



BY NAGOOR BABU

HIBERNATE

COURSE MATERIAL

BY

NAGOOR BABU

DURGA

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Index

1. Course Content.....	Page 04
2. Introduction.....	Page 06
3. Steps to prepare First Hibernate Application.....	Page 20
4. Hibernate Applications with Eclipse.....	Page 31
5. Hibernate Applications.....	Page 37
6. Hibernate Persistence Object Lifecycle.....	Page 118
7. Hibernate Schema Generation Tools.....	Page 121
8. Creating SessionFactory object in Hibernate4.x version.....	Page 128
9. Primary Key Generation Algorithms in Hibernate.....	Page 132
10. Transaction Management.....	Page 148
11. Connection Pooling in Hibernate.....	Page 157
12. Bulk Operations.....	Page 178
13. Hibernate Filters.....	Page 249
14. Hibernate Mappings.....	Page 254
..	

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

COURSE CONTENT/SYLLABUS

1. Introduction

- ◆ Enterprise
- ◆ Enterprise Application
- ◆ Data Persistence
- ◆ Data Persistence through Serialization and Deserialization
- ◆ Data Persistence through JDBC
- ◆ Data Persistence through ORM
- ◆ Hibernate History
- ◆ Hibernate Features
- ◆ Hibernate Architecture

2. Steps to prepare First Hibernate Application

- ◆ Prepare Persistence Class or Object
- ◆ Prepare Mapping File
- ◆ Prepare Hibernate Configuration File
- ◆ Prepare Client Application

3. Hibernate Applications with Eclipse

- ◆ Eclipse
- ◆ Installation Process:
- ◆ Steps To Design Hibernate Applications in Eclipse IDE

4. Hibernate Applications

- ◆ Example On Hibernate SaveOrUpdate(--) method
- ◆ Example to delete records from DB
- ◆ Example To retrieve Record from DB
- ◆ HIBERNATE with My SQL Database
- ◆ AWT/GUI-Hibernate Integration Application:
- ◆ Servlet_Hibernate_Application
- ◆ JSP_Hibernate_Application
- ◆ Struts_Hibernate Integration
- ◆ Integration Of Hibernate with Struts

5. Hibernate Persistence Object Lifecycle

- ◆ Transient State
- ◆ Persistence State
- ◆ Detached State
- ◆ Removed State

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

6. Hibernate Schema Generation Tools

- ◆ SchemaExport
- ◆ SchemaUpdate
- ◆ CodeGeneration

7. Creating SessionFactory object in Hibernate4.x version

- ◆ Create Configuration Object with all configuration details
- ◆ Get all configuration details into Properties object
- ◆ Creating StandardServiceRegistryBuilder
- ◆ Get All configuration details from Properties object into StandardServiceRegistryBuilder object
- ◆ Create StandardServiceRegistry object
- ◆ Create SessionFactory object

8. Primary Key Generation Algorithms in Hibernate**9. Transaction Management****10. Connection Pooling in Hibernate**

- ◆ Connection Pooling in Hibernate
- ◆ Default Connection Pooling Mech in Hibernate
- ◆ Third Party Vendors provided Connection pooling Mechs
- ◆ Connection Pooling through Application Servers

11. Bulk Operations

- ◆ HQL [Hibernate Query Language]
- ◆ Building Blocks for HQL queries
- ◆ Native SQL
- ◆ Criterion API

12. Hibernate Filters

- ◆ Prepare a table in Database with the required columns, where we must define a filter parameter column
- ◆ Define Filter in mapping File
- ◆ Enable Filter in Session and set values to filter parameters inorder to filter the results

13. Hibernate Mappings

- ◆ Basic OR Mapping.
- ◆ Component Mapping
- ◆ Inheritance Mapping
- ◆ Association Mapping
- ◆ Retrieving the data through annotation

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

HIBERNATE

Introduction

Enterprise: It is a Business Organization or a Group of Organizations running under Single label.

Enterprise Application: It is a software Application designed for an enterprise in order to simplify their business processing.

To prepare Enterprise Applications we have to provide the following layers.

1. User Interface Layer
2. Business Processing Layer
3. Data Storage and Access Layer

1. User Interface Layer:

- It is a top most layer in enterprise applications, it will improve Look and Feel to the enterprise applications.
- It will provide starting point to the users in order to interact with enterprise application.
- It will provide very good environment to get data from users in order to execute enterprise applications.
- It will provide very good environment to perform client side data validations by executing Java script functions.
- To prepare this layer we will use a separate logic called as "Presentation Logic".
- In enterprise applications, to prepare Presentation Logic we will use the technologies like AWT, SWING, Java FX, Html, JSP, Java Script, Velocity, Freemarker,....

2. Business Processing Layer:

- It is heart of Enterprise Applications.
- It will provide very good environment to define and execute business rules and regulations which are required by the client.
- To prepare this Layer we will use a separate logic called as "Business Logic".
- To prepare this layer we will use the tech like Servlets, EJBs-Session Beans,....

3. Data Storage and Access Layer:

- It bottom most layer in enterprise applications.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

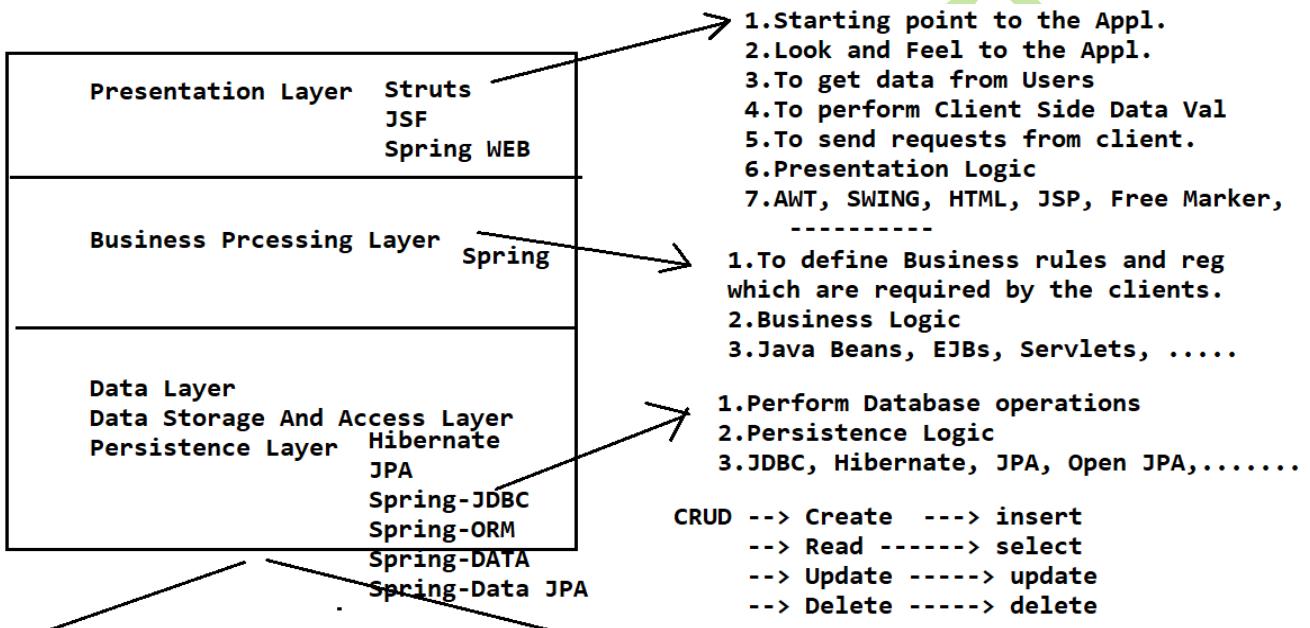
WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- Its main intention is to provide data persistency in Enterprise applications, that is, it will provide very good environment to interact with database from java applications in order to perform database operations.
- To create this layer we will use a separate logic called as "Persistence Logic".
- To prepare Persistence Logic in Enterprise applications we will use a set of technologies or products like JDBC, Hibernate, JPA, toplink,...



Data Persistency:

Representing Data permanently in Backend systems is called as "Data Persistency".

To achieve Data Persistency in database applications we will use a set of Operations [CRUD] called as Date Persistence Operations".

To achieve Data Persistency in Enterprise Applications we will use a set of tech called as "Data Persistency Tech".

To achieve Data Persistency in enterprise applications we will use the following technologies w.r.t JAVA.

- 1) Serialization and Deserialization
- 2) JDBC
- 3) ORM Implementations
 - a) Hibernate

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

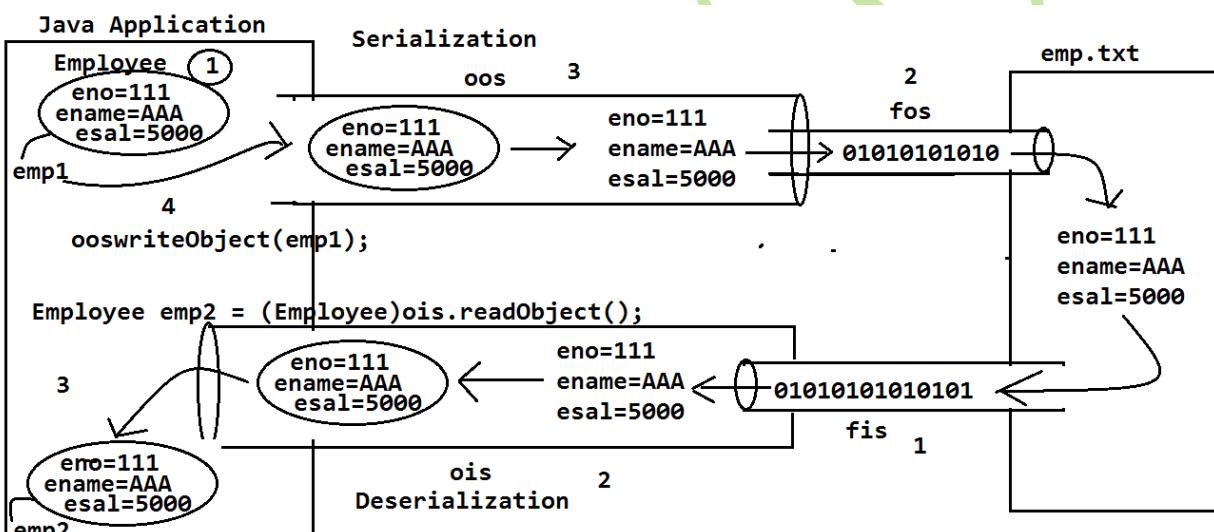
- b) EJBs Entity Beans
- c) JPA
- d) Open JPA
- e) Toplink

Data Persistence through Serialization and Deserialization: The process of Separating Data from an Object is called as "Serialization".

The process of regenerating an object on the basis of Data is called as "Deserialization".

To perform Serialization and Deserialization JAVA has provided the following two byte oriented streams

1. ObjectOutputStream for Serialization
2. ObjectInputStream for Deserialization



Steps to perform Serialization:

1. Prepare Serializable class by implementing `java.io.Serializable` marker interface:

```
class Employee implements Serializable{
    -----
}
```

```
Employee emp1 = new Employee();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Create FileOutputStream object:

```
FileOutputStream fos = new FileOutputStream("emp.txt");
```

3. Create ObjectOutputStream for Serialization.

```
ObjectOutputStream oos = new ObjectOutputStream(fos);
```

4. Write Serializable Object in ObjectOutputStream:

```
oos.writeObject(emp1);
```

Steps to perform Deserialization:**1. Create FileInputStream from Source File:**

```
FileInputStream fis = new FileInputStream("emp.txt");
```

2. Create ObjectInputStream for Deserialization:

```
ObjectInputStream ois = new ObjectInputStream(fis);
```

3. Read Deserialized Object from ObjectInputStream:

```
Employee emp2 = (Employee)ois.readObject();
```

- In Serialization and Deserialization Data Persistence mechanism, we will store data in file system ,where file system is providing permanent storage for our data, so that, Serialization and Deserialization is a Data Persistence Mechanism.
- In Serialization and Deserialization data persistence mechanism, we will use a flat file to store data, it is platform dependent, it is not suitable for the platform independent front end technologies like JAVA, .NET,....
- In Serialization and Deserialization Data Persistence mechanism, we will use two byte oriented streams like ObjectOutputStream and ObjectInputStream to perform Serialization and Deserialization, so that, this data persistence mechanism is suitable for only byte oriented data.
- In this data persistency mechanism, we are using file system to store data , it able to store less data , it able to provide less security and it able to increase data redundancy, it is not suggestible in applications.

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

- This persistency mechanism is suitable for Standalone applications, not suitable for enterprise applications.
- If we use this data persistency mechanism in Enterprise applications then we have to provide lot of java code.
- In Serialization and Deserialization data persistency mechanism we are able to perform only insertion and retrieval operations, we are unable to perform update, delete,... operations.
- In this data persistency mechanism, query language support is not existed so that database operations are very much difficult.

Data Persistency through JDBC:

JDBC is a step by step process or a technology or an API or the collection of predefined classes and interfaces or an abstraction, it will provide very good environment to interact with database from java applications in order to perform database operations from java applications.

To prepare JDBC applications we have to use the following steps.

1. Load and Register Driver:
`Class.forName("oracle.jdbc.OracleDriver");`
2. Establish Connection between Java application and Database.
`Connection con = DriverManager.getConnection(" jdbc:oracle:thin:@localhost:1521:xe", "system", "durga");`
3. Create either Statement or PreparedStatement or CallableStatement as per the requirement.
`Statement st = con.createStatement();`
4. Write and execute SQL Queries.
`ResultSet rs = st.executeQuery("select * from emp1");`
5. Close the resources.
`rs.close();
st.close();
con.close();`

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

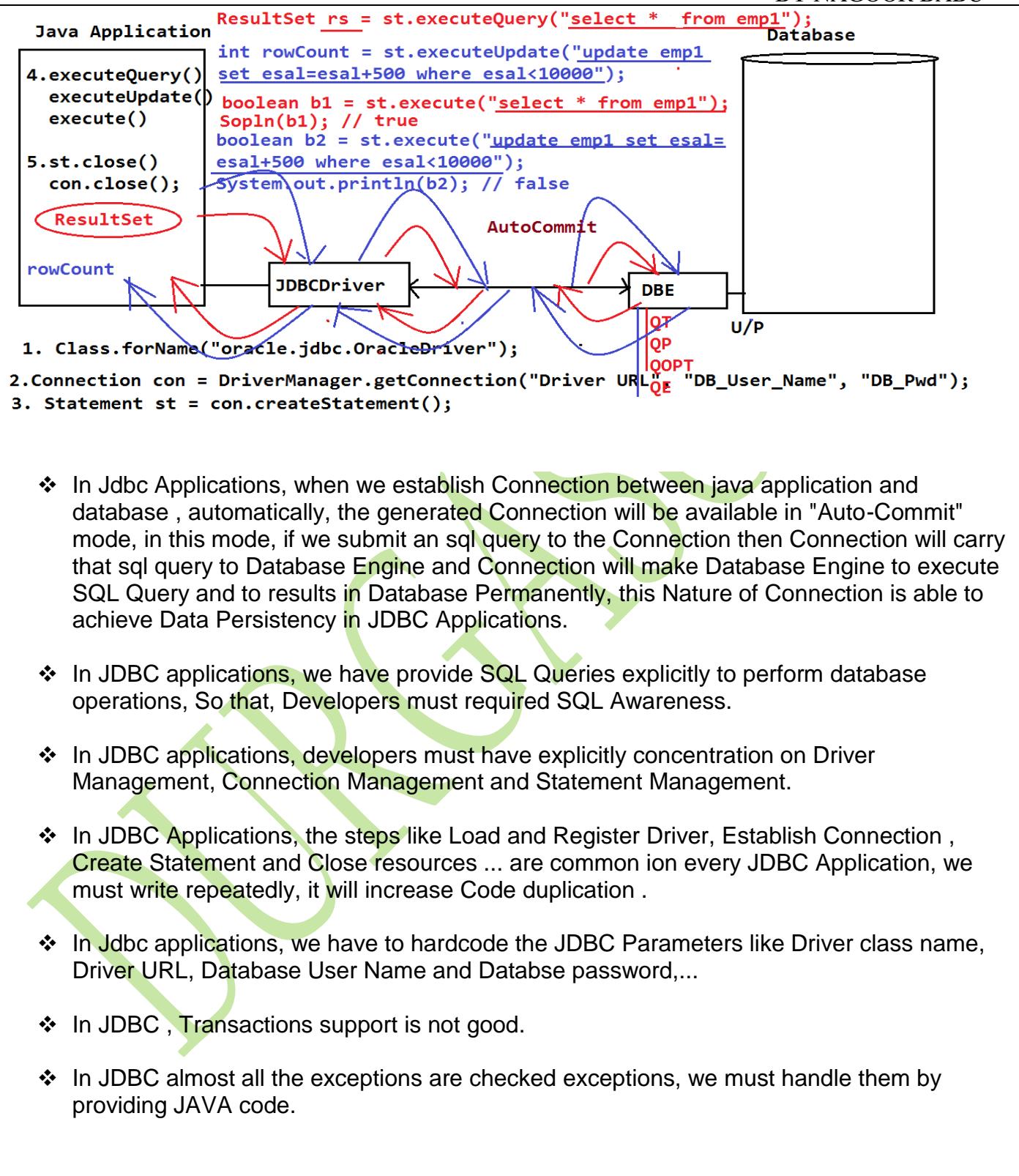
Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ❖ In JDBC applications, if we retrieve data from database then that will be stored in ResultSet object at java applications, it is not implementing java.io.Serializable interface and which is not eligible to carry in network.

Data Persistence through ORM:

In general, in Enterprise Applications to represent data we will use more than one data model like Object Oriented Data Model, Relational data Model,.....

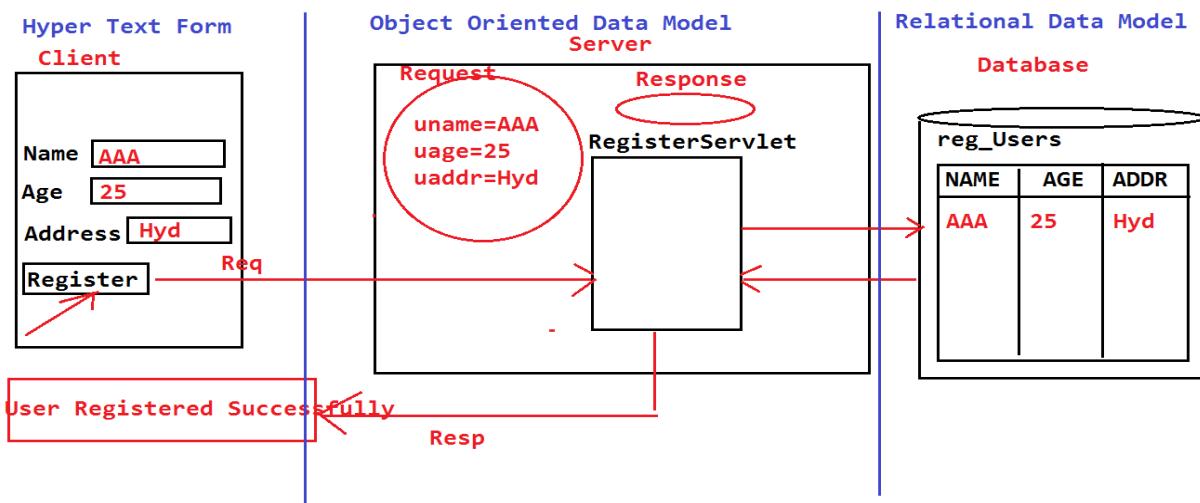
1. Paradigm Mismatches:

In enterprise applications both the data models are having their own approaches to represent data in effective manner, these differences are able to provide Paradigm Mismatches, and these mismatches are able to reduce data persistency in enterprise applications.

In general, Object Oriented Data Model and Relational data model are having the following mismatches.

In general, Object Oriented Data Model and Relational data model are having the following mismatches.

In general, Object Oriented Data Model and Relational data model are having the following mismatches.



In general, Object Oriented Data Model and Relational data model are having the following mismatches.

1. Granularity Mismatch
2. Sub types mismatch
3. Associations Mismatch

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4. Identity Mismatch

5.

To improve Data persistency in Enterprise applications we have to resolve the above specified mismatches between Data models, for this, we have to use "ORM" implementations.

1. Granularity Mismatch:

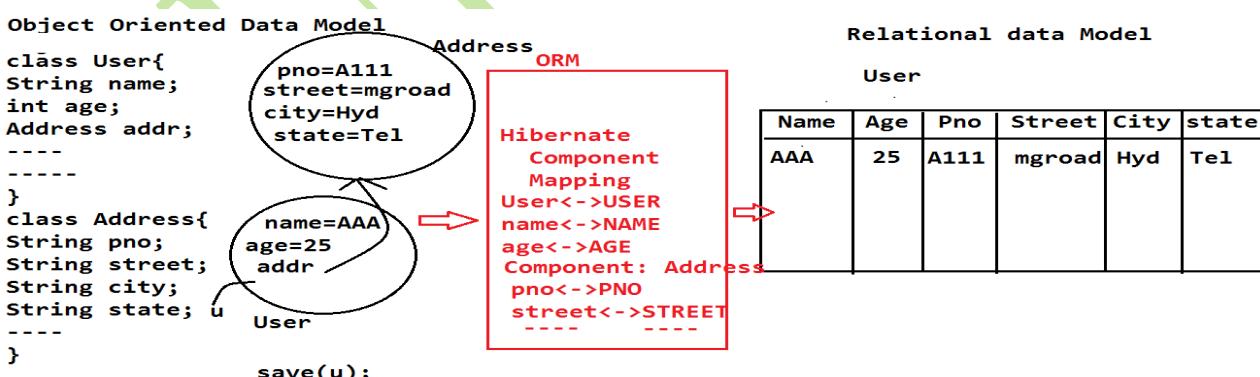
In general, Granularity is representing a count .

EX: In a database no of tables , in a table no of columns, In a class no of properties, in an application no of classes,.....

- IN Enterprise applications, we are able to represent an User in Object Oriented Data Model in one approach and in Relational data Model in another approach.
- In Object oriented Data model we may take a class User with the properties uname, uage to represent an User , here if we want to represent address details of an user then we will take a seperate class Address with the properties like pno, street, city,.... and we will declare Address class reference variable in User class.
- In Relational data Model, we may represent a User entity and its address details in the form of a single table or in the form of multiple tables with primary key and foreign key relationship.

In the above context, the no of classes [Granularity] in Object Oriented Data Model and the no of tables [Granularity] in Relational data model may not be matched, it may reduce data persistency in enterprise applications, in this context, to improve Data persistency we have to use ORM implementations like Hibernate.

For the above Granularity mismatch, Hibernate has provided a solution in the form of "**Component Mapping**".



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Sub types Mismatch:

In Object Oriented Data Model we are able to use inheritance inorder to improve Code Reusability. In Java applications, we are able to represent inheritance by using "extends" keyword.

- In Relational data Model, we are not having any feature to represent inheritance between tables, in Relational Data model we are able to achieve inheritance kind of feature by using either of the following approaches.
 1. We may take single table to represent all the properties of the classes which are included in inheritance.
 2. We may take a seperate table for each and every sub class with the super class properties and sub class properties.
 3. We may take a seperate table for super class and for sub classes with Primary Key and Foreign Key relationships.
- In the above context, Object Oriented Data Model is having its own representation to achieve INheritance and relational Data Model is having its own approaches to represent Inheritance kind of feature, this difference is representing "Inheritance Mismatch", It may reduce data persistency in enterprise applications.
- In the above context, to improve Data Persistency we have to resolve Inheritance Mismatch problem, for this, we have to use ORM implementations like Hibernate.

For the above Inheritance Mismatch, Hibernate has provided the following three types of solutions.

1. Table Per Class Hierarchy.
2. Table Per Sub class
3. Table Per Concrete Class

```
class Person{
String name;
String age;
-----
}
class Student
    extends Person{
String qual;
float agg_Per;
}
class Employee
    extends Person{
float esal;
String des;
}
```

Student
name=AAA
age=25
qual=MCA
AGG=75

Employee
name=BBB
age=30
esal=25000
des=Mgr

std
ORM
Hibernate
Inheritance
Mapping

1.Table per
class Hierarchy

2.Table per
concrete class

3.Table per
Sub class

Person					
Name	Age	qual	Agg	esal	des
AAA	25	MCA	75	null	null
BBB	30	null	null	25000	Mgr

student				Employee			
Name	Age	Qual	Agg	Name	Age	Esal	Des
AAA	25	MCA	75	BBB	30	25000	Mgr

Person	
PK - FK	Student
AAA	25
BBB	30

Employee		
Qual	AGG	Name
MCA	25	AAA

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

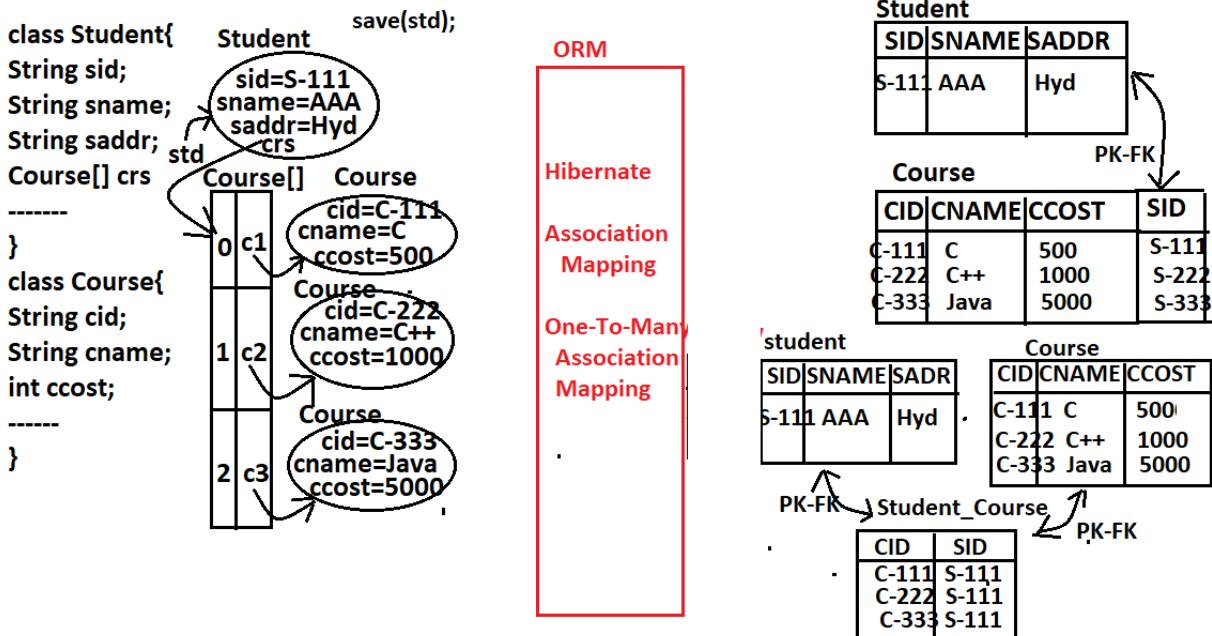
FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3. Associations Mismatch:

- ❖ IN general, in enterprise applications we will use Associations to improve communication between entities and to improve data navigation between entities.
- ❖ In Object Oriented data Model, we are able to implement associations by declaring either single reference variable or array or collection of reference variables of a class in another class.
- ❖ In Relational Data Model , we are able to achieve associations between entities either by defining Primary-Key and Foreign-Key relationships between entity tables or by using join columns or by using join tables between entity tables.
- ❖ In the above context, Object Oriented Data Model and Relational data Model both are having their own approaches to implement Associations, this difference in their approaches may create mismatch between these two data models, it may reduce Data Persistence in enterprise applications.
- ❖ In the above context, to improve data persistancy we have to resolve associations mismatch between data models, for this, we have to use ORM. Hibernate is one of the ORM implementation, it will provide solution for association mismatch in the form of "Associations Mapping" as part of Hibernate Mapping feature.



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4. Identity mismatch:

- ❖ In Object Oriented model, we are able to represent all the entities in the form of classes.
- ❖ In Object Oriented model, we are able to prepare no.of instances from a particular class.
- ❖ In the above context, it is possible to check whether the two objects are identical or not by using either == operator or .equals () method from object class.
- ❖ In case of relational model, all the records in the entity tables be identical if we use primary key column.so that no specific mechanism is existed to check whether the two records are identical or not.
- ❖ If we consider the above two data models. There is a mismatch about identity checking, it will make data persistency is compel.
- ❖ In the above context, to simplify data we have to use ORM implementations. With respect to java technology.
- ❖ ORM is a set of rules and regulations or a set guidelines to provide mapping between Object Oriented Programming elements like classes, id properties, normal properties and the relational data model elements like table, primary key columns, and normal columns either by using XML file or by using Annotations.

Examples:

1. EJB'S
2. HIBERNATE
3. JPA
4. TOPLINK
5. OPEN JPE

2. EJB Vs HIBERNATE:**WHAT IS THE DIFFERENCE B/W EJB&HIBERNATE?**

- i. In case of EJB'S entity beans, we should require 1-1 relation b/w database table and the entity classes. But it is not mandatory in case of hibernate.in case of hibernate, a single entity class may represent multiple no.of tables and single table may represent multiple no.of entity classes.
- ii. EJB'S entity beans is more API dependent because in EJB'S entity beans all the entity classes must implement or extend predefined library provided by EJB API.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- iii. hibernate is less API dependent because in Hibernate all the entity classes are by default POJO classes(plain old java object),these are normal java bean classes which should not extend (or) implement any predefined library.
- iv. Due to the above reasons, Testing & Debugging is very difficult in entity but which are very simple incase of Hibernate.
- v. In case of entity beans, all the bean classes are not participated in inheritance directly. But in case of hibernate, it is possible to provide inheritance relation b/w POJO classes.
- vi. EJB'S entity beans should require application server to execute. But hibernate applications will be executed with or without application server.
- vii. Due to the above reasons, entity beans portability is very less. Due to the above reason, hibernate portability is very good.
- viii. Due to the above reason entity bean is heavy weight data persistency solution. Due to the above reason, hibernate is light weight data persistency solution.
- ix. Entity beans are relatively slow and hibernate is faster data persistency solution except startup.

3. JPA Vs HIBERNATE:

What is the difference between JPA and Hibernate?

- JPA is an abstraction provided by SUN Microsystems and implemented by all Application Server vendors like Web logic Server vendor, JBOSS Server vendor,..... JPA provides a set of conventions to implement ORM rules and regulations.
- Hibernate is a product, it has implemented ORM rules and regulations as per JPA guidelines in order to provide data Persistency in enterprise applications.

Hibernate History:

Home	: Read Hat
Objective	: To simplify data persistency in Enterprise
Author	: Gavin King
Type	: ORM Implementation Product.
Open/Licenced	: Open Source Software.
Initial Version	: Hibernate1.0 [2001]
User versions	: Hibernate3.x [2005] Hibernate4.x [2011]
Latest Version	: Hibernate5.x [2017]
Designed On	: JAVA
Website	: www.hibernate.org

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

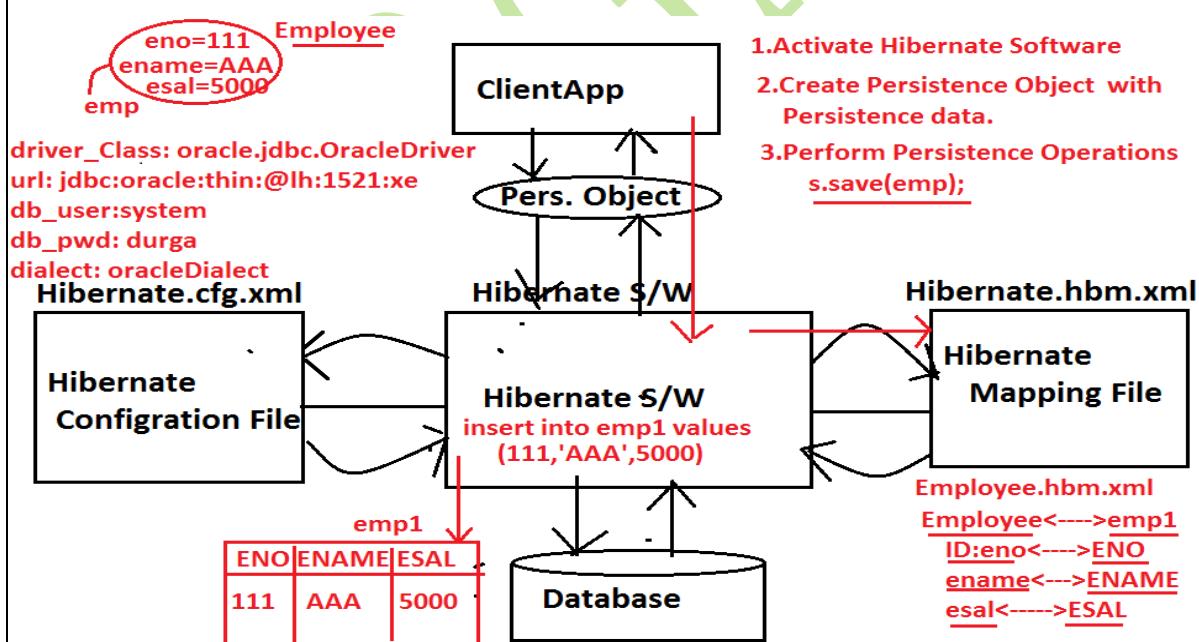


BY NAGOOR BABU

Hibernate Features:

1. Hibernate is Database independent, it can be used for any type of Database.
2. Hibernate is applicable for both standalone applications and Enterprise Applications.
3. Hibernate is providing very good support for Associations and Joins.
4. Hibernate is having very good Annotations in order to reduce XML tech dependency in enterprise applications.
5. Hibernate is having very good implementations for Primary key generation algorithms in order to generate and insert a unique primary key value for each and every insertion operation.
6. Hibernate is providing very good Collections support to manage data.
7. Hibernate is having its own query language, which is database independent, Object Oriented, that is, HQL in order to perform database operations.
8. Hibernate has provided very good Cache mechanisms in order to reuse the results.
9. Hibernate is having very good support for Connection Pooling mechanisms in order to improve Connection re usability.
10. Hibernate is supported by almost all IDEs and Application Servers.
11. Hibernate is against for SQL Queries in enterprise applications directly, but, if we want to write database dependent sql queries then it is possible to provide database dependent sql queries by using "Native SQL".
12. Hibernate has provided very good transactions support.

Hibernate Architecture:



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

From the above Arch.,

Hibernate Configuration File will provide all the configuration details like driver class name, driver URL, Database User name , Database password,... which we required to establish connection with database and to setup JDBC environment.

Hibernate Mapping file will provide all the mapping details like Bean Class name and Database table name, ID property and Primary key column , Normal Properties and Normal Columns,....

- 1) Client Application will perform the following three actions in Hibernate applications mainly.
- 2) Activating Hibernate Software, in this case, Hibernate Software will take all configuration details from hibernate configuration file and Hibernate Software will set up the required JDBC Environment to perform database operations.
- 3) Prepare Persistence Object with the persistence data.
- 4) Perform Persistence Operation.

When Client Application perform persistence operation, Hibernate Software will perform the following actions.

- 1) Hibernate Software will take persistence method call and identify persistence Object.
- 2) Hibernate Software will take all mapping details from hibernate mapping file like database table name and all column names on the basis of Persistence object.
- 3) Hibernate Software will prepare database dependent sql query on the basis of table names and column names and with the persistence object provided data.
- 4) Hibernate Software will execute the generated database dependent sql query and perform the required persistence operation.

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28 **US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

Steps to prepare First Hibernate Application

Steps to prepare First Hibernate Application:

- 1) Prepare Persistence Class or Object.
- 2) Prepare Mapping File.
- 3) Prepare Hibernate Configuration File
- 4) Prepare Hibernate Client Application

Note: To run Hibernate Applications we have to download all Hibernate JAR file and we have to add them for our Hibernate application.

1) Prepare Persistence Class or Object:

The main intention of Persistence class or object in Hibernate applications is to manage Persistence data which we want to store in Database or by using this we want to perform the database operations like select, update, delete,

In Hibernate applications, to prepare Persistence classes we have to use the following Guidelines

- 1) In Hibernate applications Persistence classes must be POJO classes [Plain Old Java Object], they must not extend or implement predefined Library.
- 2) In hibernate Applications Persistence classes must be public, Non abstract and non-final.
 - ⊕ Where the main intention to declare persistence classes as public is to bring persistence classes scope to Hibernate software in order to create objects.
 - ⊕ Where the main intention to declare persistence classes as Non abstract is to allow to create Objects for Persistence classes
 - ⊕ Where the main intention to declare persistence classes as Non final is to allow to extend one persistence class to another persistence class as per the requirement.
- 3) In Persistence classes all Properties must be declared as per database table provided columns, where names are not required to be matched, but, data types must be compatible.
- 4) In Persistence classes, all properties must be declared as private in order to improve Encapsulation.
- 5) In Persistence classes , we must define a separate set of setXXX () and getXXX () methods for each and every property
- 6) In persistence classes, we must declare all methods are public.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- 7) In Persistence classes, if we want to provide any constructor then it must be public and 0-arg constructor, because, while creating object for persistence class Hibernate software will search and execute only public and 0-arg constructor.
- 8) In Hibernate applications , if we want to provide our own comparison mechanisms while comparing Persistence objects then it is suggestible to Override equals(--) method.
- 9) In Hibernate applications, if we want to provide our own hash code values then it is suggestible to Overrid hashCode () method.
- 10)In Hibernate applications, we will use POJO classes, which are not extending and implementing predefined library, but, it is suggestible to implement java.io.Serializable marker interface in order to make eligible Persistence object for Serialization and Deserialization.
- 11)In Persistence classes, we have to declare a property as an ID property, it must represent primary key column in the respective table.

EX: Employee.java

```
1. public class Employee{  
2.     private int eno;  
3.     private String ename;  
4.     private float esal;  
5.     private String eaddr;  
6.  
7.     public void setEno(int eno){  
8.         this.eno = eno;  
9.     }  
10.    public void setEname(String ename){  
11.        this.ename = ename;  
12.    }  
13.    public void setEsal(float esal){  
14.        this.esal = esal;  
15.    }  
16.    public void setEaddr(String eaddr){  
17.        this.eaddr = eaddr;  
18.    }  
19.    public int getEno(){  
20.        return eno;  
21.    }  
22.    public String getEname(){  
23.        return ename;  
24.    }  
25.    public float getEsal(){  
26.        return esal;  
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
27.}  
28. public String getEaddr(){  
29. return eaddr;  
30.}  
31.}
```

2. Prepare Mapping File:

The main intention of mapping file in Hibernate applications is to provide mapping between a class ,id property and normal properties from Object Oriented Data Model and a table, primary key column and normal columns from Relational data model.

In Hibernate applications, mapping file is able to provide the mapping details like Basic OR mapping, Component mapping, inheritance mapping, Collections mapping, Associations mapping,

To prepare mapping file in Hibernate applications we have to provide mapping file name with the following format.

POJO_Class_Name.hbm.xml

The above format is not mandatory, we can use any name but we must provide that intemation to the hibernate software.

In Hibernate applications, we can provide any no of POJO classes, w.r.t each and every POJO class we can define a separate mapping file.

Note: In Hibernate applications, it is possible to configure more than POJO class in single mapping file.

Upto Hibernate3.2.5 version Hibernate Mapping file is mandatory to provide mapping details, but, right from Hibernate3.2.5 version mapping file is optional, because, Hibernate 3.2.5 version has provided annotations as an alternative to mapping file.

To provide Basic mapping between POJO class and table in mapping file we have to use the following XML tags provided by Hibernate.

1. <!DOCTYPE.....>
2. <hibernate-mapping>
3. <class name="--" table="--">
4. <id name="--" column="--"/>
5. <property name="--" column="--"/>

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
6. ----  
7. <class>  
8. </hibernate-mapping>
```

- ✓ Where <hibernate-mapping> tag is root tag in mapping file, it will include no of classes configuration.
- ✓ Where <class> tag is able to provide single POJO class configuration inorder to provide mapping between POJO class name and the respective table name.
- ✓ Where "name" attribute in <class> tag is able to provide fully qualified name of the POJO class.
- ✓ Where "table" attribute in <class> tag is able to provide the respective table name.
- ✓ Where <id> tag is able to provide ID property configuration inorder to provide mapping between ID property and the respective primary key column.
- ✓ Where "name" attribute in <id> tag is able to provide id property name of the POJO class.
- ✓ Where "column" attribute in <id> tag is able to provide primiry key column defined in table.
- ✓ Where <property> tag is able to provide normal bean property configuration inorder to provide mapping between normal bean property and normal table column.
- ✓ Where "name" attribute and "column" attribute will take bean property name and table column name respectively.

Note: If bean property names and table column names are same then it is optional to provide "column" attribute in mapping file.

EX:

```
1. <!DOCTYPE.....>  
2. <hibernate-mapping>  
3.   <class name="com.durgasoft.Employee" table="emp1">  
4.     <id name="eno" column="eno"/>  
5.     <property name="ename"/>  
6.     <property name="esal" column="emp_Sal"/>  
7.     <property name="eaddr" column="eaddr"/>  
8.   </class>  
9. </hibernate-mapping>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3. Prepare Hibernate Configuration File:

The main purpose of hibernate configuration file is to provide all configuration details of hibernate application which includes Jdbc parameters to prepare connection , Transactions configurations, Cache mechanisms configurations, Connection pooling configurations,.....

In Hibernate applications, the standard name of hibernate configuration file is "hibernate.cfg.xml", it is not fixed, we can provide any name but that name must be given to Hibernate software.

In Hibernate applications, we are able to provide more than one configuration file , but, for each and every database, that is, in hibernate applications if we use multiple databases then we are able to prepare multiple configuration files.

To prepare hibernate configuration file with basic configuration details we have to use the following XML tags.

```
1. <!DOCTYPE....>
2. <hibernate-configuration>
3. <session-factory>
4.   <property name="--" value </property>
5.   -----
6.   -----
7.   <mapping resource="--"/>
8.   -----
9. </session-factory>
10.</hibernate-configuration>
```

- ✓ Where <hibernate-configuration> is root tag.
- ✓ Where <session-factory> tag is able to include hibernate properties configurations.
- ✓ Where <property> tag is able to provide single property configuration.
- ✓ Where "name" attribute in <property> tag is able to provide property name and body of the <property> tag will take property value.
- ✓ Where <mapping> tag is able to provide mapping file configuration.
- ✓ Where "resource" attribute is able to provide mapping file name and location.

EX:**hibernate.cfg.xml:**

```
1. <!DOCTYPE....>
2. <hibernate-configuration>
```

CONTACT US:**Mobile: +91- 8885 25 26 27** +91- 7207 21 24 27/28**US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```
3. <session-factory>
4.   <property name="connection.driver_Class"> oracle.jdbc.OracleDriver
5.   </property>
6.   <property name="connection.url">
7.     jdbc:oracle:thin:@localhost:1521:xe
8.   </property>
9.   <property name="connection.user">system</property>
10.  <property name="connection.password">durga</property>
11.  <property name="hibernate.dialect"> org.hibernate.dialect.OracleDialect
12.  </property>
13.  <mapping resource="Employee.hbm.xml"/>
14. </session-factory>
15. </hibernate-configuration>
```

In Hibernate configurations, "dialect" is a property, it able to provide dialect class name, it is providing the respective database native implementations which we are using in hibernate applications in order to prepare database dependent sql queries by Hibernate software.

Hibernate has provided separate predefined class in the form of org.hibernate.dialect.XXXDialect for each and every database.

EX: org.hibernate.dialect.Oracle9Dialect
org.hibernate.dialect.Oracle10gDialect
org.hibernate.dialect.MySQLDialect

4. Prepare Client Application:

The main intention of Hibernate Client application is to activate Hibernate Software, creating persistence objects and performing Persistence operations.

To prepare Client Application in hibernate applications we have to use the following steps as per Hiberante3.x version.

1. Create Configuration class object
2. Create Session Factory object
3. Create Session Object
4. Create Transaction object if it is required.
5. Perform Persistence operations
6. Close Session Factory and Session objects.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. Create Configuration class object

In Hibernate, the main intention of Configuration object is to store all the configuration details which we provided in hibernate configuration file.

To represent Configuration object Hibernate has provided a predefined class in the form of "org.hibernate.cfg.Configuration".

To create Configuration class object we have to use the following constructor from Configuration class.

```
public Configuration()
```

EX: Configuration cfg = new Configuration();

If we use the above instruction in Hibernate applications then we are able to get an empty Configuration object in heap memory, it will not include any Configuration details.

If we want to store Configuration details from Configuration file we have to use either of the following methods.

1. public Configuration configure()

This method will get configuration details from the configuration file with the name hibernate.cfg.xml

2. public Configuration configure(String config_file_Name)

This method can be used to get configuration details from hibernate configuration file with any name, it will be used when we change configuration file name from hibernate.cfg.xml file to some other name .

3. public Configuration configure(File file)

This method can be used to get configuration details from a file which is represented in the form of java.io.File class object.

```
public Configuration configure(URL url)
```

This method can be used to get Configuration details from a file which is available in network represented in the form of java.net.URL .

EX: Configuration cfg = new Configuration();
cfg.configure();

When we use configure() method then Hibernate Software will search for hibernate.cfg.xml file, if it is available then Hibernate software will load the content of hibernate.cfg.xml file, parse it and read content from configuration file to Configuration object.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Create Session Factory object:

In Hibernate, the main intention of Session Factory object is to manage Connections, Statements, Cache levels, and it able to provide no of Hibernate Session objects.

To represent Session Factory object Hibernate has provided a predefined interface in the form of "org.hibernate.Session Factory".

To get Session Factory object we have to use the following method from Configuration class.

```
public Session Factory buildSessionFactory()
```

EX: SessionFactory sf = cfg.buildSessionFactory();

Note: The above approach to get Session Factory object is available upto Hibernate3.x version, buildSessionFactory() was deprecated in Hibernate4.x version.

In Hibernate applications, if we use multiple Databases then we have to prepare multiple Configuration files, multiple Configuration Object, w.r.t this, we have to prepare multiple Session Factory objects.

Session Factory object is heavy weight and it is thread safe upto a particular Database, because, it able to allow more than one thread at a time.

3. Create Session Object:

In Hibernate, for each and every database interaction a separate Session will be created.

In Hibernate, Session is able to provide no of persistence methods in order to perform persistence operations.

To represent Session object, Hibernate has provided a predefined interface in the form of "org.hibernate.Session".

To get Session object, we have to use the following method from Session Factory.

```
public Session openSession()  
EX: Session s =sf.openSession();
```

In Hibernate, Session object is light weight and it is not thread safe, because, for each and every thread a separate Session object will be created.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

4. Create Transaction Object:

Transaction is a unit of work performed by Front End applications on Back end systems.

To represent Transactions, Hibernate has provided a predefined interface in the form of "org.hibernate.Transaction".

To get Transaction object we will use either of the following methods.

1. public Transaction getTransaction()

It will return Transaction object with out begin, where to begin Transaction we have to use the following method.

```
public void begin()
```

2. public Transaction beginTransaction()

It will return Transaction and begin Transaction.

In Hibernate applications, after performing persistence operations we must perform either commit or rollback operations inorder to complete Transactions, for this, we have to use the following methods from Transaction.

```
public void commit()  
public void rollback()
```

Note: In Hibernate applications, Transaction is required for only Non Select operations, not required for Select operations.

5. Perform Persistence Operations:

In Hibernate applications, to perform persistence operations Session has provided the following methods.

To insert an object or record into Database table we have to use the following methods.

1. save(--)
2. persist(--)

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

What is the difference between save() method and persist() method?**Ans:**

In Hibernate applications, save() method can be used to insert a record into the Database table and it will return Primary Key value of the inserted record.

```
public Serializable save(Object obj) throws HibernateException
```

In Hibernate applications, persist() method can be used to insert a record into database table and it will not return any value.

```
public void persist(Object obj) throws HibernateException
```

To update a record in database table we have to use the following methods.

1. update(--)
2. saveOrUpdate(--)

What is the difference between update() method and saveOrUpdate() method?**Ans:**

Where update() method will perform updation on a record in database table if the specified record is existed otherwise it will rise an Exception.

```
public void update(Object obj) throws HibernateException
```

Where saveOrUpdate() method will insert the specified record in database table if the specified record is not existed . If the specified record is existed in database table then it will update the record.

```
public void saveOrUpdate(Object obj) throws HibernateException
```

To delete a record from database table we have to use the following method.

```
public void delete(Object obj) throws HibernateException
```

To retrieve a record from database table we have to use the following methods.

- 1.get(--)
- 2.load(--)

CONTACT US:**Mobile: +91- 8885 25 26 27****+91- 7207 21 24 27/28****US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

What are the differences between get(-) method and load(-) method?**Ans:**

1. get() method can be used to retrieve a record from database table if the record is existed. If the required record is not existed then get() method will return null value.

```
public Object get(String class_Name, Serializable pk_Val)  
public Object get(Class class_Type, Serializable pk_Val)
```

- load() method can be used to retrive a record from database table if the record is existed. If the required record is not existed then load() method will rise an Exception like HibernateException.

```
public Object load(String class_Name, Serializable pk_Val)  
public Object load(Class class_Type, Serializable pk_val)
```

2. get() method is able to perform eager or early loading, that is, it will interact with database directly and it will retrive data and return to Hibernate application in the form of Object on the method call.

load() method will perform Lazy or late Loading , that is, when we access load() method then a duplicate object will be created with the primary key value without interacting with database. When we use other properties of the Object then only it will fetch data from database table and return that data to Java application.

6. Close Session and SessionFactory :

In Hibernate applications, it is convention to close Session and Sessionfactory objects at the end of client application in order to avoid security problems.

```
public void close()throws HibernateException.
```

EX: s.close();
sf.close();

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Hibernate Applications with Eclipse

Eclipse:

1. Home: IBM canada [Initially]
2. Eclipse Foundations [2003]
3. website: www.eclipse.org
4. Objective: To simplify JAVA/J2EE application Development.
5. Initial Version: Ecplise3.0[2004]
6. Used Versions:Eclipse3.5,3.6,3.7[Helios,Gelilio,Indigo]
7. Latest Version: Ecplise4.2[1st march,2013][JUNO]
 - i. Eclipse4.3.1[26 june,2013][Kepler]
 - ii. Eclipse4.4[9th Aug,2013][Luna]
 - iii. Eclipse4.5[Mars]
 - iv. Eclipse4.6[Neon]
 - v. Eclipse4.7[Oxygen]
8. Designed on: Java platform.
9. Type: Open Source Software.

Installation Process:

1. download [eclipse-jee-oxygen-1a-win32.zip](#)
2. extract [eclipse-jee-oxygen-1a-win32.zip](#) file at the required location.
3. After step 2 , we will find "eclipse" folder ie installation folder.
4. Double click on eclipse icon available at c:\eclipse\ location.
5. After step4, we will get a window to select workspace, where specify the location where we want to manage all the applications.

Steps To Design Hibernate Applications in Eclipse IDE

1. Prepare Java Project.
2. Prepare Hibernate Library and Add that library to Appl.
3. Prepare PJO class.
4. Prepare Mapping File.
5. Prepare Configuration File
6. Prepare Client Application.
7. Run Client Application.

1. Prepare Java Project

- I. Right Click on "Project Exproer".
- II. Select "New"
- III. select "Java Project"

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- IV. Provide Project Name : app1
- V. Click on "Next" button.
- VI. Click on "Finish" button.

2. Add all hibernate Library to Java Application

- 1) Right Click on "Project".
- 2) Select "Properties".
- 3) Select "java build path".
- 4) Select "Library".
- 5) Select "Add Library".
- 6) Select "User Library".
- 7) Click on "Next" button.
- 8) Click on "User Libraries".
- 9) Click on "New" button.
- 10) Provide User Library Name : Hibernate3_Lib".
- 11) Click on "OK" button.
- 12) Select User Library "Hibernate3_Lib".
- 13) Click on "Add External Jars".
- 14) Select the following JAR files mandatorily.
 - a) ojdbc6.jar
 - b) hibernate3.jar
 - c) jta-1.1.jar
 - d) commons-collections-3.1.jar
 - e) dom4j-1.6.1.jar
 - f) javassist-3.12.0.GA.jar
 - g) antlr-2.7.6.jar
 - h) slf4j-api-1.6.1.jar
 - i) hibernate-jpa-2.0-api-1.0.1.Final.jar
- 15) Click on "OK" button
- 16) Click on "Finish" button.
- 17) Click on "Apply" and "OK" button.

3. Prepare POJO class

- 1) Right Click on "Src" folder.
- 2) Select "New".
- 3) Select "Class".
- 4) Provide package Name:com.durgasoft.hbn.pojo
- 5) Provide class Name: Employee
- 6) Click on "Finish" button.
- 7) Declare Properties in the generated class.
- 8) Generate SetXXX() and getXXX() methods.

CONTACT US:

Mobile: +91- 8885 25 26 27 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- a) Right Click
- b) Select "Source".
- c) Select "Generate Getters and Setters".
- d) Select "Select All".
- e) Click on "OK" button.

```

1. package com.durgasoft;
2. public class Employee{
3.     private int eno;
4.     private String ename;
5.     private float esal;
6.     setXXX() and getXXX()
7. }
```

4. Prepare Mapping File

- 1) Right Click on "Src" folder.
- 2) Select "New"
- 3) Select "Package".
- 4) Provide Package Name "com.durgasoft.hbn.mappings"
- 5) Click on "Finish".
 - a) Right click on "com.durgasoft.hbn.mappings" package.
 - b) Select "New"
 - c) Select "other"
 - d) Select "XML"
 - e) Select "xml file"
 - f) Click on "Next" button.
 - g) Provide File Name: Employee.hbm.xml
 - h) Click on "Next"
 - i) Click on "Next"
 - j) Click on "Finish" button.
 - k) Provide <!DOCTYPE..> tag from hibernate3.jar file under the package org.hibernate from hibernate-mapping-3.0.dtd and past in the mapping file.

