This PDF Created by
JPG To PDF Converter for Mac
(Unregistered Version)

User Interface Layer:-

User Interface Layer is a topmost layer in enterprise application.

2. User Interface Layer is a starting point for the users to interact with the enterprise application.

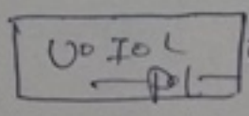
3. The main purpose of user interface layer is to improve look and feel for the enterprise applications. (ii) To accept user details in order to execute enterprise applications.

(iii) To specify different types of requests like post, head, put and so on -- from client.

(iv) To execute client side data validations with java script functions.

— To prepare User Interface layer an enterprise applications, we have to use a separate logic called as presentation logic.

— To prepare presentation logic in an enterprise application, we will use the technologies like awt, swing, html, JSP, free marker, velocity etc.



awt, swing,

Business Processing Layer:- Business processing layer is the heart of the enterprise application development. Business processing layer will provide very good environment to define and execute business rules and regulations required by the client. This layer will improve accuracy of the enterprise application.

— To prepare business processing layer, an enterprise application development, we will use a separate logic called as business logic.

JPG to PDF Converter For Mac - Unregis

To prepare business logic in enterprise applications, we will use the technologies like servlets, JSPs, Java beans, EJB's and so on--

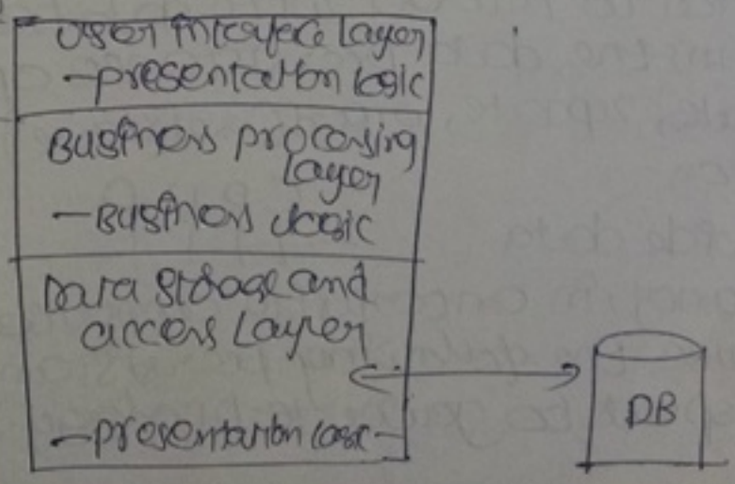
Data storage and access layer:

1. This layer is the bottom most layer in enterprise applications.

The main purpose of this layer is to interact with the database in order to perform the basic database operations.

- The main intention of data storage and access layer is to achieve data persistancy. A very good data storage and access layer will improve the overall performance of the enterprise application.
- To prepare data storage and access layer in enterprise application development, we have to use a separate logic called as persistance logic
- To prepare persistance logic in enterprise applications, we will use the technologies like JDBC, EJB's (Entity Beans), ORM (Object Relational Mapping) tools like hibernate, JPA [Java persistance API], open JPA, Toplink

Enterprise application Architecture



JPG to PDF Converter For Mac - Unregis

- In general in enterprise application execution users may provide the data at user interface layer, where user interface layer will perform client side data validations as per the application development then it will submit user provided data to the business processing layer, it may be available at server machine.

- At business processing layer, server side data validations will be executed and process the data as per the business rules and regulations which were define and finally submit the finalized data to data storage and access layer.

At data storage and access layer, the persisting data will be stored onto the database as per the persistance logic which we provide it.

Data persistancy:-

Managing the data permanancely in database is called as Data persistancy.
[observe it is not permant storage of the data into database]

In enterprise application development, it is essential to interact with database in order to perform the data persistance operations like create, update, read (retrieve), Delete and so on--

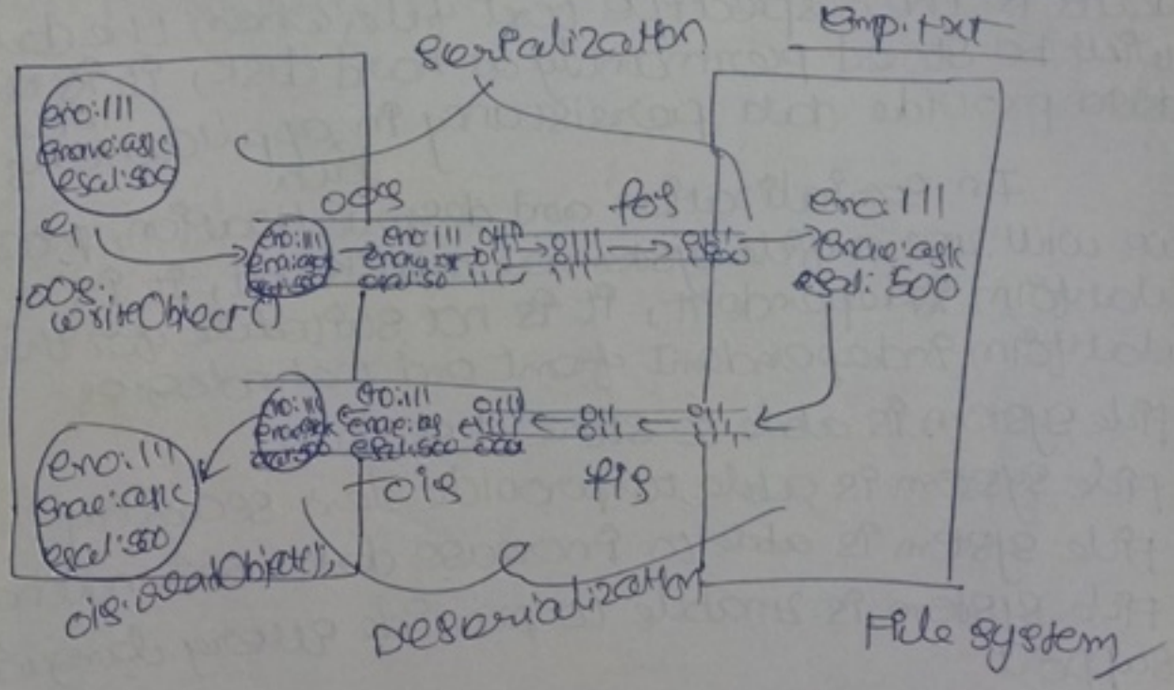
C R U D

- To provide data persistancy, in enterprise applications, we have to use the following persistance solutions with respect to Java technology.

JPG to PDF Converter For Mac - Unregis

1) Serialization - Deserialization
2) JDBC [JDBC-DAO
3) ORM [Object Relational Mapping]
   1) EJB — Entity Beans
   2) Hybernate
   3) JPA
   4) open JPA
   5) Toplink

Data persistancy through serialization and Deserialization:-



D

JPG to PDF Converter For Mac - Unregis

In general, serialization and deserialization data persistency solutions is highly recommended for standalone applications, not suggestible for enterprise applications

Date: 21st mar, 2012

- As part of serialization and deserialization of data persistancy alternative we will used text file to store serialized data, here text file is a file system provided by local operating system as a permanant storage area.
- In serialization process, if we store serialized data in the respective text file, when the data will be stored permantly at hard disk, it ~~is a~~ will ~~data~~ provide data persistancy in java applications.

In serialization and deserialization, process we will use a file system as a back end, it is a platform independent, it is not suitable for the platform independent front end technologies.

- File system is able to store less data
- File system is able to provide less security.
- File system is able to increase data redundancy.
- File system is unable to provide query language support
- In java applications serialization and deserialization data persistancy solutions is able to provide only the persistance operations like create (store/insert), retrieve (read). It unable to provide the data base operations like update and delete

JPG to PDF Converter For Mac - Unregis

serialization and deserialization are too time consuming processes. To achieve data persistancy, it will reduce the performance of the applications.

— In case of serialization, data persistancy alternative to perform read operation, a separate process we have to start that is deserialization. In this approach, byte oriented streams to carry the data, always byte oriented ~~system~~ streams are not suggestible. we will use

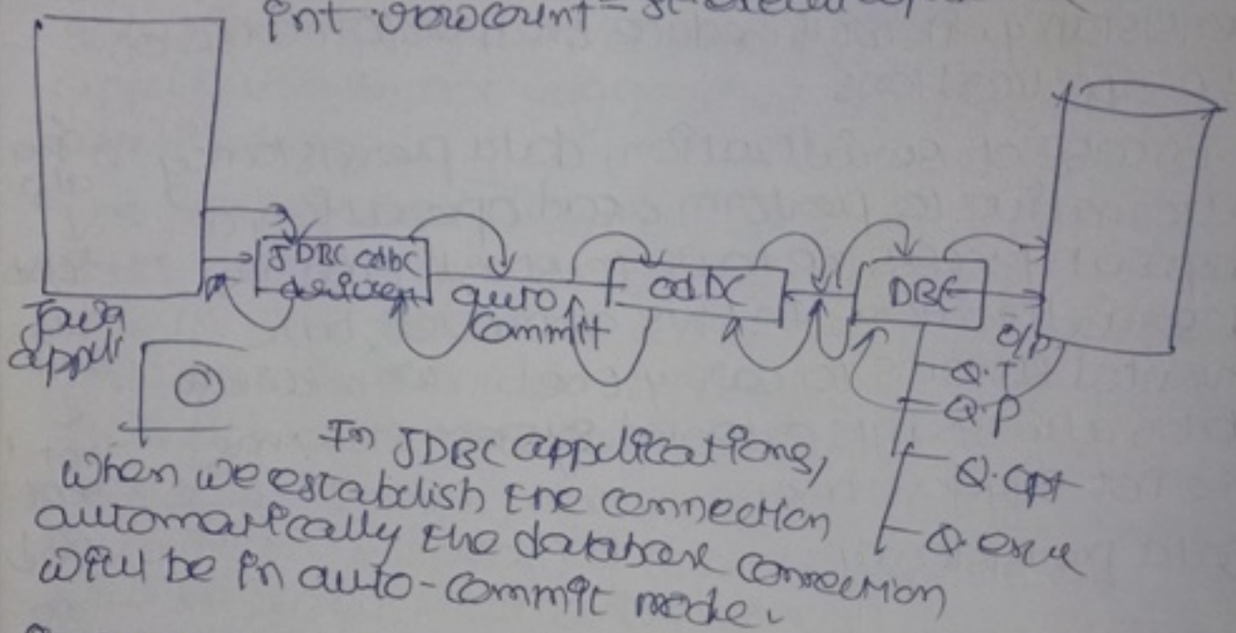
Data persistancy

fine alp
↓
good
data base
↓
not good
Dont
we dont have any right to do delete data in file

JPG to PDF Converter For Mac - Unregis

Data persistancy through JDBC :-

int rowcount = st.executeupdate(



In JDBC applications, when we establish the connection automatically the database connection will be in auto-commit mode.

In case of auto commit modes if we submit any SQL query to JDBC connection then it will carry that SQL query to Database engine, JDBC connection will make the database engine to execute SQL query and to store the results on to the database permanantly. Here representing the data in a database permanantly is called as data persistancy.

— In enterprise applications JDBC Data persistance solutions will provide hard coded approach to achieve data persistancy. (what?)

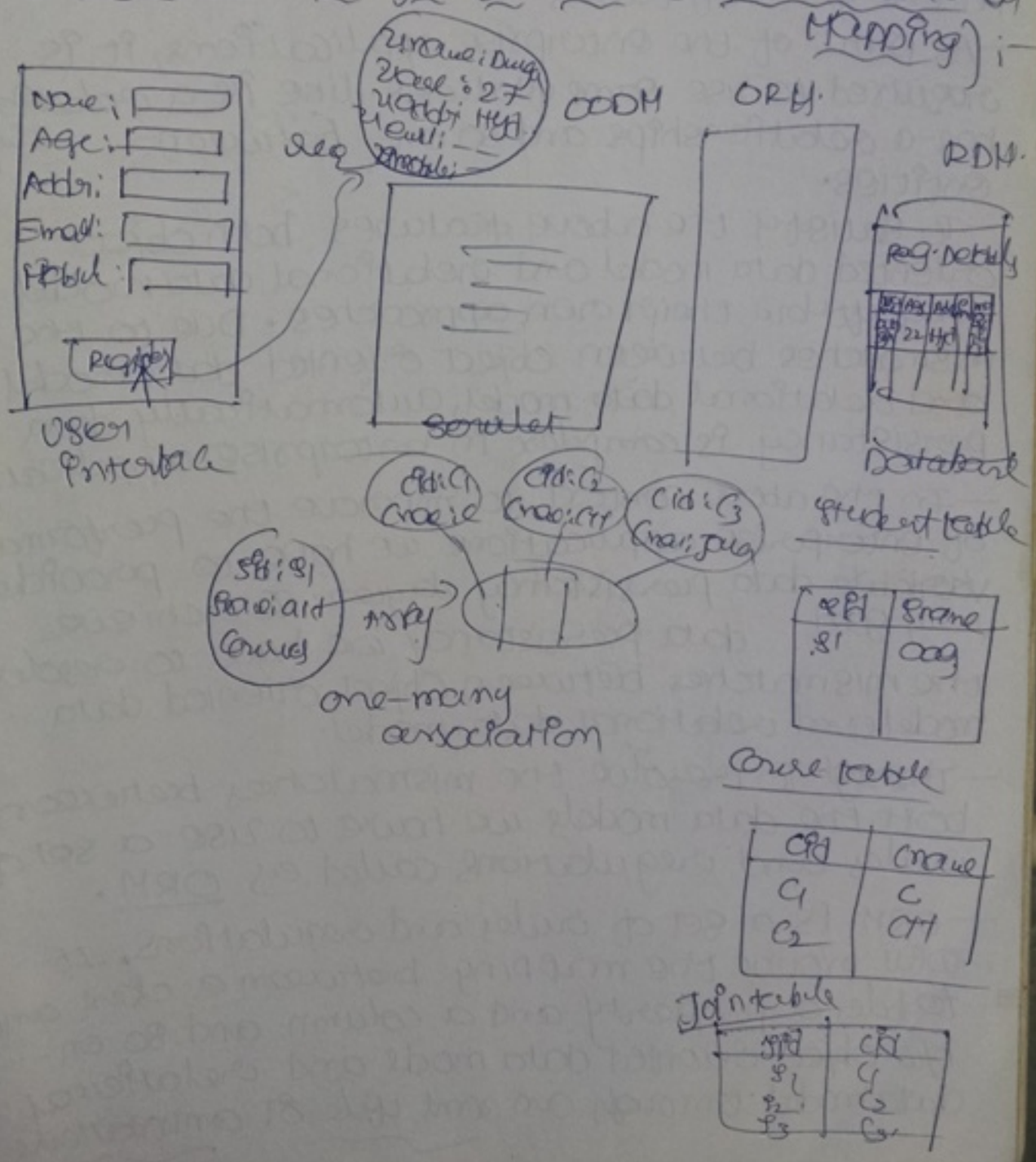
— If we use JDBC as a data persistancy alternative in enterprise applications we should require explicit monitoring on driver management, connection management, statement management.

— JDBC data persistancy solution should require explicit knowledge on SQL queries. [SQL queries are database dependent and it will make JDBC as a ~~pt~~ database dependent data persistancy solutions.

JPG to PDF Converter For Mac - Unregis

In JDBC data persistancy alternative we will fetch the data from database through result set object, which is not serializable by default. It is difficult to move result set object from one machine to another machine over the network in enterprise applications.

Requirement to go for ORM: (Object Relational Mapping) :-



JPG to PDF Converter For Mac - Unregis

Date: 29th mar, 2012

In any enterprise applications, it is essential to use object oriented data model and relational data model.

where object oriented data model, we will represents the data in the form of objects. But relational model will represent the data in the form of tables.

— As part of the enterprise applications, it is required to use some features like is-a relation ships, has-a relationships and so on -- between entities.

— To satisfy the above features both object oriented data model and relational data model they follow their own approaches. Due to the mismatches between object oriented data model and relational data model, automatically data persistancy is complex in enterprise applications.

— In the above context to improve the performance of enterprise applications we have to provide variable data persistancy layer. To achieve very good data persistancy we have to resolve the mismatches between object oriented data model and relational data model.

— ~~Through~~ To resolve the mismatches between both the data models we have to use a set of rules and regulations called as ORM.

— ORM is a set of rules and regulations. It will provide the mapping between a class and table, a property and a column and so on -- b/w object oriented data model and relational data model through an xml file or annotations

JPG to PDF Converter For Mac - Unregis

```
class Employee
{
  string eid;
  string ename;
  string esal;
  Address addr
}
class Address
{string pno;
 string street;
 string city;
}
```

ORM

class
Employee ← emp
Id:
eid ↔ eid (pk)
property
ename ↔ ename
esal ↔ esal
component
addr → Address
street ↔ street
city ↔ city

emp

eid	ename	esal	pno	street	city
e1	aaa	5000	1,	xyz	[illegible]

save(e)

e → eid: e1, ename: aaa, esal: 5000, addr

pno: 1, street: xyz, city: [illegible]

Granularity mismatch diagram

In general, in our enterprise applications the following mismatches between object oriented data model and relational data model

1. Granularity mismatch
2. Inheritance mismatch (or) subtypes mismatch
3. Associations mismatch
4. Cost Mismatch
5. Identity mismatch.

JPG to PDF Converter For Mac - Unregis

Granularity mismatch:-

The number of properties declared in parent entity class or the number of columns that provided in a particular database table is called as granularity.

- In enterprise applications to represent employee entity in object oriented data model we will declare a class with the name employee and with the properties empid, empname and esal. and so on.
- In relational model to represent the same employee entity, we have to provide a table with the name emp and the respective columns eid, ename, esal.
- In the above context, if we want to add employee address details then we have to declare a separate class with address as a name and with the properties pdt no, street, and city and we have to provide the reference of address class in employee
- Similarly if we want to add address details to emp table in relational model, we have to provide the columns pno, street, city in the same emp table. or we may declare user defined data type, which may not be supported by all the databases.

In the above context, the granularity of employee class and the granularity of emp table are different. It is granularity mismatch b/w object oriented model and relational model.

JPG to PDF Converter For Mac - Unregis

To resolve the above granularity mismatch ORM has provided an alternative in the form of componant mapping.

Componant property: is a property in an entity class. It will represent multiple no of columns available in database table.

- Here the diagram will be there

JPG to PDF Converter For Mac - Unregis

Inheritance Mismatch or subtypes mismatch:

```
class Employee
{ string eid;
  string ename;
}
```

eid: e1
ename: aaa
email: ae--
mobile: 91999

mgr.

ACCOUNT eid = e2
ename: bbb
qual: mba
desi: ACCOUNT

extends

extends

```
class Manager → Employee
{
  string email;
  string mobile;
}
```

```
class Accountant → Employee
{ string qual;
  string desi;
}
```

qual → qualification

desi → designation

ORM

Inheritance Mapping
① Table per class hierarchy
② Table per sub class
③ Table per concrete class

1st approach

① emp

eid	ename	email	mobile	qual	desi
e1	aaa	ae	91-9	—	—
e2	bbb	—	—	mba	ACCOUNT

2nd approach

HMM

eid	ename	email	mobile
e1	aaa	ae	91-9

Accountant

eid	ename	qual	mobile
e2	bbb	MBA	ACCOUNTANT

3rd approach

③ emp

emp id	ename
e1	aaa
e2	bbb

pk = fk

pk = fk

HMM

eid	ename	mobile
e1	aaa	91-

Accountant

eid	qual	desi
e2	mba	ACCOUNT

There are two ways only that can take this entry con

① All columns

② Duplicate data

③

JPG to PDF Converter For Mac - Unregis

As part of the enterprise applications it is required to use both object oriented data model and relational data model. In object oriented model, we will provide two types of relationships b/w entity classes

① is-a relationship
② has-a relationship.

where is-a relationship will describe inheritance relation between two entity classes inorder to improve code reusability.

Incase of relational data model we are able to achieve the above relationships between two entity tables by defining primary key and foreign key relationships and some other manipulations.

— In enterprise applications if we provide inheri relation between entity classes then we will create objects for the sub classes and we will store sub class objects into relational data model. when we are trying to save sub class objects into the relational model, we are not having any standardized approach to distribute the data over the respective tables, it will make the data persistancy difficult.

In the above context to simplify data persistancy we have to resolve sub types mismatch b/w object oriented model and relational model, for this we have to use ORM provided inheritance mapping, in b/w object oriented model and relational model.

JPG to PDF Converter For Mac - Unregis

- To solve sub types mismatch ORM has provided the following 3 types of inheritance mapings.
  1. Table for class hierarchy
  2. Table for sub classes
  3. Table for concrete classes.

Associations mismatch:-

Association mapping: In object oriented model, we will use has-a relationship b/w entity classes. has-a relationship will describe associations b/w the entity classes to improve communication b/w objects, in order to provide data navigation.

In object oriented data model, we are able to achieve associations either by providing a single reference or by providing collection of references of another entity class in the present entity class.

- In relational data model we are able to achieve associations b/w entity tables either by using primary key, foreign key relationships or by using joining tables.
- In the above context, in enterprise applications when we save a base object automatically the base object data and its associated objects data has to be stored onto the base table and the corresponding associated tables in relational data model.

To achieve the above requirement, no standardized approach is existed, it will make data persistancy difficult in enterprise applications.

JPG to PDF Converter For Mac - Unregis

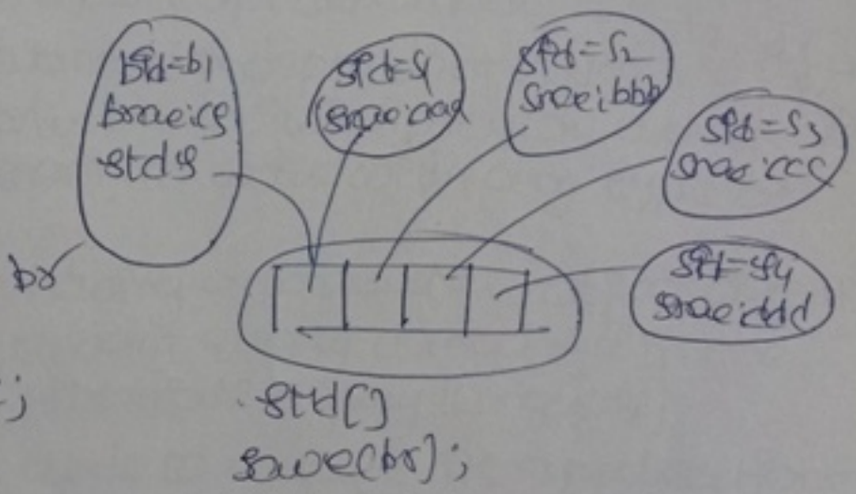
In the above context to simplify data persistency we have to use ORM (rules and regulations).

— ORM has provided four types of association mappings like

1. one-to one
2. one to many
3. many to one
4. many to many

} to resolve associations mismatch b/w object oriented model and relational model.

```
class Branch
  { string bId;
    string bname;
  student[] stds;
  }
class student
  { string sId;
    string sname;
    ==
    ==
  }
```



Association mapping
1-1
1-M
M-1
M-M

① branch

~~bId~~	~~bname~~
b	cs

pk fk

student

sId	sname	bId
s1	aaa	b1
s2	bbb	b1
s3	ccc	b1

② branch

bId	bname
b1	cs

branch student

bId	sId
b1	s1
b2	s2
b3	s3

student

sId	sname
s1	aaa
s2	bbb
s3	ccc

JPG to PDF Converter For Mac - Unregis

Hibernate History

Hibernate
- Home: soft tree
- Author: Gavin King
- First version: Hybernate 1.0 (2003) — Introduced in
- Used version: Hbn 3.2V (3.5)
- Latest version: Hbn 4.0
- Strong features

① Hibernate will support pojo or poji kind of implementations, it will show up hibernate is light weight and less api dependent.

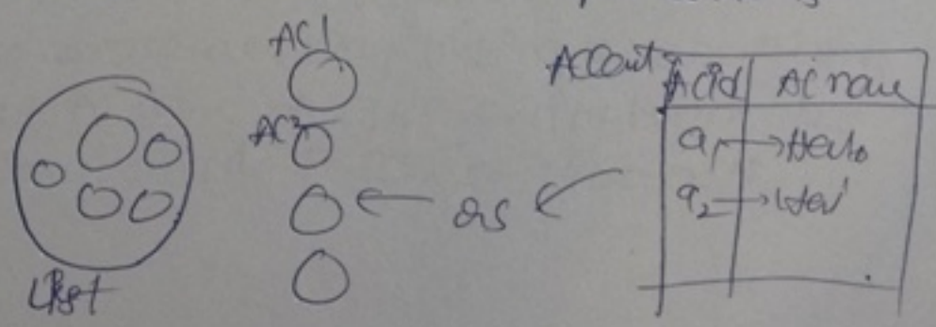
② Hibernate is database independent data persistancy solution. It will provide a very good environment to work with any type of database.

What is pojo?

③ Hibernate is having a powerful, object oriented, database independent query language in the form of H.Q.L. [Hibernate Query Language].

④ Hibernate is able to execute its applications with any type of java applications like standalone applications, enterprise applications and so on.

⑤ Hibernate is having a very good collection support to perform persistance operations.



JPG to PDF Converter For Mac - Unregis

- In general hybernate will be supported by allmost all the IDE's and servers.
- Hibernate is having very good cache mechanism to reuse objects.

Select ~~dyn~~

both yes

one are the wrt

- Hibernate is providing a very good annotation support to reduce xml dependency.
- Hibernate has provided very good associations mapping like one-one, one to many, many to one, many to many in unidirectional and in bidirec
- Hibernate has implemented number of (long) primary key generation algorithms.

JPG to PDF Converter For Mac - Unregis

Similarly, we are able to get the mismatches like identity mismatch, ~~and~~ cost mismatch, data navigation mismatch and so on b/w object oriented data model and relational data model, which may provide difficulty to achieve data persistancy, where to simplify data persistancy we have to use ORM implemented tools.

In enterprise applications we will use the following list of ORM implemented tools to provide data persistancy. [Missing notes written in another scp notes]

1. EJB- Entity Beans
2. Hibernate
3. JPA
4. Open JPA
5. Toplink

Q: What are the differences b/w EJB- entity beans and Hibernate.

A) EJB- entity beans is more API dependent, in EJB's bean classes must implement or extend some predefined library. But in case of Hibernate almost all bean classes are POJO classes, which are not dependent on pre-defined library, which should not extend or implement predefined library. Therefore Hibernate is less API dependent.

— Due to the above reason, testing & debugging are difficult in case of EJB's-entity beans but which are very simple in case of Hibernate.

JPG to PDF Converter For Mac - Unregis

- In EJB's applications we are unable to execute entity beans without using application server. But it is possible to execute Hibernate applications with or without the ~~specification~~ application server.
- Due to the above reasons EJB's entity beans are less portable. But hibernate is more portable in enterprise applications.
- To execute EJB's entity beans we must require application server so that entity beans are heavy weight data persistency solution. To execute Hibernate applications it is not at all mandatory to use application server so that hibernate is light weight data persistency solution.
- In case of EJB's entity beans it is convention to provide one-to-one association b/w an entity class is able to represent multiple no. of database tables, a single database table to represent multiple no. of entity classes.
- In case of EJB's entity beans all the bean classes will not

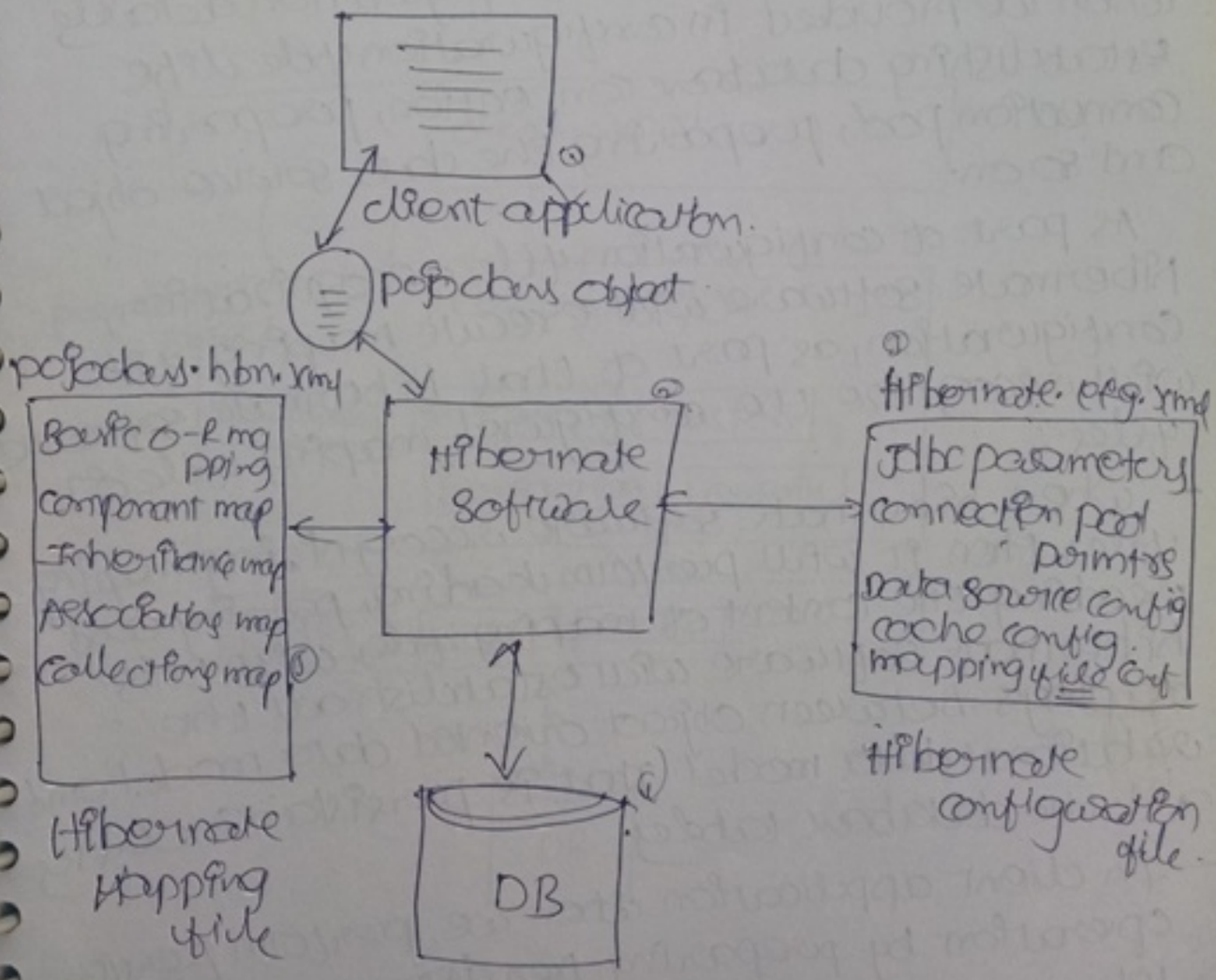
JPG to PDF Converter For Mac - Unregis

Date: 28th mar, 2012

Hibernate Architecture:

To design any hibernate application, we require the following components

1. persistance class
2. Hibernate configuration file
3. Hibernate mapping file.
4. client application.



JPG to PDF Converter For Mac - Unregis

- when we execute client application, client application will activate hybernate software, as part of activating hibernate software, it will recognize hibernate configuration file then perform configuration file loading, parsing and reading the content of configuration file.
- At the time of recognizing configuration file, hybernate will setup the complete hibernate environment as per the configuration details which we provided in configuration file like establishing data base connection, preparing connection pool, preparing the data source object and so on.

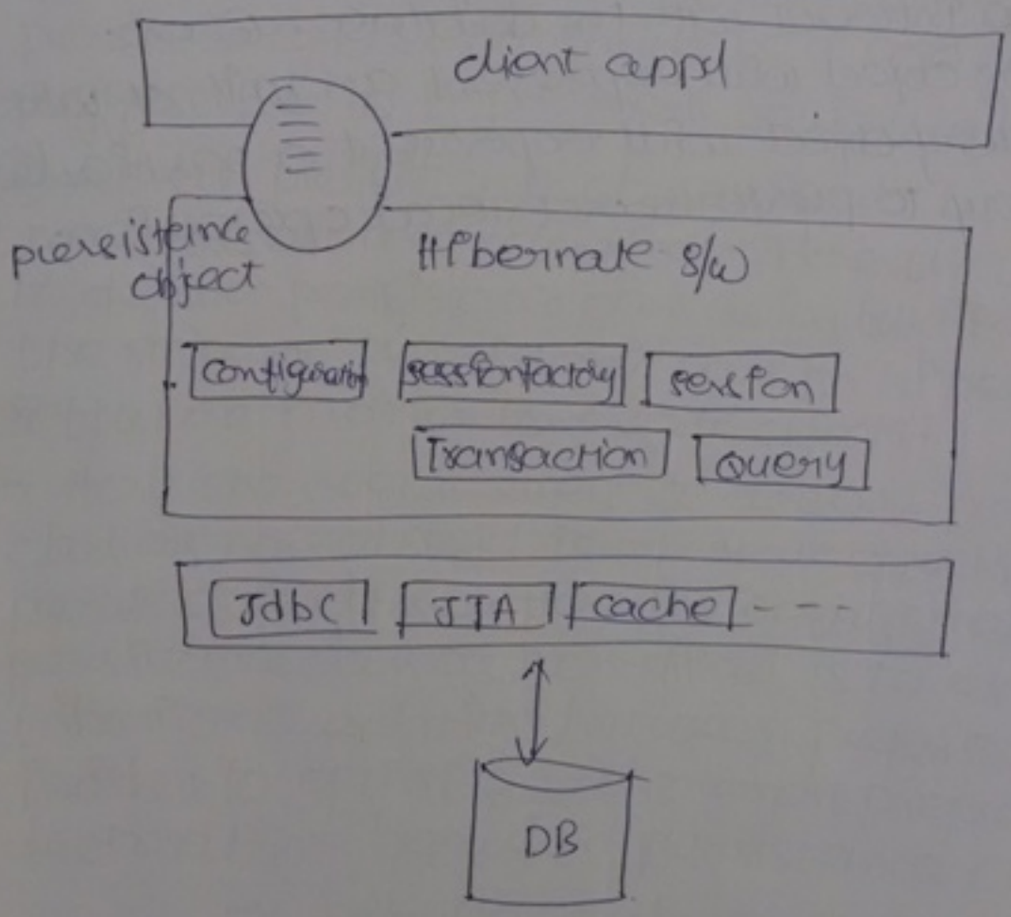
As part of configuration file recognization, hibernate software will execute maping file configuration, as part of that hibernate software will recognize the configured maping file of files.

