**Hibernate Annotation Config implementation walk through**

### Object-to-Relational Mapping in the Employee Persistence Class

In Hibernate, the Employee class represents a database table through **object-relational mapping (ORM)**. Each field in the class is annotated to map directly to a column in a relational database table. This allows Hibernate to automatically manage data persistence between Java objects and database records.

@Entity

@Table(name = "EMPLOYEE")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "id")

private int id;

@Column(name = "first\_name")

private String firstName;

@Column(name = "last\_name")

private String lastName;

@Column(name = "salary")

private int salary;

// Getters and Setters

}

This class tells Hibernate:

* Each field represents a table column.
* The class represents the EMPLOYEE table.
* The id field is a primary key, and its value is auto-generated.

### Annotations Used for Hibernate Operations

#### @Entity

Marks the class as a **persistent entity**. Hibernate treats it as a table in the database.

#### @Table(name = "EMPLOYEE")

Specifies the **exact table name** in the database to map this entity. If omitted, Hibernate uses the class name.

#### @Id

Defines the **primary key** of the entity. Each instance must have a unique identifier for database operations.

#### @GeneratedValue(strategy = GenerationType.IDENTITY)

Automatically generates the value of the primary key using the database's identity/auto-increment feature.

#### @Column(name = "column\_name")

Maps a class field to a **specific column** in the database. Custom column names can be specified, and additional constraints like nullable, length, etc., can also be used.

## Hibernate Configuration – hibernate.cfg.xml

The configuration file tells Hibernate **how to connect to the database** and how to manage sessions and transactions.

### Key Configuration Properties

<hibernate-configuration>

<session-factory>

<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

<property name="hibernate.connection.driver\_class">com.mysql.jdbc.Driver</property>

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/test</property>

<property name="hibernate.connection.username">root</property>

<property name="hibernate.connection.password">password</property>

</session-factory>

</hibernate-configuration>

#### Dialect : Specifies the **SQL dialect** that Hibernate should use. For MySQL, it is org.hibernate.dialect.MySQLDialect.

#### Driver : Defines the **JDBC driver** class to connect to the database, e.g., com.mysql.jdbc.Driver.

#### Connection URL : Gives the **JDBC URL** that defines where the database is located and which database to connect to. Example: jdbc:mysql://localhost:3306/test.

#### Username : Username to **authenticate** the database connection.

#### Password : Password corresponding to the above username for authentication.