**Hibernate XML Configuration – SME Walkthrough**

### **Object-to-Relational Mapping Using Hibernate XML**

In XML-based Hibernate configuration, object-relational mapping is defined in an external mapping file (typically Employee.hbm.xml) rather than annotations. This file maps a Java class to a database table and its fields to table columns.

**Example –** Employee.hbm.xml**:**

<hibernate-mapping>

<class name="Employee" table="EMPLOYEE">

<id name="id" type="int">

<column name="id"/>

<generator class="native"/>

</id>

<property name="firstName" column="first\_name" type="string"/>

<property name="lastName" column="last\_name" type="string"/>

<property name="salary" column="salary" type="int"/>

</class>

</hibernate-mapping>

* <class> maps the Java class to the database table.
* <id> maps the primary key field.
* <property> elements map other class fields to corresponding columns.

This enables Hibernate to persist and retrieve Java objects using standard relational database tables.

## **Core Hibernate Components for End-to-End Operations**

### **SessionFactory**

A SessionFactory is a heavyweight object created once and used to create Session instances. It is configured using the hibernate.cfg.xml file and represents a connection to the database.

SessionFactory factory = new Configuration().configure().buildSessionFactory();

### **Session**

A Session is a lightweight object used to interact with the database. It opens a single-threaded unit of work, where objects can be persisted, retrieved, or deleted.

Session session = factory.openSession();

### **Transaction**

Hibernate uses Transaction to group multiple operations into a single atomic unit. This ensures consistency and rollback capabilities.

Transaction tx = session.beginTransaction();

### **beginTransaction()**

This method starts a new database transaction. All database operations must be done within this transaction.

Transaction tx = session.beginTransaction();

### **commit()**

Commits the transaction to make all changes permanent in the database.

tx.commit();

### **rollback()**

Rolls back the transaction if an error occurs, undoing all changes made during the transaction.

tx.rollback();

### **session.save()**

Persists a transient object into the database.

session.save(employee);

### **session.createQuery().list()**

Used to fetch multiple records using HQL (Hibernate Query Language). list() returns the results as a list.

List<Employee> list = session.createQuery("FROM Employee").list();

### **session.get()**

Fetches a single object by its primary key. Returns null if not found.

Employee emp = session.get(Employee.class, id);

### **session.delete()**

Deletes a persistent object from the database.

session.delete(emp);