Spring Data JPA with Hibernate Part 2

JPQL:

1. Display the first name, last name of all employees having salary greater than average salary ordered in ascending by their age and in descending by their salary.
2. Update salary of all employees by a salary passed as a parameter whose existing salary is less than the average salary.
3. Delete all employees with minimum salary.

Native SQL Query:

1. Display the id, first name, age of all employees where last name ends with "singh"
2. Delete all employees with age greater than 45(Should be passed as a parameter)

File=Employee.java

```
package com.bootcamp.JPA.Part2.entities;

import javax.persistence.*;

@Entity
@Table(name = "employee")
public class Employee
{
  @Id
  @Column(name = "empid")
  @GeneratedValue(strategy = GenerationType.IDENTITY)
  private  int id;

  @Column(name = "empfirstname")
  private String firstname;
  @Column(name = "emplastname")
  private String lastname;
  @Column(name = "empsalary")
  private int salary;
  @Column(name = "empage")
  private  int age;
```

```java
public int getId() {
    return id;
}

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

public String getFirstName() {
    return firstname;
}

public void setFirstName(String firstname) {
    this.firstname = firstname;
}

public String getLastName() {
    return lastname;
}

public void setLastName(String lastname) {
    this.lastname = lastname;
}

public int getSalary() {
    return salary;
}

public void setSalary(int salary) {
    this.salary = salary;
}

public int getAge() {
    return age;
}

public void setAge(int age) {
    this.age = age;
}

@Override
public String toString() {
    return "Employee{" +
            "id=" + id +
            ", firstName='" + firstname + '\'' +
            ", lastName='" + lastname + '\'' +
            ", salary=" + salary +
            ", age=" + age +
            '}';
}
```

```java
}

File=EmployeeRepository.java

package com.bootcamp.JPA.Part2.repositry;

import com.bootcamp.JPA.Part2.entities.Employee;
import org.springframework.data.domain.Pageable;
import org.springframework.data.jpa.repository.Modifying;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.CrudRepository;
import org.springframework.data.repository.PagingAndSortingRepository;
import org.springframework.data.repository.query.Param;

import javax.transaction.Transactional;
import java.util.List;

public interface EmployeeRepository extends CrudRepository<Employee,Integer>,
PagingAndSortingRepository<Employee,Integer> {

   //Ques1 Display the first name, last name of all employees having salary greater than average salary ordered in
ascending by their age and in descending by their salary.
   @Query("select firstname,lastname from Employee where salary>(select avg(salary) from Employee)order
by salary desc,age asc")
   List<Object[]> findEmployeeHavingSalaryGTaverage();

   //Ques2 Update salary of all employees by a salary passed as a parameter whose existing salary is less than the
average salary.
   @Query("select avg(salary) from Employee")
   Integer findAvgSalary();

   @Transactional
   @Modifying
   @Query("update Employee set salary=:salary where salary < :sal")
   void updateSalary(@Param("salary") Integer salary,@Param("sal") Integer sal);


//    Ques3 Delete all employees with minimum salary.

   //Finding min salary
   @Query("select min(salary) from Employee")
   Integer minSalary();

   @Modifying
   @Transactional
   @Query("delete from Employee where salary=:min_sal")
   void deleteEmployee(@Param("min_sal") Integer min_sal);
```

```
 //******************Native SQL Query:*******************************************

// Ques1 Display the id, first name, age of all employees where last name ends with "singh"
  @Query(value="select empid,empage,empfirstname from employee where emplastname like '%singh'
",nativeQuery = true)
  List<Object[]> findEmployeeNQ();

  //Ques2 Delete all employees with age greater than 45(Should be passed as a parameter)
  @Modifying
  @Transactional
  @Query(value = "delete from employee where empage>:age ",nativeQuery = true)
  void deleteEmpAgeGT45(@Param("age")Integer age);

}
```

File=JpaPart2ApplicationTests.java

```
package com.bootcamp.JPA.Part2;

import com.bootcamp.JPA.Part2.embedding.EmployeeDetails;
import com.bootcamp.JPA.Part2.embedding.EmployeeDetailsRepo;
import com.bootcamp.JPA.Part2.embedding.Salary;
import com.bootcamp.JPA.Part2.entities.Check;
import com.bootcamp.JPA.Part2.entities.CreditCard;
import com.bootcamp.JPA.Part2.entities.Employee;
import com.bootcamp.JPA.Part2.repositry.EmployeeRepository;
import com.bootcamp.JPA.Part2.repositry.PaymentRep;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.data.domain.Sort;
import org.springframework.test.annotation.Rollback;


import javax.transaction.Transactional;
import java.util.List;

@SpringBootTest
class JpaPart2ApplicationTests {

//    *********************** JPQL *************************************
  @Autowired
  EmployeeRepository employeeRepository;

  @Test
```

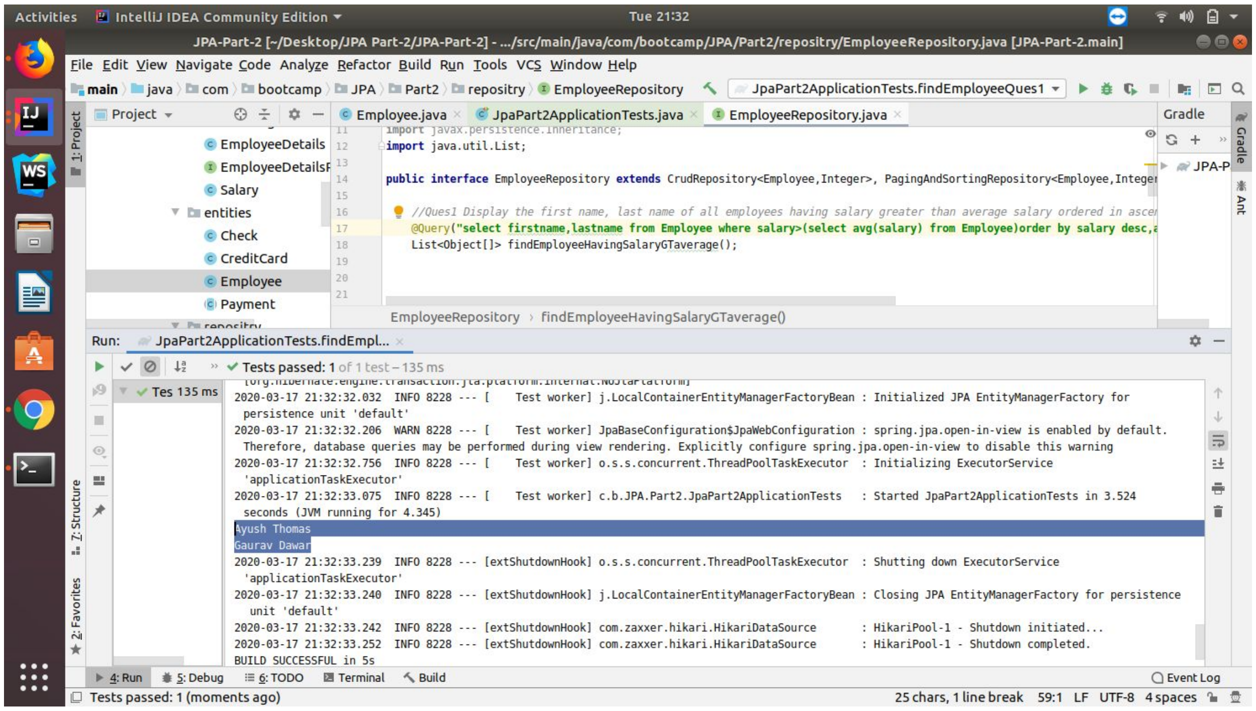
```
public void create(){
    Employee employee=new Employee();
    employee.setFirstName("Gunjan");
    employee.setLastName("Dawar");
    employee.setAge(25);
    employee.setSalary(12000);

    employeeRepository.save(employee);

    Employee employee1=new Employee();
    employee1.setFirstName("Kartik");
    employee1.setLastName("kumar");
    employee1.setAge(48);
    employee1.setSalary(15000);

    employeeRepository.save(employee1);

    Employee employee2=new Employee();
    employee2.setFirstName("Pulkit");
    employee2.setLastName("Kathuria");
    employee2.setAge(35);
    employee2.setSalary(18000);

    employeeRepository.save(employee2);

    Employee employee3=new Employee();
    employee3.setFirstName("Gaurav");
    employee3.setLastName("Dawar");
    employee3.setAge(38);
    employee3.setSalary(22000);

    employeeRepository.save(employee3);

    Employee employee4=new Employee();
    employee4.setFirstName("balpreet");
    employee4.setLastName("singh");
    employee4.setAge(32);
    employee4.setSalary(33000);

    employeeRepository.save(employee4);
}

//Ques1 Display the first name, last name of all employees having salary greater than average salary ordered in
ascending by their age and in descending by their salary.
@Test
public void findEmployeeQues1(){
    List<Object[]> employee=employeeRepository.findEmployeeHavingSalaryGTaverage();
    for (Object[] objects:employee) {
        System.out.println(objects[0]+" "+objects[1]);
    }
```

```
  }

  //Ques2 Update salary of all by a salary passed as a parameter whose existing salary is less than the average
salary.
  @Test
  public  void testAvgSalary(){
      System.out.println(employeeRepository.findAvgSalary());
  }

  @Test
  public void UpdateSalary(){
      Integer sal=employeeRepository.findAvgSalary();
     employeeRepository.updateSalary(23000,sal);
      System.out.println(employeeRepository.findAllEmployees());
  }



  //Ques3 Delete all employees with minimum salary.
  @Test
  public void minSalary(){
      System.out.println(employeeRepository.minSalary());
  }

  @Test
  public void deleteEmployee(){
      Integer minSal=employeeRepository.minSalary();
      employeeRepository.deleteEmployee(minSal);
      System.out.println(employeeRepository.findAllEmployees());
  }

  //*****************Native SQL Query:******************************************

//    Ques1 Display the id, first name, age of all employees where last name ends with "singh"
  @Test
  public void findEmployeesNQ(){
      List<Object[]> details=employeeRepository.findEmployeeNQ();
      for (Object[] objects:details) {
          System.out.println("Id : "+objects[0]+", Age : "+objects[1]+", Name : "+objects[2]);

      }
  }

  //Ques2 Delete all employees with age greater than 45(Should be passed as a parameter)
  @Test
  public void deleteEmpAge(){
      employeeRepository.deleteEmpAgeGT45(45);
      System.out.println(employeeRepository.findAllEmployees());
  }
 }
```

