

# Relational Databases with MySQL Week 4 Coding Assignment

Points possible: 70

| Category             | Criteria  | % of Grade |
|----------------------|---|------------|
| <b>Functionality</b> | Does the code work?   | 25         |
| <b>Organization</b>  | Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear. | 25         |
| <b>Creativity</b>    | Student solved the problems presented in the assignment using creativity and out of the box thinking.                                       | 25         |
| <b>Completeness</b>  | All requirements of the assignment are complete.  | 25         |

**Instructions:** Complete the coding steps. Take screenshots of the steps and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document to the repository. Additionally, push the Java project to the same repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

## Coding Steps:

1. Create a class of whatever type you want (Animal, Person, Camera, Cheese, etc.).
  - a. Do not implement the Comparable interface.
  - b. Add a name instance variable so that you can tell the objects apart.
  - c. Add getters, setters and/or a constructor as appropriate.
  - d. Add a `toString` method that returns the name and object type (like "Pentax Camera").
  - e. Create a static method named `compare` in the class that returns an int and takes two of the objects as parameters. Return -1 if parameter 1 is "less than" parameter 2. Return 1 if parameter 1 is "greater than" parameter 2. Return 0 if the two parameters are "equal".

- f. Create a static list of these objects, adding at least 4 objects to the list.
  - g. In another class, write a method to sort the objects using a Lambda expression using the compare method you created earlier.
  - h. Write a method to sort the objects using a Method Reference to the compare method you created earlier.
  - i. Create a main method to call the sort methods.
  - j. Print the list after sorting (System.out.println).
2. Create a new class with a main method. Using the list of objects you created in the prior step.
  - a. Create a Stream from the list of objects.
  - b. Turn the Stream of object to a Stream of String (use the map method for this).
  - c. Sort the Stream in the natural order. (Note: The String class implements the Comparable interface, so you won't have to supply a Comparator to do the sorting.)
  - d. Collect the Stream and return a comma-separated list of names as a single String.  
Hint: use Collectors.joining(", ") for this.
  - e. Print the resulting String.
3. Create a new class with a main method. Create a method (method a) that accepts an Optional of some type of object (Animal, Person, Camera, etc.).
  - a. The method should return the object unwrapped from the Optional if the object is present. For example, if you have an object of type Cheese, your method signature should look something like this:

```
public Cheese cheesyMethod(Optional<Cheese> optionalCheese) { ... }
```
  - b. The method should throw a NoSuchElementException with a custom message if the object is not present.
  - c. Create another method (method b) that calls method a with an object wrapped by an Optional. Show that the object is returned unwrapped from the Optional (i.e., print the object).
  - d. Method b should also call method a with an empty Optional. Show that a NoSuchElementException is thrown by method a by printing the exception message. Hint: catch the NoSuchElementException as parameter named "e" and do System.out.println(e.getMessage()).

- e. Note: your method should handle the Optional as shown in the video on Optionals using the orElseThrow method. For the missing object, you must use a Lambda expression in orElseThrow to return a NoSuchElementException with a custom message.

## Screenshots:

### URL to GitHub Repository:

The screenshot shows the Eclipse IDE interface with the title bar "eclipse-workspace - MySQLWeek5/src/questionOne/App.java - Eclipse IDE". The left side features the "Explorer" view showing a project structure with files like App.java, Dish.java, and various optional classes. The main editor window displays the following Java code for App.java:

```

1 package questionOne;
2
3 import java.util.*;
4
5 public class App {
6
7     public static void main(String[] args) {
8
9         List<Dish> dishes = new ArrayList<>(List.of(new Dish("Waffle"),
10             new Dish("Meat of your choice"),
11             new Dish("potato"), new Dish("really black coffee"),
12             new Dish("egg")));
13
14         Comparator<Dish> comp;
15
16         String type = "METHOD_REFERENCE";
17
18         // ...
19         comp = (d1, d2) -> { return Dish.compareDishes(d1, d2);};
20         System.out.println(comp);
21         dishes.sort(comp);
22         print(dishes);
23
24     }
25
26     private static void print(List<Dish>dishes) {
27
28         dishes.forEach(dish -> System.out.println(dish));
29     }
30
31
32
33 }
34
35
36

```

The bottom right corner of the code editor has a note: "3 · 20 · 41". Below the editor is the "Console" tab, which shows the output of the application's execution:

```

<terminated> App [0] Java Application [Library/Java/VirtualMachines/jdk-17.jdk/Contents/Home/bin/java] (Dec 14, 2021, 2:51:07 PM – 2:51:08 PM)
questionOne.App$$Lambda$1/0x0000000000c00c20@6b95977
Dish [name=Meat of your choice]
Dish [name=Waffle]
Dish [name=egg]
Dish [name=potato]
Dish [name=really black coffee]

