

Which of the following statements is correct when the below code is executed?

```
Customer customer = customerRepository.findById(1001).get();
```

```
customer.setCard(null);
```

```
customerRepository.delete(customer);
```

*sakshi.agrawal08, please provide your response below*

- ☐ It will delete customer 1001 from CUSTOMER table and card 1234567654
- ☐ No records will be deleted from both the tables and no exception will be thrown
- ☒ It will delete customer 1001 from CUSTOMER table and no records from CARD
- ☐ No records will be deleted from both the tables and exception will be thrown

Reset



00:13:41

## OBJECTIVE QUESTIONS

```
ApplicationContext ctx = new AnnotationConfigApplicationContext(SpringConfig.c
```

```
}
```

```
}
```

**sakshi.agrawal08, please provide your response below**

StudentRepository constructor called

☐

StudentService constructor called

StudentService constructor called

☒

StudentRepository constructor called

☐

Nothing will be displayed on console

☐

The code will throw RuntimeException exception

Reset

Save

ProductServiceImpl constructor called

- ☐ ProductRepositoryImpl constructor called  
Inside Spring Boot Application

ProductRepositoryImpl constructor called

- ☒ ProductServiceImpl constructor called  
Inside Spring Boot Application

Inside Spring Boot Application

- ☐ ProductRepositoryImpl constructor called  
ProductServiceImpl constructor called

Inside Spring Boot Application

- ☐ ProductServiceImpl constructor called  
ProductRepositoryImpl constructor called

Reset

Save

What should be inserted at Line 1 and Line 2 so that above class can handle Cus

**sakshi.agrawal08, please provide your response below**

- Line 1: @RestControllerAdvice
- ☒ Line 2: @ExceptionHandler(CustomerNotFoundException.class)
- Line 1: @ControllerAdvice
- ☐ Line 2: @ExceptionHandler(CustomerNotFoundException.class)
- Line 1: @ExceptionHandler(CustomerNotFoundException.class)
- ☐ Line 2: @RestControllerAdvice
- Line 1: @Controller
- ☐ Line 2: @ExceptionHandler(CustomerNotFoundException.class)

Reset

Save

## OBJECTIVE QUESTIONS

## 4. Transaction is managed by Spring

Which of the following statements is correct when the below code is executed?

```
departmentRepository.deleteById(1001);
```

*sakshi.agrawal08, please provide your response below*

- ☐ No records will be deleted from both the tables and `SQLIntegrityConstraintViolation`
- ☒ In Department table department with `department_id = 1001` row will be deleted and `SQLIntegrityConstraintViolation` deleted.
- ☐ In Department table department with `department_id = 1001` row will be deleted and in `SQLIntegrityConstraintViolation` table no record will be deleted.
- ☐ No records will be deleted from both the tables and no exception will be thrown

Reset



ProductServiceImpl constructor called

- ☐ ProductRepositoryImpl constructor called  
Inside Spring Boot Application

ProductRepositoryImpl constructor called

- ☒ ProductServiceImpl constructor called  
Inside Spring Boot Application

Inside Spring Boot Application

- ☐ ProductRepositoryImpl constructor called  
ProductServiceImpl constructor called

Inside Spring Boot Application

- ☐ ProductServiceImpl constructor called  
ProductRepositoryImpl constructor called

Reset



### EMPLOYEE

emp_id	sdb_id
1001	2



### SDB

sdb_id	sdb_Name
1	SDB-6
2	SDB-7



### EMPLOYEE

emp_id	sdb_id
1001	1

### SDB



[1 marks]

Which of the following pointcut expression matches all the public methods in the `com.order.service` package?

*sakshi.agrawal08, please provide your response below*

- ☐ `execution(* com.order.service.*(..))`
- ☐ `execution(public com.order.service.OrderServiceImpl.*(..))`
- ☐ `execution(* com.order.service.*.*(..))`
- ☒ `execution(* public com.order.service.OrderServiceImpl.*(..))`

Reset

Save

DELL



**sakshi.agrawal08, please provide your response below**

- Sort sortByPrice=Sort.by("price").descending();
- ☒ Iterable<Laptop> laptops = laptopRepository.findAll(sortByPrice);
- Sort sortByPrice = Sort.by("price");
- ☐ Iterable<Laptop> laptops = laptopRepository.findAll(sortByPrice);
- Sort sortByPrice = new Sort("price");
- ☐ Iterable<Laptop> laptops = laptopRepository.findAll(sortByPrice);
- Sort sortByPrice = new Sort("price");
- ☐ sortByPrice.descending();
- Iterable<Laptop> laptops = laptopRepository.findAll(sortByPrice);

Reset

Save

Which of the following statements when inserted at Line 1 brings Many-To-One

**sakshi.agrawal08, please provide your response below**

- ☒ @ManyToOne(cascade=CascadeType.ALL)
- ☐ @JoinColumn(name = "project\_id")
- ☐ @ManyToOne(cascade=CascadeType.ALL)
- ☐ @JoinColumn(name = "project\_id", unique = true)
- ☐ @ManyToOne(cascade=CascadeType.ALL)
- ☐ @JoinColumn(name = "employee\_project")
- ☐ @ManyToOne(cascade=CascadeType.ALL)
- ☐ @JoinColumn(name = "employee\_project", unique=true)

Reset

Save

@Autowired

```
public void setProductRepository(ProductRepository productRepository) {  
    this.productRepository = productRepository;  
}  
}
```

Which of the following statements are CORRECT about ProductService class? Choose all that apply.