OUTPUTS:

JPQL:

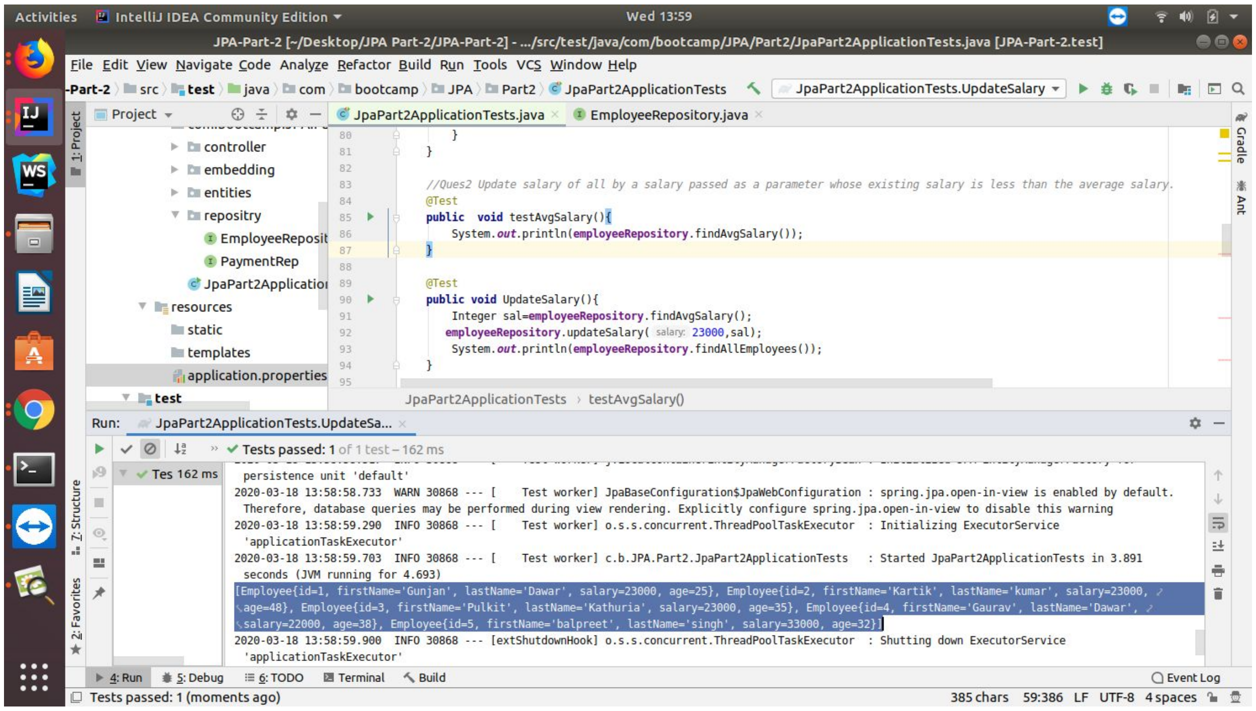
Display the first name, last name of all employees having salary greater than average salary ordered in ascending by their age and in descending by their salary.



```java
import javax.persistence.Inheritance;
import java.util.List;

public interface EmployeeRepository extends CrudRepository<Employee,Integer>, PagingAndSortingRepository<Employee,Integer

    //Ques1 Display the first name, last name of all employees having salary greater than average salary ordered in ascen
    @Query("select firstname,lastname from Employee where salary>(select avg(salary) from Employee)order by salary desc,a
    List<Object[]> findEmployeeHavingSalaryGTaverage();
```

```
Tests passed: 1 of 1 test – 135 ms
[org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
2020-03-17 21:32:32.032  INFO 8228 --- [    Test worker] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for
 persistence unit 'default'
2020-03-17 21:32:32.206  WARN 8228 --- [    Test worker] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default.
 Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2020-03-17 21:32:32.756  INFO 8228 --- [    Test worker] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService
 'applicationTaskExecutor'
2020-03-17 21:32:33.075  INFO 8228 --- [    Test worker] c.b.JPA.Part2.JpaPart2ApplicationTests   : Started JpaPart2ApplicationTests in 3.524
 seconds (JVM running for 4.345)
Ayush Thomas
Gaurav Dawar
2020-03-17 21:32:33.239  INFO 8228 --- [extShutdownHook] o.s.s.concurrent.ThreadPoolTaskExecutor  : Shutting down ExecutorService
 'applicationTaskExecutor'
2020-03-17 21:32:33.240  INFO 8228 --- [extShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for persistence
  unit 'default'
2020-03-17 21:32:33.242  INFO 8228 --- [extShutdownHook] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Shutdown initiated...
2020-03-17 21:32:33.252  INFO 8228 --- [extShutdownHook] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Shutdown completed.
BUILD SUCCESSFUL in 5s
```

Update salary of all employees by a salary passed as a parameter whose existing salary is less than the average salary.

