Les associations entre entités avec JPA

Contrôles des relations entre entités avec JPA.

Emmanuel BRUNO
Université de Toulon, LIS UMR CNRS 7020
emmanuel.bruno@univ-tln.fr

2026-11-01

Table des matières

1 One To One (1-1) 2
2 One to Many (1-N) 4
2.1 Avec deux entités 6
3 Many to Many (N-M) 8

```
[postgres-ci] starting...
[postgres-ci] OK
[maven] starting...
[maven] OK
[diag] 1/5 classpath du kernel exposée
[diag] 2/5 imports slf4j/logback OK
[diag] 3/5 contexte logback OK
[diag] 4/5 40 jars ajoutés au classpath
[diag] Postgres driver registered OK
[diag] 5/5 imports JPA/DatabaseManager OK
12:15:25.589  [IJava-executor-0]  INFO    f.u.b.d.jpa.hello.DatabaseManager  --
Database settings: URL=jdbc:postgresql://dind/notebook-db, User=dba
12:15:25.596  [IJava-executor-0]  INFO    f.u.b.d.jpa.hello.DatabaseManager  --
Starting JPA with URL: jdbc:postgresql://dind/notebook-db and User: dba
12:15:26.768  [IJava-executor-0]  INFO    com.zaxxer.hikari.HikariDataSource  --
HikariPool-1 - Starting...
12:15:26.989  [IJava-executor-0]  INFO    com.zaxxer.hikari.pool.HikariPool  --
HikariPool-1 - Added connection org.postgresql.jdbc.PgConnection@48a64560
12:15:26.994  [IJava-executor-0]  INFO    com.zaxxer.hikari.HikariDataSource  --
HikariPool-1 - Start completed.
⟶ Connexion JPA réussie (Tentative 1) !
[diag] cell init terminée ✓
```

JPA offre une prise en charge complète pour la gestion des relations entre entités de différents types, que ce soit en mode un-vers-un (OneToOne), un-vers-plusieurs (OneToMany), ou plusieurs-vers-plusieurs (ManyToMany). Ces relations peuvent être configurées de manière unidirectionnelle ou bidirectionnelle.

1 One To One (1-1)

Dans certaines situations, lors de la modélisation orientée objet, il est nécessaire d'associer deux classes dans une relation un-à-un, mais en gardant une seule relation.

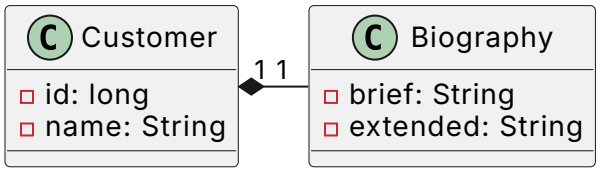


Figure 1. – Un Customer a une Biography

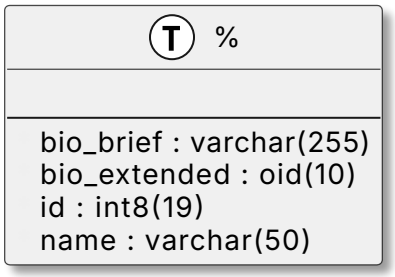


Figure 2. – L'unique relation CUSTOMER

Pour cela JPA propose des annotations spécifiques :

@Embeddable: Cette annotation indique que les membres d'une classe Java peuvent être persistés en tant qu'attributs de la classe qui les utilise. En d'autres termes, elle spécifie qu'une classe peut être utilisée comme composant réutilisable dont les propriétés peuvent être incorporées dans une autre entité.

@Embedded: Utilisée au niveau d'un membre de classe annoté avec @Embeddable, cette annotation indique que le membre annoté doit être intégré dans la relation. Cela signifie que les propriétés de la classe annotée avec @Embeddable seront incluses dans la table de la classe qui les utilise.

```
                                @Builder
                @NoArgsConstructor(access = AccessLevel.PROTECTED)
                                @AllArgsConstructor

                                    @Embeddable
                                public class Biography {
                                @Column(name = "BIO_BRIEF")
                                   private String brief;

                                            @Lob
                                @Column(name = "BIO_EXTENDED")
                                   private String extended;
                                            }
```

Figure 3. – Embeddable.java

```
@Getter
@Setter
@ToString
@RequiredArgsConstructor(staticName = "of")
@NoArgsConstructor(access = AccessLevel.PROTECTED)

@Entity
@Table(name = "CUSTOMER", schema = "ex_biography")
public class Customer {
    @Id
    @GeneratedValue
    private long id;

    @Column(length = 50, nullable = false)
    @NonNull
    private String name;

    @Embedded
    @Column()
    private Biography biography;
}
```