EX:

```

1. <!DOCTYPE hibernate-mapping PUBLIC
2.      "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
3.      "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
4. <hibernate-mapping>
5. <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

6. <id name="eno"/>
7. <property name="ename"/>
8. <property name="esal"/>
9. </class>
10. </hibernate-mapping>
```

5. Prepare Configuration File:

- 1) Right Click on "Src" folder.
- 2) Select "New"
- 3) Select "Package".
- 4) Provide Package Name "com.durgasoft.hbn.cfgs"
- 5) Click on "Finish".
 - a) Right click on "com.durgasoft.hbn.cfgs" package.
 - b) Select "New"
 - c) Select "other"
 - d) Select "XML"
 - e) Select "xml file"
 - f) Click on "Next" button.
 - g) Provide File Name: hibernate.cfg.xml
 - h) Click on "Next"
 - i) Click on "Next"
 - j) Click on "Finish" button.
 - k) provide <!DOCTYPE...> tag from hibernate3.jar file from org.hibernate package from hibernate-configuration-3.0.dtd.

EX:

```

1. <!DOCTYPE hibernate-configuration PUBLIC
2.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
3.   "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
4. <hibernate-configuration>
5. <session-factory>
6. <property name="connection.driver_class">
7. oracle.jdbc.OracleDriver
8. </property>
9. <property name="connection.url">
10. jdbc:oracle:thin:@localhost:1521:xe
11. </property>
12. <property name="connection.user">system</property>
13. <property name="connection.password">durga</property>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

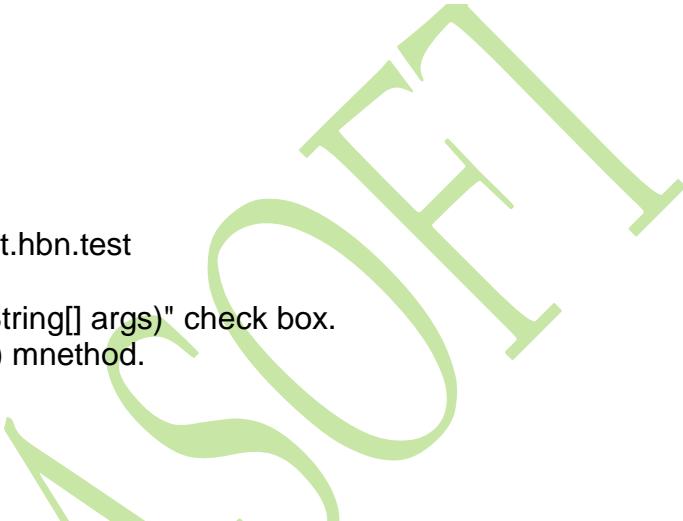
```

14. <property name="hibernate.dialect">
15. org.hibernate.dialect.OracleDialect
16. </property>
17. <mapping resources="com/durgasoft/hbn/mappings/Employee.hbm.xml"/>
18. </session-factory>
19. </hibernate-configuration>
```

6. Prepare Client App

- 1) Right click on src
- 2) Select "New" button.
- 3) Select "Class".
- 4) Provide package Name: com.durgasoft.hbn.test
- 5) Provide Class Name : ClientApp
- 6) Select "select public static void main(String[] args)" check box.
- 7) Provide Application logic inside main() method.

EX:



```

1. package com.durgasoft.hbn.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.cfg.Configuration;
7.
8. import com.durgasoft.hbn.pojo.Employee;
9.
10. public class ClientApp {
11.
12.     public static void main(String[] args) throws Exception {
13.         Configuration cfg = new Configuration();
14.         cfg.configure("/com/durgasoft/hbn/cfgs/hibernate.cfg.xml");
15.         SessionFactory session_Factory = cfg.buildSessionFactory();
16.         Session session = session_Factory.openSession();
17.         Transaction tx = session.beginTransaction();
18.         Employee emp = new Employee();
19.         emp.setEno(111);
20.         emp.setEname("AAA");
21.         emp.setEsal(5000);
22.         emp.setEaddr("Hyd");
23.         session.save(emp);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
24.     tx.commit();
25.     System.out.println("Employee Record inserted Succesfully");
26.     session.close();
27.     session_Factory.close();
28.   }
29. }
```

7. Run the application

1. Right click on ClientApp.java
2. Select "Run As".
3. Select "Java Application".

Note: Download JAR files from

<https://sourceforge.net/projects/hibernate/files/hibernate3>

DURGASOFT

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Hibernate Applications

Example:

Employee.java

```

1. package com.durgasoft.hbn.pojo;
2. public class Employee {
3.     private int eno;
4.     private String ename;
5.     private float esal;
6.     private String eaddr;
7.     setXXX() and getXXX()
8. }
```



Employee.hbm.xml

```

1. <!DOCTYPE .... >
2. <hibernate-mapping>
3. <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
4.   <id name="eno" column="eno"/>
5.   <property name="ename"/>
6.   <property name="esal"/>
7.   <property name="eaddr"/>
8. </class>
9. </hibernate-mapping>
```



hibernate.cfg.xml

```

1. <!DOCTYPE...>
2. <hibernate-configuration>
3.   <session-factory>
4.     <property name="connection.driver_Class"> oracle.jdbc.OracleDriver</proper
rty>
5.     <property name="connection.url"> jdbc:oracle:thin:@localhost:1521:xe</proper
rty>
6.     <property name="connection.username">system</property>
7.     <property name="connection.password">durga</property>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
8. <property hibernate.dialect"> org.hibernate.dialect.OracleDialect</property>
9. <mapping resource="com/durgasoft/hbn/mappings/Employee.hbm.xml"/>
10. </session-factory>
11.</hibernate-configuration>
```

ClientApp.java

```
1. package com.durgasoft.hbn.test;
2. import org.hibernate.*;
3. import org.hibernate.cfg.Configuration;
4. import com.durgasoft.hbn.pojo.Employee;
5. public class ClientApp {
6. public static void main(String[] args) throws Exception {
7. Configuration cfg = new Configuration();
8. cfg.configure("/com/durgasoft/hbn/cfgs/hibernate.cfg.xml");
9. SessionFactory session_Factory = cfg.buildSessionFactory();
10.Session session = session_Factory.openSession();
11.Transaction tx = session.beginTransaction();
12.Employee emp = new Employee();
13.emp.setEno(111);
14.emp.setEname("AAA");
15.emp.setEsal(5000);
16.emp.setEaddr("Hyd");
17.session.save(emp);
18.tx.commit();
19.System.out.println("Employee Record inserted Succesfully");
20.session.close();
21.session_Factory.close();
22.}
23.}
```

In Hibernate Applications, when we access persistence methods , Hibernate Software will take that persistence method call and it will prepare database dependent native sql query on the basis of hibernation mapping details and configuration details . In this context, to display the generated database dependent sql query on console we have to use "show_sql" property with the value "true" in hibernate configuration file.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example On Hibernate SaveOrUpdate(--) method:

Employee.java

```
1. package com.durgasoft.hbn.pojo;
2.
3. public class Employee {
4.     private int eno;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.
9.     public int getEno() {
10.         return eno;
11.     }
12.     public void setEno(int eno) {
13.         this.eno = eno;
14.     }
15.     public String getEname() {
16.         return ename;
17.     }
18.     public void setEname(String ename) {
19.         this.ename = ename;
20.     }
21.     public float getEsal() {
22.         return esal;
23.     }
24.     public void setEsal(float esal) {
25.         this.esal = esal;
26.     }
27.     public String getEaddr() {
28.         return eaddr;
29.     }
30.     public void setEaddr(String eaddr) {
31.         this.eaddr = eaddr;
32.     }
33.
34. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
7.     <id name="eno"/>
8.     <property name="ename"/>
9.     <property name="esal"/>
10.    <property name="eaddr"/>
11.   </class>
12. </hibernate-mapping>
```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="show_sql">true</property>
12.    <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
13.    <mapping resource="com/durgasoft/hbn/mappings/Employee.hbm.xml"/>
14.  </session-factory>
15. </hibernate-configuration>
```

ClientApp.java

```

1. package com.durgasoft.hbn.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. import org.hibernate.Transaction;
6. import org.hibernate.cfg.Configuration;
7.
8. import com.durgasoft.hbn.pojo.Employee;
9.
10. public class ClientApp {
11.
12.     public static void main(String[] args) {
13.         SessionFactory session_Factory = null;
14.         Session session = null;
15.         try {
16.             Configuration cfg = new Configuration();
17.             cfg.configure("/com/durgasoft/hbn/cfgs/hibernate.cfg.xml");
18.             session_Factory = cfg.buildSessionFactory();
19.             session = session_Factory.openSession();
20.             Transaction tx = session.beginTransaction();
21.             tx.begin();
22.             Employee emp = new Employee();
23.             emp.setEno(111);
24.             emp.setEname("XXX");
25.             emp.setEsal(9000);
26.             emp.setEaddr("Sec");
27.             session.saveOrUpdate(emp);
28.             tx.commit();
29.             System.out.println("Employee Updated Successfully");
30.         } catch (Exception e) {
31.             e.printStackTrace();
32.         } finally {
33.             session.close();
34.             session_Factory.close();
35.         }
36.
37.     }
38. }
```

Example to delete records from DB:

Employee.java

```
1. package com.durgasoft.hbn.pojo;
2.
3. public class Employee {
4.     private int eno;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

5.  private String ename;
6.  private float esal;
7.  private String eaddr;
8.
9.  public int getEno() {
10.    return eno;
11. }
12. public void setEno(int eno) {
13.   this.eno = eno;
14. }
15. public String getEname() {
16.   return ename;
17. }
18. public void setEname(String ename) {
19.   this.ename = ename;
20. }
21. public float getEsal() {
22.   return esal;
23. }
24. public void setEsal(float esal) {
25.   this.esal = esal;
26. }
27. public String getEaddr() {
28.   return eaddr;
29. }
30. public void setEaddr(String eaddr) {
31.   this.eaddr = eaddr;
32. }
33.
34.}
```

Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
7.     <id name="eno"/>
8.     <property name="ename"/>
9.     <property name="esal"/>
10.    <property name="eaddr"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

11. </class>
 12. </hibernate-mapping>

hibernate.cfg.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7.   <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.   <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.   <property name="connection.user">system</property>
10.  <property name="connection.password">durga</property>
11.  <property name="show_sql">true</property>
12.  <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
13.  <mapping resource="com/durgasoft/hbn/mappings/Employee.hbm.xml"/>
14. </session-factory>
15. </hibernate-configuration>
```

ClientApp.java

```
1. package com.durgasoft.hbn.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.cfg.Configuration;
7.
8. import com.durgasoft.hbn.pojo.Employee;
9.
10. public class ClientApp {
11.
12.     public static void main(String[] args) {
13.         SessionFactory session_Factory = null;
14.         Session session = null;
15.         try {
16.             Configuration cfg = new Configuration();
17.             cfg.configure("/com/durgasoft/hbn/cfgs/hibernate.cfg.xml");
18.             session_Factory = cfg.buildSessionFactory();
19.             session = session_Factory.openSession();
20.             Transaction tx = session.beginTransaction();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

21.     Employee emp = new Employee();
22.     emp.setEno(111);
23.     session.delete(emp);
24.     tx.commit();
25.     System.out.println("Employee Deleted Successfully");
26. } catch (Exception e) {
27.     e.printStackTrace();
28. } finally {
29.     session.close();
30.     session_Factory.close();
31. }
32.
33. }
34.
35.}
```

Example To retrieve Record from DB:

Employee.java

```

1. package com.durgasoft.hbn.pojo;
2.
3. public class Employee {
4.     private int eno;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.
9.     public int getEno() {
10.         return eno;
11.     }
12.     public void setEno(int eno) {
13.         this.eno = eno;
14.     }
15.     public String getEname() {
16.         return ename;
17.     }
18.     public void setEname(String ename) {
19.         this.ename = ename;
20.     }
21.     public float getEsal() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

22.     return esal;
23. }
24. public void setEsal(float esal) {
25.     this.esal = esal;
26. }
27. public String getEaddr() {
28.     return eaddr;
29. }
30. public void setEaddr(String eaddr) {
31.     this.eaddr = eaddr;
32. }
33.
34.

```

Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
7.     <id name="eno"/>
8.     <property name="ename"/>
9.     <property name="esal"/>
10.    <property name="eaddr"/>
11.  </class>
12.</hibernate-mapping>

```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

9. <property name="connection.user">system</property>
10. <property name="connection.password">durga</property>
11. <property name="show_sql">true</property>
12. <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
13. <mapping resource="com/durgasoft/hbn/mappings/Employee.hbm.xml"/>
14. </session-factory>
15. </hibernate-configuration>
```



ClientApp.java

```

1. package com.durgasoft.hbn.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.cfg.Configuration;
6.
7. import com.durgasoft.hbn.pojo.Employee;
8.
9. public class ClientApp {
10.
11.     public static void main(String[] args) {
12.         SessionFactory session_Factory = null;
13.         Session session = null;
14.
15.         try {
16.             Configuration cfg = new Configuration();
17.             cfg.configure("/com/durgasoft/hbn/cfgs/hibernate.cfg.xml");
18.             session_Factory = cfg.buildSessionFactory();
19.             session = session_Factory.openSession();
20.             Employee emp = (Employee)session.get("com.durgasoft.hbn.pojo.Employee",222 );
21.
22.             if(emp == null) {
23.                 System.out.println("Employee Not Existed");
24.             }else {
25.                 System.out.println("Employee Details");
26.                 System.out.println("-----");
27.                 System.out.println("Employee Number :" +emp.getEno());
28.                 System.out.println("Employee Name :" +emp.getEname());
29.                 System.out.println("Employee Salary :" +emp.getEsal());
30.                 System.out.println("Employee Address :" +emp.getEaddr());
31.             }
32.         }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

31.     } catch (Exception e) {
32.         e.printStackTrace();
33.     }finally {
34.         session_Factory.close();
35.         session.close();
36.     }
37.
38. }
39.
40.}
```

HIBERNATE with My SQL Database :

If we want to use MySQL Database for Hibernate applications then we have provide the following changes.

In Hibernate Configuration File:

Driver Class Name	: com.mysql.jdbc.Driver
Driver URL	: jdbc:mysql://localhost:3306/db_Name
DB User Name	: root
DB Password	: root
Dialect	: org.hibernate.dialect.MySQLDialect

Note: We must add mysql-connector-java-5.0.8-bin.jar file to Hibernate3_Lib.

In Hibernate Applications, it is not suggestible to manage Configuration object and SessionFactory object in Client Application directly, because, these two objects are providing Hibernate Boot strapping. In future if we want to change boot strapping mechanisms then we have to provide changes in Client Application, it may effect our persistence operations. To avoid this type of problems we have to use a separate Utility class to prepare Configuration and SessionFactory object.

EX:

```

1. class HibernateUtil{
2. private static SessionFactory sessionfactory;
3. static{
4. try{
5.     Configuration cfg = new Configuration();
6.     cfg.configure(---);
7.     sessionfactory = cfg.buildSessionFactory();
8. }catch(Exception e){
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9.     e.printStackTrace();
10. }
11.}
12. public static SessionFactory getSessionFactory(){
13.     return sessionFactory;
14.}
15. public static void cleanUp(SessionFactory sf, Session s){
16.     s.close();
17.     sf.close();
18.}
19.}
```

Example:**Employee.java**

```
1. package com.durgasoft.hbn.pojo;
2.
3. public class Employee {
4.     private int eno;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.
9.     public int getEno() {
10.         return eno;
11.     }
12.     public void setEno(int eno) {
13.         this.eno = eno;
14.     }
15.     public String getEname() {
16.         return ename;
17.     }
18.     public void setEname(String ename) {
19.         this.ename = ename;
20.     }
21.     public float getEsal() {
22.         return esal;
23.     }
24.     public void setEsal(float esal) {
25.         this.esal = esal;
26.     }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

27. public String getEaddr() {
28.     return eaddr;
29. }
30. public void setEaddr(String eaddr) {
31.     this.eaddr = eaddr;
32. }
33.

```



Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
7.     <id name="eno"/>
8.     <property name="ename"/>
9.     <property name="esal"/>
10.    <property name="eaddr"/>
11.  </class>
12.</hibernate-mapping>

```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">com.mysql.jdbc.Driver</property>
8.     <property name="connection.url">jdbc:mysql://localhost:3306/durgadb</property>
9.     <property name="connection.user">root</property>
10.    <property name="connection.password">root</property>
11.    <property name="show_sql">true</property>
12.    <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
13.    <mapping resource="com/durgasoft/hbn/mappings/Employee.hbm.xml"/>
14.  </session-factory>
15. </hibernate-configuration>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

HibernateUtil.java

```
1. package com.durgasoft.hbn.util;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.cfg.Configuration;
6.
7. public class HibernateUtil {
8.     private static SessionFactory sessionFactory;
9.     static {
10.         try {
11.             Configuration cfg = new Configuration();
12.             cfg.configure("/com/durgasoft/hbn/cfgs/hibernate.cfg.xml");
13.             sessionFactory = cfg.buildSessionFactory();
14.         } catch (Exception e) {
15.             e.printStackTrace();
16.         }
17.     }
18.
19.     public static SessionFactory getSessionFactory() {
20.         return sessionFactory;
21.     }
22.     public static void cleanUp(SessionFactory sessionFactory, Session session) {
23.         session.close();
24.         sessionFactory.close();
25.     }
26. }
```

ClientApp.java

```
1. package com.durgasoft.hbn.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5.
6. import com.durgasoft.hbn.pojo.Employee;
7. import com.durgasoft.hbn.util.HibernateUtil;
8.
9. public class ClientApp {
10.
11.     public static void main(String[] args) {
12.         try {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

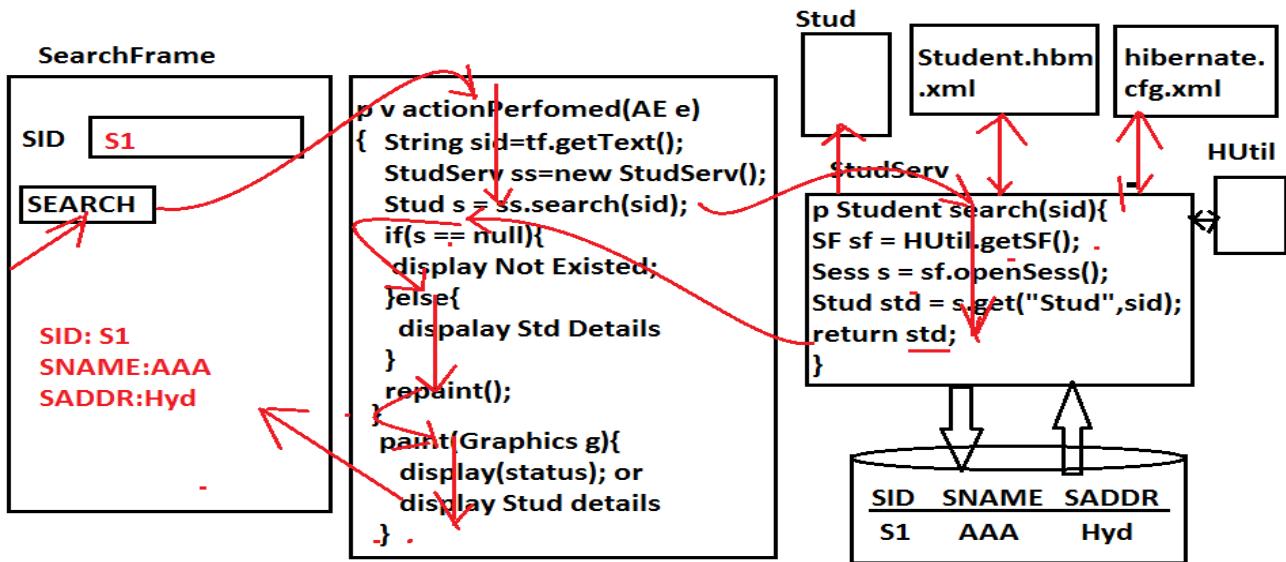
```

13. SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
14. Session session = sessionFactory.openSession();
15. Employee emp = (Employee)session.get("com.durgasoft.hbn.pojo.Employee", 111);

16. if(emp == null) {
17.     System.out.println("Employee Not Existed");
18. }else {
19.     System.out.println("Employee Details");
20.     System.out.println("-----");
21.     System.out.println("Employee Number :" +emp.getEno());
22.     System.out.println("Employee Name :" +emp.getEname());
23.     System.out.println("Employee Salary :" +emp.getEsal());
24.     System.out.println("Employee Address :" +emp.getEaddr());
25. }
26. HibernateUtil.cleanUp(sessionFactory, session);
27. } catch (Exception e) {
28.     e.printStackTrace();
29. }
30.
31. }
32.
33.

```

AWT/GUI-Hibernate Integration Application:



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Student.java

```
1. package com.durgasoft.hbn.pojo;
2.
3. public class Student {
4.     private String sid;
5.     private String sname;
6.     private String saddr;
7.
8.     public String getSid() {
9.         return sid;
10.    }
11.    public void setSid(String sid) {
12.        this.sid = sid;
13.    }
14.    public String getSname() {
15.        return sname;
16.    }
17.    public void setSname(String sname) {
18.        this.sname = sname;
19.    }
20.    public String getSaddr() {
21.        return saddr;
22.    }
23.    public void setSaddr(String saddr) {
24.        this.saddr = saddr;
25.    }
26.
27.
28.}
```

Student.hbm.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.     "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.     "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.     <class name="com.durgasoft.hbn.pojo.Student" table="student">
7.         <id name="sid" column="SID"/>
8.         <property name="sname"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

9.   <property name="saddr"/>
10.  </class>
11.</hibernate-mapping>
```



hibernate.cfg.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE hibernate-configuration PUBLIC
3.      "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.      "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5.  <hibernate-configuration>
6.      <session-factory>
7.          <property name="connection.driver_Class">com.mysql.jdbc.Driver</property>
8.          <property name="connection.url">jdbc:mysql://localhost:3306/durgadb</property>
9.          <property name="connection.user">root</property>
10.         <property name="connection.password">root</property>
11.         <property name="show_sql">true</property>
12.         <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
13.     <mapping resource="com/durgasoft/hbn/mappings/Student.hbm.xml"/>
14. </session-factory>
15.</hibernate-configuration>
```

HibernateUtil.java

```

1. package com.durgasoft.hbn.util;
2.
3. import org.hibernate.SessionFactory;
4. import org.hibernate.cfg.Configuration;
5.
6. public class HibernateUtil {
7.     private static SessionFactory sessionFactory;
8.     static {
9.         try {
10.             Configuration cfg = new Configuration();
11.             cfg.configure("/com/durgasoft/hbn/cfgs/hibernate.cfg.xml");
12.             sessionFactory = cfg.buildSessionFactory();
13.         } catch (Exception e) {
14.             e.printStackTrace();
15.         }
16.     }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

17. public static SessionFactory getSessionFactory() {
18.     return sessionFactory;
19. }
20.

```

StudentService.java



```

1. package com.durgasoft.service;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5.
6. import com.durgasoft.hbn.pojo.Student;
7. import com.durgasoft.hbn.util.HibernateUtil;
8.
9. public class StudentService {
10.     Student std = null;
11.     public Student search(String sid) {
12.         try {
13.             SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
14.             Session session = sessionFactory.openSession();
15.             std = (Student) session.get("com.durgasoft.hbn.pojo.Student", sid);
16.         } catch (Exception e) {
17.             e.printStackTrace();
18.         }
19.         return std;
20.     }
21. }

```

SearchFrame.java

```

1. package com.durgasoft.client;
2.
3. import java.awt.Button;
4. import java.awt.Color;
5. import java.awt.FlowLayout;
6. import java.awt.Font;
7. import java.awt.Frame;
8. import java.awt.Graphics;
9. import java.awt.Label;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10. import java.awt.TextField;
11. import java.awt.event.ActionEvent;
12. import java.awt.event.ActionListener;
13. import java.awt.event.WindowAdapter;
14. import java.awt.event.WindowEvent;
15.
16. import com.durgasoft.hbn.pojo.Student;
17. import com.durgasoft.service.StudentService;
18.
19. public class SearchFrame extends Frame implements ActionListener {
20.
21.     Label l;
22.     TextField tf;
23.     Button b;
24.     Student std;
25.
26.     SearchFrame(){
27.         this.setVisible(true);
28.         this.setSize(500, 500);
29.         this.setTitle("Student Search Frame");
30.         this.setLayout(new FlowLayout());
31.         this.setBackground(Color.green);
32.         this.addWindowListener(new WindowAdapter() {
33.             public void windowClosing(WindowEvent we) {
34.                 System.exit(0);
35.             }
36.         });
37.
38.         l = new Label("Student ID");
39.         tf = new TextField(20);
40.         b = new Button("SEARCH");
41.         b.addActionListener(this);
42.
43.         Font f = new Font("arial", Font.BOLD, 20);
44.         l.setFont(f);
45.         tf.setFont(f);
46.         b.setFont(f);
47.
48.         this.add(l);
49.         this.add(tf);
50.         this.add(b);
51.     }
52.     @Override
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

53. public void actionPerformed(ActionEvent ae) {
54.
55.     String sid = tf.getText();
56.     StudentService std_Serv = new StudentService();
57.     std = std_Serv.search(sid);
58.     repaint();
59. }
60. public void paint(Graphics g) {
61.     Font f = new Font("arial", Font.BOLD, 30);
62.     this.setForeground(Color.red);
63.     g.setFont(f);
64.
65.     if(std == null) {
66.         g.drawString("Student Not Existed", 50, 300);
67.     }else {
68.         g.drawString("Student ID :" +std.getId(), 50, 300);
69.         g.drawString("Student Name :" +std.getName(), 50, 350);
70.         g.drawString("Student Address :" +std.getAddress(), 50, 400);
71.     }
72. }
73. }
```

ClientApp.java

```

1. package com.durgasoft.client;
2.
3. public class ClientApp {
4.
5.     public static void main(String[] args) {
6.         SearchFrame searchFrame = new SearchFrame();
7.
8.     }
9. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

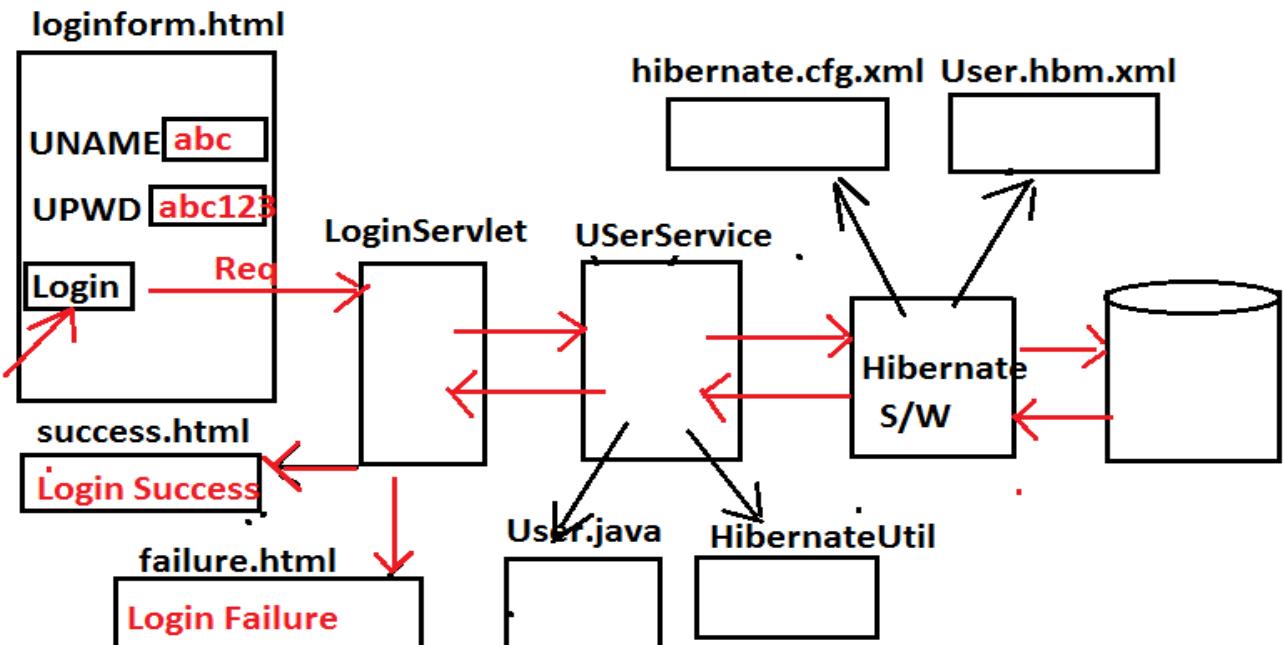
WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Servlet_Hibernate_Application:



loginform.html



```

1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
2. <html>
3. <head>
4. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Form</h3>
10. <form method = "POST" action = "./login">
11. <table>
12. <tr>
13.   <td>User Name</td>
14.   <td><input type="text" name="uname"/></td>
15. </tr>
16. <tr>
17.   <td>Password</td>
18.   <td><input type="password" name="upwd"/></td>
  
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

19.</tr>
20.<tr>
21. <td><input type="submit" value="Login"/></td>
22.</tr>
23.</table>
24.</form>
25.</body>
26.</html>
```

success.html

```

1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
2. <html>
3. <head>
4. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Status Page</h3>
10. <font color="red" size="6">
11. <b>
12. User Login Success
13. </b>
14. </font>
15. <h3>
16. <a href=".//loginform.html">|User Login Form|</a>
17. </h3>
18. </body>
19. </html>
```

failure.html

```

1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
2. <html>
3. <head>
4. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
5. <title>Insert title here</title>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Status Page</h3>
10. <font color="red" size="6">
11. <b>
12. User Login Failure
13. </b>
14. </font>
15. <h3>
16. <a href=".//loginform.html">|User Login Form|</a>
17. </h3>
18. </body>
19. </html>
```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="hibernate.connection.driver_Class">oracle.jdbc.OracleDriver</prop
erty>
8.     <property name="hibernate.connection.url">jdbc:oracle:thin:@localhost:1521:xe</pro
perty>
9.     <property name="hibernate.connection.username">system</property>
10.    <property name="hibernate.connection.password">durga</property>
11.    <property name="show_sql">true</property>
12.    <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
13.    <mapping resource="com/durgasoft/hbn/mappings/User.hbm.xml"/>
14.  </session-factory>
15. </hibernate-configuration>
```

User.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

5. <hibernate-mapping>
6.   <class name="com.durgasoft.hbn.pojo.User" table="Reg_Users">
7.     <id name="uname" column="UNAME"/>
8.     <property name="upwd" column="UPWD"/>
9.   </class>
10. </hibernate-mapping>

```



HibernateUtil.java

```

1. package com.durgasoft.hbn.util;
2.
3. import org.hibernate.SessionFactory;
4. import org.hibernate.cfg.Configuration;
5.
6. public class HibernateUtil {
7.   private static SessionFactory sessionFactory;
8.   static {
9.     try {
10.       Configuration cfg = new Configuration();
11.       cfg.configure("/com/durgasoft/hbn/cfgs/hibernate.cfg.xml");
12.       sessionFactory = cfg.buildSessionFactory();
13.       System.out.println(sessionFactory);
14.     } catch (Exception e) {
15.       e.printStackTrace();
16.     }
17.   }
18.   public static SessionFactory getSessionFactory() {
19.     return sessionFactory;
20.   }
21. }

```

User.java

```

1. package com.durgasoft.hbn.pojo;
2.
3. public class User {
4.   private String uname;
5.   private String upwd;
6.   public String getUname() {

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
7.     return uname;
8.   }
9.   public void setUsername(String uname) {
10.    this.uname = uname;
11.  }
12.  public String getPassword() {
13.    return upwd;
14.  }
15.  public void setPassword(String upwd) {
16.    this.upwd = upwd;
17.  }
18.
19.}
```

UserService.java



```
1. package com.durgasoft.service;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5.
6. import com.durgasoft.hbn.pojo.User;
7. import com.durgasoft.hbn.util.HibernateUtil;
8.
9. public class UserService {
10.   String status = "";
11.   public String checkLogin(String uname, String upwd) {
12.     try {
13.       SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
14.       Session session = sessionFactory.openSession();
15.       System.out.println(session);
16.       User user = (User)session.get(com.durgasoft.hbn.pojo.User.class, uname);
17.       if(user == null) {
18.         status = "failure";
19.       }else {
20.         if(user.getPassword().equals(upwd)) {
21.           status = "success";
22.         }else {
23.           status = "failure";
24.         }
25.       }
26.     }
27.   }
28. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

26.     } catch (Exception e) {
27.         e.printStackTrace();
28.     }
29.     return status;
30. }
31.

```



LoginServlet.java

```

1. package com.durgasoft.servlets;
2.
3. import java.io.IOException;
4.
5. import javax.servlet.RequestDispatcher;
6. import javax.servlet.ServletException;
7. import javax.servlet.http.HttpServlet;
8. import javax.servlet.http.HttpServletRequest;
9. import javax.servlet.http.HttpServletResponse;
10.
11. import com.durgasoft.hbn.util.HibernateUtil;
12. import com.durgasoft.service.UserService;
13. public class LoginServlet extends HttpServlet {
14.     public void init() {
15.         HibernateUtil.getSessionFactory();
16.         System.out.println("init()");
17.     }
18.     private static final long serialVersionUID = 1L;
19.     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
20.             ServletException, IOException {
21.         String uname = request.getParameter("uname");
22.         String upwd = request.getParameter("upwd");
23.
24.         UserService user_Service = new UserService();
25.         String status = user_Service.checkLogin(uname, upwd);
26.
27.         RequestDispatcher requestDispatcher = null;
28.         if(status.equals("success")) {
29.             requestDispatcher = request.getRequestDispatcher("/success.html");
30.             requestDispatcher.forward(request, response);
31.         }else {
32.             requestDispatcher = request.getRequestDispatcher("/failure.html");
33.         }
34.     }
35. }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
32.         requestDispatcher.forward(request, response);
33.     }
34. }
35.}
```

**web.xml**

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.c
   om/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
   app_2_5.xsd" id="WebApp_ID" version="2.5">
3.   <display-name>app07</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
11.   </welcome-file-list>
12.   <servlet>
13.     <description></description>
14.     <display-name>LoginServlet</display-name>
15.     <servlet-name>LoginServlet</servlet-name>
16.     <servlet-class>com.durgasoft.servlets.LoginServlet</servlet-class>
17.     <load-on-startup>1</load-on-startup>
18.   </servlet>
19.   <servlet-mapping>
20.     <servlet-name>LoginServlet</servlet-name>
21.     <url-pattern>/login</url-pattern>
22.   </servlet-mapping>
23. </web-app>
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

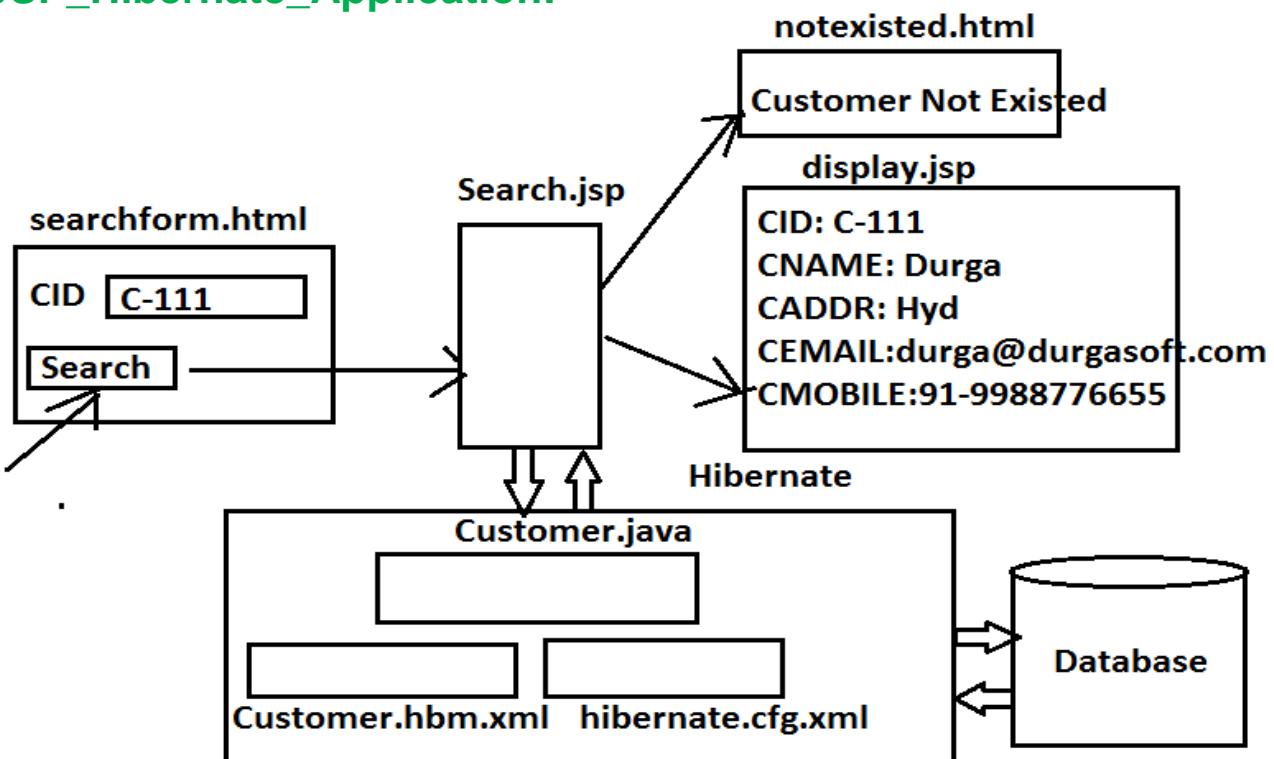
Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

JSP_Hibernate_Application:



searchform.html



```

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>Customer Search Form</h3>
10. <form method="POST" action=".//search.jsp">
11. <table>
12. <tr>
13. <td>Customer ID</td>
14. <td><input type="text" name="cid"/></td>
15. </tr>
16. <tr>
  
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

17. <td><input type="submit" value="SEARCH"/></td>
18.</tr>
19.</table>
20.</form>
21.</body>
22.</html>
```

notexisted.html



```

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>Customer Search Status</h3>
10.<font color="red" size="6">
11.<b>
12.Customer Not Existed
13.</b>
14.</font>
15.<h3>
16.<a href=".//searchform.html">|Customer Search Form|</a>
17.</h3>
18.</body>
19.</html>
```

display.jsp



```

1. <%@page import="com.durgasoft.hbn.pojo.Customer"%>
2. <%!
3. Customer customer = null;
4. %>
5. <%
6. customer = (Customer)request.getAttribute("customer");
7. %>
8. <html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

9. <body>
10. <h2>Durga Software Solutions</h2>
11. <h3>Customer Details</h3>
12. <table border="1">
13. <tr>
14.   <td>Customer ID</td>
15.   <td><%=customer.getCid() %> </td>
16. </tr>
17. <tr>
18.   <td>Customer Name</td>
19.   <td><%=customer.getCname() %> </td>
20. </tr>
21. <tr>
22.   <td>Customer Address</td>
23.   <td><%=customer.getAddr() %> </td>
24. </tr>
25. <tr>
26.   <td>Customer Email</td>
27.   <td><%=customer.getEmail() %> </td>
28. </tr>
29. <tr>
30.   <td>Customer Mobile</td>
31.   <td><%=customer.getMobile() %> </td>
32. </tr>
33. </table>
34. <h3>
35. <a href=".//searchform.html">|Customer Search Form|</a>
36. </h3>
37. </body>
38. </html>

```

search.jsp

```

1. <%@page import="com.durgasoft.hbn.pojo.Customer"%>
2. <%@page import="org.hibernate.Session"%>
3. <%@page import="org.hibernate.SessionFactory"%>
4. <%@page import="org.hibernate.cfg.Configuration"%>
5. <%!
6. Configuration cfg = null;
7. SessionFactory sessionFactory = null;
8. Session ses = null;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

9. Customer customer = null;
10.%>
11.<%
12.try{
13.String cid = request.getParameter("cid");
14.cfg = new Configuration();
15.cfg.configure("/com/durgasoft/hbn/cfgs/hibernate.cfg.xml");
16.sessionFactory = cfg.buildSessionFactory();
17.ses = sessionFactory.openSession();
18.customer = (Customer)ses.get(Customer.class, cid );
19.if(customer == null){
20.%>
21.<jsp:forward page="notexisted.html"/>
22.<%
23.}else{
24.request.setAttribute("customer", customer);
25.%>
26.<jsp:forward page="display.jsp"/>
27.<%
28.}
29.}catch(Exception e){
30.e.printStackTrace();
31.}
32.%>
```

Customer.java

```

1. package com.durgasoft.hbn.pojo;
2.
3. public class Customer {
4.     private String cid;
5.     private String cname;
6.     private String caddr;
7.     private String cemail;
8.     private String cmobile;
9.
10.    public String getCid() {
11.        return cid;
12.    }
13.    public void setCid(String cid) {
14.        this.cid = cid;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

15. }
16. public String getCname() {
17.     return cname;
18. }
19. public void setCname(String cname) {
20.     this.cname = cname;
21. }
22. public String getCaddr() {
23.     return caddr;
24. }
25. public void setCaddr(String caddr) {
26.     this.caddr = caddr;
27. }
28. public String getCemail() {
29.     return cemail;
30. }
31. public void setCemail(String cemail) {
32.     this.cemail = cemail;
33. }
34. public String getCmobile() {
35.     return cmobile;
36. }
37. public void setCmobile(String cmobile) {
38.     this.cmobile = cmobile;
39. }
40.
41.
42.}
```

Customer.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.     "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.     "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.hbn.pojo.Customer" table="customer">
7.     <id name="cid" column="CID"/>
8.     <property name="cname" column="CNAME"/>
9.     <property name="caddr" column="CADDR"/>
10.    <property name="cemail" column="CEMAIL"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

11. <property name="cmobile" column="CMOBILE"/>
12. </class>
13.</hibernate-mapping>
```



hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7.   <property name="connection.driver_Class">com.mysql.jdbc.Driver</property>
8.   <property name="connection.url">jdbc:mysql://localhost:3306/durgadb</property>
9.   <property name="connection.username">root</property>
10.  <property name="connection.password">root</property>
11.  <property name="show_sql">true</property>
12.  <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
13.  <mapping resource="com/durgasoft/hbn/mappings/Customer.hbm.xml"/>
14.</session-factory>
15.</hibernate-configuration>
```

web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.c
om/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
app_2_5.xsd" id="WebApp_ID" version="2.5">
3. <display-name>app08</display-name>
4. <welcome-file-list>
5.   <welcome-file>index.html</welcome-file>
6.   <welcome-file>index.htm</welcome-file>
7.   <welcome-file>index.jsp</welcome-file>
8.   <welcome-file>default.html</welcome-file>
9.   <welcome-file>default.htm</welcome-file>
10.  <welcome-file>default.jsp</welcome-file>
11. </welcome-file-list>
12.</web-app>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Struts_Hibernate Integration:

Struts Overview:

- ❖ Framework is a semi implemented application, it will be used to design applications in simplified manner.
- ❖ Framework is prefabricated software units that programmer can share, customize, reuse in order to simplify application development.
- ❖ Framework is the collection of predefined classes and interfaces to simplify application development.

In general, in Applications development, Frameworks will provide the following advantages.

1. Frameworks will provide common implementation in all the projects as predefined implementation
EX: Controller Servlets and the services like I18N, Exception Handling, Validations ,.....
2. Frameworks will provide standard template to design applications.
3. Frameworks will provide standard flow of execution between the components.
4. Frameworks will reduce application development time.
5. Frameworks will reduce application development cost.
6. Frameworks will improve productivity.
7. Frameworks will provide modularity in application development.

There are two types of Frameworks.

1. Web Frameworks
2. Application Frameworks

What is the difference between web frameworks and Application Frameworks?

Ans:

Web Frameworks will provide very good environment to prepare and execute web applications only.

EX: Struts, JSF.

Application Frameworks will provide very good environment to prepare and execute any type of JAVA/J2EE Applications including Standalone Applications, Web applications, Distributed Applications,

EX: Spring

Struts is MVC based web framework provided by Apache Software Foundations.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Struts is existed in two versions.

1. Struts1.x
2. Struts2.x

If we want to prepare Struts applications in Struts1.x version then we have to use the following components

1. View
2. Deployment Descriptor
3. Controller [ActionServlet]
4. Action class
5. Action Form
6. Struts Configuration File

1. View:

In Struts applications, View part is representing presentation part.

In web applications, there are two types of presentation part.

1. Informational Presentation part
2. Form Based Presentation part

1. Informational Presentation part

This type view part is able to provide only information to the user, which includes status of the server side actions like Success, failure, ... and display a particular database table data,.....

EXAMPLE: success.html

```
<h1> Login Success </h1>
```

2. Form Based Presentation part:

This type of view part is able to provide a form to collect data from users and to submit data to the server side applications.

In Struts applications, to prepare Presentation part we are able to use plain HTML tags, but, which are not suggestible. In Struts applications it is suggestible to use Struts provided tag library.

EX: loginform.html

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
1. <html>
2. <body>
3. <form method="POST" action="login.*">
4. <table>
5. <tr>
6.   <td>User Name</td>
7.   <td><input type="text" name="uname"/></td>
8. </tr>
9. <tr>
10.  <td>Password</td>
11.  <td><input type="password" name="upwd"/></td>
12.</tr>
13.<tr>
14.  <td><input type="submit" value="Login"/></td>
15.</tr>
16.</table></form></body></html>
```

The above login form with Struts provided html tag library

EX: loginform.jsp

```
1. <%@taglib uri="http://struts.apache.org/tags-html" prefix="html" %>
2. <html:html>
3. <body>
4. <html:form method="POST" action="login.do">
5. <table>
6. <tr>
7.   <td>User Name</td>
8.   <td><html:text property="uname"/></td>
9. </tr>
10. <tr>
11.   <td>Password</td>
12.   <td><html:password property="upwd"/></td>
13. </tr>
14. <tr>
15.   <td><html:submit>Login</html:submit></td>
16. </tr>
17. </table></html:form></body></html>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. Deployment Descriptor:

Deployment descriptor is web.xml file, it will provide description about our web application which is required by the container in order to perform server side actions.

In general, in web applications, web.xml file will provide the following configuration details.

1. Display names configurations.
2. Welcome Files Configurations.
3. Servlets configurations
4. Filters Configurations
5. Listeners Configurations.
6. Session time out configurations
7. Error Pages Configurations
8. Taglib configurations

In Struts based web applications, the main intention of web.xml file is to configure controller Servlet that is ActionServlet.

EX:

```
1. <web-app>
2. <servlet>
3.   <servlet-name>actionServlet</servlet-name>
4.   <servlet-class>org.apache.struts.action.ActionServlet
5.   </servlet-class>
6.   <load-on-startup>1</load-on-startup>
7. </servlet>
8. <servlet-mapping>
9.   <servlet-name>actionServlet</servlet-name>
10.  <url-pattern>*.do</url-pattern>
11. </servlet-mapping>
12.</web-app>
```

3. Controller [ActionServlet]

In Struts Framework, ActionServlet is predefined Controller, it was provided in the form of org.apache.struts.action.ActionServlet .

In Struts applications, ActionServlet will perform the following actions.

1. ActionServlet will take request from Client.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2. ActionServlet will identify the names and locations of the ActionForm and Action class through Struts configuration file.
3. ActionServlet will load, instantiate ActionForm class.
4. ActionServlet will store form data in ActionForm object.
5. ActionServlet will load and instantiation Action class.
6. ActionServlet will execute execute(--) method in Action class.
7. ActionServlet will identify view page through Struts configuration file.
8. ActionServlet will forward request to view page inorder to generate response.

Note: In Struts applications, ActionServlet is performing all the above actions by using "RequestProcessor" internally.

4. Action Class or Controller Component:

In Struts based web applications, the main intention of ActionForm or FormBean component is to manage a particular form data at Server side inorder to perform Server side data validations, to transfer data from Controller layer to model layer or Vie layer,.....

To prepare Form Bean components in Struts applications we have to use the following rules and regulations.

1. Declare an user defined class, it must be extended from org.apache.struts.action.ActionForm .
2. Form Bean class must be public, it must not be abstract and final.
3. In Form Bean class, we must declare properties with the same names of the form properties.
4. In Form Bean class we must declare all properties as private and all behaviours as public.
5. In FormBean class , we must provide a seperate set of setXXX() and getXXX() methods for each and every property.
6. In FormBean class, if we want to provide any constructor then we can provide constructor but that constructor must be public and 0-arg constructor.
7. In FormBean class, we can override equals(-) method and hashCode() method as per the requirement.

EXAMPLE:

LoginActionForm.java

- ```
1. public class LoginActionForm extends org.apache.struts.action.ActionForm{
2. private String uname;
3. private String upwd;
4. public void setUsername(String uname){
```

#### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. this.uname = uname;
6. }
7. public void setUpwd(String upwd){
8. this.upwd = upwd;
9. }
10. public String getUsername(){
11. return uname;
12. }
13. public String getUpwd(){
14. return upwd;
15. }
16.}
```

## 5. Action Class or Controller Component:

In Struts based web applications, the main intention of Action class is to manage application logic which we want to execute by getting request from client.

In Struts based web applications, to prepare Action class we have to use the following steps.

1. Declare an user defined class and it must be extended from org.apache.struts.action.Action class.
2. Action class must be public class and it must not be abstract class.
3. In Action class we must override either of the following methods.

public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request, HttpServletResponse response) throws Exception

public ActionForward execute(ActionMapping mapping, ActionForm form, ServletRequest request, ServletResponse response) throws Exception.

**Note:** In execute() method we must return ActionForward object with a particular Forward key inorder to identify target view page, for this, we have to use the following method from ActionMapping.

public ActionForward findForward(String key)

### EX: LoginAction.java

1. **public class** LoginAction **extends** org.apache.struts.action.Action{
2. **public** ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request, HttpServletResponse response) **throws** Exception{

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
3. ----Appl logic----
4. String status = "success"/"failure";
5. return mapping.findForward(status);
6. }
7. }
```

## 6. Struts Configuration File:

### struts-config.xml

- In Struts based web applications, the main intention of struts configuration file is to provide mappings between user forms and the respective ActionForm classes and Action classes ,..... which are required by the ActionServlet inorder to perform Server side actions.
- In Struts based web applications, the default name and location of configuration file is "struts-config.xml" and "WEB-INF" location, but, it is possible to change this default name and location but we must give that new name and location to the Struts Framework.
- In Struts based web applications, configuration file is able to provide the following configurations.
  1. Datasource configurations.
  2. Global Forwards configurations.
  3. Form Beans Configurations.
  4. Action classes configurations.
  5. Message Resources configurations.
  6. Controller configurations.
  7. Plugin configurations
  8. Global Exceptions configurations.

To prepare basic Struts based web applications we need to provide ActionForm class configuration and action class configuration in struts configuration file.

```
1. <!DOCTYPE ---- >
2. <struts-config>
3. <form-beans>
4. <form-bean name="--" type="--"/>
5. -----
6. </form-beans>
7. <action-mappings>
8. <action path="/---" name="--" type="--">
```

## CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

9. <forward name="--" path="/---"/>
10. -----
11. </action>
12. -----
13. </action-mappings>
14. -----
15. </struts-config>
```

- ✓ Where <struts-config> is a root tag, it will include struts application configuration details.
- ✓ Where <form-beans> tag is able to include no of form beans configurations.
- ✓ Where <form-bean> tag is able to provide single form bean class configuration.
- ✓ Where "name" attribute in <form-bean> tag is able to provide logical name to Form Bean component.
- ✓ Where "type" attribute in <form-bean> tag will take fully qualified name of the respective form bean class.
- ✓ Where <action-mappings> tag is able to include no of actions configurations.
- ✓ Where <action> tag is able to provide mapping between user form, ActionForm class and the respective Action class.
- ✓ Where "path" attribute in <action> tag is able to take url pattern which we specified in User form.
- ✓ Where "name" attribute in <action> tag will take logical name of the form bean class which we specified in struts configuration file under form beans configurations.
- ✓ Where "type" attribute in <action> tag is able to take fully qualified name of the Action class.
- ✓ Where <forward> tag is able to provide mapping between forward key which is returned from execute() method in Action class and the target view page.
- ✓ Where "name" attribute in <forward> tag is able to take forward key.
- ✓ Where "path" attribute in <forward> tag is able to take the name and location of target view page.

EX:

```

1. <!DOCTYPE >
2. <struts-config>
3. <form-beans>
4. <form-
 bean name="loginForm" type="com.durgasoft.beans.LoginActionForm"/>
5. </form-beans>
6. <action-mappings>
7. <action path="/login" name="loginForm" type="com.durgasoft.action.LoginAct
 ion">
```

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
8. <forward name="success" path="/success.html"/>
9. <forward name="failure" path="/failure.html"/>
10. </action-mappings>
11. </struts-config>
```

**Steps:**

1. Download Struts JARs from internet.
2. Create Dynamic Project in Eclipse.
3. Copy all Struts related JARs in lib folder in dynamic web project.
4. Configure ActionServlet in web.xml file.
5. Prepare User forms
6. Prepare Form Bean class.
7. Create Action class.
8. Create Struts configuration file.
9. Run dynamic web application.

