CS544 EA

Hibernate

EntityManager: Updating Entity Objects

Implicit Update

- When a **managed** entity is changed
  - Changes are **automatically pushed** to the DB

```
em.getTransaction().begin();
Person p1 = em.find(Person.class, 1L);
p1.setName("Ben James");
em.getTransaction().commit();
```

- On tx.commit(), em.flush(), or before a query
  - Hibernate notices that an object is ‘dirty’
  - Uses SQL Update to push changes to the DB

Detached Objects

- The Original hibernate API has an .update() method
  - Used to bring updates from **detached** objects to the database.
  - Changed state from Detached to Managed and marked object dirty

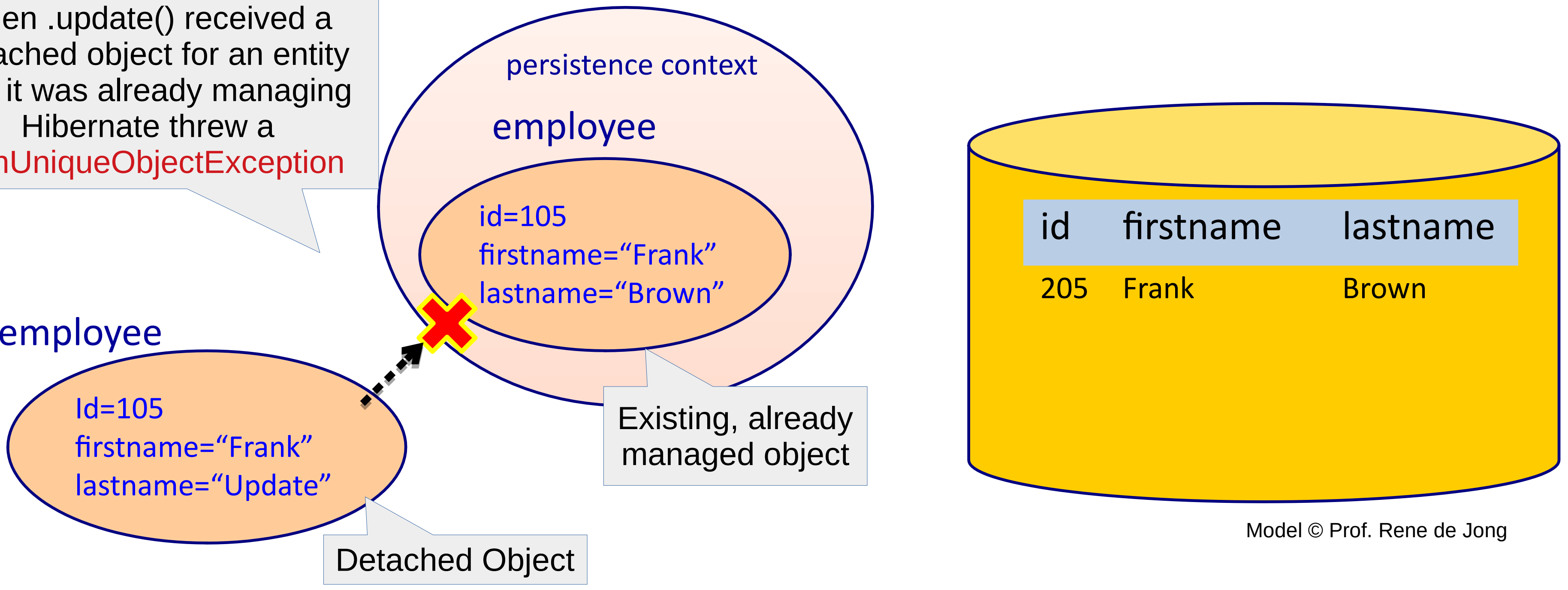
- **Big Problem**:
  - If the persistence context already has an object with the same type and primary key value a **NonUniqueObjectException** is created
  - Therefore JPA does not include this method

Problem

Not included in JPA

- **NonUniqueObjectException** with .update()

When .update() received a detached object for an entity that it was already managing Hibernate threw a NonUniqueObjectException

