

CLASE DEPARTAMENT

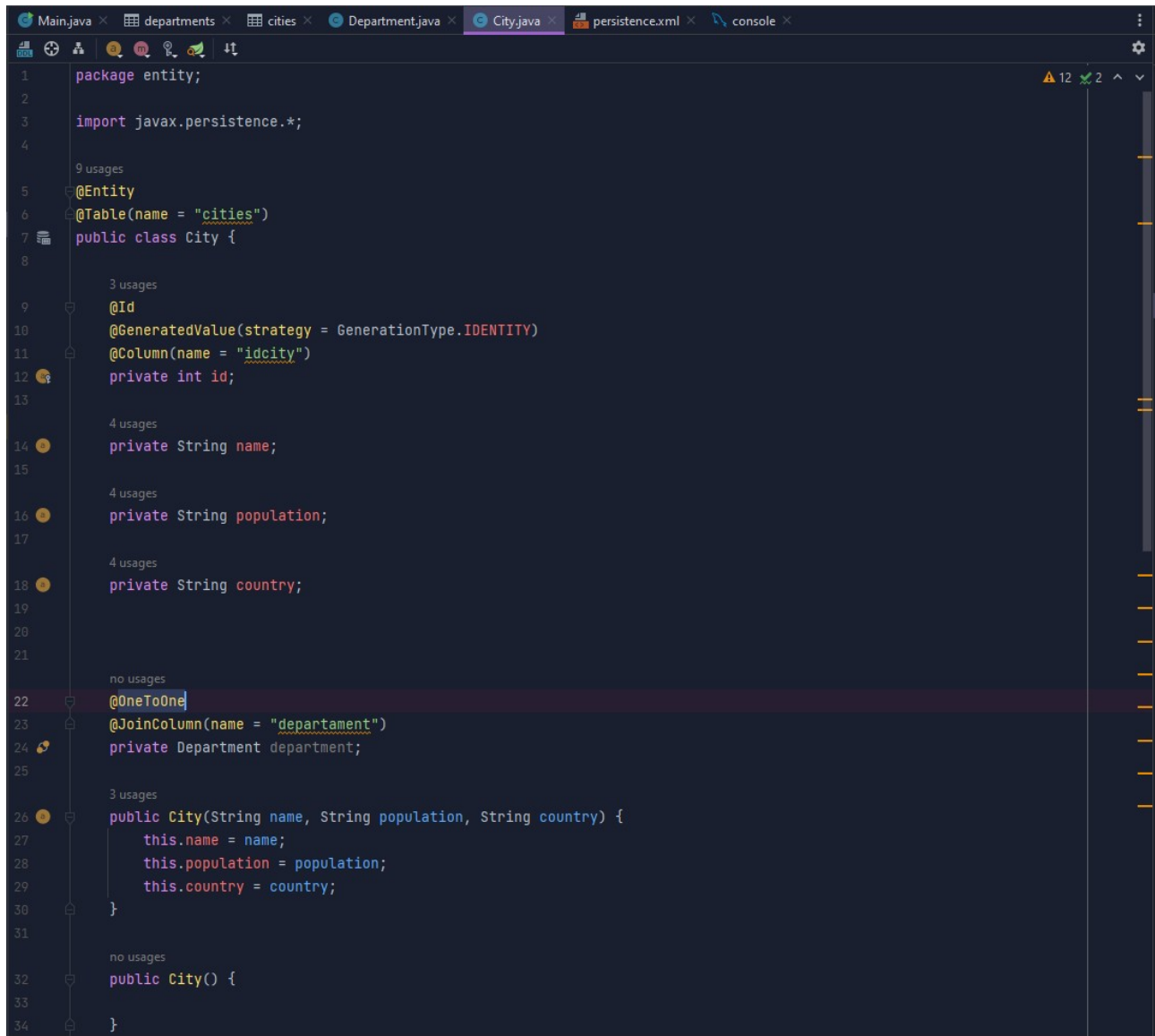
Creamos la relación uno uno y la unimos con la ciudad

```
1 package entity;
2
3 import ..
4
5
6
7 10 usages
8 @Entity
9 @Table(name="departments")
10 public class Department {
11
12 1 usage
13 @Id
14 @GeneratedValue(strategy = GenerationType.IDENTITY )
15 @Column(name="iddepartment")
16 private int id;
17
18 3 usages
19 private String name;
20
21 4 usages
22 @ManyToMany(mappedBy = "ldepartments")
23 private List<Employee> lemployees;
24
25
26 2 usages
27 @OneToOne
28 @JoinColumn(name = "idcity")
29 private City city;
30
31
32 no usages
33 public List<Employee> getLemployees() {
34     return lemployees;
35 }
36
37
38 no usages
39 public void setLemployees(List<Employee> lemployees) {
40     this.lemployees = lemployees;
41 }
42
43
44 no usages
45 public City getCity() {
46     return city;
47 }
48
49
50 3 usages
51 public void setCity(City city) {
52     this.city = city;
53 }
```

```
37
38 3 usages
39 public Department(String name) {
40     this.name = name;
41     lemployees = new ArrayList<Employee>();
42 }
43
44 no usages
45 public Department() {
46 }
47
48 no usages
49 public int getId() { return id; }
50
51 no usages
52 public String getName() { return name; }
53
54 no usages
55 public void setName(String name) { this.name = name; }
56
57
58 1 usage
59 public void addEmployee(Employee e) {
60     lemployees.add(e);
61     e.addDepartment( d: this);
62 }
63
```