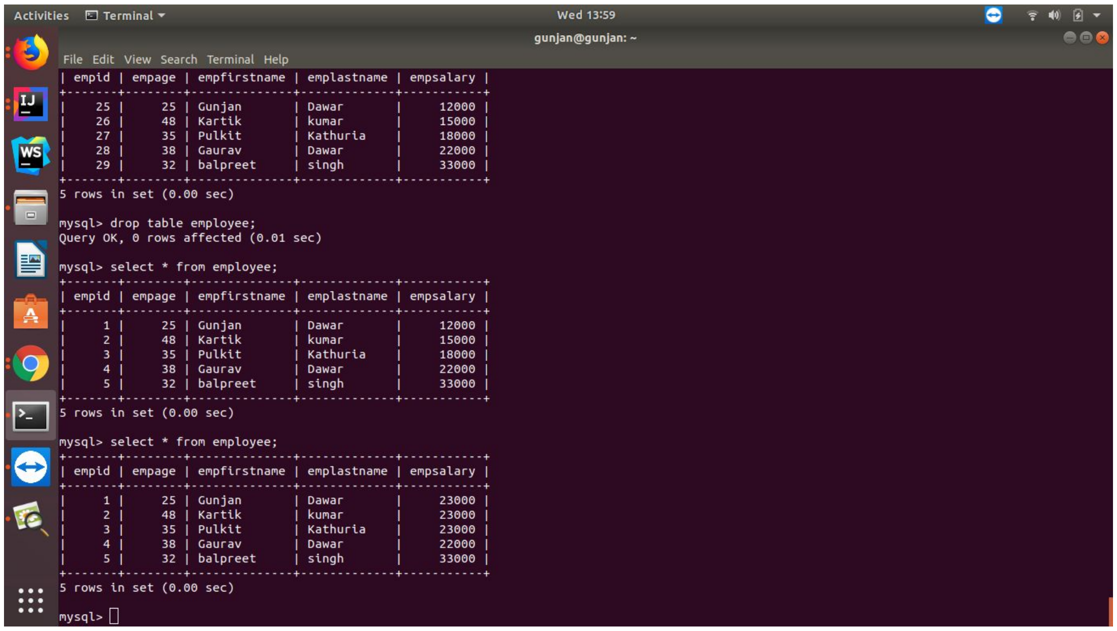


```java
        }
    }

    //Ques2 Update salary of all by a salary passed as a parameter whose existing salary is less than the average salary.
    @Test
    public  void testAvgSalary(){
        System.out.println(employeeRepository.findAvgSalary());
    }

    @Test
    public void UpdateSalary(){
        Integer sal=employeeRepository.findAvgSalary();
       employeeRepository.updateSalary( salary: 23000,sal);
        System.out.println(employeeRepository.findAllEmployees());
    }
```

Tests passed: 1 of 1 test – 162 ms

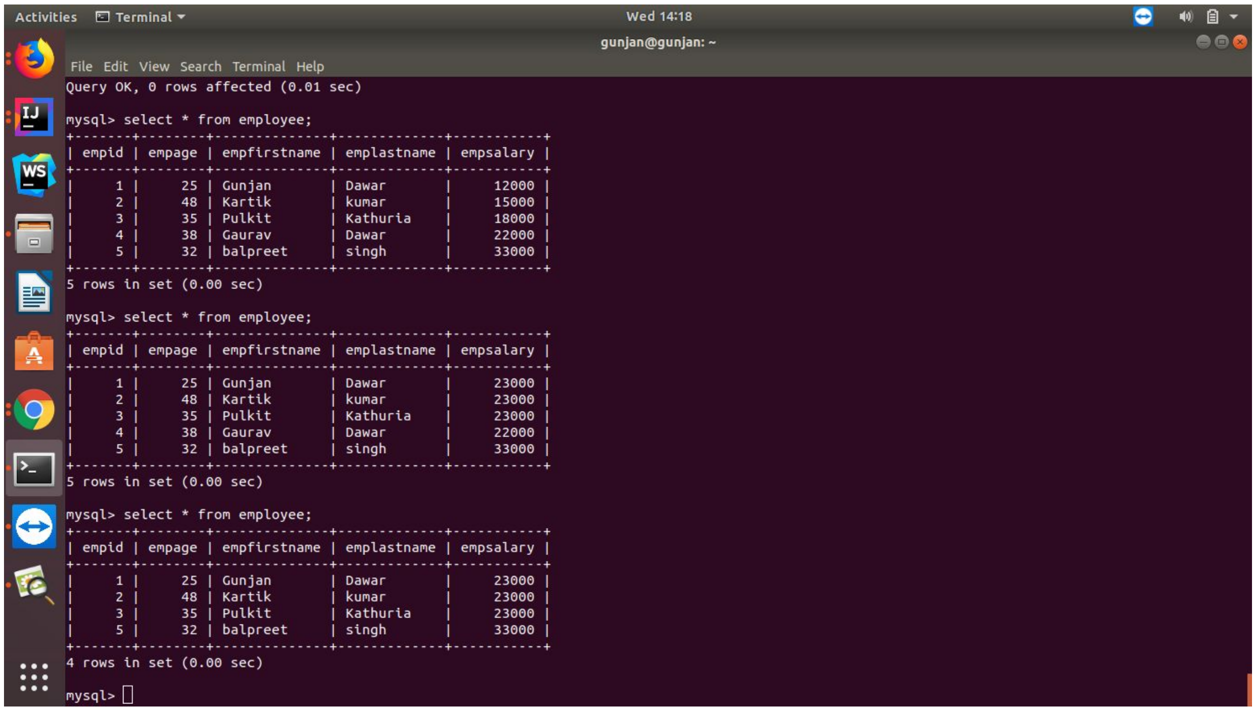
```
 persistence unit 'default'
2020-03-18 13:58:58.733  WARN 30868 --- [    Test worker] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default.
 Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2020-03-18 13:58:59.290  INFO 30868 --- [    Test worker] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService
 'applicationTaskExecutor'
2020-03-18 13:58:59.703  INFO 30868 --- [    Test worker] c.b.JPA.Part2.JpaPart2ApplicationTests   : Started JpaPart2ApplicationTests in 3.891
 seconds (JVM running for 4.693)
[Employee{id=1, firstName='Gunjan', lastName='Dawar', salary=23000, age=25}, Employee{id=2, firstName='Kartik', lastName='kumar', salary=23000,
 age=48}, Employee{id=3, firstName='Pulkit', lastName='Kathuria', salary=23000, age=35}, Employee{id=4, firstName='Gaurav', lastName='Dawar',
 salary=22000, age=38}, Employee{id=5, firstName='balpreet', lastName='singh', salary=33000, age=32}]
2020-03-18 13:58:59.900  INFO 30868 --- [extShutdownHook] o.s.s.concurrent.ThreadPoolTaskExecutor  : Shutting down ExecutorService
 'applicationTaskExecutor'
```

Tests passed: 1 (moments ago)



```
| empid | empage | empfirstname | emplastname | empsalary |
+-------+--------+--------------+-------------+-----------+
|    25 |     25 | Gunjan       | Dawar       |     12000 |
|    26 |     48 | Kartik       | kumar       |     15000 |
|    27 |     35 | Pulkit       | Kathuria    |     18000 |
|    28 |     38 | Gaurav       | Dawar       |     22000 |
|    29 |     32 | balpreet     | singh       |     33000 |
+-------+--------+--------------+-------------+-----------+
5 rows in set (0.00 sec)

mysql> drop table employee;
Query OK, 0 rows affected (0.01 sec)

mysql> select * from employee;
+-------+--------+--------------+-------------+-----------+
| empid | empage | empfirstname | emplastname | empsalary |
+-------+--------+--------------+-------------+-----------+
|     1 |     25 | Gunjan       | Dawar       |     12000 |
|     2 |     48 | Kartik       | kumar       |     15000 |
|     3 |     35 | Pulkit       | Kathuria    |     18000 |
|     4 |     38 | Gaurav       | Dawar       |     22000 |
|     5 |     32 | balpreet     | singh       |     33000 |
+-------+--------+--------------+-------------+-----------+
5 rows in set (0.00 sec)

mysql> select * from employee;
+-------+--------+--------------+-------------+-----------+
| empid | empage | empfirstname | emplastname | empsalary |
+-------+--------+--------------+-------------+-----------+
|     1 |     25 | Gunjan       | Dawar       |     23000 |
|     2 |     48 | Kartik       | kumar       |     23000 |
|     3 |     35 | Pulkit       | Kathuria    |     23000 |
|     4 |     38 | Gaurav       | Dawar       |     22000 |
|     5 |     32 | balpreet     | singh       |     33000 |
+-------+--------+--------------+-------------+-----------+
5 rows in set (0.00 sec)

mysql> 
```

