**Step 1: Hibernate-Specific Annotations**

@Entity

@DynamicInsert

@DynamicUpdate

@Table(name = "employees")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(nullable = false, length = 50)

private String firstName;

@Column(nullable = false, length = 50)

private String lastName;

@ManyToOne

@JoinColumn(name = "department\_id")

private Department department;

// getters and setters

}

**Step 2: Configuring Hibernate Dialect and Properties**

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5Dialect

spring.jpa.properties.hibernate.format\_sql=true

spring.jpa.properties.hibernate.use\_sql\_comments=true

spring.jpa.properties.hibernate.generate\_statistics=true

**Step 3: Batch Processing**

**EmployeeRepository.java:**

public interface EmployeeRepository extends JpaRepository<Employee, Long> {

@Modifying

@Query("UPDATE Employee e SET e.salary = e.salary \* 1.1 WHERE e.department.id = :departmentId")

void updateSalaries(@Param("departmentId") Long departmentId);

}

**EmployeeService.java:**

@Service

public class EmployeeService {

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public void updateSalaries(Long departmentId) {

employeeRepository.updateSalaries(departmentId);

}

}

**BatchProcessingConfig.java:**

@Configuration

public class BatchProcessingConfig {

@Bean

public HibernateJpaVendorAdapter jpaVendorAdapter() {

HibernateJpaVendorAdapter vendorAdapter = new HibernateJpaVendorAdapter();

vendorAdapter.setShowSql(true);

vendorAdapter.setGenerateDdl(true);

vendorAdapter.setDatabase(Database.MYSQL);

return vendorAdapter;

}

@Bean

public LocalContainerEntityManagerFactoryBean entityManagerFactory() {

LocalContainerEntityManagerFactoryBean factory = new LocalContainerEntityManagerFactoryBean();

factory.setDataSource(dataSource());

factory.setJpaVendorAdapter(jpaVendorAdapter());

factory.setPackagesToScan("com.example.employee.entity");

factory.setJpaProperties(hibernateProperties());

return factory;

}

@Bean

public DataSource dataSource() {

return DataSourceBuilder.create().build();

}

@Bean

public Properties hibernateProperties() {

Properties properties = new Properties();

properties.put("hibernate.batch\_size", 50);

properties.put("hibernate.jdbc.batch\_versioned\_data", true);

return properties;

}

}