**Example (struts Application)****loginform.html**

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Form</h3>
10. <form method="POST" action="login.do">
11. <table>
12. <tr>
13. <td>User Name</td>
14. <td><input type="text" name="uname"/></td>
15. </tr>
16. <tr>
17. <td>Password</td>
18. <td><input type="password" name="upwd"/></td>
19. </tr>
20. <tr>
21. <td><input type="submit" value="Login"/></td>
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

22.</tr>
23.</table>
24.</form>
25.</body>
26.</html>
```

### success.html



```

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Status</h3>
10.
11.User Login Success
12.
13.<h5>
14.|User Login Form|
15.</h5>
16.</body>
17.</html>
```

### Failure.html



```

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>User Login Status</h3>
10.
11.User Login Failure
12.
13.<h5>
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
14. |User Login Form|
15. </h5>
16. </body>
17. </html>
```



### LoginActionForm.java

```
1. package com.durgasoft.beans;
2.
3. import org.apache.struts.action.ActionForm;
4.
5. public class LoginActionForm extends ActionForm {
6. private String uname;
7. private String upwd;
8.
9. public String getUsername() {
10. return uname;
11. }
12. public void setUsername(String uname) {
13. this.uname = uname;
14. }
15. public String getUpwd() {
16. return upwd;
17. }
18. public void setUpwd(String upwd) {
19. this.upwd = upwd;
20. }
21.
22. }
```

### LoginAction.java

```
1. package com.durgasoft.action;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.apache.struts.action.Action;
7. import org.apache.struts.action.ActionForm;
8. import org.apache.struts.action.ActionForward;
9. import org.apache.struts.action.ActionMapping;
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10.
11. import com.durgasoft.beans.LoginActionForm;
12.
13. public class LoginAction extends Action {
14. @Override
15. public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request,
16. HttpServletResponse response) throws Exception {
17. LoginActionForm laf = (LoginActionForm)form;
18. String uname = laf.getUname();
19. String upwd = laf.getUpwd();
20. String status = "";
21. if(uname.equals("durga") && upwd.equals("durga")) {
22. status = "success";
23. }else {
24. status = "failure";
25. }
26. return mapping.findForward(status);
27. }
28. }
```

### struts-config.xml



```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE struts-config PUBLIC
3. "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
4. "http://struts.apache.org/dtds/struts-config_1_3.dtd">
5. <struts-config>
6. <form-beans>
7. <form-bean name="loginForm" type="com.durgasoft.beans.LoginActionForm"/>
8. </form-beans>
9. <action-mappings>
10. <action path="/login" name="loginForm" type="com.durgasoft.action.LoginAction">
11. <forward name="success" path="/success.html"/>
12. <forward name="failure" path="/failure.html"/>
13. </action>
14. </action-mappings>
15. </struts-config>
```

### web.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

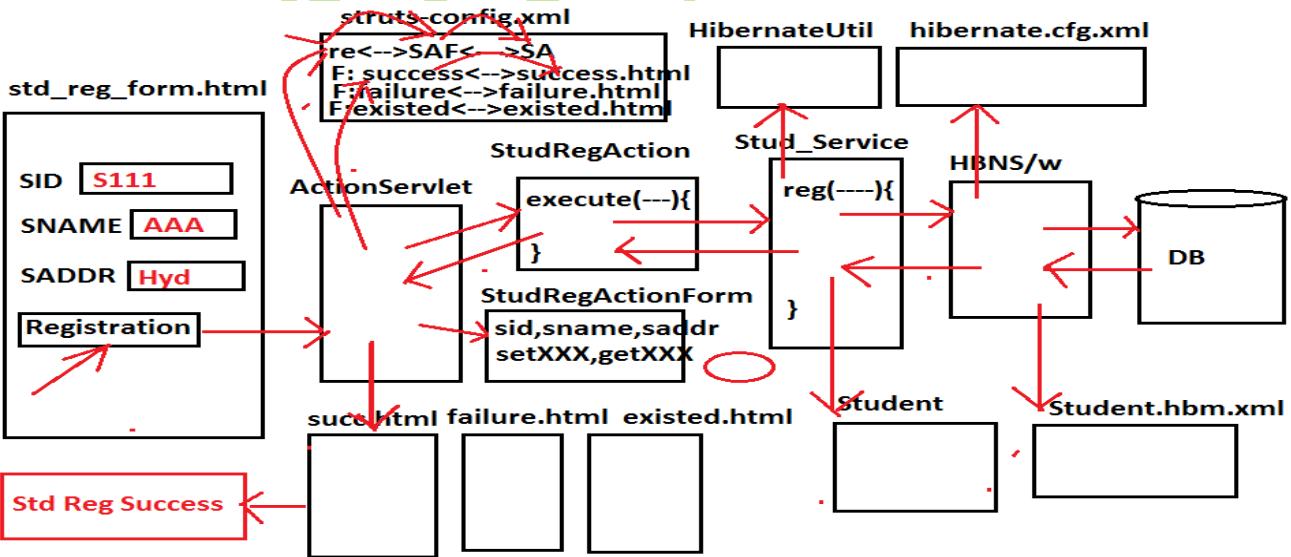
```

2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.c
om/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
app_2_5.xsd" id="WebApp_ID" version="2.5">
3. <display-name>loginapp</display-name>
4. <welcome-file-list>
5. <welcome-file>loginform.html</welcome-file>
6. </welcome-file-list>
7. <servlet>
8. <servlet-name>actionServlet</servlet-name>
9. <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
10. <load-on-startup>1</load-on-startup>
11. </servlet>
12. <servlet-mapping>
13. <servlet-name>actionServlet</servlet-name>
14. <url-pattern>*.do</url-pattern>
15. </servlet-mapping>
16. </web-app>

```

## Integration Of Hibernate with Struts:

1. Manual Approach:
2. By Customizing ActionServlet
3. By Using PlugIn Approach



## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

## 1. Manual Approach:

### Hibernate:

1. Student.java
2. Student.hbm.xml
3. hibernate.cfg.xml
4. HibernateUtil.java
5. StudentService.java

### Struts:

1. StudentRegistrationAction.java
2. StudentRegistrationActionForm.java
3. student\_registration\_form.html
4. success.html
5. failure.html
6. existed.html
7. struts-config.xml
8. web.xml

### Example:

#### registrationform.html

```
1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>Student Registration Form</h3>
10. <form method="POST" action="reg.do">
11. <table>
12. <tr>
13. <td>Student Id</td>
14. <td><input type="text" name="sid"/></td>
15. </tr>
16. <tr>
17. <td>Student Name</td>
18. <td><input type="text" name="sname"/></td>
19. </tr>
20. <tr>
```



### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

21. <td>Student Address</td>
22. <td><input type="text" name="saddr"/></td>
23.</tr>
24.<tr>
25. <td><input type="submit" value="Registration"/></td>
26.</tr>
27.</table>
28.</form>
29.</body>
30.</html>
```

### success.html

```

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>Student Registration Status</h3>
10.
11.
12. Student Registration Success
13.
14.
15. <h5>
16. |Student Registration Form|
17. </h5>
18. </body>
19. </html>
```

### failure.html

```

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

8. <h2>Durga Software Solutions</h2>
9. <h3>Student Registration Status</h3>
10.
11.
12. Student Registration Failure
13.
14.
15. <h5>
16. |Student Registration Form|
17. </h5>
18. </body>
19. </html>
```

### existed.html

```

1. <!DOCTYPE html>
2. <html>
3. <head>
4. <meta charset="ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <h2>Durga Software Solutions</h2>
9. <h3>Student Registration Status</h3>
10.
11.
12. Student Existed Already
13.
14.
15. <h5>
16. |Student Registration Form|
17. </h5>
18. </body>
19. </html>
```

### StudentActionForm.java

```

1. package com.durgasoft.beans;
2.
3. import org.apache.struts.action.ActionForm;
4.
5. public class StudentActionForm extends ActionForm {
6. private String sid;
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

7. private String sname;
8. private String saddr;
9. public String getSid() {
10. return sid;
11. }
12. public void setSid(String sid) {
13. this.sid = sid;
14. }
15. public String getSname() {
16. return sname;
17. }
18. public void setSname(String sname) {
19. this.sname = sname;
20. }
21. public String getSaddr() {
22. return saddr;
23. }
24. public void setSaddr(String saddr) {
25. this.saddr = saddr;
26. }
27.
28.
29.}
```

### StudentAction.java

```

1. package com.durgasoft.action;
2.
3. import javax.servlet.http.HttpServletRequest;
4. import javax.servlet.http.HttpServletResponse;
5.
6. import org.apache.struts.action.Action;
7. import org.apache.struts.action.ActionForm;
8. import org.apache.struts.action.ActionForward;
9. import org.apache.struts.action.ActionMapping;
10.
11. import com.durgasoft.beans.StudentActionForm;
12. import com.durgasoft.service.StudentService;
13.
14. public class StudentAction extends Action {
15. @Override
16. public ActionForward execute(ActionMapping mapping, ActionForm form, HttpServletRequest request,
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

17. HttpServletRequest response) throws Exception {
18. String status = "";
19. StudentActionForm saf = (StudentActionForm)form;
20. String sid = saf.getSid();
21. String sname = saf.getSname();
22. String saddr = saf.getSaddr();
23. StudentService stdService = new StudentService();
24. status = stdService.registration(sid, sname, saddr);
25. return mapping.findForward(status);
26. }
27.

```

**StudentService.java**

```

1. package com.durgasoft.service;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6.
7. import com.durgasoft.hbn.pojo.Student;
8. import com.durgasoft.hbn.util.HibernateUtil;
9.
10. public class StudentService {
11. String status = "";
12. public String registration(String sid, String sname, String saddr) {
13. try {
14. SessionFactory sessionFactory = HibernateUtil.getSessionFactory();
15. Session session = sessionFactory.openSession();
16. Student std = (Student) session.get(Student.class, sid);
17. if(std == null) {
18. std = new Student();
19. std.setSid(sid);
20. std.setSname(sname);
21. std.setSaddr(saddr);
22. Transaction tx = session.beginTransaction();
23. session.save(std);
24. tx.commit();
25. status = "success";
26. }else {
27. status = "existed";
28. }
29. } catch (Exception e) {

```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
30. status = "failure";
31. e.printStackTrace();
32. }
33. return status;
34. }
35.}
```

**Student.java**

```
1. package com.durgasoft.hbn.pojo;
2.
3. public class Student {
4. private String sid;
5. private String sname;
6. private String saddr;
7.
8. public String getSid() {
9. return sid;
10. }
11. public void setSid(String sid) {
12. this.sid = sid;
13. }
14. public String getSname() {
15. return sname;
16. }
17. public void setSname(String sname) {
18. this.sname = sname;
19. }
20. public String getSaddr() {
21. return saddr;
22. }
23. public void setSaddr(String saddr) {
24. this.saddr = saddr;
25. }
26.
27.
28.}
```

**HibernateUtil.java**

```
1. package com.durgasoft.hbn.util;
2.
3. import org.hibernate.SessionFactory;
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)



BY NAGOOR BABU

```

4. import org.hibernate.cfg.Configuration;
5.
6. public class HibernateUtil {
7. static SessionFactory sessionFactory;
8. static {
9. try {
10. Configuration cfg = new Configuration();
11. cfg.configure("/com/durgasoft/hbn/cfgs/hibernate.cfg.xml");
12. sessionFactory = cfg.buildSessionFactory();
13. } catch (Exception e) {
14. e.printStackTrace();
15. }
16. }
17. public static SessionFactory getSessionFactory() {
18. return sessionFactory;
19. }
20. }
```

### Student.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3. "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="com.durgasoft.hbn.pojo.Student" table="student">
7. <id name="sid" column="SID"/>
8. <property name="sname" column="SNAME"/>
9. <property name="saddr" column="SADDR"/>
10. </class>
11. </hibernate-mapping>
```

### hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="connection.driver_Class">com.mysql.jdbc.Driver</property>
8. <property name="connection.url">jdbc:mysql://localhost:3306/durgadb</property>
9. <property name="connection.username">root</property>
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10. <property name="connection.password">root</property>
11. <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
12. <mapping resource="com/durgasoft/hbn/mappings/Student.hbm.xml"/>
13. </session-factory>
14.</hibernate-configuration>
```

### struts-config.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE struts-config PUBLIC
3. "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
4. "http://struts.apache.org/dtds/struts-config_1_3.dtd">
5. <struts-config>
6. <form-beans>
7. <form-
8. bean name="studentActionForm" type="com.durgasoft.beans.StudentActionForm"/>
9. </form-beans>
10. <action-mappings>
11. <action path="/reg" name="studentActionForm" type="com.durgasoft.action.StudentA
ction">
12. <forward name="success" path="/success.html"/>
13. <forward name="failure" path="/failure.html"/>
14. <forward name="existed" path="/existed.html"/>
15. </action>
16. </action-mappings>
17.</struts-config>
```

### web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.c
om/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
app_2_5.xsd" id="WebApp_ID" version="2.5">
3. <display-name>struts_hibernate_app1</display-name>
4. <welcome-file-list>
5. <welcome-file>registrationform.html</welcome-file>
6. </welcome-file-list>
7. <servlet>
8. <servlet-name>actionServlet</servlet-name>
9. <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10. <load-on-startup>1</load-on-startup>
11. </servlet>
12. <servlet-mapping>
13. <servlet-name>actionServlet</servlet-name>
14. <url-pattern>*.do</url-pattern>
15. </servlet-mapping>
16.</web-app>
```

In the above application, Struts Framework will be loaded at the time of Server Startup, because, ActionServlet initialization is performed at the time of Server Startup due to `<load-on-startup>` configuration in ActionServlet configuration in web.xml file, but, Hibernate software is loaded after the Server startup and after getting first request from client, it may take lot of time to load Hibernate Software, it will increase response time and it will reduce application performance.

In the above context, to improve application performance we have to load Hibernate Software along with Struts framework loading, that is, at the time of Server startup and as part of ActionServlet initialization.

To achieve the above requirement we have to Customize ActionServlet initialization process.

## 2) Hibernate integration with Struts by Customizing ActionServlet:

### 1) Prepare Our own Controller class by extending ActionServlet and override "init()" method:

Example:

```
1. package com.durgasoft.controller;
2. import javax.servlet.ServletConfig;
3. import javax.servlet.ServletException;
4. import org.apache.struts.action.ActionServlet;
5. import com.durgasoft.hbn.util.HibernateUtil;
6. public class MyController extends ActionServlet {
7. public void init(ServletConfig config) throws ServletException {
8. super.init(config);
9. HibernateUtil.getSessionFactory();
10. System.out.println("*****MyController-init()*****");
11. System.out.println("Struts Framework is Loaded");
12. System.out.println("Hibernate is Loaded");
13. System.out.println("*****MyController-init()*****");
14. }
15. }
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**2) Configure User Defined Controller class in web.xml file:**

```

1. <web-app>
2. <welcome-file-list>
3. <welcome-file>registrationform.html</welcome-file>
4. </welcome-file-list>
5. <servlet>
6. <servlet-name>myController</servlet-name>
7. <servlet-class>com.durgasoft.controller.MyController</servlet-class>
8. <load-on-startup>1</load-on-startup>
9. </servlet>
10. <servlet-mapping>
11. <servlet-name>myController</servlet-name>
12. <url-pattern>*.do</url-pattern>
13. </servlet-mapping>
14. </web-app>

```

In Struts based web applications, it is not suggestible to customize predefined controller ActionServlet , because, ActionServlet is having no of responsibilities as part of its initialization, if we have not overridden init() method properly then the total ActionServlet may not be executed, it may not load Struts Framework.

**3. Integration of Hibernate with Struts by using plugin Approach:**

- a) Prepare Hibernate Plugin class.
- b) Configure hibernate Plugin class in struts configuration file.

**a) Prepare Hibernate Plugin class:**

To prepare Hibernate Plugin class, declare an user defined class and implement org.apache.struts.action.Plugin interface and provide implementation for the following methods.

1. public void init(ActionServlet as, ModuleConfig config) throws ServletException
2. public void destroy()

**Ex:**

```

1. package com.durgasoft.plugin;
2. import javax.servlet.ServletException;
3. import org.apache.struts.action.ActionServlet;
4. import org.apache.struts.action.Plugin;
5. import org.apache.struts.config.ModuleConfig;

```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

6. import com.durgasoft.hbn.util.HibernateUtil;
7. public class HibernatePlugIn implements PlugIn {
8. public void init(ActionServlet arg0, ModuleConfig arg1) throws ServletException {
9. System.out.println("*****HibernatePlugIn-init()*****");
10. HibernateUtil.getSessionFactory();
11. System.out.println("Hibernate is Loaded along with Struts Loading");
12. System.out.println("*****HibernatePlugIn-init()*****");
13. }
14. public void destroy() {
15. }
16. }
```

### b) Configure Hibernate Plugin class in Struts configuration File:

```

1. <struts-config>
2. ----
3. <plug-in className="com.durgasoft.plugin.HibernatePlugIn"/>
4. ----
5. </struts-config>
```

### Internal Flow:

When we start Server, Container will perform the following actions.

1. Container will deploy all web applications which are available in webapps folder or in deployments folder.
2. Container will recognize web.xml file and perform web.xml file loading and parsing.
3. Container will recognize <load-on-startup> configuration in ActionServlet configuration in web.xml file then performs ActionServlet loading, instantiation and initialization.
4. As part of ActionServlet initialization, ActionServlet will recognize struts configuration file then performs struts configuration file loading and parsing.
5. ActionServlet will recognize <plug-in> tag in struts configuration file and identify HibernatePlugIn class .
6. ActionServlet will perform HibernatePlugIn class loading , instantiation and initialization.
7. As part of HibernatePlugIn class initialization,ActionServlet will load Hibernate software and prepare Configuration and SessionFactory objects with HibernateUtil class at the time of Server startup.

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**Is it possible to interact with more than one Database from a single Hibernate Application?****Ans:**

Yes, in single hibernate application we are able to interact with more than one Database, but, we must use the following conventions.

- 1) We must provide separate configuration file for each and every Database.
- 2) Use either single mapping file if we have same table name and same columns in both the databases tables and use different mapping file for each and every database if we have difference in table names and column names.
- 3) In Client application.
  - a) Prepare Separate Configuration object for each and every DB.
  - b) Prepare separate SessionFactory object for each and every DB.
  - c) Prepare separate Session object for each and every DB.
  - d) Prepare Separate Transaction object for each and every DB as per the requirement.
  - e) Perform persistence operations on DB respective Session objects.

**Example:****Employee.java**

```
1. package com.durgasoft.hbn.pojo;
2.
3. public class Employee {
4. private int eno;
5. private String ename;
6. private float esal;
7. private String eaddr;
8. public int getEno() {
9. return eno;
10. }
11. public void setEno(int eno) {
12. this.eno = eno;
13. }
14. public String getEname() {
15. return ename;
16. }
17. public void setEname(String ename) {
18. this.ename = ename;
19. }
20. public float getEsal() {
21. return esal;
22. }
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

22. }
23. public void setEsal(float esal) {
24. this.esal = esal;
25. }
26. public String getEaddr() {
27. return eaddr;
28. }
29. public void setEaddr(String eaddr) {
30. this.eaddr = eaddr;
31. }
32. }
```

### Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3. "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
7. <id name="eno" column="ENO"/>
8. <property name="ename" column="ENAME"/>
9. <property name="esal" column="ESAL"/>
10. <property name="eaddr" column="EADDR"/>
11. </class>
12.</hibernate-mapping>
```

### hibernate\_oracle.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8. <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9. <property name="connection.user">system</property>
10. <property name="connection.password">durga</property>
11. <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
12. <property name="show_sql">true</property>
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

13. <mapping resource="com/durgasoft/hbn/mappings/Employee.hbm.xml"/>
14. </session-factory>
15.</hibernate-configuration>
```



### hibernate\_mysql.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="connection.driver_Class">com.mysql.jdbc.Driver</property>
8. <property name="connection.url">jdbc:mysql://localhost:3306/durgadb</property>
9. <property name="connection.user">root</property>
10. <property name="connection.password">root</property>
11. <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
12. <property name="show_sql">true</property>
13. <mapping resource="com/durgasoft/hbn/mappings/Employee.hbm.xml"/>
14. </session-factory>
15.</hibernate-configuration>
```

### ClientApp.java

```

1. package com.durgasoft.hbn.client;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.cfg.Configuration;
7.
8. import com.durgasoft.hbn.pojo.Employee;
9.
10. public class ClientApp {
11.
12. public static void main(String[] args) {
13. SessionFactory oracle_SessionFactory = null;
14. SessionFactory mysql_SessionFactory = null;
15. Session oracle_Session = null;
16. Session mysql_Session = null;
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
17. try {
18. Configuration oracle_Cfg = new Configuration();
19. Configuration mysql_Cfg = new Configuration();
20.
21. oracle_Cfg.configure("hibernate_oracle.cfg.xml");
22. mysql_Cfg.configure("hibernate_mysql.cfg.xml");
23.
24. oracle_SessionFactory = oracle_Cfg.buildSessionFactory();
25. mysql_SessionFactory = mysql_Cfg.buildSessionFactory();
26.
27. oracle_Session = oracle_SessionFactory.openSession();
28. mysql_Session = mysql_SessionFactory.openSession();
29.
30. Employee emp = (Employee) oracle_Session.get(Employee.class, 111);
31. System.out.println("Employee retrived from Oracle DB");
32.
33. Transaction mysql_Tx = mysql_Session.beginTransaction();
34. mysql_Session.save(emp);
35. mysql_Tx.commit();
36. System.out.println("Employee Inserted in MySQL DB");
37.
38. } catch (Exception e) {
39. e.printStackTrace();
40. }finally {
41. oracle_Session.close();
42. mysql_Session.close();
43.
44. oracle_SessionFactory.close();
45. mysql_SessionFactory.close();
46. }
47.
48. }
49.}
```



Is it possible to run hibernate applications without using mapping file?

ANS:

Yes, it is possible to run hibernate applications without using mapping file, but, we have to use Annotations.

#### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

To provide mapping details in Hibernate applications we have to use the following annotations provided by JPA in the form of "javax.persistence" package.

### 1. @Entity:

It is a class level annotation, it can be used to declare a class as an Entity class.

### 2. @Table(name="--")

This annotation is a class level annotation, it can be used to configure table name for the entity class. Where "name" member is able to provide the respective table name.

### 3. @Id :

This annotation is Field/Method[getXXX()] level annotation, it can be used to declare a property as ID property.

### 4. @Column(name="--")

This annotation is Field/Method[getXXX()] level annotation, it can be used to provide a database table column name inorder to provide mapping .

- 1) In Hibernate Applications, to use annotations we have to use the following steps.
- 2) Remove mapping file and provide annotations ion POJO class.
- 3) In Configuration File , provide <mapping> tag with "class" attribute inplace of "resource" attribute."class" attribute will take fully qualified name of the respective POJO class.
- 4) In Client Application, Create Either AnnotationConfiguration object or Configuration object to process Annotations.

**Note:** AnnotationConfiguration is deprecated class, so it is better to use Configuration class object to process annotations.

### Ex: Employee.java

```
1. @Entity
2. @Table(name="emp1")
3. public class Employee{
4. @Id
5. @Column(name="ENO")
6. private int eno;
7. @Column(name="ENAME")
8. private String ename;
9. @Column(name="ESAL")
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10. private float esal;
11. @Column(name="EADDR")
12. private String eaddr;
13. setXXX() and getXXX()
14. }
```

### hibernate.cfg.xml

```

1. <hibernate-configuration>
2. ---
3. <mapping class="com.durgasoft.hbn.pojo.Employee"/>
4. ---
5. </hibernate-Configuration>
```

### ClientApp.java:

Same as previous applications.

### Example:

### Employee.java

```

1. package com.durgasoft.hbn.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp1")
10. public class Employee {
11. @Id
12. @Column(name="ENO")
13. private int eno;
14. @Column(name="ENAME")
15. private String ename;
16. @Column(name="ESAL")
17. private float esal;
18. @Column(name="EADDR")
19. private String eaddr;
20. }
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

21. /*
22. @Id
23. @Column(name="ENO")
24. */
25. public int getEno() {
26. return eno;
27. }
28. public void setEno(int eno) {
29. this.eno = eno;
30. }
31. // @Column(name="ENAME")
32. public String getEname() {
33. return ename;
34. }
35. public void setEname(String ename) {
36. this.ename = ename;
37. }
38. //@Column(name="ESAL")
39. public float getEsal() {
40. return esal;
41. }
42. public void setEsal(float esal) {
43. this.esal = esal;
44. }
45. //@Column(name="EADDR")
46. public String getEaddr() {
47. return eaddr;
48. }
49. public void setEaddr(String eaddr) {
50. this.eaddr = eaddr;
51. }
52.
53.
54.}

```

**hibernate.cfg.xml**

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>

```

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

6. <session-factory>
7. <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8. <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9. <property name="connection.user">system</property>
10. <property name="connection.password">durga</property>
11. <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
12. <property name="show_sql">true</property>
13. <mapping class="com.durgasoft.hbn.pojo.Employee"/>
14.</session-factory>
15.</hibernate-configuration>
```

### ClientApp.java

```

1. package com.durgasoft.hbn.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.cfg.AnnotationConfiguration;
6. import org.hibernate.cfg.Configuration;
7.
8. import com.durgasoft.hbn.pojo.Employee;
9.
10. public class ClientApp {
11.
12. public static void main(String[] args) {
13. SessionFactory sessionFactory = null;
14. Session session = null;
15. try {
16. //Configuration cfg = new AnnotationConfiguration();
17. Configuration cfg = new Configuration();
18. cfg.configure();
19. sessionFactory = cfg.buildSessionFactory();
20. session = sessionFactory.openSession();
21. Employee emp = (Employee)session.get(Employee.class, 111);
22. if(emp == null) {
23. System.out.println("Employee Not Existed");
24. }else {
25. System.out.println("Employee Details");
26. System.out.println("=====");
27. System.out.println("Employee Number :" +emp.getEno());
28. System.out.println("Employee Name :" +emp.getEname());
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
29. System.out.println("Employee Salary :" +emp.getEsal());
30. System.out.println("Employee Address :" +emp.getEaddr());
31. }
32. } catch (Exception e) {
33. e.printStackTrace();
34. }finally {
35. session.close();
36. sessionFactory.close();
37.
38. }
39.
40. }
41.
42. }
```

### Is it possible to run Hibernate Applications without using configuration file?

**ANS:**

Yes, it is possible to run Hibernate applications without using configuration file.

In Hibernate Applications, we are able to provide configuration details in the following two approaches.

- 1) Programmatic Approach
- 2) Declarative Approach
  - a) By using properties file
  - b) By Using XML file

#### 1) Programmatic Approach

In Programmatic approach, to provide configuration details, first we have to create Configuration class object then we have to set all hibernate properties to Configuration class object explicitly by using the following method.

```
public void.setProperty(String prop_Name, String prop_Val)
```

To add mapping file name and location to Configuration file we have to use the following method.

```
public void addResource(String mapping_File_Name)
```

#### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**Note:** If we are using annotations in place of mapping file then we have to use the following method to add annotated class.

```
public void addAnnotatedClass(Class class)
```

### Example on without configuration File and with mapping file in Programmatic Approach:

#### Employee.java

```
1. package com.durgasoft.hbn.pojo;
2.
3. public class Employee {
4. private int eno;
5. private String ename;
6. private float esal;
7. private String eaddr;
8.
9. public int getEno() {
10. return eno;
11. }
12. public void setEno(int eno) {
13. this.eno = eno;
14. }
15. public String getEname() {
16. return ename;
17. }
18. public void setEname(String ename) {
19. this.ename = ename;
20. }
21. public float getEsal() {
22. return esal;
23. }
24. public void setEsal(float esal) {
25. this.esal = esal;
26. }
27. public String getEaddr() {
28. return eaddr;
29. }
30. public void setEaddr(String eaddr) {
31. this.eaddr = eaddr;
32. }
33.
34. }
```



#### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

### Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3. "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
7. <id name="eno" column="ENO"/>
8. <property name="ename" column="ENAME"/>
9. <property name="esal" column="ESAL"/>
10. <property name="eaddr" column="EADDR"/>
11. </class>
12. </hibernate-mapping>
```

### ClientApp.java

```

1. package com.durgasoft.hbn.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.cfg.AnnotationConfiguration;
6. import org.hibernate.cfg.Configuration;
7.
8. import com.durgasoft.hbn.pojo.Employee;
9.
10. public class ClientApp {
11.
12. public static void main(String[] args) {
13. SessionFactory sessionFactory = null;
14. Session session = null;
15. try {
16. // Configuration cfg = new AnnotationConfiguration();
17. Configuration cfg = new Configuration();
18. cfg.setProperty("hibernate.connection.driver_Class", "oracle.jdbc.OracleDriver");
19. cfg.setProperty("hibernate.connection.url", "jdbc:oracle:thin:@localhost:1521:xe");
20. cfg.setProperty("hibernate.connection.username", "system");
21. cfg.setProperty("hibernate.connection.password", "durga");
22. cfg.setProperty("hibernate.dialect", "org.hibernate.dialect.OracleDialect");
23. cfg.addResource("Employee.hbm.xml");
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

24. //cfg.addAnnotatedClass(com.durgasoft.hbn.pojo.Employee.class);
25. sessionFactory = cfg.buildSessionFactory();
26. session = sessionFactory.openSession();
27. Employee emp = (Employee) session.get(Employee.class, 222);
28. if (emp == null) {
29. System.out.println("Employee Not Existed");
30. } else {
31. System.out.println("Employee Details");
32. System.out.println("=====");
33. System.out.println("Employee Number :" + emp.getEno());
34. System.out.println("Employee Name :" + emp.getEname());
35. System.out.println("Employee Salary :" + emp.getEsal());
36. System.out.println("Employee Address :" + emp.getEaddr());
37. }
38. } catch (Exception e) {
39. e.printStackTrace();
40. } finally {
41.
42. session.close();
43. sessionFactory.close();
44.
45. }
46. }
47.

```

Example on without configuration file and without mapping file in Programmatic approach:

**Employee.java**

```

1. package com.durgasoft.hbn.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp1")
10. public class Employee {
11. @Id
12. @Column(name="ENO")
13. private int eno;

```

#### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
14. @Column(name="ENAME")
15. private String ename;
16. @Column(name="ESAL")
17. private float esal;
18. @Column(name="EADDR")
19. private String eaddr;
20.
21. public int getEno() {
22. return eno;
23. }
24. public void setEno(int eno) {
25. this.eno = eno;
26. }
27. public String getEname() {
28. return ename;
29. }
30. public void setEname(String ename) {
31. this.ename = ename;
32. }
33. public float getEsal() {
34. return esal;
35. }
36. public void setEsal(float esal) {
37. this.esal = esal;
38. }
39. public String getEaddr() {
40. return eaddr;
41. }
42. public void setEaddr(String eaddr) {
43. this.eaddr = eaddr;
44. }
45.
46.
47.}
```

**ClientApp.java**

```
1. package com.durgasoft.hbn.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. import org.hibernate.cfg.AnnotationConfiguration;
6. import org.hibernate.cfg.Configuration;
7.
8. import com.durgasoft.hbn.pojo.Employee;
9.
10. public class ClientApp {
11.
12. public static void main(String[] args) {
13. SessionFactory sessionFactory = null;
14. Session session = null;
15. try {
16. // Configuration cfg = new AnnotationConfiguration();
17. Configuration cfg = new Configuration();
18. cfg.setProperty("hibernate.connection.driver_Class", "oracle.jdbc.OracleDriver");
19. cfg.setProperty("hibernate.connection.url", "jdbc:oracle:thin:@localhost:1521:xe");
20. cfg.setProperty("hibernate.connection.username", "system");
21. cfg.setProperty("hibernate.connection.password", "durga");
22. cfg.setProperty("hibernate.dialect", "org.hibernate.dialect.OracleDialect");
23. //cfg.addResource("Employee.hbm.xml");
24. cfg.addAnnotatedClass(com.durgasoft.hbn.pojo.Employee.class);
25. sessionFactory = cfg.buildSessionFactory();
26. session = sessionFactory.openSession();
27. Employee emp = (Employee) session.get(Employee.class, 222);
28. if (emp == null) {
29. System.out.println("Employee Not Existed");
30. } else {
31. System.out.println("Employee Details");
32. System.out.println("=====");
33. System.out.println("Employee Number :" + emp.getEno());
34. System.out.println("Employee Name :" + emp.getEname());
35. System.out.println("Employee Salary :" + emp.getEsal());
36. System.out.println("Employee Address :" + emp.getEaddr());
37. }
38. } catch (Exception e) {
39. e.printStackTrace();
40. } finally {
41. session.close();
42. sessionFactory.close();
43. }
44. }
45.
46. }
47. }
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In Programmatic approach, if we want to change configuration details like connection properties or database properties,... then we have to perform modifications in JAVA code, if we perform modifications in JAVA code then we must recompile the java application , it is not suggestible in enterprise applications. To overcome the problem we have to use Declarative approach.

## 2) Declarative approach:

In Declarative approach, we will provide all configuration details in either properties file or in XML file then we will send configuration details to Hibernate application.

There are two ways to provide configuration details to Hibernate Applications in Declarative Approach.

- 1) By using properties file.
- 2) By Using XML file.

### 1) Using properties file in Declarative Approach:

In This approach, to provide all hibernate configuration details , we have to declare a properties file with the default name "hibernate.properties" under "src" folder . In this context, when we create Configuration class object , Hibernate Software will search for the properties file with the name "hibernate.properties" and get all the configuration details into Configuration class object.

In this approach we must add mapping file explicitly to Configuration File by using the following method.

```
public void addResource(String mapping_File-Name)
```

**Example:**

**hibernate.properties:**

```
hibernate.connection.driver_Class = oracle.jdbc.OracleDriver
hibernate.connection.url = jdbc:oracle:thin:@localhost:1521:xe
hibernate.connection.username = system
hibernate.connection.password = durga
hibernate.dialect = org.hibernate.dialect.OracleDialect
```

**ClientApp.java**

```
1. class ClientApp{
2. public static void main(String[] args){
3. Configuration cfg = new Configuration();
4. cfg.addResource("Employee.hbm.xml");
```

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. ----
6. }
7. }
```

**Note:** If we change name and location of properties file from hibernate.properties to some other abc.properties then we have to give that intemation to Hibernate software, for this, we have to create FileInputStream and Properties class object then we have to set Properties object to Configuration class object by using the following method.

```
public void setProperties(Properties p)
```

**Example:**

**Example to provide configuration details to Hibernate by using properties file:**

**abc.properties:**

```
hibernate.connection.driver_Class = oracle.jdbc.OracleDriver
hibernate.connection.url = jdbc:oracle:thin:@localhost:1521:xe
hibernate.connection.username = system
hibernate.connection.password = durga
hibernate.dialect = org.hibernate.dialect.OracleDialect
```

**Employee.hbm.xml**

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3. "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
7. <id name="eno" column="ENO"/>
8. <property name="ename" column="ENAME"/>
9. <property name="esal" column="ESAL"/>
10. <property name="eaddr" column="EADDR"/>
11. </class>
12. </hibernate-mapping>
```

**Employee.java**

```
1. package com.durgasoft.hbn.pojo;
2. public class Employee {
3. private int eno;
```

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

4. private String ename;
5. private float esal;
6. private String eaddr;
7.
8. public int getEno() {
9. return eno;
10. }
11. public void setEno(int eno) {
12. this.eno = eno;
13. }
14. public String getEname() {
15. return ename;
16. }
17. public void setEname(String ename) {
18. this.ename = ename;
19. }
20. public float getEsal() {
21. return esal;
22. }
23. public void setEsal(float esal) {
24. this.esal = esal;
25. }
26. public String getEaddr() {
27. return eaddr;
28. }
29. public void setEaddr(String eaddr) {
30. this.eaddr = eaddr;
31. }
32.
33.
34.}
```

### ClientApp.java

```

1. package com.durgasoft.hbn.test;
2.
3. import java.io.FileInputStream;
4. import java.util.Properties;
5.
6. import org.hibernate.Session;
7. import org.hibernate.SessionFactory;
8. import org.hibernate.cfg.AnnotationConfiguration;
9. import org.hibernate.cfg.Configuration;
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10.
11. import com.durgasoft.hbn.pojo.Employee;
12.
13. public class ClientApp {
14.
15. public static void main(String[] args) {
16. SessionFactory sessionFactory = null;
17. Session session = null;
18. try {
19. // Configuration cfg = new AnnotationConfiguration();
20. Configuration cfg = new Configuration();
21. FileInputStream fis = new FileInputStream("D:\\hibernate3\\eclipse_hibernate\\app1
2\\src\\abc.properties");
22. Properties p = new Properties();
23. p.load(fis);
24. cfg.setProperties(p);
25. cfg.addResource("Employee.hbm.xml");
26. //cfg.addAnnotatedClass(com.durgasoft.hbn.pojo.Employee.class);
27. sessionFactory = cfg.buildSessionFactory();
28. session = sessionFactory.openSession();
29. Employee emp = (Employee) session.get(Employee.class, 333);
30. if (emp == null) {
31. System.out.println("Employee Not Existed");
32. } else {
33. System.out.println("Employee Details");
34. System.out.println("-----");
35. System.out.println("Employee Number :" + emp.getEno());
36. System.out.println("Employee Name :" + emp.getEname());
37. System.out.println("Employee Salary :" + emp.getEsal());
38. System.out.println("Employee Address :" + emp.getEaddr());
39. }
40. } catch (Exception e) {
41. e.printStackTrace();
42. } finally {
43. session.close();
44. sessionFactory.close();
45.
46. }
47.
48. }
49.
50. }
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ❖ In hibernate applications, if we use "properties" file then we are able to provide only hibernate connection properties, dialect properties,... , but, we are unable to provide mapping properties,... through properties file, to provide mapping properties we have to use java methods like addResource(--) or addAnnotatedClass(--) methods from Configuration class.
- ❖ In Hibernate applications, if we want to provide all configuration details in declarative manner then we have to use XML file approach.
- ❖ If we want to use xml file to provide all configuration details then we have to use configure() method to get all configuration details from xml file. Here configure() method will search for xml file with the default name hibernate.cfg.xml under src folder inorder to get configuration details. If we change the default name and location of hibernate configuration file then we have to pass that name and location to configure(--) method as parameter.

### Composite Keys:

In Database applications, if one column is not sufficient to manage uniqueness as primary key then it is possible to declare more than one column combination as primary key, such type of column combination as primary key is called as "Composite Key".

```
sql>create table emp1(ENO number, ENAME varchar2(10), ESAL float, EADDR varchar2(10),
primary key(ENO,ENAME));
sql>commit
```

To represent Composite keys in Hibernate applications we have to use the following xml tags in mapping file.

```
1. <hibernate-mapping>
2. <class name="--" table="--">
3. <composite-id>
4. <key-property name="--" column="--"/>
5. -----
6. </composite-id>
7. -----
8. </class>
9. </hibernate-mapping>
```

- ✓ Where <composite-id> tag is representing composite key configuration in mapping file.
- ✓ Where <key-property> tag is able to represent single ID or column configuration as part of composite key.

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ✓ Where "name" attribute will take id property name and column attribute will take "database table column name".

**EX:Employee.hbm.xml**

```
1. <hibernate-mapping>
2. <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
3. <composite-id>
4. <key-property name="eno" column="ENO"/>
5. <key-property name="ename" column="ENAME"/>
6. </composite-id>
7. <property name="esal" column="ESAL"/>
8. <property name="eaddr" column="EADDR"/>
9. </class>
10.</hibernate-mapping>
11.
12. Note: To declare composite keys in annotated classes , no need to use any separate Annotation, simply we have to declare @Id annotation at both the columns.
13.
14. EX: Employee.java
15. -----
16. @Entity
17. @Table(name="emp1")
18. public class Employee implements Serializable {
19. @Id
20. @Column(name="ENO")
21. private int eno;
22. @Id
23. @Column(name="ENAME")
24. private String ename;
25. @Column(name="ESAL")
26. private float esal;
27. @Column(name="EADDR")
28. private String eaddr;
29. setXXX() and getXXX();
30. }
```

**Note:** To use Composite keys feature in Hibernate applications then it is mandatory to implement java.io.Serializable marker interface to POJO class.

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**Example:****Employee.java**

```
1. package com.durgasoft.hbn.pojo;
2.
3. import java.io.Serializable;
4.
5. import javax.persistence.Column;
6. import javax.persistence.Entity;
7. import javax.persistence.Id;
8. import javax.persistence.Table;
9. @Entity
10. @Table(name="emp1")
11. public class Employee implements Serializable {
12. @Id
13. @Column(name="ENO")
14. private int eno;
15. @Id
16. @Column(name="ENAME")
17. private String ename;
18. @Column(name="ESAL")
19. private float esal;
20. @Column(name="EADDR")
21. private String eaddr;
22.
23. public int getEno() {
24. return eno;
25. }
26. public void setEno(int eno) {
27. this.eno = eno;
28. }
29. public String getEname() {
30. return ename;
31. }
32. public void setEname(String ename) {
33. this.ename = ename;
34. }
35. public float getEsal() {
36. return esal;
37. }
38. public void setEsal(float esal) {
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

39. this.esal = esal;
40. }
41. public String getEaddr() {
42. return eaddr;
43. }
44. public void setEaddr(String eaddr) {
45. this.eaddr = eaddr;
46. }
47.
48.
49.

```

### Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3. "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
7. <composite-id>
8. <key-property name="eno" column="ENO"/>
9. <key-property name="ename" column="ENAME"/>
10. </composite-id>
11. <property name="esal" column="ESAL"/>
12. <property name="eaddr" column="EADDR"/>
13. </class>
14. </hibernate-mapping>

```

### hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8. <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9. <property name="connection.user">system</property>
10. <property name="connection.password">durga</property>
11. <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>

```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
12. <property name="show_sql">true</property>
13. <!-- <mapping resource="Employee.hbm.xml"/> -->
14. <mapping class="com.durgasoft.hbn.pojo.Employee"/>
15. </session-factory>
16. </hibernate-configuration>
```

### ClientApp.java



```
1. package com.durgasoft.hbn.test;
2.
3.
4. import org.hibernate.Session;
5. import org.hibernate.SessionFactory;
6. import org.hibernate.Transaction;
7. import org.hibernate.cfg.Configuration;
8.
9. import com.durgasoft.hbn.pojo.Employee;
10.
11. public class ClientApp {
12.
13. public static void main(String[] args) {
14. SessionFactory sessionFactory = null;
15. Session session = null;
16. try {
17.
18. Configuration cfg = new Configuration();
19. cfg.configure();
20. sessionFactory = cfg.buildSessionFactory();
21. session = sessionFactory.openSession();
22.
23. Employee emp1 = new Employee();
24. emp1.setEno(222);
25. emp1.setEname("BBB");
26. emp1.setEsal(8000);
27. emp1.setEaddr("Hyd");
28. Transaction tx = session.beginTransaction();
29. session.save(emp1);
30. tx.commit();
31. System.out.println("Employee Inserted Successfully");
32.
33. /*
34. Employee emp_Key = new Employee();
35. emp_Key.setEno(222);
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
36. emp_Key.setEname("XXX");
37. Employee emp = (Employee) session.get(Employee.class, emp_Key);
38. if (emp == null) {
39. System.out.println("Employee Not Existed");
40. } else {
41. System.out.println("Employee Details");
42. System.out.println("=====");
43. System.out.println("Employee Number :" + emp.getEno());
44. System.out.println("Employee Name :" + emp.getEname());
45. System.out.println("Employee Salary :" + emp.getEsal());
46. System.out.println("Employee Address :" + emp.getEaddr());
47. }
48. */
49.
50. } catch (Exception e) {
51. e.printStackTrace();
52. } finally {
53. session.close();
54. sessionFactory.close();
55.
56. }
57. }
58. }
```

DURGA

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

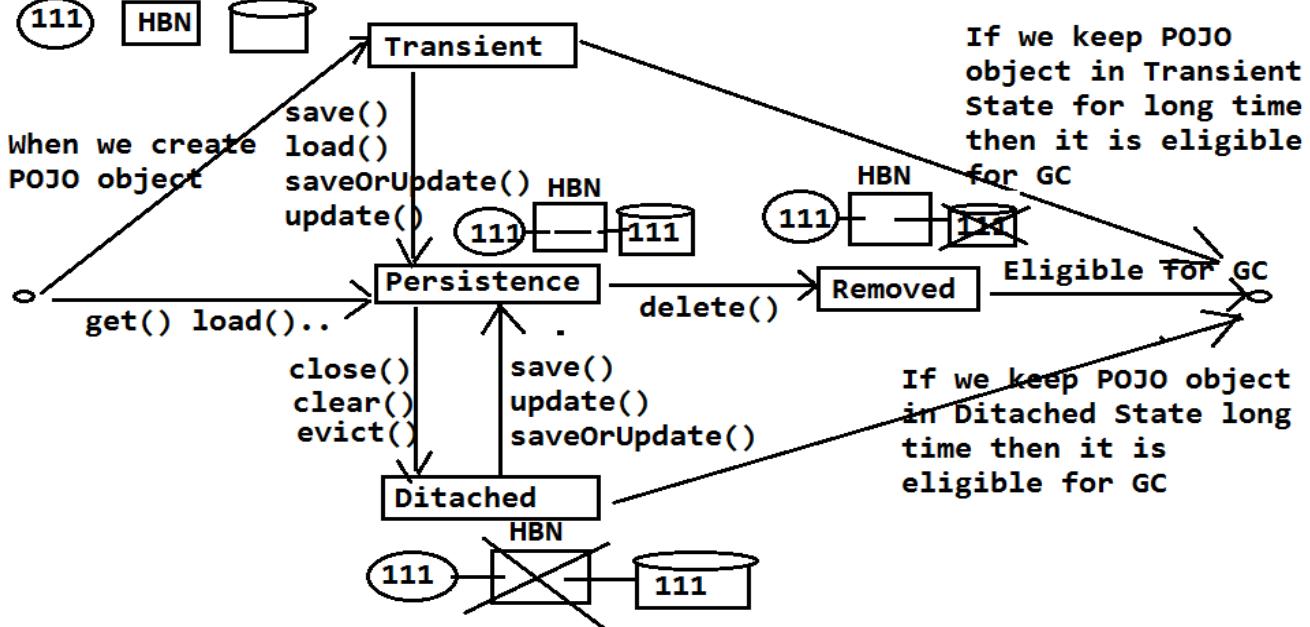


BY NAGOOR BABU

## Hibernate Persistence Object Lifecycle

The collective information of any object right from starting point to end point is called as Object Lifecycle.

In Hibernate Applications, Persistence object is having the following lifecycle states.



1. Transient State
2. Persistence State
3. Detached State
4. Removed State

### 1. Transient State

- ❖ In Hibernate Applications, when we create POJO object then automatically POJO object will come to Transient State.
- ❖ If POJO object is available in Transient State then Hibernate Software is not aware about POJO object.

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ❖ If POJO object is available in Transient State then POJO object is representing any record in Database table.
- ❖ In this state, Hibernate Software is unable to provide synchronization between POJO object and record in Database.
- ❖ In this state, if we perform modifications in POJO object then that modifications are not reflected to any record in Database.
- ❖ If we keep POJO object in Transient State for long time then POJO object is eligible for Garbage Collection then Garbage Collector will destroy POJO object.

## 2. Persistence State:

- ❖ If we perform the Operations like save() , update(), saveOrUpdate(),.... over the POJO object which is available in Transient State then POJO object will come to Persistence State.
- ❖ In Hibernate Applications, if we perform the objects like get() , load(), find(),... automatically Hibernate Software will create POJO object in Persistence State directly without Transient State.
- ❖ In Persistence State, Hibernate Software is aware about POJO Object and database table record and its synchronization.
- ❖ In Persistence State, if we perform modifications on POJO object then that modifications are reflected to Database table record.

## 3. Detached State:

- ❖ If we perform the operations like close(), clear(), evict(),.... over the Session object then POJO object will come from Persistence State to Detached State.
- ❖ In Detached State, Hibernate Software is not aware about POJO object and it will not provide synchronization between POJO object and database table record.
- ❖ In Detached State, POJO object is representing valid record in database table,but, modifications on POJO object are not reflected to that record.

### CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ❖ In Detached State, if we perform operations like save() , update(), saveOrUpdate(),... over the persistence object after getting Session back then POJO object will transfer from Detached State to Persistence State again.
- ❖ If we keep POJO object in Detached State for long time then POJO object is eligible for Garbage Collection, where Garbage Collector will destroy that object.

#### 4. Removed State:

- ❖ If we perform the operations like remove() or delete() over the POJO object in Persistence state then POJO object will come from Persistence State to Removed State.
- ❖ In Removed State, Hibernate Software is not aware about POJO object.
- ❖ In Removed State, POJO object is not representing any record in Database table.
- ❖ In Hibernate Applications, Once if POJO object is in Removed State then it is not possible to get back into Persistence State, directly, it must go for Garbage Collection only.

##### Example:

```
Employee emp = new Employee(); // Transient State
```

```

session.save(emp); // Persistence State

```

```


session.delete(emp); // Removed State
```

##### Example:

```
Account acc = new Account(); // Transient State
```

```


session.save(acc); // Persistence State

```

```


session.close(); // Detached State

```

```


sessionFactory.getCurrentSession(); or sf.openSession();
```

```


session.update(acc); // Persistence
Hibernate Schema Generation Tools:
```

#### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

## Hibernate Schema Generation Tools:

In Hibernate applications, to perform DML operations like insert, update, delete ,... we will use the methods like save(), update(), delete(),... from org.hibernate.Session interface provided by Hibernate Software.

In Hibernate applications, to perform DDL operations like create, alter, drop,... Hibernate has provided the following implicit tools.

1. SchemaExport
2. SchemaUpdate
3. CodeGeneration

Note: The above Shema generation tools are usefull to create tables when we want to work with unknown databases.

### 1. SchemaExport

This tool is provided by Hibernate software in the form of org.hibernate.tool.hbm2ddl.SchemaExport predefined class in hibernate3.jar file.

If we activate this tool then it will perform the following actions.

- a) First SchemaExport tool will take name and location of hibernate configuration file which we have provided as input parameter.
- b) SchemaExport tool will load and parse hibernate configuration file.
- c) SchemaExport tool will recognize mapping file configuration in hibernate configuration file then it will load and parse mapping file.
- d) As per mapping file details, SchemaExport tool will check whether any table is existed or not in Database.
- e) If no table is existed then SchemaExport tool will create new table as per mapping file details.
- f) If any table is existed with the name then SchemaExport tool will drop that existed table and creates new table as per mapping file configuration details.

To activate SchemaExport tool manually through command prompt then we have to use the following command on command prompt.

D:\apps>java org.hibernate.tool.hbm2ddl.SchemaExport --config =hibernate.cfg.xml

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**Note:** To run the above command we must copy all hibernate jars to "C:\Java\jdk1.7.0\jre\lib\ext" folder and we must provide POJO class, mapping file and configuration file at the same location from where we are activating SchemaExport tool.

**Example:****Customer.java**

```
1. public class Customer
2. {
3. private String cid;
4. private String cname;
5. private String caddr;
6. private String cemail;
7.
8. public void setCid(String cid)
9. {
10. this.cid = cid;
11. }
12. public void setCname(String cname)
13. {
14. this.cname = cname;
15. }
16. public void setCaddr(String caddr)
17. {
18. this.caddr = caddr;
19. }
20. public void setCemail(String cemail)
21. {
22. this.cemail = cemail;
23. }
24.
25. public String getCid()
26. {
27. return cid;
28. }
29. public String getCname()
30. {
31. return cname;
32. }
33. public String getCaddr()
34. {
35. return caddr;
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

36. }
37. public String getCemail()
38. {
39. return cemail;
40. }
41.

```

### Customer.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3. "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="Customer" table="customer">
7. <id name="cid" column="CID" length="5" type="string"/>
8. <property name="cname" column="CNAME" length="10" type="string"/>
9. <property name="caddr" column="CADDR" length="10" type="string"/>
10. <property name="cemail" column="CEMAIL" length="20" type="string"/>
11. </class>
12. </hibernate-mapping>

```

### hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8. <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9. <property name="connection.user">system</property>
10. <property name="connection.password">durga</property>
11. <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
12. <property name="show_sql">true</property>
13. <mapping resource="Customer.hbm.xml"/>
14. </session-factory>
15. </hibernate-configuration>

```

### Command on Command prompt:

D:\apps>javac Customer.java

D:\apps>java org.hibernate.tool.hbm2ddl.SchemaExport --config =hibernate.cfg.xml

### CONTACT US:

**Mobile: +91- 8885 25 26 27**

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

## 2. SchemaUpdate

This tool is provided by Hibernate software in the form of org.hibernate.tool.hbm2ddl.SchemaUpdate predefined class in hibernate3.jar file.

If we activate this tool then it will perform the following actions.

- First SchemaUpdate tool will take name and location of hibernate configuration file which we have provided as input parameter.
- SchemaUpdate tool will load and parse hibernate configuration file.
- SchemaUpdate tool will recognize mapping file configuration in hibernate configuration file then it will load and parse mapping file.
- As per mapping file details, SchemaUpdate tool will check whether any table is existed or not in Database with the same name.
- If no table is existed then SchemaUpdate tool will create new table as per mapping file details.
- If any table is existed with the name then SchemaUpdate tool will alter that existed table as per mapping file configuration details.

**Note:** When mapping contains new columns details when compared with database table then only SchemaUpdate tool will perform alter operation. If Database table contains more columns when compared with mapping file configuration details then SchemaUpdate tool will not perform alter operation.

To activate SchemaUpdate tool manually through command prompt then we have to use the following command on command prompt.

D:\apps>java org.hibernate.tool.hbm2ddl.SchemaUpdate --config =hibernate.cfg.xml

**Note:** To run the above command we must copy all hibernate jars to "C:\Java\jdk1.7.0\jre\lib\ext" folder and we must provide POJO class, mapping file and configuration file at the same location from where we are activating SchemaUpdate tool.

### Customer.java

```

1. public class Customer
2. {
3. private String cid;
4. private String cname;
5. private String caddr;
6. private String cemail;
7.

```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

8. public void setCid(String cid)
9. {
10. this.cid = cid;
11. }
12. public void setCname(String cname)
13. {
14. this.cname = cname;
15. }
16. public void setCaddr(String caddr)
17. {
18. this.caddr = caddr;
19. }
20. public void setCemail(String cemail)
21. {
22. this.cemail = cemail;
23. }
24.
25. public String getCid()
26. {
27. return cid;
28. }
29. public String getCname()
30. {
31. return cname;
32. }
33. public String getCaddr()
34. {
35. return caddr;
36. }
37. public String getCemail()
38. {
39. return cemail;
40. }
41.

