DAY 4
Implementation Of Relationship In Hibernate

Hibernate mappings are one of the key features of hibernate . they establish the relationship between two database tables as attributes in your model. that allows you to easily navigate the associations in your model and criteria queries.

you can establish either unidirectional or bidirectional i.e you can either model them as an attribute on only one of the associated entities or on both. it will not impact your database mapping tables, but it defines in which direction you can use the relationship in your model and criteria queries.

the relationship that can be established between entities are-

- **one to one** — It represents the one to one relationship between two tables.
- **one to many/many to one** — It represents the one to many relationship between two tables.
- **many to many** — It represents the many to many relationship between two tables.

Now let’s understand these relationships with scenario of Food delivery app for that will consider two table one is customer and one more is customer_details to work with

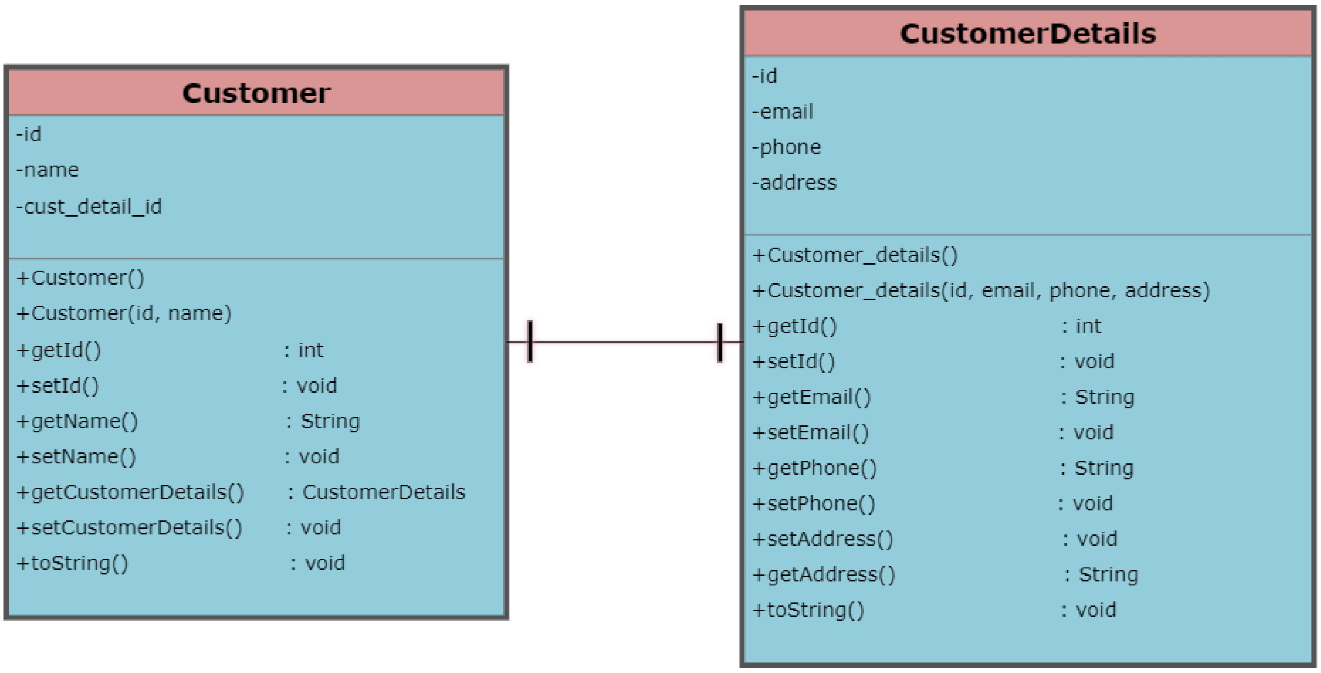
Scenario 1:

Consider customer and customer_details table with all the entities shown below

Customer
Id
Name
Cust_detail_id

Customer_details
Id
email
Phone
address

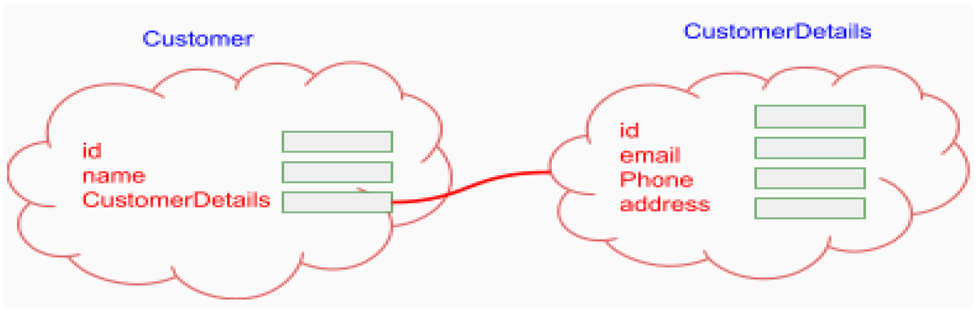
Class diagram for the above table is as shown below



