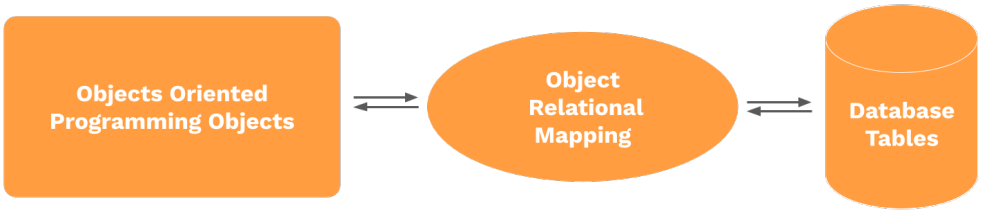
Java Application Development

Assignment Solutions

PW SKILLS

1. What is ORM in Hibernate?

Ans: Hibernate ORM stands for Object Relational Mapping. This is a mapping tool pattern mainly used for converting data stored in a relational database to an object used in object-oriented programming constructs. This tool also helps greatly in simplifying data retrieval, creation, and manipulation.



2. What are the advantages of Hibernate over JDBC?

Ans: The advantages of Hibernate over JDBC are listed below:

Clean Readable Code: Using hibernate, helps in eliminating a lot of JDBC API-based boiler-plate codes, thereby making the code look cleaner and readable.

HQL (Hibernate Query Language): Hibernate provides HQL which is closer to Java and is object-oriented in nature. This helps in reducing the burden on developers for writing database independent queries. In JDBC, this is not the case. A developer has to know the database-specific codes.

Transaction Management: JDBC doesn't support implicit transaction management. It is upon the developer to write transaction management code using commit and rollback methods. Whereas, Hibernate implicit provides this feature.

Exception Handling: Hibernate wraps the JDBC exceptions and throws unchecked exceptions like JDBCException or HibernateException. This along with the built-in transaction management system helps developers to avoid writing multiple try-catch blocks to handle exceptions. In the case of JDBC, it throws a checked exception called SQLException thereby mandating the developer to write try-catch blocks to handle this exception at compile time.

Special Features: Hibernate supports OOPs features like inheritance, associations and also supports collections. These are not available in JDBC.

3. What are some of the important interfaces of the Hibernate framework?

Ans: Hibernate core interfaces are:

- Configuration
- SessionFactory
- Session
- Criteria
- Query
- Transaction

4. What can you tell about the Hibernate Configuration File?

Ans: Hibernate Configuration File or hibernate.cfg.xml is one of the most required configuration files in Hibernate. By default, this file is placed under the src/main/resource folder.
The file contains database related configurations and session-related configurations.

Hibernate facilitates providing the configuration either in an XML file (like hibernate.cfg.xml) or a properties file (like hibernate.properties).
This file is used to define the below information:

- Database connection details: Driver class, URL, username, and password.
- There must be one configuration file for each database used in the application, suppose if we want to connect with 2 databases, then we must create 2 configuration files with different names.
- Hibernate properties: Dialect, show_sql, second_level_cache, and mapping file names.

5. What is the difference between first level cache and second level cache?

Ans:

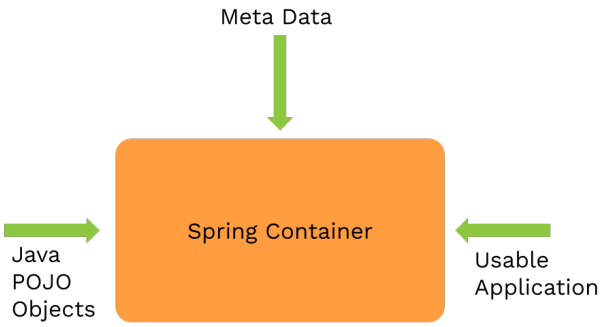
First Level Cache	Second Level Cache
This is local to the Session object and cannot be shared between multiple sessions.	This cache is maintained at the SessionFactory level and shared among all sessions in Hibernate.
This cache is enabled by default and there is no way to disable it.	This is disabled by default, but we can enable it through configuration.
The first level cache is available only until the session is open, once the session is closed, the first level cache is destroyed.	The second-level cache is available through the application's life cycle, it is only destroyed and recreated when an application is restarted.

6. What are the features of Spring Framework?

- Spring framework follows a layered architecture pattern that helps in the necessary components selection along with providing a robust and cohesive framework for J2EE applications development.
- The AOP (Aspect Oriented Programming) part of Spring supports unified development by ensuring separation of application's business logic from other system services.
- Spring provides a highly configurable MVC web application framework which has the ability to switch to other frameworks easily.
- Provides provision of creation and management of the configurations and defining the lifecycle of application objects.
- Spring has a special design principle which is known as IoC (Inversion of Control) that supports objects to give their dependencies rather than looking for creating dependent objects.
- Spring is a lightweight, java based, loosely coupled framework.
- Spring provides a generic abstraction layer for transaction management that is also very useful for container-less environments.
- Spring provides a convenient API to translate technology-specific exceptions (thrown by JDBC, Hibernate or other frameworks) into consistent, unchecked exceptions. This introduces abstraction and greatly simplifies exception handling.

7. What do you mean by IoC (Inversion of Control) Container?

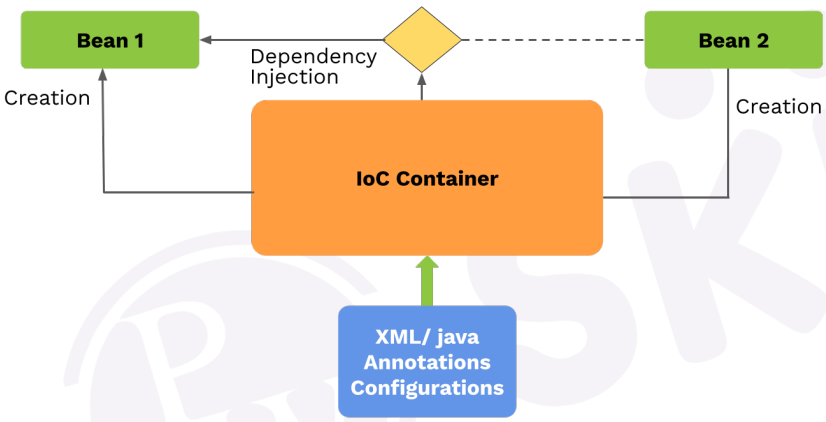
Ans: Spring container forms the core of the Spring Framework. The Spring container uses Dependency Injection (DI) for managing the application components by creating objects, wiring them together along with configuring and managing their overall life cycles. The instructions for the spring container to do the tasks can be provided either by XML configuration, Java annotations, or Java code.



8. What do you understand about Dependency Injection?

Ans: The main idea in Dependency Injection is that you don't have to create your objects but you just have to describe how they should be created.

The components and services need not be connected by us in the code directly. We have to describe which services are needed by which components in the configuration file. The IoC container present in Spring will wire them up together.



- In Java, the 2 major ways of achieving dependency injection are:
- **Constructor injection:** Here, the IoC container invokes the class constructor with a number of arguments where each argument represents a dependency on the other class.

- **Setter injection:** Here, the spring container calls the setter methods on the beans after invoking a no-argument static factory method or default constructor to instantiate the bean.