JPA Cheat Sheet

EntityManager · JPQL · Criteria · Relationships

SETUP & PERSISTENCE.XML

```
<persistence-unit name="myPU"
transaction-type="RESOURCE_LOCAL">
<class>com.example.User</class>
<properties>..db props..</properties>
</persistence-unit>
@PersistenceUnit EntityManagerFactory
emf;
```

ENTITY MANAGER

```
EntityManager em =
emf.createEntityManager();
em.getTransaction().begin();
em.persist(entity);   // INSERT
em.merge(entity);     // UPDATE detached
em.remove(entity);    // DELETE
em.find(User.class, id); // SELECT by PK
em.getTransaction().commit();
```

ENTITY MAPPING

```
@Entity @Table(name="users")
public class User implements
Serializable {
@Id
@GeneratedValue(strategy=GenerationType.IDENTITY)
private Long id;
@Column(nullable=false,length=100)
private String name;
}
```

JPQL QUERIES

```
TypedQuery<User> q = em.createQuery(
"SELECT u FROM User u WHERE u.active =
:a", User.class);
q.setParameter("a", true);
List<User> list = q.getResultList();
User u = q.getSingleResult();
```

JPQL uses entity/field names, not SQL table/column names

NAMED QUERIES

```
@NamedQuery(name="User.findByEmail",
query="SELECT u FROM User u WHERE
u.email=:e")
em.createNamedQuery("User.findByEmail",
User.class)
.setParameter("e",
email).getSingleResult();
```

Parsed at startup fail fast on typos

RELATIONSHIPS

```
@OneToOne  @OneToMany  @ManyToOne
@ManyToMany
@JoinColumn(name="user_id")   FK column
@JoinTable   for ManyToMany bridge table
```

mappedBy on non-owning (inverse) side

Bidirectional: manage both sides in code

FETCH TYPES

LAZY (default for *ToMany): loads on first access

EAGER (default for *ToOne): loads with parent

```
@OneToMany(fetch = FetchType.EAGER) //
N+1 risk
```

Fix N+1: use JOIN FETCH in JPQL

```
"FROM Order o JOIN FETCH o.items"
@EntityGraph for ad-hoc fetch plans
```

CASCADE & ORPHAN REMOVAL

```
@OneToMany(cascade=CascadeType.ALL,
orphanRemoval=true)
```

CascadeType: PERSIST, MERGE, REMOVE, REFRESH, DETACH, ALL

orphanRemoval=true auto-deletes children removed from list

Cascade REMOVE on Many side = delete all children

Only cascade from parent to child, not vice versa

ENTITY LIFECYCLE CALLBACKS

```
@PrePersist  @PostPersist
@PreUpdate   @PostUpdate
@PreRemove   @PostRemove
@PostLoad
@EntityListeners(AuditListener.class)
```

Used for auditing: createdAt, updatedAt auto-fill

TRANSACTIONS

```
@Transactional  // Spring integration
em.getTransaction().begin()  //
resource-local
@PersistenceContext EntityManager em  //
injected
```

JTA: container-managed in Jakarta EE

```
@Transactional(propagation=REQUIRES_NEW)
```

CRITERIA API

```
CriteriaBuilder cb =
em.getCriteriaBuilder();
CriteriaQuery<User> cq =
cb.createQuery(User.class);
Root<User> r = cq.from(User.class);
cq.where(cb.equal(r.get("status"),"active"));
em.createQuery(cq).getResultList();
```

Type-safe; use metamodel (User_.name) for extra safety

COMMON GOTCHAS

Detached entities: merge() instead of persist()

LazyInitializationException outside transaction scope

Bidirectional always sync both sides

GenerationType.AUTO may use SEQUENCE on PostgreSQL

@Transactional only works on Spring-managed beans