In the above class diagram cust_detail_id in Customer is the foreign key to Customer Details

Steps to establish 1:1 relationship between two Module Classes

- Create POJO class of Customer with id and name attributes with necessary annotations
- Create CustomerDetails with id, email, phone, address and add necessary annotations
- Now to establish relationship between two module class we will add customerDetails field in Customer class which is the object reference of CustomerDetails



- To map one entity with another entity, **@OneToOne** annotations on the related entity field, CustomerDetails. For the CustomerDetail field mention the cascade type, **“cascade=CascadeType.ALL”** essentially **CascadeType.ALL** means that any change which happens on a Customer Entity must cascade to CustomerDetails Entity as well.

  @OneToOne(cascade = cascadeType.ALL)

  private CustomerDetails customerDetails;

- Add annotation **@JoinColumn(name = “cust_detail_id”)** to oneToOne annotations, here @JoinColumn is used to map cust_detail_id to primary key of CustomerDetails class **Customer**{

  @OneToOne(cascade = cascadeType.ALL)

  @JoinColumn(name = "cust_detail_id")

  private CustomerDetails customerDetails;

}

Customer.java

```
package com.tap.app;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToOne;
import javax.persistence.Table;

@Entity
@Table(name = "customer")
public class Customer {

     @Id
     @Column(name = "id")
     private int id;

     @Column(name = "name")
     private String name;

     @OneToOne(cascade = CascadeType.ALL)
     @JoinColumn(name = "cust_detail_id")
     private CustomerDetails customerDetails;

     public Customer() {
     }
```

```java
	public Customer(int id, String name) {
		super();
		this.id = id;
		this.name = name;
	}

	public int getId() {
		return id;
	}

	public void setId(int id) {
		this.id = id;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
	}


	public CustomerDetails getCustomerDetails() {
		return customerDetails;
	}

	public void setCustomerDetails(CustomerDetails 
customerDetails) {
		this.customerDetails = customerDetails;
	}

	@Override
	public String toString() {
		return  id + ", " + name + ", " + 
customerDetails ;
```

```
     }
}
```

CustomerDetails.java

```
package com.tap.app;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "customer_details")
public class CustomerDetails {

     @Id
     @Column(name = "id")
     private int id;

     @Column(name = "email")
     private String email;

     @Column(name = "phone")
     private String phone;

     @Column(name = "address")
     private String address;

     public CustomerDetails() {
     }

     public CustomerDetails(int id, String email, String
phone, String address) {
          super();
          this.id = id;
```

```java
        this.email = email;
        this.phone = phone;
        this.address = address;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getPhone() {
        return phone;
    }

    public void setPhone(String phone) {
        this.phone = phone;
    }

    public String getAddress() {
        return address;
    }

    public void setAddress(String address) {
        this.address = address;
    }
    @Override
    public String toString() {
        return "CustomerDetails [" + id + ", " + email
+ ", " + phone + ", " + address + "]";
    }

}
```

Now let’s try to insert the data into Customer and CustomerDetails using Customer object.

Logic:

```
Customer c1 = new Customer(1, "alex"); // Create new
Customer Object

CustomerDetails cd1 = new CustomerDetails(501,
"alex@gmail.com", "789999909", "Bangalore"); // Create
new CutomerDetails Object

c1.setCustomerDetails(cd1); //Using Customer object call
setter method of CustomerDetails to set Customer details
object

Serializable i = session.save(c1); //session.save() will
save the object into the database
```

OneToOne.java

```
package com.tap.model;
import java.io.Serializable;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

import com.tap.app.Customer;
import com.tap.app.CustomerDetails;

public class OneToOne {
     public static void main(String[] args) {
          SessionFactory sessionFactory = null;
          Session session = null;

          try {
//             Create session Factory
               sessionFactory = new Configuration()
                         .configure()

.addAnnotatedClass(Customer.class)

.addAnnotatedClass(CustomerDetails.class)
                         .buildSessionFactory();
//             Create session
               session = sessionFactory.openSession();

//             Create Transaction
               Transaction transaction =
session.beginTransaction();

//             CRUD Operation
               Customer c1 = new Customer(1, "alex");
               CustomerDetails cd1 = new
CustomerDetails(501, "alex@gmail.com", "789999909",
"Bangalore");

               c1.setCustomerDetails(cd1);

               Serializable i = session.save(c1);
               System.out.println(i);

               transaction.commit();
          } finally {
//            Closing Resources
               session.close();
               sessionFactory.close();

          }
     }
}
```

Now in the database data of Customer and CustomerDetails table is been inserted as shown below

Customer table

id	**name**	**cust_detail_id**
1	alex	501

CustomerDetails table

id	**address**	**email**	**phone**
501	Bangalore	alex@gmail.com	789999909