

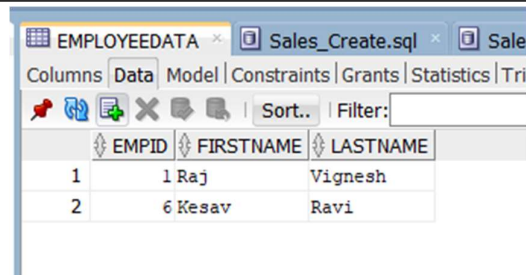
Hibernate Assignment

By Raj Vignesh Karunakaran

1. Created Employee Data Table

```
public class employeeData {  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.TABLE)  
    private int empId;  
  
    @Column  
    private String firstName;  
  
    @Column  
    private String lastName;  
  
}
```

```
employeeData emp1 = new employeeData("Raj", "Vignesh");  
employeeData emp2 = new employeeData("Kesav", "Ravi");
```

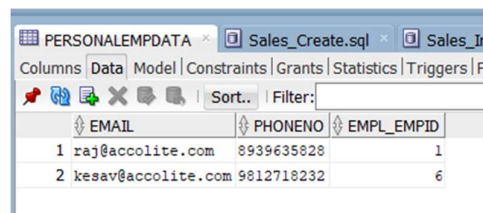


	EMPID	FIRSTNAME	LASTNAME
1	1	Raj	Vignesh
2	6	Kesav	Ravi

2. One to One mapping with Personal Employee Data

```
public class personalEmpData implements Serializable {  
  
    @Column  
    private String phoneNo;  
  
    @Column  
    private String email;  
  
    @OneToOne  
    @Id  
    @MapsId  
    private employeeData empl;  
  
}
```

```
//OneToOne  
personalEmpData personalEmp1 = new personalEmpData( ph: "8939635828", email: "raj@accolite.com", emp1);  
personalEmpData personalEmp2 = new personalEmpData( ph: "9812718232", email: "kesav@accolite.com", emp2);
```

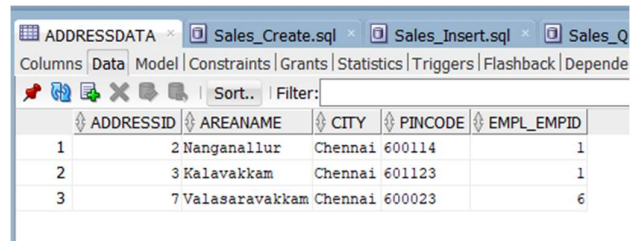


	EMAIL	PHONENO	EMPL_EMPID
1	raj@accolite.com	8939635828	1
2	kesav@accolite.com	9812718232	6

3. One to Many with Address Data

```
public class addressData {  
    @Id  
    @GeneratedValue(strategy = GenerationType.TABLE)  
    private int addressId;  
  
    @Column  
    private String areaName;  
  
    @Column  
    private String city;  
  
    @Column  
    private String pincode;  
  
    @ManyToOne(cascade = CascadeType.ALL)  
    private employeeData emp1;  
}
```

```
addressData add1 = new addressData( areaName: "Nanganallur", city: "Chennai", pincode: "600114", emp1);  
addressData add2 = new addressData( areaName: "Kalavakkam", city: "Chennai", pincode: "601123", emp1);  
  
addressData add3 = new addressData( areaName: "Valasaravakkam", city: "Chennai", pincode: "600023", emp2);  
  
emp1.getAddList().add(add1);  
emp1.getAddList().add(add2);  
emp2.getAddList().add(add3);
```



ADDRESSID	AREANAME	CITY	PINCODE	EMPL_EMPID
1	2 Nanganallur	Chennai	600114	1
2	3 Kalavakkam	Chennai	601123	1
3	7 Valasaravakkam	Chennai	600023	6

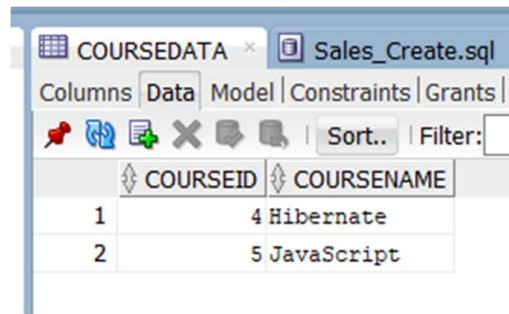
4. Many to Many with Course Data

```
public class courseData {  
    @Id  
    @GeneratedValue(strategy = GenerationType.TABLE)  
    private int courseId;  
  
    @Column  
    private String courseName;  
  
    @ManyToMany(mappedBy = "course")  
    private List<employeeData> empList = new ArrayList<>();  
}
```

```
@ManyToMany(cascade = CascadeType.ALL)  
@JoinTable(  
    name = "Courses_taken",  
    joinColumns = @JoinColumn(name = "Empl_id"),  
    inverseJoinColumns = @JoinColumn(name = "Course_id"))  
private Collection<courseData> course = new HashSet<>();  
  
public void addCourse(courseData courseInst){  
    this.course.add(courseInst);  
    courseInst.getEmpList().add(this);  
}
```

```
//ManyToMany  
courseData course1 = new courseData( cName: "Hibernate");  
courseData course2 = new courseData( cName: "JavaScript");  
  
emp1.addCourse(course1);  
emp1.addCourse(course2);  
emp2.addCourse(course1);  
emp2.addCourse(course2);
```

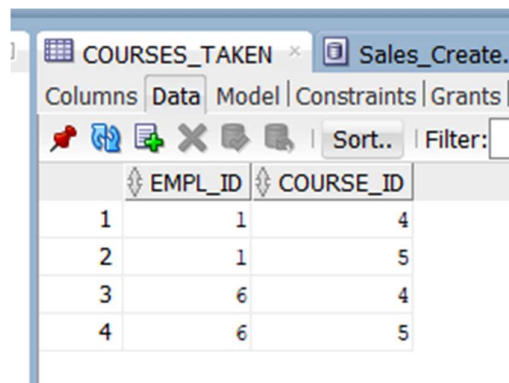
a. Course Data



The screenshot shows the SQL Developer interface with the 'COURSEDATA' table selected. The 'Data' tab is active, displaying two rows of data. The columns are 'COURSEID' and 'COURSENAME'. The first row has '1' for COURSEID and '4 Hibernate' for COURSENAME. The second row has '2' for COURSEID and '5 JavaScript' for COURSENAME.

	COURSEID	COURSENAME
1	4	Hibernate
2	5	JavaScript

b. Join Table



The screenshot shows the SQL Developer interface with the 'COURSES_TAKEN' table selected. The 'Data' tab is active, displaying four rows of data. The columns are 'EMPL_ID' and 'COURSE_ID'. The rows are: (1, 4), (2, 5), (3, 4), and (4, 5).

	EMPL_ID	COURSE_ID
1	1	4
2	1	5
3	6	4
4	6	5

Disclaimer : GenerationType.IDENTITY did not work with OracleDB thus the ID ordering is not proper.