- when hibernate software recognizing maping files then it will perform loading, parsing and reading the content of maping file, with this hibernate software will establish all the mapings between object oriented data model and relational data model. That is persistance classes and the database tables.
- In client application when we perform persistance operation by preparing persistance object, hibernate software will pickup persistance object, it will get persistable data, prepare the underlying database dependent sql queries,

JPG to PDF Converter For Mac - Unregis

prepare the equivalent JDBC code and execute as per the persistence operation.

- with the above process hibernate software will perform the required database operation by getting the mapping details from mapping file.



JPG to PDF Converter For Mac - Unregis

- when we perform a persistance operations with persistance objects then hibernate software will perform same persistance operation by preparing the sequence of objects, configuration, sessionfactory, session, Transaction and Query.
- Here configuration object will encapsulate all the configuration details, where sessionfactory object will manage all the connection objects, statement objects, preparedstatement objects and so on. where session object will represent a particular user request to interact with the database. where transaction object will represent an unit of work. where query object will represent a particular HQL query to perform retrieval operation.

e.g.

JPG to PDF Converter For Mac - Unregis

persistance class: The main purpose of persistance class is to represent persistable data in the form of persistance objects. persistance class is a pojo class, which should not extend and implement any predefined library except Serializable + interface.

- To prepare persistance classes in hibernate applications, we have to use the following guidelines.

① we have to prepare an user defined class as persistance class, it should be a pojo class, it is suggestible to implement Serializable interface.

② All the persistance classes should be public, non-abstract and non-final. Here the requirement to declare persistance class as public is to bring the scope of persistance class to hibernate software in order to create objects.

- Here the requirement to declare persistance class as non-abstract is to create objects for persistance classes. Here the requirement to declare persistance class as non-final is to establish inheritance relation between persistance classes in order to optimize the common properties declarations between persistance classes.

In persistance classes it is suggestible to declare all the properties as private properties, among all the properties, one property should be declared as Id property, to represent primary key column in the respective database table.

JPG to PDF Converter For Mac - Unregis

In persistance classes, it is suggestible to declare all the behaviours as public.

— In persistance classes if we want to provide any construction then it must be public and zero arguement.

— In persistance classes if we want to provide our own customization to compare persistance objects then it is suggestible to override equals() method.

— If we want to provide our own customization while generating hash codes for the persistance objects then it is suggestible to override hashcode() method.

—

JPG to PDF Converter For Mac - Unregis

ex:

Date: 2nd Apr, 2012

```
class User implements
                    java.io.Serializable
{ private String uname;
  private String upwd;
  public void setUname(String uname)
        { this.uname = uname;
        }
  public void setupwd(String upwd)
        { this.upwd = upwd;
        }
  public String getuname()
        { return uname;
        }
  public String getupwd()
        { return upwd;
        }
}
```

= Hibernate Mapping file:- The main purpose of the mapping file in hybernate applications is to provide the mapping between object oriented constructs like classes, properties and the relation model constructs like tables, columns and so on.

In general, in hibernate applications mapping files will include the metadata about basic ORM mapping, inheritance mapping, associations mapping, components mapping, collection mappings and so on.

In hibernate applications mapping file name is not fixed, it is possible to change mapping filename, but hibernate has provided a separate convention to provide mapping filenames.

JPG to PDF Converter For Mac - Unregis

pojoClassName.hbm.mapping.mapping

— In hibernate applications, in general we only provide a separate mapping file to provide the mapping b/w a particular persistance class and the respective database table.

— In hibernate applications it is possible to provide more than one mapping file and it is possible to provide more than one persistance class declaration within a single mapping file.

— In hibernate applications mapping file is optional because hibernate 3.2 version has provided annotations as an alternative for mapping file.

— To provide the mapping between a persistance class and the respective database table, we have to use the following xml tags.

```
<!DOCTYPE- - - -
  - - - - - - -/>
<hibernate-mapping>
    <class name="__" table="__">
      <id name="__" column="__" type="__" size="__"/>
    <property name="__" column="__" type="__"
                                    size="__"/>
      ==
    </class>
    ==
  </hibernate-mapping>
```

— where <hibernate-mapping> is the root tag to provide persistance class configuration, where <class> tag can be used to provide the mapping b/w the persistance class and the respective database table.

JPG to PDF Converter For Mac - Unregis

- where name attribute in class > tag will take fully qualified name of persistance class where table attribute will take respective db table name.
- where <id> tag can be used to provide the mapping between id property from persistance class and primary key column from the respective database table.
- where property tag can be used to provide mapping between normal property from persistance class and the respective column from database table.
- where name attribute will take property name from persistance class. where column attribute will take the respective database table column.
- where column attribute is optional when we use the same name to the respective property in pojo class and the respective column in database table.
- where type attribute will take java type to insert values. where size attribute will take the respective database table column size.

Ex:

```
<!DOCTYPE---
  ------/>
<hibernate-mapping>
    <class name="User" table="login_users">
     <id name="uname" column="uname"/>
    <property name="upass"/>
    </class>
    </hibernate-mapping>
```

JPG to PDF Converter For Mac - Unregis

hibernate configuration file :- The main purpose of the configuration file in hibernate applications is to provide all the configuration details required by hibernate software to perform all the persistance operations.

- example: In hibernate application when we access persistance methods automatically hibernate software will prepare the equivalent JDBC code and execute. To achieve this requirement hibernate software has to setup JDBC environment like driver registration, connection establishment and so on.

- To setup the JDBC environment hibernate software should require the configuration details like driver class name, driver url, database user name and password, database respective dialect configuration and so on.

- In the above context hibernate should expect all the above required configuration details from the configuration file that we have to provide.

- In general in hibernate applications configuration file may include the meta data about connection configuration details like driver class name, driver url, database user name and password and so on, connection pooling configuration, data source configurations, transaction configurations, cache mechanisms configurations and so on.

JPG to PDF Converter For Mac - Unregis

The default name of the configuration file in Hibernate application is

hibernate.cfg.xml

- In hibernate applications, it is possible to change hibernate configuration file name but we have to provide that information to hibern software. To design and execute basic hibernate applications we have to provide minimum the configuration details JDBC param dialect configuration, mapping files etc, configurations

- To prepare configuration file in hibernate application we have to use the following xml tags in ~~hibernate~~ configuration file

```
<!DOCTYPE---/>
<hibernate-configuration>
    <session-factory>
        <property name="__">value</property>
        =
        <mapping resource="__"/>
        =
    </session-factory>
</hibernate-configuration>
```

JPG to PDF Converter For Mac - Unregis

Date: 3rd Apl 2012

where hibernate configuration> is a root tag in configuration file. where session factory tag will include all the configuration details which are required for hibernate software.

where property tag will represent a particular property in the configuration file. where name attribute in property tag can be used to specify property name. The property value, we have to provide as value to property tag.

— To provide all the connection related properties we have to use the following convention.

connection.xxx (or) hibernate.connection.xxx

Here xxx may be driver class, URL, user, password and so on...

— To provide ~~xxx~~ dialective configuration we have to use the following property name hibernate.dialect.

The main purpose of dialect configuration is to give an information to the hibernate software the type of database we are going to use and the type of database respective configuration details required in hibernate software.

— hibernate is database independent, from database to database SQL queries will be varied. In hibernate applications when we access persistance methods hibernate software has to prepare the database respective SQL queries. In this context hibernate s/w should require which database respective SQL queries and configurations has to be prepared from the configuration file.

JPG to PDF Converter For Mac - Unregis

To achieve this requirement only we have to provide dialect configuration in configuration file.

- To represent all the databases hibernate f/w has provided set of predefined classes in the form of org.hibernate.dialect package in hibernate 3.jar
- where mapping tag can be used to represent file a particular hibernate mapping file configuration.
- where resource attribute can be used to specify hibernate mapping file.

hibernate.cfg.xml

```
<hibernate-configuration>
        <session-factory>
  <property name="connection.driver_class"
            oracle.jdbc.driver.OracleDriver
                                  </property>
<property name="connection.url">jdbc:oracle:
      thin:@localhost:1521:xe </property>
<property name="connection.user">system </property>
<property name="connection.password">durga </property>
<property name="hibernate.dialect">
        org.hibernate.dialect.OracleDialect
                                  </property>
<mapping resource="user.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

JPG to PDF Converter For Mac - Unregis

To represent below versions of oracle9 database hibernate has provided a separate dialect class

org.hibernate.dialect.Oracle9Dialect ✓

— For oracle9 and above versions, hibernate has provided a separate dialect class that is

org.hibernate.dialect.Oracle9Dialect ✓

Int. Ques:

Is it possible to provide more than one configuration file within a single hibernate application?

Ans In a single hibernate application it is possible to provide more than one configuration file.

— In general a single configuration file will provide a particular database configuration details.

— In our enterprise applications if we have any requirement to interact with multiple no. of databases then to represent multiple no. of databases we have to provide more than one configuration file ✓

Note: In hibernate applications, it is not mandatory to provide configuration file. As an alternative to configuration file we will use programmatic approach (using predefined methods), to provide all the configuration details to hibernate software

JPG to PDF Converter For Mac - Unregis

client application:

- In general hibernate will execute its applications with or without the application servers. That is we are able to utilize hibernate in any of the Java/J2ee applications, that is with standalone applications and enterprise applications.
- In enterprise applications or in standalone application the client application may be a CUI application (main method), a GUI application (action performed method), a web application like servlets (doGet() or doPost() methods) and JSPs (scriptlet) and so on--
- To prepare client applications we have to use the following steps
  1. prepare configuration object
  2. prepare session factory object.
  3. prepare session object.
  4. prepare transaction object
  5. perform persistance operations
  6. close session, session factory objects

Step①

prepare Configuration object:

① Configuration is an object in the hibernate application. It can be used to manage all the configuration details which are provided in hibernate configuration file.

JPG to PDF Converter For Mac - Unregis

~~Step ① Prepare~~ Date: 4th Apr, 2012

② To represent configuration object hibernate s/w as provided a predefined class i.e

org.hibernate.cfg.Configuration in hibernate3.jar file

Ex: Configuration cfg = new Configuration();

If we execute the above instruction, then hibernate software will prepare configuration class object only in heap memory without configuration details.

- To encapsulate configuration details on configuration object we have to use the following methods from configuration class.

public Configuration configure()

Ex: cfg.configure();

- If we execute the above instruction hibernate software will search for the configuration file with default name

hibernate.cfg.xml

If hibernate software identify configuration file then hibernate software will perform configuration file loading, parsing and reading the content of configuration file then encapsulate all the configuration details in configuration type object.

JPG to PDF Converter For Mac - Unregis

In hibernate applications configuration file name is not fixed, it is possible to change configuration file name, if we change configuration file name then we have to provide new configuration file name to the hibernate software.

— To achieve the above we have to use the following methods

Ex: public configuration configure(String name)
↳ cfg.configure("myconfig.xml");

— If the configuration file is represented in the form of file class object then we have to use the following method

public Configuration configure(File f)

Ex: File f = new File("myconfig.xml");
Configuration cfg = new Configuration();
cfg.configure(f);

— If the configuration file is represented in the form of URL class object then we have to use the following method.

public Configuration configure(URL u)

Ex: URL u = new URL("../../Myconfig.xml");
Configuration cfg = new Configuration();
cfg.configure(u);

— In hibernate applications, configuration object will represent a particular database, if we use multiple no. of databases then we have to prepare multiple no. of configuration objects.

JPG to PDF Converter For Mac - Unregis

— In hibernate applications, a single configuration object is able to process multiple number of database requests that is threads. Therefore configuration object is thread safe w.r.to a particular database.

2. Prepare SessionFactory object :—

The main purpose of SessionFactory() object in hibernate applications is to manage connection objects, statement objects, prepared statement and so on---

To prepare connection objects, statements and so on SessionFactory object will pickup all the configuration details from configuration object.

— In hibernate applications to represent session factory hibernate software has provided the following predefined interfaces.

org.hibernate.SessionFactory

To prepare SessionFactory object, we will use the following methods from configuration class.

public SessionFactory buildSessionFactory()

ex: SessionFactory sf = cfg.buildSessionFactory();

— In hibernate applications, we will prepare a separate SessionFactory object for each and every database.

In hibernate applications a single SessionFactory object is able to manage multiple number of client requests that is threads so that SessionFactory object is thread safe.

JPG to PDF Converter For Mac - Unregis

Session Object creation:- In hibernate applications session object will represent a particular client request. To perform persistance operations in hibernate applications we will use some predefined methods, here the required predefined methods are provided by hibernate software in the form of session.

- In hibernate applications, to represent session object we ~~will~~ use hibernate software has provided a predefined interface. That is org.hibernate.Session ✓

To get session object we have to use the following methods from sessionfactory.

public Session openSession()

ex: Session s=sf.openSession()

*In hibernate applications configuration object, *sessionfactory object are thread safe. But session object is not thread safe.

Step ④: Create Transaction Object ✓

In hibernate applications when we establish the database connection implicitly hibernate software will change the connections default mode from auto commit to non-auto commit ✓

- whenever the database connection is in non-auto commit then we must perform either commit or rollback operations at the end of the database application.

- Transaction is representing an unit of work in order to perform the required persistance operations.

JPG to PDF Converter For Mac - Unregis

- In hibernate applications we must require a transaction object when we want to perform the persistance operations like create[insert], update, delete. Transaction object is not required for retrieval persistance operations.
- To represent transaction object in hibernate applications, hibernate software has provided a predefined interface that is

org.hibernate.Transaction ✓

- There are two ways to get transaction object

1. Using beginTransaction method

public Transaction beginTransaction()

ex: Transaction tx = s.beginTransaction();

- If we execute the above instruction, beginTransaction() method will return transaction object and it will start the transaction automatically.

(2) Using getTransaction() method:-

public Transaction getTransaction();

- If we execute the above instruction then getTransaction() method will return transaction object and it will not start transaction automatically.
- In the above context to start transaction, we have to use the following methods from Transaction.

public void begin() ✓

ex: tx.begin();

JPG to PDF Converter For Mac - Unregis

If we use transaction in hibernate application, at the end of the transaction we must perform either commit or rollback operations. For this we have to use the following methods

public void commit()
public void rollback()

perform persistance operations:- In hibernate applications after getting session object we have to use the following methods from session to perform all the persistance operations.

1) save() method
2) persist() method
3) saveOrUpdate() method
4) update() method
5) delete() method
6) get() method
7) load() method

- Hibernate software has provided all the above methods in session only to perform single record manipulations. In hibernate applications to perform multiple records manipulation we have to use the following hibernate features

1. HQL [Hibernate Query Language]
2. Native-SQL
3. Criteria API

JPG to PDF Converter For Mac - Unregis

— what is the difference b/w save() method and persist() method?

Date: 5th Apr 2012

A: In hibernate applications both save method and persist method can be used to insert a persistence object in database table. To insert a record in database table if we use save() method then save() method will insert the specified record in the database table and return its primary key value.

```
public Serializable save(object o) throws HibernateException
```

why? only

ex:

```
Employee e=new Employee("e1","a",5000);
String id=(String) s.save(e);
```

session object → return Serializable

To insert a record on database table if we use persist() method then persist() method will insert the specified records on to the database table but it will not return primary key value.

```
public void persist(object o) throws HibernateException
```

ex: s.persist(e);

JPG to PDF Converter For Mac - Unregis

saveOrUpdate() method: saveOrUpdate() method can be used to insert a specified record on database table if respective record is not available at database already and this method will perform update() operation when the respective record is already available at database.

public void saveOrUpdate(object o) throws HibernateException

update() method:

update() method can be used to perform updations on database table.

public void update(object o) throws Hibernate Exception

delete() method:

where delete() method can be used to delete a particular record from database table.

public void delete(object o) throws Hibernate Exception

Int. question:

What is the diff b/w get() method and load() method?

Ans: In hibernate applications, both get() method and load() method can be used to retrieve a particular record from database table.

To retrieve a record from database table if we use get() method then hibernate software will search for the respective record in database table on the basis of the provided primary key value, if it is available then hibernate software will retrieve that record data and return in the form of specified persistence object.

JPG to PDF Converter For Mac - Unregis

— If the specified record is not available in the database table then Hibernate software will return null as a return value from get method.

public Object get(String EntityClassName, Serializable primaryKeyValue) throws HibernateException

ex) Employee e=(Employee)s.get("Employee",4);

— If the specified record is available then Hibernate will return that object in the form of persistent object not as object that may type cast to object

— public Object get(class clas, Serializable primaryKeyValue) throws HibernateException

ex) Employee e=(Employee) s.get(Employee.class, "4");

JPG to PDF Converter For Mac - Unregis

load() method: To retrieve a particular record from database table if we use load() method then first hibernate software will search for the respective record on database table on the basis of the provided primary key value, if it is available hibernate software will retrieve the data from database table and return in the form of specified persistance class object. If the respective record is not available in database table then hibernate s/w will raise HibernateException.

```
public Object load(Class clas, Serializable pk-value)
        throws HibernateException
```

ex:

```
Employee e=(Employee) s.load
        (Employee.class, "e1");
```

Close Session and SessionFactory objects:

In hibernate applications, it is convention to close session and sessionfactory at the end of the hibernate applications to eliminate security issues.

- To close session and sessionfactory(), we will use the following method from Session and SessionFactory.

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

ex:

```
s.close();
sf.close();
```

JPG to PDF Converter For Mac - Unregis

Resource Structure: (D:\hbhu\appH)

- Employee.java [.class]
- employee.hbm.xml
- hibernate.cfg.xml
- ClientApp.java [.class]

hbm4
↑ hibernate
4th batch
teaching
by Nagoor
sir

ex① public class Employee
{ private int eno;
private String ename;
private float esal;
public void setENo(int eno)
{ this.eno=eno;
}
public void setEname(String ename)
{ this.ename=ename;
}
public void setEsal(float esal)
{ this.esal=esal;
}
public int getENo()
{ return eno;
}
public String getEname()
{ return ename;
}
public float getEsal()
{ return esal;
}
}

written in text editor

JPG to PDF Converter For Mac - Unregis

Save the above program as Employee.java in d:\hibernate\app folder.

→ In hibernate applications to prepare mapping file we have to provide doc type definition in order to perform xml parsing. To get hibernate mapping file DOC TYPE we have to use the following path✓

① extract hibernate3.jar file → org → hibernate

② Copy DOCTYPE tag into our xml file✓

open file

hibernate-mapping-3.0.dtd

Employee.hbm.xml

```
<!DOCTYPE hibernate-mapping
PUBLIC "-//Hibernate/Hibernate
Mapping DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate
mapping-3.0.dtd">
<hibernate-mapping>
<class name="Employee" table="empl">
<id name="eno" column="eno"/>
<property name="ename"/>
<property name="esal"/>
</class>
</hibernate-mapping>
```

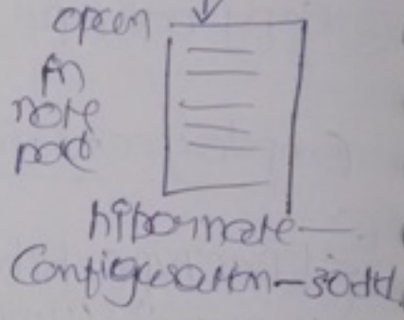
JPG to PDF Converter For Mac - Unregis

In hibernate applications to prepare configuration file, we have to provide DOCTYPE definition in our configuration file to perform xml parsing.

— To provide DOCTYPE definition in configuration file we have to use the following path.

① extract hibernate3.jar file → org → hibernate

and copy DOCTYPE tag and paste into our configuration file.



hibernate.cfg.xml

<DOCTYPE

```
<hibernate-configuration>
<session-factory>
<property
 name="connection.driver_class">oracle.jdbc.
        driver.OracleDriver</property>
<property name="connection.url">
 jdbc:oracle:thin@localhost:1521:xe
<property name="connection.user">  </property>
        system</property>
```

It is the database user name

JPG to PDF Converter For Mac - Unregis

```
<property name="connection.password">
durga </property>

<property name="hibernate.dialect">
org.hibernate.dialect.Oracle9Dialect </property>
<mapping>
    resource="Employee.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

↳ save this file as

```
import org.hibernate.*;
import org.hibernate.cfg.*;
public class ClientApp
{ public static void main(String[] args)
                throws Exception
  {
    Configuration cfg=new Configuration()
            cfg.configure();
  SessionFactory sf=cfg.buildSessionFactory();
Session s=sf.openSession();
    Employee e=new Employee();
            e.setEno(111);
            e.setEname("baal");
            e.setEsal(5000.0f);
```

JPG to PDF Converter For Mac - Unregis

```
Transaction tx=s.beginTransaction();
        s.save(e);
        tx.commit();
System.out.println("employee inserted
        s.close();                successfully!!");
        ff.close();
    }
}
```

↳ Save this file as CallerApp.java in D:/hibernate/app1

— cmd:>

c:>javac *.java

= compile time error.

c:> hbrsettings.jar

hbr settings 'jar' because that check where hibernate jar table is available or not

c:> Connect to the database

JPG to PDF Converter For Mac - Unregis

Date: 7th Apr, 2012

- To compile hibernate client application, we have to set class path environment variable to hibernate3.jar file.
- To execute hibernate applications we require the following list of jar files in class path environment variable.

1. ant-1.6.5.jar
2. antlr-2.7.6.jar
3. asm.jar
4. asm-attrs.jar
5. cglib-2.1.3.jar
6. commons-collections-2.1.1.jar
7. commons-logging-1.0.4.jar
8. dom4j-1.6.1.jar
9. ehcache-1.2.3.jar
10. hibernate3.jar
11. jaas.jar
12. jta.jar
13. ojdbc6.jar (oracle11xe)
14. mysql-connector-java-3.1.6-bin.jar (MySQL)

- If u want to use annotations in our hibernate applications we have to use the following jar files in class path environment variable along with the above list of jar files.

JPG to PDF Converter For Mac - Unregis

1. hibernate-annotations.jar
2. hibernate-commons-annotations.jar
3. ejb3-persistance.jar

— hibernate is having an inbuilt connection pooling mechanism, which is not suitable for the production mode. It is suggestible to use third party provided connection pooling mechanisms in hibernate applications at production mode. The well known third party provided connection pooling mechanisms are c3po, proxool

c3po-0.9.1.jar
proxool-0.8.3.jar

If we want to use any of the third party provided connection pooling mechanisms in hibernate applications, we have to use the following jar files in class path environment variables.

c3po-0.9.1.jar
proxool-0.8.3.jar

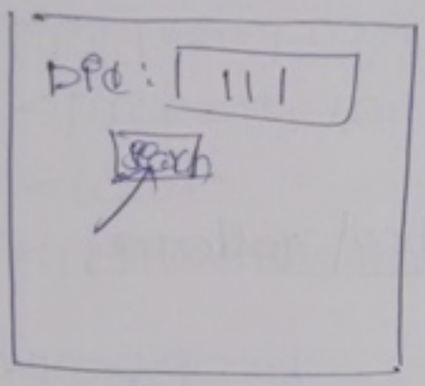
— If we want to use application servers provided connection pooling mechanism or data source or jdbc as middle ware service then we have to use the application server provided jar file in class path environment variable. In the case of weblogic server, we have to use the following jar file.

webulogic.jar

JPG to PDF Converter For Mac - Unregis

secondapplication:

If suppose I will enter a product id (111) on the GUI then it have to display the product details like.



d:\hbm\app2\
- product.java(.class)
- product.hbm.xml
- myConfig.xml
- searchframe.java (.class)
- clientApp.java (.class)

product.java

```
public class product
{ private String pid;
  private String pname;
  private int pcost;
public void setpid(String pid)
  { this.pid=pid;
  }
public String getPid()
  { return pid;
  }
public void setPname(String pname)
  { this.pname=pname;
  }
public String getpname()
  { return pname;
  }
```

JPG to PDF Converter For Mac - Unregis

```
public void setpcost(int pcost)
        {  this.pcost = pcost;
        }
public int getpcost()
        {  return pcost;
        }
}
```

— save the above class in <u>D:\hbm\</u> folder as product.java ✓

<u>product.hbm.xml</u> (mapping file)

```
<!DOCTYPE
```

JPG to PDF Converter For Mac - Unregis

```
<hibernate-mapping>
<class name="product" table="product">
 <id name="pid"/>
 <property name="pname"/>
 <property name="pcost"/>
 </class>
 </hibernate-mapping>
```

myconfig.xml

```
<!DOCTYPE

<hibernate-configuration>
<session-factory>
<property name="connection.driver_class">
oracle.jdbc.driver.OracleDriver </property>
<property name="connection.url"> jdbc:
oracle:thin:@localhost:1521:xe </property>
<property name="connection.user">system
</property>
<property name="connection.
password">
durga </property>
<property name="hibernate.dialect">
```

JPG to PDF Converter For Mac - Unregis

```
org.hibernate.dialect.OracleDialect
                                    </property>
<mapping resource="product.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

- Save the above file as MyConfig.cfg.xml in D:

④

```
import java.awt.*;
import java.awt.event.*;

import org.hibernate.cfg.*;
import org.hibernate.*;
public class SearchFrame extends Frame
    {                    implements ActionListener
        Configuration cfg;
        SessionFactory sf;
        Session s;
        product p;

        Label l;
        TextField tf;
        Button b;
```

JPG to PDF Converter For Mac - Unregis

```
public SearchFrame()
{ try
  {
      this.setVisible(true);
      this.setSize(500,500);
      this.setBackground(Color.pink);
      this.setTitle("GUI-HBN application");
  this.setLayout(new FlowLayout());
   l=new Label("product id");
   tf=new TextField(15);
   b=new Button("SEACH");
   b.addActionListener(this);
   this.add(l);
   this.add(tf);
   this.add(b);

cfg=new Configuration();
      cfg.configure("MyConfig.cfg");
   sf=cfg.buildSessionFactory();

  }
catch(Exception e)
      { e.printStackTrace();
      }
}
```

JPG to PDF Converter For Mac - Unregis

```
public void actionperformed(ActionEvent ae)
{   try
    {   String pid = tf.getText();
        s = sf.openSession();
        p = (Product) s.get(Product.class, pid);
        s.close();
    }
    catch(Exception e)
    {   e.printStackTrace();
    }
public void paint(Graphics g)
{   Font f = new Font("Arial", Font.BOLD, 30);
    g.setFont(f);
    g.drawString("productId--"+p.getPid(),
                 50,200);
    g.drawString("product name--"+p.getPname(),
                 50,250);
    g.drawString("product Cost..."+p.getPcost(),50,300);
}
}
```

— save the above file as SearchFrame.java in DBLHIBY1 folder ✓

JPG to PDF Converter For Mac - Unregis

clientApp.java — client application ✓

```
class ClientApp
    { public static void main(String[] args)
        { SearchFrame sf
        }
    }
```

— cd D:\Python\app21

— D:>

javac *.java

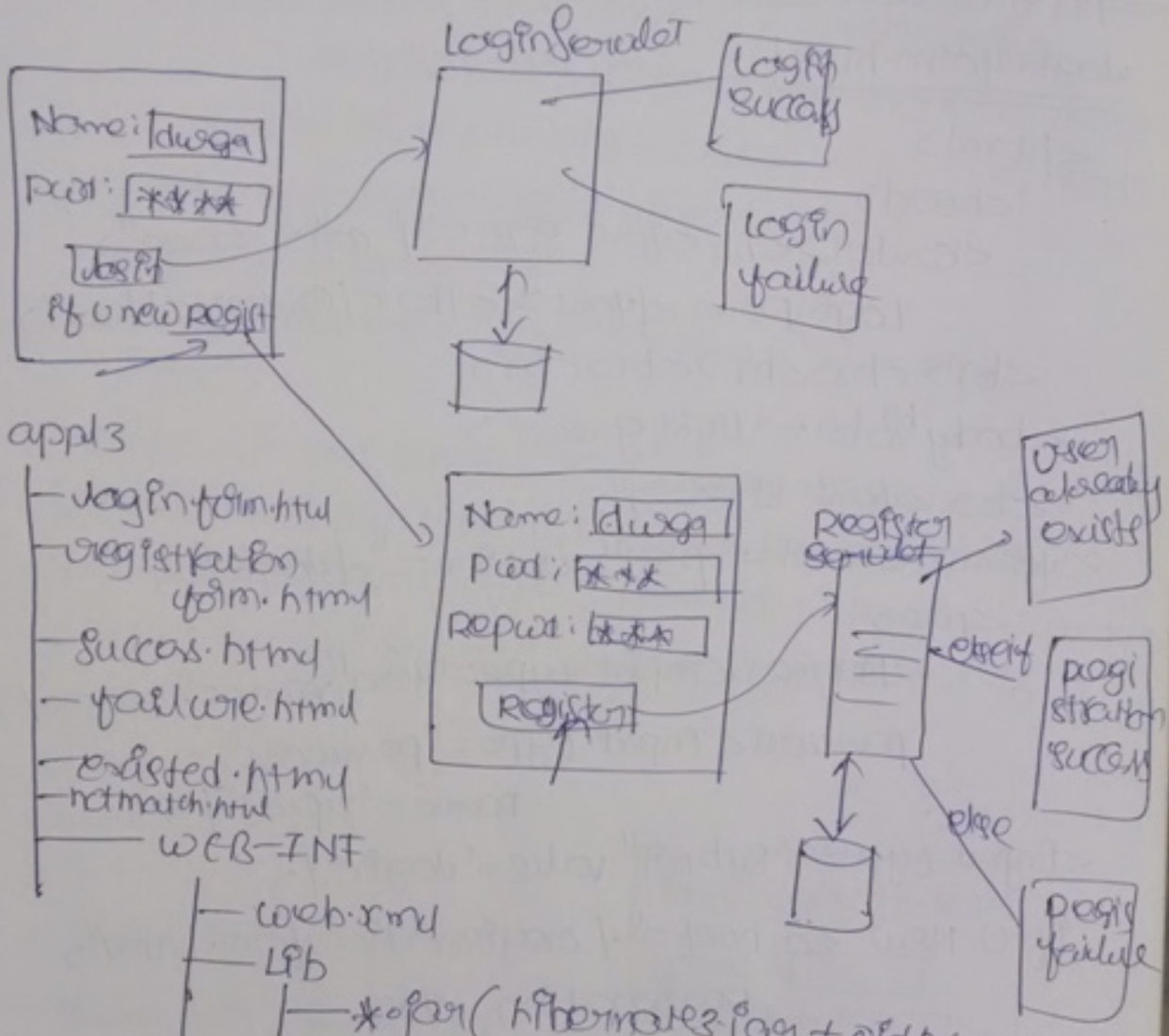
— now connect to the database

```
SQL> drop table product;
SQL> Commit;
SQL> create table product(pid varchar2(5), pname
SQL> Commit;
SQL> select * from product
```

PID	PName	PCOST

JPG to PDF Converter For Mac - Unregis

Date: 8th Apr, 2012



appl3
- login form.html
- registration form.html
- success.html
- failure.html
- existed.html
- notmatch.html
- WEB-INF
  - web.xml
  - Lib
    - *.jar (hibernate3.jar + ojdbc.jar)
  - classes
    - LoginServlet.java
    - RegistrationServlet.java
    - User.java (.class)
    - User.hbm.xml
    - hibernate.cfg.xml
    - Hibernate.Util.java (.class)

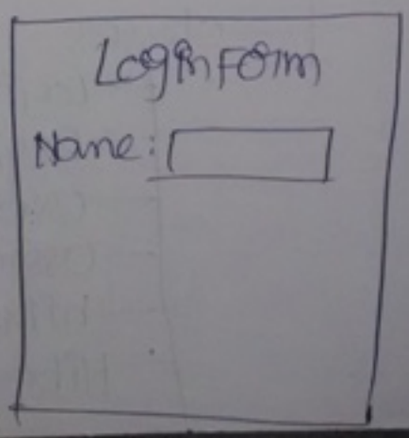
JPG to PDF Converter For Mac - Unregis

— Connect to the database
— create table registration_details(

— First of all create the view part ✓

loginform.html

```
<html>
    <head>
     <center><b><font size="7" color="red">
        LoginForm </font></b></center></head>
   <br><hr><hr><br><br>
  <body bgcolor="light green">
   <b><font size="7">
 <form method="post" action="./login">
    <pre>
        Name <input type="text" name="uname"/>
        password < input type="password"
                        name="upwd"/>
 <input type="submit" value="login"/>
 If U New <a href="./registrationform.html">
                Registration </a>
 </pre></form>
        </font>
       </b>
       </body>
    </html>
```



JPG to PDF Converter For Mac - Unregis

save under D:\hbn\app3 ✓

registrationform.html :-

