**MCQ Spring and JPA**

**Total Marks 50 (each 2 marks)**

**Question 1: Radhika is working in an application where she is using Spring-JPA integration. Now she wants to create Pk-Fk relation b/w two tables , so Which of the following annotation she should use?**

1. @ JoinedKey
2. **@JoinColumn**
3. @ ForeignKey
4. @JoinedColumn

**Question 2: While implementing JPA, Shivansh wants to link two tables in his Application through a relation table Which of the following annotation, he should use to implement the same?**

1. @RelationTable
2. @RelationalTable
3. @JoinedTable
4. **@JoinTable**

**Question 3: Brijesh wants to implement Inheritance in his Banking application. Which of the following strategy of Is-A mapping, requires discriminator column?**

1. **Single Table Mapping**
2. Joined Table Mapping
3. Table Per Class Mapping
4. None of the Above

**Question 4: Ram is building an Employee management system where he wants to relate Manager IS-A Employee ,In order to do the same Which of the following annotation he should use in his application?**

1. **@Inheritance**
2. @IsAMapping
3. @InheritanceMapping
4. @MappedBy

### Question 5: Observe following code…

### @Component ("resourceBean")

### @Scope("singleton")

### public class GlobalInvestment {

### private @Value ("GI Pvt. Ltd.")String firmName;

### Which statement is TRUE?

1. Bean will be declared as Entity Bean of Hibernate
2. Scope of the bean is 'Prototype' in JVM
3. **The instance field in an object will be initialized by a string 'GI Pvt. Ltd.’**
4. The @Value annotation is applicable on property and not on private fields.

**Question 6: Which of the following is not the common implementations of the ApplicationContext?**

1. The FileSystemXmlApplicationContext container loads the definitions of the beans from an XML file. The full path of the XML bean configuration file must be provided to the constructor.
2. The ClassPathXmlApplicationContext container also loads the definitions of the beans from an XML file. Here, you need to set CLASSPATH properly because this container will look bean configuration XML file in CLASSPATH.
3. The WebXmlApplicationContext: container loads the XML file with definitions of all beans from within a web application.
4. **The ServletContextApplicationContext: Container converts a servlet into Spring Application context.**

### Question 7: \_\_\_\_\_\_\_\_\_\_\_\_\_ annotation simply indicates that the affected bean property must be populated at configuration time, through an explicit property value in a bean definition or through autowiring. The container throws BeanInitializationException if the affected bean property has not been populated.

### @Required

### @Qualifier

### @Autowired

### @Resource

### Question 8: The \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ represents a point in an application where we can plug-in an AOP aspect. It is the actual place in the application where an action will be taken using Spring AOP framework.

### Proxy

### join point

### pointcut

### Advice

### Target

### Question 9: Which Transaction management type is more preferable in Spring Framework?

### Declarative transaction management

### Programmatic transaction management

### None of the Above.

### Both A and B

### Question 10: Which out of following is NOT sub-annotation of @Component?

### The @Bean

### The @Service

### The @Repository

### The @Controller

### Question 11: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ provide access to the application behaviour that you typically define through a service interface. It interprets user input and transform it into a model that is represented to the user by the view. Spring implements it in a very abstract way, which enables you to create its wide variety.

1. **Controllers**
2. Advice
3. Request Mapping
4. WebApplicationContext
5. None of the Above

**Question 12: Which of the following collection Spring offers for collection configuration** **elements:**

1. The <list> type is used for injecting a list of values, in the case that duplicates are allowed.
2. The <set> type is used for wiring a set of values but without any duplicates.
3. The <map> type is used to inject a collection of name-value pairs where name and value can be of any type.
4. The <props> type can be used to inject a collection of name-value pairs where the name and value are both Strings.
5. **All the Above.**

### Question 13: Which are the important beans lifecycle methods?

1. afterPropertiesSet()
2. destroy()
3. **Both A and B**
4. None of the Above

### Question 14: If the bean implements IntializingBean, its \_\_\_\_\_\_\_\_\_\_\_\_ method is called. If the bean has init method declaration, the specified initialization method is called.