*sakshi.agrawal08, please provide your response below*

- ☒ ProductRepository is a dependency of ProductService.
- ☐ ProductService is a dependency of ProductRepository.
- ☒ The dependency is injected using setter injection.
- ☐ The dependency is injected using constructor injection.

Reset

Save

```
public interface TraineeRepository extends CrudRepository<Trainee, Integer> {  
}
```

Which of the following is a valid method supported by the `TraineeRepository` interface?

*sakshiLagrowal08, please provide your response below*

- ☐ void delete(Integer traineeId)
- ☒ void deleteById(Integer traineeId)
- ☐ void deleteById(String traineeId)
- ☐ void delete(String traineeId)

Cancel

Save

```
Product product2 = new Product();
```

```
product2.setProductId("P103");
```

```
product2.setProductName("AC");
```

```
entityManager.persist(product2);
```

***sakshi.agrawal08, please provide your response below***

- ☐ 1 SELECT query and 2 INSERT queries
- ☒ 1 SELECT query, 1 UPDATE query and 1 INSERT query
- ☐ 1 SELECT query and 1 INSERT query
- ☐ 1 SELECT query and after that code will throw UniqueConstraintViolationException

Reset

Save

@Bean

private Trainee trainee() {

☐ return new Trainee ();

}

@Bean

public Trainee trainee() {

☒ return new Trainee ();

}

@Bean

public void trainee(){

☐ Trainee trainee = new Trainee ();

}

Reset

Save

String query = "select a from Account a where a.accountStatus = :sta

Query q = entityManager.createQuery(query);

//Line 1

List<Account> list = query.getResultList();



***sakshi.agrawal08, please provide your response below***

- ☐ q.setParameter(1, "INACTIVE");
- ☐ q.setParameter("accountStatus", "INACTIVE");
- ☐ q.setParameter("INACTIVE", "status" );
- ☒ q.setParameter("status", "INACTIVE");

Reset





}

Assumption:

1. All necessary imports and configurations are done

Which of the following statements can be added at Line 1 to handle HTTP GET request?

**sakshi.agrawal08, please provide your response below**

- ☒ @GetMapping("/flight/{flightNumber}")
- ☐ @GetMapping("flight/fNumber")
- ☐ @GetMapping("/flight/{fNumber}")
- ☐ @GetMapping("/flight/flightNumber")

Reset

Save

2. entityManager is a valid reference of JPA EntityManager
3. Transaction is managed by Spring

Choose the correct statement which can be used to execute the below query?

Query query = entityManager.createQuery("SELECT a.authorName,a.authorAge FROM

**sakshi.agrawal08, please provide your response below**

- ☐ List<String> list = query.getResultList();
- ☐ Author author = query.getSingleResult();
- ☒ List<Object []> list = query.getResultList();
- ☐ List<Author> list = query.getResultList();

Reset



☒ Line 2: @NotEmpty @Size(min=10)

Line 3: @PastOrPresent

Line 1: @NotNull @Size(min=4, max=4)

☐ Line 2: @NotEmpty @Min(value=10)

Line 3: @PastOrPresent

Line 1: @NotNull @Size(min=4, max=4)

☐ Line 2: @NotEmpty @Size(min=10)

Line 3: @Past

Line 1: @NotNull @Max(value=9999) @Min(value=1000)

☐ Line 2: @NotEmpty @Size(min=10)

Line 3: @FutureOrPresent

Reset

Save

@Entity

```
public class Product{
```

```
    @Id
```

```
    @Column(name="productid")
```

```
    private Integer productid;
```

```
    private String productname;
```

```
    //getters & setters
```

```
}
```

***sakshi.agrawal08, please provide your response below***

- ☐ Both (i) and (ii)
- ☒ Only (i)
- ☐ Neither (i) nor (ii)
- ☐ Only (ii)

Reset

Save

Sam

Sam@123

Assumption:

1. All necessary imports and configurations are done
2. Password is stored in memory as plain text

**sakshi.agrawal08, please provide your response below**

- ☐ `auth.inMemoryAuthentication().withUser("Sam").password("Sam@123").roles("ROLE_USER")`
- ☒ `auth.inMemoryAuthentication().withUser("Sam").password("{noop}Sam@123").roles("ROLE_USER")`
- ☐ `auth.inMemoryAuthentication().withUser("{noop}Sam").password("Sam@123").roles("ROLE_USER")`
- ☐ `auth.inMemoryAuthentication().withUser("Sam").password("{noop}Sam@123").roles("ROLE_USER")`

Reset

Save