# Laborator 7: Hibernate, Conectarea la o bază de date MySql

## Hibernate

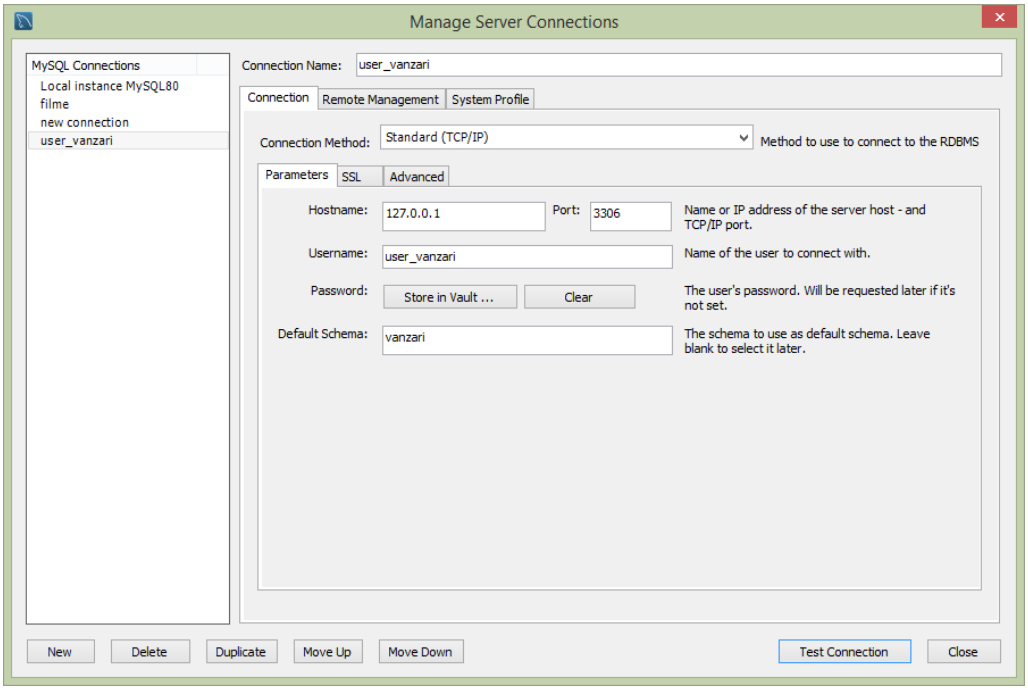
**ORM** Object Relational Mapping
Este utilizat pentru persitența în baza de date.
Folosește JDBC
Minimizează codul scris pentru utilizarea JDBC.

http://hibernate.org/

1. Instalați MySql https://dev.mysql.com/downloads/installer/ și MySQL Workbench. Ip-ul/port default sunt 127.0.0.1/3306. Creați utilizatorul “filme”.

```
CREATE DATABASE sales;
CREATE USER 'user_sales'@'localhost' IDENTIFIED BY 'user_sales';
GRANT ALL PRIVILEGES ON * . * TO 'user_sales'@'localhost';
```

2. Creați o conexiune pentru utilizatorul user_vanzari.



3. Creați tabelul customer:

```
DROP TABLE IF EXISTS sales.customer;

CREATE TABLE sales.customer (
  id int(10) NOT NULL AUTO_INCREMENT,
  first_name varchar(40) DEFAULT NULL,
  last_name varchar(40) DEFAULT NULL,
  email varchar(40) DEFAULT NULL,
  PRIMARY KEY (id)
)
ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=latin1;
```

4. Deschideți proiectul LAB7_START.
5. Adugați în fișierul .pom dependența:

```
<hibernate.version>5.4.1.Final</hibernate.version>

<dependency>
      <groupId>org.hibernate</groupId>
      <artifactId>hibernate-core</artifactId>
      <version>${hibernate.version}</version>
</dependency>

<dependency>
      <groupId>mysql</groupId>
      <artifactId>mysql-connector-java</artifactId>
      <version>8.0.13</version>
</dependency>
```

6. Adaugați în pachetul model clasa Customer cu atributele id, firstName, lastName, email. Ce adnotări sunt necesare?
7. Adaugați în directorul src directorul **test/java/**. Marcați directorul **test/java** ca Test Sources Root. Adăugați în acest director pachetul com.apbdoo.lab7.

**SessionFactory**
configurată conform fișierului hibernate.cfg.xml
crează obiecte de tip sesiune
este inițializată o singură dată la pronirea aplicației

**Session**
Destinată unei conexiuni JDBC
Obiectul utilizat pentru regăsirea și actualizarea entităților
Furnizat de SessionFactory

8. Creați un test care să inițializeze un obiect SessionFactory.

```
SessionFactory factory = new Configuration()
    .configure("hibernate.cfg.xml")
    .addAnnotatedClass(Customer.class)
    .buildSessionFactory();
```

9. Salvați în baza de date un obiect de tip Customer.

```
Session session = factory.getCurrentSession();
…

// start a transaction
session.beginTransaction();

// save customer object
log.info("Saving customer...");
session.save(tempCustomer);

// commit
session.getTransaction().commit();
```

10. Adăugați în fișierul de configurare spring-mvc-servlet bean-ul care va conține un obiect de tip DataSource

```
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-c3p0</artifactId>
    <version>${hibernate.version}</version>
</dependency>
```

```
<bean id="dataSource"
class="com.mchange.v2.c3p0.ComboPooledDataSource"
          destroy-method="close">
        <property name="driverClass"
value="com.mysql.cj.jdbc.Driver" />
        <property name="jdbcUrl"
value="jdbc:mysql://localhost:3306/sales" />
        <property name="user" value="user_sales" />
        <property name="password" value="user_sales" />
        <property name="initialPoolSize" value="2"/>
        <property name="minPoolSize" value="2" />
        <property name="maxPoolSize" value="10" />
        <property name="maxIdleTime" value="20000" />
    </bean>
```

**DAO → SessionFactory → DataSouce → Data Base**

11. Configurați un bean de tip SessionFatory.

```
<dependency>
   <groupId>org.springframework</groupId>
   <artifactId>spring-orm</artifactId>
   <version>${spring.version}</version>
</dependency>
```

```
</bean><bean id="sessionFactory"

class="org.springframework.orm.hibernate5.LocalSessionFactoryBean
">
        <property name="dataSource" ref="dataSource" />
        <property name="packagesToScan"
value="com.apbdoo.lab7.model" />
        <property name="hibernateProperties">
            <props>
                <prop
key="hibernate.dialect">org.hibernate.dialect.MySQLDialect</prop>
                <prop key="hibernate.show_sql">true</prop>
            </props>
        </property>
</bean>
```

12. Adăugați configurările pentru un bean de tip TransactionManager:

```
<bean id="transactionManager"
class="org.springframework.orm.hibernate5.HibernateTransactionMan
ager">
        <property name="sessionFactory" ref="sessionFactory"/>
    </bean>

<tx:annotation-driven transaction-manager="transactionManager" />
```

13. Implementați un obiect DAO pentru entitatea Customer.

```
@Repository
public class CustomerDAOImpl implements CustomerDAO {
	@Autowired
	private SessionFactory sessionFactory;

	@Override
	@Transactional
	public List<Customer> getCustomers() {
	    Session currentSession =
sessionFactory.getCurrentSession();
	    Query<Customer> theQuery =
	                currentSession.createQuery("from Customer",
Customer.class);
	    List<Customer> customers = theQuery.getResultList();
	    return customers;
	}
}
```