Customer.java

On peut donc rendre persistente une instance de `Customer` associé à une instance de `Biography`.

```java
import fr.univtln.bruno.demos.jpa.hello.samples.ex_biography.Customer;
try (EntityManager entityManager = emf.createEntityManager()) {
   entityManager.getTransaction().begin();

   Customer customer = Customer.of("Jim");
   customer.setBiography(Biography.builder()
      .brief("bla")
      .extended("bla bla")
      .build());
   entityManager.persist(customer);

   entityManager.getTransaction().commit();
}
```

dans une seule relation

id	name	bio_brief	bio_extended
1	Jim	bla	16818

Figure 4. – La table CUSTOMER

2 One to Many (1-N)

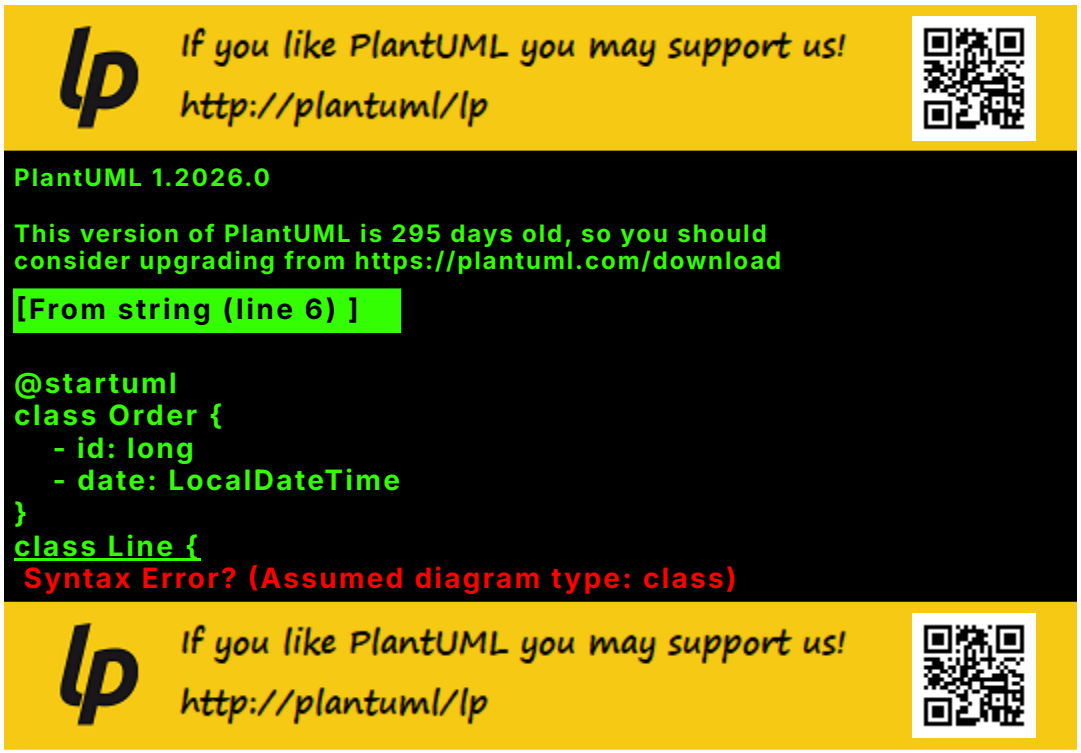


Figure 5. – Une Order a plusieurs Line

```java
@Getter
@Setter
@ToString
@AllArgsConstructor(staticName = "of")
@NoArgsConstructor(access = AccessLevel.PROTECTED)

@Embeddable
@Table(name = "LINE", schema = "EX_ONE_TO_MANY_B")
public class Line {
    @NonNull
    private String product;

    private double price;
}
```

```java
@Getter
@Setter
@ToString
@NoArgsConstructor
@Builder
@AllArgsConstructor

@Entity
@Table(name = "EORDER", schema = "EX_ONE_TO_MANY_B")
public class Order {
    @Id
    @GeneratedValue(strategy = GenerationType.SEQUENCE)
    private long id;

    @Builder.Default
    @Column(nullable = false)
    private LocalDateTime date = LocalDateTime.now();

    @ElementCollection
    @CollectionTable(name = "LINE", schema = "EX_ONE_TO_MANY_B")
    @Singular
    private Set<Line> lines = new HashSet<>();

}
```