```
<html>
  <head>
  <center><b><font size="7" color=red>
        Registration form </font></b></center>
    <br><br><br><br><br>
  </head>
<body bgcolor="lightpink">
  <b><font size="7">
  <form method="post" action="./reg">
  <pre>
Name <input type="text" name="uname"/>
password <input type="password name="upwd"/>
Repassword <input type="password" name="rpwd"/>
<input type="submit" value="Registration"/>
</pre>
</form>
</font>
</b>
</body></html>
```

— save this pgm under D:\hbn8\app3\

JPG to PDF Converter For Mac - Unregis

success.html :-

```
<html>
<body bgcolor="light yellow">
<center><b><font size="7" color="red">
<br><br>
success
</font></b></center></body></html>
```

↳ save under D:\hbm4\app3

success

failure.html

```
<html>
<body bgcolor="light yellow">
<center><b><font size="7" color="red">
<br><br>
Failure
</font></b></center></body></html>
```

save under D:\hbm4\app3

failure

existed.html

```
<html>
<body bgcolor="light yellow">
<center><b><font size="7" color="red">
<br><br>
User already exists
</font></b></center></body></html>
```

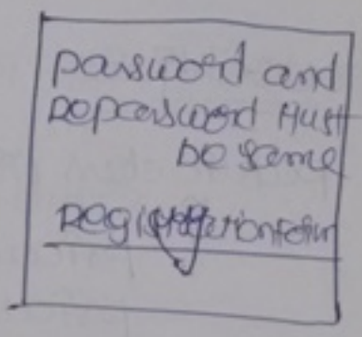
save under D:\hbm4\app3

user already exists

JPG to PDF Converter For Mac - Unregis

notmatch.html

```
<html>
  <body bgcolor="light yellow">
  <center><b><font size="7" color="red">
    <br><br>
  password and repassword values must be same
    <br><br>
  <a href="./registrationform.html">registration form</a>
  </font>
  </b>
  </center></body></html>
```



same window

D:\html\app3.

web.xml

```
<web-app>
  <welcome-file-list>
    <welcome-file> loginform.html </welcome-file>
  </welcome-file-list>
  <servlet>
    <servlet-name> ls </servlet-name>
    <servlet-class> LoginServlet </servlet-class>
  </servlet>
  <servlet-mapping>
  <servlet-name>
  <url-pattern> /login </url-pattern>
  </servlet-mapping>
  </servlet>
```

JPG to PDF Converter For Mac - Unregis

```
<servlet>
<servlet-name>.as</servlet-name>
<servlet-class>RegistrationServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>as</servlet-name>
<url-pattern>/reg</url-pattern>
</servlet-mapping>
</web-app>
```

↳ save under app3/WEB-INF/

= User.java

```
public class User
{
    private String uname;
    private String upwd;
    public void setUname(String uname)
        { this.uname=uname;
    }
    public String getUname()
        { return uname;
    }
    public void setUpwd(String upwd)
        { this.upwd=upwd;
    }
    public String getUpwd()
        { return upwd;
    }
}
```

save this under D:\bbnu\app3\WEB-INF\classes\

JPG to PDF Converter For Mac - Unregis

User.hbm.xml

```
<!DOCTYPE ---
    --- 3.0.dtd">
<hibernate-mapping>
 <class name="user" table="reg_Details">
 <id name="uname"/>
 <property name="upwd"/>
 </class>
 </hibernate-mapping>
```

↳ save under

hibernate.cfg.xml app3/WEB-INF/classes/

```
<!DOCTYPE ----
        -configuration-3.0.dtd">
<hibernate-Configuration>
  <Session-factory>
   <property
  name="Connection.driver_class">oracle.jdbc.driver.
        OracleDriver</property>
<property name="Connection.url">
  jdbc:oracle:thin@localhost:1521:xe
   </property>
<property name="connection.user">System</property>
<property name="connection.password">durga</property>
<property name="hibernate.dialect">
  org.hibernate.dialect.OracleDialect</property>
```

JPG to PDF Converter For Mac - Unregis

```
<mapping resource="User.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

↳ save the above one under

D:\
app1\WEB-INF\classes

HibernateUtil.java

```
import org.hibernate.cfg.*;
import org.hibernate.*;
public class HibernateUtil
{ static Configuration cfg;
  static SessionFactory sf;
  static Session s;
  static
  {
    try
    {
      cfg = new Configuration();
      cfg.configure();
      sf = cfg.buildSessionFactory();
    }//try
    catch(Exception e)
    {
      e.printStackTrace();
    }//catch
  }//static
  public static Session getSession()
  {
```

JPG to PDF Converter For Mac - Unregis

```
try
{
    s=sf.openSession();
}
catch(Exception e)
    {
        e.printStackTrace();
    }
    return s;
}
}
```

↳ save the program under \classes\

LoginServlet.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import org.hibernate.*;
public class LoginServlet extends HttpServlet
    {   public void doPost(HttpServletRequest
            req, HttpServletResponse res)
        throws ServletException, IOException
        {
            try
            {
            String uname=req.getParameter("uname");
        String upass=req.getParameter("upass");
        ~~User u~~
            Session s=HibernateUtil.getSessionFactory();
            User u=(User)s.get("User",uname);
```

JPG to PDF Converter For Mac - Unregis

```
if(u==null)
  {
req.getRequestDispatcher("/failure.html").
    }                    forward(req,res);
  else
  {
    req.getRequestDispatcher("/success.html").
    }                    forward(req,res);
  }
catch(Exception e)
  {  e.printStackTrace();
  }
}
```

↳ save-under

}

/classes ✓

RegistrationServlet.java

```
~~class~~ import javax.servlet.*;
      import javax.servlet.http.*;
      import java.io.*;
import org.hibernate.*;
public class RegistrationServlet extends HttpServlet
  {
    public void doPost(HttpServletRequest req,
                       HttpServletResponse res)
    throws ServletException, IOException
    {   try
        {
```

JPG to PDF Converter For Mac - Unregis

Date: 8th Apr, 2012

```
String uname= req.getParameter("uname");
String upwd = req.getParameter("upwd");
String cpwd=req.getParameter("cpwd");
    if(upwd.equal(cpwd))
    {
  Session s=HibernateUtil.getSession();
User user=(User)s.get("User,uname);
    if(user==null)
      {  User u=new User();
         u.setUname(uname);
         u.setUpwd(upwd);
     Transaction tx=s.beginTransaction();
  String pkValue=(String)s.save(u);
      tx.Commit();
    if(pkValue.equals(uname))
     {
   req.getRequestDispatcher("success.html")
                .forward(req,res);
      }
     else
      {
req.getRequestDispatcher("failure.html")
                .forward(req,res);
        }
    }
   else
    {
req.getRequestDispatcher("existed.html").
                forward(req,res);
```

Size of every pk register at run time

JPG to PDF Converter For Mac - Unregis

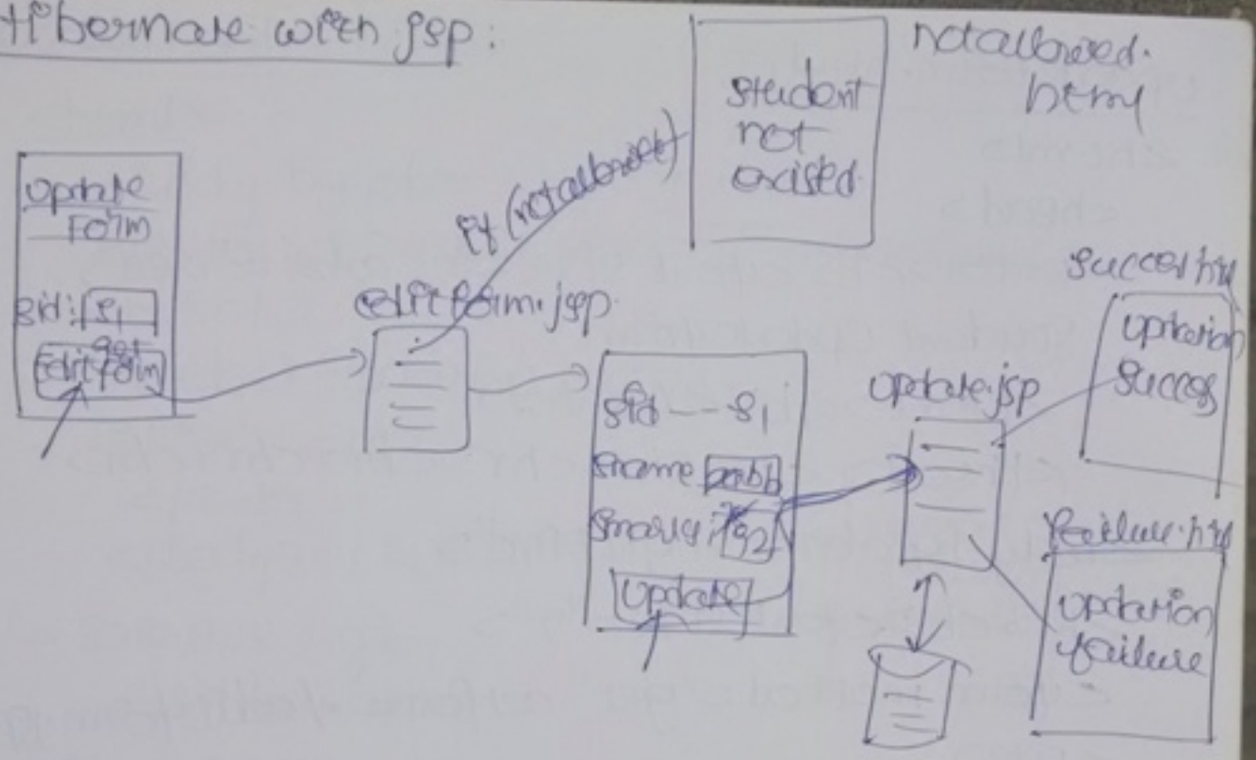
```
    }
  }
  else
  {
    req.getRequestDispatcher("/notmatch.html").
                    forward(res, req);

    }
  }
  catch(Exception e)
      {
        e.printStackTrace();
      }
    }
  }
```

Execution

JPG to PDF Converter For Mac - Unregis

Hibernate with jsp:




```
app4
 |- updateform.html
 |- notexisted.html
 |- success.html
 |- failure.html
 |- editform.jsp
 |- update.jsp
 |- student.hbm.xml
 |- hibernate.cfg.xml
 |- Student.class
 |- WEB-INF
     |- web.xml
     |- lib
         |- *.jar
            (hibernate3.jar
            ojdbc.jar)
     |- classes
         |- com.dss.
            Student.class
         |- student.hbm.xml
         |- hibernate.cfg.xml
```

JPG to PDF Converter For Mac - Unregis

updateform.html:-

```
<html>
  <head>
  <center><b><font size="4" color="red">
   Student Updateform
    </font></b></center>
    </head><br><br><hr><br><br><br>
 <body bgcolor="lightpink">
  <br><b><font size="4">
  <form method="get" action="./editform.jsp">
  <pre>
 Student id : <input type="text" name="sid"/>
 <input type="submit" value="getEditform"/>
   </pre></form></font></b></body>
   </html>
```

↳ save this pgin under, D:\hbm4\app4\ ✓

Notexisted.html:-

```
<html>
 <body bgcolor="lightyellow">
  <center><b><font size="4" color="red">
   <br><b>
   Student not existed.
   </font></b></center></body></html>
```

↳ save

D:\hbm4\app4\

Student not existed

JPG to PDF Converter For Mac - Unregis

success.html

```
<html>
  <body bgcolor="light yellow">
  <center><b><font size="7" color="red">
  <br><b>
   student updation success
   </font></b>
   </center>
   </body></html>
```

↳ save this under
D:\hbnu\app\

student updation success

failure.html:-

```
<html>
  <body bgcolor="light yellow">
   <center><b><font size="7" color="red">
   <br><b>
   student updation failure
   </font></b>
   </center>
   </body></html>
```

student updation failure

student.java

```
package com.dss;
public class student
  {
    private string sid;
    private string sname;
    private int smarks;
  }
```

JPG to PDF Converter For Mac - Unregis

```
public void setSid(String sid)
    { this.sid=sid;
    }
public String getSid()
    { return sid;
    }
public void setSname(String sname)
    { this.sname=sname;
    }
public String getSname()
    { return sname;
    }
public void setSmarks(int smarks)
    { this.smarks=smarks;
    }
public int getSmarks()
    { return smarks;
    }
}
```

↳ save this pgm under D:\hbnu\appu\ ✓

student.hbm.xml

JPG to PDF Converter For Mac - Unregis

```
<hibernate-mapping>
 <class name="student" table="student">
  <id name="sid"/>
  <property name="sname"/>
  <property name="smarks"/>
  </class>
   </hibernate-mapping>
```

↳ save this under

D:\hbm4\app4\ ✓ (why?)

observe here application folder (save this here)

— hibernate.cfg.mm

```
<!DOCTYPE ----->
<hibernate-configuration>
 <session-factory>
  <property name="connection.driver_class">
   oracle.jdbc.driver.OracleDriver </property>
  <property name="connection.url">
   jdbc:oracle:thin@localhost:1521:xe. </property>
  <property name="connection.user">system
                                    </property>
  <property name="connection.password">
                                    durga </property>
  <property name="hibernate.dialect">
   org.hibernate.dialect.Oracle9Dialect. </property>
  <mapping resource="student.hbm.mm"/>
 </session-factory>
</hibernate-configuration>
```

↳ save this under

D:\hbm4\app4\ ✓

JPG to PDF Converter For Mac - Unregis

editform.jsp

```
<%@ page
import="org.hibernate.cfg.*; org.hibernate.*;
                                com.dss.student"%>
<%!
Configuration cfg;
SessionFactory sf;
Session s;
String sid;
%>
<%
try
{
  cfg=new Configuration().configure();
  sf=cfg.buildSessionFactory();
  application.setAttribute("sf", sf);
  sid=request.getParameter("sid");
  s=sf.openSession();
  Student std=(student)s.get("com.dss.student", sid);
  if(std==null)
    {  %>
       <jsp:forward page="notexisted.html"/>
    <%
  }
  else
    {  %>
       <html>
       <head><center><b>
         <font size="7" color="red">
```

no quote

student.class

JPG to PDF Converter For Mac - Unregis

```
Student Edit form
</font>
</b>
</center></head><br><br><br><br><br>
<body bgcolor="lightpink">
<b><font size="7">
<form method="get" action="./update.jsp">
<pre>
        Student Id.....<%=sid%>
<input type="hidden" name="sid" value=
                        '<%=sid%>' />
Student Name<input type="text" name="sname"
                value='<%=std.getSname()%>'/>
Student marks.
<input type="text" name="smarks"
                value='<%=std.getSmarks()%>' />
<input type="submit" value="update"/>
</pre></form></font></b></body></html>
<%
}
} catch(Exception e)
    { e.printStackTrace()
    }
%>
```

save this pgm in D:\bhanu\apache

JPG to PDF Converter For Mac - Unregis

update.jsp

```
<%@page import="org.hibernate.*, com.dss.*"%>
<%!
  SessionFactory sf;
  Session s;
  Student std;
%>
<%
  try
  {  sf=(SessionFactory)application.getAttribute
     s=sf.openSession();                    ("sf");
  String sid=request.getParameter("sid");
String sname=request.getParameter("sname");
int smarks=(Integer) request.getParameter("smarks");
  std=new Student();
     std.setSid(sid)
     std.setSname(sname);
     std.setSmarks(smarks);
Transaction tx=s.beginTransaction();
        s.update(std);
      tx.commit();
%>
<jsp:forward page="success.html"/>
<%
}catch(Exception e)
    {
```

JPG to PDF Converter For Mac - Unregis

```
%>
<jsp:forward page="failure.html"/>
<%
e.printStackTrace();
}
%>
```

↳ save this under,
D:\hbm\appy\

web.xml

```
<web-app>
  <welcome-file-list>
    <welcome-file> updateform.html </welcome-file>
  </welcome-file-list>
</web-app>
```

↳ D:\hbm\appy\WEB-INF\

2→ Note: If we want to use hibernate in web applications then we have to provide all the hibernate respective xml files under classes folder. To use pojo class in jsp pages, we have to provide package declaration statement in pojo class and we have to compile it with -d option.

JPG to PDF Converter For Mac - Unregis

MySQL

Date: 11th April 2012

mysql> show databases;

durga

mysql> Drop database durga;
1 row effected

mysqll> create database durga;
1 row effected

mysql> show databases;

mysql> use database durga;

mysql> show tables;

mysql>

JPG to PDF Converter For Mac - Unregis

mysql> insert into product values("p1", 'AAA', 500);
1 row effected.
mysql> commit;
mysql> select * from product;

pid	pname	pcost
p1	AAA	500.

mq> update product set pcost = pcost + 200 where pid = 'p1';
1 row effected.
mq> select * from product;

pid	pname	pcost
p1	AAA	700

mq> delete from product where pid = 'p1';
1 row effected.
mq> commit;
mq> select * from product;
Empty set ✓
mq> drop table product;
0 rows effected.
mq> show tables;
Empty set ✓

we can real storage of rollback

In oracle → rollback (optional)

JPG to PDF Converter For Mac - Unregis

If we want to interact with mysql database, we have to provide the following modifications in configuration file.

1) connection.driver_class : com.mysql.jdbc.Driver
2) connection.url : //localhost:3306/db_Name
3) connection.user : root
4) connection.password : root
5) hibernate.dialect.MySQLDialect
   ↳ small

HSQLD } Cap like

2) To interact with mysql database, from hibernate application, we have to keep the following jar file in class path environment variable.

mysql-connector-java-3.1.6-bin.

— In hibernate applications, when we perform a persistance operation automatically hibernate software will prepare the database dependent sql query. Here if we want to see hibernate prepared sql queries on command prompt, we have to use the following property in configuration file

<property name="show-sql">true</property>

— Here now, I want to prepare an application in which I interact with two databases one oracle and another one mysql at a time simultaneously

JPG to PDF Converter For Mac - Unregis

Here in this program, maintain the same database table name in both databases

Two Configuration files for each database

apps
- Employee.java (.class)
- Employee.hbm.xml
- OracleConfig.xml
- mySqlConfig.xml
- ClientApp.java

```
mS> show tables;
mS> use dargatb;
mS> create table empl (eno int(5), ename char(10));
mS> desc employee;
```

JPG to PDF Converter For Mac - Unregis

Employee.java

```
public class Employee
{   private int eno;
    private String ename;
    private float esal;

    public void setEno( int eno)
        { this.eno=eno;
        }
    public int getEno()
        { return eno;
        }
    Public void setEname(String ename)
        { this.ename=ename;
        }
    public String getEname()
        { return ename;
        }
    public void setEsal( float esal)
        { this.esal=esal;
        }
    Public float getEsal()
        { return esal;
        }
}
```

↳ save this pgm under
0:\hbnd\apps\

JPG to PDF Converter For Mac - Unregis

Employee.hbm.xml

```
<DOCTYPE ---- 3.0.dtd">
<hibernate-mapping>
 <class name="employee" table="emp1">
<id name="eno"/>
<property name="ename"/>
<property name="esal"/>
</class>
</hibernate-mapping>
```

↳ save this pgm under, 'D:\hbm4\apps\' ✓

Enter act hibernate-jar ↳ hibernate-mapping-3.0.dtd open and copy DOCTYPE

oracleConfig.xml :-

```
<DOCTYPE--- >
<hibernate-configuration>
<session-factory>
<property name="connection.driver_class">
 oracle.jdbc.driver.OracleDriver </property>
<property name="connection.url">
 jdbc:oracle:thin@localhost:1521:xe </property>
<property name="connection.user">system
<property name="connection.password">
 durga </property>
<property name="hibernate.dialect">
   org.hibernate.dialect.OracleDialect
<property name="show_sql">
 true </property>
<mapping resource="employee.hbm.xml"/>
```

copy from org.hibernate.

JPG to PDF Converter For Mac - Unregis

```
</session-factory>
</hibernate-configuration>
```

↳ save this pgm under D:\hbm\apps\ ✓

~~MySQL~~ mysqlconfig.xml

```
<DOCTYPE -->
<hibernate-configuration>
<session-factory>
<property name="connection.driver_class">
    com.mysql.jdbc.Driver </property>
<property name="connection.url">
    jdbc:mysql://localhost:3306/dwgadb
<property name="connection.user">root </property>
<property name="connection.password">root
<property name="hibernate.dialect">
    org.hibernate.dialect.MySQLDialect </property>
<property name="show-sql">true </property>
<mapping resource="employee.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

JPG to PDF Converter For Mac - Unregis

ClientApp.java

```
import org.hibernate.cfg.*;
import org.hibernate.*;
~~class~~ public
class ClientApp
{
public static void main(String args[])
{                                throws Exception
Configuration daoConfig = new Configuration();
daoConfig.configure("daoConfig.xml");
Configuration mysqlConfig = new Configuration();
mysqlConfig.configure("mysqlConfig.xml");
SessionFactory daoSf = daoConfig.
        buildSessionFactory();
SessionFactory mysqlSf = mysqlConfig.
        buildSessionFactory();
Session daoSes = daoSf.openSession();
Employee e=(Employee) daoSes.get
                (Employee.class, 111);
Session mysqlSes = mysqlSf.openSession();
Transaction tx = mysqlSes.beginTransaction();
mysqlSes.save(e);
tx.commit();
daoSes.close();
mysqlSes.close();
daoSf.close();
mysqlSf.close();
}
}
```

Two Cf objects

Two session factory objects

D:\hbm4\apps

JPG to PDF Converter For Mac - Unregis

both jar files are required-
open cmd prompt ✓
D:\hpnu\apps> hbrsettings.bat
\apps> javac *.java

mysql>select * from empl;

ero	ename	esal
111	aaa	500 ✓

JPG to PDF Converter For Mac - Unregis

Int. Que :

— Is it possible to design and execute any hibernate application without providing Configuration file.

Ans yes ✓.

How it is possible?

It is possible to design hibernate applications without using Configuration file. But by using programatic approach.

Date: 12th apr, 2012

In hibernate applications, the main purpose of Configuration file is to provide all the Configuration details to hibernate software inorder to perform all the persistance operations. In hibernate applications there are two ways to provide Configuration details to hibernate software

① programatic approach.
② Declarative approach.

— programatic approach; —

In programatic approach we will not provide hibernate Configuration file. we will provide all the Configuration details by using some predefined methods provided by Configuration class explicitly.

To set a particular hibernate property to Configuration object we have to use the following methods from Configuration class.

JPG to PDF Converter For Mac - Unregis

public void setProperty(String name, String value);

where name should be hibernate.connection.xxx for connection oriented properties.

— To specify dialect property we have to use "hibernate.dialect" as property name.

— To add mapping file to the configuration object we have to use the following method from Configuration class:

public void addFile(String file_name)

Ex: The above procedure programmatic approach applying to the first application in our hibernate.cfg.xml) and delete hibernate.cfg.xml class.

~~Ex~~ ~~Pgm~~ ClientApp

```
import org.hibernate.*;
import org.hibernate.cfg.*;

public class ClientApp
{
public static void main(String[] args)
                throws Exception
{
}
```

JPG to PDF Converter For Mac - Unregis

```
Configuration cfg=new Configuration();
cfg.setproperty("hibernate.connection.driver_class",
"oracle.jdbc.driver.OracleDriver");
cfg.setproperty("hibernate.connection.url",
        "jdbc:oracle:thin@localhost:1521:xe");
cfg.setproperty("hibernate.connection.user",
                "system");
cfg.setproperty("hibernate.connection.password",
                "durga");
cfg.setproperty("hibernate.dialect",
  "org.hibernate.dialect.Oracle9Dialect");
  cfg.addFile("employee.hbm.cmf");
SessionFactory sf=cfg.buildSessionFactory();
  Session s=sf.openSession();
      Employee e=new Employee();
        e.setEno(111);
        e.setEname("bbb");
        e.setEsal(9000.0f);
  Transaction tx=s.beginTransaction();
          s.saveOrUpdate(e);
          tx.commit();
  System.out.println("employee inserted or
```

JPG to PDF Converter For Mac - Unregis

```
uploaded successfully");
        s.close();
        st.close();
    }
}
```

JPG to PDF Converter For Mac - Unregis

— Note: If we use programmatic approach to provide configuration details then we should not access configure() method on configuration object reference.

Drawback: In hibernate applications programmatic approach is not suggestable to set all the configuration details because it is difficult to perform modifications or changes on configuration details in the future. To overcome this problem we have to use declarative approach.

② Declarative approach:—

In declarative approach, we will define a file with the required configuration details and we will send that file to hibernate software. In this context hibernate software will pick up all the configuration details from the specified configuration file and prepare configuration object.

There are two ways to set configuration details in declarative approach,

(i) using properties file
(ii) using xml file

— Using properties file:—

In this approach we will define a properties file, where we will provide the required hibernate configuration details in the form of key, value pairs.

JPG to PDF Converter For Mac - Unregis

After preparing properties file, we will load all the properties file content into properties object through file input stream and we will set properties object to configuration object.

— To set properties object to configuration object we will use the following method from Configuration class without using configure method.

public void setProperties(Properties p)

Note: In this mechanism, we should not provide mapping file configuration in properties file, we have to add mapping file to the Configuration object by using addFile method.

Ex:

MyConfig.properties

hibernate.connection.driver_class = oracle.jdbc.driver.OracleDriver

hibernate.connection.url = jdbc:oracle:thin@localhost:1521:xe

hibernate.connection.user = system

hibernate.connection.password = durga

hibernate.dialect = org.hibernate.dialect.Oracle9Dialect

JPG to PDF Converter For Mac - Unregis

Client Application.

```
import org.hibernate.*;
import org.hibernate.cfg.*;
import java.util.*;
import java.io.*;
public class clientApp
{ public static void main(String a) throws
                                    Exception
  {
    Configuration cfg=new Configuration();
    properties p=new properties();
FileInputStream fis=new FileInputStream
                    ("myconfig.properties");
    p.load(fis);
    cfg.setproperties(p);
    cfg.addFile("employee.hbm.cfg");
SessionFactory sf=cfg.buildSessionFactory();
```

JPG to PDF Converter For Mac - Unregis

Note: In hibernate application if we have not used configure method on configuration object then hibernate software will search for hibernate.properties as a default properties file to get all the configuration details.

— If we provide properties file name as hibernate.properties then in client application it is not required to set properties object to configuration object explicitely and it is not required to load properties file content into properties object.

Date: 14th Apr, 2012

— Declarative approach using xml file:—

In this approach we will provide all the configuration details to the hibernate software by using an xml file.

— In case of properties file declarative approach we are able to provide all the configuration details in the properties file except mapping resources but in case of xml file declarative approach we are able to provide all the configuration details including mapping resources.

— If we provide all the configuration details in an xml file then we have to send that xml file name and location to hibernate software.

To achieve this we have to use the following method from Configuration class.

JPG to PDF Converter For Mac - Unregis

public Configuration configure(String name).

— If we use hibernate.cfg.xml as configuration file name, it is not required to send configuration file name to hibernate software, simply we have to access the following method

— To provide all the configuration details we will use the following xml tags in configuration file.

```
<hibernate-configuration>
    <session-factory>
    <property name="name">value</property>
    <mapping resource="mapping file"/>
    </session-factory>
</hibernate-configuration>
```

— Is it possible to design any hibernate application without using mapping file or not?

Ans: It is possible to design hibernate application without using mapping file but by using annotations. If we want use annotations in hibernate application then we have to use the following steps.

Step①: eliminate hibernate mapping file and use annotations in the respective persistance class.

In hibernate applications mapping file will provide the meta data about the mappings b/w persistance class name and table name,

JPG to PDF Converter For Mac - Unregis

Id property and primary key column name, properties name and column name and soon.

— If we want to eliminate mapping file from hibernate applications then we have to provide the same meta data by using annotations.

To achieve the above requirement EJB3 has provided the following four basic annotations in javax.persistance package.

① @Entity
② @Table
③ @Id
④ @column

— where entity annotation will represents the persistance class as an entity class, it is representing a particular table in database. @entity is a marker annotation.

Syntax @Entity

— where table annotation is a class level annotation, it can be used to provide the mapping b/w persistance class and the respective database table.

Syntax: @Table (name="__")

where name parameter will take table name.

2 — where @Id annotation is a marker annotation, it will make the present property as Id property, representing primary key column in the respective database table.

JPG to PDF Converter For Mac - Unregis

Syntax @Id

— where @Column annotation will provide the mapping between the property and the respective database table column.

@Column annotation is single valued annotation

Syntax: @Column(name="__")

where name parameter will take the column name defined in the respective db table.

Step no ②: configure annotated class in configuration file in place of mapping file configuration.

To configure annotated persistance class in configuration file we have to use the following xml tag

```
<hibernate-configuration>
    <session-factory>
        =
        <mapping class="__"/>
    </session-factory>
</hibernate-configuration>
```

where class attribute in mapping tag will take fully qualified name of annotated persistance class.

Step③: prepare annotation configuration class object in client application.

JPG to PDF Converter For Mac - Unregis

— In general in hibernate applications, we will use configuration object to encapsulate all the configuration details which we provided in configuration file.

— Configuration object is not sufficient to encaps configuration details provided by annotations ware from annotated persistance class. To encapsulate annotations provided configuration details hibernate has provided a separate configuration class as a sub class to org.hibernate.cfg.Configuration. That is

org.hibernate.cfg.AnnotationConfiguration

Example:

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

Database

accNo	accName	accType

dsbbn4\
app6

- Account.java (class)
- hibernate.cfg.xml
- ClientApp.java (class)

JPG to PDF Converter For Mac - Unregis

Account.java

```
import javax.persistance.*;
@Entity
@Table(name="account")
public class Account
{
    @Id
    @Column(name="accNo")
    private String accNo;
    @Column(name="accName")
    private String accName;
    @Column(name="accType")
    private String accType;
    public void setAccNo(String accNo)
        { this.accNo=accNo;
        }
    public String getAccNo()
        { return accNo;
        }
    public void setAccName(String accName)
        { this.accName=accName;
        }
    public String getAccName()
        { return accName;
        }
    public void setAccType(String accType)
        { this.accType=accType;
        }
    public String getAccType()
        { return accType;
        }
}
```

JPG to PDF Converter For Mac - Unregis

Notes: In annotated persistance class it is possible to use @column annotation at both property level and method level. That is along with getter methods.

hibernate.cfg.xml:

```
<hibernate-configuration>
 <session-factory>
 <property name="connection.driver_class">
 oracle.jdbc.driver.OracleDriver </property>
 <property name="connection.url">
jdbc:oracle:thin@localhost:1521:xe </property>
 <property name="connection.user">system
                                  </property>
 <property name="connection.password">dugq
 <property name="hibernate.dialect">  </property>
 org.hibernate.dialect.Oracle9Dialect </property>
 <mapping class="Account"/>
 </session-factory>
 </hibernate-configuration>
```

— ClientApp.java

```
import org.hibernate.cfg.*;
import org.hibernate.*;
public class ClientApp
  { public static void main(String args[])
          throws Exception
    {
```

JPG to PDF Converter For Mac - Unregis

```
Configuration cfg = new AnnotationConfiguration
        cfg.configure();                    ();
SessionFactory sf = cfg.buildSessionFactory();
Session s = sf.openSession();
    Account a = new Account();
        a.setACCNo("a1");
        a.setACCName("AAA");
        a.setACCType("saveProps");
Transaction tx = s.beginTransaction();
        s.save(a);
        tx.commit();
    System.out.println("Account Inserted
        s.close();                  successfully");
        sf.close();
    }
}
```

dpl

JPG to PDF Converter For Mac - Unregis

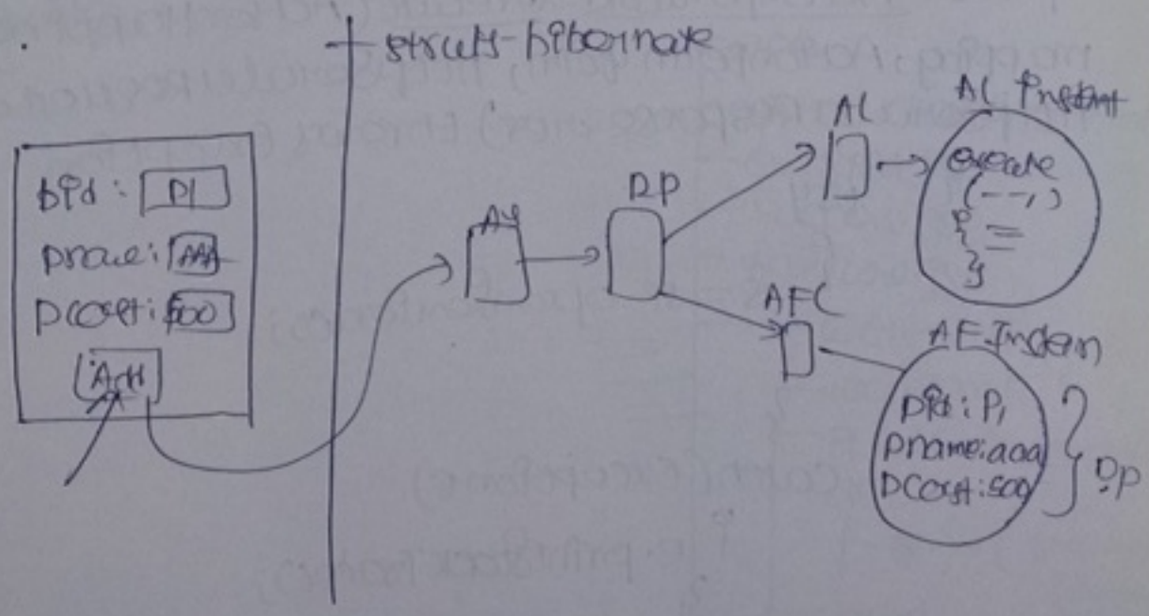
Struts-Hibernate Integration: [Date: 16th April, 2011]

— In enterprise application development there are three ways to integrate hibernate with struts based web application.

