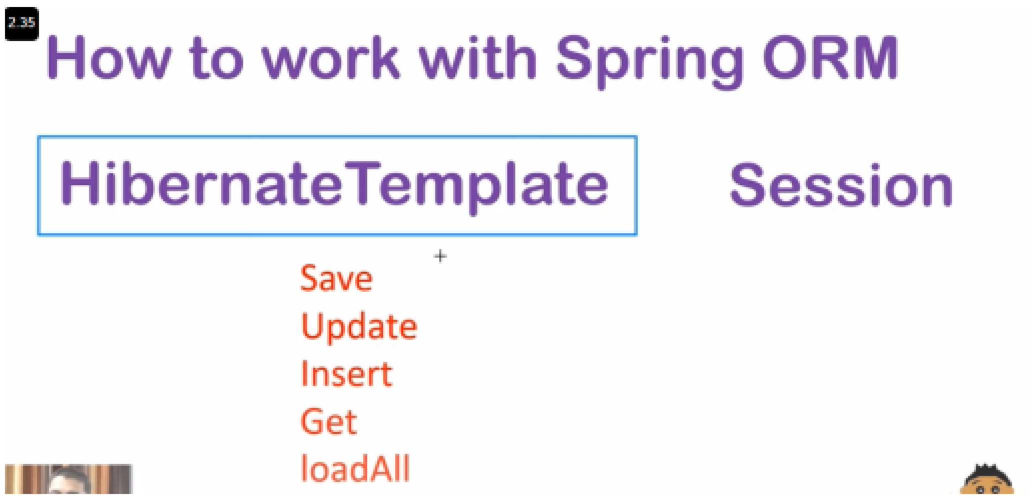
Spring ORM or Spring with ORM

Advantages of Using Spring ORM :

1)Less Coding 2) Easy to Test 3) Better Exception Handling
4) Integerated Transaction Manag.