```

### **Customer.hbm.xml**

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3. "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="Customer" table="customer">

```

### **CONTACT US:**

**Mobile:** +91- 8885 25 26 27

+91- 7207 21 24 27/28



**US NUM:** 4433326786

**Mail ID:** [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

**WEBSITE:** [www.durgasoftonline.com](http://www.durgasoftonline.com)

**FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

```

7. <id name="cid" column="CID" length="5" type="string"/>
8. <property name="cname" column="CNAME" length="10" type="string"/>
9. <property name="caddr" column="CADDR" length="10" type="string"/>
10. <property name="cemail" column="CEMAIL" length="20" type="string"/>
11. </class>
12.</hibernate-mapping>
```

### hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8. <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9. <property name="connection.user">system</property>
10. <property name="connection.password">durga</property>
11. <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
12. <property name="show_sql">true</property>
13. <mapping resource="Customer.hbm.xml"/>
14. </session-factory>
15. </hibernate-configuration>
```

### 3. CodeGeneration:

The main intention of CodeGeneration tool is to create POJO class as per database table configurations which we provided in mapping file.

**Note:** This is available upto Hibernate2.x version, but, it is deprecated in Hibernate3.x version.

In Hibernate applications, we are able to activate SchemaExport and SchemaUpdate tools in declarative manner also, for this, we have to use the following property in hibernate configuration file.

hibernate.hbm2ddl.auto

The above property will take the following three values.

1. create
2. update
3. create-drop

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**1. Create:** This option for "hibernate.hbm2ddl.auto" property will activate SchemaExport tool, it will create a table irrespective of the table is existed or not.

**EX:**

```
1. <hibernate-configuration>
2. ---
3. <property name="hibernate.hbm2ddl.auto">create</property>
4. ---
5. </hibernate-configuration>
```

**2. Update:** This option for "hibernate.hbm2ddl.auto" property will activate SchemaUpdate tool, it will create table if table is not existed, it will alter the table if table is existed as per the mapping file.

**EX:**

```
1. <hibernate-configuration>
2. ---
3. <property name="hibernate.hbm2ddl.auto">update</property>
4. ---
5. </hibernate-configuration>
```

**3. Create-Drop:** This option for "hibernate.hbm2ddl.auto" will create table when SessionFactory object is created and it will drop table when SessionFactory object is closed.

**EX:**

```
1. <hibernate-configuration>
2. ---
3. <property name="hibernate.hbm2ddl.auto">create-drop</property>
4. ---
5. </hibernate-configuration>
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

## Creating SessionFactory object in Hibernate4.x version:

In Hibernate4.x version, `cfg.buildSessionFactory()` method was deprecated, because, to load hibernate software and to setup Hibernate environment it may take lot of time. To simplify Hibernate Software loading and to set up Hibernate Environment in simplified manner, Hibernate4.x version has included some booting features .

In Hibernate4.x version, to create SessionFactory object we have to use the following steps.

### a) Create Configuration Object with all configuration details:

```
Configuration cfg = new Configuration();
cfg.configure();
```

### b) Get all configuration details into Properties object:

```
public Properties.getProperties()
EX: Properties p = cfg.getProperties();
```

### c) Creating StandardServiceRegistryBuilder:

```
StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
```

### d) Get All configuration details from Properties object into StandardServiceRegistryBuilder object:

```
public StandardServiceRegistryBuilder applySettings(Map m)
EX: builder = builder.applySettings(p);
```

### e) Create StandardServiceRegistry object:

```
public StandardServiceRegistry build()
EX: StandardServiceRegistry registry = builder.build();
```

### f) Create SessionFactory object:

```
public SessionFactory buildSessionFactory(StandardServiceRegistry registry)
```

**EX:** Sessionfactory sf = cfg.buildSessionFactory(registry);

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**Example:****Employee.java**

```
1. package com.durgasoft.hbn.pojo;
2.
3. public class Employee {
4. private int eno;
5. private String ename;
6. private float esal;
7. private String eaddr;
8.
9. public int getEno() {
10. return eno;
11. }
12. public void setEno(int eno) {
13. this.eno = eno;
14. }
15. public String getEname() {
16. return ename;
17. }
18. public void setEname(String ename) {
19. this.ename = ename;
20. }
21. public float getEsal() {
22. return esal;
23. }
24. public void setEsal(float esal) {
25. this.esal = esal;
26. }
27. public String getEaddr() {
28. return eaddr;
29. }
30. public void setEaddr(String eaddr) {
31. this.eaddr = eaddr;
32. }
33.
34.
35. }
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

### Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3. "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="com.durgasoft.hbn.pojo.Employee" table="emp1">
7. <id name="eno"/>
8. <property name="ename"/>
9. <property name="esal"/>
10. <property name="eaddr"/>
11. </class>
12. </hibernate-mapping>
```

### hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8. <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9. <property name="connection.username">system</property>
10. <property name="connection.password">durga</property>
11. <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
12. <property name="show_sql">true</property>
13. <mapping resource="Employee.hbm.xml"/>
14. </session-factory>
15. </hibernate-configuration>
```

### ClientApp.java

```

1. package com.durgasoft.hbn.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.boot.registry.StandardServiceRegistry;
6. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
7. import org.hibernate.cfg.Configuration;
8.
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9. import com.durgasoft.hbn.pojo.Employee;
10.
11. public class ClientApp {
12.
13. public static void main(String[] args) throws Exception {
14. Configuration cfg = new Configuration();
15. cfg.configure();
16. StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
17. builder = builder.applySettings(cfg.getProperties());
18. StandardServiceRegistry registry = builder.build();
19. SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
20. Session session = sessionFactory.openSession();
21. Employee emp = (Employee)session.get("com.durgasoft.hbn.pojo.Employee", 111);
22. if(emp == null) {
23. System.out.println("Employee Not Existed");
24. }else {
25. System.out.println("Employee Details");
26. System.out.println("-----");
27. System.out.println("Employee Number :" +emp.getEno());
28. System.out.println("Employee Name : " +emp.getEname());
29. System.out.println("Employee Salary : " +emp.getEsal());
30. System.out.println("Employee Address : " +emp.getEaddr());
31. }
32. session.close();
33. sessionFactory.close();
34. }
35.
36. }
```

DURGA

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

## Primary Key Generation Algorithms in Hibernate

- Primary key is single column or Collection of columns in a table to recognize the records individually.
- In Database applications, to perform the database operations like retrieving a record, updating a record, deleting a record,...we need primary key and its value.
- In Database applications, we are unable to give option to the users to enter primary key values , because, there is no guarantee for the data entered by the users whether it is unique data or not, but, in Database tables only unique values are accepted by primary key columns.
- To give guarantee for uniqueness in primary key values we have to use Primary key generation algorithms.
- Almost all the Persistence mechanisms like Hibernate, JPA, Open JPA, Toplink,... are having their own implementation for primary key generation algorithms.

Hibernate has provided support for the following primary key generation algorithms inorder to generate primary key values.

1. assigned
2. increment
3. sequence
4. identity
5. hilo
6. native.
7. seq-hilo
8. select
9. UUID
10. GUID
11. foreign

Hibernate has represented all the above primary key generation algorithms in the form of a set of predefined classes provided in

"org.hibernate.id" package.

If we want to use any of the above algorithms in Hibernate applications then we have to configure that algorithms in hibernate mapping file by using the following tags.

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

1. <hibernate-mapping>
2. <class name="--" table="--">
3. <id name="--" column="--">
4. <generator class="--">
5. <param name="-->value</param>
6. -----
7. </generator>
8. </id>
9. -----
10. </class>
11.</hibernate-mapping>
```

- ✓ Where <generator> tag will represent a particular primary key generation algorithm.
- ✓ Where "class" attribute in <generator> tag will take either short name or Fully qualified name of Generator class.
- ✓ Where <param> tag will provide single input parameter to the Primary key generator inorder to generate primary key value.
- ✓ Where "name" attribute in <param> tag will take parameter name , here we have to provide parameter value as body to the <param> tag.

### 1. assigned

- ⊕ This algorithm is default primary key generation algorithm in hibernate applications, it will not required configurations in mapping file.
- ⊕ This algorithm will not have its own mechanism to generate primary key value , it will request to Client Application to provide primary key value explicitly.
- ⊕ This algorithm is able to support for any type of primary key values like short, int, long, String,.....
- ⊕ This alg is supported by almost all the databases like Oracle, MySQL, DB2, .....
- ⊕ This alg is not required any input parameter.
- ⊕ This alg is represented by Hibernate in the form of a predefined class "org.hibernate.id.Assigned".

EX:

```

1. <class name="Employee" table="emp1">
2. <id name="eno" column="ENO">
3. <generator class="assigned"/>
4. </id>
5. -----
6. </class>
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

## 2. increment

- + This primary key generation algorithm is able to generate primary key value by incrementing max value of the primary key column.  
New\_Val = max(PK\_Column)+1
- + This alg is able to generate primary key values of the data types like short, int, long,...
- + This alg is not required any input parameter to generate primary key values.
- + This alg is supported by almost all the databases which are supporting numeric values as Primary key values.
- + This alg is represented by Hibernate Software in the form of a short name "increment" and in the form of a predefined class like "org.hibernate.id.IncrementGenerator".

EX:

```

1. <hibernate-mapping>
2. <class name="Employee" table="emp1">
3. <id name="eno" column="ENO">
4. <generator class="org.hibernate.id.IncrementGenerator"/>
5. </id>
6. ---
7. </class>
8. </hibernate-mapping>
```

## 3. sequence

- + This primary key generation algorithm is able to generate primary key value on the basis of the sequence provided by the underlying Database.
- + This alg is able to generate primary key values of the data types like short, int, long,...
- + This alg required "sequence" input parameter with the sequence name as value inorder to generate primary key value.
- + If we have not provided any sequence name as input parameter then this alg is able to take "hibernate\_sequence" as default sequence name, here developers must create "hibernate\_sequence" in database explicitly.
- + This alg is supported by almost all the databases which are supporting sequences like Oracle, DB2,....
- + This alg is represented by Hibernate Software in the form of a short name like "sequence" and in the form of a predefined class like "org.hibernate.id.SequenceGenerator".

EX:

In Database:

SQL>create sequence my\_sequence increment by 5;  
 SQL>commit;

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

### In mapping File:

```

1. <hibernate-mapping>
2. <class name="Employee" table="emp1">
3. <id name="eno" column="ENO">
4. <generator class="sequence">
5. <param name="sequence">my_sequence</param>
6. </generator>
7. </id>
8. -----
9. </class>
10.</hibernate-mapping>
```

**Note:** If we dont want to use explicit sequence then we must create default sequence in database , that is, "hibernate\_sequence".

SQL>create sequence hibernate\_sequence increment by 5;

### 4. identity

- + This alg is able to generate primary key values on the basis on the underlying database table provided identity column.
- Note:** Identity column is a primary key Column with "auto\_increment" capability.
- + This alg is able to provide the primary key values of the data types like short, int, long,....
- + This alg is not required any input parameter.
- + This alg is supported by almost all the databases which are supporting Identity column.

**EX:** MySQL.

- + To represent this alg, Hibernate has provided "identity" as short name and "org.hibernate.id.IdentityGenerator" as predefined class.

**EX:**

### IN MySQL database:

sql>create table emp1(ENO primary key auto\_increment, ENAME char(10), ESAL float, EADDR char(10));

### IN Mapping File:

```

1. <hibernate-mapping>
2. <class name="Employee" table="emp1">
3. <id name="eno" column="ENO">
4. <generator class="identity"/>
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

5. </id>
6. </class>
7. ----
8. </hibernate-mapping>
```

## 5. hilo[High-Low]

- This alg is able to generate primary key value by getting high value from a particular global resource and low value from mapping file.

**Note:** Global resource is a particular table with a column in the same database.

- This alg is able to generate primary key values of the data types like short, int, long,....
- To represent this alg. Hibernate software has provides a seperate short in the form of "hilo" and a seperate predefined class in the form of "org.hibernate.id.TableHiLoGenerator".
- This alg is supported by almost all the databases like Oracle, MySQL,....
- To generate primary key value this alg will take the following three parameters.

1. table: Global resource name , that is, a table name which provides high value.
2. column:column name existed in Global Resource to manage high value.
3. max\_lo:It will provide low value .

**Note:** If we have not provided Global resources parameters like table and column in mapping file then Hibernate Software will take high value from default global resource table "hibernate\_unique\_key" and column "next\_hi".

**EX:**

```

1. <hibernate-mapping>
2. <class name="Employee" table="emp1">
3. <id name="eno" column="ENO">
4. <generator class="hilo">
5. <param name="table>my_table</param>
6. <param name="column">my_column</param>
7. <param name="max_lo">10</param>
8. </generator>
9. </id>
10. ---
11. </class>
12. </hibernate-mapping>
13. In Database:
14. SQL>create table my_table(my_column number primary key);
15. SQL>insert into my_table values(100);
```

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

16. SQL&gt;commit;

**Note:** If we want to use default table name and column instead of my\_table and my\_column then we have to remove "table" and "column" parameters in mapping file and provide the following in Database.

```
SQL>create table hibernate_unique_key(next_hi number primary key);
SQL>insert into hibernate_unique_key values(100);
SQL>commit;
```

## 6. native

- + This alg is able to generate primary key value by selecting a particular primary key generation alg depending on the database which we used.
- + This alg is not having its own alg to generate primary key value. It will select "SequenceGenerator" alg if we are using Oracle database, "IdentityGenerator" alg if we are using MySQL database and "TableHiLoGenerator" if we are using some other database which is not supporting SequenceGenerator and IdentityGenerator.
- + This Primary key generator is able to generate primary key values of the data types like short, int, long,...
- + This primary key generator is supported by almost all the databases.
- + To represent this mechanism, Hibernate has provided a short name in the form of "native", Hibernate has not provided any predefined class.
- + This alg is able to take parameters on the basis of the selected primary key generator depending on the database. If it is using "SequenceGenerator" then we must provide "sequence" parameter. If it is using TableHiLoGenerator then we have to provide "table", "column", "max\_lo" parameters.

EX:

```
1. <hibernate-mapping>
2. <class name="Employee" table="emp1">
3. <id name="eno" column="ENO">
4. <generator class="native">
5. ---provide parameters depending on requirement---
6. </generator>
7. </id>
8. </class>
9. </hibernate-mapping>
```

## CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

## 7. seq-hilo

- + This alg is able to generate primary key values on the basis of the sequence value provided by the underlying database and the mapping file provided max\_lo property value.
- + This alg is almost all same as HiLo alg but in "seqhilo" alg we will get sequence value instead of the high value from Global resource.
- + This alg is able to generate primary key values of the data types like short, int and long.
- + This alg is represented by Hibernate software in the form a short name like "seqhilo" and a predefined class name like "org.hibernate.id.SequenceHiLoGenerator".
- + This alg is supported by almost all the databases which are supporting sequences internally.
- + This alg is required the following two input parameters.

1. sequence: It will take sequence name provided by Database.
2. max\_lo : It will provide low value.

**Note:** If we have not provided "sequence" input parameter then hibernate software will search for the default sequence name like "hibernate\_sequence".

## EX:

```

1. <hibernate-mapping>
2. <class name="Employee" table="emp1">
3. <id name="eno" column="ENO">
4. <generator class="seqhilo">
5. <param name="sequence">my_sequence</param>
6. <param name="max_lo">10</param>
7. </generator>
8. </id>
9. ---
10. </class>
11. </hibernate-mapping>
```

## 8. select

- + This alg is able to generate primary key values on the basis of the underlying database provided triggers.
- + This alg is able to provide primary key values of the data types like short, int and long.
- + This alg is represented by Hibernate software in the form of a short name like "select" and a predefined class name "org.hibernate.id.SelectGenerator".
- + This alg may take input parameters which are related to triggers.
- + This alg is supported by almost all the databases which are supporting triggers.

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**EX:**

```

1. <hibernate-mapping>
2. <class name="Employee" table="emp1">
3. <id name="eno" column="ENO">
4. <generator class="select">
5. --- parameters related to triggers---
6. </generator>
7. </id>
8. ---
9. </class>
10.</hibernate-mapping>
```

## 9. UUID[Unicersal Unique Identity]

- + This alg is able to generate primary key value on the basis of System IP-Address , JVM Startup time, Current System time,.... factors.
- + This alg is able to generate primary key value in the form of 32 bit hexa decimal value.
- + This alg is able to generate primary key value of the data type like String.
- + This alg is represented by Hibernate software in the form of a short name "uuid" and a predefined class name "org.hibernate.id.UUIDHexGenerator".
- + This alg is supported by almost all the databases which are supporting String type primary key values.
- + This alg is not required any input parameters, but, it must required "varchar" type primary key column with 32 column size minimum in database table.

**EX:**

```

1. <hibernate-mapping>
2. <class name="Employee" table="emp1">
3. <id name="eno" column="ENO" type="string">
4. <generator class="org.hibernate.id.UUIDHexGenerator"/>
5. </id>
6. ---
7. </class>
8. </hibernate-mapping>
```

## 10. GUID

- + This alg is able to generate primary key values on the basis of the underlying database provided identity column with String type.
- + This alg is represented in the form of a predefined class org.hibernate.id.GUIDGenerator and in the form of a short name like "guid".
- + This alg is not requires any input parameters to generate primary values.

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- This alg is supported by the databases which are supporting string type id columns.
- EX:**

```

1. <hibernate-mapping>
2. <class name="Employee" table="emp1">
3. <id name="eno" column="ENO">
4. <generator class="org.hibernate.id.GUIDGenerator"/>
5. </id>
6. ---
7. </class>
8. </hibernate-mapping>
```

## 11. foreign

- This primary key generation alg is able to generate primary key values on the basis of the foreign key of the present table.
- This alg is able to generate primary key values of the data types like short, int and long.
- This alg is not required any input parameters to generate primary key values.
- This alg is represented by Hibernate software in the form of a short name "foreign" and in the form of a predefined class like "org.hibernate.id.ForeignGenerator".

**EX:**

```

1. <hibernate-mapping>
2. <class name="Employee" table="emp1">
3. <id name="eno" column="ENO">
4. <generator class="org.hibernate.id.ForeignGenerator"/>
5. </id>
6. ---
7. </class>
8. </hibernate-mapping>
```

## Annotations Support for Primary Key Generation Alg:

Hibernate has provided annotations for almost all the primary key generation alg in the form of javax.persistence package and org.hibernate.annotation package.

To provide annotations for primary key generation alg javax.persistence package has provided the following annotations.

- @GeneratedValue(--,--)
- @SequenceGenerator(--)
- @TableGenerator(--)

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

---

To provide a particular primary key generation alg in hibernate applications, we have to use "strategy" member in @GeneratedValue(-) annotation, it will take either of the following constants from "GenerationType" Enum.

1. IDENTITY
2. SEQUENCE
3. TABLE
4. AUTO

**1. IDENTITY:** This value of GenerationType enum will represent IdentityGenerator or "identity" primary key generation algorithm to generate primary key value on the basis of the underlying database provided identity column.

#### EX: Employee.java

```

1. @Entity
2. @Table(name="emp1")
3. public class Employee {
4. @Id
5. @Column(name="ENO")
6. @GeneratedValue(strategy=GenerationType.IDENTITY)
7. private int eno;
8. @Column(name="ENAME")
9. private String ename;
10. @Column(name="ESAL")
11. private float esal;
12. @Column(name="EADDR")
13. private String eaddr;
14. setXXX() and getXXX()
15.}
```

**2. SEQUENCE :** This constant from GenerationType enum is able to represent SequenceGenerator or "sequence" primary key generation algorithm inorder to generate primary key value on the basis of the sequence which we defined in database.

To configure "sequence" name we have to use "@SequenceGenerator" annotation with the following members.

1. **name:** It will take logical name of the @SequenceGenerator.
  2. **sequenceName:** it will take "sequence" name provided by underlying database.
- To apply @SequenceGenerator to @GeneratedValue annotation we have to use "generator" member in "@GeneratedValue" annotation.

#### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**EX: Employee.java**

```
1. @Entity
2. @Table(name="emp1")
3. public class Employee {
4. @Id
5. @Column(name="ENO")
6. @SequenceGenerator(name="seqGen", sequenceName="my_sequence")
7. @GeneratedValue(strategy=GenerationType.SEQUENCE, generator="seqGen")
8. private int eno;
9. @Column(name="ENAME")
10. private String ename;
11. @Column(name="ESAL")
12. private float esal;
13. @Column(name="EADDR")
14. private String eaddr;
15. setXXX() and getXXX()
16.}
```

**Note:** In the above example , if we remove @SequenceGenerator annotation and "generator" member in @GeneratedValue annotation then Hibernate software will search for the default sequence name like "hibernate\_sequence" in the database.

**3. TABLE:** This constant from GenerationType enum is able to represent HiLo Primary key generation alg inorder to generate primary key values on the basis of Global resource. To configure Global resource table and its properties we have to use @TableGenerator() annotation with the following members.

1. **name:** it will take logical name to the @TableGenerator
2. **pkColumnName:** it will take primary key column name of the Global resource table.
3. **pkColumnNameValue:** It will take initial value to the primary key column in global resource table.
4. **valueColumnName:** it will take a column name which is generating our required primary key value.
5. **table:** it will take Global resource table name.

To apply @TableGenerator annotation to @GeneratedValue annotation we have to use "generator" member in @GeneratedValue annotation.

**EX: Employee.java****CONTACT US:****Mobile:** +91- 8885 25 26 27  
+91- 7207 21 24 27/28**US NUM:** 4433326786Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

1. @Entity
2. @Table(name="emp1")
3. public class Employee {
4. @Id
5. @Column(name="ENO")
6. @TableGenerator(name="tableGen",
7. table="my_table",
8. pkColumnName="id",
9. pkColumnValue="10", valueColumnName="next_hi")
10. @GeneratedValue(strategy=GenerationType.TABLE, generator="table
Gen")
11. private int eno;
12. @Column(name="ENAME")
13. private String ename;
14. @Column(name="ESAL")
15. private float esal;
16. @Column(name="EADDR")
17. private String eaddr;
18. setXXX() and getXXX()
19.}
```

**4. AUTO:** This constant from GenerationType enum is able to represent "native" primary key generation alg inorder to generate primary key value by selecting either sequence or identity or hilo primary key generation algorithms on the basis of the under lying database.

**Note:** Depending on the database or depending on the internal primary key generation alg we have to provide 4 input parameters by using @SequenceGenerator() or @TableGenerator() annotations.

#### EX: Employee.java

```

1. @Entity
2. @Table(name="emp1")
3. public class Employee {
4. @Id
5. @Column(name="ENO")
6. @SequenceGenerator(name="seqGen", sequenceName="my_sequence")
7. @GeneratedValue(strategy=GenerationType.AUTO, generator="seqGen")
8. private int eno;
9. @Column(name="ENAME")
10. private String ename;
11. @Column(name="ESAL")
12. private float esal;
```

#### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

13. @Column(name="EADDR")
14. private String eaddr;
15. setXXX() and getXXX()
16.

```

**Note:** For the remaining primary key generation alg , Hibernate has provided the following annotation in org.hibernate.annotation package.

```

@GenericGenerator(name="--", strategy="--",parameters={...})
@GeneratedValue(generator="--")

```

1. name: It will take logical name to @GenericGenerator
2. strategy: It will take short name of the primary key generation alg.
3. parameters: It will take array of @Parameter() with the following members to declare parameter values.
  1. name: it will take param name.
  2. value: It will take param value.

#### EX:Employee.java

```

1. @Entity
2. @Table(name="emp1")
3. public class Employee{
4. @Id
5. @Column(name="ENO")
6. @GenericGenerator(name="incGen" strategy="increment")
7. @GeneratedValue(generator="incGen")
8. private int eno;
9. ----
10. ----
11.

```

#### EXAMPLE:

#### Employee.java

```

1. package com.durgasoft.hbn.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.GeneratedValue;
6. import javax.persistence.GenerationType;
7. import javax.persistence.Id;
8. import javax.persistence.SequenceGenerator;

```

#### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9. import javax.persistence.Table;
10. import javax.persistence.TableGenerator;
11.
12. import org.hibernate.annotations.Generated;
13. import org.hibernate.annotations.GenericGenerator;
14. import org.hibernate.annotations.Parameter;
15.
16. @Entity
17. @Table(name="emp1")
18. public class Employee {
19. @Id
20. @Column(name="ENO")
21. /*
22. @SequenceGenerator(name="seqGen", sequenceName="my_sequence")
23. @GeneratedValue(strategy=GenerationType.SEQUENCE, generator="seqGen")
24. */
25. // @GeneratedValue(strategy=GenerationType.IDENTITY)
26. // @GeneratedValue(strategy=GenerationType.AUTO)
27. /*
28. @TableGenerator(name="tableGen", table="my_table", pkColumnName="id", pkColumnValue="10", valueColumnName="next_hi")
29. @GeneratedValue(strategy=GenerationType.TABLE, generator="tableGen")
30. */
31. @GenericGenerator(name="incrementGen", strategy="increment")
32. @GeneratedValue(generator="incrementGen")
33.
34. private int eno;
35. @Column(name="ENAME")
36. private String ename;
37. @Column(name="ESAL")
38. private float esal;
39. @Column(name="EADDR")
40. private String eaddr;
41.
42. public int getEno() {
43. return eno;
44. }
45. public void setEno(int eno) {
46. this.eno = eno;
47. }
48. public String getEname() {
49. return ename;
50. }
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

51. public void setEname(String ename) {
52. this.ename = ename;
53. }
54. public float getEsal() {
55. return esal;
56. }
57. public void setEsal(float esal) {
58. this.esal = esal;
59. }
60. public String getEaddr() {
61. return eaddr;
62. }
63. public void setEaddr(String eaddr) {
64. this.eaddr = eaddr;
65. }
66.
67.

```

### hibernate.cfg.xml



```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.
7. <session-factory>
8. <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
9. <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
10. <property name="connection.username">system</property>
11. <property name="connection.password">durga</property>
12. <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</propert
y>
13. <property name="show_sql">true</property>
14. <mapping class="com.durgasoft.hbn.pojo.Employee"/>
15. </session-factory>
16.
17. <!--
18. <session-factory>
19. <property name="connection.driver_Class">com.mysql.jdbc.Driver</property>
20. <property name="connection.url">jdbc:mysql://localhost:3306/durgadb</property>
21. <property name="connection.username">root</property>
22. <property name="connection.password">root</property>

```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
23. <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
24. <property name="show_sql">true</property>
25. <mapping class="com.durgasoft.hbn.pojo.Employee"/>
26. </session-factory>
27.-->
28.</hibernate-configuration>
```

**ClientApp.java**

```
1. package com.durgasoft.hbn.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10. import com.durgasoft.hbn.pojo.Employee;
11.
12. public class Test {
13.
14. public static void main(String[] args) throws Exception {
15. Configuration cfg = new Configuration();
16. cfg.configure();
17. StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
18. builder = builder.applySettings(cfg.getProperties());
19. StandardServiceRegistry registry = builder.build();
20. SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
21. Session session = sessionFactory.openSession();
22. Employee emp = new Employee();
23. //emp.setEno(111);
24. emp.setEname("AAA");
25. emp.setEsal(5000);
26. emp.setEaddr("Hyd");
27. Transaction tx = session.beginTransaction();
28. session.save(emp);
29. tx.commit();
30. System.out.println("Employee Inserted Successfully");
31.
32. }
33. }
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

## Transaction Management

### Transaction Management:

Transaction is a unit of work performed by Front End applications on Back End System.

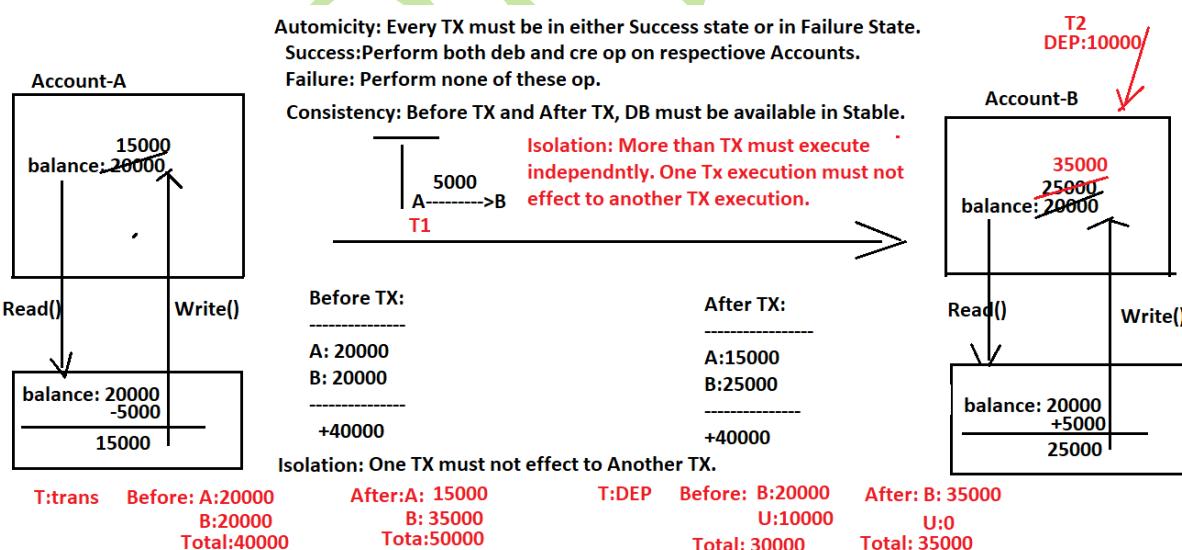
**EX:** Deposit some amount in an Account.

Withdraw some amount from an Account

Transfer some amount from one account to another account.

IN database applications, Every Transaction must follow ACID properties

1. **Atomicity:** This property will make the Transaction either in SUCCESS state or in FAILURE state.  
In Database related applications, if we perform all operations then the Transaction is available in SUCCESS State, if we perform none of the operations then the Transaction is available in FAILURE state.
2. **Consistency:** In database applications, Before the Transaction and After the Transaction Database state must be in stable.
3. **Isolation:** If we run more than one Transaction on a single Data item then that Transactions are called as "Concurrent Transactions". In Transactions Concurrency , one transaction execution must not give effect to another Transaction, this rule is called as "Isolation" property.
4. **Durability:** After committing the Transaction, if any failures are coming like Power failure, OS failure,...after getting the System if we open the transaction then the modifications which we performed during the transaction must be preserved.



### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In JDBC, to perform Automicity property we have to change Connections auto-commit nature and we have to perform either commit() or rollback() at the end of Transaction.

**EX:**

```

1. Connection con = DriverManager.getConnection(...);
2. con.setAutoCommit(false);
3. try{
4. ---instructions-----
5. con.commit();
6. }catch(Exception e){
7. e.printStackTrace();
8. con.rollback();
9. }
```

In Hibernate applications, if we want to manage Transactions Automicity property then we have to use the following steps.

1. Declare Transaction Before try.
2. Create Transaction object inside try block.
3. Perform commit() operation at end of Transaction.
4. Perform rollback() operation at catch block.

```
Transaction tx = null;
try{

 tx = session.beginTransaction();

 tx.commit();
}catch(Exception e){
 tx.rollback();
}
```

**Example:****Employee.java**

```

1. package com.durgasoft.pojo;
2.
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="account")
10. public class Account {
11. @Id
12. @Column(name="ACCNo")
13. private String accNo;
14. @Column(name="BALANCE")
15. private int balance;
16.
17. public String getAccNo() {
18. return accNo;
19. }
20. public void setAccNo(String accNo) {
21. this.accNo = accNo;
22. }
23. public int getBalance() {
24. return balance;
25. }
26. public void setBalance(int balance) {
27. this.balance = balance;
28. }
29.
30.
31. }
```

**oracle\_cfg.xml**

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8. <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9. <property name="connection.username">system</property>
10. <property name="connection.password">durga</property>
11. <property name="show_sql">true</property>
```

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

12. <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
13. <mapping class="com.durgasoft.pojo.Account"/>
14. </session-factory>
15.</hibernate-configuration>
```



### mysql\_cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3. "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4. "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="connection.driver_Class">com.mysql.jdbc.Driver</property>
8. <property name="connection.url">jdbc:mysql://localhost:3306/durgadb</property>
9. <property name="connection.username">root</property>
10. <property name="connection.password">root</property>
11. <property name="show_sql">true</property>
12. <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

13. <mapping class="com.durgasoft.pojo.Account"/>
14. </session-factory>
15.</hibernate-configuration>
```

### Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10.import com.durgasoft.pojo.Account;
11.
12.public class Test {
13.
14. public static void main(String[] args) {
15. Configuration oracle_Cfg = null;
16. Configuration mysql_Cfg = null;
17. SessionFactory oracle_Sf = null;
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
18. SessionFactory mysql_Sf = null;
19. Session oracle_Session = null;
20. Session mysql_Session = null;
21. Transaction oracle_Tx = null;
22. Transaction mysql_Tx = null;
23. try {
24. oracle_Cfg = new Configuration();
25. oracle_Cfg.configure("oracle_cfg.xml");
26. mysql_Cfg = new Configuration();
27. mysql_Cfg.configure("mysql_cfg.xml");
28.
29. StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
30. builder = builder.applySettings(oracle_Cfg.getProperties());
31. StandardServiceRegistry oracle_registry = builder.build();
32. oracle_Sf = oracle_Cfg.buildSessionFactory(oracle_registry);
33.
34. builder = builder.applySettings(mysql_Cfg.getProperties());
35. StandardServiceRegistry mysql_registry = builder.build();
36. mysql_Sf = mysql_Cfg.buildSessionFactory(mysql_registry);
37.
38. oracle_Session = oracle_Sf.openSession();
39. mysql_Session = mysql_Sf.openSession();
40.
41. Account source_Account = (Account)oracle_Session.get("com.durgasoft.pojo.Account", "abc123");
42. int source_Balance = 0;
43. source_Balance = source_Account.getBalance();
44. source_Balance = source_Balance - 5000;
45. source_Account.setBalance(source_Balance);
46.
47. Account target_Account = (Account)mysql_Session.get("com.durgasoft.pojo.Account", "xyz123");
48. int target_Balance = 0;
49. target_Balance = target_Account.getBalance();
50. target_Balance = target_Balance + 5000;
51. target_Account.setBalance(target_Balance);
52.
53.
54. oracle_Tx = oracle_Session.beginTransaction();
55. mysql_Tx = mysql_Session.beginTransaction();
56. oracle_Session.update(source_Account);
57. mysql_Session.update(target_Account);
58. oracle_Tx.commit();
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
59. mysql_Tx.commit();
60. System.out.println("5000Rs Transferred from "+source_Account.getAccNo()+" to "+target_Account.getAccNo());
61. System.out.println("Transaction SUCCESS");
62. } catch (Exception e) {
63. e.printStackTrace();
64. oracle_Tx.rollback();
65. mysql_Tx.rollback();
66. System.out.println("Transaction Failure");
67. }finally {
68. oracle_Session.close();
69. mysql_Session.close();
70. oracle_Sf.close();
71. mysql_Sf.close();
72. }
73. }
74. }
```

If we execute more than one transaction on a single data item then that transactions are called as Concurrent Transactions.

In Transactions concurrency we are able to get the following data consistency problems while executing more than one transaction at a time.

1. Lost Update Problem
2. Dirty Read Problem
3. Non Repeatable Read Problem
4. Phantasm Read Problem

### 1. Lost Update Problem

In Transactions concurrency, if one transaction perform updatons over the data with out commit operation , mean while, other transactions perform updatons with commit operation then the first transaction updatons are lost, this data consistency problem is called as Lost Update problem.

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

| T1 bal=500  | T2           |
|-------------|--------------|
| Read(bal)   |              |
| bal=bal+500 |              |
| write(bal); |              |
|             | Read(bal);   |
|             | bal=bal+1000 |
|             | write(bal);  |
|             | commit();    |
| commit();   |              |

## 2. Dirty Read Problem:

In Transactions concurrency, if one transaction perform updatons over data with out performing commit / rollback, mean while if other Transaction perform Read operation over the uncommitted data with out performing commit/rollback operations, in this context, if first transaction perform Rollback operation then the read operation performed by second transaction is Dirty Read, this problem is called as Dirty Read problem.

| T1          | bal=500      | T2         |
|-------------|--------------|------------|
| Read(bal)   |              |            |
| bal=bal+500 |              |            |
| Write(bal)  |              |            |
|             | Read(bal)    | Dirty Read |
|             | bal=bal+1000 |            |
|             | Write(bal);  |            |
| Rollback(); |              |            |
|             | Commit();    |            |

## CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

### 3. Non Repeatable Read Problem

In Transactions concurrency, One transaction perform continuous read operations to get same results, mean while, between two read operations another transaction performs update operation over the same data, in this context, in the next read operation performed by first transaction may not get same repeatable results, this problem is called as Non Repeatable Read Problem.

|           | T1     | bal=500 | T2       |
|-----------|--------|---------|----------|
| A-Results | Read() |         |          |
| B-Results | Read() |         | Update() |
| C-Results | Read() |         | Update() |
| D-Results | Read() |         | Update() |
|           | -----  |         |          |



### 4. Phantasm Read Problem

In Transactions concurrency, one transaction perform read operation continuously to get same no of results at each and every read operation , mean while, other transactions may perform insert operations between two read operations performed by first transactions, in this context, in the next read operation performed by first transaction may not generate the same no of results, this problem is called as "Phantom Read" Problem, here the extra records inserted by second transaction are called as "Phantom Records".

|                       | T1     | T2       |
|-----------------------|--------|----------|
| 10 Results            | Read() |          |
| 20 Results            | Read() | insert() |
| <del>10 Results</del> | Read() | insert() |
| 30 Results            | Read() | insert() |
| <del>10 Results</del> | Read() |          |
| 40 Results            | Read() | insert() |
| <del>10 Results</del> | Read() |          |

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. public static final int TRANSACTION\_NONE = 0;
2. public static final int TRANSACTION\_READ\_UNCOMMITTED = 1;
3. public static final int TRANSACTION\_READ\_COMMITTED = 2;
4. public static final int TRANSACTION\_REPEATABLE\_READ = 4;
5. public static final int TRANSACTION\_SERIALIZABLE = 8;

**public void setTransactionIsolation(int isolation\_Level)****EX:** con.setTransactionIsolation(Connection.TRANSACTION\_READ\_COMMITTED);

&lt;property name="hibernate.connection.isolation&gt;val&lt;/property&gt;

Where value may be either of the following Constants

NONE = 0;  
READ\_UNCOMMITTED = 1;  
READ\_COMMITTED = 2;  
REPEATABLE\_READ = 4;  
SERIALIZABLE = 8;

**EX:**

```
1. <hibernate-configuration>
2. <session-factory>
3. ---
4. <property name="hibernate.connection.isolation">
5. SERIALIZABLE
6. </property>
7. ---
8. </session-factory>
9. </hibernate-configuration>
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

## Connection Pooling in Hibernate

### Connection Pooling in Hibernate:

- ★ In general, in Database related applications, if we want to perform database operations first we have to create connection object before database operations then we have to destroy connection object after the database operations. If we use this approach in database related applications then application performance will be reduced, because, Creating connection object and destroying Connection objects are two expensive processes.
- ★ In the above context, to improve application performance we have to avoid Connection object creation and Destruction processes every time, for this, we have to use Connection Pooling.
- ★ In Connection Pooling, first, we will create a pool object with a set of Connection objects at application loading time , then, we get Connection object from Pool when we want to perform database operations, after the database operations we will send Connection object back to Pool object without destroying Connection object.

To provide Connection pooling in JDBC applications we have to use the following steps.

#### 1. Create DataSource object :

DataSource is an object , it able to manage all JDBC parameters inorder to create Connection objects and it able to manage Pool objects with Connection objects.

In JDBC, to represent DataSource object, JDBC has provided a predefined interface in the form of javax.sql.DataSource and its implementation are provided by database vendors.

**EX1:** OracleDataSource provided by Oracle.

**EX2:** MySQLDataSource provided by MySQL.

**EX:** OracleDataSource ds = new OracleDataSource();

#### 2. Set JDBC Parameters to create Connection objects in POOL:

In DataSource, to create Connection objects in POOL , we have to provide Driver URL, Database User Name and Database password,....., for this, we have to use the following methods.

```
public void setURL(String driver_URL)
public void setUser(String db_User_Name)
public void setPassword(String password)
```

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**EX:**

```
ds.setURL("jdbc:oracle:thin:@localhost:1521:xe");
ds.setUser("system");
ds.setPassword("durga");
```

**3. Get Connection Object from DataSource:**

To get Connection object from DataSource we have to use the following method.

```
public Connection getConnection()
```

**EX:** Connection con = ds.getConnection();

**Note:** Perform Database operations with the Connection object

**4. Close Connection object:**

```
public void close()
```

**EX:** con.close();

**Note:** When we access close() on Connection object which we get from Pool , then, Connection object will not be destroyed, where Connection object will be send back to Pool object.

**EX:**

```
1. package com.durgasoft.jdbc;
2.
3. import java.sql.Connection;
4. import java.sql.ResultSet;
5. import java.sql.Statement;
6.
7. import oracle.jdbc.pool.OracleDataSource;
8.
9. public class ConnectionPoolingEx {
10. public static void main(String[] args) throws Exception {
11. OracleDataSource ds = new OracleDataSource();
12. ds.setURL("jdbc:oracle:thin:@localhost:1521:xe");
13. ds.setUser("system");
14. ds.setPassword("durga");
15. Connection con = ds.getConnection();
16. Statement st = con.createStatement();
17. ResultSet rs = st.executeQuery("select * from emp1");
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

18. System.out.println("ENO\tENAME\tESAL\tEADDR");
19. System.out.println("-----");
20. while(rs.next()) {
21. System.out.println(rs.getInt(1)+"\t"+rs.getString(2)+"\t"+ rs.getFloat(3)+"\t"+rs.getString(4));
22. }
23. con.close();
24. }
25.

```

In Hibernate applications, there are three ways to implement Connection Pooling.

1. Default Connection Pooling Mech in Hibernate
2. Third Party Vendors provided Connection pooling Mechs
3. Application Servers provided Connection pooling Mechs

### 1. Default Connection Pooling Mech in Hibernate:

In Hibernate applications, to interact with databases hibernate software is generating Connection objects by using its built-in Connection pooling mechanism .

Hibernate Software has implemented its built-in connection pooling mechanism in the form of a predefined class like

`org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl`

Hibernate provided built-in Connection pooling mechanism is able to allow 20 Connections as max count and 1 connection as min count, we can modify this pool size in hibernate applications as per the requirement by using "connection.pool\_size" property in hibernate configuration file.

```

1. <hibernate-configuration>
2. <session-factory>
3. -----
4. <property name="connection.pool_size">10</property>
5. -----
6. </session-factory>
7. </hibernate-configuration>

```

**Note:** Hibernate Software provided built-in connection pooling mechanism is suggestible upto Development and Testing phases, it is not suggestible for Production mode of our project.

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

## 2. Third Party Vendors provided Connection pooling Mechs:

In general, in database related applications we will use the following three third party vendors provided connection pooling mechanisms.

1. DBCP
2. C3P0
3. Proxool

Note: Hibernate is not providing support for DBCP Connection pooling mechanism, but, Hibernate has provided predefined support for C3P0 and Proxool connection pooling mechanisms.

### 1. C3P0 Connection Pooling Mechanism In Hibernate:

If we want to use C3P0 Connection pooling mechanism in hibernate applications then we have to declare the following properties in hibernate configuration file.

#### 1. `hibernate.connection.provider_class`:

This property will take Connection Pooling provider class which was provided by hibernate software for C3P0 connection pooling mechanism in the form of "org.hibernate.c3p0.internal.C3P0ConnectionProvider".

**Note:** C3P0ConnectionProvider class will activate C3P0 connection pooling mechanism in Hibernate applications.

#### 2. `hibernate.c3p0.min_size`:

It will take an int value which is representing minimum no of Connection objects in a pool.

#### 3. `hibernate.c3p0.max_size`:

It will take int value which is representing maximum no of Connection objects in a pool.

#### 4. `hibernate.c3p0.timeout`:

It will take connections idle time to destroy.

#### 5. `hibernate.c3p0.max_statements`:

It will take an int value representing no of statements max for Connection objects.

### CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: [durgasoftonlinetraining@gmail.com](mailto:durgasoftonlinetraining@gmail.com)

WEBSITE: [www.durgasoftonline.com](http://www.durgasoftonline.com)

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

**EX: hibernate.cfg.xml**

```
1. <hibernate-configuration>
2. <session-factory>
3. ----
4. <property name="connection.pool_size">10</property>
5. ----
6. </session-factory>
7. </hibernate-configuration>
8. <hibernate-configuration>
9. <session-factory>
10. <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
11. <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
12. <property name="connection.username">system</property>
13. <property name="connection.password">durga</property>
14. <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
15. <property name="show_sql">true</property>
16. <!-- C3P0 Connection Pooling Properties -->
17. <property name="hibernate.connection.provider_class">org.hibernate.c3p0.internal.C3P0
 ConnectionProvider</property>
18. <property name="hibernate.c3p0.min_size">1</property>
19. <property name="hibernate.c3p0.max_size">19</property>
20. <property name="hibernate.c3p0.timeout">120</property>
21. <property name="hibernate.c3p0.max_statements">10</property>
22. <mapping class="com.durgasoft.pojo.Employee"/>
23. </session-factory>
24. </hibernate-configuration>
```

**Note:** Add the following jars to the Hibernate Library.

1. c3p0-0.9.2.1.jar
2. hibernate-c3p0-4.3.11.Final.jar
3. mchange-commons-java-0.2.3.4.jar

**Example:****Employee.java**

- ```
1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp1")
10. public class Employee {
11.     @Id
12.     @Column(name="ENO")
13.     private int eno;
14.     @Column(name="ENAME")
15.     private String ename;
16.     @Column(name="ESAL")
17.     private float esal;
18.     @Column(name="EADDR")
19.     private String eaddr;
20.
21.     public int getEno() {
22.         return eno;
23.     }
24.     public void setEno(int eno) {
25.         this.eno = eno;
26.     }
27.     public String getEname() {
28.         return ename;
29.     }
30.     public void setEname(String ename) {
31.         this.ename = ename;
32.     }
33.     public float getEsal() {
34.         return esal;
35.     }
36.     public void setEsal(float esal) {
37.         this.esal = esal;
38.     }
39.     public String getEaddr() {
40.         return eaddr;
41.     }
42.     public void setEaddr(String eaddr) {
43.         this.eaddr = eaddr;
44.     }
45.
46. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.username">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.  <property name="show_sql">true</property>
13.  <!-- <property name="connection.pool_size">10</property> -->
14.  <!-- C3P0 Connection Pooling Properties -->
15.  <property name="hibernate.connection.provider_class">org.hibernate.c3p0.internal.C
    3P0ConnectionProvider</property>
16.  <property name="hibernate.c3p0.min_size">1</property>
17.  <property name="hibernate.c3p0.max_size">19</property>
18.  <property name="hibernate.c3p0.timeout">120</property>
19.  <property name="hibernate.c3p0.max_statements">10</property>
20.
21.  <mapping class="com.durgasoft.pojo.Employee"/>
22. </session-factory>
23.
24.</hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10. import com.durgasoft.pojo.Employee;
11.
12. public class Test {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
13.  
14. public static void main(String[] args) {  
15.     Transaction tx = null;  
16.     SessionFactory sessionFactory = null;  
17.     Session session = null;  
18.     try {  
19.         Configuration config = new Configuration();  
20.         config.configure("hibernate.cfg.xml");  
21.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();  
22.         builder = builder.applySettings(config.getProperties());  
23.         StandardServiceRegistry registry = builder.build();  
24.         sessionFactory = config.buildSessionFactory(registry);  
25.         session = sessionFactory.openSession();  
26.         Employee emp = new Employee();  
27.         emp.setEno(333);  
28.         emp.setEname("CCC");  
29.         emp.setEsal(7000);  
30.         emp.setEaddr("Hyd");  
31.         tx = session.beginTransaction();  
32.         session.save(emp);  
33.         tx.commit();  
34.         System.out.println("Employee Inserted Successfully");  
35.     } catch (Exception e) {  
36.         e.printStackTrace();  
37.         tx.rollback();  
38.         System.out.println("Employee Insertion Failure");  
39.     } finally {  
40.         session.close();  
41.         sessionFactory.close();  
42.     }  
43. }  
44. }
```

2. Proxool Connection Pooling Mechanism:

If we want to use Proxool Connection Pooling mechanism in hibernate applications then we have to use the following steps.

1. Provide Proxool connection pooling mechanism configurations in an xml file called as proxool configuration file.
2. Configure proxool configuration file in hibernate configuration file.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3.proxool configuration file:

It is an xml file, it includes all configuration details of proxool connection pooling mechanism which includes driver class name, driver url, database user name, datyabase password, minimum connection count, maximum connection count,.....

To provide the above configuration details we have to use the following xml tags.

```

1. <proxool-config>
2. <proxool>
3.   <alias>proxool_alias_name</alias>
4.   <driver-class> driuver class name</driver-class>
5.   <driver-url> driver url </driver-url>
6.   <driver-properties>
7.     <property name="prop_name" value="prop_value">
8.     -----
9.   </driver-properties>
10.  <minimum-connection-count> min_size </minimum-connection-count>
11.  <maximum-connection-count> max_size </maximum-connection-count>
12. </proxool>
13.</proxool-config>
```

EX:

proxool.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <proxool-config>
3. <proxool>
4.   <alias>proxool</alias>
5.   <driver-class>oracle.jdbc.OracleDriver</driver-class>
6.   <driver-url>jdbc:oracle:thin:@localhost:1521:xe</driver-url>
7.   <driver-properties>
8.     <property name="user" value="system"></property>
9.     <property name="password" value="durga"></property>
10.    </driver-properties>
11.    <minimum-connection-count>10</minimum-connection-count>
12.    <maximum-connection-count>20</maximum-connection-count>
13.  </proxool>
14. </proxool-config>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

To configure proxool configuration file in hibernate configuration file we have to use the following tags along with dialect and mapping configurations

1. **hibernate.connection.provider_class:** It will take Proxool connection pooling provider class which was provided by hibernate software in the form of "org.hibernate.connection.ProxoolConnectionProvider" inorder to activate proxool connection pooling mechanism.
2. **hibernate.proxool.pool_alias:** It will take proxool alias name which we defined in proxool configuration file.
3. **hibernate.proxool.xml:** It will take the name and location of proxool configuration file.

EX:

hibernate.cfg.xml

```

1. <hibernate-configuration>
2. <session-factory>
3. <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect </property>
4. <property name="hibernate.connection.provider_class">org.hibernate.connection.Proxool
   ConnectionProvider</property>
5. <property name="hibernate.proxool.pool_alias">proxool</property>
6. <property name="hibernate.proxool.xml">proxool.xml</property>
7. <mapping class="com.durgasoft.hbn.pojo.Employee"/>
8. </session-factory>
9. </hibernate-configuration>
```

Note: To use this mechanism, we have to add the following JAR files in Hibernate Library.