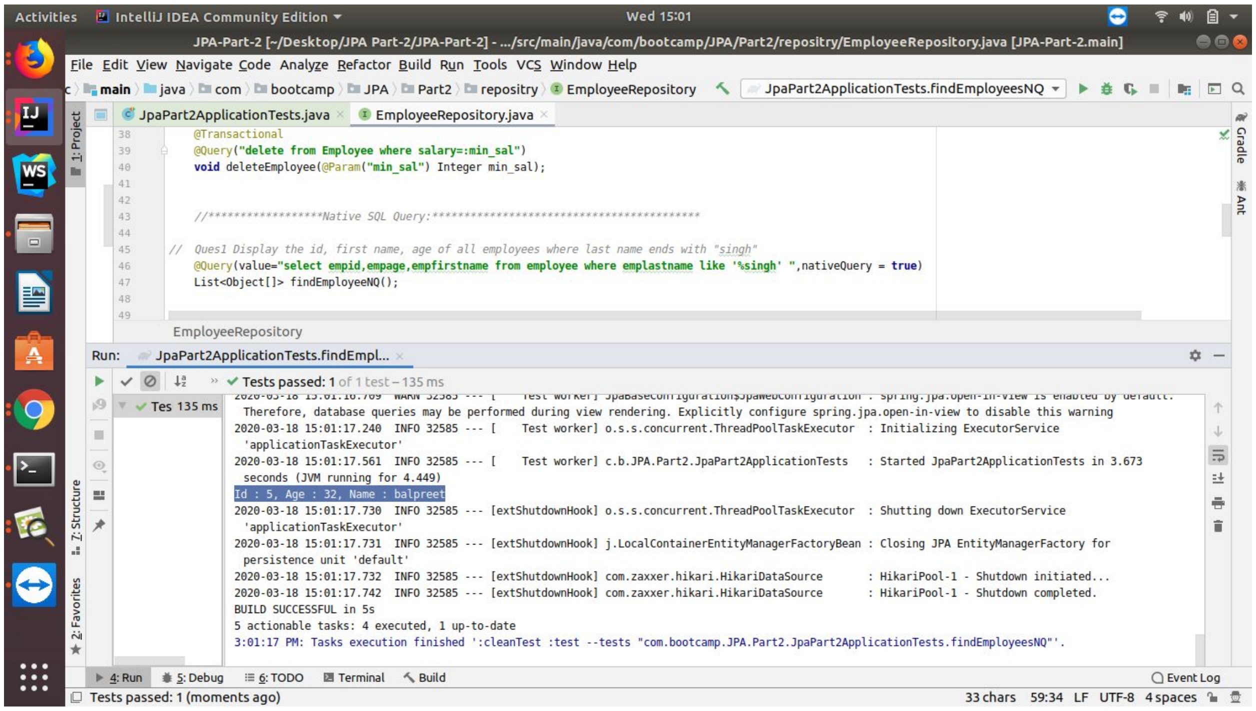
Delete all employees with minimum salary.



```
Query OK, 0 rows affected (0.01 sec)

mysql> select * from employee;
+-------+--------+--------------+-------------+-----------+
| empid | empage | empfirstname | emplastname | empsalary |
+-------+--------+--------------+-------------+-----------+
|     1 |     25 | Gunjan       | Dawar       |     12000 |
|     2 |     48 | Kartik       | kumar       |     15000 |
|     3 |     35 | Pulkit       | Kathuria    |     18000 |
|     4 |     38 | Gaurav       | Dawar       |     22000 |
|     5 |     32 | balpreet     | singh       |     33000 |
+-------+--------+--------------+-------------+-----------+
5 rows in set (0.00 sec)

mysql> select * from employee;
+-------+--------+--------------+-------------+-----------+
| empid | empage | empfirstname | emplastname | empsalary |
+-------+--------+--------------+-------------+-----------+
|     1 |     25 | Gunjan       | Dawar       |     23000 |
|     2 |     48 | Kartik       | kumar       |     23000 |
|     3 |     35 | Pulkit       | Kathuria    |     23000 |
|     4 |     38 | Gaurav       | Dawar       |     22000 |
|     5 |     32 | balpreet     | singh       |     33000 |
+-------+--------+--------------+-------------+-----------+
5 rows in set (0.00 sec)

mysql> select * from employee;
+-------+--------+--------------+-------------+-----------+
| empid | empage | empfirstname | emplastname | empsalary |
+-------+--------+--------------+-------------+-----------+
|     1 |     25 | Gunjan       | Dawar       |     23000 |
|     2 |     48 | Kartik       | kumar       |     23000 |
|     3 |     35 | Pulkit       | Kathuria    |     23000 |
|     5 |     32 | balpreet     | singh       |     33000 |
+-------+--------+--------------+-------------+-----------+
4 rows in set (0.00 sec)

mysql> 
```

Native SQL Query:

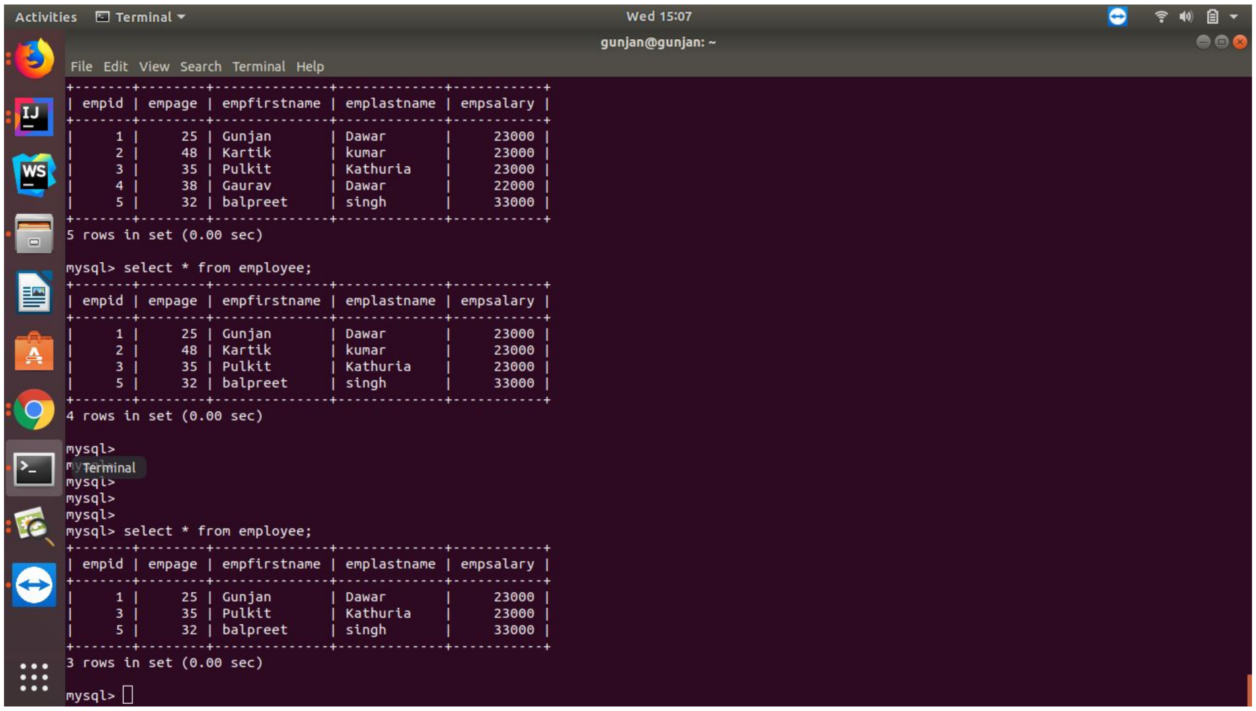
Display the id, first name, age of all employees where last name ends with "singh"


