## További egyszerű műveletek JPA-val

## (Entitás lekérdezésének lépései: entityManager.createQuery(JPQL uti, osztály), majd .getResultList(); JPQL def és használat

**Módosítás lépései: 1) betöltés find(osztály, id)-al azonosító alapján 2. módosítás setterrel 3) tranzakció végi commit, a háttérben legenerálódik egy update sql uti;**

**Törlés lépései: betöltés find()-al majd entityManager.remove(employee)), háttérben JPA mit generál le**

**elején EntityFactory- és Manager kreálások, .begin(), a végén close()ok mindig kellenek!**

**Lekérdezés**

az **entityManager.createQuery()** metódusát kell meghívni

**paraméterei:** első egy **JPQL lekérdezés**, második **az entitás osztálya**, így már nem kell castolni

a **JPQL ez a JPA-ban lévő szabvánnyelv**, egy **OO lekérdező nyelv**; hasonló az SQL-hez, de nem a táblaneveket é oszlopneveket kell használni, hanem az **osztálynevet és az attribútumneveket**

végül a .**getResultList()** visszaad egy Employee típusú listát



**A háttérben a jpa legenerál egy sql utasítást és ezt adja ki jdbcn keresztül, a visszaadott resultsetet átkonvertálja az annotációk alapján employee objektumokká és ezeket listába gyűjti.**

**Beolvasás azonosító alapján és módosítás**

egy **entitás módosításához először be kell azt töltenünk;** pl. **azonosító alapján** is be tudjuk tölteni az **entityManager.find()** metódusával, melynek **paraméterei: osztály és azonosító**

**setter** metódus meghívásával a JPA automatikusan fog arról gondoskodni a tranzakció végén(ha majd meghívjuk a **commitot**), hogy a módosításokat lementse adatbázisba-> **a háttérben legenerál egy update utasítást**



**Törlés azonosító alapján**

megint csak első lépés .**find()al az entitás betöltése**

majd a .**remove()nak** átadjuk a letöltött entitást, és a jpa a háttérben legenerál egy delete utasítást



## További egyszerű műveletek JPA-val - gyakorlat

1. *Entitás létrehozása és adatbázisba mentése* (new Employee() és .persist() és .commit() ), em.close, factory.close()
2. Az entitás azonosítója fel lesz töltve a lementés után, ha az azonosítója automatikusan generált:

sout id

1. *Azonosító alapján entitás betöltése*

find()-al, az előzőleg létrehozott employee id-jára; sout employee (toString felülírással)

**Nem történt select; az entity manager cacheli az entitásokat;** **ha nem ugyanazon nyitott entitymanageren keresztül kérdeznénk le, akkor lenne select**

1. *Update*

tranzakció indítása: em.getTransaction().begin(), find() és setter végül commit()

1. lekérdezés jpql-el: . createQuery(jpql összes Employeera, és class) majd getResultList()
2. *Törlés*

.begin(), majd, .find()al betöltés, majd remove() és commit(); törlés beemelés a listalekérdezés elé, aztán csekk

**Kérdések**

*Hogyan indítunk ekérdezést JPA-val és milyne paraméterei vannak?*

*Milyen metódussal kreáljuk a visszatérési értéket?*

*Mi az a JPQL? Mi fut JPQL kiadásakor a háttérben és ennek hatására mi történik?*

*Hogy olvasunk ber azonosító alapján és hogy módosítunk?*

*Hogy törlünk azonosító alapján?*