1. Manual approach.
2. ActionServlet customization.
3. By preparing plugin class.

Struts-hibernate integration through manual approach

In this approach we will make ready only struts environment at the time of server startup, and we will prepare hibernate environment as part of the respective Action classes. That is we will prepare hibernate respective configuration object, SessionFactory object as part of the respective action class.



JPG to PDF Converter For Mac - Unregis

Ans --- Import section ---

```
Public class productAddAction extends Action
{   Configuration cfg;
    SessionFactory sf;
    Session s;
  public productAddAction()
  {   try
      {
        cfg=new Configuration();
        cfg.configure();
      }  sf=cfg.buildSessionFactory();
    catch(Exception e)
      {
        e.printStackTrace();
      }
  }
public ActionForward execute(ActionMapping
mapping, ActionForm form, HttpServletRequest req,
HttpServletResponse res) throws Exception
{   try
    {   s=sf.openSession();
    }
    catch(Exception e)
      {  e.printStackTrace();
      }
  }
}
```

JPG to PDF Converter For Mac - Unregis

In the above approach, when we perform the respective Action class instantiation then only hibernate environment will be prepared, this approach will take lot of time to perform respective action, as a result the performance of the application will be reduced.

② Struts-hibernate integration by customizing ActionServlet:-

In this approach we will prepare both struts framework environment and hibernate environment at the time of server startup only.

— with this mechanism in the respective Action class we will get directly Session object and we will interact with the database through hibernate environment inorder to perform the respective action.

```
shapp
 |- product addform.html
 |- existed.jsp
 |- success.jsp
 |- failure.jsp
 |- WEB-INF
    |- web.xml
    |- struts-config.xml
    |- lib
       |- *.jar (struts + hibernate jars)
    |- classes
       |- com.dss.
          MyController.class
       |- hibernate.cfg.
          xml
```

JPG to PDF Converter For Mac - Unregis

```
WEB-INF
 |- classes
    |- product.hbm.xml
    |- com.dss.HibernateUtil.class
    |- com.dss.productAddActionForm.class
    |- com.dss.productAddAction.class
    |- com.dss.product.class ✓
```

productadditionform.html

```
<html>
<head>
<center><b><font size="7" color="red">
    productAddForm
</font></b></center></head><br><br><br>
<br>
<body bgcolor="light pink">
<b><font size="7">
<form method="get" action="add.do">
<pre>
    productId <input type="text" name="pid"/>
product name <input type="text" name="pname"/>
product cost <input type="text" name="pcost"/>
<input type="submit" value="ADD"/>
</pre></form></font></b></body></html>.
```

↳ save under \shapp\ ✓

JPG to PDF Converter For Mac - Unregis

existed.jsp

```
<html>
 <body bgcolor="lightyellow">
<center><b><font size="7" color="red">
  <br> <br>
  product existed already
  </font></b></center></body></html>
```

↳

\shrapp\ ✓

success.jsp

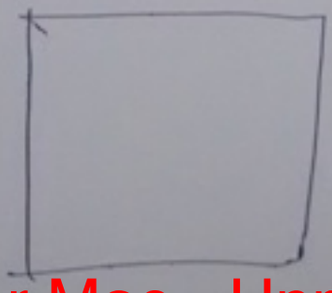
```
<html>
 <body bgcolor="lightyellow">
 <center><b><font size="7" color="red">
   <br><br>
   product Added successfully
   </font></b></center></body></html>
```

failure.jsp

```
<html>
  <body bgcolor="lightyellow">
   <center><b><font size="7" color="red">
     <br><br>
     product insertion failure
     </font></b></center></body></html>
```

↳ save under

\shrapp\ ✓



(Venport completed)

JPG to PDF Converter For Mac - Unregis

productAddActionForm.java

```
package com.dsf;
import org.apache.struts.
public class productAddActionForm extends ActionForm
{
    private String pId;
    private String pname;
    private int pcost;
    public void setpId(String pId)
        { this.pId=pId;
        }
    public String getpId()
        { return pId;
        }
    public void setpname(String pname)
        { this.pname=pname;
        }
    public String getpname()
        { return pname;
        }
    public void setpcost(String pcost)
        { this.pcost=pcost;
        }
    public int getpcost()
        { return pcost;
        }
```

JPG to PDF Converter For Mac - Unregis

ProductAddAction.java

```
package com.dss;
 import org.apache.struts.action.*;
 import javax.servlet.http.*;
 import org.hibernate.*;

public class ProductAddAction extends Action
  { public ActionForward execute(ActionMapping
mapping, ActionForm form, HttpServletRequest
req, HttpServletResponse res) throws Exception
    { String status="";
      try
      {
productAddActionForm as=(ProductAddActionForm)
                                        form;
 String pid= paf.getPid();
  String pname=paf.getPname();
   int pcost= paf.getCost();

  Session s= HibernateUtil.getSession();
  product p=(product) s.get(product.class, pid);
    if(p==null)
      { Transaction tx=s.beginTransaction();
   product prd=new product();
      prd.setPid(pid);
       prd.setPname(pname);
        prd.setPcost(pcost);
          s.save(prd); tx.commit();
```

JPG to PDF Converter For Mac - Unregis

```
    status="success";
  }
  else
  {
    status="existed";
  }
}
catch(Exception e)
        { status="failure";
          e.printStackTrace();
        }
  return mapping.findForward(status);
  }
}
```

HibernateUtil.java

```
package com.dss;
import org.hibernate.cfg.*;
import org.hibernate.*;
public class HibernateUtil
  {  static Configuration cfg;
     static SessionFactory sf;
     static Session s;
  public static void createSessionFactory()
     {  try
        {
          cfg=new Configuration();
              cfg.configure();
              sf=cfg.buildSessionFactory();
        }
      Catch(Exception e)
          { e.printStackTrace();
        }
```

JPG to PDF Converter For Mac - Unregis

```
}
public static Session getSession()
    {  try
       {
       s=sf.openSession();
       }
       catch(Exception e)
         { e.printStackTrace();
         }
       return s;
    }
```

↳ \closes\

— MyController.class

```
package com.dss;
import org.apache.struts.action.*;
import javax.servlet.*;
public class MyController extends ActionServlet
 {  public void init(ServletConfig config) throws
                                ServletException
    {
HibernateUtil.createSessionFactory();
      super.init(config);
    }
 }
```

JPG to PDF Converter For Mac - Unregis

product.java

```
package com.dss;
public class product
{
    private String pId;
    private String pname;
    private String pcost;
    public void setpId(String pId)
        { return this.pId=pId;
        }
    public String getpId()
        { return pId;
        }
    public void setpname(String pname)
        { this.pname = pname;
        }
    public String getpname()
        { return pname;
        }
    public void setpcost(int pcost)
        { this.pcost=pcost;
        }
    public int getpcost()
        { return pcost;
        }
}
```

JPG to PDF Converter For Mac - Unregis

struts-Config.xml

Date: 16th Apr/2012

```
<struts-Config>
<form-beans>
<form-bean name="addform" type="com.dss.
            productAddActionForm"/>
</form-beans>
<action-mappings>
<action path="/add" name="addform"
   type="com.dss.productAddAction">
<forward name="success" path="/success.jsp"/>
<forward name="failure" path="/failure.jsp"/>
<forward name="existed" path="/existed.jsp"/>
</action>
</action-mappings>
</struts-Config>
```

web.xml

```
<web-app>
<welcome-file-list>
<welcome-file> productaddform.html </welcome-file>
</welcome-file-list>
<servlet>
<load-on-startup>1</load-on-startup>
<servlet-name>ay</servlet-name>
<servlet-class> com.dss.MyController
                </servlet-class>
</servlet>
```

JPG to PDF Converter For Mac - Unregis

```
<servlet-mapping>
<servlet-name>a</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>
</web-app>
```

JPG to PDF Converter For Mac - Unregis

product-hbm.xml

```
<hibernate-mapping>
<class name="com.dss.product" table="product">
<id name="pid"/>
<property name="pname"/>
<property name="price"/>
</class>
</hibernate-mapping>
```

hibernate.cfg.xml

```
<hibernate-configuration>
<session-factory>
<property name="connection.driver_class">
oracle.jdbc.driver.OracleDriver </property>
<property name="connection.url">
jdbc:oracle:thin:@localhost:1521:xe </property>
<property name="connection.user">system
</property>
<property name="connection.
password">durga </property>
<property name="hibernate.dialect">org.hibernate.
dialect.Oracle9Dialect </property>
<mapping resource="product.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

JPG to PDF Converter For Mac - Unregis

- To prepare hibernate environment at the time of server startup along with struts framework environment the above approach is not all suggestible.
- In struts framework ActionServlet is the front controller it is not suggestible customize ActionServlet
- while in the customization of ActionServlet if we provide any misconfiguration in then the entire struts framework may not perform its functionalities so that the respective action may not be executed.
- To overcome the problem we use plugin

struts-hibernate integration through plugins

- To integrate hibernate with struts framework has not provided any predefined support
- In struts framework, validator framework is a sub framework to get data validation support.
- In struts framework tiles framework is a sub framework to get all the layout mechanisms.
- To integrate tiles framework and validator framework struts has provided some predefined plugin classes like org.apache.struts.validator.ValidatorPlugin and org.apache.struts.tiles.TilesPlugin
- If we configured above plugin classes in struts configuration in order to provide data validations and layout supports.
- ActionServlet will perform plugin classes loading, initialization and instantiation as part of ActionServlet initialization that is at the time of server startup.

JPG to PDF Converter For Mac - Unregis

— If we want to integrate hibernate with struts based web application we have to use the following 2 steps explicitly.

Step① prepare plug in class.

Step②: Configure plug in class in struts-Configuration file.

→ Step①: prepare plugin class:

To prepare plugin classes struts frame work has provided a predefined interface i.e org.apache.struts.action.PlugIn

— If we want to prepare our own plug in class to integrate hibernate with struts then we have to take an user defined class which must implement plugin interface provided by struts framework.

```
public interface PlugIn
{ public void init(ActionServlet as, ModuleConfig mc)
      throws ServletException
  public void destroy();
}

public class HibernatePlugIn implements PlugIn
{
  ≡
}
```

where the purpose of init() method is to perform plugIn class initialization, If we want to integrate with hibernate with struts framework then we need to provide all the hibernate configurations inside init() i.e preparing configuration object and

JPG to PDF Converter For Mac - Unregis

Session factory object

— where destroy() can be used to perform plugin class deinstantiation, where we have to provide all the cleanup operations like destroying the configuration object and sessionFactory object.

In hibernate integrate with struts.

— Configure plugIn class in struts configuration file to configure any plugIn class in struts configuration file we have to use the following tag.

```
<struts-config>
    ==
    <plug-in className="__">
    <set-property name="__" value="__">
    </plug-in>
    ==
</struts-config>
```

where class name of the attribute will take fully qualified name of plugIn class where setproperty tag

① used to pass any input parameters to plugin class, where name attribute setproperty will take parameter name and value attribute will take parameter value.

JPG to PDF Converter For Mac - Unregis

IDE

Eclipse IDE

eclipse-jee-indigo-sr2-win32
org.easystruts
Struts

Steps to design J2SE application with eclipse IDE:

1. Create java project

File → new → other → java project ✓

click on [next] button

If we click on next button, a window will be open to take the details about project

project name : [firstapp]

- ☑ use default location

JRE

⊙ use an execution environment JRE [JavaSE-1.7]

○

○

project Layout

○ use project folder as root for sources and class files

⊙ create separate folders for sources and class files.

[Next] [Finish] [cancel] another window open ✓

then click on [next] ✓

If we click on finish button, project folder will be created in the left side frame of eclipse.

project explorer

△ firstapp

▷ src

▷ JRE System Library

JPG to PDF Converter For Mac - Unregis

step②: create a java class

- Right click on project → new → class

If we select class then a window will be open to take all the details about class.

Java class

source folder: firstapp/src

package: com.dss

[ ] enclosing type.

Name: First

Modifiers: ⊙ public ○ default ○ private ○ protected

[ ] abstract [ ] final [ ] static.

Superclass: java.lang.object

Interfaces:

Which method stubs would you like to create?

[✓] public static void main(String[] args)

[ ] constructors from superclass

[ ] Inherited abstract methods

Do you want to add comments?

[ ] Generate comments.

Finish Cancel

JPG to PDF Converter For Mac - Unregis

- If we click on Finish button then a class with main method will be created in the middle frame of eclipse. where we have to provide our application logic.

Step ③ Run the java application

select Run from menu then Run as → Java application

Then we are able to seen the output on console located at bottom ✓

Note: In general if we want to execute any JDBC application we will use driver jar file provided by the respective database vendor in class path environment variable.

Due to this reason, if we want to design and execute JDBC application with eclipse IDE we have to add the required driver jar file to our Java project.

- To do this we have to use the following path

- Right click on project (first) → properties → Java Build path → Libraries

then select the required jar file through the file dialogue box ✓

Add external jars

here we select

C:\oracle\--ojdbc6.jar

ok

JPG to PDF Converter For Mac - Unregis

Steps to design hibernate application with eclipse IDE:-

Step①: prepare java project with hibernate library.

File → new → java project.

project name: hbnappl (maintain the remaining ones as usual)

Next

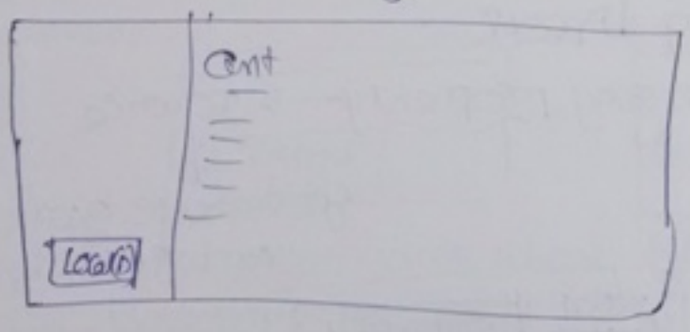
If we click on next button another window will be open to provide all java settings.

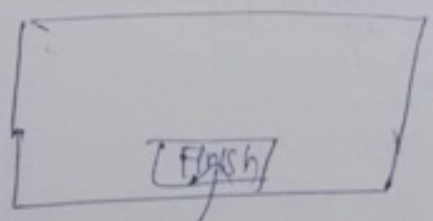
To add hibernate library we have to use the following path

Libraries

Add external jars

select all the hibernate jars only which are available in our system hard disk.





After a referenced library will be added to our project (hbnappl) on the left hand side.

JPG to PDF Converter For Mac - Unregis

Step ② : Prepare persistance class:

Right click on project (hbnapp) → New → class

Source folder: hbnapp/src

package: com.dss

Name: Employee

◉ public

Super class: java.lang.Object

Finish

```
package com.dss;
public class Employee
{
    private int eno;
    private String ename;
    private float esal;
```

// To generate getters and setters we have to use the following direction.

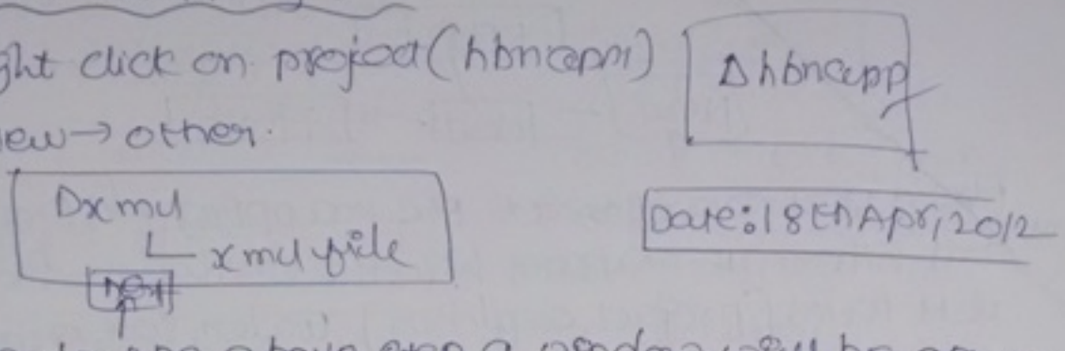
Right click on any property → source → generate getters and setters.

setsal
getsal
selectall
ok

Step no ③

Prepare mapping file:

Right click on project (hbnapp) → New → other.

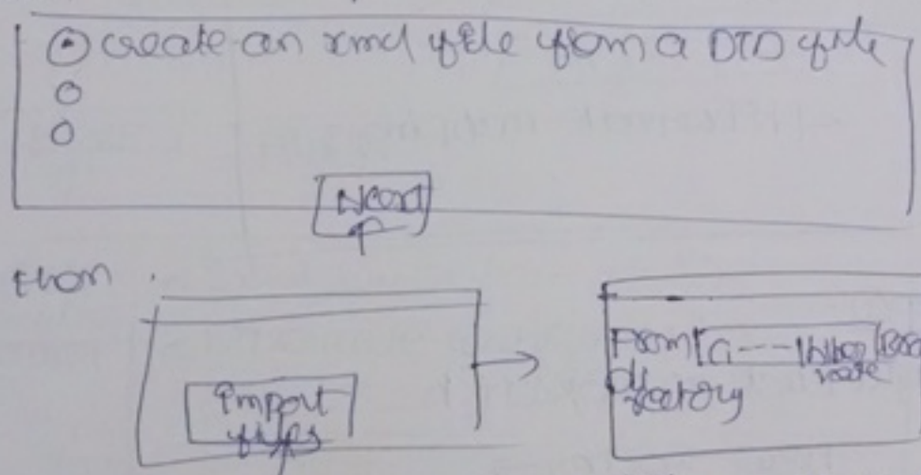


If we do the above step, a window will be open, ~~where we~~ where the project directory structure will be available where we have to select src folder and we have to provide Employee.hbm.xml as file name

File name [Employee.hbm.xml]

Next

then



Select the folder name where hibernate-mapping-3.0.dtd from directory ~~Textfield~~

↳ C:\users\durga\desktop\hibernate3\dtd\hibernate

If we select the above then automatically all the files will be listed in the right side window, where we have to select hibernate-mapping-3.0.dtd.

JPG to PDF Converter For Mac - Unregis

select project root into folder: [hbncap] [Browse]

[Finish]

[Next] → [Next] → [Finish]

then you can observe the mapping file Employee.hbm.xml and hibernate-mapping-3.dtd on the left hand [project explorer] under src and session

— If we do the above steps

where we edit the file as per our requirement

Body part

```
<DOCTYPE ---">
 <hibernate-mapping>
   =
   =
   =
 </hibernate-mapping>
```

— Step③:

right click on project (h

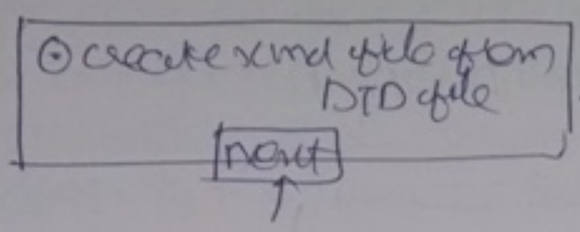
new → other →

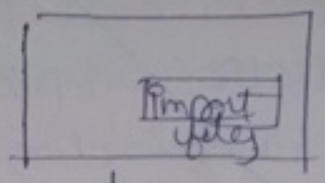
expand the project directory structure select src and give hibernate-cfg.xml as file name ←

[☐ xml
└ xml file
[Next]

src
[hibernate-cfg.xml]
[Next]

JPG to PDF Converter For Mac - Unregis

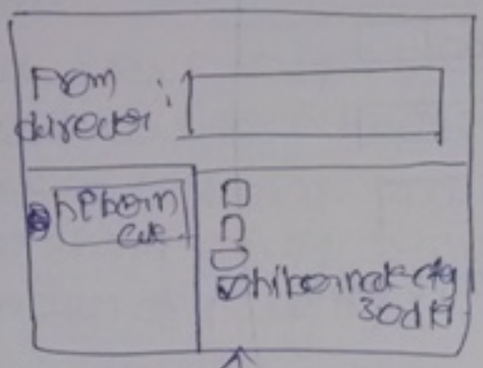




Select a folder which includes hibernate-config-3.0.dtd.

From directory

C:\Users\dugal\Desktop\hibernate\org\hibernate.

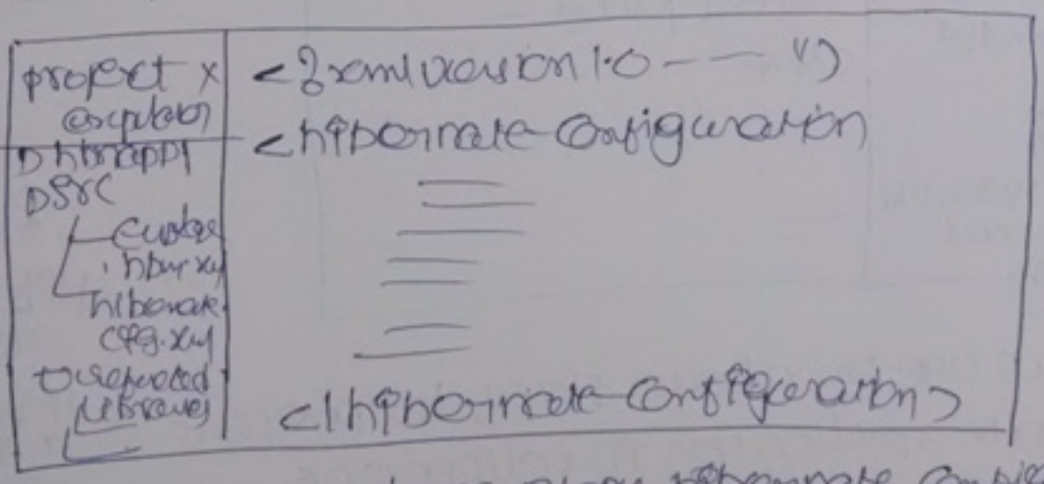


— If we do so as above a list of files will be appear on the right hand side. Select hibernate-Config-1.3.10.dtd file

Select project root into folder hbrnapp1 Browse

Finish

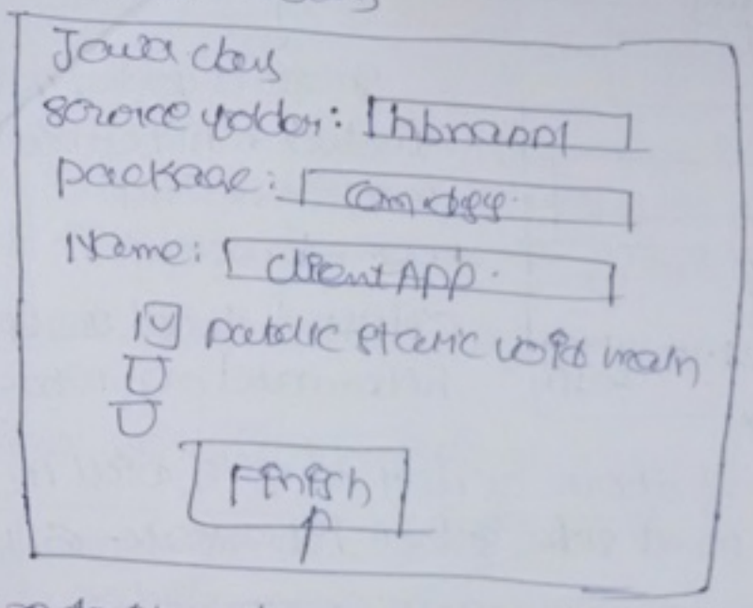
Next Finish



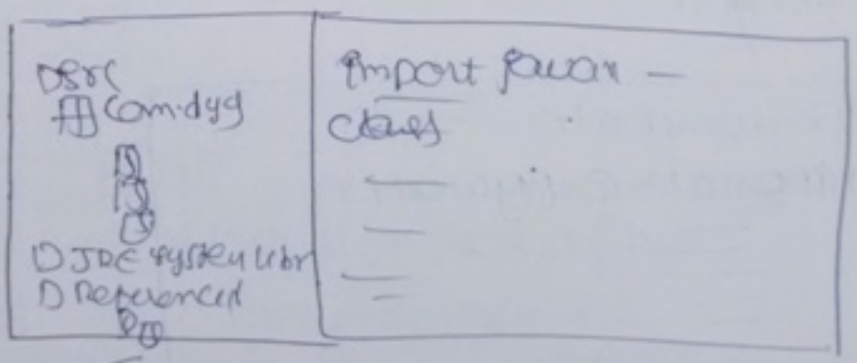
If we do the above steps hibernate Configuration file will be open in the middle frame of eclipse where we have to edit the Configuration file

JPG to PDF Converter For Mac - Unregis

right click on project (hbnapp1)
new → class



If we do the above steps automatically a class will be created in middle frame of eclipse IDE, where we have to provide the required persistence logic



If we use the above steps to prepare hibernate application in eclipse IDE then we are unable to get any predefined support from IDE like the support way provided by some other IDE's like MyEclipse, Net Beans and so on --

plugins to eclipse IDE
- Hibernate
- Hiber eclipse
- JBoss Tools
- Hibernate synchronizer

— In general from java based IDE's we will expect the configuration file, mapping file from POJO class or persistance class from mapping file as automatic process. But eclipse IDE is not providing this type of support.

If we want to get the above specified support we need to use plugins explicitly.

— To design hibernate application with eclipse IDE some third party organizations has provided number of plugins.

1. Hibernator
2. Hiber eclipse
3. JBoss Tools.
4. Hibernate synchronizer.
5. Hiber n8 IDE

JPG to PDF Converter For Mac - Unregis

Persistance Object Life Cycle

Date: 19th Apr, 2012

— In every persistance mechanism, it is required to store some objects on to the database table. Here the objects which are stored on the database table are called as persistance objects.

— In hibernate applications, each and every persistance object may exhibit either of the following states throughout the hibernate application.

1. transient
2. persistant
3. detached
4. Removed

Transient: In hibernate applications if we create a pojo class object by using new keyword then the state of the pojo class object is transient.

— If any persistance object is not representing any record in the database table and if it is not in the scope of hibernate then the state of that object is transient.

— If any pojo class object is available in transient state then it is not possible to perform updations on database table from pojo class object. That is synchronization is not possible b/w persistance object and database table record.

— In hibernate applications if we keep transient object in ideal mode upto some long time automatically that transient object is eligible for garbage collection.

Ex: Account a=new Account();

JPG to PDF Converter For Mac - Unregis

Persistant State

- In hibernate applications if we perform the persistent operations like save, update, saveOrUpdate() on the session with the transient object automatically the state of the persistance object is persistance. If any pojo class object is representing a particular database table record and if it is in the scope of hibernate then the state of the persistance object is persistant.
- In hibernate applications if we perform the operations like get, load, and so on-- on session automatically hibernate software will prepare persistance object with the record data, here the pojo class object will be available directly in persistant state without transient state
- If any persistance object is in persistant state then it is possible to provide synchronization b/w pojo class object and database table record

Ex
```
Session s = sf.openSession();
Transaction tx = s.beginTransaction();
    s.save(a); //persistent
    tx.commit();
```

JPG to PDF Converter For Mac - Unregis

Detached state:

- In hibernate application, when we access closed() method on session that is when we terminated the session automatically the persistance object will transfer from persistant state to detached state.

If any persistance object is available in detached state then it is possible to recover in to persistent state by performing the persistance methods like save, update, saveOrUpdate() by re establish the session.

- If any pojo class object is representing a particular database table record and which is not in the scope of hibernate, then that persistance object will be available in detached state.
- If any persistance object is available in detached state then it is not possible to perform synchronization between pojo class object and database table record.

Ex:

```
Session s1 = sf.openSession();
    Transaction ta = s.beginTransaction();
        s.save(a); //persistent
        ta.commit();
        s.close(); //Detached
    Session s2 = sf.openSession();
Transaction ta = s2.beginTransaction();
        s.saveOrUpdate(a); //persistent
        ta.commit();
```

JPG to PDF Converter For Mac - Unregis

Removed state:

— In hibernate applications if we remove or delete a particular database record with any persistent object then automatically that persistance object will be in removed state.

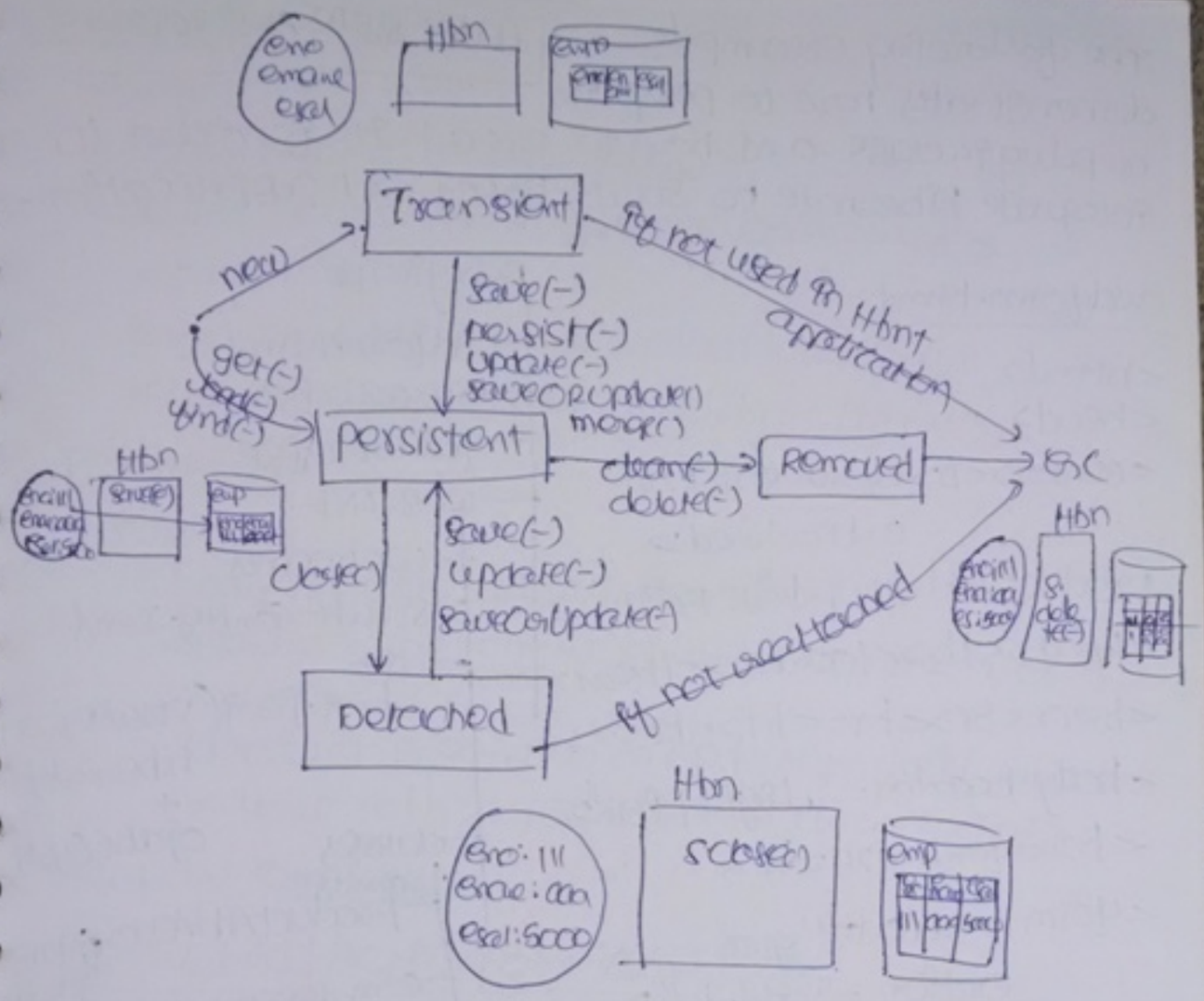
— In hibernate applications if any persistance object is available in removed state then it is not possible to recover into persistance state.
If any persistance object is available in removed state then it is not possible to perform synchronization b/w pojo class object and database table record.

— In hibernate applications removed objects may refer a particular database table record previously but it will not represent any table record at present and removed objects will be eliminated from the hibernate scope.

— In general in hibernate applications we will keep persistance objects in removed state at the end of the applications.

Ex:
```
Account a=(Account) s.get(Account.class,111);
s.delete(a); //Removed // persistance state
tx.commit();
```

JPG to PDF Converter For Mac - Unregis



JPG to PDF Converter For Mac - Unregis

The following example demonstrates how to prepare a plugin class and how to execute it, in order to integrate hibernate to struts based web application.

Date: 22nd Apr, 2012

addform.html

```
<html>
<head>
<center><b><font size='7'
        color='red'>
product Add or update Form
</font></b></center></head>
<br><br><hr><br><br>
<body bgcolor="lightpink">
<b><font size='7'>
<form method="get"
      action="add.do">
<pre>
product Id: <input type="text"
      name="pid"/>
product Name: <input type="text"
      name="pname"/>
product Cost= <input type="text" name="pcost"/>
<input type="submit" value="ADD"/>
</pre></form></font></b></body></html>
```

pluginapp

- addform.html
- success.jsp
- failure.jsp
- WEB-INF
  - web.xml
  - struts-config.xml
  - lib
    - *.jar (struts.jar + hibernate.jar + ojdbc6.jar)
  - classes
    - com.dss.productAddAction.form.java
    - com.dss.productAddAction.java
    - com.dss.product.java
    - product.hbm.xml
    - hibernate.cfg.xml
    - com.dss.HibernatePlugIn.java

JPG to PDF Converter For Mac - Unregis

success.jsp

```
<html>
<body bgcolor="lightyellow">
<center><b><font size='7' color='red'>
<br><br>
product Addition or updation success.
</font></b></center></body></html>.
```

failure.jsp