Activities
IntelliJ IDEA Community Edition
Wed 15:01
JPA-Part-2 [~/Desktop/JPA Part-2/JPA-Part-2] - .../src/main/java/com/bootcamp/JPA/Part2/repositry/EmployeeRepository.java [JPA-Part-2.main]
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
main java com bootcamp JPA Part2 repositry EmployeeRepository
JpaPart2ApplicationTests.findEmployeesNQ
JpaPart2ApplicationTests.java
EmployeeRepository.java
1: Project
7: Structure
2: Favorites
Gradle
Ant
EmployeeRepository
Run: JpaPart2ApplicationTests.findEmpl...
Tests passed: 1 of 1 test – 135 ms
Tes 135 ms
4: Run
5: Debug
6: TODO
Terminal
Build
Event Log
Tests passed: 1 (moments ago)
33 chars 59:34 LF UTF-8 4 spaces

```java
    @Transactional
    @Query("delete from Employee where salary=:min_sal")
    void deleteEmployee(@Param("min_sal") Integer min_sal);


    //*****************Native SQL Query:*****************************************

//  Ques1 Display the id, first name, age of all employees where last name ends with "singh"
    @Query(value="select empid,empage,empfirstname from employee where emplastname like '%singh' ",nativeQuery = true)
    List<Object[]> findEmployeeNQ();
```

```
2020-03-18 15:01:16.709  WARN 32585 --- [    Test worker] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default.
 Therefore, database queries may be performed during view rendering. Explicitly configure spring.jpa.open-in-view to disable this warning
2020-03-18 15:01:17.240  INFO 32585 --- [    Test worker] o.s.s.concurrent.ThreadPoolTaskExecutor  : Initializing ExecutorService
 'applicationTaskExecutor'
2020-03-18 15:01:17.561  INFO 32585 --- [    Test worker] c.b.JPA.Part2.JpaPart2ApplicationTests   : Started JpaPart2ApplicationTests in 3.673
 seconds (JVM running for 4.449)
Id : 5, Age : 32, Name : balpreet
2020-03-18 15:01:17.730  INFO 32585 --- [extShutdownHook] o.s.s.concurrent.ThreadPoolTaskExecutor  : Shutting down ExecutorService
 'applicationTaskExecutor'
2020-03-18 15:01:17.731  INFO 32585 --- [extShutdownHook] j.LocalContainerEntityManagerFactoryBean : Closing JPA EntityManagerFactory for
 persistence unit 'default'
2020-03-18 15:01:17.732  INFO 32585 --- [extShutdownHook] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Shutdown initiated...
2020-03-18 15:01:17.742  INFO 32585 --- [extShutdownHook] com.zaxxer.hikari.HikariDataSource       : HikariPool-1 - Shutdown completed.
BUILD SUCCESSFUL in 5s
5 actionable tasks: 4 executed, 1 up-to-date
3:01:17 PM: Tasks execution finished ':cleanTest :test --tests "com.bootcamp.JPA.Part2.JpaPart2ApplicationTests.findEmployeesNQ"'.
```

Delete all employees with age greater than 45(Should be passed as a parameter)



```
+-------+--------+--------------+-------------+-----------+
| empid | empage | empfirstname | emplastname | empsalary |
+-------+--------+--------------+-------------+-----------+
|     1 |     25 | Gunjan       | Dawar       |     23000 |
|     2 |     48 | Kartik       | kumar       |     23000 |
|     3 |     35 | Pulkit       | Kathuria    |     23000 |
|     4 |     38 | Gaurav       | Dawar       |     22000 |
|     5 |     32 | balpreet     | singh       |     33000 |
+-------+--------+--------------+-------------+-----------+
5 rows in set (0.00 sec)

mysql> select * from employee;
+-------+--------+--------------+-------------+-----------+
| empid | empage | empfirstname | emplastname | empsalary |
+-------+--------+--------------+-------------+-----------+
|     1 |     25 | Gunjan       | Dawar       |     23000 |
|     2 |     48 | Kartik       | kumar       |     23000 |
|     3 |     35 | Pulkit       | Kathuria    |     23000 |
|     5 |     32 | balpreet     | singh       |     33000 |
+-------+--------+--------------+-------------+-----------+
4 rows in set (0.00 sec)

mysql>
mysql>
mysql>
mysql>
mysql>
mysql> select * from employee;
+-------+--------+--------------+-------------+-----------+
| empid | empage | empfirstname | emplastname | empsalary |
+-------+--------+--------------+-------------+-----------+
|     1 |     25 | Gunjan       | Dawar       |     23000 |
|     3 |     35 | Pulkit       | Kathuria    |     23000 |
|     5 |     32 | balpreet     | singh       |     33000 |
+-------+--------+--------------+-------------+-----------+
3 rows in set (0.00 sec)

mysql>
```

Inheritance Mapping:

1. Implement and demonstrate Single Table strategy.

File=Payment.java

```java
package com.bootcamp.JPA.Part2.entities;

import javax.persistence.*;

@Entity
@Inheritance(strategy = InheritanceType.SINGLE_TABLE)
@DiscriminatorColumn(name = "pmode", discriminatorType = DiscriminatorType.STRING)
public abstract class Payment {
   @Id
   @GeneratedValue(strategy = GenerationType.AUTO)
   private int id;
   private int amount;


   public int getId() {
       return id;
   }

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

   public int getAmount() {
       return amount;
   }

   public void setAmount(int amount) {
       this.amount = amount;
   }

   @Override
   public String toString() {
       return "Payment{" +
               "id=" + id +
               ", amount=" + amount +
               '}';
   }
}
```

File=CreditCard.java

```java
package com.bootcamp.JPA.Part2.entities;
```

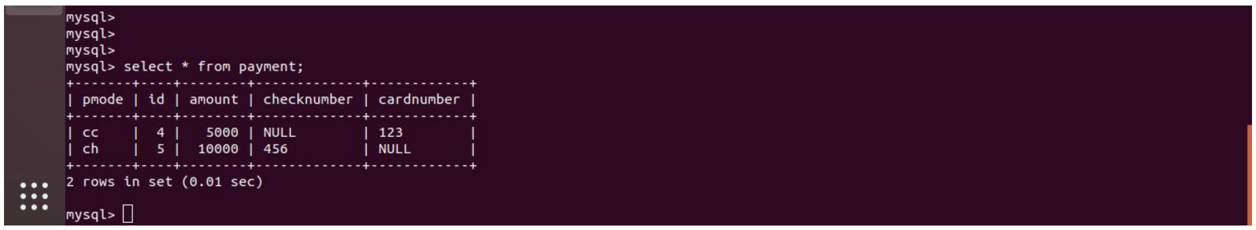
```java
import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;

@Entity
@DiscriminatorValue("cc")
public class CreditCard extends Payment {
  private String cardnumber;

  public String getCardnumber() {
     return cardnumber;
  }

  public void setCardnumber(String cardnumber) {
     this.cardnumber = cardnumber;
  }
}
```

File=Check.java

```java
package com.bootcamp.JPA.Part2.entities;

import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;

@Entity
@DiscriminatorValue("ch")
public class Check extends Payment {
  private String checknumber;

  public String getChecknumber() {
     return checknumber;
  }

  public void setChecknumber(String checknumber) {
     this.checknumber = checknumber;
  }
}
```

