

```
1 package model;
2
3 import jakarta.persistence.*;
4
5 @Entity
6 @Table(name="USER")
7 public class User {
8     @Column(name="USER_ID")
9     @Id
10    @GeneratedValue(strategy = GenerationType.
IDENTITY)
11    private Integer id;
12    private String fullName;
13    private String email;
14    private String password;
15    private int age;
16    private double salary;
17    private String city;
18
19    public User(String fullName, String email, String
password, int age, double salary, String city) {
20        this.fullName = fullName;
21        this.email = email;
22        this.password = password;
23        this.age = age;
24        this.salary = salary;
25        this.city = city;
26    }
27
28    public User(){
29
30    }
31
32    public Integer getId() {
33        return id;
34    }
35
36    public void setId(Integer id) {
37        this.id = id;
38    }
39
```

```
40     public String getFullName() {
41         return fullName;
42     }
43
44     public void setFullName(String fullName) {
45         this.fullName = fullName;
46     }
47
48     public String getEmail() {
49         return email;
50     }
51
52     public void setEmail(String email) {
53         this.email = email;
54     }
55
56     public String getPassword() {
57         return password;
58     }
59
60     public void setPassword(String password) {
61         this.password = password;
62     }
63
64     public int getAge() {
65         return age;
66     }
67
68     public void setAge(int age) {
69         this.age = age;
70     }
71
72     public double getSalary() {
73         return salary;
74     }
75
76     public void setSalary(double salary) {
77         this.salary = salary;
78     }
79
80     public String getCity() {
```

```
81         return city;
82     }
83
84     public void setCity(String city) {
85         this.city = city;
86     }
87 }
88
```

```
1 package controller;
2
3 import jakarta.persistence.TypedQuery;
4 import model.User;
5 import org.hibernate.Session;
6 import org.hibernate.SessionFactory;
7 import org.hibernate.Transaction;
8 import org.hibernate.cfg.Configuration;
9
10 import java.util.List;
11
12 public class UserController {
13     public static void main(String[] args) {
14         SessionFactory factory = new Configuration().
            configure().buildSessionFactory();
15         Session session = factory.openSession();
16         //      addUser(factory,session);
17         //      findUser(factory,session,3);
18         //
19         //      updateUser(session,3);
20         //
21         //      deleteUser(session,4);
22
23         //      findUserHql(factory, session);
24         //      getRecordbyId(factory, session);
25
26         //      getRecords(session);
27         //      getMaxSalary(session);
28         namedQueryExample(session);
29         factory.close();
30
31
32         session.close();
33     }
34     public static void namedQueryExample(Session
        session){
35         String hql = "FROM User u WHERE u.id = :id";
36         TypedQuery<User> query = session.createQuery(
            hql, User.class);
37         query.setParameter("id", 2);
38         List<User> result = query.getResultList();
```

```
39
40         System.out.printf("%s%13s%17s%34s%21s%n", "|
    User Id", "|Full name", "|Email", "|Password", "|
    Salary");
41         for(User u: result){
42             System.out.printf(" %-10d %-20s %-30s %-
23s %s %n", u.getId(), u.getFullName(), u.getEmail
    (), u.getPassword(), u.getSalary());
43         }
44     }
45     public static void findUserHql(SessionFactory
    factory, Session session){
46         String hqlFrom = "FROM User";
47         String hqlSelect = "SELECT u FROM User u";
48         TypedQuery<User> query = session.createQuery(
    hqlFrom, User.class);
49         List<User> results = query.getResultList();
50
51         System.out.printf("%s%13s%17s%34s%n", "|User
    Id", "|Full name", "|Email", "|Password");
52
53         for(User u: results){
54             System.out.printf(" %-10d %-20s %-30s %s
    %n", u.getId(), u.getFullName(), u.getEmail(), u.
    getPassword());
55         }
56     }
57
58     public static void addUser(SessionFactory factory
    , Session session) {
59         Transaction transaction = session.
    beginTransaction();
60         User uOne = new User();
61         uOne.setEmail("haseeb@gmail.com");
62         uOne.setFullName("Moh Haseeb");
63         uOne.setPassword("has123");
64         uOne.setSalary(2000.69);
65         uOne.setAge(20);
66         uOne.setCity("NYC");
67
68         User uTwo = new User();
```

```

69         uTwo.setEmail("James@gmail.com");
70         uTwo.setFullName("James Santana");
71         uTwo.setPassword("James123");
72         uTwo.setSalary(2060.69);
73         uTwo.setAge(25);
74         uTwo.setCity("Dallas");
75
76         User uThree = new User();
77         uThree.setEmail("Shahparan@gmail.com");
78         uThree.setFullName("AH Shahparan");
79         uThree.setPassword("Shahparan123");
80         uThree.setSalary(3060.69);
81         uThree.setAge(30);
82         uThree.setCity("Chicago");
83
84         User uFour = new User("Christ", "christ@
gmail.com", "147852", 35, 35000.3, "NJ");
85         User uFive = new User("Sid", "Sid", "s258",
29, 4000.36, "LA");
86
87         session.persist(uOne);
88         session.persist(uTwo);
89         session.persist(uThree);
90         session.persist(uFour);
91         session.persist(uFive);
92
93         transaction.commit();
94         System.out.println("successfully saved");
95         factory.close();
96         session.close();
97
98
99     }
100
101     public static void findUser(SessionFactory
factory, Session session, int userId) {
102         Transaction tx = session.beginTransaction();
103
104         User u = session.get(User.class, userId);
105         System.out.println("FullName: " + u.
getFullName());