```
<html>
<body bgcolor="lightyellow">
<center><b><font size='7' color="red">
<br><br>
product insertion or updation failure
</font></b></center></body></html>
```

View part completed

productAddActionForm.~~xxx~~.java

```
package com.dss;
import org.apache.struts.action.*;
public class productAddActionForm extends
    ActionForm
{
    private String pid;
    private String pname;
    private int price;
    public void setpid(String pid)
        {
            this.pid = pid;
        }
    public String getpid()
        {
            return pid;
        }
```

JPG to PDF Converter For Mac - Unregis

```
public void setpname(String pname)
    {
        this.pname=pname;
    }
public String getpname()
    {
        return pname;
    }
public void setpcost(int pcost)
    {
        this.pcost=pcost;
    }
public int getpcost()
    {
        return pcost;
    }
}
```

productAddAction.java

```
package com.dse;
import org.apache.struts.action.*;
import javax.servlet.http.*;
import javax.servlet.*;
import org.hibernate.*;
public class productAddAction extends Action
{ public ActionForward execute(ActionMapping
mapping, ActionForm form, HttpServletRequest
HttpServletResponse res) throws Exception ex,
    {
        String str="";
        try
        {
```

JPG to PDF Converter For Mac - Unregis

```
ProductAddActionForm paf=(ProductAddActionForm) form;
String pid= paf.getPid();
String pname =paf.getpname();
int pcost=paf.getpcost();
ServletContext context = req.getServletContext();
SessionFactory sf=(SessionFactory) context.getAttribute
                    ("Session_Factory");
Session s= sf.openSession();
Transaction tx= s.beginTransaction();
        Product prd= new Product();
          prd.setpid(pid);
          prd.setpname(pname);
          prd.setpcost(pcost);
           s.saveOrUpdate(prd);
              tx.commit();
              status="success";
          }
        catch(Exception e)
          {  status ="failure";
             e.printStackTrace();
          }
        return mapping.findForward(status);
      }
    }
```

JPG to PDF Converter For Mac - Unregis

product.java

```
package com.dbs;
public class product
{ private string pid;
  private string pname;
  private int pcost;
  public void setpid( string pid)
    { this.pid=pid;
    }
  public string getpid()
    { return pid;
    }
  public void setpname( string pname)
    { this.pname=pname;
    }
  public string getpname()
    { return pname;
    }
  public void setpcost( int pcost)
    { this.pcost=pcost;
    }
  public int getpcost()
    { return pcost;
    }
}
```

JPG to PDF Converter For Mac - Unregis

Hibernateplugin.txt

```
package com.dss;
import org.apache.struts.action.*;
import org.apache.struts.Config.*;
import org.hibernate.cfg.*;
import org.hibernate.*;
import javax.servlet.*;
public class HibernatePlugIn implements PlugIn
    {  Configuration cfg;
       SessionFactory sf;
public void init(ActionServlet as, ModuleConfig mc)
                                    throws ServletException
  {  try
     {
        cfg=new Configuration();
        cfg.configure();
        sf = cfg.buildSessionFactory();
ServletContext context = as.getServletConfig().
                                    getServletContext();
context.setAttribute("session_factory", sf);
     }
     catch(Exception e)
       {
         e.printStackTrace();
       }
  }
  public void destroy()
       {  sf=null;
          cfg=null;
       }
  }
```

JPG to PDF Converter For Mac - Unregis

product.hbm.xml

```
<DOCTYPE ---
"">
<hibernate-mapping>
<class name="com.dss.product" table="product"/>
<id name="pid"/>
<property name="pname"/>
<property name="price"/>
</class>
</hibernate-mapping>
```

hibernate.cfg.xml

```
<DOCTYPE ---">
<hibernate-configuration>
<session-factory>
<property name="connection.driver_class">
oracle.jdbc.driver.OracleDriver </property>
<property name="connection.url">
jdbc:oracle:thin@localhost:1521:xe </property>
<property name="connection.user">system </property>
<property name="connection.password">durga9
<property name="hibernate.dialect">
org.hibernate.dialect.OracleDialect </property>
<mapping resource="product.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

JPG to PDF Converter For Mac - Unregis

struts-config.xml

```
<struts-config>
<form-beans>
<form-bean name="addForm"
        type="com.dss.productAddActionForm"/>
</form-beans>
<action-mappings>
<action path="/add" name="addForm"
                type="com.dss.productAddAction">
<forward name="success" path="/success.jsp"/>
<forward name="failure" path="/failure.jsp"/>
</action>
</action-mappings>
<plug-in classname="com.dss.HibernatePlugIn"/>
</struts-config>
```

web.xml

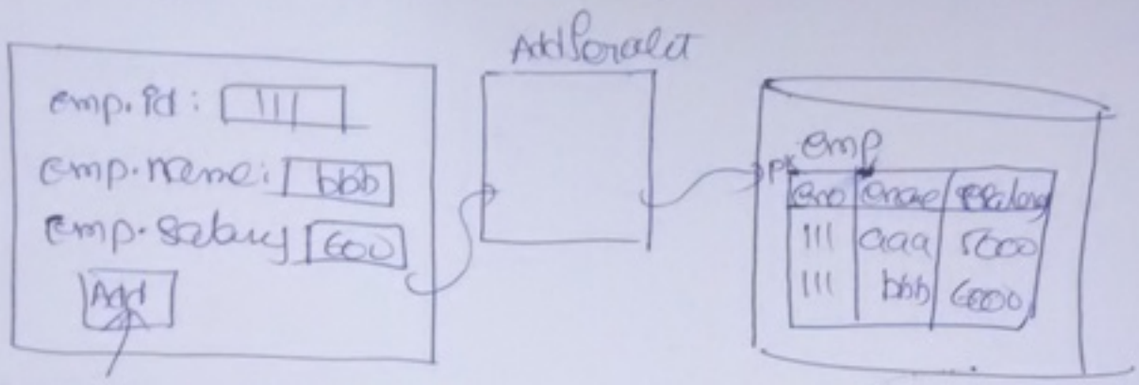
```
<web-app>
<welcome-file-list>
<welcome-file>addSelectionform.html</welcome-file>
</welcome-file-list>
<servlet>
<load-on-startup>1</load-on-startup>
<servlet-name>a</servlet-name>
<servlet-class>org.apache.struts.action.
                ActionServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>a</servlet-name>
```

JPG to PDF Converter For Mac - Unregis

```
<url-pattern>*.do</url-pattern>
</servlet-mapping>
</web-app>
```

JPG to PDF Converter For Mac - Unregis

Primary key generation Algorithms:



Date: 22nd Apr, 2012 Sun

- In general in database applications to simplify the persistance operations like retrieve, insert, update and so on we should require primary key column in the respective database table.
- The main purpose of primary key columns in database applications is to recognize all the database table records independently.
- primary key in database table is one (or) a set of columns, which should not allow duplicate values and which should not allow null values.
- In general in enterprise applications we will prepare the user forms like registration form to store a particular record data in database table, where if we give an option to the user to generate primary key values then users may not provide unique values because users may not get the complete database table instance every time.
- In the above context if user provide any record of data with primary key value then that data can be inserted in the respective database table if that record is not existed already.

JPG to PDF Converter For Mac - Unregis

- If the user specified record is existed already in database table then there may be chance to get exceptions like the ~~[illegible]~~ violation of primary key constraints.
- To overcome the above problem we should not give the any option to the users to generate primary key values, we should give an option to use the hibernate s/w to generate primary key values automatically for every record
- To achieve the above requirement insertion we have to use primary key generation algorithm
- Hibernate s/w has provided the following list of primary key generation algorithms to perform insert operation.

1. assigned
2. Increament
3. sequence
4. Identity
5. Native
6. hilo
7. seqhilo
8. uuid
9. guid
10. select
11. foreign

JPG to PDF Converter For Mac - Unregis

— Hibernate f/w has provided, a separate predefined class for each and every primary key generation algorithm except native in org.hibernate.id package in hibernate.jar file

— To specify a particular primary key generation algorithm in our hibernate applications we need to configure either the generator class or the short name of generator class. To configure a particular primary key generation algorithm in our mapping file then we have to use the following tags in mapping file.

```
<hibernate-mapping>
    <class name="_" table="_">
        <id name="_" column="_">
        <generator class="_">
            <param name="_">value</param>
            ==
        </generator>
        </id>
        ==
    </class>
</hibernate-mapping>
```

— where generator tag can be used to configure primary key generation algorithm. where class attribute in generator tag will take either fully qualified name of primary key generator class or short name of the primary key generator.

2 where <param> tag can be used to provide a single parameter to the primary key generator in order to generate primary key values.

JPG to PDF Converter For Mac - Unregis

— where name attribute in param tag will take Parameter name.

① Assigned primary key generation algorithm (Assigned primary key algorithm)

Assigned primary key generation algorithm is not having its own mechanism to generate primary key values, it will give an option to the client application to generate primary key values.

This mechanism can be used to generate primary key values of any datatype. This mechanism could be supported by almost all the databases. To represent this mechanism hibernate s/w has provided a separate predefined class that is org.hibernate.id.Assigned with the short name assigned

This mechanism should not require any input parameter.

```
<id name="eno" column="eno">
    <generator class="assigned"/>
</id>
```

— Increament algorithm:— This primary key generation algorithm will prepare the primary key value by increamenting the maximum primary key value available in the existed database table record.

$$PK_value = \boxed{max(PK_column\ in\ database\ table) + 1}$$

This primary key generation algorithm will prepare the primary key values of the datatypes short, int and long.

— This primary key generation algorithm should not require any input parameters. This algorithm will be supported by almost all the databases.

JPG to PDF Converter For Mac - Unregis

→ To represent this primary key generation algorithm hibernate s/w has provided a separate predefined class that is org.hibernate.id.Increament with the short name Increament Generator

ex:
```
<id name="eid" column="eid">
<generator class="Increament"/>
</id>
```

2→ In hibernate applications, if we configure a particular primary key generation algorithm in mapping file, at the same time if we provide another primary key value explicitly then hibernate s/w will give highest priority to the respective primary key generation algorithm provided primary key value. That is hibernate s/w will store primary key generation algorithm provided value onto the database table instead of our explicit primary key value.

JPG to PDF Converter For Mac - Unregis

Sequence:

Date: 23rd apr, 2012, Mon

This algorithm will generate primary key value on the basis of the sequence value provided by underlying database value.

— sequence generator primary key generation algorithm will generate a new primary key value with the following calculation

PK_value = max(PK_value column) + sequence_value

— This primary key generation algorithm will generate the primary key values of the data types short, int and long.

— To represent this primary key generation algorithm hibernate s/w has provided a separate predefined class that is org.hibernate.id.SequenceGenerator with the short name sequence.

— This primary key generation algorithm could be supported by almost all the databases which are supporting sequences like DB2, oracle, postgre sql and so on...

— This primary key generation algorithm should require an input parameter that is sequence.

In Database

```
sql> create sequence my_sequence increament
                                  by 10;
sql> commit;
```

In mapping file

```
<id name="eno" column="eno">
<generator class="sequence">
<param name="sequence">my_seq </param>
</generator> </id>
```

JPG to PDF Converter For Mac - Unregis

note; If we have not provided sequence input parameter to sequence primary key generation algorithm then hibernate software will search for "hibernate_sequence" as a sequence name by default in the underlying database in order to generate primary key values.

In DBA

JPG to PDF Converter For Mac - Unregis

Identity ~~pk~~ primary key generation algorithm:

mysql:> create table empl(eno int primary key auto_increment, ename char(10), salary float);
>commit;
>desc empl;

in

<pes
<property name="connection.driver_class">
com.mysql.jdbc.Driver </property>

jdbc:mysql://localhost:3306/dungadb

14/sep
Wed

check mysql connector in hibernate.jar

JPG to PDF Converter For Mac - Unregis

This primary key generation algorithm will generate primary key value on the basis of the database table identity column.

— Identity column is a primary key column and should have auto increament nature.

This primary key generation algorithm will make the underlying database to generate primary key value with the identity column that is by increamenting '1' on the maximum value of the primary key column.

— This primary key generation algorithm will provide the primary key values of the data types short, int and long.

— To represent this primary key generation algorithm hibernate software has provided a separate predefined class that is org.hibernate.id.IdentityGenerator with the short name identity.

— To prepare primary key values this algorithm should not require any input parameters.

— This primary key generation algorithm will be supported by almost all the databases which are supporting identity column like DB2, mysql, sqlserver, hypersonic sql and so on——

— In mysql database

```
mysql> create table emp1(eno int(5) primary key
       auto increament, ename char(10),
                        esal float);
mysql> commit;
```

JPG to PDF Converter For Mac - Unregis

In mapping file

<id name="ord" column="ord">
<generator class="org.hibernate.id.IdentityGenerator">
</generator>
</id>

— What is the diff b/w increament and identity primary key generation algorithms.

Expl: Both the primary key generation algorithms will prepare primary key value by increamenting one on the maximum value of primary key column.

Increament primary key generation algorithm will make hibernate software to generate primary value by getting maximum value from primary key column and by increamenting it with one value.

— Identity primary key generation algorithm will make the underlying database to prepare primary key value on the basis of identity column in the respective database table. That is by getting maximum value of primary key column and by increamenting it with one value.

JPG to PDF Converter For Mac - Unregis

Native primary key generation algorithm

```
mysql> create table empl(eno int(5), ename char(5),
                         esal float);
mysql> commit;
mysql> create database designdb;
— exception caused.
now
mysql> create table empl(eno int(5) primary key
  auto increament, ename char(10), esal float);
  >commit;
```

JPG to PDF Converter For Mac - Unregis

- This primary key generation algorithm will not have its own mechanism to generate primary key value, it will select either sequence or identity or hilo on the basis of the underlying database.
- If we use oracle as underlying database, then native algorithm will select sequence primary key generation algorithm. If the underlying database is mysql's then native algorithm will select identity primary key generation algorithm.
- If any underlying database is not supporting both sequence and identity primary key generation algorithms then native algorithm will select hilo primary key generation algorithm.
- This primary key generation algorithm should not have any separate predefined class provided by hibernate software but it should have a short name that is native
- This primary key generation algorithm will take input parameters on the basis of the internal selected primary key generation algorithm.
- This algorithm will generate primary key value of the data types short, integer and long.

ex: In mapping file

```
<id name="eno" column="eno">
  <generator class="native">
  </generator>
</id>.
```

JPG to PDF Converter For Mac - Unregis

This primary key generation algorithm will be supported by almost all the databases which are supporting either sequence or identity or native primary key generation algorithms.

hi-lo algorithm [high-low]

Date: 24th Apr, 2012 Tue

This primary key generation algorithm will generate on the basis of high value from a global resource (a separate table in the database) and low value from max_lo ~~property~~ (parameter) from mapping file.

This primary key generation algorithm will generate the primary key values of datatypes like short, int and long.

— This primary key generation algorithm will be supported by allmost all the databases. To refer this primary key generation algorithm hibernate software has provided a separate predefined class that is org.hibernate.id.TableHiLoGenerator with the short name hilo

— This primary key generation algorithm should require the following 3 parameters

1. table :- It will take a table name acting as global resource.
2. column :- It will take a column name in the global resource table to provide ~~max~~ high value.
3. max_lo :- It will take low value.

JPG to PDF Converter For Mac - Unregis

SQL> create table my table (my column number);
SQL> insert into my table values(100);
SQL> commit;

In mapping file :-

```
<hibernate-mapping>
  ==
  <id name="eno" column="eno">
    <generator class="hilo">
<param name="table">my_table</param>
<param name="column">my_column</param>
<param name="max_lo">10</param>
</generator>
</id>
  ==
```

Note: In the above primary key generator configuration if we have not provide table and column parameters then hibernate software will search for hibernate_unique_key table as global resource and next_high as column to get high value.

JPG to PDF Converter For Mac - Unregis

Seqhilo [sequence hilo] primary key gen. algorithm

This primary key generation algorithm is almost all same as hilo algorithm. But in case of seqhilo primary key generation algorithm, we will use a sequence provided by the underlying database in place of global resource table to get high value.

- This primary key generation algorithm will generate the primary key values on the basis of the underlying database provided and max_parameter value which we configured in mapping file.
- This primary key generation algorithm will generate the primary key values of the datatypes short, int and long.
- This primary key generation algorithm will be supported by all most all the databases which are supporting sequences.
- To refer this primary key generation algorithm hibernate f/w has provided a separate predefined class that is org.hibernate.id.SequenceHiloGenerator, with the short name seqhilo.

In database

```
SQL> create sequence my_sequence increament by 5;
SQL> Commit;
```

JPG to PDF Converter For Mac - Unregis

- This primary key generation algorithm only take two parameters

1. sequence → To provide hi value
2. max_lo → To provide lo value

In mapping file

```
<class name="employee" table="emp">
  <id name="eno" column="eno">
    <generator class="seqhilo">
<param name="sequence">my_sequence</param>
<param name="max_lo">10</param>
</generator>
</id>
```

JPG to PDF Converter For Mac - Unregis

uuid (universal unique id) primary key gen. algorithm

This primary key generation algorithm will generate primary key value on the basis of system IP address, JVM startup time, machine registers counters.

- This primary key generation algorithm will generate the primary key values of the datatype string.
- This primary key generation algorithm will be supported by almost all the databases. This algorithm should not require any input parameter explicitly.
- This primary key generation algorithm will be secured by using predefined class provided by hibernate software, that is org.hibernate.id.UUIDHexGenerator, with the short name "uuid"
- If we want to use this primary key generation algorithm in hibernate applications then the underlying database table should maintain primary key column with the length 32 characters

(or) In mapping file

```
<id name="eid" column="eid">
  <generator class="uuid">
  </generator>
```

Cast in class file eid as String

GUID

Date: 25th Apr, 2012 Wed

This primary key generation algorithm will not have its own mechanism to generate primary key value, it will make the underlying database to generate primary key value on the basis of the identity column, available in the respective database table.

— This primary key generation algorithm will generate the primary key values of the datatype String.

— UUID primary key generation algorithm will generate the primary key value of length 32 characters but GUID primary key generation algorithm will generate the primary key value of length 16 characters.

— In hibernate applications, identity primary key generation algorithm will generate the primary key value of the data types short, int and long on the basis of the identity column available in the database table but GUID primary key generation algorithm will generate the primary key of data type String on the basis of the identity column available in the respective database table.

— This primary key generation algorithm will be supported by MySQL and Microsoft's SQL Server

— To refer this primary key generation algorithm hibernate software has provided a separate predefined class like org.hibernate.id.GUIDGenerator with the short name guid

JPG to PDF Converter For Mac - Unregis

This primary key generation algorithm should not require any input parameter.

```
<id name="id" column="cid">
<generator class="guid"/>
</id>
```

JPG to PDF Converter For Mac - Unregis

Select primary key generation algorithm

- This primary key generation algorithm will generate the primary key values on the basis of the database provided triggers.
- This primary key generation algorithm will generate the primary key values of the datatype short, int and long.
- This primary key generation algorithm will be supported by allmost all the databases which are supporting triggers internally.
- To create this primary key generation algorithm hibernate software has provided a separate predefined class i.e org.hibernate.select Generator with the short name "select".

JPG to PDF Converter For Mac - Unregis

Foreign primary key generation algorithm

- This primary key generation algorithm will generate the primary key values on the basis of the associated table primary key column.
- To generate primary key values with this mechanism we should require one-to-one mapping between two database tables.
- This primary key generation algorithm will generate the primary key values of the datatype short, int and long.
- This primary key generation algorithm will be supported by almost all the databases which are supporting associations like one-to-one, one-many, many-one and many-many.
- To refer this primary key generation algorithm hibernate software has provided a separate predefined class like org.hibernate.id.ForeignGenerator with the short name "foreign"

JPG to PDF Converter For Mac - Unregis

Composite key primary key generation algorithm

In general in database applications, we will declare a particular column as primary key to refer all the records independently.

- As per the application requirements, one column may not sufficient to represent all the records independently, where it is required to provide more than one column as primary key.
- To declare more than one column combination as primary key we have to use composite keys.

- Syntax:

```
<composite-id>
  <key-property name="" column="_">
  ==
  </composite-id>
```

In the database

```
SQL> create table empl(eno number, ename varchar2(30),
     esal number, primary key(eno, ename));
SQL> commit;
```

JPG to PDF Converter For Mac - Unregis

In mapping file:

```
<hibernate-mapping>
<class name="Employee" table="emp1">
<composite-id>
<key-property name="eid"/>
<key-property name="ename"/>
</composite-id>
</hibernate-mapping>
```

— If we want to use composite keys in our hibernate applications then the respective POJO class must implement Serializable interface.

JPG to PDF Converter For Mac - Unregis

- In hibernate application if we want to use primary key generation algorithms to provide primary key values then we have to configure the respective generator class in the mapping file.

=> In general in hibernate applications we will use annotations to provide metadata to the hibernate software as an alternative to mapping file. In the above context to provide primary key generation algorithms through annotations we will use the following annotation provided by EJB-3.

Date: 26th Apr, 2012 Thur

```
@GeneratedValue(strategy:__, generator:"__")
```

where strategy parameter will take a particular constant from GenerationType enum. where GenerationType enum has provided the constants like AUTO, IDENTITY, SEQUENCE, TABLE.

where AUTO constant will represent native primary key generation algorithm, it will select either sequence or identity or hilo on the basis of the underlying database.

where IDENTITY constant will represent identity primary key generation algorithm, where SEQUENCE constant will represent sequence primary key generation algorithm.

where TABLE constant will represent hilo algorithm

JPG to PDF Converter For Mac - Unregis

exple
Question

```
import javax.persistance.*;
@Entity
@Table(name="empl")
public class Employee
    {   @Id
@GeneratedValues(strategy=GenerationType.
                                IDENTITY)
@Column(name="eno")
    private int eno;
@Column(name="ename")
    private String ename;
@Column(name="esal")
    private float esal;
    public void setEno(int eno)
            { this.eno=eno;
            }
    public int getEno()
            { return eno;
            }
    public void setEname(String ename)
            { this.ename=ename;
            }
    public String getEname()
            { return ename;
            }
```

Here we are that algorithm acts on my sql.

JPG to PDF Converter For Mac - Unregis

```
public void setEsal(float esal)
        {
            this.esal=esal;
        }
public float getEsal()
        {
            return esal;
        }
}
```

SQL> desc emp1;

Name
eno
ename
esal

SQL> create sequence my_sequence increamented by 5;

2 → If we want to use sequence primary key generation algorithm in our hibernate applications by using annotations then we have to use the following annotation

@SequenceGenerator(name="__", sequenceName="__");

where name parameter will take logical name of the sequence generator.

Sring letter

— where sequenceName parameter will take a sequence name provided by the underlying database.

— After preparing sequence generator we have to pass the logical name of sequence generator to generatedValue annotation

JPG to PDF Converter For Mac - Unregis

example

pgm

```
import javax.persistance.*;
@Entity
@Table(name="empt")
public class Employee
{ @Id
@SequenceGenerator(name="seqgen",
                   sequenceName="my_sequence")
@GeneratedValue(strategy=GenerationType.
    SEQUENCE, generator="seqgen")
@Column name="eno"
  private int eno;
@Column name="ename"
  private String ename;
@Column name="esal"
  private float esal;
public void setEno(int eno)
    { this.eno=eno;
    }
public int getEno()
    { return eno;
    }
public void setEname(String ename)
    { this.ename=ename;
    }
public String getEname()
    { return ename;
    }
public void setEsal(float esal)
    { this.esal=esal;
    }
```

JPG to PDF Converter For Mac - Unregis

```
public float getEsal()
    {
        return esal;
    }
}
```

JPG to PDF Converter For Mac - Unregis

SQL> create table my_table(id number, next_id number)

SQL> insert into my_table (10,100);

SQL> commit;

— If we want to use table (hilo) primary key generation algorithm in our hibernate applications by using annotations then EJB3 ~~API~~ key provided the following annotation to configure global resource table.

select * from api; select * from my_table 10 101

@TableGenerator(name="_", table="_", pkColumnName="ID", pkColumnValue="b", valueColumnName="NextID")

my_table

Id	Next_Id
10	100

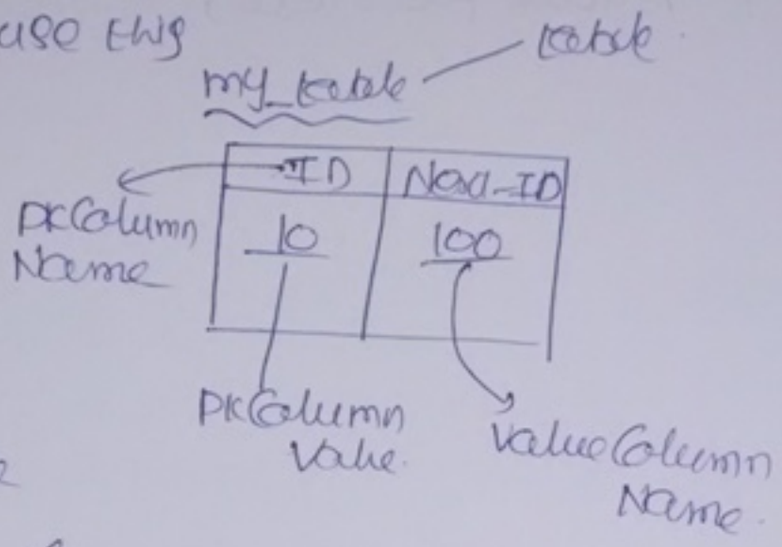
- where name parameter will take logical name of the table generator,
- where table parameter name will take global resource table name.
- where pkColumnName will take primary key column name of the global resource table.
- where pkColumnValue parameter will take the initial value of the primary key column in the global resource table.
- where valueColumnName parameter will take the column name in the global resource

JPG to PDF Converter For Mac - Unregis

table generate primary key value.

— After getting TableGenerator, we need to pass its logical name to generatedValueAnnotation.

If we want to use this annotation in hibernate application we ~~have~~ to ~~u~~ should require the following table




```
@GeneratedValue(strategy=GenerationType.TABLE,
                generator='tablegen')
```

Example

```
import javax.persistence.*;
@entity
@Table(name="empl")

public class Employee
{
    @Id
@TableGenerator(name="tablegen", table="my_table",
pkColumnName="id", pkColumnValue="10",
                valueColumnName="next_id")
@GeneratedValue(strategy=GenerationType.TABLE,
                generator="tablegen")
    @Column(name="eno")
    private int eno;
```

JPG to PDF Converter For Mac - Unregis

```
@Column(name="ename")
private String ename;
@Column(name="esal")
private float esal;
```

JPG to PDF Converter For Mac - Unregis

2→ EJB3 has provided annotation support upto AUTO, IDENTITY, SEQUENCE and TABLE but in hibernate nearly 11 number of primary key generation algorithms are available. To represent the remaining primary key generation algorithm hibernate has provided the following annotation.

@GenericGenerator(name="_", strategy="_", parameters=_)

where

name parameter will take logical name of the generator. where strategy parameter will take short name of the primary key generation algorithm. where parameters will take array of parameter annotations which are required to the respective primary key generation algorithm.

— To represent a particular parameter hibernate has provided the following annotation

@parameter(name="_", value="_")

After preparing GenericGenerator, we have to pass its logical name to GeneratedValue annotation

@GeneratedValue(generator="_")

JPG to PDF Converter For Mac - Unregis

example
pkm

```
import javax.persistance.*;
// import org.hibernate.
@Entity
@Table(name="emp11)
public class Employee
{ @Id
@org.hibernate.annotations.GenericGenerator
(name="gen" strategy="seqhilo", parameters=
{ @org.hibernate.annotations.parameter
(name="sequence", value="mysequence"),
@org.hibernate.annotations.parameter
(name="max_lo", value="10")})
@GeneratedValue(generator="gen11)
@Column(name="eno11)
private int eno;
@Column(name="ename")
private String ename;
@Column(name="esal")
private float esal;
```

JPG to PDF Converter For Mac - Unregis

Hibernate Tools

Date: 27th Apr, 2012 Fri

In general in hibernate applications before going to execute client application that is before performing persistance operations first we have to check whether the respective database tables and data is existed or not, if it is not existed then we have to prepare the required database tables and data in the tables, before performing persistance operations.

If we have any requirement to work with unknown databases from hibernate applications it is very difficult to prepare the required database tables in database before hibernate application execution.

— In the above context to prepare the required database in unknown databases before hibernate application execution hibernate s/w has provided some inbuilt tools

1. schema export
2. schema update
3. code generator.

— In general in hibernate applications, we may perform persistance operations like insert, update, delete and so on dml operations.

— If we want to perform ddl operations like create, alter and drop, we have to use the above specified hibernate tools.

JPG to PDF Converter For Mac - Unregis

Schema export: The main purpose of Schema export tool is to prepare a new table in database as per configuration file, mapping file and the respective POJO class irrespective of the table existancy.

2) To represent schema export tool hibernate s/w has provided the following predefined class in hibernate.jar file ✓

org.hibernate.tool.hbm2ddl.SchemaExport

2) To execute Schema Export tool we have to use the following command on command prompt from the location where configuration file is available

```
d:|hbm4|app1> java org.hibernate.tool.
hbm2ddl.SchemaExport --config=
hibernate.cfg.xml
```

If we use the above command on command prompt then schema export tool will be activated when performing the following action

① Schema export tool will get config input parameter value that is hibernate configuration file name.

② SchemaExport tool will recognize the specified hibernate configuration file, from where it will recognize mapping file.

③ From hibernate mapping file SchemaExport tool will get mapped table name, mapped column names and their datatypes as per the specified POJO class

JPG to PDF Converter For Mac - Unregis

— After getting the complete details of the respective table schema export tool will check whether any table is existed in database with the same name or not. If schema export tool identified any table with the same name in database then it will drop that table and create a new table as per the table description what it identified in the respective mapping file.

— If SchemaExport tool is not identified any table with the same name then it will create a new table in database as per the respective mapping file.

JPG to PDF Converter For Mac - Unregis

Schema Update: The main purpose of Schema Update tool is either to create a new table in database or to alter the existed table in database.

— To represent schema update tool hibernate software has provided the following predefined class in hibernate3.jar file.

org.hibernate.tool.hbm2ddl.SchemaUpdate

To create SchemaUpdate tool we will use the following command on command prompt from the location where hibernate-configuration file is available

d:\hbn4\app1 > java org.hibernate.tool.hbm2ddl.SchemaUpdate --config=hibernate.cfg.cmy

— If we use the above command on command prompt then schemaUpdate tool will be activated then perform the following actions.

① SchemaUpdate tool will pick up config input parameter value that is hibernate configuration file.

② SchemaUpdate tool will recognize hibernate configuration file, where SchemaUpdate tool will recognize hibernate mapping file.

In hibernate mapping file, schema update tool will get a mapped table name, mapped column names and their datatypes as per pojo class properties.

JPG to PDF Converter For Mac - Unregis

(ii) After getting the complete mapped table description from mapping file schema update tool will check whether any table exists with in the database with the same name or not.

If schema update tool identify any existed table in database with the same name then schemaupdate tool will check whether any new columns descriptions is available in the mapping file or not when compared with the database table columns.

— If schemaupdate tool identify new columns description in mapping file then only it will perform alter operation on the existed table.

2→ If schemaupdate tool is not identify new columns description in mapping file then it will not perform alter operation on the existed table.

→ If schema update tool is not identify any table in the database with the same name then schema update tool will create a new table in database as per the table description identified in the respective mapping file.

JPG to PDF Converter For Mac - Unregis

Code generation:- The main purpose of this tool is to generate a pojo class on the basis of the respective database table, mapping file and configuration file.

2) code generation tool is available with hibernate upto hibernate 2.x version, it was deprecated now in hibernate 3.x version.

JPG to PDF Converter For Mac - Unregis

My Eclipse IDE

Date: 29th apr, 2012 Sun

Steps to design J2SE application with My Eclipse IDE:-

Step①: create java project.

File → new → java project.

If we do the above a window will be open to take project details

Project Name: [firstapp]

Contents:

- ⊙ create new project in work space.
- ○ create project from existing source

JRE

- ⊙ use an execution environment JRE: [JavaSE-1.6] [Δ]
- ○ use project specific JRE: [Sun Jdk 1.6.0.13]
- ○ use default JRE (current sun JDK 1.6.0.13)

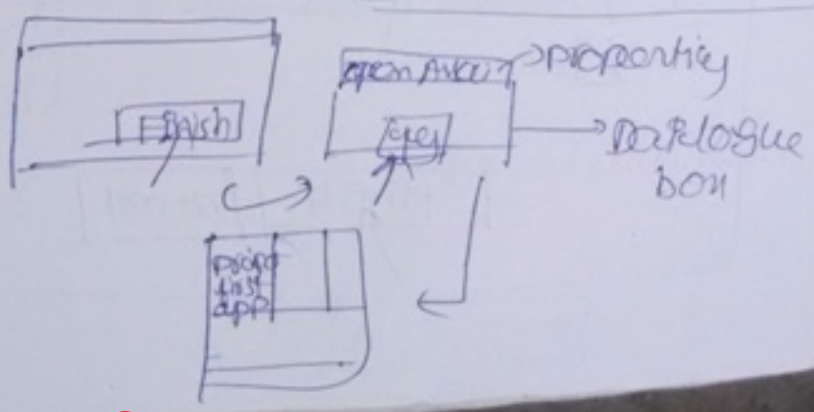
Project layout

- ○ use project folder as root for classes and sources files.
- ⊙ create separate folders for sources and class files.

Working set

☐ Add projects to working sets.

[Next] [Finish] [Cancel]



JPG to PDF Converter For Mac - Unregis

— If we do the above steps then a new project (firstapp) will be created on the left hand side of the MyEclipse IDE.