File=JpaPart2ApplicationTests.java

```java
package com.bootcamp.JPA.Part2;

import com.bootcamp.JPA.Part2.entities.Check;
import com.bootcamp.JPA.Part2.entities.CreditCard;
import com.bootcamp.JPA.Part2.entities.Employee;
import com.bootcamp.JPA.Part2.repositry.EmployeeRepository;
import com.bootcamp.JPA.Part2.repositry.PaymentRep;
import net.minidev.json.JSONUtil;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.data.domain.Sort;
import org.springframework.test.annotation.Rollback;

import javax.transaction.Transactional;
import java.util.List;

@SpringBootTest
class JpaPart2ApplicationTests {


@Autowired
  PaymentRep paymentRep;

  @Test
  public void createCCPayment() {
     CreditCard creditCard = new CreditCard();
     creditCard.setId(100);
     creditCard.setAmount(5000);
     creditCard.setCardnumber("123");
     paymentRep.save(creditCard);

  }

  @Test
  public void createCHPayment(){
     Check check=new Check();
     check.setId(102);
     check.setAmount(10000);
     check.setChecknumber("456");
     paymentRep.save(check);
  }

}
```



```
mysql>
mysql>
mysql>
mysql> select * from payment;
+-------+----+--------+-------------+------------+
| pmode | id | amount | checknumber | cardnumber |
+-------+----+--------+-------------+------------+
| cc    |  4 |   5000 | NULL        | 123        |
| ch    |  5 |  10000 | 456         | NULL       |
+-------+----+--------+-------------+------------+
2 rows in set (0.01 sec)

mysql> 
```

2. Implement and demonstrate Joined strategy.

File=Payment.java

```java
package com.bootcamp.JPA.Part2.entities;

import javax.persistence.*;

@Entity
@Inheritance(strategy = InheritanceType.JOINED)
public abstract class Payment {
   @Id
   @GeneratedValue(strategy = GenerationType.AUTO)
   private int id;
   private int amount;


   public int getId() {
       return id;
   }

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

   public int getAmount() {
       return amount;
   }

   public void setAmount(int amount) {
       this.amount = amount;
   }

   @Override
   public String toString() {
       return "Payment{" +
               "id=" + id +
               ", amount=" + amount +
               '}';
   }
}
```

File=CreditCard.java

```java
package com.bootcamp.JPA.Part2.entities;

import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;

@Entity
@Table(name = "cheque")
@PrimaryKeyJoinColumn(name = "id")
public class CreditCard extends Payment {
  private String cardnumber;

  public String getCardnumber() {
     return cardnumber;
  }

  public void setCardnumber(String cardnumber) {
     this.cardnumber = cardnumber;
  }
}
```

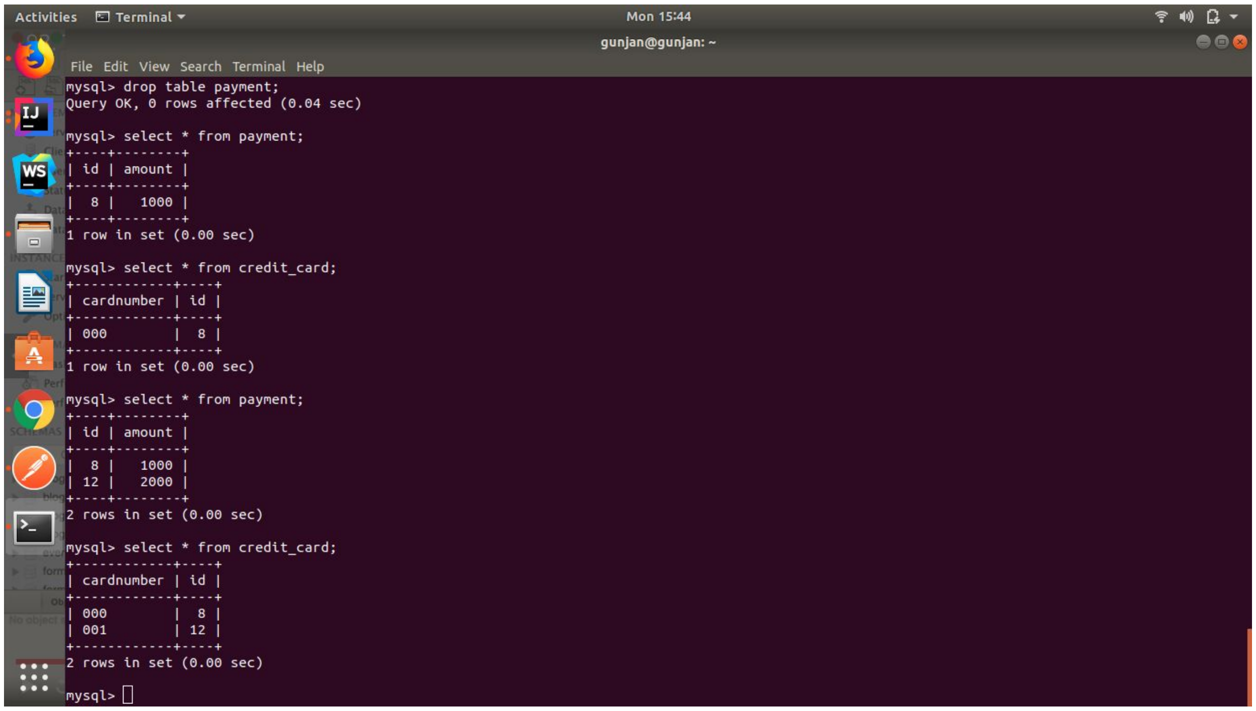
File=Check.java

```
package com.bootcamp.JPA.Part2.entities;

import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;

@Entity
@Table(name = "cheque")
@PrimaryKeyJoinColumn(name = "id")
public class Check extends Payment {
  private String checknumber;

  public String getChecknumber() {
     return checknumber;
  }

  public void setChecknumber(String checknumber) {
     this.checknumber = checknumber;
  }
}
```