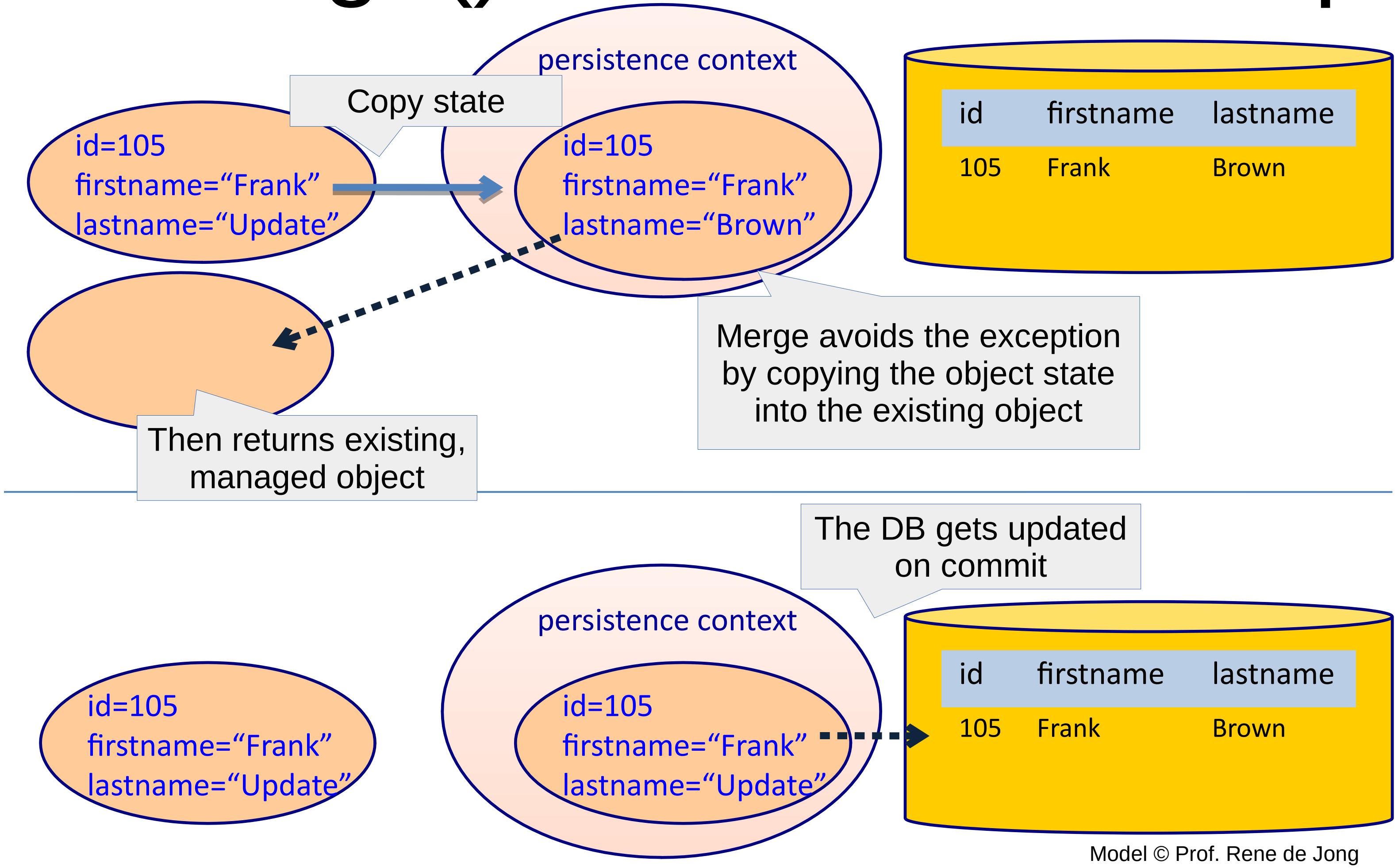


id	firstname	lastname
205	Frank	Brown

Model © Prof. Rene de Jong

Solution: use merge()

.merge() - Avoids NonUnique



Model © Prof. Rene de Jong

Merge - Misunderstood

- Merge does not behave like .persist()
  - The object you pass to the method never becomes managed
  - Instead **returns a different managed object** (of the same type)