1. proxool-0.8.3.jar
2. hibernate-proxool-4.3.11.Final.jar

Example:

Employee.java

```

1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp1")
10. public class Employee {
11.     @Id
12.     @Column(name="ENO")
13.     private int eno;
14.     @Column(name="ENAME")
15.     private String ename;
16.     @Column(name="ESAL")
17.     private float esal;
18.     @Column(name="EADDR")
19.     private String eaddr;
20.
21.     public int getEno() {
22.         return eno;
23.     }
24.     public void setEno(int eno) {
25.         this.eno = eno;
26.     }
27.     public String getEname() {
28.         return ename;
29.     }
30.     public void setEname(String ename) {
31.         this.ename = ename;
32.     }
33.     public float getEsal() {
34.         return esal;
35.     }
36.     public void setEsal(float esal) {
37.         this.esal = esal;
38.     }
39.     public String getEaddr() {
40.         return eaddr;
41.     }
42.     public void setEaddr(String eaddr) {
43.         this.eaddr = eaddr;
44.     }
45.
46. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
8. <property name="hibernate.connection.provider_class">org.hibernate.connection.Proxool
ConnectionProvider</property>
9. <property name="hibernate.proxool.pool_alias">proxool</property>
10.<property name="hibernate.proxool.xml">proxool.xml</property>
11.<mapping class="com.durgasoft.pojo.Employee"/>
12.</session-factory>
13.</hibernate-configuration>
```

proxool.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <proxool-config>
3. <proxool>
4. <alias>proxool</alias>
5. <driver-class>oracle.jdbc.OracleDriver</driver-class>
6. <driver-url>jdbc:oracle:thin:@localhost:1521:xe</driver-url>
7. <driver-properties>
8. <property name="user" value="system"></property>
9. <property name="password" value="durga"></property>
10.</driver-properties>
11.<minimum-connection-count>10</minimum-connection-count>
12.<maximum-connection-count>20</maximum-connection-count>
13.</proxool>
14.</proxool-config>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10. import com.durgasoft.pojo.Employee;
11.
12. public class Test {
13.
14.     public static void main(String[] args) {
15.         Transaction tx = null;
16.         SessionFactory sessionFactory = null;
17.         Session session = null;
18.         try {
19.             Configuration config = new Configuration();
20.             config.configure("hibernate.cfg.xml");
21.             StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
22.             builder = builder.applySettings(config.getProperties());
23.             StandardServiceRegistry registry = builder.build();
24.             sessionFactory = config.buildSessionFactory(registry);
25.             session = sessionFactory.openSession();
26.             Employee emp = new Employee();
27.             emp.setEno(111);
28.             emp.setEname("AAA");
29.             emp.setEsal(5000);
30.             emp.setEaddr("Hyd");
31.             tx = session.beginTransaction();
32.             session.save(emp);
33.             tx.commit();
34.             System.out.println("Employee Inserted Successfully");
35.         } catch (Exception e) {
36.             e.printStackTrace();
37.             tx.rollback();
38.             System.out.println("Employee Insertion Failure");
39.         } finally {
40.             session.close();
41.             sessionFactory.close();
42.         }
43.     }
44. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



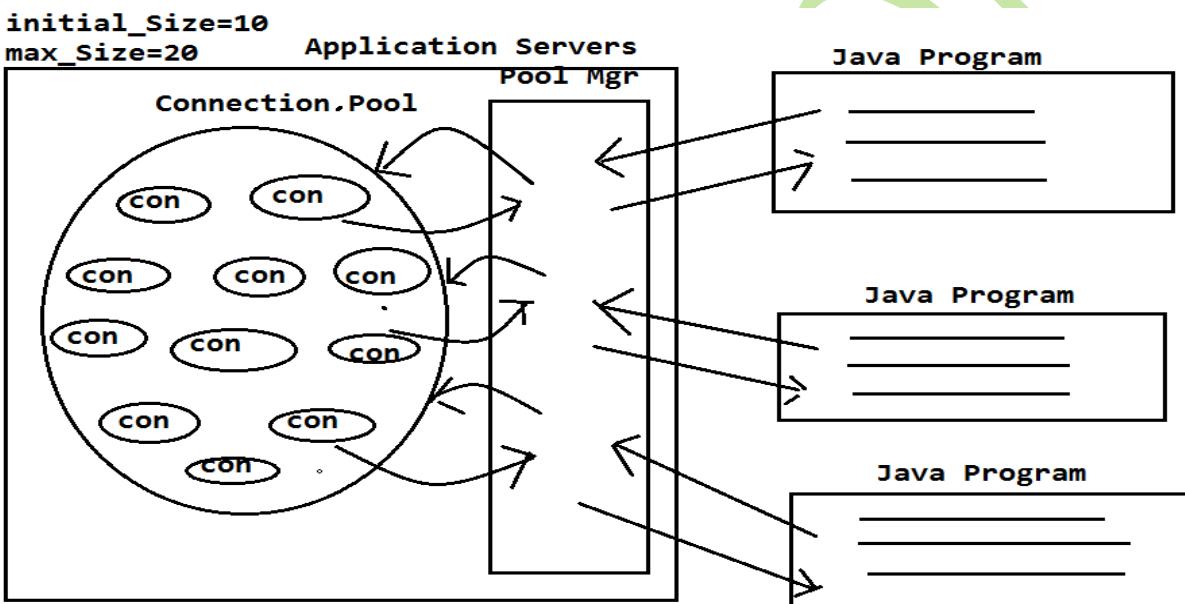
BY NAGOOR BABU

3. Connection Pooling through Application Servers

1. Connection Pooling through Application Servers built in mechanisms by using JNDI:

JNDI [Java Naming And Directory Interface]: JNDI is a Middleware Service or an abstraction provided by SUN Microsystems as part of J2EE and which is implemented by all the Application Servers vendors like Weblogic, JBOSS, Glassfish,.....

JNDI is existed inside the application Servers to provide any resource with Global Scope, that is, JNDI will share any resource like "DataSource" to all the applications which are running in the present application server.



In general, almost all the Application Servers are having their own Connection Pooling mechanisms, if we want to use Application Servers provided Connection pooling mechanisms we have to use the following steps.

- 1) Install Application Server.
 - 2) Configure Connection Pooling and Datasource in JNDI provided by Application Servers.
 - 3) Access Application Servers provided Datasource in our Application.

1) Install Application Server[Weblogic Server]

1. Download fmw_12.2.1.3.0_wls_quick.jar from internet[oracle.com]
 2. Open command prompt in Administrator mode.
 3. Set JAVA8 or JAVA7 in path.

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: **durgasoftonlinetraining@gmail.com**

WEBSITE: www.durgasoftonline.com



BY NAGOOR BABU

- set path=C:\Java\jdk1.8.0_144\bin;
4. Goto setup file location and execute JAR file with the following command.
F:\softwares\servers\weblogic>java -jar fmw_12.2.1.3.0_wls_quick.jar
 5. Click on "Next" button.
 6. Click on "Next" Button
 7. Specify Home directory "Oracle"
 8. Click on "Next" button.
 9. Click on "Next" button.
 10. Click on "Next" button
 11. Click on "Install" button.
 12. Click on "Next" button.
 13. Click on "Finish" button.
 14. Provide domain name "durga_domain".
 15. Click on "Next" button.
 16. Click on "Next" button.
 17. Provide user name and password.
 user name: weblogic
 password: weblogic_weblogic
 confirm password: weblogic_weblogic
 18. Click on "Next" button.
 19. Click on "Next" button.
 20. Select "Administration Server"
 21. Click on "Next" button.
 22. Click on "Next" button.
 23. Click on "Create" button.
 24. Click on "Next" button.
 25. Click on "Finish" button.

2) Configure Connection Pooling and Datasource in JNDI provided by Application Servers:

1. Goto durga_domain location
C:\Oracle\user_projects\domains\durga_domain
2. double click on "startWeblogic" batch file.
3. Open Browser and provide the following url to open Administration console.
<http://localhost:7001/console>
4. Provide domain user name and password.
 user name: weblogic
 password: weblogic_weblogic
5. Click on "Login" button.
6. Go for Domain Structure and select "Services".
7. Select "DataSource".
8. Click on "New" button.
9. Select "Generic datasource".
10. Provide the following details.

CONTACT US:

Mobile: +91- 8885 25 26 27 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Name: durgads
 Scope: GLOBAL
 JNDI Name: durgajndi
 Database Type: Oracle

11. Click on "Next" button.
12. Click on "Next" button.
13. Click on "Next" button.
14. Provide the following details.

Database Name: xe
 Host Name: localhost
 Port : 1521
 Database User Name: system
 password: durga
 confirm password: durga

15. Click on "Next" button.
16. Click on "Next" button.
17. Select "admin server".
18. Click on "Finish" button.

If we want to get DataSource object from Weblogic Server provided JNDI into JDBC Application then we ahve to use the following steps.

1. Create Hashtable object and put the following Weblogic Server provided JNDI configurations:
 - a) INITIAL_CONTEXT_FACTORY: It is a context from javax.naming.Context class , it will take Server provided ContextFactory. Weblogic Server has provided a seperate ContextFactory in the form of weblogic.jndi.ELInitialContextFactory.
 - b) PROVIDER_URL: It is a constant provided by javax.naming.Context class, it will take JNDI server URL provided by Application Servers inorder to get Datasource object. Web logic Server has provided JNDI url like "t3://localhost:7001"

EX:

```

1.     Hashtable<String, String> ht = new Hashtable<String, String>();
2.     ht.put(Context.INITIAL_CONTEXT_FACTORY, "weblogic.jndi.WLInitialContextFactory");
3.     ht.put(Context.PROVIDER_URL, "t3://localhost:7001");

```

2. Create InitialContext object:

javax.naming.InitialContext class is representing JNDI server or registry, it can be used to keep resources in JNDI Server and retrive resources from JNDI server.

To create InitialContext object we have to use the following constructor.

public InitialContext(Map map)

EX: InitialContext context = new InitialContext(ht);

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3. Get DataSource object from JNDI on the basis of JNDI Name which we configured in Appl Server:

To get DataSource object from weblogic application Server we have to use the following method.

```
public static Object lookup(String jndi_Name)
```

EX: `DataSource ds = (DataSource)Context.lookup("durgajndi");`

4. Get Connection from DataSource:

To get Connection from dataSource we have to use the following method.

```
public Connection getConnection()
```

EX: `Connection con = ds.getConnection();`

Note: After getting Connection object we will provide our own JDBC Application logic inorder to perform Database operations.

EX:

JDBC Application to use Weblogic Server priveded Connection pooling Mechanism through JNDI.

```
1. package com.durgasoft.jdbc;
2. import java.sql.Connection;
3. import java.sql.ResultSet;
4. import java.sql.Statement;
5. import java.util.Hashtable;
6. import javax.naming.Context;
7. import javax.naming.InitialContext;
8. import javax.sql.DataSource;
9.
10. public class JdbcApp {
11.
12.     public static void main(String[] args) throws Exception {
13.         Hashtable<String, String> ht = new Hashtable<String, String>();
14.         ht.put(Context.INITIAL_CONTEXT_FACTORY, "weblogic.jndi.WLInitialContextFactory");
15.         ht.put(Context.PROVIDER_URL, "t3://localhost:7001");
16.         InitialContext context = new InitialContext(ht);
17.         DataSource ds = (DataSource)context.lookup("durgajndi");
18.         Connection con = ds.getConnection();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

19. Statement st = con.createStatement();
20. ResultSet rs = st.executeQuery("select * from emp1");
21. System.out.println("ENO\tENAME\tESAL\tEADDR");
22. System.out.println("-----");
23. while(rs.next()) {
24.     System.out.println(rs.getInt(1)+"\t"+rs.getString(2)+"\t"+rs.getFloat(3)+"\t"+rs.getString(4));
25. }
26. }
27.

```

Note: To run this example We have to keep odbc6.jar and weblogic.jar in build path. weblogic.jar is existed in weblogic server

C:\Oracle\wlserver\server\lib\weblogic.jar

If we want to use Application Server provided DataSource object in Hibernate applications then we have to provide the following properties configuration in hibernate configuration File.

1. **hibernate.connection.provider_class:**

In Hibernate, Every Connection Pooling mechanism will have a separate Provider Class, this property will take the Connection pooling mechanisms provider class which we defined by hibernate. For Application Servers Connection Pooling mechanisms we have to provide "org.hibernate.engine.jdbc.connections.internal.DataSourceConnectionProviderImpl".

2. **hibernate.connection.datasource:**

This property will take JNDI name of the datasource object which we configured in application Server.

3. **hibernate.jndi.class:**

It will take ConnectionFactory class provided by application Servers. Weblogic server will use "weblogic.jndi.WLInitialContextFactory" as value to this property.

4. **hibernate.jndi.url:**

This property will take JNDI url to get DataSource object , All application Servers has to provide a separate URL to access JNDI registry, Weblogic has provided "t3://localhost:7001" as JNDI url.

EX:

1. <hibernate-configuration>

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2. <session-factory>
3. <property name="hibernate.connection.provider_class"> org.hibernate.engine.jdbc.connections.internal.DataSourceConnectionProviderImpl
4. </property>
5. <property name="hibernate.connection.datasource">oraclejndi</property>
6. <property name="hibernate.jndi.class">weblogic.jndi.WLInitialContextFactory</property>

7. <property name="hibernate.jndi.url">t3://localhost:7001</property>
8. <mapping class="com.durgasoft.pojo.Employee"/>
9. </session-factory>
10.</hibernate-configuration>

```

Note: To run this example we have to keep weblogic.jar in Hibernate Lib and we must remove ojdbc6.jar from Hibernate Lib.

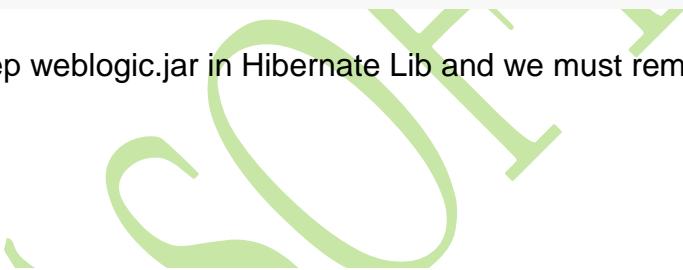
Example:

Employee.java

```

1. package com.durgasoft.hbn.pojo;
2. import javax.persistence.Column;
3. import javax.persistence.Entity;
4. import javax.persistence.Id;
5.
6. @Entity
7. @Table(name="emp1")
8. public class Employee {
9.     @Id
10.    @Column(name="ENO")
11.    private int eno;
12.    @Column(name="ENAME")
13.    private String ename;
14.    @Column(name="ESAL")
15.    private float esal;
16.    @Column(name="EADDR")
17.    private String eaddr;
18.
19.    public int getEno() {
20.        return eno;
21.    }
22.    public void setEno(int eno) {
23.        this.eno = eno;
24.    }
25.    public String getEname() {

```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

26.     return ename;
27. }
28. public void setEname(String ename) {
29.     this.ename = ename;
30. }
31. public float getEsal() {
32.     return esal;
33. }
34. public void setEsal(float esal) {
35.     this.esal = esal;
36. }
37. public String getEaddr() {
38.     return eaddr;
39. }
40. public void setEaddr(String eaddr) {
41.     this.eaddr = eaddr;
42. }
43.
44.
45.}
```

hibernate.cfg.xml



```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6. <session-factory>
7. <property name="hibernate.connection.provider_class">org.hibernate.engine.jdbc.connections.internal.DataSourceConnectionProviderImpl</property>
8. <property name="hibernate.connection.datasource">oraclejndi</property>
9. <property name="hibernate.jndi.class">weblogic.jndi.WLInitialContextFactory</property>
10. <property name="hibernate.jndi.url">t3://localhost:7001</property>
11. <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12. <mapping class="com.durgasoft.hbn.pojo.Employee"/>
13.
14. </session-factory>
15.
16.
17.</hibernate-configuration>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```
1. package com.durgasoft.hbn.test;
2.
3. import java.util.Properties;
4.
5. import javax.naming.Context;
6. import javax.naming.InitialContext;
7.
8. import org.hibernate.Session;
9. import org.hibernate.SessionFactory;
10. import org.hibernate.Transaction;
11. import org.hibernate.boot.registry.StandardServiceRegistry;
12. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
13. import org.hibernate.cfg.Configuration;
14.
15. import com.durgasoft.hbn.pojo.Employee;
16.
17. public class Test {
18.
19.     public static void main(String[] args) throws Exception {
20.
21.         Configuration cfg = new Configuration();
22.         cfg.configure();
23.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
24.         builder = builder.applySettings(cfg.getProperties());
25.         StandardServiceRegistry registry = builder.build();
26.         SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
27.         Session session = sessionFactory.openSession();
28.
29.         Employee emp = new Employee();
30.         emp.setEno(111);
31.         emp.setEname("AAA");
32.         emp.setEsal(5000);
33.         emp.setEaddr("Hyd");
34.         Transaction tx = session.beginTransaction();
35.         session.save(emp);
36.         tx.commit();
37.         System.out.println("Employee Saved Successfully");
38.     }
39. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Bulk Operations

In General, in Hibernate Applications, by using org.hibernate.Session interface provided methods like save (), update (), saveOrUpdate (), delete (), get () ... we are able to perform manipulations over single record.

In Hibernate applications, if we want to perform manipulations over multiple records then we must use the following features provided by Hibernate.

1. HQL[Hibernate Query Language]
2. Native SQL
3. Criteria API

1. HQL [Hibernate Query Language]

- ★ HQL is a powerful query language provided by Hibernate in order to perform manipulations over multiple records.
- ★ HQL is an object oriented query language, it able to support for the object oriented features like encapsulation, polymorphism,.... , but, SQL is structured query language.
- ★ HQL is database independent query language, but, SQL is database dependent query language.
- ★ In case of HQL, we will prepare queries by using POJO class names and their properties, but, in case of SQL , we will prepare queries on the basis of database table names and table columns.
- ★ HQL queries are prepared by using the syntaxes which are similar to SQL queries syntaxes.
- ★ HQL is mainly for retrieval operations , but, right from Hibernate3.x version we can use HQL to perform insert , update and delete operations along with select operations, but, SQL is able to allow any type of database operation.
- ★ In case of JDBC, in case of SQL, if we execute select sql query then records are retrieved from database table and these records are stored in the form of ResultSet object, which is not implementing java.io.Serializable , so that, it is not possible to transfer in the network, but, in the case of HQL, if we retrieve records then those records will be stored in Collection objects, which are Serializable by default, so that, we are able to carry these objects in the network.
- ★ HQL is database independent query language, but, SQL is database dependent query language.
- ★ In case of Hibernate applications, if we process any HQL query then Hibernate Software will convert that HQL Query into database dependent SQL Query and Hibernate software will execute that generated SQL query.

Note: HQL is not suitable where we want to execute Database dependent sql queries

EX: PL/SQL procedures and functions are totally database dependent, where we are unable to use HQL queries.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Procedure to use HQL queries in Hibernate Applications:

1. Represent HQL query by creating Query object.
2. Apply custom properties on HQL Query or on Query object.
3. Execute HQL Query

1. Represent HQL query by creating Query object.

Query object is able to store HQL query, to represent Query object Hibernate has provided a predefined interface in the form of "org.hibernate.Query".

To get Query object we have to use the following method from org.hibernate.Session .

```
public Query createQuery(String hql_Query)
```

EX: Query q = s.createQuery("from Employee");

2. Apply custom properties on HQL Query or on Query object.

In hibernate applications, after getting Query object we have to set the custom properties like providing fetch size , making the results as Cache results and read only results, providing start record position and max no of records,..... To perform all these custom properties we have to use the following methods.

1. public void setFetchSize(int size)
2. public void setCacheable(boolean b)
3. public void setMaxResults(int value)
4. public void setFirstResult(int value)
5. public void setReadOnly(boolean b)
6. public void setComment(String comment)
7. public void setTimeOut(int time)

EX:

```
q.setFetchSize(10);
q.setCacheable(true);
q.setMaxResults(10);
q.setFirstResult(5);
q.setReadOnly(true);
q.setComment("Employee Details");
q.setTimeOut(10000);
```

3. Execute HQL Query:

To execute HQL queries we will use the following methods.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

1. public List list()

- ❖ It will execute HQL query and generate the results in the form of List.

EX:

```

1. List<Employee> list = query.list();
2. System.out.println("ENO\tENAME\tESAL\tEADDR");
3. System.out.println("-----");
4. for(Employee e: list) {
5.     System.out.print(e.getEno()+"\t");
6.     System.out.print(e.getEname()+"\t");
7.     System.out.print(e.getEsal()+"\t");
8.     System.out.println(e.getEaddr());
9. }
```

2. public Iterator iterate()

- ❖ It will execute HQL query and generate the results in the form of Iterator.

EX:

```

1. Iterator<Employee> it = query.iterate();
2. System.out.println("ENO\tENAME\tESAL\tEADDR");
3. System.out.println("-----");
4. while(it.hasNext()) {
5.     Employee e = (Employee)it.next();
6.     System.out.print(e.getEno()+"\t");
7.     System.out.print(e.getEname()+"\t");
8.     System.out.print(e.getEsal()+"\t");
9.     System.out.println(e.getEaddr());
10. }
```

3. public ScrollableResults scroll()

- ❖ It able to execute HQL query and generate the results in form of ScrollableResults, which is same as ScrollableResultSet object , it allows to read data in both forward and backward directions.

Note: In the case of ScrollableResults we are able to use the following methods inorder to retrieve data.

public boolean next()

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public boolean previous()
public void first()
public void last()
public Object get(int position)
```

EX:

```
1. ScrollableResults results = query.scroll();
2.     System.out.println("Employee Details in Forward Direction");
3.     System.out.println("ENO\tENAME\tESAL\tEADDR");
4.     System.out.println("-----");
5.     while(results.next()) {
6.         Object[] obj = results.get();
7.         for(Object o: obj) {
8.             Employee e = (Employee)o;
9.             System.out.print(e.getEno()+"\t");
10.            System.out.print(e.getEname()+"\t");
11.            System.out.print(e.getEsal()+"\t");
12.            System.out.println(e.getEaddr());
13.        }
14.    }
15.
16.    System.out.println("Employee Details in Backward Direction");
17.    System.out.println("ENO\tENAME\tESAL\tEADDR");
18.    System.out.println("-----");
19.    while(results.previous()) {
20.        Object[] obj = results.get();
21.        for(Object o: obj) {
22.            Employee e = (Employee)o;
23.            System.out.print(e.getEno()+"\t");
24.            System.out.print(e.getEname()+"\t");
25.            System.out.print(e.getEsal()+"\t");
26.            System.out.println(e.getEaddr());
27.        }
28.    }
```

4. public Object uniqueResult()

- ❖ It will execute HQI query and it will return only one result, if more than one result is identified then it will rise an Exception.

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX:

```

1. Query query1 = session.createQuery("from Employee where eno= 111");
2. Object obj = query1.uniqueResult();
3. Employee e = (Employee)obj;
4. System.out.println("Employee Details");
5. System.out.println("-----");
6. System.out.println("Employee Number :" +e.getEno());
7. System.out.println("Employee Name : " +e.getEname());
8. System.out.println("Employee Salary : " +e.getEsal());
9. System.out.println("Employee Address:" +e.getEaddr());

```

5. public int executeUpdate()

- ❖ It can be used to perform the database operations like insert, update, delete,..... and it will generate rowCount value.

EX:

```

1. Query query = session.createQuery("update Employee set esal = esal + 500 where esal < 10000");
2. Transaction tx = session.beginTransaction();
3. int rowCount = query.executeUpdate();
4. tx.commit();
5. System.out.println("Records Updated : " +rowCount);

```

EX:

```

1. Query query = session.createQuery("delete from Employee where esal < 10000");
2. Transaction tx = session.beginTransaction();
3. int rowCount = query.executeUpdate();
4. tx.commit();
5. System.out.println("No of Records Deleted : " +rowCount);

```

Example:

Employee.java

```

1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp1")
10. public class Employee {
11.     @Id
12.     @Column(name="ENO")
13.     private int eno;
14.     @Column(name="ENAME")
15.     private String ename;
16.     @Column(name="ESAL")
17.     private float esal;
18.     @Column(name="EADDR")
19.     private String eaddr;
20.
21.     public int getEno() {
22.         return eno;
23.     }
24.     public void setEno(int eno) {
25.         this.eno = eno;
26.     }
27.     public String getEname() {
28.         return ename;
29.     }
30.     public void setEname(String ename) {
31.         this.ename = ename;
32.     }
33.     public float getEsal() {
34.         return esal;
35.     }
36.     public void setEsal(float esal) {
37.         this.esal = esal;
38.     }
39.     public String getEaddr() {
40.         return eaddr;
41.     }
42.     public void setEaddr(String eaddr) {
43.         this.eaddr = eaddr;
44.     }
45.
46.
47. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.    <property name="show_Sql">true</property>
13.    <mapping class="com.durgasoft.pojo.Employee"/>
14.  </session-factory>
15. </hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import java.util.Iterator;
4. import java.util.List;
5.
6. import org.hibernate.Query;
7. import org.hibernate.ScrollableResults;
8. import org.hibernate.Session;
9. import org.hibernate.SessionFactory;
10. import org.hibernate.boot.registry.StandardServiceRegistry;
11. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
12. import org.hibernate.cfg.Configuration;
13.
14. import com.durgasoft.pojo.Employee;
15.
16. public class Test {
17.
18.   public static void main(String[] args) throws Exception{
19.     Configuration cfg = new Configuration();
20.     cfg.configure();
21.     StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
22.     builder = builder.applySettings(cfg.getProperties());
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
23. StandardServiceRegistry registry = builder.build();
24. SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
25. Session session = sessionFactory.openSession();
26. Query query = session.createQuery("from Employee");
27.
28. System.out.println("Using list() method");
29. System.out.println("-----");
30. List<Employee> list = query.list();
31. System.out.println("ENO\tENAME\tESAL\tEADDR");
32. System.out.println("-----");
33. for(Employee e: list) {
34.     System.out.print(e.getEno()+"\t");
35.     System.out.print(e.getEname()+"\t");
36.     System.out.print(e.getEsal()+"\t");
37.     System.out.println(e.getEaddr());
38. }
39. System.out.println();
40. System.out.println("Using iterate() method");
41. System.out.println("-----");
42. Iterator<Employee> it = query.iterate();
43. System.out.println("ENO\tENAME\tESAL\tEADDR");
44. System.out.println("-----");
45. while(it.hasNext()) {
46.     Employee e = (Employee)it.next();
47.     System.out.print(e.getEno()+"\t");
48.     System.out.print(e.getEname()+"\t");
49.     System.out.print(e.getEsal()+"\t");
50.     System.out.println(e.getEaddr());
51. }
52. System.out.println();
53.
54. System.out.println("Using scroll() method");
55. System.out.println("-----");
56. ScrollableResults results = query.scroll();
57. System.out.println("Employee Details in Forward Direction");
58. System.out.println("ENO\tENAME\tESAL\tEADDR");
59. System.out.println("-----");
60. while(results.next()) {
61.     Object[] obj = results.get();
62.     for(Object o: obj) {
63.         Employee e = (Employee)o;
64.         System.out.print(e.getEno()+"\t");
65.         System.out.print(e.getEname()+"\t");
```

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
66.         System.out.print(e.getEsal()+"\t");
67.         System.out.println(e.getEaddr());
68.     }
69. }
70.
71. System.out.println("Employee Details in Backward Direction");
72. System.out.println("ENO\tENAME\tESAL\tEADDR");
73. System.out.println("-----");
74. while(results.previous()) {
75.     Object[] obj = results.get();
76.     for(Object o: obj) {
77.         Employee e = (Employee)o;
78.         System.out.print(e.getEno()+"\t");
79.         System.out.print(e.getEname()+"\t");
80.         System.out.print(e.getEsal()+"\t");
81.         System.out.println(e.getEaddr());
82.     }
83. }
84. System.out.println();
85. System.out.println("Using uniqueResult() method");
86. System.out.println("-----");
87. Query query1 = session.createQuery("from Employee where eno= 111");
88. Object obj = query1.uniqueResult();
89. Employee e = (Employee)obj;
90. System.out.println("Employee Details");
91. System.out.println("-----");
92. System.out.println("Employee Number :" +e.getEno());
93. System.out.println("Employee Name : " +e.getEname());
94. System.out.println("Employee Salary : " +e.getEsal());
95. System.out.println("Employee Address:" +e.getEaddr());
96.
97. session.close();
98. sessionFactory.close();
99. }
100. }
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example on executeUpdate() method to perform update and delete operations:**Employee.java**

```
1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp1")
10. public class Employee {
11.     @Id
12.     @Column(name="ENO")
13.     private int eno;
14.     @Column(name="ENAME")
15.     private String ename;
16.     @Column(name="ESAL")
17.     private float esal;
18.     @Column(name="EADDR")
19.     private String eaddr;
20.
21.     public int getEno() {
22.         return eno;
23.     }
24.     public void setEno(int eno) {
25.         this.eno = eno;
26.     }
27.     public String getEname() {
28.         return ename;
29.     }
30.     public void setEname(String ename) {
31.         this.ename = ename;
32.     }
33.     public float getEsal() {
34.         return esal;
35.     }
36.     public void setEsal(float esal) {
37.         this.esal = esal;
38.     }
39.     public String getEaddr() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

40.     return eaddr;
41. }
42. public void setEaddr(String eaddr) {
43.     this.eaddr = eaddr;
44. }
45.
46.
47.

```



hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.    <property name="show_Sql">true</property>
13.    <mapping class="com.durgasoft.pojo.Employee"/>
14.  </session-factory>
15. </hibernate-configuration>

```

Test.Java:

```

1. package com.durgasoft.test;
2.
3. import java.util.Iterator;
4. import java.util.List;
5.
6. import org.hibernate.Query;
7. import org.hibernate.ScrollableResults;
8. import org.hibernate.Session;
9. import org.hibernate.SessionFactory;
10. import org.hibernate.Transaction;
11. import org.hibernate.boot.registry.StandardServiceRegistry;
12. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
13. import org.hibernate.cfg.Configuration;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
14.  
15. import com.durgasoft.pojo.Employee;  
16.  
17. public class Test {  
18.  
19.     public static void main(String[] args) throws Exception{  
20.         Configuration cfg = new Configuration();  
21.         cfg.configure();  
22.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();  
23.         builder = builder.applySettings(cfg.getProperties());  
24.         StandardServiceRegistry registry = builder.build();  
25.         SessionFactory sessionFactory = cfg.buildSessionFactory(registry);  
26.         Session session = sessionFactory.openSession();  
27.  
28.         Query query1 = session.createQuery("update Employee set esal = esal + 500 where esal < 10000");  
29.         Transaction tx = session.beginTransaction();  
30.         int rowCount = query1.executeUpdate();  
31.         tx.commit();  
32.         System.out.println("Records Updated :"+rowCount);  
33.  
34.         Query query2 = session.createQuery("delete from Employee where esal < 10000");  
35.         Transaction tx = session.beginTransaction();  
36.         int rowCount = query2.executeUpdate();  
37.         tx.commit();  
38.         System.out.println("No of Records Deleted :"+rowCount);  
39.         session.close();  
40.         sessionFactory.close();  
41.     }  
42. }
```

In HQL , we are able to use "insert" query to copy multiple records from one table to another table , not to insert a record into the database table directly.

EX:

```
Query query = session.createQuery("insert into Employee2(eno,ename,esal,eaddr)select e.eno,  
e.ename, e.esal, e.eaddr from Employee1 as e");  
Transaction tx = session.beginTransaction();  
int rowCount = query.executeUpdate();  
tx.commit();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Example:**Employee1.java**

```
1. package com.durgasoft.pojo;
2.
3. public class Employee1 {
4.     private int eno;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.
9.     public int getEno() {
10.         return eno;
11.     }
12.     public void setEno(int eno) {
13.         this.eno = eno;
14.     }
15.     public String getEname() {
16.         return ename;
17.     }
18.     public void setEname(String ename) {
19.         this.ename = ename;
20.     }
21.     public float getEsal() {
22.         return esal;
23.     }
24.     public void setEsal(float esal) {
25.         this.esal = esal;
26.     }
27.     public String getEaddr() {
28.         return eaddr;
29.     }
30.     public void setEaddr(String eaddr) {
31.         this.eaddr = eaddr;
32.     }
33.
34.
35. }
```

Employee2.java

```
1. package com.durgasoft.pojo;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
2.  
3. public class Employee2 {  
4.     private int eno;  
5.     private String ename;  
6.     private float esal;  
7.     private String eaddr;  
8.  
9.     public int getEno() {  
10.         return eno;  
11.     }  
12.     public void setEno(int eno) {  
13.         this.eno = eno;  
14.     }  
15.     public String getEname() {  
16.         return ename;  
17.     }  
18.     public void setEname(String ename) {  
19.         this.ename = ename;  
20.     }  
21.     public float getEsal() {  
22.         return esal;  
23.     }  
24.     public void setEsal(float esal) {  
25.         this.esal = esal;  
26.     }  
27.     public String getEaddr() {  
28.         return eaddr;  
29.     }  
30.     public void setEaddr(String eaddr) {  
31.         this.eaddr = eaddr;  
32.     }  
33.  
34.  
35.}
```

Employee1.hbm.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>  
2. <!DOCTYPE hibernate-mapping PUBLIC  
3.     "-//Hibernate/Hibernate Mapping DTD 3.0//EN"  
4.     "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">  
5. <hibernate-mapping>  
6. <class name="com.durgasoft.pojo.Employee1" table="emp1">
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

7. <id name="eno" column="ENO"/>
8. <property name="ename" column="ENAME"/>
9. <property name="esal" column="ESAL"/>
10. <property name="eaddr" column="EADDR"/>
11.</class>
12.</hibernate-mapping>
```

Employee2.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="com.durgasoft.pojo.Employee2" table="emp2">
7.   <id name="eno" column="ENO"/>
8.   <property name="ename" column="ENAME"/>
9.   <property name="esal" column="ESAL"/>
10.  <property name="eaddr" column="EADDR"/>
11.</class>
12.</hibernate-mapping>
```

hibernate.cfg.xml

```

1. <!DOCTYPE hibernate-configuration PUBLIC
2.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
3.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
4. <hibernate-configuration>
5. <session-factory>
6.   <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
7.   <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
8.   <property name="connection.user">system</property>
9.   <property name="connection.password">durga</property>
10.  <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
11.  <property name="hibernate.show_sql">true</property>
12.  <mapping resource="Employee1.hbm.xml"/>
13.  <mapping resource="Employee2.hbm.xml"/>
14.</session-factory>
15.</hibernate-configuration>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Query;
4. import org.hibernate.Session;
5. import org.hibernate.SessionFactory;
6. import org.hibernate.Transaction;
7. import org.hibernate.boot.registry.StandardServiceRegistry;
8. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
9. import org.hibernate.cfg.Configuration;
10.
11. public class Test {
12.
13.     public static void main(String[] args) throws Exception {
14.         Configuration config = new Configuration();
15.         config.configure();
16.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
17.         builder = builder.applySettings(config.getProperties());
18.         StandardServiceRegistry registry = builder.build();
19.         SessionFactory sessionFactory = config.buildSessionFactory(registry);
20.         Session session = sessionFactory.openSession();
21.         Query query = session.createQuery("insert into Employee2(eno,ename,esal,eaddr)select e.eno,e.ename,e.esal,e.eaddr from Employee1 as e");
22.         Transaction tx = session.beginTransaction();
23.         int rowCount = query.executeUpdate();
24.         tx.commit();
25.         System.out.println("Employee details are transferred from emp1 to emp2");
26.         session.close();
27.         sessionFactory.close();
28.     }
29. }
```

The above Example with single mapping file with two pojo classes configuration

Example:**Employee1.java**

```

1. package com.durgasoft.pojo;
2.
3. public class Employee1 {
4.     private int eno;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

5. private String ename;
6. private float esal;
7. private String eaddr;
8.
9. public int getEno() {
10.    return eno;
11. }
12. public void setEno(int eno) {
13.    this.eno = eno;
14. }
15. public String getEname() {
16.    return ename;
17. }
18. public void setEname(String ename) {
19.    this.ename = ename;
20. }
21. public float getEsal() {
22.    return esal;
23. }
24. public void setEsal(float esal) {
25.    this.esal = esal;
26. }
27. public String getEaddr() {
28.    return eaddr;
29. }
30. public void setEaddr(String eaddr) {
31.    this.eaddr = eaddr;
32. }
33.
34.
35.}
```

Employee2.java

```

1. package com.durgasoft.pojo;
2.
3. public class Employee2 {
4.    private int eno;
5.    private String ename;
6.    private float esal;
7.    private String eaddr;
8.
9.    public int getEno() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10.    return eno;
11. }
12. public void setEno(int eno) {
13.     this.eno = eno;
14. }
15. public String getEname() {
16.     return ename;
17. }
18. public void setEname(String ename) {
19.     this.ename = ename;
20. }
21. public float getEsal() {
22.     return esal;
23. }
24. public void setEsal(float esal) {
25.     this.esal = esal;
26. }
27. public String getEaddr() {
28.     return eaddr;
29. }
30. public void setEaddr(String eaddr) {
31.     this.eaddr = eaddr;
32. }
33.
34.}
```

Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6. <class name="com.durgasoft.pojo.Employee1" table="emp1">
7.   <id name="eno" column="ENO"/>
8.   <property name="ename" column="ENAME"/>
9.   <property name="esal" column="ESAL"/>
10.  <property name="eaddr" column="EADDR"/>
11. </class>
12. <class name="com.durgasoft.pojo.Employee2" table="emp2">
13.   <id name="eno" column="ENO"/>
14.   <property name="ename" column="ENAME"/>
15.   <property name="esal" column="ESAL"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

16. <property name="eaddr" column="EADDR"/>
17. </class>
18. </hibernate-mapping>
```

hibernate.cfg.xml

```

1. <!DOCTYPE hibernate-configuration PUBLIC
2.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
3.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
4. <hibernate-configuration>
5. <session-factory>
6.   <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
7.   <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
8.   <property name="connection.user">system</property>
9.   <property name="connection.password">durga</property>
10.  <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>

11. <property name="hibernate.show_sql">true</property>
12. <mapping resource="Employee.hbm.xml"/>
13. <!-- <mapping resource="Employee2.hbm.xml"/> -->
14. </session-factory>
15. </hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Query;
4. import org.hibernate.Session;
5. import org.hibernate.SessionFactory;
6. import org.hibernate.Transaction;
7. import org.hibernate.boot.registry.StandardServiceRegistry;
8. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
9. import org.hibernate.cfg.Configuration;
10.
11. public class Test {
12.
13.     public static void main(String[] args) throws Exception {
14.         Configuration config = new Configuration();
15.         config.configure();
16.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
17.         builder = builder.applySettings(config.getProperties());
18.         StandardServiceRegistry registry = builder.build();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

19. SessionFactory sessionFactory = config.buildSessionFactory(registry);
20. Session session = sessionFactory.openSession();
21. Query query = session.createQuery("insert into Employee2(eno,ename,esal,eaddr)select e.eno,e.ename,e.esal,e.eaddr from Employee1 as e");
22. Transaction tx = session.beginTransaction();
23. int rowCount = query.executeUpdate();
24. tx.commit();
25. System.out.println("Employee details are transferred from emp1 to emp2");
26. session.close();
27. sessionFactory.close();
28. }
29.

```

Above Example With Annotations:**Employee1.java**

```

1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp1")
10. public class Employee1 {
11.     @Id
12.     @Column(name="ENO")
13.     private int eno;
14.     @Column(name="ENAME")
15.     private String ename;
16.     @Column(name="ESAL")
17.     private float esal;
18.     @Column(name="EADDR")
19.     private String eaddr;
20.
21.     public int getEno() {
22.         return eno;
23.     }
24.     public void setEno(int eno) {
25.         this.eno = eno;
26.     }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
27. public String getEname() {
28.     return ename;
29. }
30. public void setEname(String ename) {
31.     this.ename = ename;
32. }
33. public float getEsal() {
34.     return esal;
35. }
36. public void setEsal(float esal) {
37.     this.esal = esal;
38. }
39. public String getEaddr() {
40.     return eaddr;
41. }
42. public void setEaddr(String eaddr) {
43.     this.eaddr = eaddr;
44. }
45.
46.
47.}
```

Employee2.java



```
1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp2")
10. public class Employee2 {
11.     @Id
12.     @Column(name="ENO")
13.     private int eno;
14.     @Column(name="ENAME")
15.     private String ename;
16.     @Column(name="ESAL")
17.     private float esal;
18.     @Column(name="EADDR")
19.     private String eaddr;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

20.
21. public int getEno() {
22.     return eno;
23. }
24. public void setEno(int eno) {
25.     this.eno = eno;
26. }
27. public String getEname() {
28.     return ename;
29. }
30. public void setEname(String ename) {
31.     this.ename = ename;
32. }
33. public float getEsal() {
34.     return esal;
35. }
36. public void setEsal(float esal) {
37.     this.esal = esal;
38. }
39. public String getEaddr() {
40.     return eaddr;
41. }
42. public void setEaddr(String eaddr) {
43.     this.eaddr = eaddr;
44. }
45.
46.
47.}
```

hibernate.cfg.xml

```

1. <!DOCTYPE hibernate-configuration PUBLIC
2.      "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
3.      "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
4. <hibernate-configuration>
5.   <session-factory>
6.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
7.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
8.     <property name="connection.user">system</property>
9.     <property name="connection.password">durga</property>
10.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
11.    <property name="hibernate.show_sql">true</property>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

12. <!--
13. <mapping resource="Employee.hbm.xml"/>
14. <mapping resource="Employee2.hbm.xml"/>
15. -->
16. <mapping class="com.durgasoft.pojo.Employee1"/>
17. <mapping class="com.durgasoft.pojo.Employee2"/>
18.</session-factory>
19.</hibernate-configuration>
```



Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Query;
4. import org.hibernate.Session;
5. import org.hibernate.SessionFactory;
6. import org.hibernate.Transaction;
7. import org.hibernate.boot.registry.StandardServiceRegistry;
8. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
9. import org.hibernate.cfg.Configuration;
10.
11. public class Test {
12.
13.     public static void main(String[] args) throws Exception {
14.         Configuration config = new Configuration();
15.         config.configure();
16.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
17.         builder = builder.applySettings(config.getProperties());
18.         StandardServiceRegistry registry = builder.build();
19.         SessionFactory sessionFactory = config.buildSessionFactory(registry);
20.         Session session = sessionFactory.openSession();
21.         Query query = session.createQuery("insert into Employee2(eno,ename,esal,eaddr)select e.eno,e.ename,e.esal,e.eaddr from Employee1 as e");
22.         Transaction tx = session.beginTransaction();
23.         int rowCount = query.executeUpdate();
24.         tx.commit();
25.         System.out.println("Employee details are transferred from emp2 to emp1");
26.         session.close();
27.         sessionFactory.close();
28.     }
29. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Building Blocks for HQL queries:

To prepare HQL queries what are the various elements we have to provide

1. Clauses
2. Aggregate Functions
3. Generic Expressions
4. Parameters
5. Subqueries

1. Clauses:

These are building blocks to HQL queries, which are able to specify POJO classes property names, conditional expressions,.....

EX:

```
from  
select  
where  
order by  
group by  
having  
---  
---
```

1. from:

It can be used to specify POJO class names and their alias names in HQL queries.

Syntax:

```
From POJO_Class_Name [[AS] var_Name]
```

EX:

```
From Employee  
FROM Employee  
from Employee e  
From Employee AS e
```

2. select:

This clause can be used to specify POJO class properties names inorder to retrive individual

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

column values. If we provide select clause with individual POJO class properties in HQL query then results are generated in the form of Object[].

Syntax:

[select prop_Names] from POJO_Class_Name [[AS] var_Name]

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employee AS e");
2. List<Object[]> list = query.list();
3. System.out.println("ENO\tENAME\tESAL\tEADDR");
4. System.out.println("-----");
5. for(Object[] obj: list) {
6. for(Object o: obj) {
7. System.out.print(o+"\t");
8. }
9. System.out.println();
10.

```

3. Where clause:

In HQL, where clause can be used to provide a particular conditional expression inorder to retrieve the results as per the condition.

Syntax:

[select prop_Names] from POJO_Class_Name [[AS] var_Name][where condition]

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employee WHERE e.esal<=7000");
2. List<Object[]> list = query.list();
3. System.out.println("ENO\tENAME\tESAL\tEADDR");
4. System.out.println("-----");
5. for(Object[] obj: list) {
6. for(Object o: obj) {
7. System.out.print(o+"\t");
8. }
9. System.out.println();

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

10. }

4. Order by:

It can be used to retrieve all results either in ascending order or in descending order w.r.t a particular column. In HQL, by default, all results are generated in Ascending order only.

Syntax:

```
[select prop_Names] from POJO_Class_Name [[AS] var_Name][where condition][order by prop_Name asc/desc]
```

EX:

1. Query `query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employee AS e where e.esal<=10000 order by e.ename desc");`
2. `List<Object[]> list = query.list();`
3. `System.out.println("ENO\tENAME\tESAL\tEADDR");`
4. `System.out.println("-----");`
5. `for(Object[] obj: list) {`
6. `for(Object o: obj) {`
7. `System.out.print(o+"\t");`
8. `}`
9. `System.out.println();`
10. `}`

5. group by:

This clause can be used to specify groups over the retrieved results w.r.t a particular column.
Note: In HQL queries, we are able to implement group by clause with the combination of aggregate functions only.

Syntax:

```
[select prop_Names] from POJO_Class_Name [[AS] var_Name][where condition][order by prop_Name asc/desc]
[group by column_Name]
```

EX:

1. Query `query = session.createQuery("select sum(e.esal) FROM Employee AS e group by e.ename");`

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2. List<Double> list = query.list();
3. for(Double val : list) {
4. System.out.println(val);
5. }

```

6.having:

The main intention of having clause is to implement a conditional expression while including elements in groups as per group by clause.

Syntax:

[select prop_Names] from POJO_Class_Name [[AS] var_Name][where condition][order by prop_Name asc/desc]
 [group by column_Name][having Condition]

EX:

```

1. Query query = session.createQuery("select count(e.esal) FROM Employee AS e group by
e.esal having e.esal<=8000");
2. List<Long> list = query.list();
3. for(Long val : list) {
4. System.out.println(val);
5. }

```

2. Aggregate Functions:

The main intention of Aggregate functions is to provide small arithmetic calculations over the results.

EX:

1. count: It able to count the no of results are generated from HQL query and the resultant count value is generated in the form of java.lang.Long type.

EX:

```

1. Query query = session.createQuery("select count(e.esal) FROM Employee AS e");
2. List<Long> list = query.list();
3. for(Long val : list) {
4. System.out.println(val);
5. }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2.sum: It able to perform addition operation over the results which are generated from HQL query.

EX:

```
1. Query query = session.createQuery("select sum(e.esal) FROM Employee AS e");
2. List<Double> list = query.list();
3. for(Double val : list) {
4. System.out.println(val);
5. }
```

3. min: It able to get min value over all the results which are generated from HQL query.

EX:

```
1. Query query = session.createQuery("select min(e.esal) FROM Employee AS e");
2. List<Float> list = query.list();
3. for(Float val : list) {
4. System.out.println(val);
5. }
```

4. max: It able to get max value over all the results which are generated from HQL query.

EX:

```
1. Query query = session.createQuery("select max(e.esal) FROM Employee AS e");
2. List<Float> list = query.list();
3. for(Float val : list) {
4. System.out.println(val);
5. }
```

5. avg: It able to generate average value over all the generated results from HQL query.

EX:

```
1. Query query = session.createQuery("select avg(e.esal) FROM Employee AS e");
2. List<Double> list = query.list();
3. for(Double val : list) {
4. System.out.println(val);
5. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3. Generic Expressions

The main intention of Generic Expressions is to provide expressions in HQL queries inorder to perform simple Arithmetic calculations, comparisions,.....

To provide generic expressions in HQL queries we have to use the following elements.

a)Arithmetic Operators: +, -, *, /, %,.....

EX:

1. Query `query = session.createQuery("select (e.esal-500) FROM Employee AS e");`
2. `List<Float> list = query.list();`
3. `for(Float val : list) {`
4. `System.out.println(val);`
5. `}`

b)Comparision Operators: ==, !=, <, >, <=, >= ,....

c)Logical Operators: && or AND, || or OR ,....

EX:

1. Query `query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employee AS e where e.esal>=6000 AND e.esal<=8000");`
2. `List<Object[]> list = query.list();`
3. `System.out.println("ENO\tENAME\tESAL\tEADDR");`
4. `System.out.println("-----");`
5. `for(Object[] val : list) {`
6. `for(Object o: val) {`
7. `System.out.print(o+"\t");`
8. `}`
9. `System.out.println();`
10. `}`

d)Scalar Functions:

`lower():` To get results in lower case letters.

`upper():` To get results in Upper case letters.

EX:

1. Query `query = session.createQuery("select e.eno, lower(e.ename), e.esal, upper(e.eaddr) FROM Employee AS e");`

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2. List<Object[]> list = query.list();
3. System.out.println("ENO\tENAME\tESAL\tEADDR");
4. System.out.println("-----");
5. for(Object[] val : list) {
6. for(Object o: val) {
7. System.out.print(o+"\t");
8. }
9. System.out.println();
10.

```

e) IN: It will be used along with where clause to specify a list of values inorder to retrive matched results.

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employ
ee AS e where e.ename IN ('BBB', 'CCC')");
2. List<Object[]> list = query.list();
3. System.out.println("ENO\tENAME\tESAL\tEADDR");
4. System.out.println("-----");
5. for(Object[] val : list) {
6. for(Object o: val) {
7. System.out.print(o+"\t");
8. }
9. System.out.println();
10.

```

f) BETWEEN : It able to specify min value and max value inorder to retrive the results which are between the specified min value and max value.

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employ
ee AS e where e.ename BETWEEN 'BBB' and 'DDD'");
2. List<Object[]> list = query.list();
3. System.out.println("ENO\tENAME\tESAL\tEADDR");
4. System.out.println("-----");
5. for(Object[] val : list) {
6. for(Object o: val) {
7. System.out.print(o+"\t");
8. }
9. System.out.println();
10.

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

g) LIKE: It able to provide a particular pattern in HQL query inorder to retrive matched results.

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employ
ee AS e where e.ename LIKE 'B%'");
2. List<Object[]> list = query.list();
3. System.out.println("ENO\tENAME\tESAL\tEADDR");
4. System.out.println("-----");
5. for(Object[] val : list) {
6. for(Object o: val) {
7. System.out.print(o+"\t");
8. }
9. System.out.println();
10.

```

h) IS NULL: It able to retrive all the results from database table w.r.t a particular column whose value is null.

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employ
ee AS e where e.ename IS NULL");
2. List<Object[]> list = query.list();
3. System.out.println("ENO\tENAME\tESAL\tEADDR");
4. System.out.println("-----");
5. for(Object[] val : list) {
6. for(Object o: val) {
7. System.out.print(o+"\t");
8. }
9. System.out.println();
10.

```

i) IS NOT NULL: It able to retrive all the results from database table w.r.t a particular column whose value is not null.

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employ
ee AS e where e.ename IS NOT NULL");
2. List<Object[]> list = query.list();
3. System.out.println("ENO\tENAME\tESAL\tEADDR");

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

4. System.out.println("-----");
5. for(Object[] val : list) {
6. for(Object o: val) {
7. System.out.print(o+"\t");
8. }
9. System.out.println();
10.}

```

4. Parameters

The main intention of parameters in HQL queries is to take dynamic values in HQL queries.
In HQL, there are two types of parameters

1. Positional parameters
2. Named Parameters

1. Positional parameters:

These parameters are represented in the form of '?' in HQL queries.

After specifying these parameters in HQL queries we must set values to these parameters, to set values to positional parameters we have to use the following method.

`public void setParameter(int param_Index, xxx value)`

Where `param_Index` may start with 0.

Where `xxx` may be byte, short, int,....

Note: In JDBC, positional parameter indexes will start from 1 , but, in HQL parameters indexes will start from 0.

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employ
ee AS e where e.esal<?");
2. query.setParameter(0, 10000.0f);
3. List<Object[]> list = query.list();
4. System.out.println("ENO\tENAME\tESAL\tEADDR");
5. System.out.println("-----");
6. for(Object[] val : list) {
7. for(Object o: val) {
8. System.out.print(o+"\t");
9. }
10. System.out.println();
11.}

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

In HQL queries, we are able to provide more than one positional parameter.

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employee e where e.esal>=? and e.esal<=?");
2. query.setParameter(0, 6000.0f);
3. query.setParameter(1, 8000.0f);
4. List<Object[]> list = query.list();
5. System.out.println("ENO\tENAME\tESAL\tEADDR");
6. System.out.println("-----");
7. for(Object[] val : list) {
8. for(Object o: val) {
9. System.out.print(o+"\t");
10. }
11. System.out.println();
12. }
```

2. Named Parameters:

These parameters are represented in the form of ':Param_Name' in HQL queries, after providing named parameters in HQL query we have to set values to named parameters, for this, we have to use the following method.

