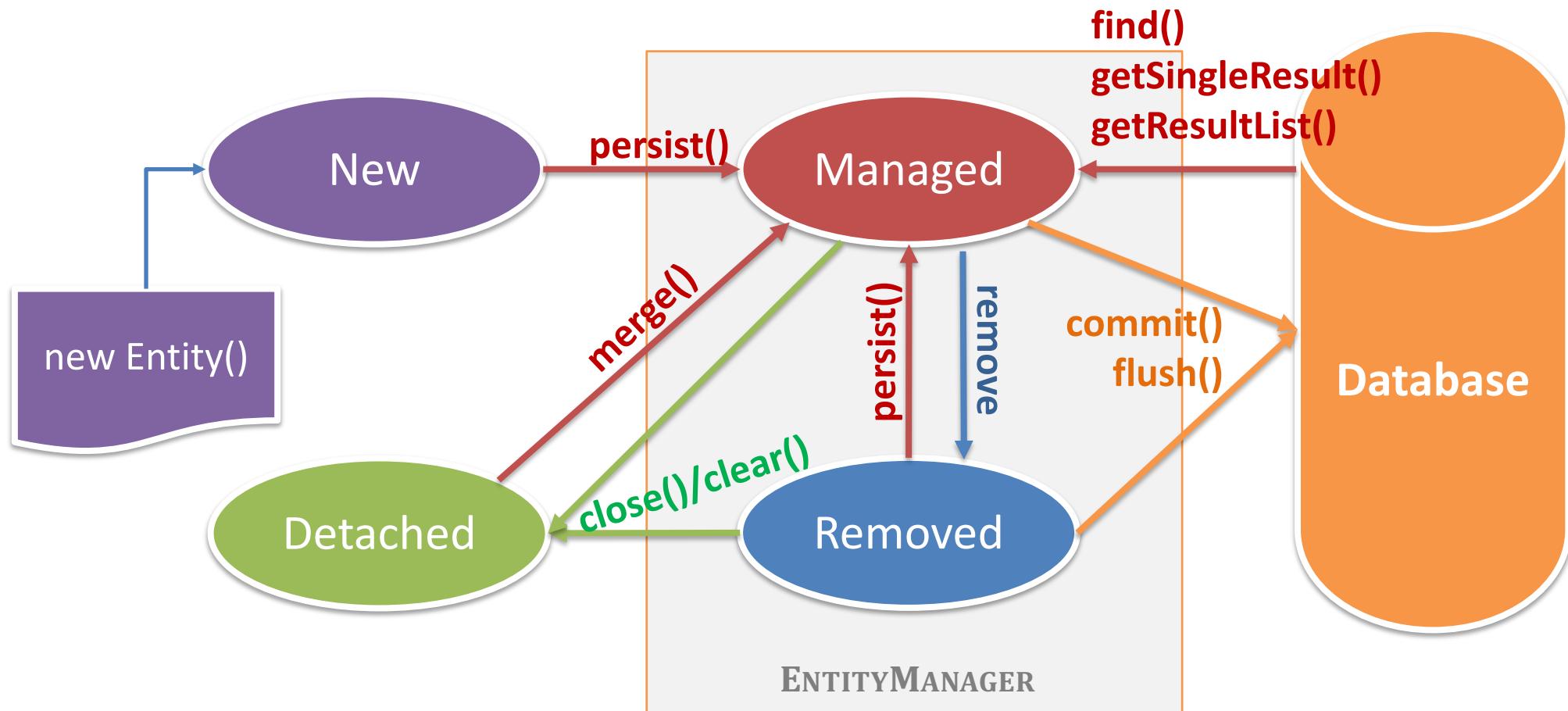
❑ UserTransaction ut = em.getTransaction()

- ❖ ut.begin(): bắt đầu transachtion
- ❖ ut.commit(): chấp nhận
- ❖ ut.rollback(): hủy bỏ



DEMOSTATION

Insert nhiều users với lệnh persist() trong đó lệnh persist ở giữa trùng khóa



- ✓ Giới thiệu các thành phần JPA
- ✓ EntityManagerFactory
- ✓ EntityManager
- ✓ UserTransaction
- ✓ TypedQuery
- ✓ Entity LifeCycle





Cảm ơn