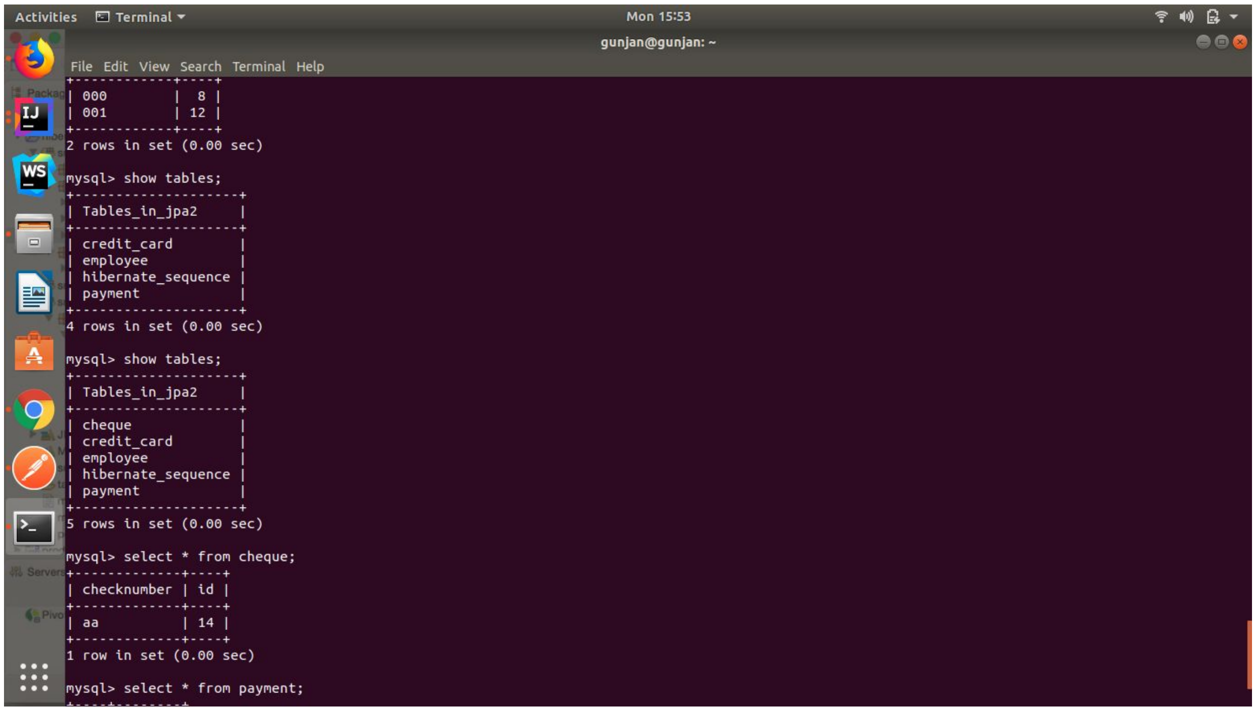
File=JpaPart2ApplicationTests.java

```
package com.bootcamp.JPA.Part2;

import com.bootcamp.JPA.Part2.entities.Check;
import com.bootcamp.JPA.Part2.entities.CreditCard;
import com.bootcamp.JPA.Part2.entities.Employee;
import com.bootcamp.JPA.Part2.repositry.EmployeeRepository;
import com.bootcamp.JPA.Part2.repositry.PaymentRep;
import net.minidev.json.JSONUtil;
import org.junit.jupiter.api.Test;
```

```java
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.data.domain.Sort;
import org.springframework.test.annotation.Rollback;

import javax.transaction.Transactional;
import java.util.List;

@SpringBootTest
class JpaPart2ApplicationTests {


@Autowired
  PaymentRep paymentRep;

@Test
public void createCCPayment() {
  CreditCard creditCard = new CreditCard();
  creditCard.setAmount(2000);
  creditCard.setCardnumber("001");
  paymentRep.save(creditCard);

}

@Test
public void createCHPayment(){
  Check check=new Check();
  check.setAmount(2000);
  check.setChecknumber("aa");
  paymentRep.save(check);
}


}
```



```
mysql> drop table payment;
Query OK, 0 rows affected (0.04 sec)

mysql> select * from payment;
+----+--------+
| id | amount |
+----+--------+
|  8 |   1000 |
+----+--------+
1 row in set (0.00 sec)

mysql> select * from credit_card;
+------------+----+
| cardnumber | id |
+------------+----+
| 000        |  8 |
+------------+----+
1 row in set (0.00 sec)

mysql> select * from payment;
+----+--------+
| id | amount |
+----+--------+
|  8 |   1000 |
| 12 |   2000 |
+----+--------+
2 rows in set (0.00 sec)

mysql> select * from credit_card;
+------------+----+
| cardnumber | id |
+------------+----+
| 000        |  8 |
| 001        | 12 |
+------------+----+
2 rows in set (0.00 sec)

mysql> 
```



```
+------------+----+
| 000        |  8 |
| 001        | 12 |
+------------+----+
2 rows in set (0.00 sec)

mysql> show tables;
+--------------------+
| Tables_in_jpa2     |
+--------------------+
| credit_card        |
| employee           |
| hibernate_sequence |
| payment            |
+--------------------+
4 rows in set (0.00 sec)

mysql> show tables;
+--------------------+
| Tables_in_jpa2     |
+--------------------+
| cheque             |
| credit_card        |
| employee           |
| hibernate_sequence |
| payment            |
+--------------------+
5 rows in set (0.00 sec)

mysql> select * from cheque;
+-------------+----+
| checknumber | id |
+-------------+----+
| aa          | 14 |
+-------------+----+
1 row in set (0.00 sec)

mysql> select * from payment;
```

3. Implement and demonstrate Table Per Class strategy.

File=Payment.java

```
package com.bootcamp.JPA.Part2.entities;

import javax.persistence.*;

@Entity
@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)
public abstract class Payment {
   @Id
   @GeneratedValue(strategy = GenerationType.AUTO)
   private int id;
   private int amount;


   public int getId() {
       return id;
   }

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

   public int getAmount() {
       return amount;
   }

   public void setAmount(int amount) {
       this.amount = amount;
   }

   @Override
   public String toString() {
       return "Payment{" +
               "id=" + id +
               ", amount=" + amount +
               '}';
   }
}
```

File=CreditCard.java

```
package com.bootcamp.JPA.Part2.entities;

import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;
```

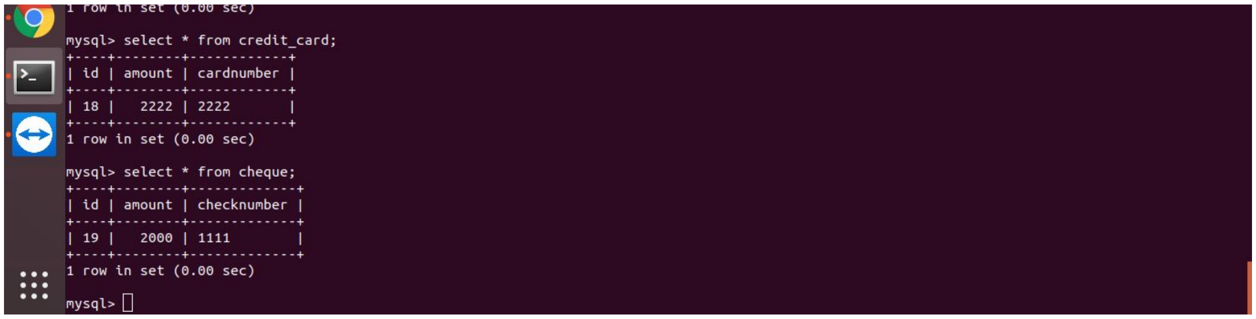
```
@Entity
@Table(name = "cheque")
@PrimaryKeyJoinColumn(name = "id")
public class CreditCard extends Payment {
   private String cardnumber;

   public String getCardnumber() {
       return cardnumber;
   }

   public void setCardnumber(String cardnumber) {
       this.cardnumber = cardnumber;
   }
}
```

File=Check.java

```
package com.bootcamp.JPA.Part2.entities;

import javax.persistence.DiscriminatorValue;
import javax.persistence.Entity;

@Entity
@Table(name = "cheque")
@PrimaryKeyJoinColumn(name = "id")
public class Check extends Payment {
   private String checknumber;

   public String getChecknumber() {
       return checknumber;
   }

   public void setChecknumber(String checknumber) {
       this.checknumber = checknumber;
   }
}
```

File=JpaPart2ApplicationTests.java

```
package com.bootcamp.JPA.Part2;

import com.bootcamp.JPA.Part2.entities.Check;
import com.bootcamp.JPA.Part2.entities.CreditCard;
import com.bootcamp.JPA.Part2.entities.Employee;
import com.bootcamp.JPA.Part2.repositry.EmployeeRepository;
import com.bootcamp.JPA.Part2.repositry.PaymentRep;
import net.minidev.json.JSONUtil;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.data.domain.Sort;
import org.springframework.test.annotation.Rollback;

import javax.transaction.Transactional;
import java.util.List;

@SpringBootTest
class JpaPart2ApplicationTests {


@Autowired
   PaymentRep paymentRep;

@Test
public void createCCPayment() {
   CreditCard creditCard = new CreditCard();
   creditCard.setAmount(2222);
   creditCard.setCardnumber("2222");
   paymentRep.save(creditCard);

}

@Test
public void createCHPayment(){
   Check check=new Check();
   check.setAmount(2000);
   check.setChecknumber("1111");
   paymentRep.save(check);
}

}
```



```
1 row in set (0.00 sec)

mysql> select * from credit_card;
+----+--------+------------+
| id | amount | cardnumber |
+----+--------+------------+
| 18 |   2222 | 2222       |
+----+--------+------------+
1 row in set (0.00 sec)

mysql> select * from cheque;
+----+--------+-------------+
| id | amount | checknumber |
+----+--------+-------------+
| 19 |   2000 | 1111        |
+----+--------+-------------+
1 row in set (0.00 sec)

mysql> 
```

Component Mapping:

1. Implement and demonstrate Embedded mapping using employee table having following fields: id, firstName, lastName, age, basicSalary, bonusSalary, taxAmount, specialAllowanceSalary.