Step②: create a java class with main method.

Right click on firstapp → new → class.

If you do the above a window will be created to take class details.

Source folder: [firstapp/src] [browse]

package: [com.dss] [browse]

enclosing type

Name: [First]

Modifiers ⊙ public ○ default ○ private ○ protected

☐ abstract ☐ final ☐ static

Super class: [java.lang.Object] [browse]

by default

Interfaces: [ ] [Add]

Which method stubs you have to create

☐ public static void main(String[] args)

☐

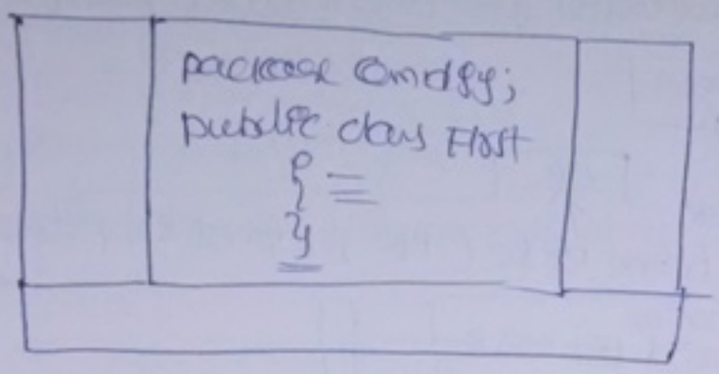
☐

Do you want to comment?

☐

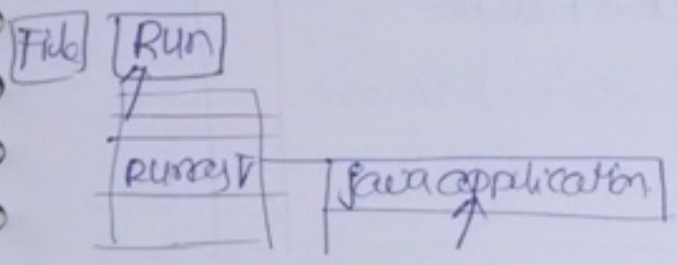
[Finish] [cancel]

JPG to PDF Converter For Mac - Unregis

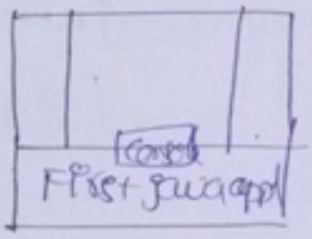


If we do the above step a java file will be created in the middle frame of MyEclipse IDE, where we have to provide our application logic as per the requirement.

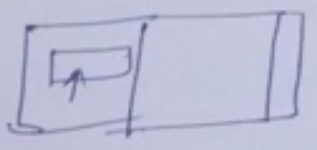
③ Run the java application.



If we do this step, automatically java application will be executed and the required o/p will be generated on console.



Note: If we want to design J2EE applications with MyEclipse IDE then we have to use the same steps which are provided in the above, but we need to add driver jar file to the project library. For this we have to use the following path

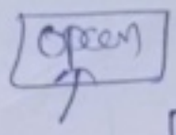


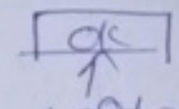
Right click on project
↳ properties
↓
Java build path

Libraries

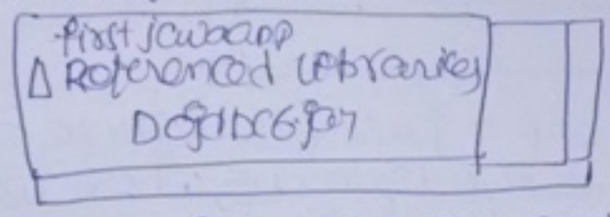
Add external JARs

Select the required jar file ojdbc6.jar (or) ojdbc5.jar

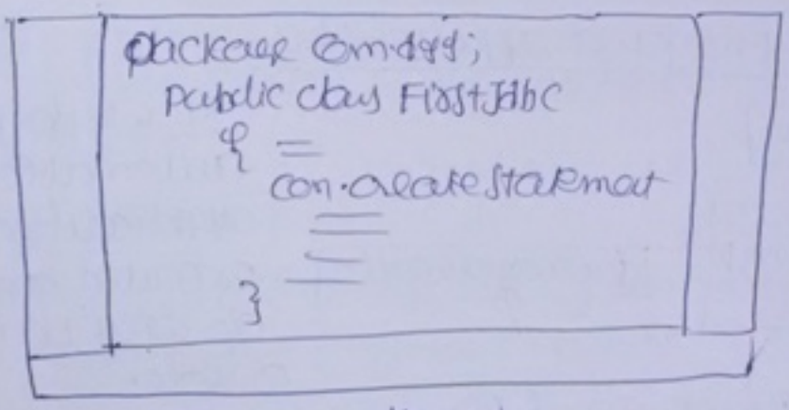




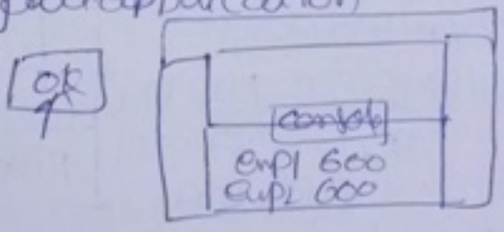
Now, if we click on the left hand side (i.e project explorer)



→ Type the JDBC program on the body part of the IDE



Run → Runas → Java application



JPG to PDF Converter For Mac - Unregis

Steps to design Hibernate application with MyEclipse IDE

Step①: create Java project

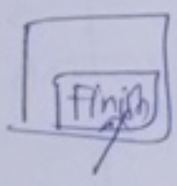
File → New → Java project.

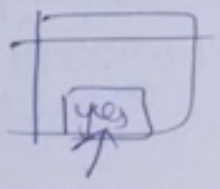
Project Name : HibernateApp1

Contents:

- ◉ create new project in workspace
- ○
- JRE
- ◉ use an execution environment JRE [JavaSE-1.6]
- ○
- ○
- Project layout
- ○
- ◉ create separate folders for classes and sources.

[next]





Step②: Add hibernate capabilities:-

Right click on project (HibernateApp1)

MyEclipse → Add Hibernate capabilities

If we do this step then a window will be open to take hibernate settings.

Hibernate support for myeclipse

Hibernate specification: ○ Hib1 ○ Hib2 ○ Hib3 | — ◉ Hib3

[] enable Hibernate Annotation support

---

select libraries to add to the build path

show MyEclipse libraries [] user libraries

JPG to PDF Converter For Mac - Unregis

[✓] Hibernate 3 annotations & entity manager
[✓] Hibernate 3.3 core libraries
[ ]

Jar library installation
(•) Add checked libraries to project build path
( ) copy checked library jars to project folder and add to build path.

[Next]

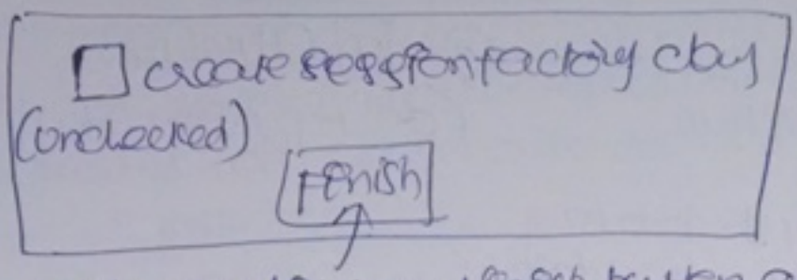
Hibernate support for myeclipse
Hibernate config file (•) new
configuration folder: [src]
configuration file name: [hibernate.cfg.xml]
[✓] open configuration file after wizard completion.

[next]

Hibernate support for Myeclipse
[✓] specify database connection details
Datasource: (•) use JDBC driver ( ) use JNDI Data source
DB Driver: [ ]
Connect URL: [jdbc:oracle:thin@localhost:1521:xe]
Driver class: [oracle.jdbc.driver.OracleDriver]
User name: [system]
password: [durga]
Dialect: [Oracle 9/10g]
[ ] Enable dynamic table creation
[Next]

JPG to PDF Converter For Mac - Unregis

Uncheck the create session factory class

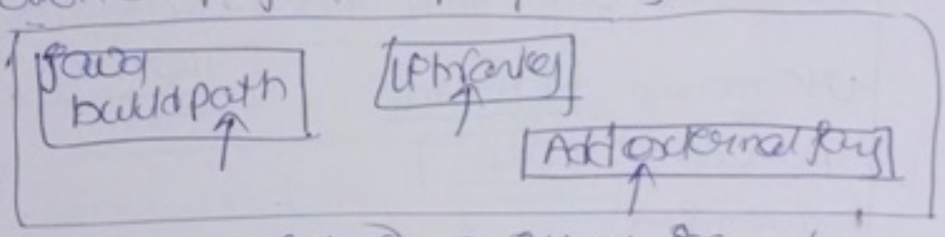


→ If we click on finish button automatically hibernate.cfg.cmd file will be created in src folder and it will be open in middle frame of my Eclipse IDE.

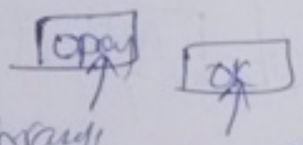
Note: with the above steps, all the hibernate specific jar files will be added to project library. To execute hibernate application we have to add driver specific jar file to project library.

Add DSP

Right click on project → properties



select ojdbc6 or ojdbc5 or ojdbc14.jar file



Now, in referenced libraries we will see the respective jar file.

JPG to PDF Converter For Mac - Unregis

step no:③: open database explorer:

[Window] (menu) ← ⓐ [Hyperlipse]. (on top right side)

↳ open perspective

↓

Hyperlipse database explorer ✓

If we do the above step automatically db browser will be open in left side frame (project explorer place)

— step④: prepare database ✓

Rightclick on Db browser → new.

If we do the above step automatically a window will be open to take the database connection details.

New database connection Driver.

Driver Template: oracle(thindriver)

Driver name: [oracledb]

Connection URL: [jdbc:oracle:thin@localhost:1521:xe]

Username: [system]

password: [dba]

Driver Jars:

[C:\oracle\ojdbc14.jar]

[Add Jar]

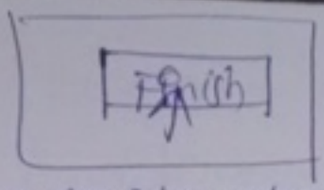
select ojdbc6 ⓐ ojdbc5.jar ⓑ ojdbc14.jar

[open]

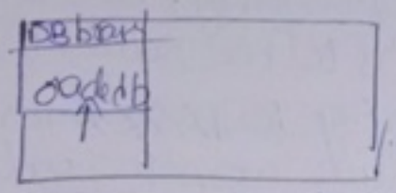
Driver Classnae: [oracle.jdbc.driver.OracleDriver]

[Next]

JPG to PDF Converter For Mac - Unregis

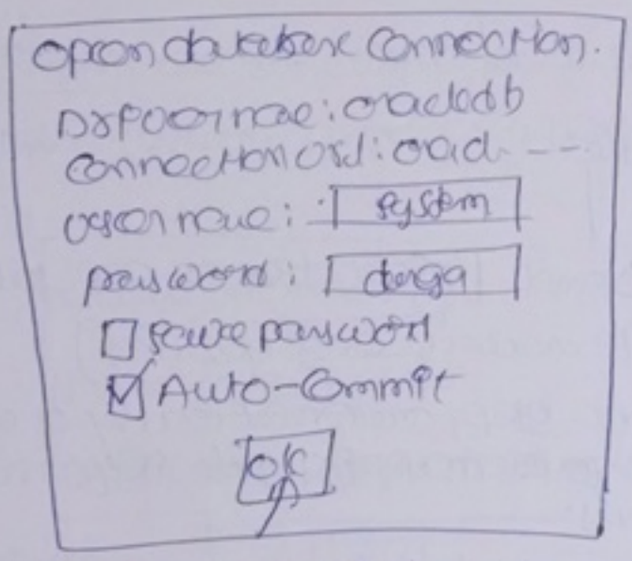


If we do the above step database icon will be created with the name oracledb in db browser.



Right click on oracledb → open Connection

If we do the above,
a window will be open to take database username and password.



Now, elaborate oracledb → oracledb connected to oracledb

elaborate connected to oracledb → system

Select system

Right click on system → new table

If we do the above a window will be created to take table information.

JPG to PDF Converter For Mac - Unregis

New database table

Table Name: [emp]

— (select columns)

by click on add button repeatedly, we are able to add number of columns to the table.

perform reverse engineering to ~~perform~~ create pojo class and the respective mapping file.

— expand oracledb.

↑ oracledb

↑ connected to oracledb

↑ ssystem

↑ table

Demp [Right click on emp table]

(select Hibernate reverse engineering)

— If we do the above step, automatically a wizard will be created to take mapping file information and pojo class details.

Hibernate mapping and Application generation

Java src folder: [ ] [Browse]

Java package: [com.dfg]

☑ create pojo ⇔ DB table mapping file

⊙ create a Hibernate mapping file

○ Add Hibernate mapping annotations to pojo

☑ update Hibernate configuration with mapping resource

JPG to PDF Converter For Mac - Unregis

☑ Java bean object (POJO ↔ DB table)

☐ create abstract class

Base persistent class: [ ] [ ]

☐ Java data Access object (DAO)

[next]

[next] [Finish] [Run in background]

→ If we do the above empl.java, empl.hbm.cfg.cfg will be created under the specified package structure ✓

2 → Step (6): create client application ✓

Right click on project → new → class

package: [com.dtg]

Name: [ClientApp]

☑ public static void main(String[])

[Finish]

After execution see the db

↳ empl

↳ edit data

[Apply]

```
class ClientAP
{
  ==
  PS Um(S[]a)
  {
    Configuration cfg = —
    ==
```

window

↳ DB browse

↳ SQL.

Run

↳ eclipse

↳ Java application ✓

JPG to PDF Converter For Mac - Unregis

Steps to design Servlets & JSP (web application) with Myeclipse IDE

Step①: prepare web project

File → new → webproject ✓

If we do the above a window will be created to take web project details.

project name: [webapp1]
Location: ☑ use default location
Source folder: [src]
web root folder: [webroot]
Context root URL: [/webapp1]
J2ee specification level
◉ J2ee 5.0 ○ J2ee 1.4 ○ J2ee 1.3
Maven
☐ Add maven support
JSTL support ☐ Add JSTL libraries to WEB-INF folder
[Finish]

Dialog box
Incompatible
[Yes]

Step②: prepare web resources as per the application requirement

→ To prepare a jsp page or html we will use the following.

Right click on project

New → HTML (Advanced Template)

JPG to PDF Converter For Mac - Unregis

If we do the above then a window will be open to take html page details.

Html wizard.

File path: /webapp1/webRoot

File name: loginform.html

DOCTYPE: HTML 4.01 Transcriptional

Template to use: HTML template with a form.

Finish

— If we do the above step loginform.html will be created under webroot and it will be open in the middle frame of my eclipse. (Now type the data)

— To prepare jsp pages we have to use the following

Right click on project

↳ new → jsp (advanced Templates)

If we do the above a window will be open to take jsp file informations.

jsp wizard

File path: /webapp1/webroot Browse

File name: success.jsp

Template to use: Default JSP Template

Finish

JPG to PDF Converter For Mac - Unregis

— To prepare a servlet, we have to use the following path

Right click on project → new → servlet

If we do the above a window will be open to take servlets details

Servlet wizard

source folder: [webapp1/src]

package: [com.dsg]

Name: [LoginServlet]

◉ public

Super class: [javax.servlet.http.HttpServlet]

Template to use: [polaris Template for servlet]

which method stubs would like to create?

☐ inherited abstract methods

☐ constructors from superclass

☐ init & destroy ☐ doDelete() ☐ doGet

☑ doPost

[next]

servlet wizard

☑ generate/Map web.xml file

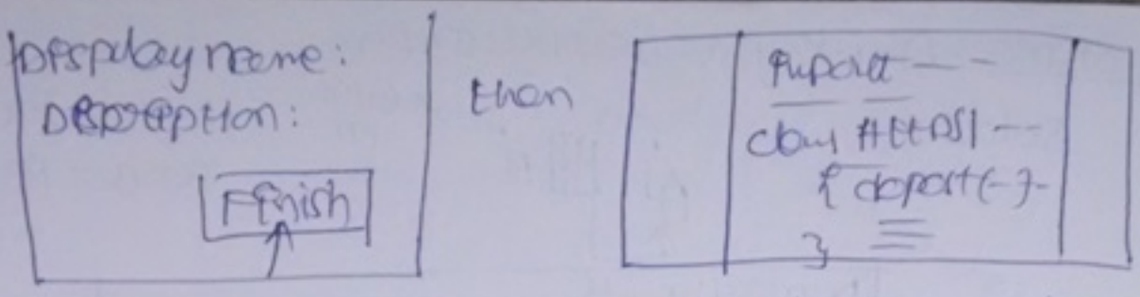
servlet/JSP class name: [com.dsg.LoginServlet]

servlet/JSP Name: [LoginServlet]

Servlet/JSP Mapping URL: [/login]

File path of web.xml [/webapp1/webRoot/WEB-INF]

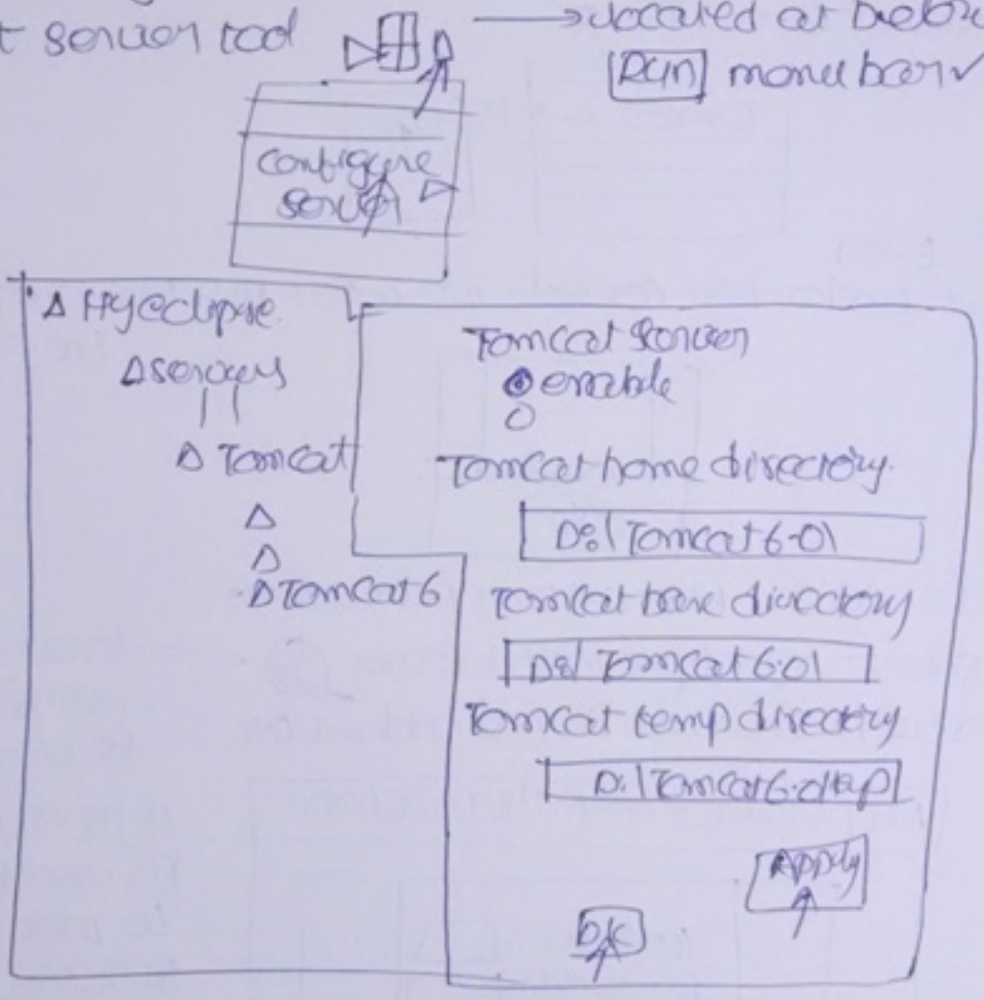
JPG to PDF Converter For Mac - Unregis



If we do the above steps LoginServlet.java file will be created under source folder, where we have to open LoginServlet.java and provide the required application logic ✓

Step③: configure server to Myeclipse IDE.

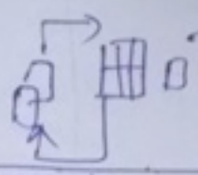
select server tool → located at below Run menu bar ✓



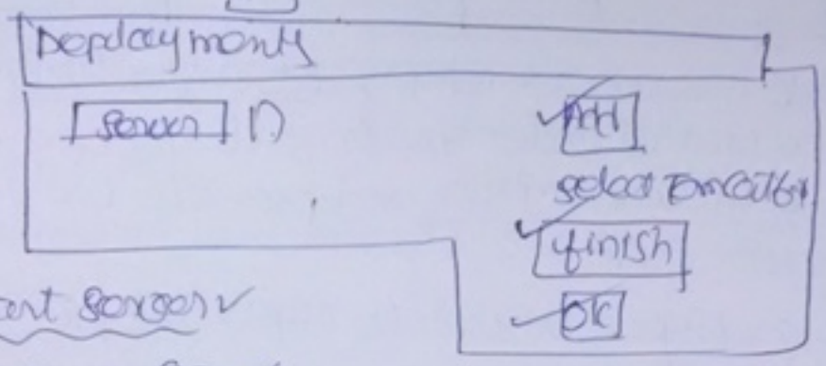
JPG to PDF Converter For Mac - Unregis

Step④: Deploy web application.

Select

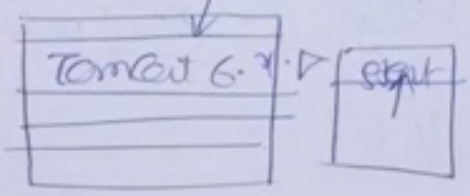


located at → left tool bar of server icon.



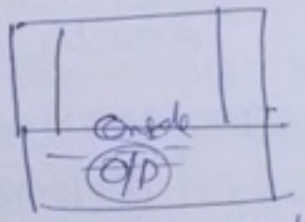
Step⑤: Start server ✓

— select server icon



then

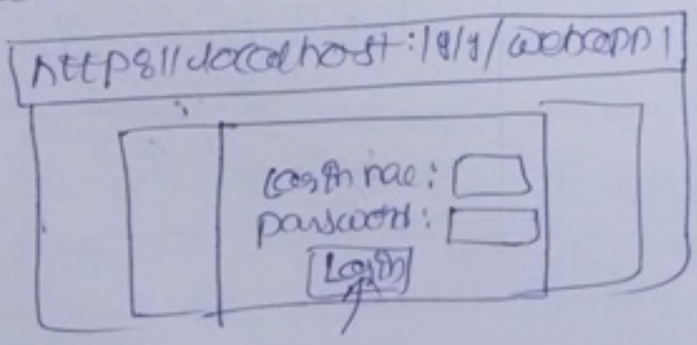
at under the console, we will see the output of the pgm.



Step⑥: Access the web application.

select myeclipse web browser → located at right side of server icon.

provide the url in the address bar.



Myeclipse has its own browser to provide the services to the user.

JPG to PDF Converter For Mac - Unregis

date: 30th Apr, 2012 (Mon)

In hibernate applications
it is possible to activate all the hibernate tools through declarative approach also.

— To declare hibernate tools in hibernate applications we have to use the following property in any configuration file.

— hibernate.hbm2ddl.auto

ex:

```
<hibernate-configuration>
    <session-factory>
        ==
<property name="hibernate.hbm2ddl.auto">
        ==  value </property>
    </session-factory>
</hibernate-configuration>
```

— In hibernate configuration file, we are able to provide the following 3 values to hibernate.hbm2ddl.auto property.

① create
② update
③ create-drop
↑
hiphen

create:
if we provide create as a value to hibernate.hbm2ddl.auto property in configuration file then schema export tool will be accurated, it will create a new table on database irrespective of the table existancy. that is, it will

JPG to PDF Converter For Mac - Unregis

create a new table in database if table is not existed, if any table is existed with the same name then this tool will drop that existed table and create a new table as per mapping file configurations.

2 ⇒ update:

If we provide update is a value to hibernate.hbm2ddl.auto property then hibernate software will activate schema update tool internally.

In hibernate applications, schema update tool will create a table on database as per the mapping file configuration, if the table is not existed in database.

Schema update tool will alter the table when it identify new columns description in the respective mapping file when compared with the existed table. If schema update tool is not identify new columns description in the mapping file then it will not alter the existed table.

create-drop:- If we provide create-drop as value to hibernate.hbm2ddl.auto property then hibernate software will create table as per the mapping file at the time of creating session factory object and hibernate software will drop the table as per the mapping file when session-factory object is closed.

JPG to PDF Converter For Mac - Unregis

Bulk operations :-

Date: 1st May, 2012 Tue

In general in hibernate applications to perform persistance operations we will use the methods like save, persist, saveOrUpdate, update and so on---.

— In hibernate applications by using the above methods we are able to perform persistance operations with single record only.

— Single record manipulations are not sufficient to achieve client requirements in hibernate applications, here to achieve client requirements we have to perform persistence operations with multiple records.

— To perform multiple records manipulations in hibernate applications we will use the following alternatives.

1. HQL [Hibernate Query Language]
2. native SQL
3. criteria Api

where HQL is an object oriented, database independent query language to perform bulk operations. But HQL is not sufficient in the requirements where to use the underlying database supported queries like accessing stored procedures and functions.

To overcome the above problem with HQL we will use native SQL.

JPG to PDF Converter For Mac - Unregis

The main purpose of native SQL is to allow the underlying database supported query language in hibernate applications directly.

Where the main purpose of criteria api is to provide the complete persistence logic by using Java methods, without using any query language including HQL.

HQL:-

① HQL is a powerful object oriented query language prepared by hibernate. HQL will perform manipulations with objects only where as SQL will perform manipulations with relations (tables).

② HQL is database independent query language where as SQL is database dependent query language.

③ In hibernate applications if we execute any HQL query then hibernate software will convert that HQL query into the database dependent SQL query. By the execution of the database dependent SQL queries only the persistence operations will be performed.

Due to the above reason, we will conclude that HQL is a query language designed on the basis of SQL.

④ In case of HQL, we will prepare all the queries on the basis of pojo class names and pojo class properties. But in case of SQL we will prepare

Why take 1/20 when 1/2 Convert HQL Query to db dependent query

JPG to PDF Converter For Mac - Unregis

all the queries on the basis of table names and column names.

⑤ HQL is case insensitive query language except pojo class names and property names.

⑥ HQL was designed on the basis of SQL so that HQL is able to support allmost all the syntaxes of SQL, (all the functions supported by SQL, all the keywords supported by SQL) and (the advanced features like associations, joins and so on supported by SQL).

⑦ HQL is an object oriented query language (but not structure oriented language) so that it able to support the object oriented features like polymorphism, data abstraction, data encapsulation and so on.

⑧ upto hibernate 2.x version HQL was implemented to perform only retrieval operations. But from Hibernate 3.x version HQL was implemented to perform both retrieval operations, insert operations, updations and so on--.

⑨ HQL is not suitable when we have a requirement to use database dependent queries.

exple: In general if we want to access stored procedures and functions defined at database from java applications we need to provide the database dependent procedure calls and function calls in java applications.

JPG to PDF Converter For Mac - Unregis

In this context, HQL is not suitable, for this we have to use native SQL.

— incase of JDBC, when we perform retrieval operations by using select SQL query then automatically all the results will be stored in the form of Resultset object, here Resultset object is not serializable by default. So that it is not possible to move Resultset object from one machine to another machine over the network in distributed applications. But in case of HQL when we perform retrieval operations then all the results will be retrieved in the form of collection objects, here all the collection objects are by default serializable, so that is it very easy to carry collection objects from one machine to another machine over the network.

JPG to PDF Converter For Mac - Unregis

Date: 2nd May, 2012 Wed

If you want to use HQL, in our hibernate applications then we have to use the following steps.

① prepare query object
② provide customizations on result through query object
③ execute HQL query.

prepare query object: In hibernate applications to represent a particular HQL query, hibernate fw has provided the predefined interface in the form of org.hibernate.Query in hibernate3.jar file.

— To create query object we have to use the following method from Session.

```
public Query createQuery(String hql)
```

ex: Query q = s.createQuery("from Employee");

Step②: provide customizations on results through query object

To set FetchSize in order to retrieve the results, we have to use the following method from Query

```
public void setFetchSize(int size)
```

ex: q.setFetchSize(10);

Note: In our hibernate applications if 100 number of results can be generated with the provided HQL query then by using FetchSize(10) we are able to retrieve all 100 number of results in 10 attempts, every time 10 records.

JPG to PDF Converter For Mac - Unregis

— To set maximum number of result, we have to use the following method from Query

public void setMaxResults(int si); eg q.setMaxResults(10);

Note: In our hibernate application 100 number of records are matched with our provided HQL query then if we specify 10 as maxResult then we are able to retrieve first 10 number of records

— To specify a particular record as first record in the results then we have to use the following method from query

public void setFirstResult(int no)

eg q.setFirstResult(10);

— In our hibernate application, if 100 number of records are identified as a result of HQL query execution then if we specify first result is 10 and maximum results is 10, then we are able to retrieve the records from 10 to 20 over the result

— If we want to make our results as read only results then we have to use the following method from Query class

public void setReadOnly(boolean b)

If b==true, then the results are read only, else the results are not Read only.

eg q.setReadOnly(true);

JPG to PDF Converter For Mac - Unregis

— If we want to make our results as Cacheable results then we have to use the following method from Query.

```
public void setCacheable(boolean b)
```

If b==true, then the results are cacheable else the results are not cacheable.

Note: The main intention to make the results as Cacheable results is to provide reusability.

What is cacheable record?

— If we want to set a particular comment to the result, we have to use the following method

```
public void setComment(String comment)
```

ex: q.setComment("Employee Results");

— If we want to specify a particular idle time for the results we have to use the following method

```
public void setTimeOut(int time)
```

ex: q.setTimeOut(1000);

m.s

JPG to PDF Converter For Mac - Unregis

Step(3): execute HQL queries.

In hibernate applications, after preparing HQL queries we have to execute them, for this we have to use the following methods.

1. list
2. iterate()
3. scroll()
   ↳ small
4. uniqueResult()

— where list() method can be used to execute a specified HQL query and generate all the results in the form of List object.

public List list() throws HibernateException

— where iterate() method can be used to execute the specified HQL query and generate the results in the form of iterator.

public Iterator iterate() throws HibernateException.

— where scroll() method can be used to execute the specified HQL query and generate the results in the form of ScrollableResults.

public ScrollableResults scroll() throws HibernateException.

JPG to PDF Converter For Mac - Unregis

Note: where ScrollableResults is almost all equal to ScrollableResultSet, which can be used to iterate the results in both forward and backward directions.

But this ScrollableResults has provided the following methods

```
public void first()
public void last()
public void beforeFirst()
public void afterLast()
public boolean next()
public boolean previous()
```

— where uniqueResult() method can be used to execute HQL query and retrieve a single result in the form of Object.

If we use uniqueResult() method then it will check whether the result contains only one record or not, if only one record is available then it will return that record in the form of Object.