1. postProcesserBeforeInitialization()
2. **afterPropertySet()**
3. setBeanFactory()
4. setBeanName()

### Question 15: What is the role of ContextLoaderListener?

1. It starts a web container
2. It stats a WebApplicationContext
3. It handles all requests and responses
4. **It creates and starts the ApplicationContext.**

### Question 16: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ an aspect of Inversion of Control (IoC), is a general concept, and it can be expressed in many different ways. This concept says that you do not create your objects but describe how they should be created. You don’t directly connect your components and services together in code but describe which services are needed by which components in a configuration file. A container (the IOC container) is then responsible for hooking it all up.

1. BeanPostProcessor
2. ApplicationContext
3. Aspect Oriented Programming
4. **Dependency Injection**
5. None of the above

### Question 17: The\_\_\_\_\_\_\_\_\_\_ is built on the application context module, providing a context that is appropriate for web-based applications. This module also contains support for several web-oriented tasks such as transparently handling multi-context environment and programmatic binding of request parameters to your business objects. It also contains integration support with different view technologies like Apache Velocity.

1. **Spring WEB module**
2. Spring AOP module
3. Spring ORM module
4. Spring DAO module
5. None of the above

### Question 18: Which of the Following are some of the important roles of an entity manager:

1. The entity manager implements the API and encapsulates all of them within a single interface.
2. Entity manager is used to read, delete and write an entity.
3. An object referenced by an entity is managed by entity manager.
4. **All the Above.**
5. None of the above

### Question 19: The WebApplicationContext is an extension of the plain ApplicationContext that has some extra features necessary for web applications. It differs from a normal ApplicationContext in that it is capable of resolving themes, and that it knows which servlet it is associated with.

1. **True**
2. False

**Question 20: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ method is to create and return an EntityManagerFactory for the named persistence unit. Thus, this method contains the name of persistence unit passed in the Persistence.xml file.**

1. **createEntityManagerFactory()**
2. getEntityManager ()
3. obtainEntityManager()
4. createEntityManager()
5. None of the Above

### Question 21: What is the output of the following code?(Assume all the other configuration file and entity class exists).

1. **import** javax.persistence.\*;
3. **import** com.sbk.jpa.student.\*;
5. **public** **class** FindStudent {
6. **public** **static** **void** main(String args[])
7. {
8. EntityManagerFactory emf=Persistence.createEntityManagerFactory("Student\_details");  //Line 1
9. EntityManager em=emf.createEntityManager();   //Line 2


13. StudentEntity s=em.find(StudentEntity.**class**,101);  //Line 3
15. System.out.println("Student id = "+s.getStudentId());
16. System.out.println("Student Name = "+s.getStudentName());
17. System.out.println("Student Age = "+s.getStudentAge());
19. }
20. }
21. Error at Line 1
22. Error at Line 2
23. Error at Line 3
24. **No Error, the program work normally and print student details**

**Question 22: What does the code at line 7 state?**

1. **package** com.sbk.jpa.inheritence;
2. **import** java.io.Serializable;
3. **import** javax.persistence.\*;
5. @Entity
6. @Table(name="employee\_details")
7. @Inheritance(strategy=InheritanceType.TABLE\_PER\_CLASS)
8. **public** **class** Employee **implements** Serializable {
10. @Id
11. **private** **int** eid;
12. **private** String ename;
14. **public** Employee(**int** eid, String ename) {
15. **super**();
16. **this**.eid = eid;
17. **this**.ename = ename;
18. }
19. **public** Employee() {
20. **super**();
21. //getters and setter
22. }
23. **for each sub entity class a separate table is generated.**
24. a separate table is generated for every entity class. The attribute of each table is joined with the primary key. It removes the possibility of duplicacy.
25. instances of the multiple entity classes are stored as attributes in a single table only.
26. None of the Above

### Question 23: JPQL is an extension of Entity JavaBeans Query Language (EJBQL), adding the following important features to it

1. **True**
2. False

### Question 24: Which of the following is used to contain the information about the database such as driver class name, connnection URL, username and password.

1. DriverManager
2. DataSource
3. **DriverManagerDataSource**
4. **OracleDriverManager**

### Question 25: What is the role of @RequestMapping annotation?

1. The @RequestMapping annotation indicates that a particular class serves the role of a request controller.
2. **The @RequestMapping annotation is used to map a URL to either an entire class or a particular handler method.**
3. The  @RequestMapping  annotation simply indicates that the affected bean property must be populated at configuration time, through an explicit property value in a bean definition or through autowiring.
4. The  @RequestMapping  Annotation is used along with @Autowired annotation to remove the confusion by specifying which exact bean will be wired.