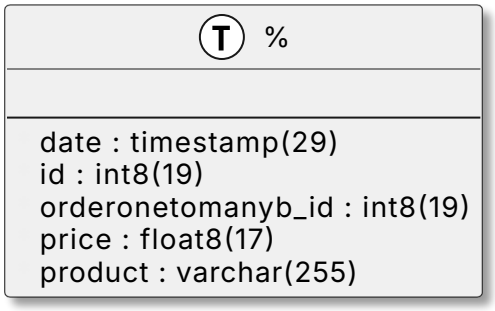


Figure 6. – Les relations LINE et EORDER

2.1 Avec deux entités

Dans le cas général, l'attribut de type `Collection<T>` est annoté avec `@OneToMany`.

L'attribut optionnel `mappedBy` indique le nom de l'attribut correspondant dans le type `T` si l'on souhaite une relation bi-directionnelle. L'autre entité (de Type `T`) est alors annotée avace `@ManyToOne`

```
@Getter
@Setter
@ToString
@NoArgsConstructor

@Entity
@Table(name = "EORDER", schema = "EX_ONE_TO_MANY_A")
public class Order {
    @Id
    @GeneratedValue(strategy = GenerationType.SEQUENCE)
    private long id;

    private LocalDateTime date = LocalDateTime.now();

    @OneToMany(mappedBy = "order",
            cascade = {CascadeType.PERSIST,
                    CascadeType.REMOVE},
            orphanRemoval = true)
    @ToString.Exclude
    private Set<Line> lines = new HashSet<>();

    public Order addLine(Line line) {
        line.setOrder(this);
        lines.add(line);
        return this;
    }
```

```
    public Order removeLine(Line line) {
        line.setOrder(null);
        lines.remove(line);
        return this;
    }
}
```

```
@Getter
@Setter
@ToString
@RequiredArgsConstructor(staticName = "of")
@NoArgsConstructor(access = AccessLevel.PROTECTED)

@Entity
@Table(name = "LINE", schema = "EX_ONE_TO_MANY_A")
public class Line {
    @Id
    @GeneratedValue(strategy = GenerationType.SEQUENCE)
    private long id;
    @NonNull
    private String product;

    @NonNull // just for requiredargsconstructor
    private double price;

    @ManyToOne
    @JoinColumn(nullable = false)
    @ToString.Exclude
    private Order order;

    public void setOrder(Order order) {
        if (this.order ≠ null) {
            this.order.getLines().remove(this);
        }
        this.order = order;
        if (order ≠ null) {
            order.getLines().add(this);
        }
    }
}
```

```
12:15:33.240    [IJava-executor-0]    INFO        notebook    --    Order(id=1,
date=2026-11-01T12:15:33.085824)    [Line(id=3,    product=Paper,    price=5.0),
Line(id=1, product=Pen, price=1.0), Line(id=2, product=Scissor, price=8.0)]
```

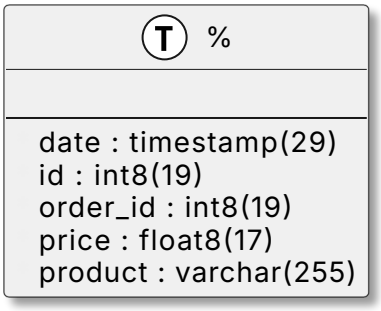


Figure 7. – Les relations LINE et EORDER

order_id	order_date	line_id	product	price
1	2026-11-01 12:15:33.085824	1	Pen	1.0
1	2026-11-01 12:15:33.085824	2	Scissor	8.0
1	2026-11-01 12:15:33.085824	3	Paper	5.0
2	2026-11-01 12:15:33.095399	4	Moto	8000.0
2	2026-11-01 12:15:33.095399	5	Car	20000.0

Figure 8. – SELECT * FROM EORDER INNER JOIN LINE

3 Many to Many (N-M)

L'annotation @ManyToMany en Java Persistence API (JPA) est utilisée pour modéliser une relation many-to-many entre deux entités. La table d'association sera alors générée et utilisée implicitement.