— If no records are identified then it will throws HibernateException. If this method identify more than one result then it will raise HibernateException.

```
public Object uniqueResult() throws HibernateException
```

JPG to PDF Converter For Mac - Unregis

Student.java

```
public class Student
{
private String sid;
private String sname;
private int smarks;
public void setSid(String sid)
        { this.sid=sid;
        }
public String getSid()
        { return sid;
        }
public void setSname(String sname)
        { this.sname=sname;
        }
public String getSname()
        { return sname;
        }
public void setSmarks(int smarks)
        {
        }
public int getSmarks()
        { return smarks;
        }
}
```

d:\hbm

hbmapp
- student.java
- Student.hbm.mm
- hibernate.cfg.xml
- ClientApp.java

JPG to PDF Converter For Mac - Unregis

Student.hbm.xml

```
<DOCTYPE --->
<hibernate-mapping>
 <class name="student" table="student">
   <id name="sid"/>
   <property name="sname"/>
   <property name="smarks"/>
   </class>
  </hibernate-mapping>
```

hibernate.cfg.xml

```
  <!DOCTYPE --->
<hibernate-configuration>
 <session-factory>
 <property name="connection.driver_class">
  oracle.jdbc.driver.OracleDriver </property>
 <property name="connection.url">
   jdbc:oracle:thin@localhost:1521/XE </property>
 <property name="connection.user">system</property>
 <property name="connection.password">durga</property>
 <property name="hibernate.dialect">
   org.hibernate.dialect.OracleDialect </property>
 <property name="show_sql">true </property>
 <mapping resource="student.hbm.xml"/>
   </session-factory>
   </hibernate-configuration>
```

JPG to PDF Converter For Mac - Unregis

ClientApp.java

```
import java.util.*;
import org.hibernate.*;
import org.hibernate.cfg.*;
public class ClientApp
  {
  public static void main(String[] args) throws
                                        Exception
  {
Configuration cfg = new Configuration();
          cfg.configure();
SessionFactory sf = cfg.buildSessionFactory();
Session s = sf.openSession();
Query q = s.createQuery("from student");
      List l = q.list();
System.out.println("SID SNAME SMARKS");
System.out.println("- - - - - - - - -");
    for(int i=0; i<l.size(); i++)
      { Student std = (Student)l.get(i);
System.out(std.getSid() + " " + std.getSname() + " " +
                                std.getSmarks());
      }
    }
  }
```

For variable operation Try catch is not req

Import * java.sql.*

```
d:\hbnu\hsdapp> javac *.java
d:\hbnu\hsdapp> java ClientApp
```

JPG to PDF Converter For Mac - Unregis

Date: 4th may, 2012 Fri

Pagination:- The process of displaying results page by page is called as pagination. To achieve pagination, we have to use setMaxResults() and setFirstResult() method from Query

Go To Tomcat folder and create folder

paginationapp

└ WEB-INF
  ├ lib
  ├ classes

copy and paste the jar files into lib folder ✓

paginationapp
├ WEB-INF
  ├ web.xml
  ├ lib
    ├ *.jar (ojdbc6.jar, hbn.jar)
  ├ classes
    ├ DisplayServlet.class
    ├ product.class
    ├ product-hbm.xml
    ├ hibernate.cfg.xml

web.xml

```
<web-app>
<servlet>
    <servlet-name>ds</servlet-name>
<servlet-class>DisplayServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>ds</servlet-name>
    <url-pattern>/display</url-pattern>
</servlet-mapping>
</web-app>
```

↳ save this under WEB-INF folder

JPG to PDF Converter For Mac - Unregis

Before preparing the DisplayServlet, first of all prepare

product.java

```
public class product
{ private String pid;
  private String pname;
  private int pcost;
public void setpid(String pid)
{ this.pid = pid;
}
public String getpid()
{ return pid;
}
public void setpname(String pname)
{ this.pname = pname;
}
public String getpname()
{ return pname;
}
public void setpcost(int pcost)
{ this.pcost = pcost;
}
public int getpcost()
{ return pcost;
}
}
```

JPG to PDF Converter For Mac - Unregis

product.hbm.xml

```
<!DOCTYPE hibernate-mapping ...!!>
<hibernate-mapping>
<class name="product" table="product">
  <id name="pid"/>
  <property name="pname"/>
  <property name="price"/>
  </class>
  </hibernate-mapping>
```

capital

small

small

small

hibernate.cfg.xml.

```
<!DOCTYPE hibernate-configuration ...!!>
<hibernate-configuration>
<session-factory>
<property name="connection.driver_class">
oracle.jdbc.driver.OracleDriver </property>
<property name="connection.url">
jdbc:oracle:thin:@localhost:1521:xe </property>
<property name="connection.user">system
<property name="connection.password">
durga </property>
<property name="hibernate.dialect">
org.hibernate.dialect.Oracle9Dialect </property>
<property name="show_sql">true </property>
<mapping resource="product.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

copy paste the DOCTYPE.

JPG to PDF Converter For Mac - Unregis

DisplayServlet.java

```
import org.hibernate.*;
import org.hibernate.cfg.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
public class DisplayServlet extends HttpServlet
  {  Configuration cfg;
     SessionFactory sf;
     Session s;
  public void init()
     {  try
        {
  cfg=new Configuration();
        cfg.configure();
        sf=cfg.buildSessionFactory();
        }
        catch(Exception e)
           {  e.printStackTrace();
           }
     }
  public void doGet(HttpServletRequest req,
HttpServletResponse res) throws ServletException,
                                IOException
  {  try
     {  res.setContentType("text/html");
     PrintWriter out=res.getWriter();
```

JPG to PDF Converter For Mac - Unregis

```
String val = req.getParameter("button");
if(val==null)
  { getForm(out);
  }
else
  {
    s=sf.openSession();
    Query q = s.createQuery("from product")
    q.setMaxResults(3);
     if(val.equals("1"))
        { q.setFirstResult(0);
        }
      if(val.equals("2"))
          { q.setFirstResult(3);
          }
       if(val.equals("3"))
          { q.setFirstResult(6);
          }
       Iterator it=q.iterate();
      out.println("<html>");
out.println("<body bgcolor="lightgreen"><center>");
out.println("<br><br>");
out.println("<table bgcolor='lightyellow' border='1'>");
out.println("<tr><td>PID</td><td>PNAME</td>
        <td>PCOST</td></tr>");
      while(it.hasNext())
          { product p=(product)it.next();
            out.println("<tr>");
```

JPG to PDF Converter For Mac - Unregis

```
out.println("<td>"+p.getPid()+"</td><td>"+
    p.getpname()+"</td><td>"+p.getpcost()+
    "</td>");
    out.println("</tr>");
  }
  out.println("<br><br>");
  out.println("</table></center></body>
    getForm(out);                    </html>");
  }
}//catch try
  catch(Exception e)
    { e.printStackTrace();
    }
  }
  public void getForm(PrintWriter out)
  {
    out.println("<html>")
                "<body>"
                ("<center><b><font size='6'>")
    ("<form method='get' action='./display'>");
out.println("<input type="submit" value='1'
                name='button'/>");
out.println("<input type='submit' value='2'
                name='button'/>");
out.println("<input type='submit' value='3'
                name='button'/>");
out.println("</form></font></b></center>");
out.println("</body></html>");
  }
}
```

JPG to PDF Converter For Mac - Unregis

C:\Tomcat 7.0 --\classes > javac *.java

— SQL> select * product;

>

> delete product;

> commit;

> insert into product values('p1', 'aaa', 500);

> insert into product values('p2', 'bbb', 600);

insert

('A', 'AAA', 700);

> commit;

> select * from product

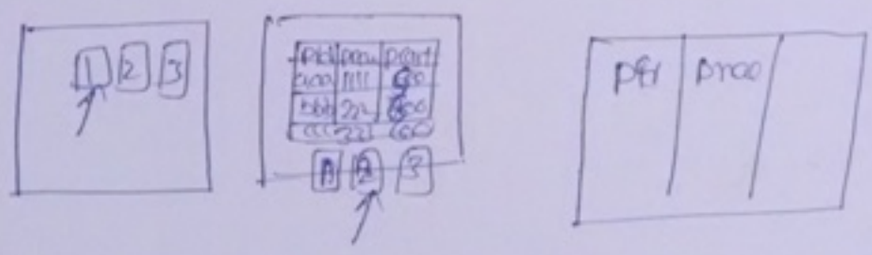
—

— Now I need to access the application.

— > bin > startup.bat

> java DisplayServlet

— open browser and type.



JPG to PDF Converter For Mac - Unregis

HQL Elements.

HQL elements
- classes (from, select, where, groupby, having, orderby)
- Aggregate functions [count, min, max, avg, sum]
- Generic Expressions (operators, in, between, isnull, isnotnull, --)
- parameters (positional parameters
- Associations & joins [1-1, 1-M, M-1, M-M / Inner Join, Outer Join, Left Outer Join --]

— In hibernate applications, to prepare HQL queries we have to use the following elements.

1. classes
2. Aggregate functions
3. Generic expressions
4. parameters.
5. Associations & joins.

JPG to PDF Converter For Mac - Unregis

Clauses: HQL has provided the following list of classes to retrieve results

1. from class
2. select class
3. where class
4. groupby class
5. having class
6. orderby class

from class can be used to specify pojo class name to retrieve the results from the respective table

Syntax:
1. from Entity_Name [Alias_Name]
2. from Entity_Name [as Alias_Name]

Ex:
1. from Employee ✓
2. from Employee as e ✓
3. from Employee e ✓
4. FROM Employee ✓
5. from EMPLOYEE X

Example part:

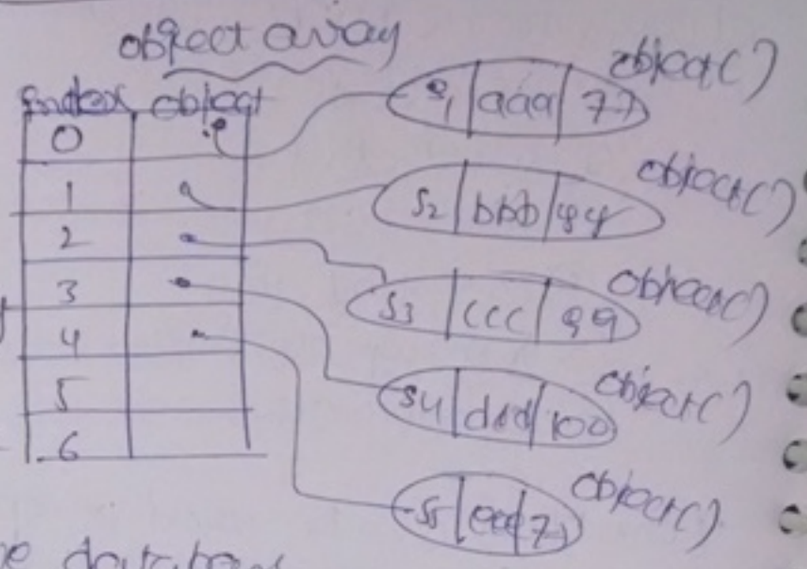
```
Query q=s.createQuery("from Student");
List l=q.list();
```

JPG to PDF Converter For Mac - Unregis

Date: 5th May, 2012 Sat

Select class:

The main purpose of select class in HQL is to specify pojo class properties in HQL queries in order to retrieve the data ~~in the of~~ from the respective database table columns.



If we use select class in HQL query automatically the data will be retrieved in the form of object array but not in the form of pojo class objects.

Syntax:

[select prop1, prop2 --] from pojo_class [(as) alias name]

Exple pgm:

JPG to PDF Converter For Mac - Unregis

```
sf = cfg.buildSessionFactory();
Session s = sf.openSession();
Query q = s.createQuery("select s.sid, s.sname,
s.marks from Student s");
    List l = q.list();
  sop(" SID SNAME SMARKS ");
  sop("------------------------");
    for(int i=0; i<l.size(); i++)
    {
      Object[] o = (Object[]) l.get(i);
    sop(o[0] + "  " + o[1] + "  " + o[2]);
    }
  }
}
```

letter id zero

JPG to PDF Converter For Mac - Unregis

— Where clause: The main purpose of using where clause in HQL queries is to implement conditional expressions in HQL queries.

Syntax: [select prop1, prop2] from pojo-class [(as) alias-name] [where conditional_expression]

Exp/ Pgm

```
Session s = sf.openSession();
Query q = s.createQuery("select s.sid, s.sname, s.smarks from Student s where s.smarks>70 and s.smarks<90");
List l = q.list();
for(int i=0; i<l.size(); i++)
{
Object[] o = (Object[]) l.get(i);
sop(o[0]+" "+o[1]+" "+o[2]);
}
}
}
```

o/p

Groupby clause :-

mysql> select * from student;

SID	Sname	Smarks
S2	aaa	66
S1	bbb	77
S3	ccc	88
S4	ddd	99
S5	aaa	26
S6	ccc	63
S7	bbb	54
S8	ddd	45
S9	aaa	47

I want the avg of their 3 marks subject of group aaa ✓

— All the records are divided into 4 groups based on Sname ✓

— Groupby clause can be used to perform a particular aggregate function over the results retrieved as per groupby column.

If you want to use groupby clause in our HQL queries we must use aggregate function over the results.

Syntax:

[select prop1, prop2 ---] from pojo_class [[as] alias name] [where conditional expression] [group by prop_name]

In exple pgm

```
Query q = s.createQuery("select avg(s.smarks)
from student s group by s.sname");
List l = q.list();
for(int i=0; i<l.size(); i++)
{ object o = l.get(i);
    sop(o);
}
}
```

JPG to PDF Converter For Mac - Unregis

o/p: 67.1, 88.2, 99.3, 70.2 } Avg marks of the groups which are not in an order. Randomly they were returned.

Having class: This class can be used to implement a conditional expression over the groups which are formed as per the group by property.

Syntax:

[select prop1, prop2, ---) from pojo_class[(as) alias_name]
[where conditional Expression] [group by prop_name]
[having condition]

eg: Query q = s.createQuery("select count(s.sname)
from Student s group by s.sname having s.sname
='bob' or s.sname='bbb' or sname='ccc'");

```
List l = q.list();
for(int i=0; i<l.size(); i++)
{
    Object o = l.get(i);
    sop(o);
}
}
```

JPG to PDF Converter For Mac - Unregis

— order by clause :- order by clause can be used to arrange all the results as per a particular property specified along with order by class either with ascending or descending.

Syntax: [select prop1, prop2, ---] from pojo class [(as) alias name] [where condition/ expr] [group by prop name] [having condition] [order by prop [asc/desc]]

exp1

```
Query q = s.createQuery("from Student s
    order by s.sname desc");
        List l = q.list();
        for(int i=0; i<l.size(); i++)
            {
Student std = (Student) l.get(i);
    sop( std.getSid() + " " + std.getSname() + " " +
                                std.getSmarks());
    }
```

prop class obj

exp1

JPG to PDF Converter For Mac - Unregis

— Aggregate functions: The main purpose of aggregate functions is to perform calculations over the results which are retrieved from the database table as per provided HQL query.

— In general in HQL queries we often use aggregate functions with the POJO class properties which are specified along with select class.

— HQL has provided the following list of aggregate functions.

count(–)
sum(–)
min(–)
max(–)
avg(–)

Date: 7th May, 2012 Mon

Count: This aggregate function can be used to count the number of results which are generated when the provided HQL query.

JPG to PDF Converter For Mac - Unregis

Ex: Query q = s.createQuery("select count(s.sid) from Student s");
List l = q.list();
System.out.println(l.get(0));
}
}

Ex: \bin\> hbrsetting.bat
> javac ClientApp.java
> java ClientApp

⑥

Check in the database whe.

— Sum: —

These aggregate function can be used to add all the results w.r.to particular property which are generated with provided HQL query.

Ex: Query q = s.createQuery("select sum(s.marks) from Student s");
List l = q.list();

JPG to PDF Converter For Mac - Unregis

System.out.println(l.get(0));

min:

This aggregate function will be used to return min value
which are generated with the provided HQL query.

Ex: Query q = s.createQuery("select min(s.marks) from Student s");
List l = q.list();
sop(l.get(0));

> javac
java
60

max: This aggregate function can be used to return maximum value among the results generated with the provided HQL query w.r.to a particular entity property.

Ex:

JPG to PDF Converter For Mac - Unregis

```
Query q=s.createQuery("select max(s.smarks)
                        from Student s");
List l=q.list();
sop(l.get(0));
```

```
>javac
>java
```

100

Avg: This function can be used to calculate avg value over the results which are generated with respect to the provided HQL query.

ex:

```
Query q=s.createQuery("select avg(s.smarks)
                        from Student s");
List l=q.list();
sop(l.get(0));
```

```
>javac
>java
```

— Generic Expressions: — The main purpose of the generic expressions is to perform the arithmetic operations, comparison operations and so on over the results, in HQL queries.

HQL has provided the following operators to design HQL queries.

① Mathematical operators.

+, -, *, - -

② Binary operators -

<, >, <=, >=, - -

JPG to PDF Converter For Mac - Unregis

③ Logical operators ④ - In, between, like, is null, is not null.

and, or, - -

- HQL has provided the following scalar functions to display the results in uppercase and lowercase.

① upper(-)
② lower(-)

Ex: Query q = s.createQuery("select upper(s.sname) from Student s");
List l = q.list();
for(int i=0; i<l.size(); i++)
{ sop(l.get(i));
}
}

- In hibernate, HQL has provided the following keywords to implement conditions in HQL queries.

① In: - This keyword can be used to specify list of values along with where clause to retrieve the matched results.

Exple: Query q = s.createQuery("from Student s where s.sname in ('aaa', 'bbb')");
List l = q.list();
for(int i=0; i<l.size(); i++)
{ Student std = (Student) l.get(i);

JPG to PDF Converter For Mac - Unregis

```
sop(std.getSid()+" "+std.getSname()+" "+
                std.getSmarks());
}
```

(Pn) → displays the results that are only matched with aaa or bbb in the table.

```
>
>
```

S_1	aaa	10
S_2	bbb	30
S_4	aaa	60
S_5	bbb	70

between:- This keyword can be utilized to specify a range of values that is minimum and maximum values of a particular property; to retrieve the between results.

exple: Query q = s.createQuery("from student s where s.smarks between 55 and 85");

```
List l = q.list();
for(int i=0; i<l.size(); i++)
  {
student std = (student)l.get(i);
  sop(std.getSid()+" "+std.getSname()+" "+
                std.getSmarks());
  }
>
>
```

S_1
S_2
S_4
S_5

JPG to PDF Converter For Mac - Unregis

like :- This keyword can be used to define a part cular pattern to retrieve the results matched.

```
Query q = s.createQuery("from Student s where
                s.sname like 'b%'");
    List l = q.list();
```

Note: The above HQL query will return all the results where the student name must be started with b but ends with any thing.

Ex: ("from student s where s.sname like '%b'");

not like:

This keyword can be used to specify part pattern to retrieve the results which are not matched with the pattern. in HQL queries

exp: Query q = s.createQuery("from Student s where s.sname not like '%b%'");

Note: The above HQL query can be used to retrieve all the results where the student names are not included the letter 'b'.

JPG to PDF Converter For Mac - Unregis

This keyword can be used to retrieve all the results where the specified property value is null.

Expl Query q = s.createQuery("from Student s where s.sname is null");

```
List l = q.list();
for(int i=0; i<l.size(); i++)
{
```

Old Sql
Sql

S9	null	60
S10	null	70

— is not null : This keyword can be used to retrieve all the results where the specified property value is not null.

space space

expl: Query q = s.createQuery("from Student s where s.sname is not null");

JPG to PDF Converter For Mac - Unregis

HQL parameters :- In general in hibernate applications we will use HQL queries to retrieve multiple number of results. While preparing HQL queries we may provide some values directly that is input values at the time of preparing HQL queries, that is static input.

— If we want to provide input values to the HQL queries dynamically that is after preparing HQL query we have to use parameters in HQL queries.

— HQL has provided two types of parameters

1. positional parameters
2. Named parameters.

positional parameters :— In HQL queries positional parameters will be represented in the form of '?' marks. After specifying positional parameters in HQL queries, we need to set values to the positional parameters. To set a value to a particular positional parameter, we have to use the following method from query

public void setparameter (int index, xxx value)

where xxx may be byte, short, int ---

— where positional parameter index values may be 0, 1, 2, 3, ---

Note: In JDBC positional parameters index values will start from '1'. But in case of HQL, the positional parameters index values will start from '0' (zero).

JPG to PDF Converter For Mac - Unregis

— In HQL queries it is possible to provide more than one positional parameters.

```
Query q=s.createQuery("from student s where
 s.smarks<? and s.smarks>?");
        q.setparameter(0,85);
        q.setparameter(1,75);
        List l=q.list();
    for(int i=0; i<l.size(); i++)
        {
    Student std=(Student) l.get(i);
        System.out.(std.getSid()+" "+std.getSname()+" "+
                                    std.getSmarks());
    }
```

o/p javac
java

S4	aaa	76
S1	bbb	78
S2	ccc	81
S3	aaa	83
S4	bbb	85

— Note: In hibernate applications, we are unable to use positional parameters along with select class, it is possible to use positional parameters in where class with HQL queries.

JPG to PDF Converter For Mac - Unregis

Named parameters:

Date: 8th May 2012 Tue

In HQL queries we can provide named parameters by using the format :name

— If we provide any named parameters in HQL queries then we need to set values to the named parameters. For this we have to use the following method.

```
public void setXxx(String name, xxx value)
```

where xxx may be byte, short, integer, ...

Ex:

```
Query q = s.createQuery("select s.sid,
s.sname, s.smarks from Student s where
s.smarks<:marks");
q.setInteger("marks", 90);
for(int i=0; i<u.size(); i++)
{
Object[] o=(Object[]) u.get(i);
sop(o[0]+" "+o[1]+" "+o[2]);
}
}
```

— In HQL queries, it is possible to provide more than one named parameter.

JPG to PDF Converter For Mac - Unregis

— In HQL queries, it is possible to provide both positional parameters and named parameters with in a single HQL query. But the first parameter should be positional parameters.

In the above context, we are unable to provide positional parameters after the named parameters.

Ex:

```
Query q = s.createQuery("select s.sid, s.names,
s.marks from Student s where s.marks < 7 and
s.s.marks > :marks");
        q.setparameter(0,90);
        q.setInteger("marks", 70);
        List d = q.list();
    for(int i=0; i<d.size(); i++)
        {
  object[] o = (object[]) d.get(i);
    sop( o[0] + " " + o[1] + " " + o[2]);
        }
    }
```

Obj

Dotten

JPG to PDF Converter For Mac - Unregis

Nested queries :- writing an HQL query inside another HQL query, is called as nested queries. HQL will allow nested queries along with where class

```
Query q=s.createQuery("select s.sid, s.sname,
s.smarks from Student s where s.smarks <
(select std.smarks from Student std where
std.name='ddd')");
```

↳ result into (90)

```
List u=q.list();
for(int i=0; i<u.size(); i++)
  {  object o[]=(object[])u.get(i);
     sop(o[0]+" "+o[1]+" "+o[2]);
  }
}
```

O/P

Database

JPG to PDF Converter For Mac - Unregis

~~HQL/SQL:~~

Note: upto hibernate 2.x version, HQL was implemented only to perform retrieval operations. But from hibernate 3.x version, HQL was implemented to perform retrieval operations as well as the operations like update, delete, insert and so on.

— If we want to perform the persistance operations like insert, update, delete and so on -- then we have to use the following method from Query

public int executeUpdate()

If we use executeUpdate(-) method in Hibernate application then we have to perform commit operations on transaction object in order to make the updations as permanent.

exp:

```
Query q = s.createQuery("update Student as
s set s.sname = 'aaa' where s.sname is null");
Transaction tx = s.beginTransaction();
int rowCount = q.executeUpdate();
        tx.commit();
    System.out.println("records updated--" + rowCount);
    }
}
```

o/p) java
java

Database

S1	aaa	100
S2	bbb	90
S3	abc	70
S4	—	66
S5	—	66
S6	ccc	76

JPG to PDF Converter For Mac - Unregis

— if we want to perform insert operations by using HQL we have to use the following HQL query.

```
insert into pojo_class1 (prop1, prop2, --) select
prop1, prop2, -- from pojo_class2 alias name
```

↳ copy one table data having records to

where pojo_class2 is the source and pojo_class1 is the target.

— To copy the records from one table to another table by using the above specified HQL query, we have to use executeUpdate() method from Query

app7

- ClientApp.java
- Student.java (sid, sname, smarks)
- Student1.java (sid, sname, smarks)
- Student.hbm.xml
- Student1.hbm.xml
- hibernate.cfg.xml

JPG to PDF Converter For Mac - Unregis

Student.hbm.xml

Student ⟷ Student
sId ⟷ sId
sname ⟷ sname
smarks ⟷ smarks

Student1.hbm.xml

Student1 ⟷ Student1
sId ⟷ sId
sname ⟷ sname
smarks ⟷ smarks

ClientApp.java

```
Query q = s.createQuery("insert into Student1
(sId, sname, smarks) select s.sId, s.sname,
s.smarks from Student s");
Transaction tx = s.beginTransaction();
int rowCount = q.executeUpdate();
        tx.commit();
System.out.println("records updated -" + rowCount);
}}
```

Ob)

JPG to PDF Converter For Mac - Unregis

In Database (before executing ClientApp)

~~mysql~~

mysql> select * from student;

sid	sname	smarks
s4	aaa	90
s2	bbb	70
s3	abc	60
s1	ccc	90

mysql> select * from student1;

Not existed.

(After executing ClientApp)

mysql> select * from student1;

sid	sname	smarks
s4	aaa	90
s2	bbb	70
s3	abc	60
s1	ccc	90

JPG to PDF Converter For Mac - Unregis

Native SQL:

Date: 9th May, 2012 Wed

— In Hibernate applications, using the methods like save, update, delete and so on from session, we are able to perform the persistance operations with single record.

— To perform persistence operations with multiple records we will use HQL, native HQL and criteria

— By using HQL, we are able to perform DML operations like retrieve, update, insert and so on with multiple no. of records, but it is not possible to perform DDL kind of operations like create, alter and drop.

— In Hibernate applications HQL is not sufficient in the requirements where to use database dependent queries.

ex: In general we will define stored procedures and functions at database and we will access them from java applications. To access stored procedures and functions from java application we have to provide the database dependent SQL queries in java applications.

To achieve this type of requirements HQL is not sufficient.

— To overcome the above problems that is to perform DDL kind of operations and to provide support for stored procedures and functions we have to use native SQL

JPG to PDF Converter For Mac - Unregis

— In hibernate applications, native SQL will allow directly the database dependent SQL queries.

— If we want to use native SQL in hibernate application we will use the following steps

Step①:- prepare SQL query object:

In hibernate applications HQL query will be represented by using Query object. similarly, to represent a native SQL query in hibernate applications we will use a predefined interface. That is org.hibernate.

— SQLQuery is predefined interface provided by hibernate software as a sub type to org.hibernate.Query interface.

SQLQuery

— To create SQLQuery object, we have to use the following method from Session.

Query (int)
↑
SQLQuery (int)

```
public SQLQuery createSQLQuery(String query)
        throws HibernateException
```

ex:

```
SQLQuery q = s.createSQLQuery("select
        * from empl");
```

→ ** where empl is table name not pojo class

JPG to PDF Converter For Mac - Unregis

Step②: execute native SQL query.

To execute native SQL query which we provided along with createSQLQuery() method are represented by SQLQuery object. we have to use the following methods from SQLQuery.

public List list() throws HibernateException
public Iterator iterate() throws HibernateException
public ScrollableResults scroll() throws HibernateException
public Object uniqueResult() throws HibernateException.

ex: List l = q.list();

Pgm: SQLQuery q = s.createSQLQuery("select * from student")

```
List l = q.list();
for(int i=0; i<l.size(); i++)
{
Object[] o=(Object[]) l.get(i);
sop(o[0]+" "+o[1]+" "+o[2]);
}
}
```

small letter (This is table name not class name).

JPG to PDF Converter For Mac - Unregis

— In native SQL, there are two types of SQL queries.

① Entity SQL queries
② Scalar SQL queries.
↑
(individual column)

Entity SQL query :- Entity SQL query is a native SQL query, which should not include column names along with select class, which should include '*' to retrieve all the columns data in the form of a particular entity objects.

— If we want to use entity SQL queries in hibernate applications after representing native SQL query, we have to specify entity type to SQLQuery object inorder to retrieve the results in the form of entity objects.

— To achieve this we have to use the following method from SQLQuery.

public void addEntity(String entity_class_Name)
public void addEntity(Class Entity_class.class file-name)

or SQLQuery q = s.createSQLQuery ("select * from student");
s.addEntity("Student")
↑ capital
↳ database dependent query.

or
// s.addEntity(Student.class);

returned in the form of object array.

JPG to PDF Converter For Mac - Unregis

— In native SQL, there are two types of SQL queries.

① Entity SQL queries
② Scalar SQL queries.
(individual column)

Entity SQL query :- Entity SQL query is a native SQL query, which should not include column names along with select class, which should include '*' to retrieve all the columns data in the form of a particular entity objects.

— If we want to use entity SQL queries in hibernate applications after representing native SQL query, we have to specify entity type to SQLQuery object inorder to retrieve the results in the form of entity objects.

— To achieve this we have to use the following method from SQLQuery.

```
public void addEntity(String entity_class_name)
public void addEntity(Class Entity_class.class file_name)
```

or

```
SQLQuery q = s.createSQLQuery
    ("select * from student");
q.addEntity("Student")
```

↳ database dependent query.

capital

or

```
// q.addEntity(Student.class);
```

returned in the form of object array

select a object not selected object

JPG to PDF Converter For Mac - Unregis

```
List l = q.list();
for(int i=0; i<l.size(); i++)
{
Student std=(Student) l.get(i);
sop( std.getId()+" "+ std.getName+" "+
                std.getMarks());
}
```

— In the above example if we eliminate q.addEntity(-) method, that is if we have not specified Entity type then all the records will be retrieved in the form of Object array.

Note:- In case of entitySQL queries it is possible to provide both positional parameters and named parameters. But the first parameter should be positional parameter and we should not provide any positional parameter after the named parameter.

— If we provide positional parameters and named parameters in native SQL queries then we have to set values to the parameters. For this we have to use the following methods

— To set a value to the positional parameter we have to use the following method

public void setXxx(int index_value, xxx value)

where xxx may be byte, short, integer---

- To set a value to named parameter we will use the following method

```
public void setXxx(String param_name,
                   xxx value)
```

Here xxx may be byte, short, integer ---

example

```
SQLQuery q = s.createSQLQuery("select *
from student where smarks<? and smarks>?
                                  marks");
  q.setInteger(0, 90);
  q.setInteger("marks", 30);
  q.addEntity(Student.class);
  List l = q.list();
for(int i=0; i<l.size(); i++)
  {
Student std = (Student) l.get(i);
SOP(std.getSid() + " " + std.getSname() + " " +
                          std.getSmarks());
  }
}
```

Database

o/p

JPG to PDF Converter For Mac - Unregis

In hibernate applications, if we provide directly native SQL queries in client application along with createSQLQuery() method then this approach is called as programatic approach.

— In hibernate applications if we use programatic approach to provide native SQL queries then that SQL query is available upto the respective client

— If we want to share the same native SQL query to more number of clients we have to declare that SQL query in the mapping file with a particular logical name, we have to get that SQL query on the basis of the logical name in required client applications.

The above approach to declare native SQL queries is called as declarative approach.

— If we want to declare native SQL queries in declarative approach, we have to use the following steps.

Step①: Declare native SQL query in mapping file

— To declare native SQL query in mapping file, we have to use the following xml tags in hibernate mapping file.

```
<hibernate-mapping>
    <class>
    ---
    </class>
<sql-query name="__">
    <return class="__"/>
```

JPG to PDF Converter For Mac - Unregis

—sql query—
</sql-query>

</hibernate-mapping>

— where <sql-query> tag will represent a single native sql query.
— where name attribute in <sql-query> tag can be used to define a logical name for the native sql query.

where <return> tag can be used to specify entity type to retrieve the results.
— where <class attribute in <return> tag will take fully qualified name of the respective entity class.

step①: Get native sql query from mapping file in client application.
— To get named sql query from mapping file into client application we have to use the following method from Session.

```
public Query getNamedQuery(String logical-name)
        throws HibernateException
```

exp: student.hbm.xml

```
<hibernate-mapping>
<class name="Student" table="student">
    <id name="sid"/>
    <property name="sname"/>
    <property name="smarks"/>
</hibernate-mapping>
```

JPG to PDF Converter For Mac - Unregis

```
<sql-query name="entry_sql_query">
  <return class="Student"/>
  select * from student
  </sql query>
  </hibernate-mapping>
```

~~[illegible]~~

ClientApp

```
Query q=s.getNamedQuery("entry_sql_query");
    List l=q.list();
     for(int i=0;i<l.size();i++)
     {
Student std=(Student)l.get(i);
  sop(std.getSid()+" "+std.getName()+" "+
                std.getMarks());
  }
```

- In case of declarative approach, it is possible to provide both named parameters and positional parameters in named sql query of hibernate mapping file.
- If we provide positional parameters and named parameters in named sql query then we have to provide values to the parameters by using setXxx() method.

JPG to PDF Converter For Mac - Unregis

```
<hibernate-mapping>
 <class name="Student" table="student">
  <id name="sid"/>
   <property name="smarks"/>
   </class>
  <set query name="entity_sql_query">
  <return class="Student"/>
  select * from student where smarks >? and
    smarks &lt; ? : marks
  </set query>
 </hibernate mapping>
```

client API

```
      Session s = sf.openSession();
  Query q = s.getNamedQuery("entity_sql_query");
          q.setInteger(0,50);
          q.setInteger("marks", 80);
            List l = q.list();
        for(int i=0; i<l.size(); i++)
          {
  Student std = (student) l.get(i);
    sop( std.getSid()+" "+ std.getName()+" "+
                                  std.getSmarks());
      }
    }
```

JPG to PDF Converter For Mac - Unregis

O/P

Dakshan

Scalar SQL queries: Scalar SQL query is a native SQL query, it can be used to retrieve the data on the basis of the columns.

- In hibernate applications scalar SQL queries will include column names along with select clause to retrieve the data.
- If we use scalar SQL queries to retrieve the data then the data will be retrieved in the form of object array.
- If we want to use scalar SQL queries in hibernate applications then we have to create SQL query object by using createSQLQuery() method

Exp:

```
SQLQuery q = s.createSQLQuery("select sid, sname, smarks from student");
                                          ↑ small
List l = q.list();
for(int i=0; i<l.size(); i++)
    { object[] o=(object[]) l.get(i);
  sop(o[0]+" "+o[1]+" "+o[2]);
    }
}
```

JPG to PDF Converter For Mac - Unregis

Q/P)

Darshan

— In hibernate applications it is possible to provide both positional parameters and named parameters in scalarSQL query. But positional parameter must be the first parameter and we should not define any positional parameter after the named parameters.

exp: SQLQuery q = s.createSQLQuery("select sid, sname, smarks from student where smarks<? and smarks>:marks");

```
q.setInteger(0,80);
q.setInteger("marks",40);
List l = q.list();
for(int i=0; i<l.size(); i++)
{
Object[] o = (Object[])l.get(i);
sop(o[0]+" "+o[1]+" "+o[2]);
}
}
```

JPG to PDF Converter For Mac - Unregis

— In hibernate application it is possible to declare scalar sql queries in mapping file with declarative approach.

To achieve this we have to use the following xml tags in mapping file.