```
public void setXXX(String param_name, xxx value)
Where xxx may be byte, short, int, String,.....
```

Note: in HQL queries, we are able to provide more than one named parameters.

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employee e where e.esal>=:min_Sal and e.esal<=:max_Sal");
2. query.setFloat("min_Sal", 6000.0f);
3. query.setFloat("max_Sal", 8000.0f);
4. List<Object[]> list = query.list();
5. System.out.println("ENO\tENAME\tESAL\tEADDR");
6. System.out.println("-----");
7. for(Object[] val : list) {
8. for(Object o: val) {
9. System.out.print(o+"\t");
10. }
11. System.out.println();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

12. }

In Hibernate applications we are able to provide both positional parameters and named parameters with in a single HQL query, but, first we have to provide all positional parameters after that only we have to provide named parameters, we must not provide any positional parameter after named parameter.

EX:

```

1. Query query = session.createQuery("select e.eno, e.ename, e.esal, e.eaddr FROM Employ
   ee AS e where e.esal>=? and e.esal<=:max_Sal");
2. query.setParameter(0, 6000.0f);
3. query.setFloat("max_Sal", 8000.0f);
4. List<Object[]> list = query.list();
5. System.out.println("ENO\tENAME\tESAL\tEADDR");
6. System.out.println("-----");
7. for(Object[] val : list) {
8. for(Object o: val) {
9. System.out.print(o+"\t");
10.}
11. System.out.println();
12.}
```

If we provide named parameter before positional parameter ion HQL query then we are able to get the following error or Exception.

ERROR: cannot define positional parameter after any named parameters have been defined

5. Subqueries

Writing a query in another query is called as Sub Query.
HQL is supporting sub queries also.

EX:

```

1. Query query = session.createQuery("select e1.eno, e1.ename, e1.esal, e1.eaddr FROM E
   mployee AS e1 where e1.esal<(select max(e2.esal) from Employee e2)");
2. List<Object[]> list = query.list();
3. System.out.println("ENO\tENAME\tESAL\tEADDR");
4. System.out.println("-----");
5. for(Object[] val : list) {
6. for(Object o: val) {
7. System.out.print(o+"\t");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

8. }
9. System.out.println();
10.

```

6. Pagination

The process of displaying results in more than one page is called as Pagination.
Displaying 3 results in a page like three pages out of 9 results is called as "Pagination".

We are able to provide Pagination in Hibernate applications by using `setFirstResult()` and `setMaxResult()` methods over Query object.

Example:

form.html

```

1. <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
2. <html>
3. <head>
4. <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
5. <title>Insert title here</title>
6. </head>
7. <body>
8. <form method="POST" action=".display">
9. <center>
10. <br><br><br>
11. <input type="submit" value="1" name="button">
12. <input type="submit" value="2" name="button">
13. <input type="submit" value="3" name="button">
14. </center>
15. </form>
16. </body>
17. </html>

```

DisplayServlet.java

```

1. package com.durgasoft.servlets;
2.
3. import java.io.IOException;
4. import java.io.PrintWriter;
5. import java.util.List;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
6.  
7. import javax.servlet.RequestDispatcher;  
8. import javax.servlet.ServletException;  
9. import javax.servlet.http.HttpServlet;  
10. import javax.servlet.http.HttpServletRequest;  
11. import javax.servlet.http.HttpServletResponse;  
12.  
13. import com.durgasoft.beans.Employee;  
14. import com.durgasoft.service.EmployeeService;  
15. public class DisplayServlet extends HttpServlet {  
16.     private static final long serialVersionUID = 1L;  
17.     protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {  
18.         try {  
19.             response.setContentType("text/html");  
20.             PrintWriter out = response.getWriter();  
21.             int label = Integer.parseInt(request.getParameter("button"));  
22.  
23.             EmployeeService empService = new EmployeeService();  
24.             List<Employee> list = empService.getEmployees(label);  
25.             out.println("<html>");  
26.             out.println("<body>");  
27.             out.println("<center>");  
28.             out.println("<table border='1'>");  
29.             out.println("<tr><th>ENO</th><th>ENAME</th><th>ESAL</th><th>EADDR</th></tr>");  
30.             for(Employee emp : list) {  
31.                 out.println("<tr>");  
32.                 out.println("<td>" + emp.getEno() + "</td><td>" + emp.getEname() + "</td><td>" + emp  
33. .getEsal() + "</td><td>" + emp.getEaddr() + "</td>");  
34.                 out.println("</tr>");  
35.             }  
36.             out.println("</table></center></body></html>");  
37.             RequestDispatcher rd = request.getRequestDispatcher("/form.html");  
38.             rd.include(request, response);  
39.         } catch (Exception e) {  
40.             e.printStackTrace();  
41.         }  
42.     }  
43.  
44.}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EmployeeService.java

```
1. package com.durgasoft.service;
2.
3. import java.util.List;
4.
5. import org.hibernate.Query;
6. import org.hibernate.Session;
7. import org.hibernate.SessionFactory;
8.
9. import com.durgasoft.beans.Employee;
10. import com.durgasoft.util.HibernateUtil;
11.
12. public class EmployeeService {
13.     List<Employee> list;
14.     SessionFactory sessionFactory;
15.     Session session;
16.     Query query;
17.     public EmployeeService() {
18.         try {
19.             sessionFactory = HibernateUtil.getSessionFactory();
20.             session = sessionFactory.openSession();
21.             query = session.createQuery("from Employee");
22.             query.setMaxResults(3);
23.         } catch (Exception e) {
24.             e.printStackTrace();
25.         }
26.     }
27.     public List<Employee> getEmployees(int label){
28.         try {
29.             if(label == 1) {
30.                 query.setFirstResult(0);
31.             }
32.             if(label == 2) {
33.                 query.setFirstResult(3);
34.             }
35.             if(label == 3) {
36.                 query.setFirstResult(6);
37.             }
38.             list = query.list();
39.         } catch (Exception e) {
40.             e.printStackTrace();
41.         }
42.     }
43. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
42.  
43.     return list;  
44. }  
45.}
```

HibernateUtil.java

```
1. package com.durgasoft.util;  
2.  
3. import org.hibernate.SessionFactory;  
4. import org.hibernate.boot.registry.StandardServiceRegistry;  
5. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;  
6. import org.hibernate.cfg.Configuration;  
7.  
8. public class HibernateUtil {  
9.     private static SessionFactory sessionFactory;  
10.    static {  
11.        try {  
12.            Configuration config = new Configuration();  
13.            config.configure();  
14.            StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();  
15.            builder = builder.applySettings(config.getProperties());  
16.            StandardServiceRegistry registry = builder.build();  
17.            sessionFactory = config.buildSessionFactory(registry);  
18.        } catch (Exception e) {  
19.            e.printStackTrace();  
20.        }  
21.    }  
22.    public static SessionFactory getSessionFactory() {  
23.        return sessionFactory;  
24.    }  
25.}
```

Employee.java

```
1. package com.durgasoft.beans;  
2.  
3. import javax.persistence.Column;  
4. import javax.persistence.Entity;  
5. import javax.persistence.Id;  
6. import javax.persistence.Table;  
7.  
8. @Entity
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9. @Table(name="emp1")
10. public class Employee {
11.     @Id
12.     @Column(name="ENO")
13.     private int eno;
14.     @Column(name="ENAME")
15.     private String ename;
16.     @Column(name="ESAL")
17.     private float esal;
18.     @Column(name="EADDR")
19.     private String eaddr;
20.
21.     public int getEno() {
22.         return eno;
23.     }
24.     public void setEno(int eno) {
25.         this.eno = eno;
26.     }
27.     public String getEname() {
28.         return ename;
29.     }
30.     public void setEname(String ename) {
31.         this.ename = ename;
32.     }
33.     public float getEsal() {
34.         return esal;
35.     }
36.     public void setEsal(float esal) {
37.         this.esal = esal;
38.     }
39.     public String getEaddr() {
40.         return eaddr;
41.     }
42.     public void setEaddr(String eaddr) {
43.         this.eaddr = eaddr;
44.     }
45.
46.
47.}
```

hibernate.cfg.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <!--
7.     <session-factory>
8.       <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
9.       <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
10.      <property name="connection.user">system</property>
11.      <property name="connection.password">durga</property>
12.      <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
    y>
13.      <property name="show_Sql">true</property>
14.      <mapping class="com.durgasoft.beans.Employee"/>
15.    </session-factory>
16.  -->
17.  <session-factory>
18.    <property name="connection.driver_Class">com.mysql.jdbc.Driver</property>
19.    <property name="connection.url">jdbc:mysql://localhost:3306/durgadb</property>
20.    <property name="connection.user">root</property>
21.    <property name="connection.password">root</property>
22.    <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
23.    <property name="show_Sql">true</property>
24.    <mapping class="com.durgasoft.beans.Employee"/>
25.  </session-factory>
26. </hibernate-configuration>
```

web.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-
   instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.c
   om/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-
   app_2_5.xsd" id="WebApp_ID" version="2.5">
3.   <display-name>paginationapp</display-name>
4.   <welcome-file-list>
5.     <welcome-file>index.html</welcome-file>
6.     <welcome-file>index.htm</welcome-file>
7.     <welcome-file>index.jsp</welcome-file>
8.     <welcome-file>default.html</welcome-file>
9.     <welcome-file>default.htm</welcome-file>
10.    <welcome-file>default.jsp</welcome-file>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

11. <welcome-file-list>
12. <servlet>
13. <description></description>
14. <display-name>DisplayServlet</display-name>
15. <servlet-name>DisplayServlet</servlet-name>
16. <servlet-class>com.durgasoft.servlets.DisplayServlet</servlet-class>
17. </servlet>
18. <servlet-mapping>
19. <servlet-name>DisplayServlet</servlet-name>
20. <url-pattern>/display</url-pattern>
21. </servlet-mapping>
22.</web-app>

```

2. Native SQL:

- ★ In Hibernate applications, by using Session we are able to perform database operations on only single record, but, if we want to perform database operations over multiple records then we have to use HQL, but, HQL is not providing environment for Database dependent native operations.
- ★ HQL is able to provide support for DML[insert, update, delete, select] operations, but, it has not provided environment for DDL[create, alter and drop] queries.
- ★ HQL is not supporting stored procedures and functions kind of database dependent native operations.
- ★ In Hibernate applications, If we want to perform database operations over multiple records , database dependent native operations like preparing stored procedures, functions and accessing them and to perform DDL operations,... Hibernate has provided an alternative for HQL , that is, "Native SQL".

If we want to use Native SQL in Hibernate applications then we have to use the following steps.

1. Create SqlQuery object
2. Execute the SQL query.

1) Create SqlQuery object:

SqlQuery is an object provided by Hibernate in the form of org.hibernate.SqlQuery interface and it is able to represent a native sql query.

To create SqlQuery object we have to use the following method.

```
public SqlQuery createSqlQuery(String query) throws HibernateException
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX: `SqlQuery query = session.createSqlQuery("select * from emp1");`

2. Execute the SQL query:

To execute Sql Query represent by `SqlQuery` object we have to use the following methods.

```
public List list()  
public Iterator iterator()  
public ScrollableResults scroll()  
public Object uniqueResult()  
public int executeUpdate()
```

In Native SQL, there are two types of SQL queries.

1. Entity SQL Queries
2. Scalar SQL Queries

1) Entity SQL Query

Entity SQLQueries are database dependent sql queries, it can be used to retrive the complete records in the form of an Entity. It will include '*' notation to get all columns data in a record in the form of Entity object.

EX: `SqlQuery query = session.createSqlQuery("select * from emp1");`

Before executing the query we have to provide entity type to `SqlQuery` object inorder to store results, for this, we have to use the following method.

```
public void addEntity(Class cl)
```

EX: `query.addEntity(com.durgasoft.hibernate.Employee.class);`

Example:

Employee.java

```
1. package com.durgasoft.pojo;  
2.  
3. import javax.persistence.Column;  
4. import javax.persistence.Entity;  
5. import javax.persistence.Id;  
6. import javax.persistence.Table;  
7.  
8. @Entity
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

 US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9. @Table(name="emp1")
10. public class Employee {
11.     @Id
12.     @Column(name="ENO")
13.     private int eno;
14.     @Column(name="ENAME")
15.     private String ename;
16.     @Column(name="ESAL")
17.     private float esal;
18.     @Column(name="EADDR")
19.     private String eaddr;
20.
21.     public int getEno() {
22.         return eno;
23.     }
24.     public void setEno(int eno) {
25.         this.eno = eno;
26.     }
27.     public String getEname() {
28.         return ename;
29.     }
30.     public void setEname(String ename) {
31.         this.ename = ename;
32.     }
33.     public float getEsal() {
34.         return esal;
35.     }
36.     public void setEsal(float esal) {
37.         this.esal = esal;
38.     }
39.     public String getEaddr() {
40.         return eaddr;
41.     }
42.     public void setEaddr(String eaddr) {
43.         this.eaddr = eaddr;
44.     }
45.
46.
47.}
```

hibernate.cfg.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.  <property name="show_Sql">true</property>
13.  <mapping class="com.durgasoft.pojo.Employee"/>
14. </session-factory>
15.</hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import java.util.List;
4.
5. import org.hibernate.SQLQuery;
6. import org.hibernate.Session;
7. import org.hibernate.SessionFactory;
8. import org.hibernate.boot.registry.StandardServiceRegistry;
9. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
10. import org.hibernate.cfg.Configuration;
11.
12. import com.durgasoft.pojo.Employee;
13.
14. public class Test {
15.
16.   public static void main(String[] args) throws Exception{
17.     Configuration cfg = new Configuration();
18.     cfg.configure();
19.     StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
20.     builder = builder.applySettings(cfg.getProperties());
21.     StandardServiceRegistry registry = builder.build();
22.     SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
23.     Session session = sessionFactory.openSession();
24.     SQLQuery query = session.createSQLQuery("select * from emp1");
25.     query.addEntity(com.durgasoft.pojo.Employee.class);
```

**CONTACT US:**

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

26.     List<Employee> list = query.list();
27.     System.out.println("ENO\tENAME\tESAL\tEADDR");
28.     System.out.println("-----");
29.     for(Employee e : list) {
30.         System.out.print(e.getEno()+"\t");
31.         System.out.print(e.getEname()+"\t");
32.         System.out.print(e.getEsal()+"\t");
33.         System.out.print(e.getEaddr()+"\n");
34.     }
35.
36.     session.close();
37.     sessionFactory.close();
38. }
39.

```

In Native SQL, there are two types of parameters.

1. Positional Parameters
2. Named Parameters

★ Positional parameters are represented in the form of '?'s , we are able to provide more than one positional parameter with in a single sql query. To set values to the positional parameters we have to use the following method from SqlQuery.

public void setXXX(int param_Index, XXX value)
Where xxx may be byte, short, int,...

★ Names parameters are represented in the form of ':name' , we are able to provide more than one named parameter in native sql query. To provide values to the named parameters we have to use the following method.

public void setXXX(String param_Name, xxx value)
Where xxx may be byte, short, int,....

★ In a single native sql query, we are able to provide both positional parameters and named parameters, but, first we must provide all positional parameters after that only we have to provide named parameters, we must not provide any positional parameter after named parameter.

EX:

1. SQLQuery query = session.createSQLQuery("select * from emp1 where esal>=? and esal<=?");
2. query.setFloat(0, 6000);
3. query.setFloat("max_Sal", 8000);

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

4. query.addEntity(com.durgasoft.pojo.Employee.class);
5. List<Employee> list = query.list();
6. System.out.println("ENO\tENAME\tESAL\tEADDR");
7. System.out.println("-----");
8. for(Employee e : list) {
9.     System.out.print(e.getEno()+"\t");
10.    System.out.print(e.getEname()+"\t");
11.    System.out.print(e.getEsal()+"\t");
12.    System.out.print(e.getEaddr()+"\n");
13.}

```

- ★ In the above approach, we have declare sql query directly in client application, it is available upto the present client application only, it is not available to other client applications, this approach is called as "Programmatic Approach".
- ★ In Hibernate applications, if we want use the same sql query in more than one client application then programmatic approach is not suggestible, at each and every client application we have to hardcode the query, it is not suggestible, to overcome this problem we have to use "Declarative approach".
- ★ In Declarative approach, we will declare native sql query in mapping file along with a particular logical name and we will get that query from mapping file on the basis of the name, this type of sql queries are called as Named SQL Queries.

To declare sql query in mapping file we have to use the following tags in mapping file.

```

1. <hibernate-mapping>
2. -----
3. <sql-query name="-->
4.   <return class="-->
5.   ---sql query-----
6. </sql-query>
7.
8. </hibernate-mapping>

```

'name' attribute in <sql-query> tag will take logical name to the query.

<return> tag will take a pojo class name with 'class' attribute inorder to get results in the form of POJO objects.

To get named sql query from mapping file to hibernate Client Application we have to use the following method.

```
public Query getNamedQuery(String logical_Name)
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

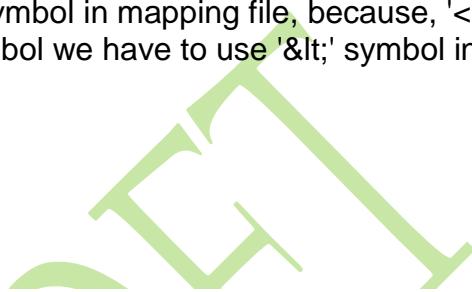
In Declarative native sql query we are able to provide both positional parameters and named parameters depending on the requirement.

Note: In Native SQL queueris we are unable to use '<' symbol in mapping file, because, '<' symbol is treated as starting tag form xml tags, in place of '<' symbol we have to use '<' symbol in mapping file.

EX:

Employee.java

```
1. package com.durgasoft.pojo;
2.
3. public class Employee {
4.     private int eno;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.
9.     public int getEno() {
10.         return eno;
11.     }
12.    public void setEno(int eno) {
13.        this.eno = eno;
14.    }
15.    public String getEname() {
16.        return ename;
17.    }
18.    public void setEname(String ename) {
19.        this.ename = ename;
20.    }
21.    public float getEsal() {
22.        return esal;
23.    }
24.    public void setEsal(float esal) {
25.        this.esal = esal;
26.    }
27.    public String getEaddr() {
28.        return eaddr;
29.    }
30.    public void setEaddr(String eaddr) {
31.        this.eaddr = eaddr;
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
32. }
33.}
```

Employee.hbm.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Employee" table="emp1">
7.     <id name="eno"/>
8.     <property name="ename"/>
9.     <property name="esal"/>
10.    <property name="eaddr"/>
11.   </class>
12.   <sql-query name="sql_Query">
13.     <return class="com.durgasoft.pojo.Employee"/>
14.     select * from emp1 where esal > ? and esal < :max_Sal
15.   </sql-query>
16. </hibernate-mapping>
```

hibernate.cfg.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.    <property name="show_Sql">true</property>
13.    <mapping resource="Employee.hbm.xml"/>
14.    <!-- <mapping class="com.durgasoft.pojo.Employee"/> -->
15.  </session-factory>
16. </hibernate-configuration>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```
1. package com.durgasoft.test;
2.
3. import java.util.List;
4.
5. import org.hibernate.Query;
6. import org.hibernate.SQLQuery;
7. import org.hibernate.Session;
8. import org.hibernate.SessionFactory;
9. import org.hibernate.boot.registry.StandardServiceRegistry;
10. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
11. import org.hibernate.cfg.Configuration;
12.
13. import com.durgasoft.pojo.Employee;
14.
15. public class Test {
16.
17.     public static void main(String[] args) throws Exception{
18.         Configuration cfg = new Configuration();
19.         cfg.configure();
20.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
21.         builder = builder.applySettings(cfg.getProperties());
22.         StandardServiceRegistry registry = builder.build();
23.         SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
24.         Session session = sessionFactory.openSession();
25.         Query query = session.getNamedQuery("sql_Query");
26.         query.setFloat(0, 6000);
27.         query.setFloat("max_Sal", 8000);
28.         List<Employee> list = query.list();
29.         System.out.println("ENO\tENAME\tESAL\tEADDR");
30.         System.out.println("-----");
31.         for(Employee e : list) {
32.             System.out.print(e.getEno()+"\t");
33.             System.out.print(e.getEname()+"\t");
34.             System.out.print(e.getEsal()+"\t");
35.             System.out.print(e.getEaddr()+"\n");
36.         }
37.
38.         session.close();
39.         sessionFactory.close();
40.     }
41. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

2) Scalar SQL Queries:

It is a native SQL query, it able to retrieve records data from individual columns and it able to generate results in the form of Object[].

EX: select eno, ename, esal, eaddr from emp1;

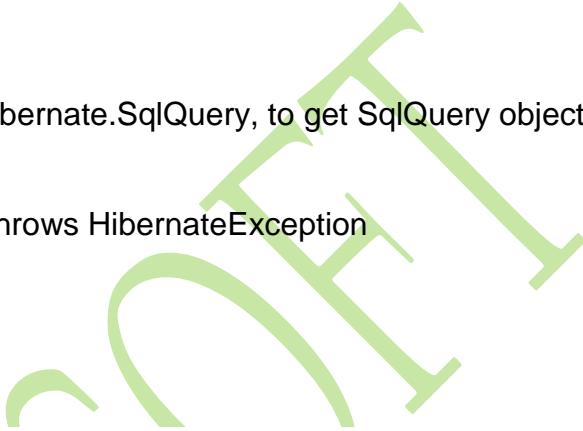
To represent scalar sql queries we will use org.hibernate.SqlQuery, to get SqlQuery object we will use the following method.

```
public SqlQuery createSqlQuery(String query) throws HibernateException
```

Example:

Employee.java

```
1. package com.durgasoft.pojo;
2. public class Employee {
3.     private int eno;
4.     private String ename;
5.     private float esal;
6.     private String eaddr;
7.
8.     public int getEno() {
9.         return eno;
10.    }
11.    public void setEno(int eno) {
12.        this.eno = eno;
13.    }
14.    public String getEname() {
15.        return ename;
16.    }
17.    public void setEname(String ename) {
18.        this.ename = ename;
19.    }
20.    public float getEsal() {
21.        return esal;
22.    }
23.    public void setEsal(float esal) {
24.        this.esal = esal;
25.    }
26.    public String getEaddr() {
27.        return eaddr;
28.    }
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

28. }
29. public void setEaddr(String eaddr) {
30.     this.eaddr = eaddr;
31. }
32.
33.

```

Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Employee" table="emp1">
7.     <id name="eno"/>
8.     <property name="ename"/>
9.     <property name="esal"/>
10.    <property name="eaddr"/>
11.   </class>
12.
13.</hibernate-mapping>

```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.    <property name="show_Sql">true</property>
13.    <mapping resource="Employee.hbm.xml"/>
14.  </session-factory>
15.</hibernate-configuration>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```

1. package com.durgasoft.test;
2.
3. import java.util.List;
4.
5. import org.hibernate.Query;
6. import org.hibernate.SQLQuery;
7. import org.hibernate.Session;
8. import org.hibernate.SessionFactory;
9. import org.hibernate.boot.registry.StandardServiceRegistry;
10. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
11. import org.hibernate.cfg.Configuration;
12.
13. public class Test {
14.
15.     public static void main(String[] args) throws Exception{
16.         Configuration cfg = new Configuration();
17.         cfg.configure();
18.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
19.         builder = builder.applySettings(cfg.getProperties());
20.         StandardServiceRegistry registry = builder.build();
21.         SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
22.         Session session = sessionFactory.openSession();
23.         SQLQuery query = session.createSQLQuery("select eno, ename, esal, eaddr from em
p1");
24.         List<Object[]> list = query.list();
25.         System.out.println("ENO\tENAME\tESAL\tEADDR");
26.         System.out.println("-----");
27.         for(Object[] obj: list) {
28.             System.out.println(obj[0]+\t+obj[1]+\t+obj[2]+\t+obj[3]);
29.         }
30.
31.         session.close();
32.         sessionFactory.close();
33.     }
34. }
```

- ★ In Scalar SQL Queries we are able to provide both Positional parameters and Named parameters ,but, first we must provide positional parameters after that only we must provide named parameters.
- ★ If we provide positional parameters and named parameters in sql query then we must provide values to these parameters , for this, we must use the following methods.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
public void setXXX(int index, xxx value)
public void setXXX(String param_Namem , xxx value)
```

EX:

```
1. SQLQuery query = session.createSQLQuery("select eno, ename, esal, eaddr from emp1 w
here esal >= ? and esal <= :max_Sal");
2. query.setFloat(0, 6000);
3. query.setFloat("max_Sal", 8000);
4. List<Object[]> list = query.list();
5. System.out.println("ENO\tENAME\tESAL\tEADDR");
6. System.out.println("-----");
7. for(Object[] obj: list) {
8.     System.out.println(obj[0]+"\t"+obj[1]+"\t"+obj[2]+"\t"+obj[3]);
9. }
```

In Hibernate applications, we are able to provide scalar sql queries in declarative manner in mapping file. To declare scalar sql queries in mapping file we have to use the following syntax.

```
1. <hibernate-mapping>
2. -----
3. <sql-query name="-->
4.   <return-scalar column="--" type="--"/>
5. -----
6.   -----scalar sql query -----
7. </sql-query>
8. -----
9. </hibernate-mapping>
```

where <return-scalar> tag is able to represent a particular scalar[column name], for which, we have to declare data type by using 'type' attribute.

Note: In Named scalar sql query we are able to provide both positional parameters and named parameters, first we have to provide positional parameters after that we have to provide named parameters.

Example:

Employee.java

```
1. package com.durgasoft.pojo;
2.
3. public class Employee {
4.     private int eno;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

5. private String ename;
6. private float esal;
7. private String eaddr;
8.
9. public int getEno() {
10.    return eno;
11. }
12. public void setEno(int eno) {
13.    this.eno = eno;
14. }
15. public String getEname() {
16.    return ename;
17. }
18. public void setEname(String ename) {
19.    this.ename = ename;
20. }
21. public float getEsal() {
22.    return esal;
23. }
24. public void setEsal(float esal) {
25.    this.esal = esal;
26. }
27. public String getEaddr() {
28.    return eaddr;
29. }
30. public void setEaddr(String eaddr) {
31.    this.eaddr = eaddr;
32. }
33.
34.
35.}
```

Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Employee" table="emp1">
7.     <id name="eno"/>
8.     <property name="ename"/>
9.     <property name="esal"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10.   <property name="eaddr"/>
11. </class>
12. <sql-query name="scalarl_sql_query">
13.   <return-scalar column="eno" type="int"/>
14.   <return-scalar column="ename" type="string"/>
15.   <return-scalar column="esal" type="float"/>
16.   <return-scalar column="eaddr" type="string"/>
17. select eno, ename, esal, eaddr from emp1 where esal >= ? and esal <= :max_Sal
18. </sql-query>
19.</hibernate-mapping>
```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.    <property name="show_Sql">true</property>
13.    <mapping resource="Employee.hbm.xml"/>
14.  </session-factory>
15.</hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import java.util.List;
4.
5. import org.hibernate.Query;
6. import org.hibernate.SQLQuery;
7. import org.hibernate.Session;
8. import org.hibernate.SessionFactory;
9. import org.hibernate.boot.registry.StandardServiceRegistry;
10. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
11. import org.hibernate.cfg.Configuration;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
12.  
13. import com.durgasoft.pojo.Employee;  
14.  
15. public class Test {  
16.  
17.     public static void main(String[] args) throws Exception{  
18.         Configuration cfg = new Configuration();  
19.         cfg.configure();  
20.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();  
21.         builder = builder.applySettings(cfg.getProperties());  
22.         StandardServiceRegistry registry = builder.build();  
23.         SessionFactory sessionFactory = cfg.buildSessionFactory(registry);  
24.         Session session = sessionFactory.openSession();  
25.         Query query = session.getNamedQuery("scalarl_sql_query");  
26.         query.setFloat(0, 6000);  
27.         query.setFloat("max_Sal", 8000);  
28.         List<Object[]> list = query.list();  
29.         System.out.println("ENO\tENAME\tESAL\tEADDR");  
30.         System.out.println("-----");  
31.         for(Object[] obj: list) {  
32.             System.out.println(obj[0]+\t+obj[1]+\t+obj[2]+\t+obj[3]);  
33.         }  
34.  
35.  
36.         session.close();  
37.         sessionFactory.close();  
38.     }  
39. }
```

Stored Procedures and Functions in Native SQL:

In Database related applications, first we will define database logic at JAVA applications adn we will transfer that logic to databases inorder to execute, If we have any requirement like to execute a particular database logic frequently then it is suggestible to use Stored Procedures and functions in database related applications.

In the above context, define the frequently executed database logic at the database side in the form of Stored Procedures and functions, not at java side and prepare stored procedure call and function call at java application and send that procedure or function call to Database when we want to perform that respective database action.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

What is the difference between Stored Procedure and Function?

ANS:

- ❖ Stored Procedure is a set of sql queries maintained at Database representing a particular action and it is not having return statement to return value.

Syntax:

```
create or replace PROCEDURE proc_Name[(Param_List)]
AS
---Global Declarations---
BEGIN
---Database Logic-----
END proc_Name;
/ --> To save and Compile Procedure.
```

- ❖ Stored Function is a set of sql queries maintained at Database representing a particular action and it is using return statement to return a value.

Syntax:

```
create or replace FUNCTION fun_Name[(Param_List)] return Data_Type
AS
---Global Declarations---
BEGIN
---Database Logic---
return value;
END fun_Name;
/ --> To Save and Compile Function
```

There are three types of parameters in Stored Procedures and Functions.

1)IN Type Parameter: It will get value from Procedure/function call to procedure body or function body.

EX: no IN number

2)OUT Type Parameter: It will get value from Procedure/function body to procedure/function call.

EX: sal OUT float

3)INOUT Type parameter: It is acting as both IN type parameter and OUT type parameter.

EX: sal INOUT float

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Note: If we want to execute select sql query and if we want to represent records of data then we have to use CURSOR type variable, Oracle database has provided a predefined CURSOR in the form of SYS_REFCURSOR to represent the result of a particular SQL Query.

If we want to use Stored Procedures and Functions in Hibernate applications then we have to use the following steps.

1. Define Stored Procedure or Function at Database side.
2. Configure the respective Stored procedure/Function call in hibernate mapping file.
3. In Client Application, create Query object with the procedure call or Function call logical name.
4. Execute Procedure or Function call.

EX:

Procedure at DB:

```
create or replace PROCEDURE getEmps(emps OUT SYS_REFCURSOR , sal IN float)
AS
BEGIN
open emps for
    select * from emp1 where esal<sal;
END getEmps;
```

Employee.java

```
1. package com.durgasoft.pojo;
2. public class Employee {
3.     private int eno;
4.     private String ename;
5.     private float esal;
6.     private String eaddr;
7.
8.     public int getEno() {
9.         return eno;
10.    }
11.    public void setEno(int eno) {
12.        this.eno = eno;
13.    }
14.    public String getEname() {
15.        return ename;
16.    }
17.    public void setEname(String ename) {
18.        this.ename = ename;
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

19.    }
20.    public float getEsal() {
21.        return esal;
22.    }
23.    public void setEsal(float esal) {
24.        this.esal = esal;
25.    }
26.    public String getEaddr() {
27.        return eaddr;
28.    }
29.    public void setEaddr(String eaddr) {
30.        this.eaddr = eaddr;
31.    }
32.
33.
34.}
```

Employee.hbm.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE hibernate-mapping PUBLIC
3.      "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.      "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5.  <hibernate-mapping>
6.      <class name="com.durgasoft.pojo.Employee" table="emp1">
7.          <id name="eno"/>
8.          <property name="ename"/>
9.          <property name="esal"/>
10.         <property name="eaddr"/>
11.     </class>
12.     <sql-query name="getSal_Proc" callable="true">
13.         <return class="com.durgasoft.pojo.Employee"/>
14.         {call getEmps(?, :sal)}
15.     </sql-query>
16. </hibernate-mapping>
```

hibernate.cfg.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE hibernate-configuration PUBLIC
3.      "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.      "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5.  <hibernate-configuration>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

6. <session-factory>
7.   <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.   <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.   <property name="connection.user">system</property>
10.  <property name="connection.password">durga</property>
11.  <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
    y>
12.  <property name="show_Sql">true</property>
13.  <mapping resource="Employee.hbm.xml"/>
14.  <!-- <mapping class="com.durgasoft.pojo.Employee"/> -->
15. </session-factory>
16.</hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import java.util.List;
4.
5. import org.hibernate.Query;
6. import org.hibernate.SQLQuery;
7. import org.hibernate.Session;
8. import org.hibernate.SessionFactory;
9. import org.hibernate.boot.registry.StandardServiceRegistry;
10. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
11. import org.hibernate.cfg.Configuration;
12.
13. import com.durgasoft.pojo.Employee;
14.
15. public class Test {
16.
17.     public static void main(String[] args) throws Exception{
18.         Configuration cfg = new Configuration();
19.         cfg.configure();
20.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
21.         builder = builder.applySettings(cfg.getProperties());
22.         StandardServiceRegistry registry = builder.build();
23.         SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
24.         Session session = sessionFactory.openSession();
25.         Query query = session.getNamedQuery("getSal_Proc");
26.
27.         query.setFloat("sal", 10000);
28.         List<Employee> list = query.list();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

29.     System.out.println("ENO\tENAME\tESAL\tEADDR");
30.     System.out.println("-----");
31.     for(Employee e: list) {
32.         System.out.println(e.getEno()+"\t"+e.getEname()+"\t"+e.getEsal()+"\t"+e.getEaddr())
33.     ;
34.    }
35.    session.close();
36.    sessionFactory.close();
37.  }
38.

```

Example on Store Functions:

Function at Database:

```

1 SQL> create or replace FUNCTION getEmployees return SYS_REFCURSOR
2 AS
3 employees SYS_REFCURSOR;
4 BEGIN
5 open employees for
6 select * from emp1;
7 return employees;
8 END getEmployees;
9 /

```

Function created.

Employee.java

```

1. package com.durgasoft.pojo;
2. public class Employee {
3.     private int eno;
4.     private String ename;
5.     private float esal;
6.     private String eaddr;
7.
8.     public int getEno() {
9.         return eno;
10.    }
11.    public void setEno(int eno) {
12.        this.eno = eno;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

13. }
14. public String getName() {
15.     return ename;
16. }
17. public void setName(String ename) {
18.     this.ename = ename;
19. }
20. public float getEsal() {
21.     return esal;
22. }
23. public void setEsal(float esal) {
24.     this.esal = esal;
25. }
26. public String getEaddr() {
27.     return eaddr;
28. }
29. public void setEaddr(String eaddr) {
30.     this.eaddr = eaddr;
31. }
32.
33.
34.}
```

Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Employee" table="emp1">
7.     <id name="eno"/>
8.     <property name="ename"/>
9.     <property name="esal"/>
10.    <property name="eaddr"/>
11.  </class>
12.  <sql-query name="getEmployees_Fun" callable="true">
13.    <return class="com.durgasoft.pojo.Employee"/>
14.    {? = call getEmployees}
15.  </sql-query>
16. </hibernate-mapping>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.    <property name="show_Sql">true</property>
13.    <mapping resource="Employee.hbm.xml"/>
14.    <!-- <mapping class="com.durgasoft.pojo.Employee"/> -->
15.  </session-factory>
16. </hibernate-configuration>

```

Test.java

```

1. package com.durgasoft.test;
2. import java.util.List;
3. import org.hibernate.Query;
4. import org.hibernate.SQLQuery;
5. import org.hibernate.Session;
6. import org.hibernate.SessionFactory;
7. import org.hibernate.boot.registry.StandardServiceRegistry;
8. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
9. import org.hibernate.cfg.Configuration;
10.
11. import com.durgasoft.pojo.Employee;
12.
13. public class Test {
14.
15.   public static void main(String[] args) throws Exception{
16.     Configuration cfg = new Configuration();
17.     cfg.configure();
18.     StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
19.     builder = builder.applySettings(cfg.getProperties());
20.     StandardServiceRegistry registry = builder.build();
21.     SessionFactory sessionFactory = cfg.buildSessionFactory(registry);

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
22. Session session = sessionFactory.openSession();
23. Query query = session.getNamedQuery("getEmployees_Fun");
24.
25. List<Employee> list = query.list();
26. System.out.println("ENO\tENAME\tESAL\tEADDR");
27. System.out.println("-----");
28. for(Employee e: list) {
29.     System.out.println(e.getEno()+"\t"+e.getEname()+"\t"+e.getEsal()+"\t"+e.getEaddr())
30. ;
31. }
32.
33. session.close();
34. sessionFactory.close();
35. }
36. }
```

3. Criterion API:

- ★ By using Session interface methods like save(), persist(), update(), delete(),..... we are able to perform single record manipulation, but, if we want to perform manipulations over multiple records then we must go for HQL, Native SQL and Criterion API.
- ★ Where HQL is a powerfull, Object Oriented and Database INdependent Query language provided by Hibernate, but, HQL is not providing environment to perform DDL kind of operations and it is not supporting database dependent native operations like invoking stored procedures and functions,.....
- ★ To overcome the above problem with HQL we will use "Native SQL", in case of Native SQL , we have to write database dependent SQL queries directly, but, it is against to Hibernate, as per the Hibernate view we must not write database dependent sql queries in JAVA applications.
- ★ In Hibernate applications, to avoid totally query langugaes like SQL and HQL,...and to provide the complete dPersistence logic in the form of JAVA code we must use "Criterion API".
- ★ In the case of Criterion API, we will define persistence logic by using JAVA code only, where the required predefined library was provided by Hibernate in the form of "org.hibernate" package and "org.hibernate.criterion" package.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

If we want to use Criterion API in Hibernate applications then we have to use the following steps.

1. Create Criteria Object:

Criteria object is a central object in Criteria API, it able to manage HQL query representation internally and it has provided predefined methods to defined query logic.

To create Criteria object we have to use the following method from Session.

```
public Criteria createCriteria(Class cls);
```

EX: Criteria c = session.createCriteria(com.durgasoft.pojo.Employee.class);

Note: It is equivalent to the HQL query internally "from Employee".

2. Prepare Criterion objects and add that Criterion objects to Criteria object:

Criterion is an object , it able to manage a single Conditional expression in database logic.

To create Criterion object we have to use the following methods from "org.hibernate.Restrictions" class.

```
public static Criterion isEmpty(String property)
public static Criterion isNotEmpty(String property)
public static Criterion isNull(String property)
public static Criterion isNotNull(String property)
public static Criterion in(String property, Object[] obj)
public static Criterion in(String property, Collection c)
public static Criterion between(String property, Object min_Val, Object max_Val)
public static Criterion between(String property, Object[] obj)
public static Criterion eq(String property, Object val)
public static Criterion ne(String property, Object val)
public static Criterion lt(String property, Object val)
public static Criterion le(String property, Object val)
public static Criterion gt(String property, Object val )
public static Criterion ge(String property, Object val)

-----
```

To add a particular Criterion object to Criteria object we have to use the following method.

```
public void add(Criterion c)
```

EX:

```
Criterion c1 = Restrictions.ge("esal", 60000);
Criterion c2 = Restrictions.le("esal", 90000);
c.add(c1);
c.add(c2);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Note: With the above steps, Criteria object is able to prepare the query like "from Employee esal>=6000 and esal<=9000".

3. Create Projection objects , add Projection objects to ProjectionList and add ProjectionList to Criteria object:

The main intention of Projection object is to represent a single POJO class property.

To get Projection object with a particular Property name we have to use the following method from "Projections" class.

```
public static Projection projection(String pro_Name)
```

To create ProjectionList object we have to use the following method from Projections class.

```
public static ProjectionList projectionList()
```

To add Projection object to ProjectionList we have to use the following method from ProjectionList class.

```
public void add(Projection p)
```

To set ProjectionList to Criteria object we have to use the following method.

```
public void setProjection(ProjectionList pl)
```

Ex:

```
ProjectionList pl = Projections.projectionList();
pl.add(Projections.property("eno"));
pl.add(Projections.property("ename"));
pl.add(Projections.property("esal"));
pl.add(Projections.property("eaddr"));
c.setProjection(pl);
```

EX: With the above , Criteria object is able to prepare HQL query internally like below.
 "select eno, ename, esal, eaddr from Employee where esal >=6000 and esal<=90000".

4) Provide a particular Order to the query:

To represent a particular Order over the results, we have to use either asc(-) or desc(-) methods from "Order" class.

```
public static Order asc(String prop_Name)
public static Order desc(String prop_Name)
```

To add Order object to Criteria object we have to use the following method.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



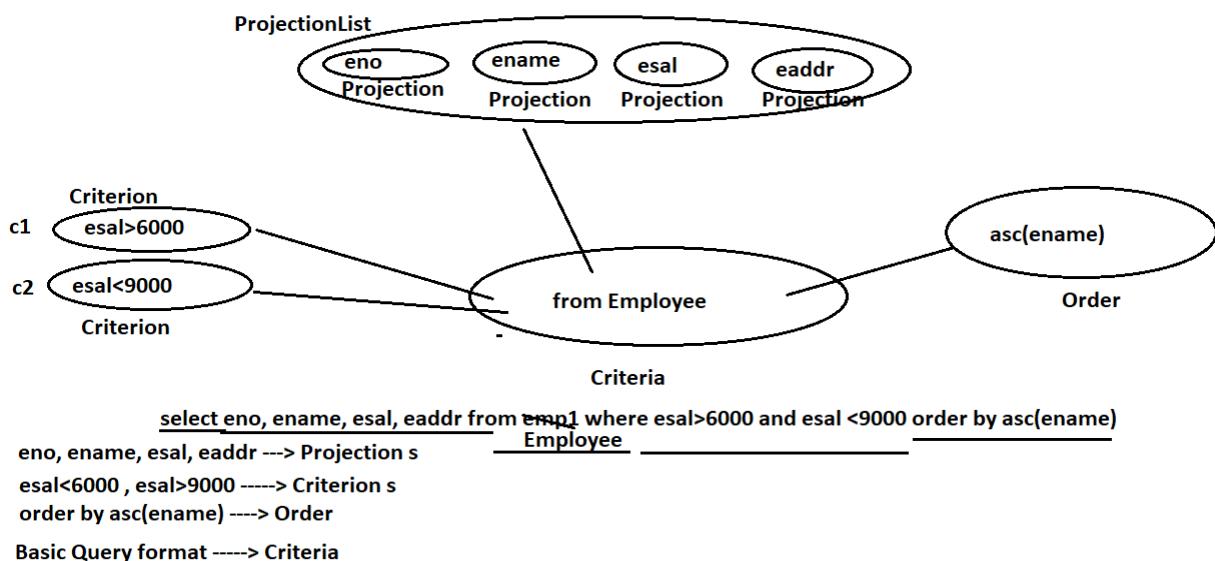
BY NAGOOR BABU

```
public void addOrder(Order o)
```

EX:

```
Order o = Order.desc("ename");
c.addOrder(o);
```

Note: With this, Criteria object is able to create HQI query like
 "select eno, ename, esal, eaddr from Employee where esal >=6000 and esal<=90000 order by desc(ename)"



5) Execute Database logic which we provided in Criteria object:

To execute database logic existed in Criteria object we have to use the following methods.

```
public List list()
public Iterator iterate()
public ScrollableResults scroll()
public Object uniqueResult()
```

EX:

```
List<Object[]> list = c.list();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Note: Like HQL, we are able to provide custom properties over the Criteria object by using the following methods.

```
public void setFetchSize(int size)
public void setFirstResult(int position)
public void setMaxResults(int val)
public void readOnly(boolean b)
public void cacheable(boolean b)
----
```

Example:

Employee.java

```
1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp1")
10. public class Employee {
11.     @Id
12.     @Column(name="ENO")
13.     private int eno;
14.     @Column(name="ENAME")
15.     private String ename;
16.     @Column(name="ESAL")
17.     private int esal;
18.     @Column(name="EADDR")
19.     private String eaddr;
20.
21.     public int getEno() {
22.         return eno;
23.     }
24.     public void setEno(int eno) {
25.         this.eno = eno;
26.     }
27.     public String getEname() {
28.         return ename;
29.     }
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

30. public void setEname(String ename) {
31.     this.ename = ename;
32. }
33. public int getEsal() {
34.     return esal;
35. }
36. public void setEsal(int esal) {
37.     this.esal = esal;
38. }
39. public String getEaddr() {
40.     return eaddr;
41. }
42. public void setEaddr(String eaddr) {
43.     this.eaddr = eaddr;
44. }
45.
46.
47.}
```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.    <property name="show_Sql">true</property>
13.    <mapping class="com.durgasoft.pojo.Employee"/>
14.  </session-factory>
15. </hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import java.util.List;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
4.  
5. import org.hibernate.Criteria;  
6. import org.hibernate.Session;  
7. import org.hibernate.SessionFactory;  
8. import org.hibernate.boot.registry.StandardServiceRegistry;  
9. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;  
10. import org.hibernate.cfg.Configuration;  
11. import org.hibernate.criterion.Criterion;  
12. import org.hibernate.criterion.Order;  
13. import org.hibernate.criterion.ProjectionList;  
14. import org.hibernate.criterion.Projections;  
15. import org.hibernate.criterion.Restrictions;  
16.  
17. import com.durgasoft.pojo.Employee;  
18.  
19. public class Test {  
20.  
21.     public static void main(String[] args) throws Exception{  
22.         Configuration cfg = new Configuration();  
23.         cfg.configure();  
24.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();  
25.         builder = builder.applySettings(cfg.getProperties());  
26.         StandardServiceRegistry registry = builder.build();  
27.         SessionFactory sessionFactory = cfg.buildSessionFactory(registry);  
28.         Session session = sessionFactory.openSession();  
29.  
30.         Criteria c = session.createCriteria(Employee.class);  
31.  
32.         Criterion c1 = Restrictions.ge("esal",6000);  
33.         Criterion c2 = Restrictions.le("esal", 8000);  
34.         c.add(c1);  
35.         c.add(c2);  
36.  
37.         ProjectionList pl = Projections.projectionList();  
38.         pl.add(Projections.property("eno"));  
39.         pl.add(Projections.property("ename"));  
40.         pl.add(Projections.property("esal"));  
41.         pl.add(Projections.property("eaddr"));  
42.         c.setProjection(pl);  
43.  
44.         Order order = Order.desc("ename");  
45.         c.addOrder(order);  
46.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
47.     List<Object[]> list = c.list();
48.
49.     System.out.println("ENO\tENAME\tESAL\tEADDR");
50.     System.out.println("-----");
51.     for(Object[] obj: list) {
52.         for(Object o: obj) {
53.             System.out.print(o+"\t");
54.         }
55.         System.out.println();
56.     }
57.
58.     session.close();
59.     sessionFactory.close();
60. }
61. }
```

DURGASU

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Hibernate Filters

In Hibernate Applications, by using HQL queries, we are able to retrieve multiple records of data from database table. While retrieving multiple Records from Database table if we want to filter the results on the basis of a particular condition then we have to use "Hibernate Filters".

If we want to use Filters in Hibernate applications we have to use the following steps.

1) Prepare a table in Database with the required columns, where we must define a filter parameter column:

```
sql>create table emp(ENO number primary key, ENAME varchar2(10), ESAL float, EADDR
varchar2(10), ETYPE varchar2(10));
```

Here "ETYPE" is Filter parameter column, by using this Filter parameter column only we are able to prepare Filter condition inorder to filter the results.

2) Define Filter in mapping File:

To define Filter in mapping file we have to use the following xml tags in mapping file.

```
1. <hibernate-mapping>
2. -----
3. <class name="--" table="--">
4. -----
5.   <filter name="--" condition="--"/>
6. </class>
7. <filter-def name="--">
8.   <filter-param name="--" type="--"/>
9. </filter-def>
10. -----
11. </hibernate-mapping>
```

- ✓ Where <filter-def> tag can be used to define a filter in Hibernate mapping file.
- ✓ Where "name" attribute in <filter-def> tag is able to take logical name of the filter.
- ✓ Where <filter-param> tag is a child tag to <filter-def> tag , it will define a particular filter parameter.
- ✓ Where "name" attribute in <filter-param> tag will take name of the filter parameter.
- ✓ Where "type" attribute in <filter-param> tag will take java type like "string", "byte", "int",....
- ✓ Where "<filter>" tag under <class> tag is able to configure Filter to mapping.
- ✓ Where "name" attribute in <filter> tag is able to take logical of the filter which we have defined in <filter-def> tag.
- ✓ Where "condition" attribute in <filter> tag is able to define condition inorder to filter the results.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3) Enable Filter in Session and set values to filter parameters inorder to filter the results:

To enable Filter in Session we have to use the following method from org.hibernate.Session.

```
public Filter enableFilter(String logical_name)
```

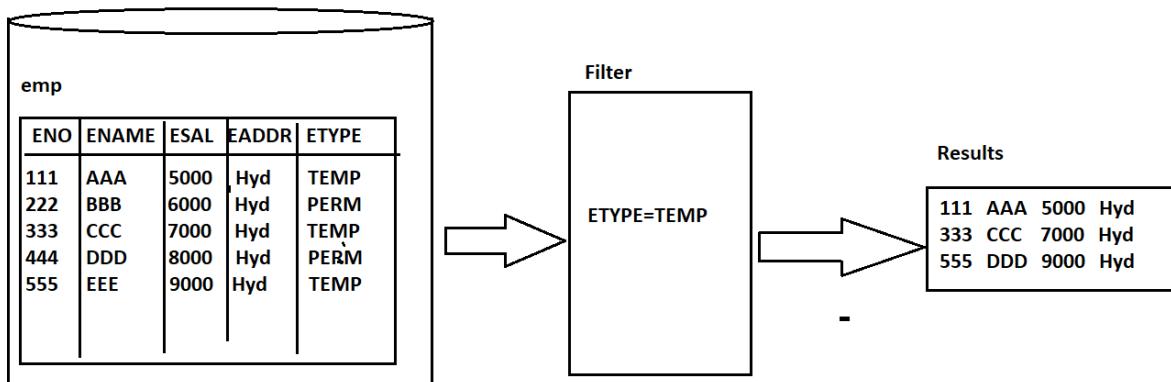
To set values to the Filter parameter we will use the following method.

```
public void setParameter(String filter_Name, xxx value)
```

Where xxx may be byte, short, int,.....

To disable Filter , we will use the following method from org.hibernate.Session.

```
public void disableFilter(String logical_Name)
```



Example:

Employee.java

```
1. package com.durgasoft.pojo;
2. public class Employee {
3.     private int eno;
4.     private String ename;
5.     private float esal;
6.     private String eaddr;
7.     private String etype;
8.
9.     public String getEtype() {
10.        return etype;
11.    }
12. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

11.    }
12.    public void setEtype(String etype) {
13.        this.etype = etype;
14.    }
15.    public int getEno() {
16.        return eno;
17.    }
18.    public void setEno(int eno) {
19.        this.eno = eno;
20.    }
21.    public String getEname() {
22.        return ename;
23.    }
24.    public void setEname(String ename) {
25.        this.ename = ename;
26.    }
27.    public float getEsal() {
28.        return esal;
29.    }
30.    public void setEsal(float esal) {
31.        this.esal = esal;
32.    }
33.    public String getEaddr() {
34.        return eaddr;
35.    }
36.    public void setEaddr(String eaddr) {
37.        this.eaddr = eaddr;
38.    }
39.
40.
41.}
```

Employee.hbm.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE hibernate-mapping PUBLIC
3.      "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.      "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5.  <hibernate-mapping>
6.      <class name="com.durgasoft.pojo.Employee" table="emp2">
7.          <id name="eno"/>
8.          <property name="ename"/>
9.          <property name="esal"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10.    <property name="eaddr"/>
11.    <property name="etype"/>
12.    <filter name="empFilter" condition=":type = ETYPE"/>
13.  </class>
14.  <filter-def name="empFilter">
15.    <filter-param name="type" type="string"/>
16.  </filter-def>
17. </hibernate-mapping>
```

hibernate.cfg.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE hibernate-configuration PUBLIC
3.    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5.  <hibernate-configuration>
6.    <session-factory>
7.      <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.      <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.      <property name="connection.user">system</property>
10.     <property name="connection.password">durga</property>
11.     <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.     <property name="show_Sql">true</property>
13.     <mapping resource="Employee.hbm.xml"/>
14.   </session-factory>
15. </hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import java.util.List;
4. import java.util.Scanner;
5.
6. import org.hibernate.Filter;
7. import org.hibernate.Query;
8. import org.hibernate.Session;
9. import org.hibernate.SessionFactory;
10. import org.hibernate.boot.registry.StandardServiceRegistry;
11. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
12. import org.hibernate.cfg.Configuration;
13.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
14. import com.durgasoft.pojo.Employee;
15.
16. public class Test {
17.
18.     public static void main(String[] args){
19.         try {
20.             Configuration cfg = new Configuration();
21.             cfg.configure();
22.             StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
23.             builder = builder.applySettings(cfg.getProperties());
24.             StandardServiceRegistry registry = builder.build();
25.             SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
26.             Session session = sessionFactory.openSession();
27.
28.             Query query = session.createQuery("from Employee");
29.
30.             Filter filter = session.enableFilter("empFilter");
31.             Scanner s = new Scanner(System.in);
32.             System.out.println("Employee Types:");
33.             System.out.println("1.PERM");
34.             System.out.println("2.TEMP");
35.             System.out.print("Your Option :");
36.             String etype = s.next();
37.             filter.setParameter("type", etype);
38.             List<Employee> list = query.list();
39.
40.             System.out.println("ENO\tENAME\tESAL\tEADDR\tETYPE");
41.             System.out.println("-----");
42.             for(Employee emp: list) {
43.                 System.out.println(emp.getEno()+"\t"+emp.getEname()+"\t"+emp.getEsal()+"\t"+em
        p.getEaddr()+"\t"+emp.getType());
44.             }
45.             session.close();
46.             sessionFactory.close();
47.             }catch(Exception e) {
48.                 e.printStackTrace();
49.             }
50.     }
51. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Hibernate Mappings

- ★ In Enterprise Applications, we are able to use the data models like Object Oriented Data Model, Relational Data Model,...In general, Front-End applications are able to use Object Oriented Data Model to represent data and Back-End Systems are able to use Relational Data Model to represent data.
- ★ In the above context, both Object Oriented Data Model and relational Data model are having their own conventions to represent data, these different conventions will create mismatches between data models, it will reduce data persistency in enterprise applications.
- ★ In the above context, to improve data persistency in enterprise applications we have to solve the mismatches between data models, for this we have to use ORM implementations.
- ★ Hibernate is one of the ORM implementation , it has provided "Hibernate Mappings" feature to resolve mismatches between data models. To resolve mismatches between data models, Hibernate has provided the following mappings.

Different Types of Mappings:

1. Basic OR Mapping.
2. Component Mapping
3. Inheritance Mapping
4. Association Mapping

1. Basic OR Mapping:

It is normal mapping between a class, an ID property, a normal property from Object Oriented data model and a table, a primary key column and normal columns.

2. Component Mapping:

IN Hibernate applications, Component Mapping will provide solution for Granularity Mismatch.

In Component mapping, we will define a component property inorder to refer multiple columns in a Database table.

To configure component property in hibernate applications we will use the following XML tags.

1. **<hibernate-mapping>**
2. **<class name="--" table="-->**
3. -----
4. **<component name="--" class="-->**

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

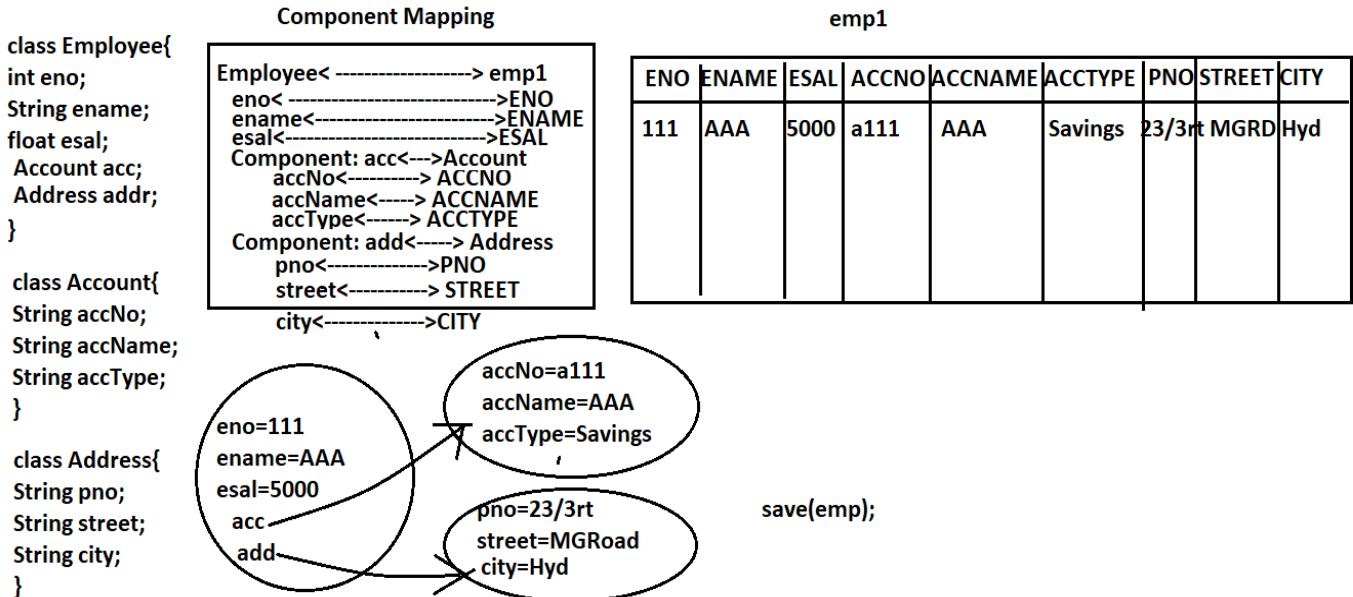


BY NAGOOR BABU

```

5. -----
6. </component>
7. -----
8. </class>
9. <hibernate-mapping>
```

- Where "<component>" tag will take Component class and its reference variable declaration, where "name" attribute will take component property reference variable and "class" attribute will take fully qualified name of the respective component class.