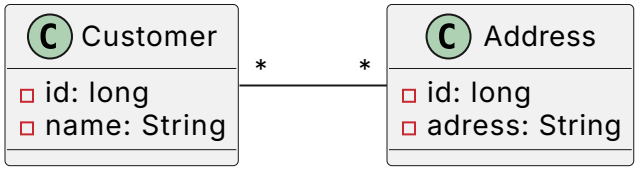


Figure 9. – Un Customer a plusieurs adresses (qui peuvent être associées à d'autres)

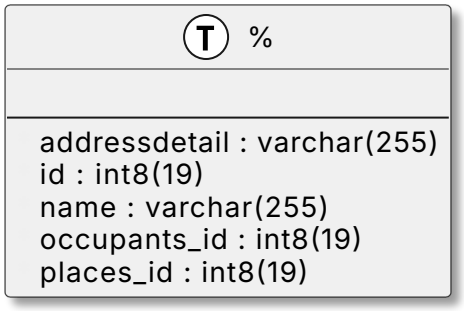


Figure 10. – Les relation CUSTOMER, ADDRESS et la relation de d'association

```java
@Getter
@Setter
@ToString
@RequiredArgsConstructor(staticName = "of")
@NoArgsConstructor(access = AccessLevel.PROTECTED)

@Entity
@Table(name = "CUSTOMER", schema = "EX_MANY_TO_MANY")
public class Customer {
    @Id
    @GeneratedValue
    private long id;

    @NonNull
    private String name;

    @ManyToMany
    @ToString.Exclude
    @JoinTable(schema = "EX_MANY_TO_MANY")
    private Set<Address> places = new HashSet<>();

    public void addPlace(Address address) {
        this.places.add(address);
        address.getOccupants().add(this);
    }

    public void removePlace(Address address) {
        this.places.remove(address);
        address.getOccupants().remove(this);
    }
}
```

```java
@Getter
@Setter
@ToString
@RequiredArgsConstructor(staticName = "of")
@NoArgsConstructor(access = AccessLevel.PROTECTED)

@Entity
@Table(name = "ADDRESS", schema = "EX_MANY_TO_MANY")
public class Address {
    @Id
    @GeneratedValue
    private long id;

    @NonNull
    private String addressDetail;

    @ManyToMany(mappedBy = "places")
    @ToString.Exclude
    private Set<Customer> occupants= new HashSet<>();

    public void addOccupant(Customer customer) {
        this.occupants.add(customer);
        customer.getPlaces().add(this);
    }

    public void removeOccupant(Customer customer) {
        this.occupants.remove(customer);
        customer.getPlaces().remove(this);
    }

}
```

```
12:15:38.591                                                [IJava-executor-0]
WARN  o.a.h.c.p.ResponseProcessCookies -- Invalid cookie header: "set-cookie:
__cf_bm=5DzMcU.qjxQK_GIizRk511U3bwCW5XyZYY_gSQqFCOk-1793535338.557569-1.0.1.1-k4CFK2cucu7VhdX15C3V
HttpOnly;   SameSite=None;   Secure;   Path=/;   Domain=repo.maven.apache.org;
Expires=Sun, 01 Nov 2026 12:45:38 GMT". Invalid 'expires' attribute: Sun, 01 Nov
2026 12:45:38 GMT
12:15:38.696  [IJava-executor-0]  WARN    o.a.h.c.p.ResponseProcessCookies  --
Invalid cookie header: "set-cookie: __cf_bm=PgQpzsmYaPgaaDJDzSUCRQA4tQ7yqeyn1yAo5chfHIA-1793535338.
UgMARsB.wgVhozZq1alIgrzDqMdclJiq7y1dfxX5BHXZaWOWF3mtK.Uz1pAEY3zI_XIQ60IYewe03BEvlYXMsExApkYXo8.X2
HttpOnly;   SameSite=None;   Secure;   Path=/;   Domain=repo.maven.apache.org;
Expires=Sun, 01 Nov 2026 12:45:38 GMT". Invalid 'expires' attribute: Sun, 01 Nov
```

```
2026 12:45:38 GMT
12:15:38.739                                                [IJava-executor-0]
WARN  o.a.h.c.p.ResponseProcessCookies -- Invalid cookie header: "set-cookie:
