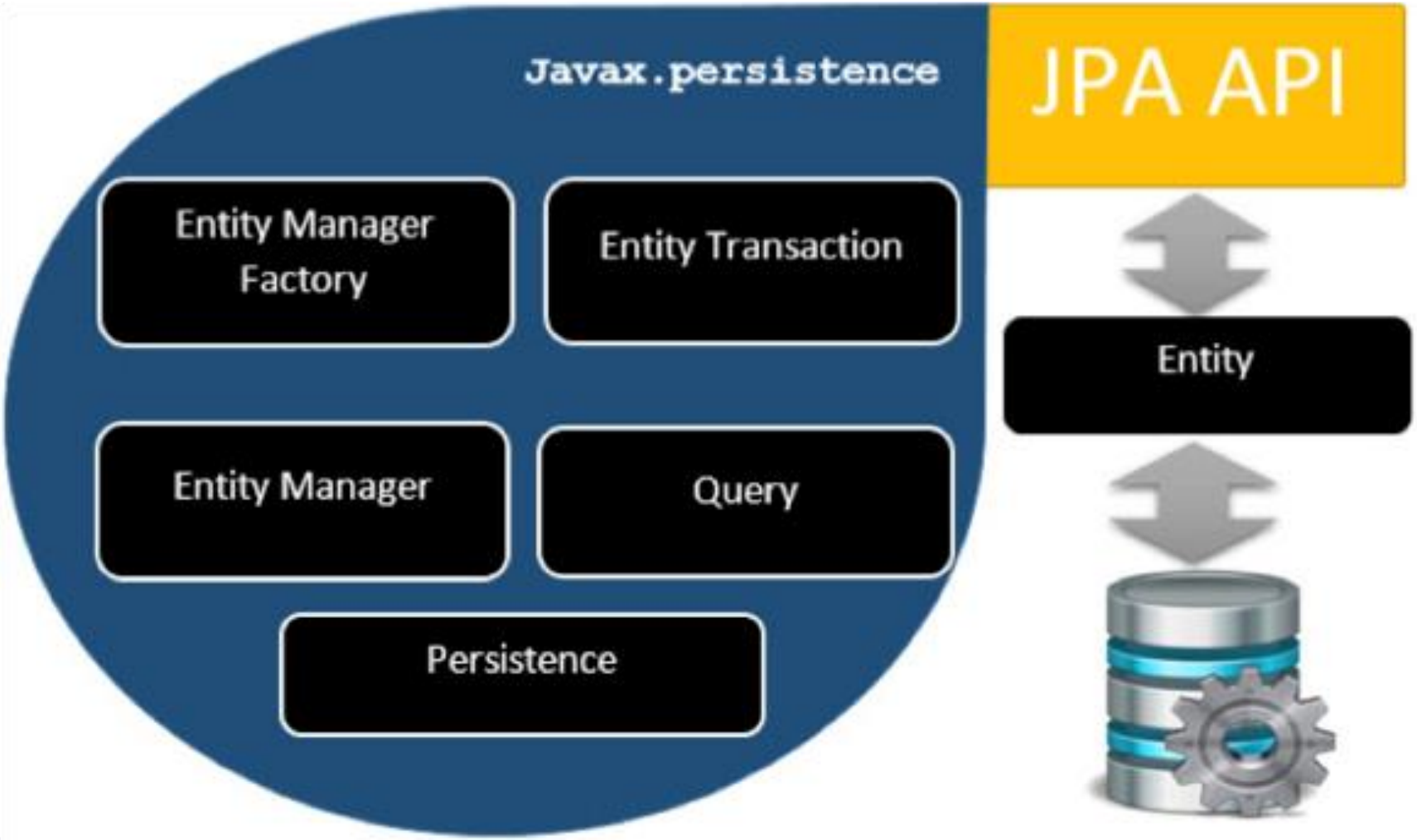
JPA – query

Veerle Ongenae

Configuratie



https://www.tutorialspoint.com/jpa/jpa_architecture.htm

Objecten opvragen

- Op basis van identifier
- Zonder identifier
  - Java Persistence Query Language
  - SQL

Java Persistence Query Language

UNIVERSITEIT GENT

- Lijkt op SQL
  - Niet hoofdlettergevoelig (behalve klassennamen en attributen)
  - Klassennamen, namen properties ↔ tabelnamen, kolomnamen
- Aanmaken zoekopdracht (`Query`)
- `javax.persistence.EntityManager`

```
Query createQuery(String queryString)
```


JPQL-zoekopdracht

UNIVERSITEIT GENT

- Voorbeeld

klasse

```
EntityManager entityManager = … ;
List<Sportclub> clubs = entityManager
    .createQuery("SELECT s FROM Sportclub s", Sportclub.class)
    .getResultList();
```

- Alle objecten van het type Sportclub ophalen en bewaren in een lijst
- Zoekopdracht uitvoeren
  - Meestal: oproepen methode getResultList()
  - Lijst van persistente objecten