Example:

Account.java

```

1. package com.durgasoft.pojo;
2.
3. public class Account {
4.     private String accNo;
5.     private String accName;
6.     private String accType;
7.
8.     public String getAccNo() {
9.         return accNo;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
10. }
11. public void setAccNo(String accNo) {
12.     this.accNo = accNo;
13. }
14. public String getAccName() {
15.     return accName;
16. }
17. public void setAccName(String accName) {
18.     this.accName = accName;
19. }
20. public String getAccType() {
21.     return accType;
22. }
23. public void setAccType(String accType) {
24.     this.accType = accType;
25. }
26.
27.
28.}
```

Address.java

```
1. package com.durgasoft.pojo;
2.
3. public class Address {
4.     private String pno;
5.     private String street;
6.     private String city;
7.     public String getPno() {
8.         return pno;
9.     }
10.    public void setPno(String pno) {
11.        this.pno = pno;
12.    }
13.    public String getStreet() {
14.        return street;
15.    }
16.    public void setStreet(String street) {
17.        this.street = street;
18.    }
19.    public String getCity() {
20.        return city;
21.    }
22. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

22. public void setCity(String city) {
23.     this.city = city;
24. }
25.
26.}
```

Employee.java



```

1. package com.durgasoft.pojo;
2.
3. public class Employee {
4.     private int eno;
5.     private String ename;
6.     private float esal;
7.     private Account eacc;
8.     private Address eaddr;
9.
10.    public int getEno() {
11.        return eno;
12.    }
13.    public void setEno(int eno) {
14.        this.eno = eno;
15.    }
16.    public String getEname() {
17.        return ename;
18.    }
19.    public void setEname(String ename) {
20.        this.ename = ename;
21.    }
22.    public float getEsal() {
23.        return esal;
24.    }
25.    public void setEsal(float esal) {
26.        this.esal = esal;
27.    }
28.    public Account getEacc() {
29.        return eacc;
30.    }
31.    public void setEacc(Account eacc) {
32.        this.eacc = eacc;
33.    }
34.    public Address getEaddr() {
35.        return eaddr;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

36.    }
37.    public void setEaddr(Address eaddr) {
38.        this.eaddr = eaddr;
39.    }
40.
41.

```

Employee.hbm.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE hibernate-mapping PUBLIC
3.      "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.      "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5.  <hibernate-mapping>
6.      <class name="com.durgasoft.pojo.Employee" table="emp3">
7.          <id name="eno"/>
8.          <property name="ename"/>
9.          <property name="esal"/>
10.         <component name="eacc" class="com.durgasoft.pojo.Account">
11.             <property name="accNo"/>
12.             <property name="accName"/>
13.             <property name="accType"/>
14.         </component>
15.         <component name="eaddr" class="com.durgasoft.pojo.Address">
16.             <property name="pno"/>
17.             <property name="street"/>
18.             <property name="city"/>
19.         </component>
20.     </class>
21. </hibernate-mapping>

```

hibernate.cfg.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE hibernate-configuration PUBLIC
3.      "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.      "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5.  <hibernate-configuration>
6.      <session-factory>
7.          <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.          <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.          <property name="connection.user">system</property>
10.         <property name="connection.password">durga</property>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

11.   <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</proper
y>
12.   <property name="show_Sql">true</property>
13.   <mapping resource="Employee.hbm.xml"/>
14. </session-factory>
15.</hibernate-configuration>
```

**Test.java**

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10.import com.durgasoft.pojo.Account;
11.import com.durgasoft.pojo.Address;
12.import com.durgasoft.pojo.Employee;
13.
14.public class Test {
15.
16. public static void main(String[] args){
17. SessionFactory sessionFactory = null;
18. Session session = null;
19. try {
20. Configuration cfg = new Configuration();
21. cfg.configure();
22. StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
23. builder = builder.applySettings(cfg.getProperties());
24. StandardServiceRegistry registry = builder.build();
25. sessionFactory = cfg.buildSessionFactory(registry);
26. session = sessionFactory.openSession();
27.
28. Account acc = new Account();
29. acc.setAccNo("a111");
30. acc.setAccName("AAA");
31. acc.setAccType("Savings");
32.
33. Address addr = new Address();
34. addr.setPno("23/3t");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



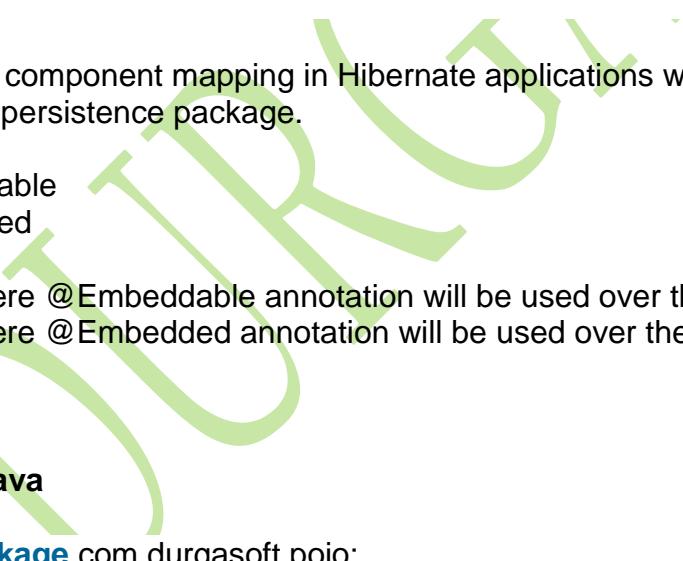
BY NAGOOR BABU

```

35.     addr.setStreet("MGRoad");
36.     addr.setCity("Hyd");
37.
38.     Employee emp = new Employee();
39.     emp.setEno(111);
40.     emp.setEname("AAA");
41.     emp.setEsal(5000);
42.     emp.setEacc(acc);
43.     emp.setEaddr(addr);
44.
45.     Transaction tx = session.beginTransaction();
46.     session.save(emp);
47.     tx.commit();
48.     System.out.println("Employee Inserted Successfully");
49. }catch(Exception e) {
50.     e.printStackTrace();
51. }finally {
52.     session.close();
53.     sessionFactory.close();
54.
55. }
56. }
57.

```

To provide component mapping in Hibernate applications we will use the following annotations from javax.persistence package.


`@Embeddable`
`@Embedded`

- ✓ Where `@Embeddable` annotation will be used over the Component class.
- ✓ Where `@Embedded` annotation will be used over the component property.

Example:

Account.java

```

1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Embeddable;
5. import javax.persistence.Entity;
6.

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

7. @Entity
8. @Embeddable
9. public class Account {
10.   @Column(name="ACCNO")
11.   private String accNo;
12.   @Column(name="ACCNAME")
13.   private String accName;
14.   @Column(name="ACCTYPE")
15.   private String accType;
16.
17.   public String getAccNo() {
18.     return accNo;
19.   }
20.   public void setAccNo(String accNo) {
21.     this.accNo = accNo;
22.   }
23.   public String getAccName() {
24.     return accName;
25.   }
26.   public void setAccName(String accName) {
27.     this.accName = accName;
28.   }
29.   public String getAccType() {
30.     return accType;
31.   }
32.   public void setAccType(String accType) {
33.     this.accType = accType;
34.   }
35.
36.
37.}
```

Address.java

```

1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Embeddable;
5. import javax.persistence.Entity;
6.
7. @Entity
8. @Embeddable
9. public class Address {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10. @Column(name="PNO")
11. private String pno;
12. @Column(name="STREET")
13. private String street;
14. @Column(name="CITY")
15. private String city;
16. public String getPno() {
17.     return pno;
18. }
19. public void setPno(String pno) {
20.     this.pno = pno;
21. }
22. public String getStreet() {
23.     return street;
24. }
25. public void setStreet(String street) {
26.     this.street = street;
27. }
28. public String getCity() {
29.     return city;
30. }
31. public void setCity(String city) {
32.     this.city = city;
33. }
34.
35.
36.
}

```

Employee.java

```

1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Embedded;
5. import javax.persistence.Entity;
6. import javax.persistence.Id;
7. import javax.persistence.Table;
8.
9. @Entity
10. @Table(name="emp3")
11. public class Employee {
12.     @Id
13.     @Column(name="ENO")

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
14. private int eno;
15. @Column(name="ENAME")
16. private String ename;
17. @Column(name="ESAL")
18. private float esal;
19. @Embedded
20. private Account eacc;
21. @Embedded
22. private Address eaddr;
23.
24. public int getEno() {
25.     return eno;
26. }
27. public void setEno(int eno) {
28.     this.eno = eno;
29. }
30. public String getEname() {
31.     return ename;
32. }
33. public void setEname(String ename) {
34.     this.ename = ename;
35. }
36. public float getEsal() {
37.     return esal;
38. }
39. public void setEsal(float esal) {
40.     this.esal = esal;
41. }
42. public Account getEacc() {
43.     return eacc;
44. }
45. public void setEacc(Account eacc) {
46.     this.eacc = eacc;
47. }
48. public Address getEaddr() {
49.     return eaddr;
50. }
51. public void setEaddr(Address eaddr) {
52.     this.eaddr = eaddr;
53. }
54.
55.
56. }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.  <property name="show_Sql">true</property>
13.  <mapping class="com.durgasoft.pojo.Employee"/>
14. </session-factory>
15. </hibernate-configuration>

```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10. import com.durgasoft.pojo.Account;
11. import com.durgasoft.pojo.Address;
12. import com.durgasoft.pojo.Employee;
13.
14. public class Test {
15.
16.     public static void main(String[] args){
17.         SessionFactory sessionFactory = null;
18.         Session session = null;
19.         try {
20.             Configuration cfg = new Configuration();
21.             cfg.configure();
22.             StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

23.     builder = builder.applySettings(cfg.getProperties());
24.     StandardServiceRegistry registry = builder.build();
25.     sessionFactory = cfg.buildSessionFactory(registry);
26.     session = sessionFactory.openSession();
27.
28.     Employee emp = (Employee)session.get(com.durgasoft.pojo.Employee.class, 111)
29.     ;
30.     System.out.println("Employee Details");
31.     System.out.println("-----");
32.     System.out.println("Employee Number :" + emp.getEno());
33.     System.out.println("Employee Name : " + emp.getEname());
34.     System.out.println("Employee Salary :" + emp.getEsal());
35.     System.out.println();
36.     Account acc = emp.getEacc();
37.     System.out.println("Account Details");
38.     System.out.println("-----");
39.     System.out.println("Account Number :" + acc.getAccNo());
40.     System.out.println("Account Name : " + acc.getAccName());
41.     System.out.println("Account Type : " + acc.getAccType());
42.     System.out.println();
43.     Address addr = emp.getEaddr();
44.     System.out.println("Address Details");
45.     System.out.println("-----");
46.     System.out.println("PNO : " + addr.getPno());
47.     System.out.println("Street : " + addr.getStreet());
48.     System.out.println("City : " + addr.getCity());
49. }catch(Exception e) {
50.     e.printStackTrace();
51. }finally {
52.     session.close();
53.     sessionFactory.close();
54. }
55. }
56.

```

3. Inheritance Mapping:

★ In Enterprise Applications, we are able to use the data models like Object Oriented Data Model, Relational Data Model,...In general, Front-End applications are able to use Object Oriented Data Model to represent data and Back-End Systems are able to use Relational Data Model to represent data.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ★ In Object Oriented Data Model, we will provide inheritance relation between entities inorder to improve code Reusability.
- ★ In Relational data Model, we will use different approaches to manage inheritance kind of features at databases like maintaining all the classes properties in a single table or defining a seperate table for each and every class and providing PK-FK relation between these tables,.....
- ★ In both the above cases, approaches are different to achieve inheritance kind of feature, it will provide Sub types mismatches between data models, it will reduce data persistency in enterprise applications.
- ★ In the above context, to improve data persistency in enterprise applications we have to use ORM implementation, Hibernate is one of the ORM implementation, it has provided the following three strategies to resolve inheritance mismatch.
 1. Table Per Class Hierarchy.
 2. Table Per Sub Class
 3. Table Per Concreate Class

1) Table Per Class Hierarchy.

- ★ In Table Per Class hierarchy, we will prepare a table with the columns representing all the properties of the classes[sub classes and super class] which are existed in inheritance.
- ★ In this mechanism, when we store any sub class object then data must be stored in single table in the respective columns, if data is not available for any column then null values will be stored in the respective columns.

If we want to use Table Per Class Hierarchy in Hibernate applications then we have to use the following steps.

1) Prepare all java classes which we want to keep in Inheritance

EX:**Account.java**

```
1. public class Account {  
2.     private String accNo;  
3.     private String accName;  
4.     private String accType;  
5.     setXXX() and getXXX()  
6. }
```

CONTACT US:**Mobile: +91- 8885 25 26 27** **+91- 7207 21 24 27/28****US NUM: 4433326786****Mail ID: durgasoftonlinetraining@gmail.com****WEBSITE: www.durgasoftonline.com****FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,**



BY NAGOOR BABU

StudentAccount.java

```

1. public class StudentAccount extends Account{
2.     private String sid;
3.     private String sbranch;
4.     private int smarks;
5.     setXXX() and getXXX()
6. }
```

**EmployeeAccount.java**

```

1. public class EmployeeAccount extends Account{
2.     private String eid;
3.     private float esal;
4.     private String eaddr;
5.     setXXX() and getXXX()
6. }
```



2) Create a table in Database with a set of columns which are representinig all properties of the classes which are existed in inheritance and discriminator column:

SQL> create table account(ACCNO varchar2(5) primary key, ACCNAME varchar2(5), accType varchar2(10), SID varchar2(5), SBRANCH varchar2(5), SMARKS number(3), EID varchar2(5), ESAL float, EADDR varchar2(10), TYPE varchar2(10));

Where the purpose Discriminator column is to specify which sub class object data is represented by the present record.

3) Provide all subclasses configuration and discriminator column configuration in hibernate mapping file by using the following tags:

```

1. <hibernate-mapping>
2. <class name="--" class="--">
3.   ---
4.   <discriminatator column="--"/>
5.   ---
6.   <subclass name="--" discriminator-value="--">
7.     -----
8.   </subclass>
9.   ---
10.  </class>
11. </hibernate-mapping>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ✓ Where "<discriminator>" tag can be used to configure discriminator column, where "column" attribute in <discriminator> tag will take the discriminator column name which we defined in database table.
- ✓ Where "<subclass>" tag is able to configure a single sub class and its properties, where "name" attribute will take fully qualified name of the respective sub class, where "discriminator-value" attribute will provide the exact discriminator value inorder to insert or retrive sub class object data.

EX:

```

1. <hibernate-mapping>
2.   <class name="com.durgasoft.pojo.Account" table="account">
3.     <id name="accNo" column="ACCNO"/>
4.     <discriminator column="TYPE" type="string"/>
5.     <property name="accName" column="ACCNAME"/>
6.     <property name="accType" column="ACCTYPE"/>
7.
8.     <subclass name="com.durgasoft.pojo.StudentAccount" discriminator-value="std">
9.       <property name="sid" column="SID"/>
10.      <property name="sbranch" column="SBRANCH"/>
11.      <property name="smarks" column="SMARKS"/>
12.    </subclass>
13.
14.    <subclass name="com.durgasoft.pojo.EmployeeAccount" discriminator-
15.      value="emp">
16.        <property name="eid" column="EID"/>
17.        <property name="esal" column="ESAL"/>
18.        <property name="eaddr" column="EADDR"/>
19.      </subclass>
20.    </class>
</hibernate-mapping>
```

4) In client application, Create Sub class objects with the data and perform the required database operation:

```

1. SessionFactory factory = cfg.buildSessionFactory(registry);
2. Session session = factory.openSession();
3.
4. StudentAccount sa = new StudentAccount();
5. sa.setAccNo("a1");
6. sa.setAccName("AAA");
7. sa.setAccType("Savings");
8. sa.setSid("S1");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

9. sa.setSbranch("CS");
10.sa.setSmarks(77);
11.
12.EmployeeAccount ea = new EmployeeAccount();
13.ea.setAccNo("a2");
14.ea.setAccName("BBB");
15.ea.setAccType("Savings");
16.ea.setEid("E1");
17.ea.setEsal(5000);
18.ea.setEaddr("Hyd");
19.
20.Transaction tx = session.beginTransaction();
21.session.save(sa);
22.session.save(ea);
23.tx.commit();
24.System.out.println("Student Account Inserted Successfully");
25.System.out.println("Employee Account Inserted Successfully");

```

```

class Account{
String accNo;
String accName;
String accType;
---}
class StudentAccount{
extends Account
String sid;
String sbranch;
int smarks;
---}
class EmployeeAccount{
extends Account
String eid;
float esal;
String eaddr;
---}

```

Inheritance Mapping [Table Per Class Hierarchy]

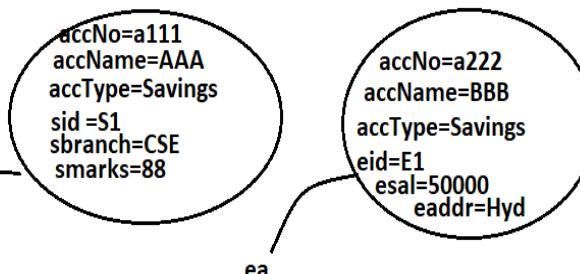
```

class: Account <----> account
accNo<----->ACCNO
accName<----->ACCNAME
accType<----->ACCTYPE
discriminator: TYPE
sub-class: StudentAccount
discriminator-value: std
sid<----->SID
sbranch<----->SBRANCH
smarks<----->SBRANCH
sub-class: EmployeeAccount
discriminator-value: emp
eid<----->EID
esal<----->ESAL
eaddr<----->EADDR

```

s.save(sa);
s.save(ea);

account									Discriminator Column
ACCNO	ACCNAME	ACCTYPE	SID	SBRANCH	SMARKS	EID	ESAL	EADDR	TYPE
a111	AAA	Savings	S1	CSE	88	null	null	null	std
a222	BBB	Savings	null	null	null	E1	5000	Hyd	emp



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EXAMPLE:**Account.java**

```
1. package com.durgasoft.pojo;
2.
3. public class Account {
4.     private String accNo;
5.     private String accName;
6.     private String accType;
7.
8.     public String getAccNo() {
9.         return accNo;
10.    }
11.    public void setAccNo(String accNo) {
12.        this.accNo = accNo;
13.    }
14.    public String getAccName() {
15.        return accName;
16.    }
17.    public void setAccName(String accName) {
18.        this.accName = accName;
19.    }
20.    public String getAccType() {
21.        return accType;
22.    }
23.    public void setAccType(String accType) {
24.        this.accType = accType;
25.    }
26.
27.
28.}
```

StudentAccount.java

```
1. package com.durgasoft.pojo;
2.
3. public class StudentAccount extends Account{
4.     private String sid;
5.     private String sbranch;
6.     private int smarks;
7.
8.     public String getSid() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
9.     return sid;
10.    }
11.    public void setSid(String sid) {
12.        this.sid = sid;
13.    }
14.    public String getSbranch() {
15.        return sbranch;
16.    }
17.    public void setSbranch(String sbranch) {
18.        this.sbranch = sbranch;
19.    }
20.    public int getSmarks() {
21.        return smarks;
22.    }
23.    public void setSmarks(int smarks) {
24.        this.smarks = smarks;
25.    }
26.
27.
28.}
```

EmployeeAccount.java



```
1. package com.durgasoft.pojo;
2.
3. public class EmployeeAccount extends Account{
4.     private String eid;
5.     private float esal;
6.     private String eaddr;
7.
8.     public String getEid() {
9.         return eid;
10.    }
11.    public void setEid(String eid) {
12.        this.eid = eid;
13.    }
14.    public float getEsal() {
15.        return esal;
16.    }
17.    public void setEsal(float esal) {
18.        this.esal = esal;
19.    }
20.    public String getEaddr() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

21.     return eaddr;
22. }
23. public void setEaddr(String eaddr) {
24.     this.eaddr = eaddr;
25. }
26.
27.

```



Account.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Account" table="account">
7.     <id name="accNo" column="ACCNO"/>
8.     <discriminator column="TYPE" type="string"/>
9.     <property name="accName" column="ACCNAME"/>
10.    <property name="accType" column="ACCTYPE"/>
11.
12.    <subclass name="com.durgasoft.pojo.StudentAccount" discriminator-value="std">
13.      <property name="sid" column="SID"/>
14.      <property name="sbranch" column="SBRANCH"/>
15.      <property name="smarks" column="SMARKS"/>
16.    </subclass>
17.
18.    <subclass name="com.durgasoft.pojo.EmployeeAccount" discriminator-
19.      value="emp">
20.        <property name="eid" column="EID"/>
21.        <property name="esal" column="ESAL"/>
22.        <property name="eaddr" column="EADDR"/>
23.    </subclass>
24. </class>
25. </hibernate-mapping>

```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

6. <session-factory>
7.   <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.   <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.   <property name="connection.user">system</property>
10.  <property name="connection.password">durga</property>
11.  <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
    y>
12.  <property name="show_Sql">true</property>
13.  <mapping resource="Account.hbm.xml"/>
14. </session-factory>
15.</hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10.import com.durgasoft.pojo.EmployeeAccount;
11.import com.durgasoft.pojo.StudentAccount;
12.
13.public class Test {
14.
15. public static void main(String[] args) throws Exception {
16.     Configuration cfg = new Configuration();
17.     cfg.configure();
18.     StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
19.     builder.applySettings(cfg.getProperties());
20.     StandardServiceRegistry registry = builder.build();
21.     SessionFactory factory = cfg.buildSessionFactory(registry);
22.     Session session = factory.openSession();
23.
24.     StudentAccount sa = new StudentAccount();
25.     sa.setAccNo("a1");
26.     sa.setAccName("AAA");
27.     sa.setAccType("Savings");
28.     sa.setSid("S1");
29.     sa.setSbranch("CS");
```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

30.    sa.setSmarks(77);
31.
32.    EmployeeAccount ea = new EmployeeAccount();
33.    ea.setAccNo("a2");
34.    ea.setAccName("BBB");
35.    ea.setAccType("Savings");
36.    ea.setEid("E1");
37.    ea.setEsal(5000);
38.    ea.setEaddr("Hyd");
39.
40.    Transaction tx = session.beginTransaction();
41.    session.save(sa);
42.    session.save(ea);
43.    tx.commit();
44.    System.out.println("Student Account Inserted Successfully");
45.    System.out.println("Employee Account Inserted Successfully");
46. }
47.

```

To represent Table Per Class Inheritance Mapping javax.persistence package has provided the following three Annotations.

- 1) @Inheritance
- 2) @DiscriminatorColumn
- 3) @DiscriminatorValue

1) @Inheritance

- ❖ This annotation is able to specify Inheritance Mapping strategy.
`@Inheritance(strategy=value)`
Where value may be SINGLE_TABLE constant from InheritanceType enum.

EX: `@Inheritance(strategy=InheritanceType.SINGLE_TABLE)`

2) @DiscriminatorColumn

- ❖ This annotation is able to represent Discriminator Column.
`@DiscriminatorColumn(name="--")`
Where "name" is able to take discriminator column name.

EX: `@DiscriminatorColumn(name="TYPE")`

Note: The above two annotations must be used at super class.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

3) @DiscriminatorValue

- ❖ It able to provide value to the discriminator column, it must be specified at sub class

Example:**Account.java**

```

1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.DiscriminatorColumn;
5. import javax.persistence.Entity;
6. import javax.persistence.Id;
7. import javax.persistence.Inheritance;
8. import javax.persistence.InheritanceType;
9. import javax.persistence.Table;
10.
11. @Entity
12. @Table(name= "account")
13. @Inheritance(strategy=InheritanceType.SINGLE_TABLE)
14. @DiscriminatorColumn(name="TYPE")
15. public class Account {
16.     @Id
17.     @Column(name="ACCNO")
18.     private String accNo;
19.     @Column(name="ACCNAME")
20.     private String accName;
21.     @Column(name="ACCTYPE")
22.     private String accType;
23.
24.     public String getAccNo() {
25.         return accNo;
26.     }
27.     public void setAccNo(String accNo) {
28.         this.accNo = accNo;
29.     }
30.     public String getAccName() {
31.         return accName;
32.     }
33.     public void setAccName(String accName) {
34.         this.accName = accName;
35.     }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
36. public String getAccType() {  
37.     return accType;  
38. }  
39. public void setAccType(String accType) {  
40.     this.accType = accType;  
41. }  
42.  
43.  
44.}
```

StudentAccount.java



```
1. package com.durgasoft.pojo;  
2.  
3. import javax.persistence.Column;  
4. import javax.persistence.DiscriminatorValue;  
5. import javax.persistence.Entity;  
6.  
7. @Entity  
8. @DiscriminatorValue("std")  
9. public class StudentAccount extends Account{  
10.    @Column(name="SID")  
11.    private String sid;  
12.    @Column(name="SBRANCH")  
13.    private String sbranch;  
14.    @Column(name="SMARKS")  
15.    private int smarks;  
16.  
17.    public String getSid() {  
18.        return sid;  
19.    }  
20.    public void setSid(String sid) {  
21.        this.sid = sid;  
22.    }  
23.    public String getSbranch() {  
24.        return sbranch;  
25.    }  
26.    public void setSbranch(String sbranch) {  
27.        this.sbranch = sbranch;  
28.    }  
29.    public int getSmarks() {  
30.        return smarks;  
31.    }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
32. public void setSmarks(int smarks) {  
33.     this.smarks = smarks;  
34. }  
35.  
36.  
37.}
```

**EmployeeAccount.java**

```
1. package com.durgasoft.pojo;  
2.  
3. import javax.persistence.Column;  
4. import javax.persistence.DiscriminatorValue;  
5. import javax.persistence.Entity;  
6.  
7. @Entity  
8. @DiscriminatorValue("emp")  
9. public class EmployeeAccount extends Account{  
10.    @Column(name="EID")  
11.    private String eid;  
12.    @Column(name="ESAL")  
13.    private float esal;  
14.    @Column(name="EADDR")  
15.    private String eaddr;  
16.  
17.    public String getEid() {  
18.        return eid;  
19.    }  
20.    public void setEid(String eid) {  
21.        this.eid = eid;  
22.    }  
23.    public float getEsal() {  
24.        return esal;  
25.    }  
26.    public void setEsal(float esal) {  
27.        this.esal = esal;  
28.    }  
29.    public String getEaddr() {  
30.        return eaddr;  
31.    }  
32.    public void setEaddr(String eaddr) {  
33.        this.eaddr = eaddr;  
34.    }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
35.  
36.  
37.}
```

hibernate.cfg.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>  
2. <!DOCTYPE hibernate-configuration PUBLIC  
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"  
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">  
5. <hibernate-configuration>  
6.   <session-factory>  
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>  
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>  
9.     <property name="connection.user">system</property>  
10.    <property name="connection.password">durga</property>  
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</propert  
y>  
12.    <property name="show_Sql">true</property>  
13.    <mapping class="com.durgasoft.pojo.StudentAccount"/>  
14.    <mapping class="com.durgasoft.pojo.EmployeeAccount"/>  
15.  </session-factory>  
16.</hibernate-configuration>
```

Test.java

```
1. package com.durgasoft.test;  
2.  
3. import org.hibernate.Session;  
4. import org.hibernate.SessionFactory;  
5. import org.hibernate.Transaction;  
6. import org.hibernate.boot.registry.StandardServiceRegistry;  
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;  
8. import org.hibernate.cfg.Configuration;  
9.  
10.import com.durgasoft.pojo.Account;  
11.import com.durgasoft.pojo.EmployeeAccount;  
12.import com.durgasoft.pojo.StudentAccount;  
13.  
14.public class Test {  
15.  
16.  public static void main(String[] args) throws Exception {  
17.    Configuration cfg = new Configuration();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

18.    cfg.configure();
19.    StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
20.    builder = builder.applySettings(cfg.getProperties());
21.    StandardServiceRegistry registry = builder.build();
22.    SessionFactory factory = cfg.buildSessionFactory(registry);
23.    Session session = factory.openSession();
24.
25.    StudentAccount sa = (StudentAccount)session.get(Account.class, "a1");
26.    System.out.println("Student Account Details");
27.    System.out.println("-----");
28.    System.out.println("Account Number :" +sa.getAccNo());
29.    System.out.println("Account Name :" +sa.getAccName());
30.    System.out.println("Account Type :" +sa.getAccType());
31.    System.out.println("Student Id :" +sa.getSid());
32.    System.out.println("Student Branch :" +sa.getSbranch());
33.    System.out.println("Student Marks :" +sa.getSmrks());
34.
35.    EmployeeAccount ea = (EmployeeAccount)session.get(Account.class, "a2");
36.    System.out.println("Employee Account Details");
37.    System.out.println("-----");
38.    System.out.println("Account Number :" +ea.getAccNo());
39.    System.out.println("Account Name :" +ea.getAccName());
40.    System.out.println("Account Type :" +ea.getAccType());
41.    System.out.println("EmployeeId :" +ea.getEid());
42.    System.out.println("EmployeeSalary :" +ea.getEsal());
43.    System.out.println("Employee Address :" +ea.getEaddr());
44.
45.    session.close();
46.    factory.close();
47.
48. }
49.
50.
}

```

2) Table Per Sub class:

- In case of "Table Per Subclass" inheritance mapping, we have to prepare a separate table for each and every class[for both super class and sub classes] existed in inheritance.
- While preparing tables for sub classes we have to provide a join column which must be primary key in super class respective table and in sub class respective tables.
- In hibernate applications, if we store sub class objects in database then Hibernate software has to recognize and store super class properties data in super class respective table and sub class respective data in sub class respective tables.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



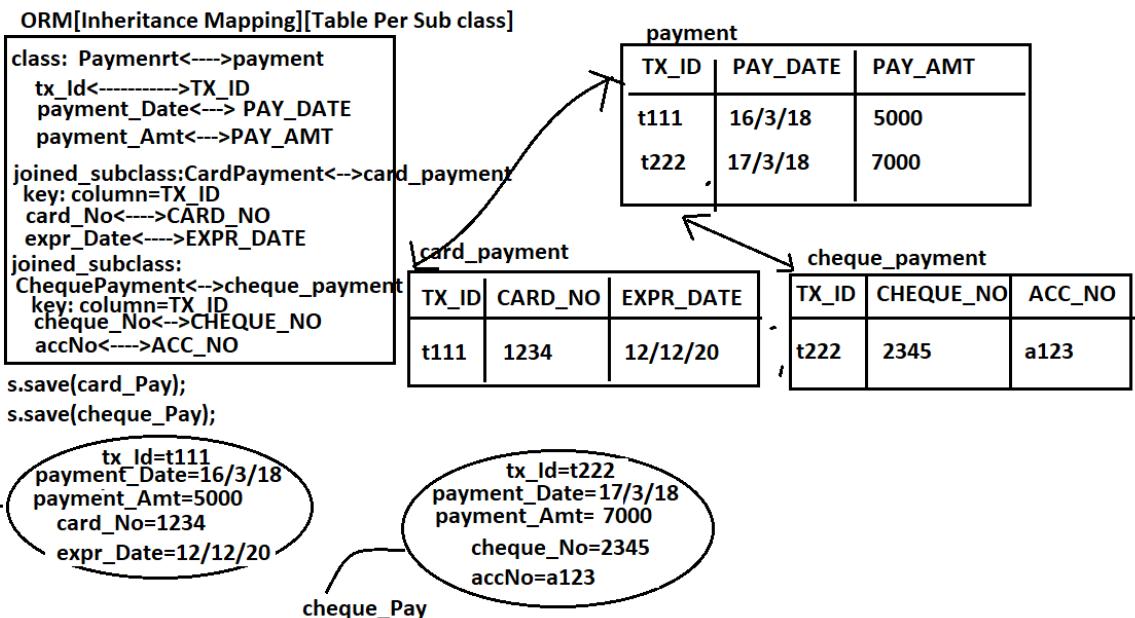
BY NAGOOR BABU

- To achieve this requirement, Hibernate software has to recognize super class properties and sub class properties separately, for this we have to provide sub classes configuration separately in hibernate mapping file.
 - To provide "Table Per Subclass" inheritance mapping we have to use the following xml tags in mapping file.

```
1. <hibernate-mapping>
2.   <class name="--" table="--"
3.     -----
4.       <joined-subclass name="--" table="--">
5.         <key column="--"/>
6.         -----
7.       </join-subclass>
8.     -----
9.   </class>
10.  </hibernate-mapping>
```

- ✓ Where "<joined-subclass>" tag is able to configure sub class, where "name" attribute will take fully qualified name of the respective sub class and "table" attribute will take sub class respective table name.
 - ✓ Where "<key>" tag is able to take join key which is available in sub class respective table.

```
class Payment{  
String tx_Id;  
String payment_Date;  
long payment_Amt;  
---  
}  
class CradPayment  
extends Payment{  
int card_no;  
String expr_Date;  
---  
}  
class ChequePayment  
extends Payment{  
int cheque_no;  
String accNo;
```



CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



 US-NHM 442226786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com



BY NAGOOR BABU

Example:**Payment.java**

```
1. package com.durgasoft.pojo;
2.
3. public class Payment {
4.     private String tx_Id;
5.     private String payment_Date;
6.     private long payment_Amt;
7.
8.     public String getTx_Id() {
9.         return tx_Id;
10.    }
11.    public void setTx_Id(String tx_Id) {
12.        this.tx_Id = tx_Id;
13.    }
14.    public String getPayment_Date() {
15.        return payment_Date;
16.    }
17.    public void setPayment_Date(String payment_Date) {
18.        this.payment_Date = payment_Date;
19.    }
20.    public long getPayment_Amt() {
21.        return payment_Amt;
22.    }
23.    public void setPayment_Amt(long payment_Amt) {
24.        this.payment_Amt = payment_Amt;
25.    }
26.
27.
28.}
```

CardPayment.java

```
1. package com.durgasoft.pojo;
2.
3. public class CardPayment extends Payment {
4.     private int card_No;
5.     private String expr_Date;
6.
7.     public int getCard_No() {
8.         return card_No;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

9.    }
10.   public void setCard_No(int card_No) {
11.     this.card_No = card_No;
12.   }
13.   public String getExpr_Date() {
14.     return expr_Date;
15.   }
16.   public void setExpr_Date(String expr_Date) {
17.     this.expr_Date = expr_Date;
18.   }
19.
20.
21.}
```

ChequePayment.java

```

1. package com.durgasoft.pojo;
2.
3. public class ChequePayment extends Payment {
4.   private int cheque_No;
5.   private String accNo;
6.
7.   public int getCheque_No() {
8.     return cheque_No;
9.   }
10.  public void setCheque_No(int cheque_No) {
11.    this.cheque_No = cheque_No;
12.  }
13.  public String getAccNo() {
14.    return accNo;
15.  }
16.  public void setAccNo(String accNo) {
17.    this.accNo = accNo;
18.  }
19.
20.
21.}
```

payment.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

4. "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Payment" table="payment">
7.     <id name="tx_Id" column="TX_ID"/>
8.     <property name="payment_Date" column="PAY_DATE"/>
9.     <property name="payment_Amt" column="PAY_AMT"/>
10.
11.   <joined-
12.     subclass name="com.durgasoft.pojo.CardPayment" table="card_payment">
13.       <key column="TX_ID"/>
14.       <property name="card_No" column="CARD_NO"/>
15.       <property name="expr_Date" column="EXPR_DATE"/>
16.   </joined-subclass>
17.   <joined-
18.     subclass name="com.durgasoft.pojo.ChequePayment" table="cheque_payment">
19.       <key column="TX_ID"/>
20.       <property name="cheque_No" column="CHEQUE_NO"/>
21.       <property name="accNo" column="ACC_NO"/>
22.   </joined-subclass>
23. </class>
24. </hibernate-mapping>

```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
12.    <property name="show_Sql">true</property>
13.    <mapping resource="payment.hbm.xml"/>
14.  </session-factory>
15. </hibernate-configuration>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```
1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10. import com.durgasoft.pojo.CardPayment;
11. import com.durgasoft.pojo.ChequePayment;
12.
13. public class Test {
14.
15.     public static void main(String[] args) throws Exception {
16.         Configuration cfg = new Configuration();
17.         cfg.configure();
18.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
19.         builder = builder.applySettings(cfg.getProperties());
20.         StandardServiceRegistry registry = builder.build();
21.         SessionFactory factory = cfg.buildSessionFactory(registry);
22.         Session session = factory.openSession();
23.
24.         CardPayment card_Payment = new CardPayment();
25.         card_Payment.setTx_Id("t111");
26.         card_Payment.setPayment_Date("16/3/18");
27.         card_Payment.setPayment_Amt(5000);
28.         card_Payment.setCard_No(1234);
29.         card_Payment.setExpr_Date("12/12/20");
30.
31.         ChequePayment cheque_Payment = new ChequePayment();
32.         cheque_Payment.setTx_Id("t222");
33.         cheque_Payment.setPayment_Date("17/3/18");
34.         cheque_Payment.setPayment_Amt(7000);
35.         cheque_Payment.setCheque_No(2345);
36.         cheque_Payment.setAccNo("a222");
37.
38.         Transaction tx = session.beginTransaction();
39.         session.save(card_Payment);
40.         session.save(cheque_Payment);
41.         tx.commit();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

42.     System.out.println("Card Payment Stored Successfully");
43.     System.out.println("Cheque Payment Stored Successfully");
44. }
45.

```

Note: The above application required the following database

SQL> desc payment;

Name	Null?	Type
TX_ID	NOT NULL	VARCHAR2(5)
PAY_DATE		VARCHAR2(10)
PAY_AMT		

SQL> desc card_payment;

Name	Null?	Type
TX_ID	NOT NULL	VARCHAR2(5)
CARD_NO		NUMBER(8)
EXPR_DATE		VARCHAR2(10)

SQL> desc cheque_payment;

Name	Null?	Type
TX_ID	NOT NULL	VARCHAR2(5)
CHEQUE_NO		NUMBER(8)
ACC_NO		VARCHAR2(10)

SQL>

To represent "Table Per Subclass" inheritance mapping mechanism, javax.persistence package has provided the following Annotations

- 1) **@Inheritance**
- 2) **@PrimaryKeyJoinColumn**

1) @Inheritance

- ❖ This annotation is able to represent a particular Inheritance mapping mechanism, it will use "strategy" member to take a particular inheritance mapping, where "strategy" attribute will take "JOINED" value constant from "InheritanceType" enum.

EX: `@Inheritance(strategy=InheritanceType.JOINED)`

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

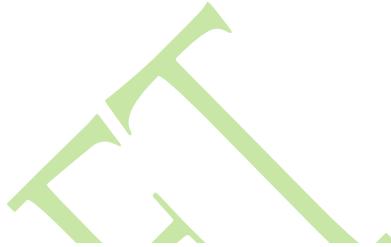


BY NAGOOR BABU

2) @PrimaryKeyJoinColumn

- ❖ This annotation will be used at sub classes inorder to provide Join Key column name. It will use "name" attribute to provide "join key column".

EX: @PrimaryKeyJoinColumn(name="TX_ID")



Example:

Payment.java

```
1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Inheritance;
7. import javax.persistence.InheritanceType;
8. import javax.persistence.Table;
9.
10. @Entity
11. @Table(name="payment")
12. @Inheritance(strategy=InheritanceType.JOINED)
13. public class Payment {
14.     @Id
15.     @Column(name="TX_ID")
16.     private String tx_Id;
17.     @Column(name="PAY_DATE")
18.     private String payment_Date;
19.     @Column(name="PAY_AMT")
20.     private long payment_Amt;
21.
22.     public String getTx_Id() {
23.         return tx_Id;
24.     }
25.     public void setTx_Id(String tx_Id) {
26.         this.tx_Id = tx_Id;
27.     }
28.     public String getPayment_Date() {
29.         return payment_Date;
30.     }
31.     public void setPayment_Date(String payment_Date) {
32.         this.payment_Date = payment_Date;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
33. }
34. public long getPayment_Amt() {
35.     return payment_Amt;
36. }
37. public void setPayment_Amt(long payment_Amt) {
38.     this.payment_Amt = payment_Amt;
39. }
40.}
```



CardPayment.java

```
1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.PrimaryKeyJoinColumn;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="card_payment")
10. @PrimaryKeyJoinColumn(name="TX_ID")
11. public class CardPayment extends Payment {
12.     @Column(name="CARD_NO")
13.     private int card_No;
14.     @Column(name="EXPR_DATE")
15.     private String expr_Date;
16.
17.     public int getCard_No() {
18.         return card_No;
19.     }
20.     public void setCard_No(int card_No) {
21.         this.card_No = card_No;
22.     }
23.     public String getExpr_Date() {
24.         return expr_Date;
25.     }
26.     public void setExpr_Date(String expr_Date) {
27.         this.expr_Date = expr_Date;
28.     }
29.
30.
31.}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

ChequePayment.java

```

1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.PrimaryKeyJoinColumn;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="cheque_payment")
10. @PrimaryKeyJoinColumn(name="TX_ID")
11. public class ChequePayment extends Payment {
12.     @Column(name="CHEQUE_NO")
13.     private int cheque_No;
14.     @Column(name="ACC_NO")
15.     private String accNo;
16.
17.     public int getCheque_No() {
18.         return cheque_No;
19.     }
20.     public void setCheque_No(int cheque_No) {
21.         this.cheque_No = cheque_No;
22.     }
23.     public String getAccNo() {
24.         return accNo;
25.     }
26.     public void setAccNo(String accNo) {
27.         this.accNo = accNo;
28.     }
29.
30.
31. }
```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.      "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.      "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

8.   <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.   <property name="connection.user">system</property>
10.  <property name="connection.password">durga</property>
11.  <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
    y>
12.  <property name="show_Sql">true</property>
13.  <mapping class="com.durgasoft.pojo.Payment"/>
14.  <mapping class="com.durgasoft.pojo.CardPayment"/>
15.  <mapping class="com.durgasoft.pojo.ChequePayment"/>
16. </session-factory>
17.</hibernate-configuration>

```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10.import com.durgasoft.pojo.CardPayment;
11.import com.durgasoft.pojo.ChequePayment;
12.import com.durgasoft.pojo.Payment;
13.
14.public class Test {
15.
16.    public static void main(String[] args) throws Exception {
17.        Configuration cfg = new Configuration();
18.        cfg.configure();
19.        StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
20.        builder = builder.applySettings(cfg.getProperties());
21.        StandardServiceRegistry registry = builder.build();
22.        SessionFactory factory = cfg.buildSessionFactory(registry);
23.        Session session = factory.openSession();
24.
25.        CardPayment card_Payment = (CardPayment) session.get(Payment.class, "t111");
26.        System.out.println("CardPayment Details");
27.        System.out.println("-----");
28.        System.out.println("Transaction ID : "+card_Payment.getTx_Id());
29.        System.out.println("Payment Date : "+card_Payment.getPayment_Date());

```



CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

30. System.out.println("Payment Amount :" + card_Payment.getPayment_Amt());
31. System.out.println("Card Number :" + card_Payment.getCard_No());
32. System.out.println("Expr Date :" + card_Payment.getExpr_Date());
33. System.out.println();
34.
35. ChequePayment cheque_Payment = (ChequePayment) session.get(Payment.class, "t
   222");
36. System.out.println("ChequePayment Details");
37. System.out.println("-----");
38. System.out.println("Transaction ID :" + cheque_Payment.getTx_Id());
39. System.out.println("Payment Date :" + cheque_Payment.getPayment_Date());
40. System.out.println("Payment Amount :" + cheque_Payment.getPayment_Amt());
41. System.out.println("Cheque Number :" + cheque_Payment.getCheque_No());
42. System.out.println("Account Number :" + cheque_Payment.getAccNo());
43.
44.
45. }
46.

```

3. Table Per Concrete Class:

- ★ In "Table per Concreate Class" inheritance mapping , we will prepare a seperate table for each and every sub class containing the columns representing both the super class properties and the respective sub class properties.
- ★ In the above context, if we save any sub class object then Hibernate software has to distribute sub class objects data in the respective table and in the respective columns. To achieve this we have to use the following xml tags in hibernate mapping file.

```

1. <hibernate-mapping>
2. -----
3. <class name="-->
4. -----
5.   <union-subclass name="--" table="-->
6.     <property name="--" column="--"/>
7.   -----
8. </union-subclass>
9. -----
10. </class>
11. </hibernate-mapping>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Where "<union-subclass>" tag is able to provide mapping between a particular sub class and the respective table, where "name" attribute will take sub class name and table attribute will take the respective table name.

```
class Person{
String pname;
String paddr;
-----
} class Employee
extends Person{
String eid;
float esal;
-----
} class Customer
extends Person{
String cid;
String mobile;
-----
}

class Person{
String pname;
String paddr;
}
class Employee
extends Person{
String eid;
float esal;
}
class Customer
extends Person{
String cid;
String mobile;
}

class Person{
String pname;
String paddr;
}
class Employee
extends Person{
String eid;
float esal;
}
class Customer
extends Person{
String cid;
String mobile;
}

class Person{
String pname;
String paddr;
}
class Employee
extends Person{
String eid;
float esal;
}
class Customer
extends Person{
String cid;
String mobile;
}
```

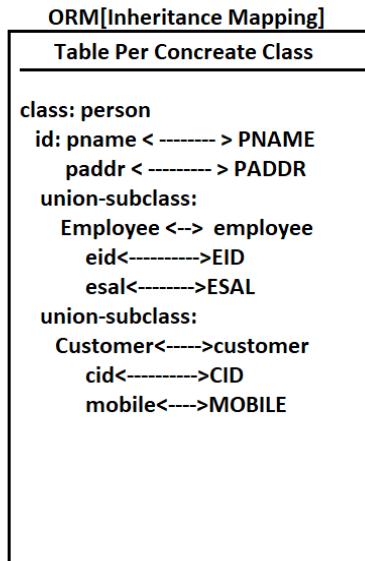
Employee <--> employee
 eid<----->EID
 esal<----->ESAL

Customer<---->customer
 cid<----->CID
 mobile<--->MOBILE

emp
 s.save(emp);
 s.ssave(cust);

cust
 cname=AAA
 paddr=Hyd
 eid=E1
 esal=5000

cname=BBB
 paddr=Hyd
 cid=C1
 cmobile=99887766



employee

PNAME	PADDR	EID	ESAL
AAA	Hyd	E1	5000

customer

PNAME	PADDR	CID	MOBILE
BBB	Hyd	C1	99887766

Example:

Person.java

```
1. package com.durgasoft.pojo;
2.
3. public class Person {
4.     private String pname;
5.     private String paddr;
6.
7.     public String getPname() {
8.         return pname;
9.     }
10.    public void setPname(String pname) {
11.        this.pname = pname;
12.    }
13.    public String getPaddr() {
14.        return paddr;
15.    }
16.    public void setPaddr(String paddr) {
17.        this.paddr = paddr;
18.    }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
19.  
20.  
21.}
```

Employee.java

```
1. package com.durgasoft.pojo;  
2.  
3. public class Employee extends Person {  
4.     private String eid;  
5.     private float esal;  
6.  
7.     public String getEid() {  
8.         return eid;  
9.     }  
10.    public void setEid(String eid) {  
11.        this.eid = eid;  
12.    }  
13.    public float getEsal() {  
14.        return esal;  
15.    }  
16.    public void setEsal(float esal) {  
17.        this.esal = esal;  
18.    }  
19.  
20.  
21.}
```

Customer.java

```
1. package com.durgasoft.pojo;  
2.  
3. public class Customer extends Person {  
4.     private String cid;  
5.     private String cmobile;  
6.  
7.     public String getCid() {  
8.         return cid;  
9.     }  
10.    public void setCid(String cid) {  
11.        this.cid = cid;  
12.    }  
13.    public String getCmobile() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

14.     return cmobile;
15. }
16. public void setCmobile(String cmobile) {
17.     this.cmobile = cmobile;
18. }
19.
20.
21.}
```



Person.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Person">
7.     <id name="pname" column="PNAME"/>
8.     <property name="paddr" column="PADDR"/>
9.     <union-subclass name="com.durgasoft.pojo.Employee" table="employee">
10.       <property name="eid" column="EID"/>
11.       <property name="esal" column="ESAL"/>
12.     </union-subclass>
13.     <union-subclass name="com.durgasoft.pojo.Customer" table="customer">
14.       <property name="cid" column="CID"/>
15.       <property name="cmobile" column="CMOBILE"/>
16.     </union-subclass>
17.   </class>
18.</hibernate-mapping>
```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.user">system</property>
10.    <property name="connection.password">durga</property>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
11.   <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</proper  
y>  
12.   <property name="show_Sql">true</property>  
13.   <mapping resource="Person.hbm.xml"/>  
14. </session-factory>  
15.</hibernate-configuration>
```

**Test.java**

```
1. package com.durgasoft.test;  
2.  
3. import org.hibernate.Session;  
4. import org.hibernate.SessionFactory;  
5. import org.hibernate.Transaction;  
6. import org.hibernate.boot.registry.StandardServiceRegistry;  
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;  
8. import org.hibernate.cfg.Configuration;  
9.  
10.  
11.import com.durgasoft.pojo.Customer;  
12.import com.durgasoft.pojo.Employee;  
13.  
14.public class Test {  
15.  
16.    public static void main(String[] args) throws Exception {  
17.        Configuration cfg = new Configuration();  
18.        cfg.configure();  
19.        StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();  
20.        builder = builder.applySettings(cfg.getProperties());  
21.        StandardServiceRegistry registry = builder.build();  
22.        SessionFactory factory = cfg.buildSessionFactory(registry);  
23.        Session session = factory.openSession();  
24.  
25.        Employee emp = new Employee();  
26.        emp.setPname("AAA");  
27.        emp.setPaddr("Hyd");  
28.        emp.setEid("E-111");  
29.        emp.setEsal(5000);  
30.  
31.        Customer cust = new Customer();  
32.        cust.setPname("BBB");  
33.        cust.setPaddr("Hyd");  
34.        cust.setCid("C-111");
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

35.     cust.setCmobile("91-9988776655");
36.
37.     Transaction tx = session.beginTransaction();
38.     session.save(emp);
39.     session.save(cust);
40.     tx.commit();
41.     System.out.println("Employee Object Inserted Successfully");
42.     System.out.println("Customer Object Inserted Successfully");
43.
44.     session.close();
45.     factory.close();
46.
47. }
48.
49.

```

Database:

SQL> create table employee (PNAME varchar2(10) primary key, PADDR varchar2(10), EID varchar2(5), ESAL float);

SQL> commit;

SQL> create table customer(PNAME varchar2(10) primary key, PADDR varchar2(10), CID varchar2(5), CMOBILE varchar2(20));

SQL> commit;

To represent this inheritance mapping mechanism we have to use `@Inheritance()` annotation , where we have to use "TABLE_PER_CLASS" constant from INheritanceType enum as value to 'strategy' member.

`@Inheritance(strategy = InheritanceType.TABLE_PER_CLASS)`

Example:

Person.java

```

1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Inheritance;
7. import javax.persistence.InheritanceType;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
8.  
9. @Entity  
10. @Inheritance(strategy=InheritanceType.TABLE_PER_CLASS)  
11. public class Person {  
12.     @Id  
13.     @Column(name="PNAME")  
14.     private String pname;  
15.     @Column(name="PADDR")  
16.     private String paddr;  
17.  
18.     public String getPname() {  
19.         return pname;  
20.     }  
21.     public void setPname(String pname) {  
22.         this.pname = pname;  
23.     }  
24.     public String getPaddr() {  
25.         return paddr;  
26.     }  
27.     public void setPaddr(String paddr) {  
28.         this.paddr = paddr;  
29.     }  
30.  
31.  
32.}
```

Employee.java

```
1. package com.durgasoft.pojo;  
2.  
3. import javax.persistence.Column;  
4. import javax.persistence.Entity;  
5. import javax.persistence.Table;  
6.  
7. @Entity  
8. @Table(name="employee")  
9. public class Employee extends Person {  
10.    @Column(name="EID")  
11.    private String eid;  
12.    @Column(name="ESAL")  
13.    private float esal;  
14.  
15.    public String getEid() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
16.     return eid;
17. }
18. public void setEid(String eid) {
19.     this.eid = eid;
20. }
21. public float getEsal() {
22.     return esal;
23. }
24. public void setEsal(float esal) {
25.     this.esal = esal;
26. }
27.
28.
29.}
```

Customer.java

```
1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Table;
6.
7. @Entity
8. @Table(name="customer")
9. public class Customer extends Person {
10.     @Column(name="CID")
11.     private String cid;
12.     @Column(name="CMOBILE")
13.     private String cmobile;
14.
15.     public String getCid() {
16.         return cid;
17.     }
18.     public void setCid(String cid) {
19.         this.cid = cid;
20.     }
21.     public String getCmobile() {
22.         return cmobile;
23.     }
24.     public void setCmobile(String cmobile) {
25.         this.cmobile = cmobile;
26.     }
}
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
27.  
28.  
29.}
```

hibernate.cfg.xml

```
1. <?xml version="1.0" encoding="UTF-8"?>  
2. <!DOCTYPE hibernate-configuration PUBLIC  
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"  
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">  
5. <hibernate-configuration>  
6.   <session-factory>  
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>  
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>  
9.     <property name="connection.user">system</property>  
10.    <property name="connection.password">durga</property>  
11.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</propert  
y>  
12.    <property name="show_Sql">true</property>  
13.    <!-- <mapping resource="Person.hbm.xml"/> -->  
14.    <mapping class="com.durgasoft.pojo.Person"/>  
15.    <mapping class="com.durgasoft.pojo.Employee"/>  
16.    <mapping class="com.durgasoft.pojo.Customer"/>  
17.  </session-factory>  
18.</hibernate-configuration>
```

Test.java

```
1. package com.durgasoft.test;  
2.  
3. import org.hibernate.Session;  
4. import org.hibernate.SessionFactory;  
5. import org.hibernate.Transaction;  
6. import org.hibernate.boot.registry.StandardServiceRegistry;  
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;  
8. import org.hibernate.cfg.Configuration;  
9. import com.durgasoft.pojo.Customer;  
10. import com.durgasoft.pojo.Employee;  
11.  
12. public class Test {  
13.  
14.   public static void main(String[] args) throws Exception {  
15.     Configuration cfg = new Configuration();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

16.    cfg.configure();
17.    StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
18.    builder = builder.applySettings(cfg.getProperties());
19.    StandardServiceRegistry registry = builder.build();
20.    SessionFactory factory = cfg.buildSessionFactory(registry);
21.    Session session = factory.openSession();
22.
23.    Employee emp = (Employee)session.get(Employee.class, "AAA");
24.    Customer cust = (Customer)session.get(Customer.class, "BBB");
25.
26.    System.out.println("Employee Details");
27.    System.out.println("-----");
28.    System.out.println("Person Name :"+emp.getPname());
29.    System.out.println("Person Address :" +emp.getPaddr());
30.    System.out.println("Employee Id :"+emp.getEid());
31.    System.out.println("Employee Salary:" +emp.getEsal());
32.    System.out.println();
33.    System.out.println("Customer Details");
34.    System.out.println("-----");
35.    System.out.println("Person Name :"+cust.getPname());
36.    System.out.println("Person Address :" +cust.getPaddr());
37.    System.out.println("Customer Id :"+cust.getCid());
38.    System.out.println("Customer Mobile:" +cust.getCmpile());
39.    session.close();
40.    factory.close();
41.
42. }
43.
44.}
```

4. Association Mapping:

- ★ In General, in Enterprise applications we will use both Object Oriented Data Model and Relational Data Model to represent data. In Enterprise applications we will use Object Oriented Data Model in Front-End applications to represent data and we will use Relational data Model in Back-End Applications to represent Data.
- ★ In the above context, both the data models are having their own approaches to represent data, the differences between data models may provide mismatches between data models, it may reduce data persistency in Enterprise Applications.
- ★ In the above context, to improve data persistency we must use ORM implementations, Hibernate is an ORM implementation , it able to provide solutions for mismatch between Data models.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ★ In Enterprise applications, we are able to get Associations mismatch while achieving associations in Object Oriented Data Model and in Relational Data Model.
- ★ IN Enterprise Applications, the main intention of Associations is to improve data navigation between entities and to improve communication between entities.
- ★ In Object Oriented Data Model, we are able to achieve associations by declaring one or Collection of reference variables of an entity class in another entity class.

EX:

```

1. class Account{
2. }
3. class Address{
4. }
5.
6. class Employee{
7. ----
8. Account acc;// one-to-one association
9. ----
10. Collection<Address> addr; //one-to-many
11. ----
12. }
```

In relational Data Model, we are able to achieve associations in the following three ways.

- 1) By defining Primary Key-Foreign Key relations between tables
- 2) By Providing Join column between tables
- 3) By Defining Join table between tables.

To resolve the above associations mismatch between both the data models Hibernate has provided "Association Mappings".

Hibernate has provided the following four types of Associations mappings inorder to resolve Associations mismatch.

1. One-To-One Association Mapping
2. One-To-Many Association Mapping
3. Many-To-One Association Mapping
4. Many-To-Many Association Mapping

1. One-To-One Association Mapping

It is a relation between entities, where one instance of an entity should be mapped with exactly one instance of another entity.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



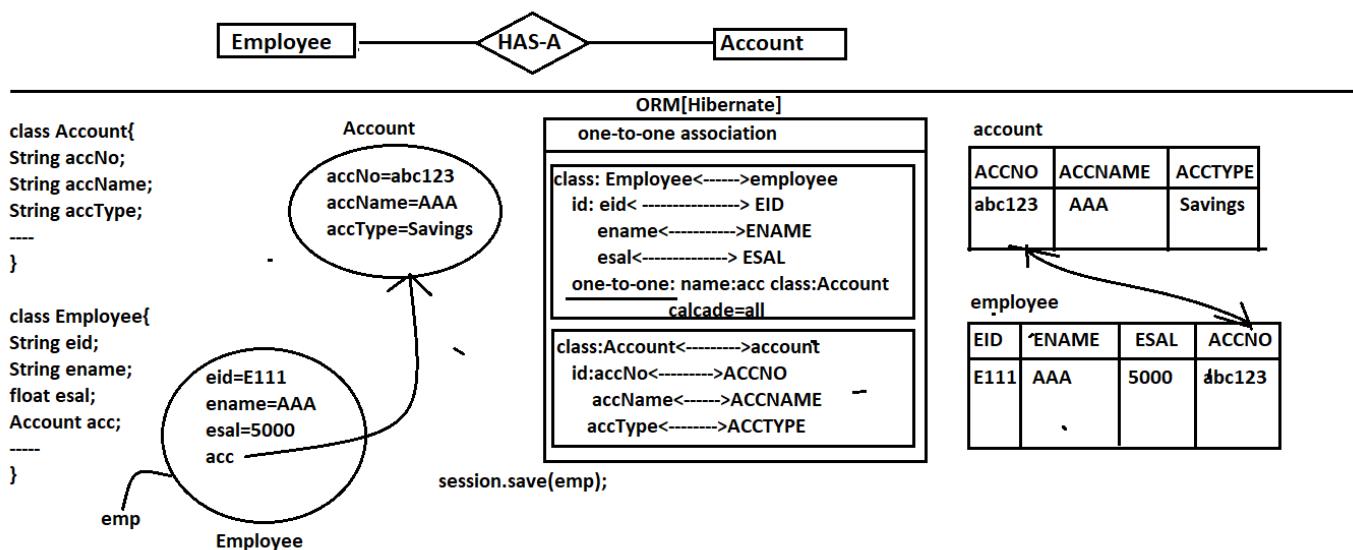
BY NAGOOR BABU

EX: Every Employee must have exactly one individual Account.

In Hibernate applications, to achieve One-To-One Associations, we have to use the following tag in mapping file.

