Agenda:

Introduction to Transaction Management

Local Transaction Vs Distributed Transaction

Need of Spring Transaction Management

Implementing of Spring Transaction management using Annotation Driven Approach

Transaction Attributes

---------------------------------------------------------------------------------------------------------------

Hibernate:

ORM Tool (Object Relational Mapping)

->Gavin King

Steps:

1.Hibernate.cfg.xml

Specify username, password,url,drivername,dialect,hbm.ddl.auto,show-sql,format-sql + Mapping files

2.Mapping (product.hbm.xml) / Annotation

Create a Configuration Object(read xml files)

Build a SessionFactory

->Thread safe

->Singleton

3.CRUD - Open a Session

Transaction

close Session

Alternate to this they came up with Hibernate Template

CRUD - Open a Session

Transaction

close Session

In Hibernate 3.2 -> getCurrentSession() (Taking care of open and closing )

Dependencies:

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>5.3.25</version>

</dependency>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.3.31.Final</version>

</dependency>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-java8</artifactId>

<version>5.3.31.Final</version>

</dependency>

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

<version>1.2.17</version>

</dependency>

---------------------------------------------------------------------------------------------------------------

Spring ORM:

1.LocalSessionFactoryBean -> Where you can create Session Factory and perform CRUD operations

2.DataSource -> DriverManagerDaaSource

update -> It will only update

Saveorupdate -> if data present it will update if not data will be inserted

Ways of implementing HIbernate:

1.Session

2.HQL Hibernate Query Language -> It supports select and delete

3.Criteria API -> All CRUD

4.Native Query -> certain sub queries , Join can be done here

Exception in thread "main" org.hibernate.HibernateException:

After Inserting a record -> commit will be performed

In Hibernate -> Auto-commit = false

Transaction Management:

1.PlatformTransactionManager (i)

Perform commit and rollback

JDBC -> Connection -> Use DataSourceTransactionManager

Hibernate -> Transaction API -> HibernateTransactionManager

JPA -> JPATransactionManager

Where to apply ?

@Transaction -> We will apply for the code which has to be managed by Transaction Manager(requires commit and rollback)

class Level -> Applied to all the method

method level

@EnableTransactionManagement -> enable auto-commit

ACID:

A white background with black and green lines

Description automatically generated with medium confidence

A white background with black text

Description automatically generated

It means that Tx should take place in such a way that it is the only Tx where resource is accessed inside the Database.

A white background with black text

Description automatically generated

After rollback the T2 which is having data is the one which is not existing because rollback is done, and that data is updated.

A white background with black text

Description automatically generated

T1 will read the data. T2 updates the same data and commits it. Now if again T1 will read the same data and retrieves different data.

A close up of a text

Description automatically generated

T1 will query he will get two values suppose.

When someone adds a record, and it is matching with the query which was run by T1. Now T1 will get 3 Rows.

When we set SQL standard with following isolation level:A white text with black text

Description automatically generated

2 -> How they eliminate Dirty Read ? On the current row I will put lock. Thus, preventing other tx with reading updating and delete. Their read will be on hold.

3 -> which ever the search criteria are matching it will put read lock on all rows.

4th one is the most reliable one.

Serializable -> Serially executing. But performance will degrade because it will read and then insert and then read etc. This isolation level it depends on SQL server. Oracle supports Read Uncommittable and Serializable. SQL supports all 4

A screenshot of a computer

Description automatically generated

Let database chose isolation level.

When to begin Transaction ? Or whether to get Existing Transaction or not that’s where Transaction Propagation comes into picture.

A white paper with black text

Description automatically generated

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

Here my behavior should respond withing 60 seconds else will throw an exception.

By default, if any unchecked exception is raised from your behavior , rollback will be performed. For checked exception it will not perform rollback