# Comparison of Hibernate and Spring Data JPA

## Setup and Boilerplate Code

**Hibernate**

* Requires us to manually manage the SessionFactory, Session, and Transaction classes
* **Example:**

Session session = factory.openSession();  
Transaction tx = session.beginTransaction();

* Need to close sessions in finally blocks.

**Spring Data JPA**

* No boiler plate; it uses the JpaRepository and the methods are already implemented.
* **Example:**

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {}

## Transaction Management

|  |  |  |
| --- | --- | --- |
| **Aspect** | **Hibernate** | **Spring Data JPA** |
| **Control** | Manual (begin, commit, rollback) | Automatic (@Transactional) |
| **Rollback** | Manual (tx.rollback()) | Auto-rollback on exceptions |

## 3. Saving an Entity

**Hibernate:**

Integer id = (Integer) session.save(employee);

* Uses session.save(employee) to save an entity.
* After saving, it will return an ID (if one was generated). You need to manage the session.

**Spring Data JPA:**

employeeRepository.save(employee);

* Uses employeeRepository.save(employee).
* The save() method is provided by JpaRepository.
* No need to manage the session or check manually for exceptions.

## Exception Handling

* **Hibernate:** Catch HibernateException and rollback manually.

**Example:**

catch (HibernateException e) {

if (tx != null) tx.rollback();

e.printStackTrace();

}

* **Spring Data JPA:** Exceptions auto-translated to DataAccessException. No need for manual rollback if @Transactional is used.

## 5. Code Readability

**Hibernate:**

* More wordy, because of doing session and transaction management manually.
* More risk for error if something isn't closed properly.

**Spring Data JPA:**

* Clean and concise due to abstraction.
* Less boilerplate code, easier to maintain.

## 6. Dependency Injection

**Hibernate:** None (manual SessionFactory setup).

**Spring Data JPA:**  Built-in (@Autowired repositories).

## 7. Default Methods

* **Hibernate:** Basic CRUD via Session.
* **Spring Data JPA:** Advanced methods (findAll(), deleteById(), etc.).