JPQL-zoekopdracht met parameters

alias
eigenschappen object

```
Query opdracht
  = entityManager.createQuery("SELECT l from Lid l where l.club.naam = ?1");
String naam = "EIKENLO";
opdracht.setParameter(1, naam);
List<Lid> leden = opdracht.getResultList();
```

volgnummer

parameter

- Alternatief

```
opdracht =
  entityManager.createQuery("SELECT l from Lid l where l.club.naam = :naam");
naam = "EIKENLO";
opdracht.setParameter("naam", naam);
leden = opdracht.getResultList();
```

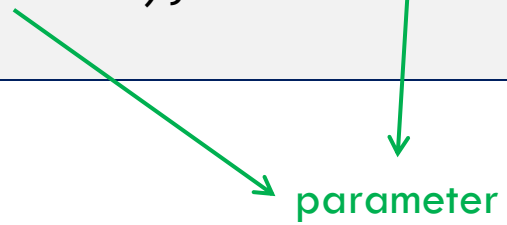
parameternaam

- Parameter
  - `?n` (telt vanaf 1) of `:naam` (een naam voor de parameter)

Zoekopdracht met Spring-repository

```
public interface UserRepository extends JpaRepository<User, Long> {

  @Query("select u from User u where u.emailAddress = ?1")
  User findByEmailAddress(String emailAddress);
}
```



JPQL

- distinct, order by

```
List<String> namen = entityManager
  .createQuery("select distinct s.naam from Sportclub s order by s.naam")
  .getResultList();
```

- count, sum, min, max, avg, group by

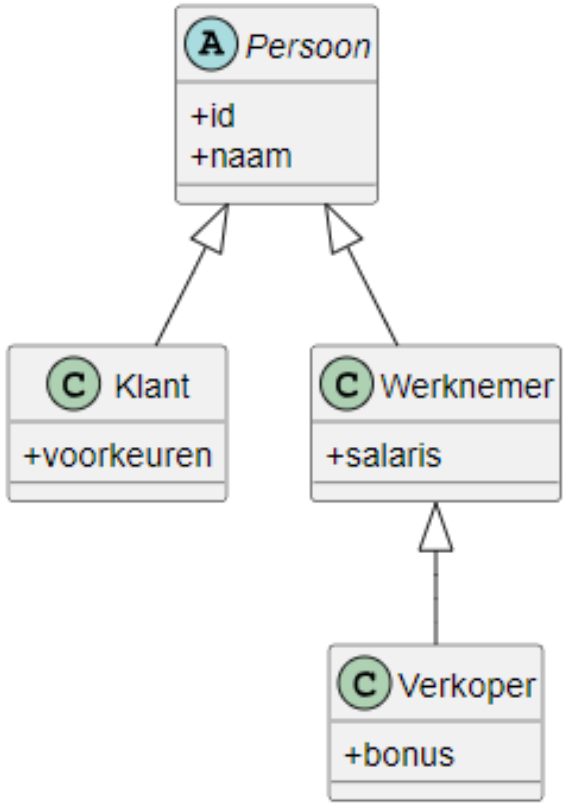
```
List overzicht = entityManager
  .createQuery("select s.naam, count(s) from Sportclub s group by s.naam having count(s) >= 2")
  .getResultList();
System.out.println("Sportclubs met een naam die minstens 2 keer voorkomt");
for (Object result : overzicht) {
  Object[] temp = (Object[])result;
  System.out.println(temp[0] + " " + temp[1]);
}
```

- Joins: inner join, left join, right join, full join

JPQL zoekopdracht met abstracte klasse

UNIVERSITEIT GENT

```
Query zoekopdracht
  = entityManager.createQuery("select p from Persoon");
List personen = opdracht.getResultList();
System.out.println("Personen");
for (Object object : personen) {
  Persoon persoon = (Persoon)object;
  System.out.println(persoon.getNaam());
}
```



Objecten opvragen

- Op basis van identifier
- Zonder identifier
  - Java Persistence Query Language
  - SQL

SQL-opdrachten – JPA

UNIVERSITEIT GENT

- Aanmaken zoekopdracht (Query)
- `Javax.persistence.EntityManager`

```
Query createNativeQuery(String sqlString, Class resultClass)
```

  - Eerste param: zoekopdracht
  - Tweede param: klasse resultaat (resultaten)

```
List<Sportclub> sportclubs = entityManager
  .createNativeQuery("select id, naam from sportclubs", Sportclub.class)
  .getResultList();
```


SQL-zoekopdracht met Spring-repository

UNIVERSITEIT GENT

```
public interface UserRepository extends JpaRepository<User, Long> {

  @Query(value = "SELECT * FROM USERS u WHERE u.status = 1", nativeQuery = true)
  Collection<User> findAllActiveUsersNative();
}
```