- If you continue working with the **original object** you can run into **unexpected problems**
  - Implicit updates are not persisted (object not managed)

Correct use of .merge()

- Correct use:
  - Always **use the return** value from .merge()

p is set to the return

```
em.getTransaction().begin();
p = em.merge(p);
p.setName("Updated name");
em.getTransaction().commit();
```

Update will be persisted

- Incorrect use:
  - Keep using object passed into .merge()

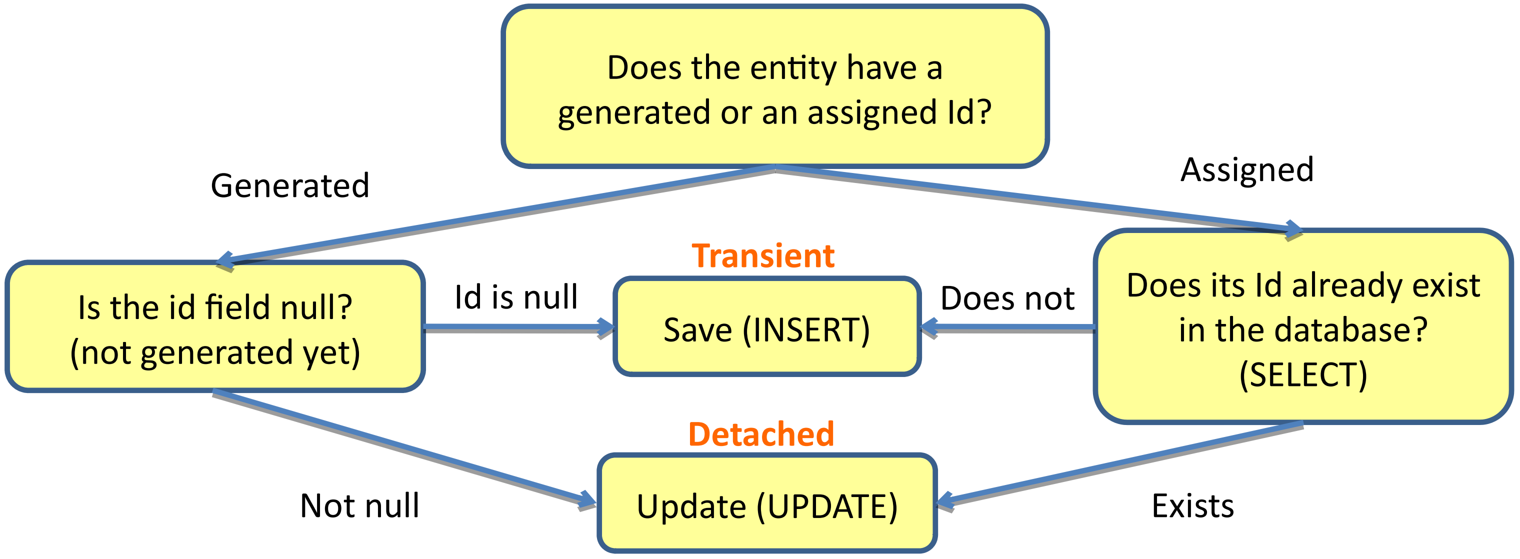
Return is lost
p is still
a detached
object

```
em.getTransaction().begin();
em.merge(p);
p.setName("Updated name");
em.getTransaction().commit();
```

Update will not reach the DB

Insert or Update

- .merge() has one more feature:
  - It inserts Transient objects (returns a managed copy)
  - Therefore can be used to **save or update** any object


Does the entity have a generated or an assigned Id?
Generated
Assigned
Is the id field null? (not generated yet)
Id is null
Transient
Save (INSERT)
Does not
Does its Id already exist in the database? (SELECT)
Detached
Not null
Update (UPDATE)
Exists

Model © Prof. Rene de Jong

.persist() or .merge()

- Is .persist() even needed?
  - You can do all inserts with .merge()

- .persist() makes your intent clearer
  - Different logic / implementation
  - Also checks id == null, but throws exception
  - **Never accidentally updates**