CLASE CITY

Hacemos lo mismo que con department



```
1 package entity;
2
3 import javax.persistence.*;
4
5 9 usages
6 @Entity
7 @Table(name = "cities")
8 public class City {
9
10 3 usages
11 @Id
12 @GeneratedValue(strategy = GenerationType.IDENTITY)
13 @Column(name = "idcity")
14 private int id;
15
16 4 usages
17 private String name;
18
19 4 usages
20 private String population;
21
22 4 usages
23 private String country;
24
25 no usages
26 @OneToOne
27 @JoinColumn(name = "departament")
28 private Department department;
29
30 3 usages
31 public City(String name, String population, String country) {
32     this.name = name;
33     this.population = population;
34     this.country = country;
35 }
36
37 no usages
38 public City() {
39 }
40 }
```

```

32     no usages
33     public City() {
34     }
35
36     no usages
37     public int getId() { return id; }
38
39
40     no usages
41     public void setId(int id) { this.id = id; }
42
43
44     no usages
45     public String getName() { return name; }
46
47
48     no usages
49     public void setName(String name) { this.name = name; }
50
51
52     no usages
53     public String getPopulation() { return population; }
54
55
56     no usages
57     public void setPopulation(String population) { this.population = population; }
58
59
60     no usages
61     public String getCountry() { return country; }
62
63
64     no usages
65     public void setCountry(String country) { this.country = country; }
66
67
68
69
70     @Override
71     public String toString() {
72         return "City{" +
73             "id=" + id +
74             ", name='" + name + '\'' +
75             ", population=" + population + '\'' +
76             ", country='" + country + '\'' +
77             ", departmentList=" +
78             '}';