__cf_bm=k4VTVaS7cjbT5UxhJwWxG60oGTNwZ7uon792WLIqBW4-1793535338.716084-1.0.1.1-EX.7mSg_MUcukS4guZHD
HttpOnly;  SameSite=None;  Secure;  Path=/;  Domain=repo.maven.apache.org;
Expires=Sun, 01 Nov 2026 12:45:38 GMT". Invalid 'expires' attribute: Sun, 01 Nov
2026 12:45:38 GMT
12:15:38.780 [IJava-executor-0] WARN   o.a.h.c.p.ResponseProcessCookies  --
Invalid cookie header: "set-cookie: __cf_bm=XQhIk05hcU9KHUlvdhjL76hqbhkYKTO9keyg9dfyi1E-1793535338.
jWb9UJeJkNkAzlSBJDSjjfXXEiPfURcTmBrb873mSv5gm6JPE36Y5SdvCr97hE__rjC93VR4dO6cZZ7wWNWpn3MK_MsE6N4g9e
HttpOnly;  SameSite=None;  Secure;  Path=/;  Domain=repo.maven.apache.org;
Expires=Sun, 01 Nov 2026 12:45:38 GMT". Invalid 'expires' attribute: Sun, 01 Nov
2026 12:45:38 GMT
12:15:38.832     [BasicRepositoryConnector-repo.maven.apache.org-0-0]     WARN
o.a.h.c.p.ResponseProcessCookies -- Invalid cookie header: "set-cookie: __cf_bm=4Tt1ajjcY47hI.pV3Qsp
w6YqYura8qVUnI4LQ7D8qDp9_chnGGqCQ58QeEOgVYGQlWO4u9kwJ1NSkY_eMBpwKgFLDQjqcBsjQQX26vcL8Q31EG27NToEoS
HttpOnly;  SameSite=None;  Secure;  Path=/;  Domain=repo.maven.apache.org;
Expires=Sun, 01 Nov 2026 12:45:38 GMT". Invalid 'expires' attribute: Sun, 01 Nov
2026 12:45:38 GMT
12:15:42.210     [BasicRepositoryConnector-repo.maven.apache.org-0-3]     WARN
o.a.h.c.p.ResponseProcessCookies -- Invalid cookie header: "set-cookie: __cf_bm=Epou7hM9UD5_owKBophr
p.fIhHKGQs.x0jHSpAi9sDZNP93uMNhVdMT9g7Dpgpd78b3NbFZyftC_rZmw44Kd70vgAVCWPCDORoH7q1z7lwfaTN3T21B87W
HttpOnly;  SameSite=None;  Secure;  Path=/;  Domain=repo.maven.apache.org;
Expires=Sun, 01 Nov 2026 12:45:42 GMT". Invalid 'expires' attribute: Sun, 01 Nov
2026 12:45:42 GMT
12:15:42.278     [BasicRepositoryConnector-repo.maven.apache.org-0-1]     WARN
o.a.h.c.p.ResponseProcessCookies  --  Invalid  cookie  header:  "set-cookie:
__cf_bm=cScrTfCO7uZiFJuKG4n1lKFiCtqPbtZdQFNTuI7axmQ-1793535338.865031-1.0.1.1-zFdYw8cPy0Bn96lr5HgW
HttpOnly;  SameSite=None;  Secure;  Path=/;  Domain=repo.maven.apache.org;
Expires=Sun, 01 Nov 2026 12:45:38 GMT". Invalid 'expires' attribute: Sun, 01 Nov
2026 12:45:38 GMT
12:15:46.077     [BasicRepositoryConnector-repo.maven.apache.org-0-2]     WARN
o.a.h.c.p.ResponseProcessCookies -- Invalid cookie header: "set-cookie: __cf_bm=.WXTvJ1Io8PEe1P1X_rJ
sTth7Yyn.MAhzbIoXQ7ct6hN1dpn4rU6304FkPsV1XoQkC0UwWHz7uM7UiWzpz1611cNdm4e24jlJXX_OTlYsz5NTux9hzcfbr
HttpOnly;  SameSite=None;  Secure;  Path=/;  Domain=repo.maven.apache.org;
Expires=Sun, 01 Nov 2026 12:45:38 GMT". Invalid 'expires' attribute: Sun, 01 Nov
2026 12:45:38 GMT
```

id	name
2	Ashlyn Yost
3	Mrs. Tresa Kiehn
4	Lara Kerluke
5	Jarred Zulauf III
6	Bree Huels III

Figure 11. – La table CUSTOMER

occupants_id	places_id
2	1
4	1
6	1
2	2
3	2
4	2
5	2

Figure 12. – La table de jointure

id	addressdetail
1	41616 Tony Bridge, East Rosita, WI 44680
2	Suite 765 776 McGlynn Green, Jolieton, NJ 20735

Figure 13. – La table ADDRESS