File=EmployeeDetails.java

```
package com.bootcamp.JPA.Part2.embedding;

import javax.persistence.*;

@Entity
public class EmployeeDetails {

  @Id
  @GeneratedValue(strategy = GenerationType.AUTO)
   int id;
   String firstname;
   String lastname;
   int age;
   @Embedded
   private Salary salary;

  public Salary getSalary() {
      return salary;
  }

  public void setSalary(Salary salary) {
      this.salary = salary;
  }

  public int getId() {
      return id;
  }

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

  public String getFirstname() {
      return firstname;
  }

  public void setFirstname(String firstname) {
      this.firstname = firstname;
  }
```

```
    public String getLastname() {
        return lastname;
    }

    public void setLastname(String lastname) {
        this.lastname = lastname;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }
}
```

File=Salary.java

```
package com.bootcamp.JPA.Part2.embedding;

import javax.persistence.Embeddable;

@Embeddable
public class Salary {

    private double basicSalary;
    private double bonusSalary;
    private double taxAmount;
    private double  specialAllowanceSalary;

    public double getBasicSalary() {
        return basicSalary;
    }

    public void setBasicSalary(double basicSalary) {
        this.basicSalary = basicSalary;
    }

    public double getBonusSalary() {
        return bonusSalary;
    }

    public void setBonusSalary(double bonusSalary) {
        this.bonusSalary = bonusSalary;
    }

    public double getTaxAmount() {
        return taxAmount;
```

```
  }

  public void setTaxAmount(double taxAmount) {
      this.taxAmount = taxAmount;
  }

  public double getSpecialAllowanceSalary() {
      return specialAllowanceSalary;
  }

  public void setSpecialAllowanceSalary(double specialAllowanceSalary) {
      this.specialAllowanceSalary = specialAllowanceSalary;
  }
}
```

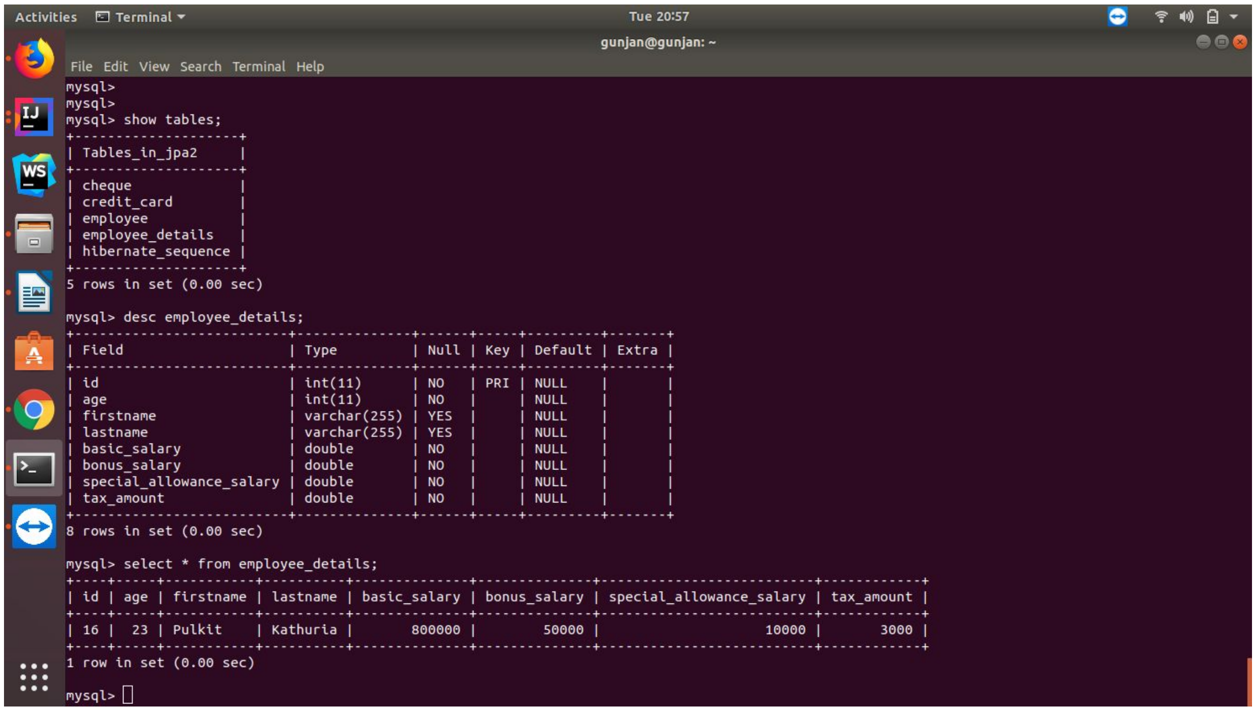
File=EmployeeDetailsRepo.java

```
package com.bootcamp.JPA.Part2.embedding;

import org.springframework.data.repository.CrudRepository;

public interface EmployeeDetailsRepo extends CrudRepository<EmployeeDetails,Integer> {

}
```

File=JpaPart2ApplicationTests.java

```
package com.bootcamp.JPA.Part2;

import com.bootcamp.JPA.Part2.entities.Check;
import com.bootcamp.JPA.Part2.entities.CreditCard;
import com.bootcamp.JPA.Part2.entities.Employee;
import com.bootcamp.JPA.Part2.repositry.EmployeeRepository;
import com.bootcamp.JPA.Part2.repositry.PaymentRep;
import net.minidev.json.JSONUtil;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.data.domain.Sort;
import org.springframework.test.annotation.Rollback;

import javax.transaction.Transactional;
import java.util.List;

@SpringBootTest
class JpaPart2ApplicationTests {
```

```
//****************************Component Mapping **************************************

@Autowired
EmployeeDetailsRepo employeeDetailsRepo;

@Test
public void testCreate(){
  EmployeeDetails employeeDetails=new EmployeeDetails();

  employeeDetails.setFirstname("Pulkit");
  employeeDetails.setLastname("Kathuria");
  employeeDetails.setAge(23);
  Salary salary=new Salary();
  salary.setBasicSalary(800000);
  salary.setBonusSalary(50000);
  salary.setSpecialAllowanceSalary(10000);
  salary.setTaxAmount(3000);

  employeeDetails.setSalary(salary);

  employeeDetailsRepo.save(employeeDetails);

}
```



```
mysql>
mysql>
mysql> show tables;
+--------------------+
| Tables_in_jpa2     |
+--------------------+
| cheque             |
| credit_card        |
| employee           |
| employee_details   |
| hibernate_sequence |
+--------------------+
5 rows in set (0.00 sec)

mysql> desc employee_details;
+--------------------------+--------------+------+-----+---------+-------+
| Field                    | Type         | Null | Key | Default | Extra |
+--------------------------+--------------+------+-----+---------+-------+
| id                       | int(11)      | NO   | PRI | NULL    |       |
| age                      | int(11)      | NO   |     | NULL    |       |
| firstname                | varchar(255) | YES  |     | NULL    |       |
| lastname                 | varchar(255) | YES  |     | NULL    |       |
| basic_salary             | double       | NO   |     | NULL    |       |
| bonus_salary             | double       | NO   |     | NULL    |       |
| special_allowance_salary | double       | NO   |     | NULL    |       |
| tax_amount               | double       | NO   |     | NULL    |       |
+--------------------------+--------------+------+-----+---------+-------+
8 rows in set (0.00 sec)

mysql> select * from employee_details;
+----+-----+-----------+----------+--------------+--------------+--------------------------+------------+
| id | age | firstname | lastname | basic_salary | bonus_salary | special_allowance_salary | tax_amount |
+----+-----+-----------+----------+--------------+--------------+--------------------------+------------+
| 16 |  23 | Pulkit    | Kathuria |       800000 |        50000 |                    10000 |       3000 |
+----+-----+-----------+----------+--------------+--------------+--------------------------+------------+
1 row in set (0.00 sec)

mysql> 
```

```
}
```