```

Q1 app

eclipse-workspace - MySQLweek5/src/questionOne/Dish.java - Eclipse IDE

```

1 package questionOne;
2
3 public class Dish {
4     private String name;
5     public String getName() {
6         return name;
7     }
8     public Dish(String name) {
9         this.name = name;
10    }
11    public static int compareDishes(Dish d1, Dish d2) {
12        return d1.getName().compareTo(d2.getName());
13    }
14    @Override
15    public String toString() {
16        return "Dish [name=" + name + "]";
17    }
18}
19
20
21
22
23
24
25
26
27
28
29
30
31

```

Console X Coverage

```

<terminated> App (2) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.jdk/Contents/Home/bin/java (Dec 14, 2021, 2:51:07 PM – 2:51:08 PM)
questionOne.App$1@0x000000800c00c20@6b95977
Dish [name=Meat of your choice]
Dish [name=Waffle]
Dish [name=egg]
Dish [name=potato]
Dish [name=really black coffee]

```

Q1 dish

eclipse-workspace - MySQLweek5/src/questionThree/App.java - Eclipse IDE

```

1 package questionThree;
2
3 import java.util.Optional;
4 import java.util.NoSuchElementException;
5
6 public class App {
7     public static void main(String[] args) {
8         Dish d = new Dish("Big5");
9         try {
10             b(d);
11         } catch(NoSuchElementException e) {
12             System.out.println(e.getMessage());
13         }
14     }
15
16     private static void b(Dish d) {
17         Optional<Dish> opt = Optional.ofNullable(d);
18         System.out.println(a(opt));
19         a(Optional.empty());
20     }
21
22     private static Dish a(Optional<Dish> opt) {
23         Dish d = opt.orElseThrow(()->new NoSuchElementException("Nada."));
24         return d;
25     }
26
27
28
29
30
31
32
33
34
35

```

Console X Coverage

```

<terminated> App (2) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.jdk/Contents/Home/bin/java (Dec 14, 2021, 2:51:07 PM – 2:51:08 PM)
questionOne.App$1@0x000000800c00c20@6b95977
Dish [name=Meat of your choice]
Dish [name=Waffle]
Dish [name=egg]
Dish [name=potato]
Dish [name=really black coffee]

```

Q3 app

eclipse-workspace - MySQLweek5/src/questionThree/Dish.java - Eclipse IDE

```

1 package questionThree;
2
3 public class Dish {
4     private String name;
5     public String getName() {
6         return name;
7     }
8     public Dish(String name) {
9         this.name = name;
10    }
11    public static int compareDishes(Dish d1, Dish d2) {
12        return d1.getName().compareTo(d2.getName());
13    }
14    @Override
15    public String toString() {
16        return "Dish [name=" + name + "]";
17    }
18}
19
20
21
22
23
24
25
26
27
28
29
30
31

```

Console X Coverage

```

<terminated> App (2) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.jdk/Contents/Home/bin/java (Dec 14, 2021, 2:51:07 PM – 2:51:08 PM)
questionOne.App$5Lambda$1/0x0000000800c20@6b95977
Dish [name=Meat of your choice]
Dish [name=Waffle]
Dish [name=egg]
Dish [name=potato]
Dish [name=really black coffee]

```

## Q3 dish

eclipse-workspace - MySQLweek5/src/questionTwo/App.java - Eclipse IDE

```

1 package questionTwo;
2
3 import java.util.ArrayList;
4
5 public class App {
6     public static void main(String[] args) {
7         List<Dish> dishes = new ArrayList<>((List.of(new Dish("Waffle"),
8             new Dish("Meat of your choice"),
9             new Dish("potato"), new Dish("really black coffee"),
10            new Dish("egg")));
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35

```

Console X Coverage

```

<terminated> App (2) [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.jdk/Contents/Home/bin/java (Dec 14, 2021, 2:51:07 PM – 2:51:08 PM)
questionOne.App$5Lambda$1/0x0000000800c20@6b95977
Dish [name=Meat of your choice]
Dish [name=Waffle]
Dish [name=egg]
Dish [name=potato]
Dish [name=really black coffee]

```

## Q2 app

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows files like App.java, Dish.java, and various question and optional files.
- Code Editor:** Displays the `Dish.java` file content:1 package questionTwo;
2
3
4 public class Dish {
5 private String name;
6
7 public String getName() {
8 return name;
9 }
10 public Dish(String name) {
11 this.name = name;
12 }
13 public static int compareDishes(Dish d1, Dish d2) {
14 return d1.getName().compareTo(d2.getName());
15 }
16
17 @Override
18 public String toString() {
19 return "Dish [name=" + name + "]";
20 }
21
22
23
24
25
26
27
28
29
30 }
- Console:** Shows the output of the application execution:<terminated> App [2] [Java Application] /Library/Java/JavaVirtualMachines/jdk-17.jdk/Contents/Home/bin/java [Dec 14, 2021, 2:51:07 PM - 2:51:08 PM]
questionOne.App\$5@lambda\$1/0x000000800c00c20@6b95977
Dish [name=Meat of your choice]
Dish [name=Waffle]
Dish [name=Egg]
Dish [name=Potato]
Dish [name=Really black coffee]

Q2 dish