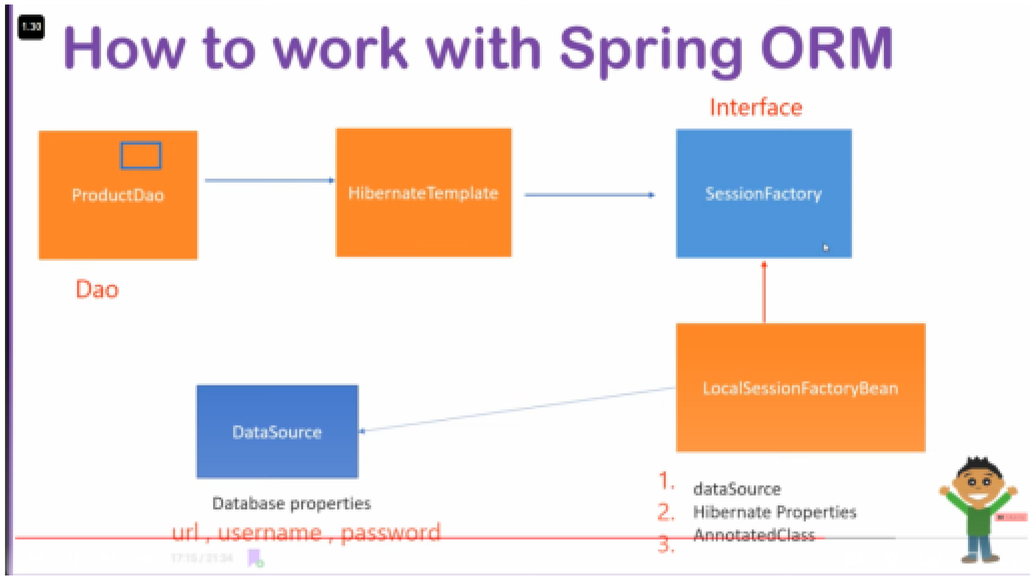


2.35

How to work with Spring ORM

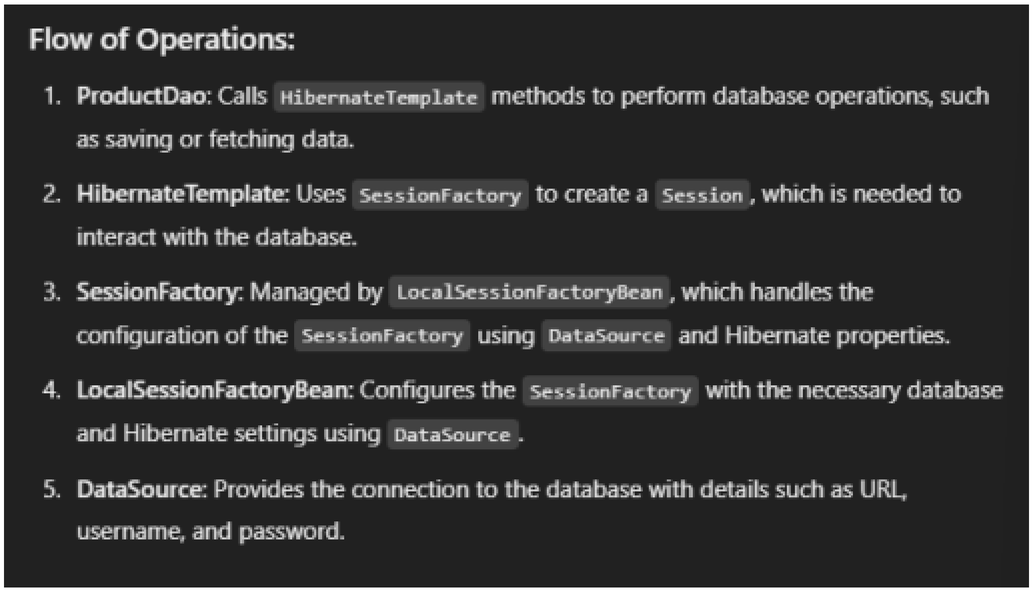
HibernateTemplate Session

Save
Update
Insert
Get
loadAll



DAO - it contains / or it is the class or interface that interacts with the dataBase.
Hibernate Template - Main class that we use to execute these operations which automatically manges sessions and transactions. Connects ProductDAO - SessionFactory

SessionFactory - HT class need an object of SessionFactory to work , and SF class is used for transactions but since it is an interface we cannot create its object so we use its implemented class LocalSessionFactoryBean
LSFB - It needs three things to work prop.. 1) DataSource (url , name , pass) , 2) HibernateProperties (dialect,sql) ,3) AnnotedClass



Flow of Operations:

1. **ProductDao:** Calls `HibernateTemplate` methods to perform database operations, such as saving or fetching data.
2. **HibernateTemplate:** Uses `SessionFactory` to create a `Session`, which is needed to interact with the database.
3. **SessionFactory:** Managed by `LocalSessionFactoryBean`, which handles the configuration of the `SessionFactory` using `DataSource` and Hibernate properties.
4. **LocalSessionFactoryBean:** Configures the `SessionFactory` with the necessary database and Hibernate settings using `DataSource`.
5. **DataSource:** Provides the connection to the database with details such as URL, username, and password.

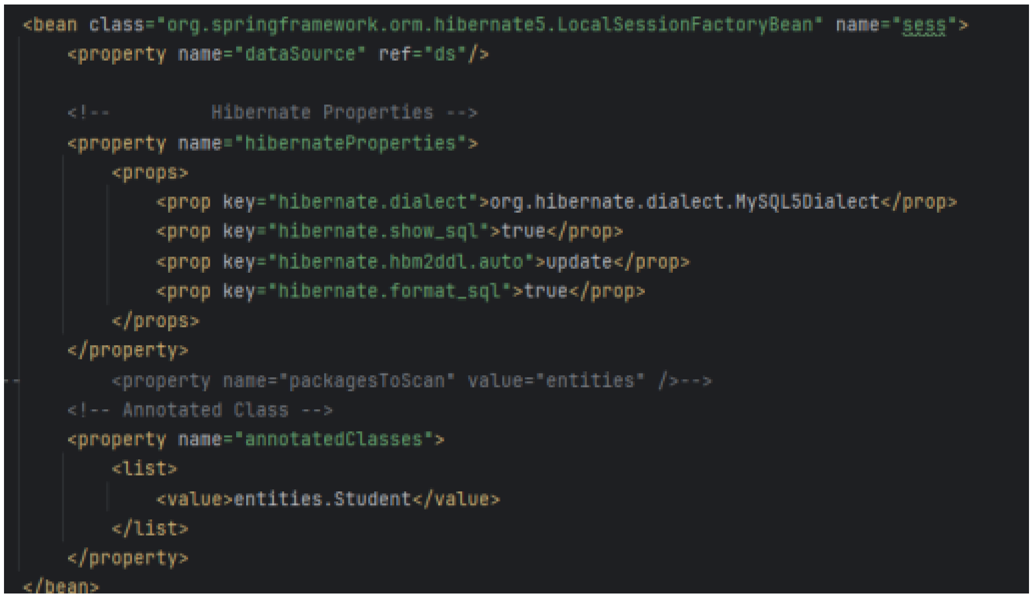
Adding Dependencies

Add Dependencies to your maven project pom.xml - > spring Core , Spring Context , SPringORM , MySQL Connector , Hibernate.

Configuring XML

Remember to use the same version of all , hibernate , spring , spring core , spring context in the pom.xml file.

How to create / add hibernateProperties and annoted classes in LocalSessionFactoryBean



```
<bean class="org.springframework.orm.hibernate5.LocalSessionFactoryBean" name="sess">
    <property name="dataSource" ref="ds"/>

    <!--         Hibernate Properties -->
    <property name="hibernateProperties">
        <props>
            <prop key="hibernate.dialect">org.hibernate.dialect.MySQL5Dialect</prop>
            <prop key="hibernate.show_sql">true</prop>
            <prop key="hibernate.hbm2ddl.auto">update</prop>
            <prop key="hibernate.format_sql">true</prop>
        </props>
    </property>
        <property name="packagesToScan" value="entities" />-->
    <!-- Annotated Class -->
    <property name="annotatedClasses">
        <list>
            <value>entities.Student</value>
        </list>
    </property>
</bean>
```

Now after doing all this like creating class Student , Creating StudentDAO (with CRUD Methods called using HibernateTemplate Object) and now using driverClass or Application u are calling methods , after creating all beans , u will get this error **" Write Operations are not allowed on read only method "**

org.springframework.orm.hibernate5.TransactionManager

@Transactional

xmlns:tx="*http://www.springframework.org/schema/tx*"

http://www.springframework.org/schema/tx

<tx:annotation-driven />