JPA

JPA – **Java Persistence API**, est une interface de programmation (API) Java permettant de stocker de manière pérenne (persistance) les données d'un programme Java.

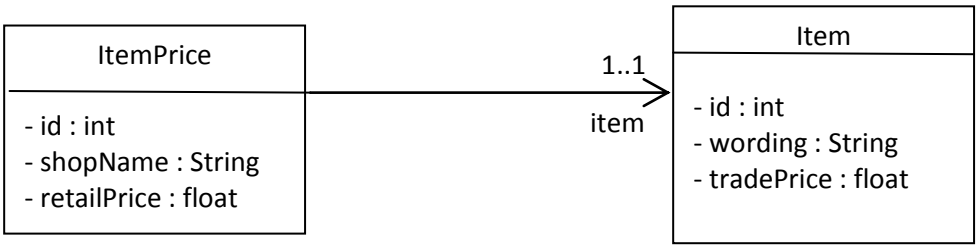
La Java Persistence API repose essentiellement sur l'utilisation des annotations java - @ -. Ces annotations permettent de définir la structure des tables dans une base de données relationnelle en fonction des classes métiers et leur annotation.

Code 1

```
@Entity
@table(name = itemPrice)
public class itemPrice {
    @Id   @GeneratedValue
    int id ;
    float retailPrice;
    String shopName;
    @ManyToOne
    @NotNull
    Item item;
          // add accessors, and
          // default constructor
}
```

```
   @Entity
   @table(name = item)
   public class Item {
      @Id
      @GeneratedValue
      int id ;
      String wording ;
      float tradePrice ;
           // add accessors, and
           // default constructor
}
```

Le diagramme de classes correspondant :



Au moment de l'insertion dans la base de données d'un « ItemPrice » le champ *item* est non nul : `@NotNull`

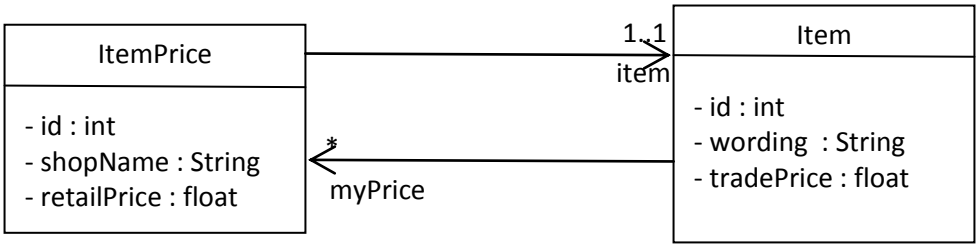
Code 2

```java
@Entity
@table(name = itemPrice)
public class itemPrice {
    @Id   @GeneratedValue
    int id ;
    float retailPrice;
    String shopName;
    @ManyToOne
    @NotNull
    Item item;
          // add accessors, and
          // default constructor
   }
```

```java
@Entity
@table(name = item)
public class Item {
   @Id
   @GeneratedValue
   int id ;
   String wording ;
   float tradePrice ;
   @OneToMany
   List<ItemPrice> myPrices;

        // add accessors, and
        // default constructor
}
```

Le diagramme de classes correspondant :



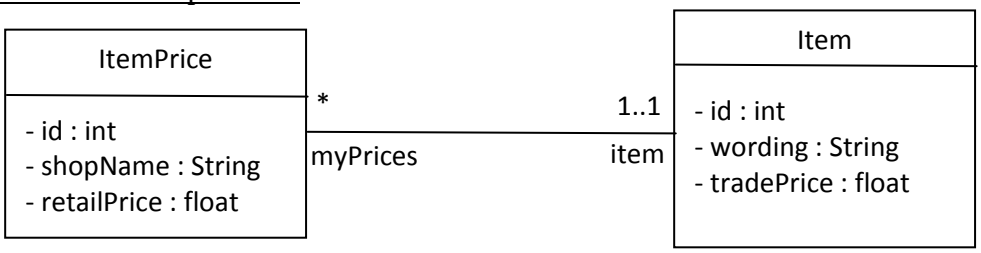
Il y a deux associations unidirectionelles.

Code 3

```
@Entity
@table(name = itemPrice)
public class itemPrice {
    @Id   @GeneratedValue
    int id ;
    float retailPrice;
    String shopName;
    @ManyToOne
    @NotNull
    Item item;
          // add accessors, and
          // default constructor
   }
```

```
@Entity
@table(name = item)
public class Item {
   @Id
   @GeneratedValue
   int id ;
   String wording ;
   float tradePrice ;
   @OneToMany(
        mappedBy = "item",
        orphanRemoval = true,
   cascade =
   CascadeType.REMOVE)
   List<ItemPrice> myPrices;
        // add accessors, and
        // default constructor
}
```

Le diagramme de classes correspondant



L'association est bidirectionnelle [mappedBy = "item"].

Retirer un enregistrement correspondant à un « item » dans la base de données a pour conséquence de retire dans la base de données tous les prix associés à « item » (enregistrement de la table « itemPrice ») [cascade = CascadeType.REMOVE].

Un « ItemPrice » qui n'est plus dans la liste myPrices de l'item concerné est retiré de la base de données (au moment de la mise à jour de cette dernière) [orphanRemoval = true].