```
<one-to-one name="--" class="--" cascade="--"/>
```

- ✓ Where "name" attribute take property name of the entity class.
- ✓ Where "class" attribute will take Fully qualified name of the associated class.
- ✓ Where "cascade" attribute will take the values like all, none, insert, update and delete inorder to perform operations in cascading style, that is, if we delete employee table automatically Account table must also be deleted.



Example:

Employee.java

```

1. package com.durgasoft.pojo;
2.
3. public class Employee {
4.     private String eid;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.     private Account acc;
9.
  
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

10. public Account getAcc() {
11.     return acc;
12. }
13. public void setAcc(Account acc) {
14.     this.acc = acc;
15. }
16. public String getEid() {
17.     return eid;
18. }
19. public void setEid(String eid) {
20.     this.eid = eid;
21. }
22. public String getEname() {
23.     return ename;
24. }
25. public void setEname(String ename) {
26.     this.ename = ename;
27. }
28. public float getEsal() {
29.     return esal;
30. }
31. public void setEsal(float esal) {
32.     this.esal = esal;
33. }
34. public String getEaddr() {
35.     return eaddr;
36. }
37. public void setEaddr(String eaddr) {
38.     this.eaddr = eaddr;
39. }
40.
41.}
```

Account.java

```

1. package com.durgasoft.pojo;
2.
3. public class Account {
4.     private String accNo;
5.     private String accName;
6.     private String accType;
7.
8.     public String getAccNo() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

9.     return accNo;
10.    }
11.   public void setAccNo(String accNo) {
12.     this.accNo = accNo;
13.   }
14.   public String getAccName() {
15.     return accName;
16.   }
17.   public void setAccName(String accName) {
18.     this.accName = accName;
19.   }
20.   public String getAccType() {
21.     return accType;
22.   }
23.   public void setAccType(String accType) {
24.     this.accType = accType;
25.   }
26.
27.

```

Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Employee" table="emp1">
7.     <id name="eid" column="EID" length="5"/>
8.     <property name="ename" column="ENAME" length="10"/>
9.     <property name="esal" column="ESAL" length="10"/>
10.    <property name="eaddr" column="EADDR" length="10"/>
11.    <one-to-one name="acc" class="com.durgasoft.pojo.Account" cascade="all"/>
12.  </class>
13. </hibernate-mapping>

```

Account.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

6. <class name="com.durgasoft.pojo.Account" table="account">
7.   <id name="accNo" column="ACCNO" length="10"/>
8.   <property name="accName" column="ACCNAME" length="10"/>
9.   <property name="accType" column="ACCTYPE" length="10"/>
10.  </class>
11. </hibernate-mapping>
```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.username">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.hbm2ddl.auto">create</property>
12.    <property name="hibernate.dialect">org.hibernate.dialect.OracleDialect</property>
13.    <mapping resource="Employee.hbm.xml"/>
14.    <mapping resource="Account.hbm.xml"/>
15.  </session-factory>
16. </hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10. import com.durgasoft.pojo.Account;
11. import com.durgasoft.pojo.Employee;
12.
13. public class Test {
14.
15.   public static void main(String[] args) throws Exception {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
16. Configuration cfg = new Configuration();
17. cfg.configure();
18. StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
19. builder = builder.applySettings(cfg.getProperties());
20. StandardServiceRegistry registry = builder.build();
21. SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
22.
23. Session session = sessionFactory.openSession();
24.
25. Account acc = new Account();
26. acc.setAccNo("abc123");
27. acc.setAccName("Durga");
28. acc.setAccType("Savings");
29.
30. Employee emp = new Employee();
31. emp.setEid("E-111");
32. emp.setEname("Durga");
33. emp.setEsal(25000);
34. emp.setEaddr("Hyd");
35. emp.setAcc(acc);
36.
37. Transaction tx = session.beginTransaction();
38. session.save(emp);
39. tx.commit();
40. System.out.println("Employee Inserted Successfully");
41. session.close();
42. sessionFactory.close();
43.
44. }
45.
46.}
```

In Hibernate Applications , to represent one-to-one association JPA has provided the following annotation as part of javax.persistence package.

`@OneToOne(cascade=Val)`

Where val may be the constants like ALL, DETACH, MERGE, REFRESH, REMOVE, PERSIST from CascadeType enum.

The above annotation must be provided just before the entity reference variable in container entity or just above of the getXXX() method w.r.t the entity reference variable.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX:**Employee.java**

```
1. package com.durgasoft.pojo;
2.
3. import javax.persistence.CascadeType;
4. import javax.persistence.Column;
5. import javax.persistence.Entity;
6. import javax.persistence.Id;
7. import javax.persistence.OneToOne;
8. import javax.persistence.Table;
9.
10. @Entity
11. @Table(name="emp1")
12. public class Employee {
13.     @Id
14.     @Column(name="EID", length=5)
15.     private String eid;
16.     @Column(name="ENAME", length=10)
17.     private String ename;
18.     @Column(name="ESAL", length=6)
19.     private float esal;
20.     @Column(name="EADDR", length=10)
21.     private String eaddr;
22.     @OneToOne(cascade=CascadeType.ALL)
23.     private Account acc;
24.
25.     public Account getAcc() {
26.         return acc;
27.     }
28.     public void setAcc(Account acc) {
29.         this.acc = acc;
30.     }
31.     public String getEid() {
32.         return eid;
33.     }
34.     public void setEid(String eid) {
35.         this.eid = eid;
36.     }
37.     public String getEname() {
38.         return ename;
39.     }
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
40. public void setEname(String ename) {  
41.     this.ename = ename;  
42. }  
43. public float getEsal() {  
44.     return esal;  
45. }  
46. public void setEsal(float esal) {  
47.     this.esal = esal;  
48. }  
49. public String getEaddr() {  
50.     return eaddr;  
51. }  
52. public void setEaddr(String eaddr) {  
53.     this.eaddr = eaddr;  
54. }  
55.  
56.}
```

Account.java



```
1. package com.durgasoft.pojo;  
2.  
3. import javax.persistence.Column;  
4. import javax.persistence.Entity;  
5. import javax.persistence.Id;  
6. import javax.persistence.Table;  
7.  
8. @Entity  
9. @Table(name="account")  
10. public class Account {  
11.     @Id  
12.     @Column(name="ACCNO", length=10)  
13.     private String accNo;  
14.     @Column(name="ACCNAME", length=10)  
15.     private String accName;  
16.     @Column(name="ACCTYPE", length=10)  
17.     private String accType;  
18.  
19.     public String getAccNo() {  
20.         return accNo;  
21.     }  
22.     public void setAccNo(String accNo) {  
23.         this.accNo = accNo;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

24. }
25. public String getAccName() {
26.     return accName;
27. }
28. public void setAccName(String accName) {
29.     this.accName = accName;
30. }
31. public String getAccType() {
32.     return accType;
33. }
34. public void setAccType(String accType) {
35.     this.accType = accType;
36. }
37.
38.

```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.username">system</property>
10.    <property name="connection.password">durga</property>
11.    <!-- <property name="hibernate.hbm2ddl.auto">create</property> -->
12.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</propert
    y>
13.    <property name="show_sql">true</property>
14.    <mapping class="com.durgasoft.pojo.Employee"/>
15.    <mapping class="com.durgasoft.pojo.Account"/>
16.  </session-factory>
17. </hibernate-configuration>

```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10. import com.durgasoft.pojo.Account;
11. import com.durgasoft.pojo.Employee;
12.
13. public class Test {
14.
15.     public static void main(String[] args) throws Exception {
16.         Configuration cfg = new Configuration();
17.         cfg.configure();
18.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
19.         builder = builder.applySettings(cfg.getProperties());
20.         StandardServiceRegistry registry = builder.build();
21.         SessionFactory sessionFactory = cfg.buildSessionFactory(registry);
22.
23.         Session session = sessionFactory.openSession();
24.         /*
25.         Account acc = new Account();
26.         acc.setAccNo("abc123");
27.         acc.setAccName("Durga");
28.         acc.setAccType("Savings");
29.
30.         Employee emp = new Employee();
31.         emp.setEid("E-111");
32.         emp.setEname("Durga");
33.         emp.setEsal(25000);
34.         emp.setEaddr("Hyd");
35.         emp.setAcc(acc);
36.
37.         Transaction tx = session.beginTransaction();
38.         session.save(emp);
39.         tx.commit();
40.         System.out.println("Employee Inserted Successfully");
41.         */
42.         Employee emp = (Employee)session.get("com.durgasoft.pojo.Employee", "E-111");
43.         System.out.println("Employee Details");
44.         System.out.println("-----");
45.         System.out.println("Employee Id : "+emp.getEid());
46.         System.out.println("Employee Name : "+emp.getEname());
47.         System.out.println("Employee Salary : "+emp.getEsal());
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

48.     System.out.println("Employee Address :" +emp.getEaddr());
49.
50.     Account acc = emp.getAcc();
51.     System.out.println(acc);
52.     System.out.println("Account Details");
53.     System.out.println("-----");
54.     System.out.println("Account Number :" +acc.getAccNo());
55.     System.out.println("Account Name :" +acc.getAccName());
56.     System.out.println("Account Type :" +acc.getAccType());
57.
58.     session.close();
59.     sessionFactory.close();
60.
61. }
62.

```

2. One-To-Many Association:

It is a relation between entities where one instance of an entity should be mapped with multiple instances of another entity.

EX: Single Department has Multiple Employees

In Hibernate applications, to represent One-To-Many association we have to use the following tags in mapping files.

```

1. <hibernate-mapping>
2.   <class name="--" table="-->
3.   -----
4.     <set name="--" cascade="-->
5.       <key column="--"/>
6.       <one-to-many class="--"/>
7.     </set>
8.   -----
9. </class>
10. </hibernate-mapping>

```

- ✓ Where **<set>** tag is able to represent Collection object, that is, many side in one-to-many and in many-to-many.
- ✓ Where "name" attribute is representing property name which is representing Collection.
- ✓ Where "cascade" attribute will take cascade values like all, insert,....
- ✓ Where **<key>** tag can be used to specify Join key configuration.
- ✓ Where "column" attribute in **<key>** tag is able to join column name.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

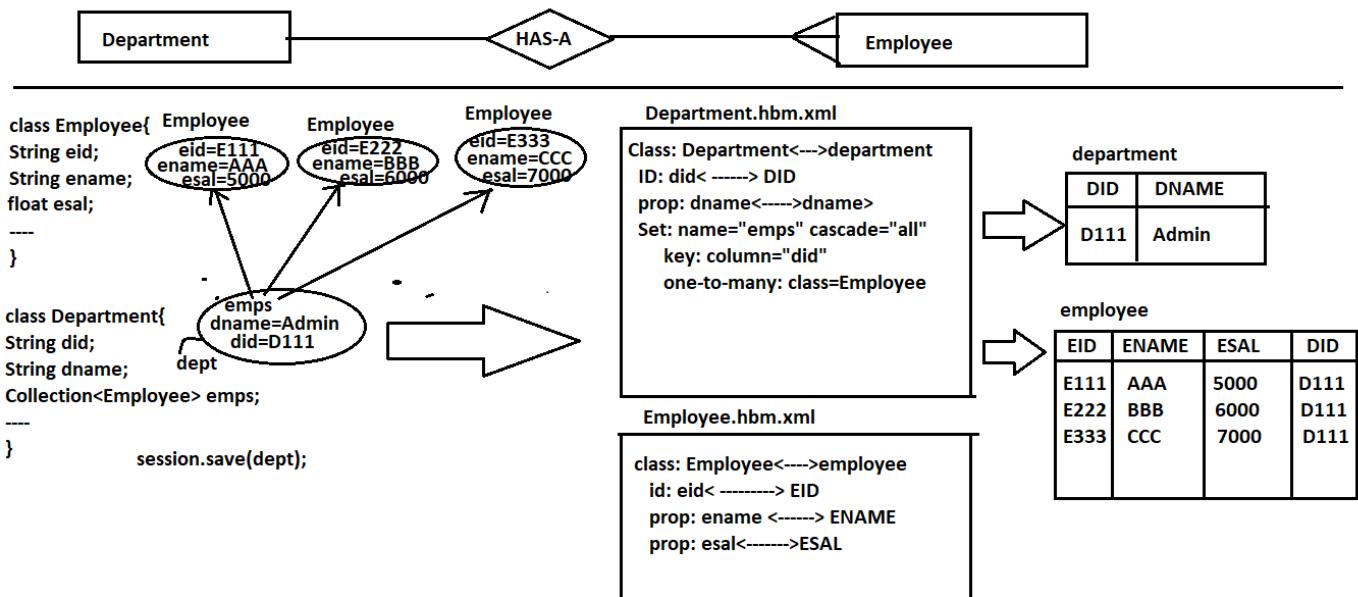
WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

- ✓ Where "<one-to-many>" tag is able to represent one-To-many association.
- ✓ Where "class" attribute in <one-to-many> tag will take class name that is contained class.



Example:

Employee.java

```

1. package com.durgasoft.pojo;
2.
3. public class Employee {
4.     private String eid;
5.     private String ename;
6.     private float esal;
7.     private String eaddr;
8.
9.     public Employee(String eid, String ename, float esal, String eaddr) {
10.         this.eid = eid;
11.         this.ename = ename;
12.         this.esal = esal;
13.         this.eaddr = eaddr;
14.     }
15.
16.     public String getEid() {
17.         return eid;
18.     }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
19. public void setEid(String eid) {  
20.     this.eid = eid;  
21. }  
22. public String getEname() {  
23.     return ename;  
24. }  
25. public void setEname(String ename) {  
26.     this.ename = ename;  
27. }  
28. public float getEsal() {  
29.     return esal;  
30. }  
31. public void setEsal(float esal) {  
32.     this.esal = esal;  
33. }  
34. public String getEaddr() {  
35.     return eaddr;  
36. }  
37. public void setEaddr(String eaddr) {  
38.     this.eaddr = eaddr;  
39. }  
40.  
41.  
42.}
```

Department.java

```
1. package com.durgasoft.pojo;  
2.  
3. import java.util.Set;  
4.  
5. public class Department {  
6.     private String did;  
7.     private String dname;  
8.     private Set<Employee> emps;  
9.  
10.    public String getDid() {  
11.        return did;  
12.    }  
13.    public void setDid(String did) {  
14.        this.did = did;  
15.    }  
16.    public String getDname() {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

17.     return dname;
18. }
19. public void setDname(String dname) {
20.     this.dname = dname;
21. }
22. public Set<Employee> getEmps() {
23.     return emps;
24. }
25. public void setEmps(Set<Employee> emps) {
26.     this.emps = emps;
27. }
28.
29.
30.}
```

Employee.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.     "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.     "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.     <class name="com.durgasoft.pojo.Employee" table="emp1">
7.         <id name="eid" column="EID" length="5"/>
8.         <property name="ename" column="ENAME" length="10"/>
9.         <property name="esal" column="ESAL" length="10"/>
10.        <property name="eaddr" column="EADDR" length="10"/>
11.    </class>
12. </hibernate-mapping>
```

Department.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.     "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.     "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.     <class name="com.durgasoft.pojo.Department" table="dept">
7.         <id name="did" column="DID" length="5"/>
8.         <property name="dname" column="DNAME" length="10"/>
9.         <set name="emps" cascade="all">
10.            <key column="DID"/>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

11.      <one-to-many class="com.durgasoft.pojo.Employee"/>
12.    </set>
13.  </class>
14.</hibernate-mapping>
```

hibernate.cfg.xml

```

1.  <?xml version="1.0" encoding="UTF-8"?>
2.  <!DOCTYPE hibernate-configuration PUBLIC
3.    "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.    "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5.  <hibernate-configuration>
6.    <session-factory>
7.      <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.      <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.      <property name="connection.username">system</property>
10.     <property name="connection.password">durga</property>
11.     <property name="hibernate.hbm2ddl.auto">create</property>
12.     <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
13.     <property name="show_sql">true</property>
14.     <mapping resource="Employee.hbm.xml"/>
15.     <mapping resource="Department.hbm.xml"/>
16.   </session-factory>
17.</hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import java.util.HashSet;
4. import java.util.Set;
5.
6. import org.hibernate.Session;
7. import org.hibernate.SessionFactory;
8. import org.hibernate.Transaction;
9. import org.hibernate.boot.registry.StandardServiceRegistry;
10. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
11. import org.hibernate.cfg.Configuration;
12.
13. import com.durgasoft.pojo.Department;
14. import com.durgasoft.pojo.Employee;
15.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
16. public class Test {  
17.  
18.     public static void main(String[] args) throws Exception{  
19.         Configuration config = new Configuration();  
20.         config.configure();  
21.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();  
22.         builder = builder.applySettings(config.getProperties());  
23.         StandardServiceRegistry registry = builder.build();  
24.         SessionFactory sessionFactory = config.buildSessionFactory(registry);  
25.         Session session = sessionFactory.openSession();  
26.  
27.         Employee e1 = new Employee("E-111", "AAA", 5000, "Hyd");  
28.         Employee e2 = new Employee("E-222", "BBB", 6000, "Hyd");  
29.         Employee e3 = new Employee("E-333", "CCC", 7000, "Hyd");  
30.         Employee e4 = new Employee("E-444", "DDD", 8000, "Hyd");  
31.  
32.         Set<Employee> emps = new HashSet<Employee>();  
33.         emps.add(e1);  
34.         emps.add(e2);  
35.         emps.add(e3);  
36.         emps.add(e4);  
37.  
38.         Department dept = new Department();  
39.         dept.setDid("D-111");  
40.         dept.setDname("ADMIN");  
41.         dept.setEmps(emps);  
42.  
43.         Transaction tx = session.beginTransaction();  
44.         session.save(dept);  
45.         tx.commit();  
46.         System.out.println("Department Inserted Successfully");  
47.         session.close();  
48.         sessionFactory.close();  
49.     }  
50. }
```

In Hibernate Applications, to represent One-To-Many association , JPA has provided the following annotation.

@OneToMany(cascade=CascadeType.ALL)

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

EX:**Employee.java**

```
1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="emp1")
10. public class Employee {
11.     @Id
12.     @Column(name="EID", length=5)
13.     private String eid;
14.     @Column(name="ENAME", length=10)
15.     private String ename;
16.     @Column(name="ESAL", length=5)
17.     private float esal;
18.     @Column(name="EADDR", length=10)
19.     private String eaddr;
20.
21.     public Employee(String eid, String ename, float esal, String eaddr) {
22.         this.eid = eid;
23.         this.ename = ename;
24.         this.esal = esal;
25.         this.eaddr = eaddr;
26.     }
27.
28.     public Employee(){
29.
30.     }
31.
32.     public String getEid() {
33.         return eid;
34.     }
35.     public void setEid(String eid) {
36.         this.eid = eid;
37.     }
38.     public String getEname() {
39.         return ename;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

40. }
41. public void setEname(String ename) {
42.     this.ename = ename;
43. }
44. public float getEsal() {
45.     return esal;
46. }
47. public void setEsal(float esal) {
48.     this.esal = esal;
49. }
50. public String getEaddr() {
51.     return eaddr;
52. }
53. public void setEaddr(String eaddr) {
54.     this.eaddr = eaddr;
55. }
56.
57.
58.

```

Department.java

```

1. package com.durgasoft.pojo;
2.
3. import java.util.Set;
4.
5. import javax.persistence.CascadeType;
6. import javax.persistence.Column;
7. import javax.persistence.Entity;
8. import javax.persistence.Id;
9. import javax.persistence.OneToMany;
10. import javax.persistence.Table;
11. @Entity
12. @Table(name="dept")
13. public class Department {
14.     @Id
15.     @Column(name="DID", length=5)
16.     private String did;
17.     @Column(name="DNAME", length=10)
18.     private String dname;
19.     @OneToMany(cascade=CascadeType.ALL)
20.     private Set<Employee> emps;
21.

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

22. public String getDid() {
23.     return did;
24. }
25. public void setDid(String did) {
26.     this.did = did;
27. }
28. public String getDname() {
29.     return dname;
30. }
31. public void setDname(String dname) {
32.     this.dname = dname;
33. }
34. public Set<Employee> getEmps() {
35.     return emps;
36. }
37. public void setEmps(Set<Employee> emps) {
38.     this.emps = emps;
39. }
40.
41.}
```

hibernate.cfg.xml



```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.username">system</property>
10.    <property name="connection.password">durga</property>
11.    <!-- <property name="hibernate.hbm2ddl.auto">create</property> -->
12.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</propert
y>
13.    <property name="show_sql">true</property>
14.    <mapping class="com.durgasoft.pojo.Employee"/>
15.    <mapping class="com.durgasoft.pojo.Department"/>
16.  </session-factory>
17.</hibernate-configuration>
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Test.java

```
1. package com.durgasoft.test;
2.
3. import java.util.HashSet;
4. import java.util.Set;
5.
6. import org.hibernate.Session;
7. import org.hibernate.SessionFactory;
8. import org.hibernate.Transaction;
9. import org.hibernate.boot.registry.StandardServiceRegistry;
10. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
11. import org.hibernate.cfg.Configuration;
12.
13. import com.durgasoft.pojo.Department;
14. import com.durgasoft.pojo.Employee;
15.
16. public class Test {
17.
18.     public static void main(String[] args) throws Exception{
19.         Configuration config = new Configuration();
20.         config.configure();
21.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
22.         builder = builder.applySettings(config.getProperties());
23.         StandardServiceRegistry registry = builder.build();
24.         SessionFactory sessionFactory = config.buildSessionFactory(registry);
25.         Session session = sessionFactory.openSession();
26.         /*
27.         Employee e1 = new Employee("E-111", "AAA", 5000, "Hyd");
28.         Employee e2 = new Employee("E-222", "BBB", 6000, "Hyd");
29.         Employee e3 = new Employee("E-333", "CCC", 7000, "Hyd");
30.         Employee e4 = new Employee("E-444", "DDD", 8000, "Hyd");
31.
32.         Set<Employee> emps = new HashSet<Employee>();
33.         emps.add(e1);
34.         emps.add(e2);
35.         emps.add(e3);
36.         emps.add(e4);
37.
38.         Department dept = new Department();
39.         dept.setDid("D-111");
40.         dept.setDname("ADMIN");
41.         dept.setEmps(emps);
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

42.
43.     Transaction tx = session.beginTransaction();
44.     session.save(dept);
45.     tx.commit();
46.     System.out.println("Department Inserted Successfully");
47.     */
48.     Department dept = (Department)session.get(Department.class, "D-111");
49.     System.out.println("Department Details");
50.     System.out.println("-----");
51.     System.out.println("Department Id : "+dept.getDid());
52.     System.out.println("Department Name : "+dept.getDname());
53.     Set<Employee> emps = dept.getEmps();
54.     System.out.println("EID\tENAME\tESAL\tEADDR");
55.     System.out.println("-----");
56.     for(Employee emp: emps) {
57.         System.out.print(emp.getId()+"\t");
58.         System.out.print(emp.getName()+"\t");
59.         System.out.print(emp.getSal()+"\t");
60.         System.out.print(emp.getAddress()+"\n");
61.     }
62.
63.     session.close();
64.     sessionFactory.close();
65. }
66.

```

3. Many-To-One Association:

It is a relation between entity classes, where multiple instances of an entity should be mapped with exactly single instance of another entity.

EX: Multiple Students have joined with single branch:

To provide Many-To-One Association in Hibernate applications then we have to use the following tag in mapping file.

<many-to-one name="--" class="--" cascade="all"/>

- ✓ where "name" attribute will take property name which is representing many to one relation.
- ✓ where "class" attribute will take contained class name.
- ✓ where "cascade" attribute will take the values like all, insert,.....

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

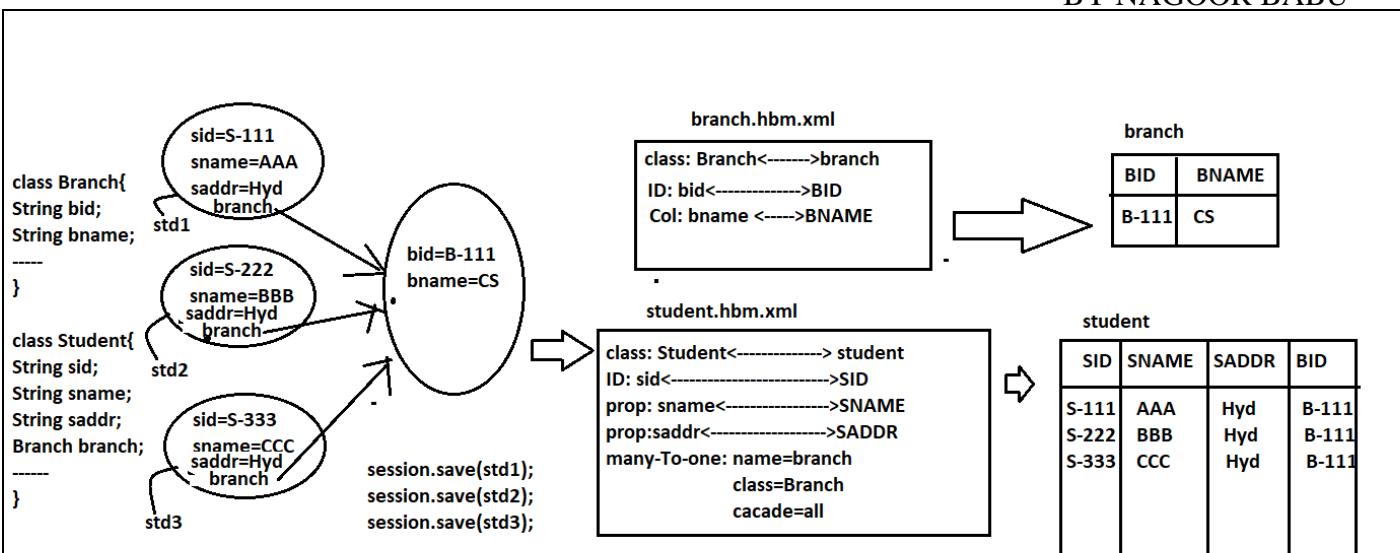
Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU



EX:

Branch.java

```

1. package com.durgasoft.pojo;
2.
3. public class Branch {
4.     private String bid;
5.     private String bname;
6.
7.     public String getBid() {
8.         return bid;
9.     }
10.    public void setBid(String bid) {
11.        this.bid = bid;
12.    }
13.    public String getBname() {
14.        return bname;
15.    }
16.    public void setBname(String bname) {
17.        this.bname = bname;
18.    }
19.
20. }

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Student.java

```
1. package com.durgasoft.pojo;
2.
3. public class Student {
4.     private String sid;
5.     private String sname;
6.     private String saddr;
7.     private Branch branch;
8.
9.     public String getSid() {
10.         return sid;
11.     }
12.     public void setSid(String sid) {
13.         this.sid = sid;
14.     }
15.     public String getSname() {
16.         return sname;
17.     }
18.     public void setSname(String sname) {
19.         this.sname = sname;
20.     }
21.     public String getSaddr() {
22.         return saddr;
23.     }
24.     public void setSaddr(String saddr) {
25.         this.saddr = saddr;
26.     }
27.     public Branch getBranch() {
28.         return branch;
29.     }
30.     public void setBranch(Branch branch) {
31.         this.branch = branch;
32.     }
33.
34.
35.}
```

**Branch.hbm.xml**

```
1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.      "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

4. "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Branch" table="branch">
7.     <id name="bid" column="BID" length="5"/>
8.     <property name="bname" column="BNAME" length="10"/>
9.   </class>
10. </hibernate-mapping>

```

Student.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Student" table="student">
7.     <id name="sid" column="SID" length="5"/>
8.     <property name="sname" column="SNAME" length="10"/>
9.     <property name="saddr" column="SADDR" length="10"/>
10.    <many-to-one name="branch" class="com.durgasoft.pojo.Branch" cascade="all"/>
11.  </class>
12. </hibernate-mapping>

```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10. import com.durgasoft.pojo.Branch;
11. import com.durgasoft.pojo.Student;
12.
13. public class Test {
14.
15.   public static void main(String[] args) throws Exception {
16.     Configuration config = new Configuration();
17.     config.configure();
18.     StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
19.     builder = builder.applySettings(config.getProperties());
20.     StandardServiceRegistry registry = builder.build();
21.     SessionFactory sessionFactory = config.buildSessionFactory(registry);
22.     Session session = sessionFactory.openSession();
23.
24.     Branch branch = new Branch();
25.     branch.setBid("B-111");
26.     branch.setBname("CS");
27.
28.     Student std1 = new Student();
29.     std1.setSid("S-111");
30.     std1.setSname("AAA");
31.     std1.setSaddr("Hyd");
32.     std1.setBranch(branch);
33.
34.     Student std2 = new Student();
35.     std2.setSid("S-222");
36.     std2.setSname("BBB");
37.     std2.setSaddr("Hyd");
38.     std2.setBranch(branch);
39.
40.     Student std3 = new Student();
41.     std3.setSid("S-333");
42.     std3.setSname("BBB");
43.     std3.setSaddr("Hyd");
44.     std3.setBranch(branch);
45.
46.     Transaction tx = session.beginTransaction();
47.     session.save(std1);
48.     session.save(std2);
49.     session.save(std3);
50.     tx.commit();
51.     System.out.println("Students are stored Successfully");
52.     session.close();
53.     sessionFactory.close();
54.
55.
56. }
57.
58.}
```

To represent many-to-one association, JPA has provided a separate annotation like

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

@ManyToOne(cascade=CascadeType.ALL)

Example:**Branch.java**

```
1. package com.durgasoft.pojo;
2.
3. import javax.persistence.Column;
4. import javax.persistence.Entity;
5. import javax.persistence.Id;
6. import javax.persistence.Table;
7.
8. @Entity
9. @Table(name="branch")
10. public class Branch {
11.     @Id
12.     @Column(name="BID", length=5)
13.     private String bid;
14.     @Column(name="BNAME", length=10)
15.     private String bname;
16.
17.     public String getBid() {
18.         return bid;
19.     }
20.     public void setBid(String bid) {
21.         this.bid = bid;
22.     }
23.     public String getBname() {
24.         return bname;
25.     }
26.     public void setBname(String bname) {
27.         this.bname = bname;
28.     }
29.
30.
31. }
```

Student.java

```
1. package com.durgasoft.pojo;
2.
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
3. import javax.persistence.CascadeType;
4. import javax.persistence.Column;
5. import javax.persistence.Entity;
6. import javax.persistence.Id;
7. import javax.persistence.ManyToOne;
8. import javax.persistence.Table;
9.
10. @Entity
11. @Table(name="student")
12. public class Student {
13.     @Id
14.     @Column(name="SID", length=5)
15.     private String sid;
16.     @Column(name="SNAME", length=10)
17.     private String sname;
18.     @Column(name="SADDR", length=10)
19.     private String saddr;
20.     @ManyToOne(cascade=CascadeType.ALL)
21.     private Branch branch;
22.
23.     public String getSid() {
24.         return sid;
25.     }
26.     public void setSid(String sid) {
27.         this.sid = sid;
28.     }
29.     public String getSname() {
30.         return sname;
31.     }
32.     public void setSname(String sname) {
33.         this.sname = sname;
34.     }
35.     public String getSaddr() {
36.         return saddr;
37.     }
38.     public void setSaddr(String saddr) {
39.         this.saddr = saddr;
40.     }
41.     public Branch getBranch() {
42.         return branch;
43.     }
44.     public void setBranch(Branch branch) {
45.         this.branch = branch;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

46. }
47.
48.
49.

```

hibernate.cfg.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.username">system</property>
10.    <property name="connection.password">durga</property>
11.    <!-- <property name="hibernate.hbm2ddl.auto">create</property>-->
12.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
13.    <property name="show_sql">true</property>
14.    <mapping class="com.durgasoft.pojo.Branch"/>
15.    <mapping class="com.durgasoft.pojo.Student"/>
16.  </session-factory>
17. </hibernate-configuration>

```

Test.java

```

1. package com.durgasoft.test;
2.
3. import org.hibernate.Session;
4. import org.hibernate.SessionFactory;
5. import org.hibernate.Transaction;
6. import org.hibernate.boot.registry.StandardServiceRegistry;
7. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
8. import org.hibernate.cfg.Configuration;
9.
10. import com.durgasoft.pojo.Branch;
11. import com.durgasoft.pojo.Student;
12.
13. public class Test {
14.
15.   public static void main(String[] args) throws Exception {

```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28 US NUM: 4433326786Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
16. Configuration config = new Configuration();
17. config.configure();
18. StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
19. builder = builder.applySettings(config.getProperties());
20. StandardServiceRegistry registry = builder.build();
21. SessionFactory sessionFactory = config.buildSessionFactory(registry);
22. Session session = sessionFactory.openSession();
23.
24. /*
25. Branch branch = new Branch();
26. branch.setBid("B-111");
27. branch.setBname("CS");
28.
29. Student std1 = new Student();
30. std1.setSid("S-111");
31. std1.setSname("AAA");
32. std1.setSaddr("Hyd");
33. std1.setBranch(branch);
34.
35. Student std2 = new Student();
36. std2.setSid("S-222");
37. std2.setSname("BBB");
38. std2.setSaddr("Hyd");
39. std2.setBranch(branch);
40.
41. Student std3 = new Student();
42. std3.setSid("S-333");
43. std3.setSname("BBB");
44. std3.setSaddr("Hyd");
45. std3.setBranch(branch);
46.
47. Transaction tx = session.beginTransaction();
48. session.save(std1);
49. session.save(std2);
50. session.save(std3);
51. tx.commit();
52. System.out.println("Students are stored Successfully");
53. */
54. Student std1 = (Student)session.get(Student.class, "S-111");
55. Branch branch = std1.getBranch();
56. System.out.println("Student Details");
57. System.out.println("-----");
58. System.out.println("Student Id : "+std1.getSid());
```

CONTACT US:

Mobile: +91- 8885 25 26 27



+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
59. System.out.println("Student Name :" +std1.getSname());
60. System.out.println("Student Address :" +std1.getSaddr());
61. System.out.println("Branch Id :" +branch.getBid());
62. System.out.println("Branch Name :" +branch.getBname());
63. System.out.println();
64.
65. Student std2 = (Student)session.get(Student.class, "S-222");
66. branch = std2.getBranch();
67. System.out.println("Student Details");
68. System.out.println("-----");
69. System.out.println("Student Id :" +std2.getId());
70. System.out.println("Student Name :" +std2.getSname());
71. System.out.println("Student Address :" +std2.getSaddr());
72. System.out.println("Branch Id :" +branch.getBid());
73. System.out.println("Branch Name :" +branch.getBname());
74. System.out.println();
75.
76. Student std3 = (Student)session.get(Student.class, "S-333");
77. branch = std3.getBranch();
78. System.out.println("Student Details");
79. System.out.println("-----");
80. System.out.println("Student Id :" +std3.getId());
81. System.out.println("Student Name :" +std3.getSname());
82. System.out.println("Student Address :" +std3.getSaddr());
83. System.out.println("Branch Id :" +branch.getBid());
84. System.out.println("Branch Name :" +branch.getBname());
85.
86. session.close();
87. sessionFactory.close();
88.
89.
90. }
91.
92.}
```

4. Many-To-Many Associations:

It is a relation between entities , where multiple instances of an entity must be mapped with multiple instances of another entity.

EX: Multiple Students have joined with Multiple Courses.

To represent Many-To-Many mapping we have to use the following tags in mapping file.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,

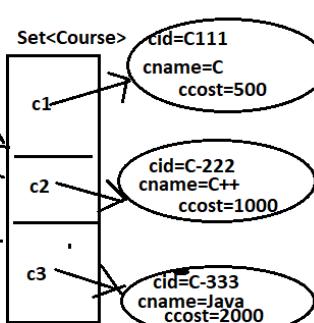


BY NAGOOR BABU

```
<set name="--" table="--" cascade="--">
<key column="--"/>
<many-tomany column="--" class="--"/>
</set>
```

- ✓ Where "<set>" tag is representing many side, where "name" attribute in <set> tag will take property name which is providing many side , where "table" attribute will take join_table name, where "cascade" will take "all", "insert", "update",.....
- ✓ Where "<key>" tag is able to take primary key of the container represented table, where "column" attribute will take that value.
- ✓ Where "<many-to-many>" tag will represent many-to-many association, where "column" attribute will take target table primary key column name, where "class" attribute will take target class name.

```
class Course{
String cid;
String cname;
int ccost;
}
class Student{
String sid;
String sname;
String saddr;
Set<Course> courses;
}
Session session = sessionFactory.openSession();
session.beginTransaction();
Student std1 = new Student();
std1.sid=S-111;
std1.sname=AAA;
std1.saddr=Hyd;
Set<Course> courses = new HashSet<Course>();
Course c1 = new Course();
c1.cid=C111;
c1.cname=C;
c1.ccost=500;
courses.add(c1);
Course c2 = new Course();
c2.cid=C-222;
c2.cname=C++;
c2.ccost=1000;
courses.add(c2);
Course c3 = new Course();
c3.cid=C-333;
c3.cname=Java;
c3.ccost=2000;
courses.add(c3);
std1.courses=courses;
session.save(std1);
}
```



Course.hbm.xml

```
Class: Course<----->course
ID: cid<----->CID
Prop: cname <----->CNAME
prop:ccost<----->CCOST
```

Student.hbm.xml

```
Class:Student<----->student
ID: sid<----->SID
prop: sname<----->SNAME
prop: SADDR<----->SADDR
set: name="courses" table:std_course
cascad=all
key: column="sid"
many-to-many:column=cid class:Course
```

course		
CID	CNAME	CCOST
C-1	C	500
C-2	C++	1000
C-3	Java	2000

Student_Course	
SID	CID
S-111	C-1
S-111	C-2
S-111	C3
S-22	C1
S-222	C2

student		
SID	SNAME	SADDR
S-111	AAA	Hyd
S-222	BBB	Hyd
S-333	CCC	Hyd

Example:

Course.java

```
1. package com.durgasoft.pojo;
2.
3. public class Course {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

4.  private String cid;
5.  private String cname;
6.  private int ccost;
7.
8.  public String getCid() {
9.      return cid;
10. }
11. public void setCid(String cid) {
12.     this.cid = cid;
13. }
14. public String getCname() {
15.     return cname;
16. }
17. public void setCname(String cname) {
18.     this.cname = cname;
19. }
20. public int getCcost() {
21.     return ccost;
22. }
23. public void setCcost(int ccost) {
24.     this.ccost = ccost;
25. }
26.
27.
28.
29.}
```

Student.java

```

1. package com.durgasoft.pojo;
2.
3. import java.util.Set;
4.
5. public class Student {
6.     private String sid;
7.     private String sname;
8.     private String saddr;
9.     private Set<Course> courses;
10.
11.    public String getSid() {
12.        return sid;
13.    }
14.    public void setSid(String sid) {
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

15.     this.sid = sid;
16. }
17. public String getSname() {
18.     return sname;
19. }
20. public void setSname(String sname) {
21.     this.sname = sname;
22. }
23. public String getSaddr() {
24.     return saddr;
25. }
26. public void setSaddr(String saddr) {
27.     this.saddr = saddr;
28. }
29. public Set<Course> getCourses() {
30.     return courses;
31. }
32. public void setCourses(Set<Course> courses) {
33.     this.courses = courses;
34. }
35.

```

Course.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Course" table="course">
7.     <id name="cid" column="CID" length="5"/>
8.     <property name="cname" column="CNAME" length="10"/>
9.     <property name="ccost" column="Ccost" length="5"/>
10.
11.   </class>
12. </hibernate-mapping>

```

Student.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-mapping PUBLIC
3.   "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

5. <hibernate-mapping>
6.   <class name="com.durgasoft.pojo.Student" table="student">
7.     <id name="sid" column="SID" length="5"/>
8.     <property name="sname" column="SNAME" length="10"/>
9.     <property name="saddr" column="SADDR" length="10"/>
10.
11.    <set name="courses" table="student_course" cascade="all">
12.      <key column="sid"/>
13.      <many-to-many column="cid" class="com.durgasoft.pojo.Course" />
14.    </set>
15.  </class>
16.</hibernate-mapping>
```

hibernate.hbm.xml

```

1. <?xml version="1.0" encoding="UTF-8"?>
2. <!DOCTYPE hibernate-configuration PUBLIC
3.   "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4.   "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
5. <hibernate-configuration>
6.   <session-factory>
7.     <property name="connection.driver_Class">oracle.jdbc.OracleDriver</property>
8.     <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
9.     <property name="connection.username">system</property>
10.    <property name="connection.password">durga</property>
11.    <property name="hibernate.hbm2ddl.auto">create</property>
12.    <property name="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</property>
13.  >
14.    <property name="show_sql">true</property>
15.    <mapping resource="Course.hbm.xml"/>
16.    <mapping resource="Student.hbm.xml"/>
17.  </session-factory>
18.</hibernate-configuration>
```

Test.java

```

1. package com.durgasoft.test;
2.
3. import java.util.HashSet;
4. import java.util.Set;
5.
6. import org.hibernate.Session;
7. import org.hibernate.SessionFactory;
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
8. import org.hibernate.Transaction;
9. import org.hibernate.boot.registry.StandardServiceRegistry;
10. import org.hibernate.boot.registry.StandardServiceRegistryBuilder;
11. import org.hibernate.cfg.Configuration;
12.
13. import com.durgasoft.pojo.Course;
14. import com.durgasoft.pojo.Student;
15.
16.
17. public class Test {
18.
19.     public static void main(String[] args) throws Exception {
20.         Configuration config = new Configuration();
21.         config.configure();
22.         StandardServiceRegistryBuilder builder = new StandardServiceRegistryBuilder();
23.         builder = builder.applySettings(config.getProperties());
24.         StandardServiceRegistry registry = builder.build();
25.         SessionFactory sessionFactory = config.buildSessionFactory(registry);
26.         Session session = sessionFactory.openSession();
27.
28.         Course c1 = new Course();
29.         c1.setCid("C-111");
30.         c1.setCname("C");
31.         c1.setCcost(500);
32.         Course c2 = new Course();
33.         c2.setCid("C-222");
34.         c2.setCname("C++");
35.         c2.setCcost(1000);
36.         Course c3 = new Course();
37.         c3.setCid("C-333");
38.         c3.setCname("Java");
39.         c3.setCcost(2000);
40.         Set<Course> courses = new HashSet<Course>();
41.         courses.add(c1);
42.         courses.add(c2);
43.         courses.add(c3);
44.
45.         Student std1 = new Student();
46.         std1.setSid("S-111");
47.         std1.setSname("AAA");
48.         std1.setSaddr("Hyd");
49.         std1.setCourses(courses);
50.         Student std2 = new Student();
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```

51.     std2.setSid("S-222");
52.     std2.setSname("BBB");
53.     std2.setSaddr("Hyd");
54.     std2.setCourses(courses);
55.     Student std3 = new Student();
56.     std3.setSid("S-333");
57.     std3.setSname("CCC");
58.     std3.setSaddr("Hyd");
59.     std3.setCourses(courses);
60.
61.     Transaction tx = session.beginTransaction();
62.     session.save(std1);
63.     session.save(std2);
64.     session.save(std3);
65.     tx.commit();
66.     System.out.println("Students Inserted Successfully");
67.
68.     session.close();
69.     sessionFactory.close();
70. }
71.

```

Retrieving the data through annotation:

To represent many _to_many association mapping, EJB3 has provide the following annotations.

```

@ManyToMany(cascade=" ")
@JoinTable(name="_",joinColumns="_",inverseJoinColumns="_")
@JoinColumn(name="_")

```

- ✓ Where, @ManyToMany annotation will represent many_to_many association mapping.
- ✓ Where, @JoinTable annotation will represent the meta data about the join table.
 - Name->join table name, joinColumns->array of @JoinColumn annotation;
 - InverseJoinColumn->array on @JoinColumn annotation to represent inverse join key configuration.
- ✓ Where, @JoinColumn annotation can be used to represent a column available in join table.
- ✓ Where @join column annotation must be user as nested annotation to @JoinTable annotation.

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

Hibernate.cfg.xml

```

1. <hibernate-configuration>
2.   ---
3.     <mapping class ="student"/>
4.     <mapping class="course"/>
5.   <session-factory>
6. </hibernate-configuration>
```



Student.java

```

1. Import java.util.*;
2. Import javax.persistence.*;
3. @entity
4. @table(name="student")
5. Public class student
6. {
7.   @id
8.   @column (name="sid");
9. Private string sid;
10. @column(name="sname")
11. Private string sname;
12. Private string sname;
13. @Many to Many(cascade=cascadetype.All)
14.
15.
16.
17. @JoinTable(name="student_course",joinColumns={@JoinColumn(name="sid")},inverseJoinColumns={@JoinColumn(name="cid")})
18. Private Collection<Course> courses;
19. setXxx(-);
20. getXxx( );
21. }
```

Course.java

```

1. Import javax.persistence.*;
2. @Entity
3. @Table(name="course")
4. Public class Course
5. {
6.   @Id
```

CONTACT US:

Mobile: +91- 8885 25 26 27

+91- 7207 21 24 27/28



US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.com

WEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
7. @Column(name="cid")
8. Private String cid;
9. @Column(name="cname")
10. Private String cname;
11. setXxx(-);
12. getXxx();
```

**ClientApp.java**

```
1. Import java.util.*;
2. Import org.hibernate.*;
3. Import org.hibernate.cfg.*;
4. Public class ClientApp
5. {
6. Public static void main(String
7. {
8. Configuration cfg=new AnnotationConfiguration( );
9. cfg.configure();
10. SessionFactory sf=cfg.buildSessionFactory();
11. Session s=sf.openSession( );
12. Student s1=(Student)s.get("Student","s1");
13. Sop1("StudentDetails");
14. Sop1(".....");
15. Sop1 ("student id"+s1.getId());
16. Sop1 ("Student name".+s1.getName());
17. Collection c=s1.getCourses();
18. Iterator it=c.iterator();
19. Sop1 ("course details");
20. Sop1 (".....");
21. Sop1 ("CID NAME");
22. Sop1 (".....");
23. While
24. {
25. Course crs=(Course)it.next();
26. Sop1 (crs.getId()+" "+crs.getName());
27. }
28. Sop1 (".....");
29. Student s2=(Student)s.get("Student","s2");
30. Sop1 ("Student Details");
31. Sop1 (".....");
32. Sop1("student id.."+s2.getId());
```

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,



BY NAGOOR BABU

```
33. C=s2.getCourses();
34. It=c.iterate();
35. Sopl("Course Details");
36. Sopl(".....");
37. Sopl("CID CNAME");
38. While(it.hasNext())
39. {
40. Course crs =(Course)it.next( );
41. Sopl(crs.getCid()+" "+crs.getCname());
42. s.close();
43. }
44. }
```

DURGASOUL

CONTACT US:

Mobile: +91- 8885 25 26 27

 +91- 7207 21 24 27/28

US NUM: 4433326786

Mail ID: durgasoftonlinetraining@gmail.comWEBSITE: www.durgasoftonline.com

FLAT NO: 202, HMDA MYTRIVANUM, AMEERPET, HYDERABAD.,