```

```
106         System.out.println("Email: " + u.getEmail  
    ());  
107         System.out.println("password: " + u.  
    getPassword());  
108         tx.commit();  
109         factory.close();  
110         session.close();  
111     }  
112  
113     public static void updateUser(Session session,  
    int userId) {  
114         // Todo comment out findUser method and  
    uncomment updateUser method  
115  
116         Transaction tx = session.beginTransaction();  
117         User u = new User();  
118         u.setId(userId);  
119         u.setEmail("mhaseeb@perscholas");  
120         u.setFullName("M Haseeb");  
121         u.setPassword("123456");  
122         session.merge(u);  
123         session.getTransaction().commit();  
124         session.close();  
125     }  
126  
127     public static void deleteUser(Session session,  
    int userId) {  
128         // Todo comment out updateUser method and  
    uncomment deleteUser method  
129  
130         SessionFactory factory = new Configuration  
    ().configure().buildSessionFactory();  
131         Transaction tx = session.beginTransaction();  
132         User u = new User();  
133         u.setId(userId);  
134         session.remove(u);  
135         tx.commit();  
136         session.close();  
137         factory.close();  
138     }  
139
```

```

140 //      public static void findUserHql(SessionFactory
      factory, Session session) {
141 //          String hqlForm = "FROM User";
142 //          String hqlSelect = "SELECT u FROM User u";
143 //          TypedQuery<User> query = session.
      createQuery(hqlSelect, User.class);
144 //          List<User> results = query.getResultList
      ();
145 //          System.out.printf("%s%13s%17s%34s\n", "|
      User Id", "|Full name", "|Email", "|Password");
146 //          for (User u : results) {
147 //              System.out.printf(" %-10d %-20s %-30s
      %s %n", u.getId(), u.getFullName(), u.getEmail(), u
      .getPassword());
148 //          }
149 //      }
150
151      public static void getRecordbyId(SessionFactory
      factory, Session session) {
152          String hql = "FROM User u WHERE u.id > 2
      ORDER BY u.salary DESC";
153          TypedQuery<User> query = session.createQuery
      (hql, User.class);
154          List<User> results = query.getResultList();
155          System.out.printf("%s%13s%17s%34s%21s\n", "|
      User Id", "|Full name", "|Email", "|Password", "|
      Salary");
156          for (User u : results) {
157              System.out.printf(" %-10d %-20s %-30s %-
      23s %s %n", u.getId(), u.getFullName(), u.getEmail
      (),
158                      u.getPassword(), u.getSalary());
159          }
160      }
161
162      public static void getRecords(Session session) {
163          TypedQuery<Object[]> query = session.
      createQuery("SELECT U.salary, U.fullName FROM User
      AS U", Object[].class);
164          List<Object[]> results = query.getResultList
      ();

```



```
165         System.out.printf("%s%13s%n", "Salary", "City"
166     );
167         for(Object[] a: results){
168             System.out.printf("%-16s%s%n", a[0], a[1
169         ]);
170     }
171     }
172     public static void getMaxSalary(Session session
173 ){
174     String hql = "SELECT max(U.salary) FROM User
175     U";
176     TypedQuery<Object> query = session.
177     createQuery(hql, Object.class);
178     Object result = query.getSingleResult();
179     System.out.printf("%s%s", "Maximum Salary:",
180     result);
181 }
```

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE hibernate-configuration PUBLIC
3     "-//Hibernate/Hibernate Configuration DTD 3.0
4     //EN"
5     "http://www.hibernate.org/dtd/hibernate-
6     configuration-3.0.dtd">
7 <hibernate-configuration>
8     <session-factory>
9         <!-- Drop and re-create the database on
10         startup -->
11         <property name="hibernate.hbm2ddl.auto">
12             update </property>
13
14         <!-- Database connection settings -->
15         <property name="connection.driver_class">com.
16             mysql.cj.jdbc.Driver</property>
17         <property name="connection.url">jdbc:mysql://
18             localhost:3306/usersDb?createDatabaseIfNotExist=true
19             </property>
20         <property name="connection.username">root</
21             property>
22         <property name="connection.password">
23             SzhengSQL123</property>
24
25         <!-- MySQL DB dialect -->
26         <property name="dialect">org.hibernate.
27             dialect.MySQLDialect</property>
28         <!-- print all executed SQL on console -->
29         <property name="hibernate.show_sql" >true </
30             property>
31         <property name="hibernate.format_sql" >true
32             </property>
33
34         <!-- Mapping entity file -->
35         <mapping class="model.User"/>
36
37     </session-factory>
38 </hibernate-configuration>
```