```

MAIN

Añadimos ciudades y departamentos y le añadimos una ciudad a cada departamento

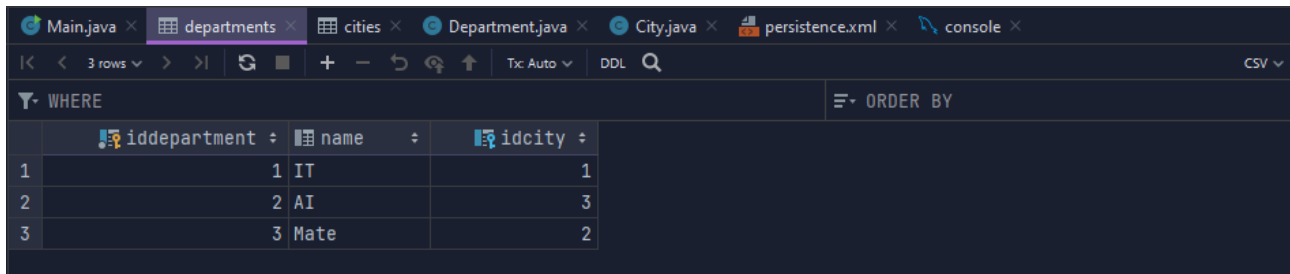
```
1  import ...
6
7  no usages
8  public class Main {
9      no usages
10     public static void main(String[] args) {
11         try {
12             City city = new City( name: "Pontevedra", population: "10000", country: "Espanha");
13             City city1 = new City( name: "Vigo", population: "2000", country: "Espanha");
14             City city2 = new City( name: "Madrid", population: "29345930", country: "Espanha");
15
16             Department department = new Department( name: "IT");
17             Department department1 = new Department( name: "AI");
18             Department department2 = new Department( name: "Mate");
19
20             department.setCity(city);
21             department1.setCity(city2);
22             department2.setCity(city1);
23
24             EntityManagerFactory emf = Persistence.createEntityManagerFactory( persistenceUnitName: "default");
25             EntityManager em = emf.createEntityManager();
26             em.getTransaction().begin();
27
28             em.persist(city);
29             em.persist(city1);
30             em.persist(city2);
31
32             em.persist(department);
33             em.persist(department1);
34             em.persist(department2);
35
36             em.getTransaction().commit();
37             em.close();
38             emf.close();
39         }
40     }
41 }
```

PERSISTENCE

Caambie el usuario, la contraseña y puse update

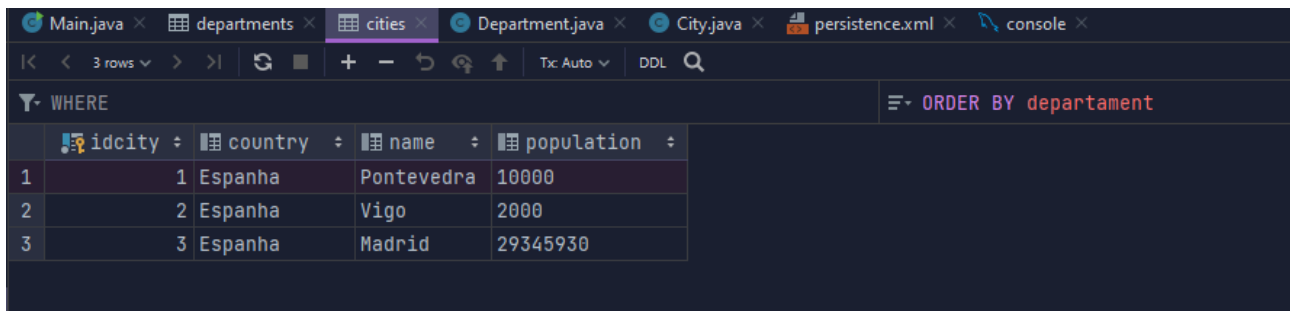
```
1  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2  <persistence xmlns="http://xmlns.jcp.org/xml/ns/persistence"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.org/xml/ns/persistence/persistence_2_2.xsd"
5      version="2.2">
6      <persistence-unit name="default">
7          <properties>
8              <property name="javax.persistence.jdbc.driver" value="com.mysql.cj.jdbc.Driver"/>
9              <property name="javax.persistence.jdbc.url" value="jdbc:mysql://127.0.0.1:3306/ac_ud3"/>
10             <property name="javax.persistence.jdbc.user" value="root"/>
11             <property name="javax.persistence.jdbc.password" value="root"/>
12             <property name="hibernate.dialect" value="org.hibernate.dialect.MySQL8Dialect"/>
13             <property name="hibernate.hbm2ddl.auto" value="update"/>
14         </properties>
15     </persistence-unit>
16 </persistence>
```

TABLA DEPARTMENT



	iddepartment	name	idcity
1	1	IT	1
2	2	AI	3
3	3	Mate	2

TABLA CITY



	idcity	country	name	population
1	1	Espanha	Pontevedra	10000
2	2	Espanha	Vigo	2000
3	3	Espanha	Madrid	29345930

BASE DE DATOS

