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)

Ans === JPQL and Native SQL Query combined solution ---

<https://github.com/pranav0297/jpa2/blob/main/src/main/java/com/example/jpa2/demo_jpa2/Employee.java>

<https://github.com/pranav0297/jpa2/blob/main/src/main/java/com/example/jpa2/demo_jpa2/repo/EmployeeRepository.java>

<https://github.com/pranav0297/jpa2/blob/main/src/test/java/com/example/jpa2/demo_jpa2/DemoJpa2ApplicationTests.java>

And full project link ===<https://github.com/pranav0297/jpa2/tree/main>

Inheritance Mapping:

1. Implement and demonstrate Single Table strategy.

Ans == <https://github.com/pranav0297/jpa2/blob/branch1/src/main/java/com/example/jpa/inheritence/demo_jpa2/inheritence/payment.java>

<https://github.com/pranav0297/jpa2/blob/branch1/src/main/java/com/example/jpa/inheritence/demo_jpa2/inheritence/CreditCard.java>

<https://github.com/pranav0297/jpa2/blob/branch1/src/main/java/com/example/jpa/inheritence/demo_jpa2/inheritence/check.java>

<https://github.com/pranav0297/jpa2/blob/branch1/src/test/java/com/example/jpa/inheritence/demo_jpa2/inheritence/ApplicationTests.java>

1. Implement and demonstrate Joined strategy.

Ans == <https://github.com/pranav0297/jpa2/blob/branch1/src/main/java/com/example/jpa/inheritence/demo_jpa2/inheritence/payment2.java>

<https://github.com/pranav0297/jpa2/blob/branch1/src/main/java/com/example/jpa/inheritence/demo_jpa2/inheritence/card.java>

<https://github.com/pranav0297/jpa2/blob/branch1/src/main/java/com/example/jpa/inheritence/demo_jpa2/inheritence/bankcheck.java>

<https://github.com/pranav0297/jpa2/blob/branch1/src/test/java/com/example/jpa/inheritence/demo_jpa2/inheritence/ApplicationTests.java>

1. Implement and demonstrate Table Per Class strategy.

Ans == <https://github.com/pranav0297/jpa2/blob/branch1/src/main/java/com/example/jpa/inheritence/demo_jpa2/inheritence/payment3.java>

<https://github.com/pranav0297/jpa2/blob/branch1/src/main/java/com/example/jpa/inheritence/demo_jpa2/inheritence/card2.java>

<https://github.com/pranav0297/jpa2/blob/branch1/src/main/java/com/example/jpa/inheritence/demo_jpa2/inheritence/bankcheck2.java>

<https://github.com/pranav0297/jpa2/blob/branch1/src/test/java/com/example/jpa/inheritence/demo_jpa2/inheritence/ApplicationTests.java>

Ans full project link == <https://github.com/pranav0297/jpa2/tree/branch1>

Component Mapping:

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

Ans == <https://github.com/pranav0297/jpa2/blob/branch2/src/main/java/com/example/jpa/componentMapping/demo_jpa2/componentMapping/employee4.java>

<https://github.com/pranav0297/jpa2/blob/branch2/src/main/java/com/example/jpa/componentMapping/demo_jpa2/componentMapping/address.java>

<https://github.com/pranav0297/jpa2/blob/branch2/src/test/java/com/example/jpa/componentMapping/demo_jpa2/componentMapping/ApplicationTests.java>

And full project link == <https://github.com/pranav0297/jpa2/tree/branch2>