```
<hibernate-mapping>
    <class ->
      =
    </class>
    <sql-query name="_">
        <return-scalar column="_" type="_"/>
        — Scalar-sql-query —
    </
```

— where return scalar tag can be used to represent single scalar that is column specified in scalar sql query.

— where column attribute will take scalar column name.

— where type attribute in return-scalar-tag can be used to specify the scalar type due

JPG to PDF Converter For Mac - Unregis

byte, short, int, String ---.
↑ small

— In declarative approach it is possible to provide both positional parameters and named parameters in scalar-sql-query

orm

```
<hibernate-mapping>
<class name="Student" table="student">
<id name="sid"/>
<property name="sname"/>
<property name="smarks"/>
</class>
<sql-query name="scalar_sql_query">
<return-scalar column="sid" type="String"/>
<return-scalar column="sname" type="String"/>
<return-scalar column="smarks"
        type="int"/>
select sid, sname, smarks from student where
 smarks > ? and smarks $lt; : marks
</sql-query>
</hibernate-mapping>
```

small

client app:

```
Query q = s.getNamedQuery("scalar-sql-query");
        q.setInteger(0,40);
        q.setInteger("marks", 80);
        List l = q.list();
```

JPG to PDF Converter For Mac - Unregis

```
for(int i=0; i<d.size(); i++)
{
object[] o = (object[]) d.get(i);
System.out.println(o[0]+" "+o[1]+" "+o[2]);
}
}
```

O/P

Database

JPG to PDF Converter For Mac - Unregis

Date: 11th May, 2012 Fri

— In general in db applications we often prefer to use SQL queries to implement the simple db application requirements.

— To implement complex db application requirement, SQL queries are not sufficient, where we have to use stored procedures and functions provided by PL/SQL.

— Int. Que: What is the diff b/w stored procedures and functions?

Ans: Stored procedure is a block of instructions maintained at database to represent a particular action and it will not use return statement to return a value.

syntax:

```
create or replace procedure procedure_Name
as                                  ([param-list])
  = global declarations
BEGIN
  = PB logic
END procedure_Name;
```

to save sql>/

— Stored function is a block of instructions provided at database to represent a particular action and it will use return statement to return a value.

JPG to PDF Converter For Mac - Unregis

```
create or replace function function_name
as                                  ([param_list])
  == Global declarations            return datatype
BEGIN
  == DB logic
return value;
END function_name;
```

to save: / ;

— To the stored procedures and functions, we have to provide parameters, where we are able to provide three types of parameters.

① In type parameter.
② OUT type parameter
③

<u>IN Type parameter</u>: This parameter will get the value from procedure call and make available to the procedure body.

<u>OUT TYPE parameter</u>: This parameter will get the value from procedure body and send that value to procedure call.

<u>INOUT type parameter</u>:— This parameter acting as both IN TYPE parameter and OUT type parameter.

var IN datatype ex: exo IN number
var OUT datatype ex: esal OUT number
var [INOUT] datatype ex: eadd INOUT varchar2 (10)

JPG to PDF Converter For Mac - Unregis

- If we want to access stored procedure or function available at database from java application we have to use callable statement
- To access stored procedures and functions from java applications we have to use the following steps.

① prepare callable statement object by providing procedure call or function call.

```
public CallableStatement prepareCall(String procCall) throws SQLException
```

ex: CallableStatement cst = con.prepareCall("{call getSal(2,2)}");

Notes: If we execute the above instruction automatically callable statement object will be created with the specified positional parameters.

② If we have IN type parameters in callable statement object then set values to IN type parameters.

```
public void setXxx(int param_index, xxx value)
```

where xxx may be byte, short, --

ex: cst.setInt(1,111);

③ If we have OUT type parameters in procedure call then register OUT type parameter with a particular data type.

JPG to PDF Converter For Mac - Unregis

```
public void registerOutParameter
        (int param_index, int type)
```

where type may be the constants like BYTE, SHORT, -- from Types class ✓

Ex: cst.registerOutParameter(2, Types.FLOAT);

④ execute Stored procedure:

```
public void execute() throws SQLException
```

Ex: cst.execute();

⑤ get values from out type parameter:

```
public xxx getXxx(int param_index)
```

where xxx may be byte, short, int, --

Ex: float egal=cst.getFloat(2);

JPG to PDF Converter For Mac - Unregis

example program.

```
create or replace procedure getSal (no IN number,
                                    sal OUT number)
as
BEGIN
select esal into sal from empl where eno=no;
 END getSal;
 /
```

↳ Type this in a notepad and copy the data and paste on SQL command prompt and then press enter. Then a procedure will be created.

```
class
 {
  p s v m(s[] a) throws Exception
    {
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection con=DriverManager.getConnection
("jdbc:oracle:thin:@localhost:1521:xe", "system",
                                        "durga");
CallableStatement cst=
 con.prepareCall("{call getSal(?,?)}");
      cst.setInt(1,111);
 cst.registerOutParameter(2,Types.FLOAT);
        cst.execute();
    sop("employee Salary--"+cst.getFloat(2));
        con.close();
    }
 }
```

JPG to PDF Converter For Mac - Unregis

d/p1

Database

```
create or replace function getAvg(no1 IN number,
no2 IN number) return number
as sal1 number;
   sal2 number;
   BEGIN
 select esal into sal1 from emp1 where eno=no1;
 select esal into sal2 from emp1 where eno=no2;
   return (sal1+sal2)/2;
   END getAvg;
   /
```

```
import java.sql.*;
public class FunctionEx
{ public static void main(String[] ars) throws
                                  Exception
  {
```

JPG to PDF Converter For Mac - Unregis

```
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection con = DriverManager.getConnection
("jdbc:oracle:thin:@localhost:1521:xe","system",
"durga");
CallableStatement cst = con.prepareCall
("{?=call getAvg(?,?)}");
    cst.setInt(2,222);
    cst.setInt(3,333);
  cst.registerOutParameter(1,Types.FLOAT);
        cst.execute();
  sop(Avg Salary--"+cst.getFloat(1));
        con.close();
    }
}
```

JPG to PDF Converter For Mac - Unregis

— In the above two examples we prepared a procedure and function to return the data of types number, varchar and so on...

— As per the application requirements we need to return records of data from stored procedure or function.

— To achieve the above requirement our normal data types number, varchar2 and so on are not sufficient, where we have to use cursor types.

— Cursor is a variable, it will provide execution environment to hold up records of data.

— To represent records of data to a cursor type, we will use a predefined cursor that is SYS_REFCURSOR.

```
create or replace procedure getEmps(emps OUT
SYS_REFCURSOR, sal IN number)
 as
 BEGIN
 open emps for
  select * from empl where esal<sal;
  END getEmps;
  /
```

```
import java.sql.*;
 public class procedureEx
  {
   P s v m(s[] a) throwes Exception
   {
```

JPG to PDF Converter For Mac - Unregis

```
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection con=DriverManager.getConnection
("jdbc:oracle:thin:@localhost:1521:xe","system",
                                    "durga");
CallableStatement cst=con.prepareCall(
 "{ call getEmps(?,?)}");
   cst.setFloat(2,100000f);
cst.registerOutParameter(1,oracle.jdbc.OracleTypes.CURSOR);
   cst.execute();
ResultSet rs=(ResultSet)cst.getObject(1);
 sop("ENO ENAME ESAL");
 sop("------------------------");
  while(rs.next())
   {
sop(rs.getInt(1)+" "+rs.getString(2)+" "+
                                rs.getFloat(3));
   }
   con.close();
   }
 }
```

JPG to PDF Converter For Mac - Unregis

Date: 12th May, 2012 Sat

```
/* create or replace function getEmployees(
sal IN number) return SYS_REFCURSOR
as
employees SYS_REFCURSOR;
BEGIN
open employees for
select * from empl where esal<sal;
return employees;
END getEmployees;
/
*/
```

```
— import java.sql.*;
  public class FunctionEx1
    {
      public static void main(String[] ar)
      {
  Class.forName("oracle.jdbc.driver.OracleDriver");
Connection con=DriverManager.getConnection
("jdbc:oracle:thin:@localhost:1521:xe", "system",
"durga");
CallableStatement cst=
con.prepareCall("{?=call getEmployees(?)}");
      cst.setFloat(2, 1000.0f);
cst.registerOutParameter(1, oracle.jdbc.
OracleTypes.CURSOR);
      cst.execute();
```

JPG to PDF Converter For Mac - Unregis

```
ResultSet rs=(ResultSet) cst.getObject();
System.out.println("E_Id ENAME ESA");
System.out.println("----------------");
        System.out.println();
    while(rs.next())
    {
System.out.println(rs.getInt(1)+" "+rs.getString(2)+" "+
        rs.getFloat(3));
    }
    con.close();
  }
}
```

JPG to PDF Converter For Mac - Unregis

- If we want to use stored procedures or functions in hibernate applications, we have to use the following steps.

Step(1): prepare a procedure or function with cursor types at database.

Step(2): configure procedure call in the respective mapping file.

- To configure procedure call in mapping file, we have to use the following sql tags.

```
<hibernate-mapping>
  <class ->
    ==
  </class>
  <sql-query name="__" callable="true">
    <return class="__"/>
    ---procedure-call---
  </sql-query>
</hibernate-mapping>
```

- where callable attribute in sql query tag is boolean attribute, it can be used to given information to the hibernate software about to access stored procedure or function by generating procedure call & function call from mapping file with the callable statement object.

JPG to PDF Converter For Mac - Unregis

Note: In the respective stored procedure or function if we use the SQL query with the scalars then in the procedure call configuration we have to use return-scalar tag in place of return tag under set-query tag.

Step③: prepare Query object by using getNamedQuery() method, set values to IN type parameters, as per procedure call and execute the respective procedure.

```
create or replace procedure getStudents(stds
OUT SYS_REFCURSOR, marks IN number)
as
BEGIN
open stds for
select * from student where smarks < marks;
END getStudents;
```

In mapping file

```
<hibernate-mapping>
<class name="Student" table="student">
<id name="sid"/>
<property name="sname"/>
<property name="smarks"/>
</class>
<set-query name="proc" callable="true">
<return class="Student"/>
{call getStudents(?,:marks)}
</sql-query>
```

JPG to PDF Converter For Mac - Unregis

<hibernate-mapping>

In client application

```
Query q = s.getNamedQuery("proc");
    q.setInteger("marks", 90);
    List l = q.list();
    for(int i=0; i<l.size(); i++)
    {
Student std = (Student)l.get(i);
    SOP(std.getSid()+"\t"+std.getSname()+
        "\t"+std.getSmarks());
    }
```

OR

Deletion

This may work on oj1bc jar only not oj1dbc.jar

oj1bcs.jar

Lotto on 11/12

JPG to PDF Converter For Mac - Unregis

— If we provide scalar sql query in the respective stored procedure and function then the data will be retrieved in the form of scalars that is columns and it can be represented in an object array.

In this case we have to configure scalars in procedure call configuration at mapping file.

```
create or replace procedure getStudents( sids
OUT SYS_REFCURSOR, marks IN number)
as
BEGIN
open sids for
select sid, sname from student where smarks<
END getStudents;                              marks;
```

<u>In hibernate mapping file</u>

```
<class>
  <sql-query name="proc" callable="true">
<return-scalar column="sid" type="serial"/>
<return-scalar column="sname" type="string"/>
  {call getStudents(?, :marks)}
  </sql-query>
  </hibernate-mapping>
```

JPG to PDF Converter For Mac - Unregis

In client application

```
Query q = s.getNamedQuery("proc");
  q.setInteger("marks", 90);
    List l = q.list();
  for(int i=0; i<l.size(); i++)
    {
  Object[] o = (Object[]) l.get(i);
  System.out.(o[0] + " " + o[1] + " " + o[2]);
  }
}
```

JPG to PDF Converter For Mac - Unregis

Net Beans IDE

[Date: 13th may, 2012 Sun]

Steps to design J2SE application with NetBeans IDE

Step①: Create Java project

File → new project → select Java under Categories
↓
select Java application under projects → [next]

project Name: [HeloAPP]

[Finish]
↑

If we do the above a project will be created in projects browser (left side frame) and a class with HeloAPP name will be created in the middle frame of net beans IDE

— After getting the class provide implementation as per the application requirement.

Step②: Run Java application.

right click on project → [Run]
↑

If we do the above output will be generated in output window. (bottom part of netbeans IDE)

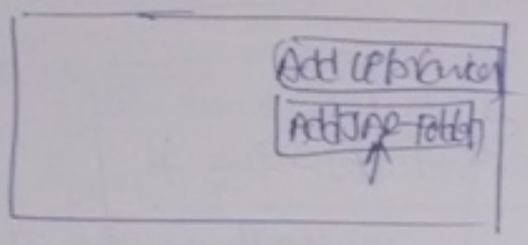
JPG to PDF Converter For Mac - Unregis

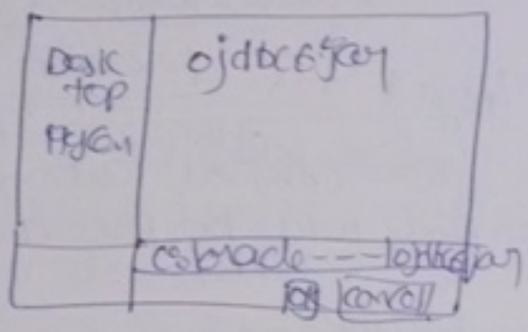
→ Design JDBC application

If we want to design JDBC application with Net Beans IDE then we have to add ~~the~~ ~~jar~~ driver respective jar file to the java project

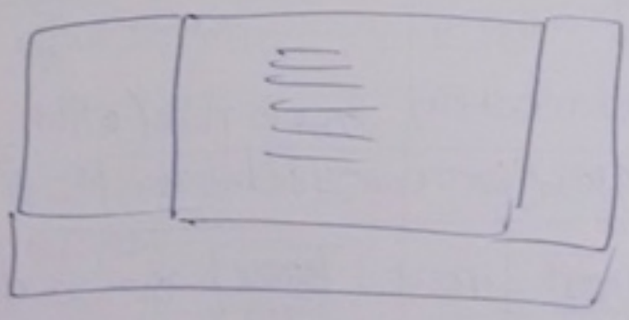
For this we have to use the following path

Right click on project (HelloApp) → properties
↓
Libraries





If we do the above, then the specified jar file will be available in libraries folder of project.



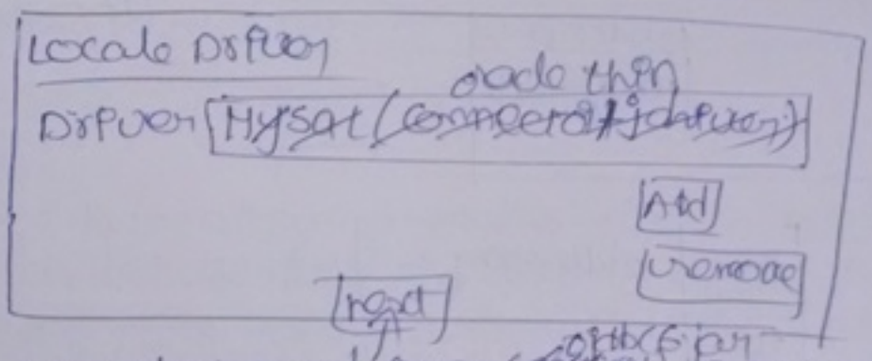
JPG to PDF Converter For Mac - Unregis

Steps to design hibernate application with Net Beans IDE

Step①: prepare database [~~mysql database~~ oracle ~~Actually there is database mysql~~]

select services located on the left hand side of project explorer.

- Right click on database → new connection.



If we select the driver (~~mysql~~ oracle jar) automatically mysql connector jar file will be provided by net beans IDE.

- If we dont want to use net beans provided jar file, then we have to remove it and add another jar file.

next

Customize connection

Driver name: [~~MySQL Connector/J Driver~~ oracle Thin (with service ID (SID))]

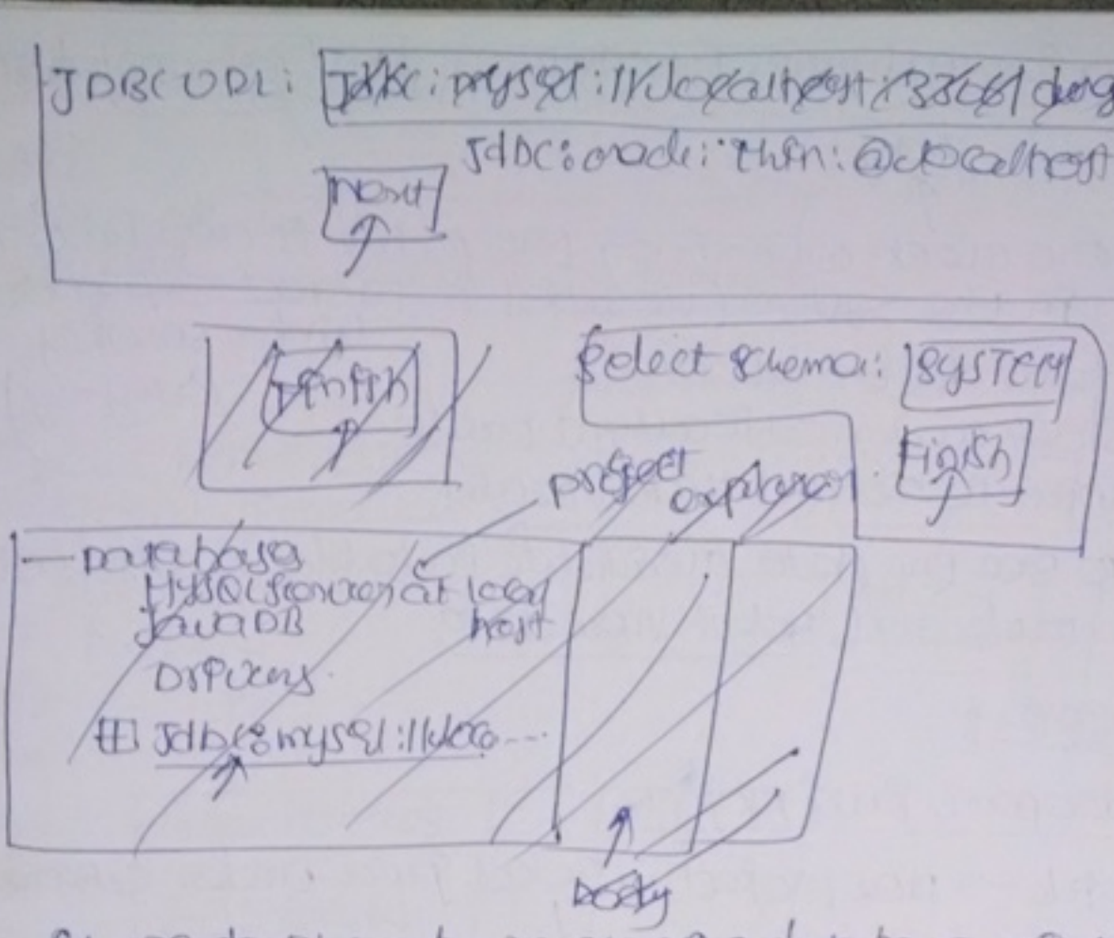
Host: [localhost] Port: [~~3306~~ 1521] Service ID: [xe]

Database: [~~mydatabase~~]

Username: [~~root~~] system

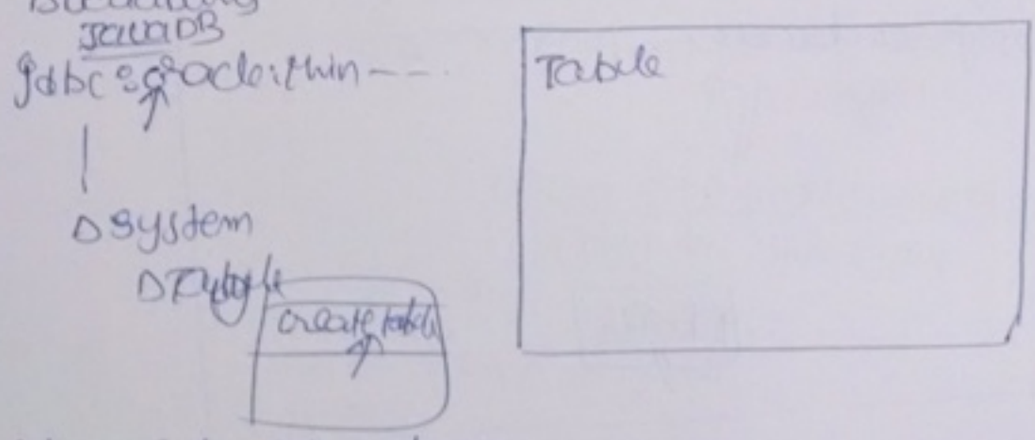
Password: [~~root~~] dungi

JPG to PDF Converter For Mac - Unregis



If we do the above steps a database will be created under database, with the driver and name.

— To prepare a table, we have to expand created database system



If we do the above,
a window will be open,
where we have to provide table name and

JPG to PDF Converter For Mac - Unregis

provide columns by click on [add column] button

[ok]

Now check whether the table is created or not in the system (located in project explorer under sources category)

— If we do the above step1 automatically a table will be created under tables.

— To see the data available in tables, right click on table and select view data

Step ②:

prepare java project

File → New project → select java under categories

↓

select java applications under projects

[next]

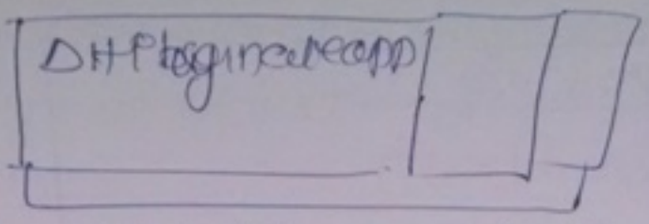
project name: [hibernateapp]

project location: |

[Finish]

JPG to PDF Converter For Mac - Unregis

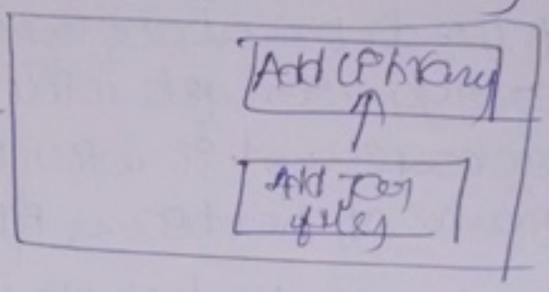
step③: Add hibernate library



right click on project

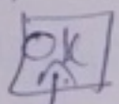
↓

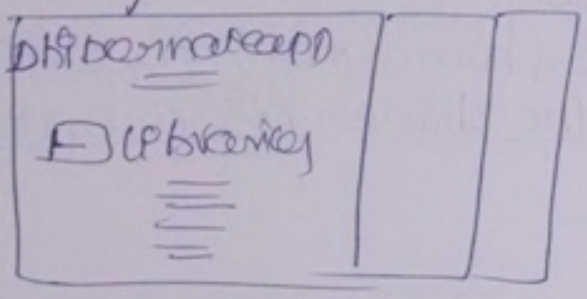
properties → libraries



select hibernate in global libraries.

click on add library button.





step④: prepare configuration file.

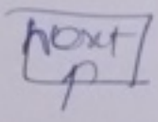
Right click on project

↓

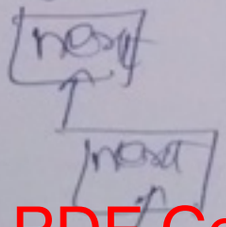
new → other → select hibernate in categories.

↓

select hibernate configuration wizard in file types

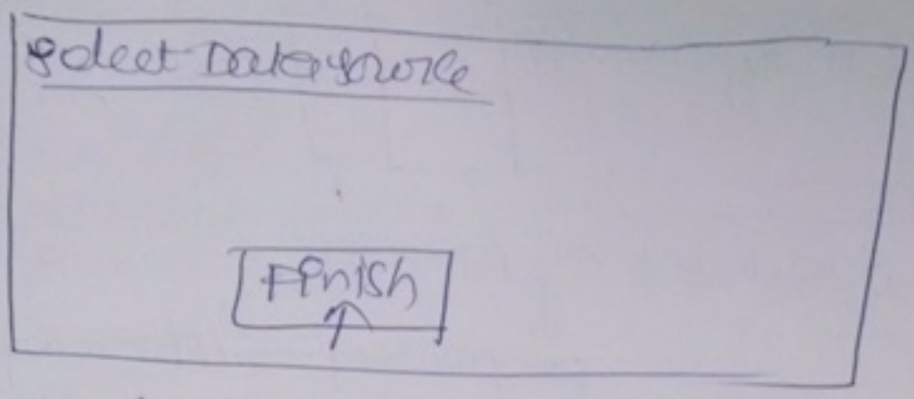


Again click on



again.

JPG to PDF Converter For Mac - Unregis



If we do the above steps, automatically configuration file will be created in source package and it will be open in the middle frame of netbeans ide.

— change dialect class name from OracleDialect to OracleDialect

— change driver class name from oracle.jdbc.OracleDriver to oracle.jdbc.driver.OracleDriver

JPG to PDF Converter For Mac - Unregis

Step(5): prepare reverse engineering file.

Right click on project → other → select Hibernate (under categories)

↓

select Hibernate reverse engineering wizard under file types.

Select emp2.

table in available tables

Add | emp2

next

[If we click on Add button emp2 table will be added to selected tables]

Finish

D source packages

JPG to PDF Converter For Mac - Unregis

Step 6 create pojo class and mapping file on the basis of the database.

Right click on project → new → other

↓

select hibernate under category

↓

select hibernate mapping file and pojos from database under file types.

next

provide package name: com.dao

finish

com.dao

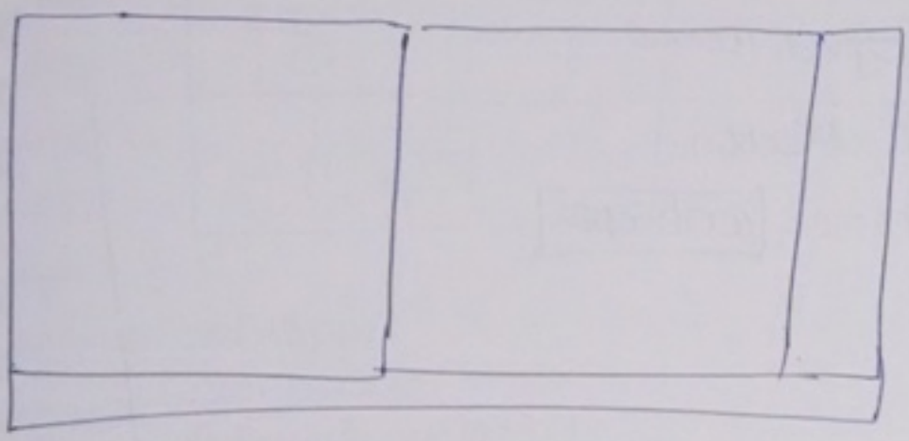
Emp2.hbm.xml

JPG to PDF Converter For Mac - Unregis

If we do the above, emp2.java file and emp2.hbm.xml file will be created in com.dsg package under source packages.

Step(7): prepare client Application.

At the time of preparing java project a java class was created, we can utilize that class as client Application.



Step(8): Run hibernate application

Right click on project → Select run →

JPG to PDF Converter For Mac - Unregis

Steps to Design web application with NetBeans IDE

Step 1: prepare web project.

File → new → select Java web under categories

↓

select web application under projects

[next]

— provide project name.

Name and location.

project name: [webapp]

[next]

~~Server and settings~~

— If we do the above steps automatically a window will be open, where we have to provide server:

Server and settings

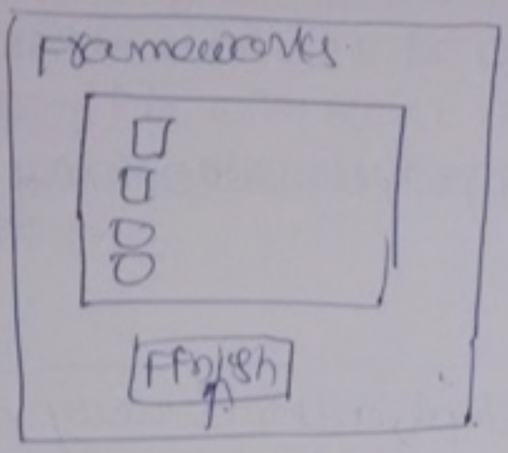
Server: [Apache Tomcat] [Add]

Apache
oracle web
logic

JPG to PDF Converter For Mac - Unregis

[next]

Select tomcat server by click on [set] button.



- webapp2
  - webpages
  - Source packages
  - Libraries.

Step②: prepare web resources as per the application requirement.

Right click on project (

↓

new → servlet →

Name and location:

Servlet class Name: [HelloServlet]

project:

Location:

package: [com.dss]

[next]

JPG to PDF Converter For Mac - Unregis

☑ Add information to deployment descriptor (web.xml)

select ClassName:

ServletName:

URLpattern: /hello

(edit URL pattern as per the requirement)

Finish

— If we do the above steps, automatically a servlet will be created under source package and it will be open in middle frame of net beans IDE

— After getting servlet, provide the implementation as per the application requirement.

JPG to PDF Converter For Mac - Unregis

Step :- Run the web application

Right click on project (webapp2)

↓

Run

If we do the above step, automatically web application will be deployed to the server, netbeans ide will start the server and access the web application automatically by open web browser.

JPG to PDF Converter For Mac - Unregis

Steps to design struts based web application when netbeans IDE

Step①: prepare web project

File → new project → select Java web under categories and select web application under projects.

next

Name and location

project Name: strutsapp1

next

server: Apache Tomcat server

next

Frameworks:

☑ struts 1.3.8

☐ Add struts lib

Finish

JPG to PDF Converter For Mac - Unregis

If we do the above steps, automatically a web project will be created with all the struts capabilities.

Step 2: prepare user form as per the requirement

right click on project → new → JSP.

Name and location.

File name: loginform

Here don't specify any extension only because it will automatically

Finish

JPG to PDF Converter For Mac - Unregis

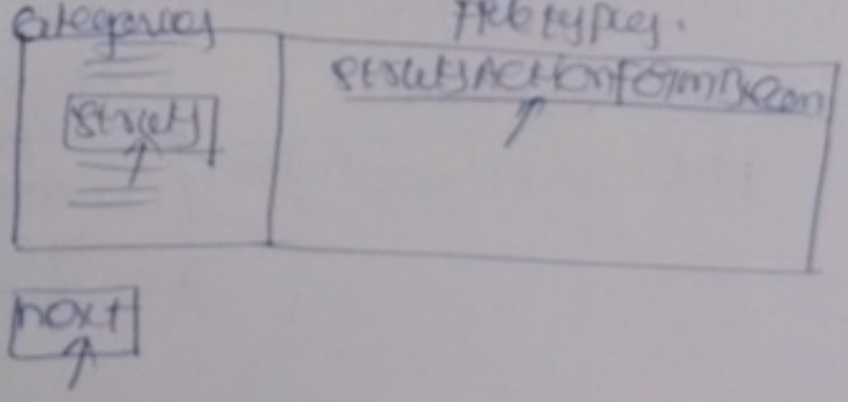
Notes: As per the application requirement, we have to prepare the JSP pages like success.jsp, failure.jsp, loginform.jsp and so on--

Step③: prepare ActionForm class.

right click on project(

↓

new → other.



Name and

class Name: LoginActionForm

package: com.dss

JPG to PDF Converter For Mac - Unregis

Finish

— If we do the above steps automatically a formBean class will be created under source package and it will be open in the middle frame of NetBeans IDE.

— In the generated formBean class, declare required properties

```
private String uname;
private String upwd;
```

— Right click on any property.

select insert code

select getter and setter

select both uname & upwd properties

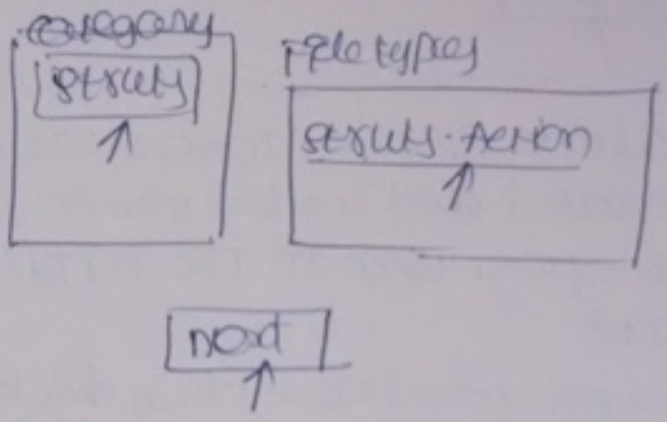
click on Generate button.

JPG to PDF Converter For Mac - Unregis

Step ④ : create Action class

right click on project (

↓ new → other →



class Name : LoginAction

package : com.dsg

Action path : /login

next

JPG to PDF Converter For Mac - Unregis

Finish

- If we do the above steps, automatically Action class will be created under source package and it will be open in middle frame of netbeans IDE.
- After getting Action class, provide application logic in execute method and define the required number of forwards in struts configuration file.

JPG to PDF Converter For Mac - Unregis

Step ⑤ : Run the application

Right click on project → Run.

JPG to PDF Converter For Mac - Unregis

Date: 14th may 2012 Mon

In hibernate application, it is possible to access stored function available database from hibernate application.

```
/* create or replace function getStds (min IN
 number, max IN number) return SYS_REFCURSOR
 as
  stds SYS_REFCURSOR;
  BEGIN
  open stds for
   select * from student where smarks > min
        and smarks < max;
    return stds;
    END getStds;
    /
  */
```

In mapping file

```
<!DOCTYPE ---->
<hibernate-mapping>
 <class- -
  ---
 </class>
<set-query name="Function" callable="true">
<return class="student"/>
  {? = call getStds(:min, :max)}
  </set-query>
  </hibernate-mapping>
```

JPG to PDF Converter For Mac - Unregis

Client APP

```
Query q=s.getNamedQuery("Function");
    q.setInteger("min",60);
    q.setInteger("max",90);
    List l=q.list();
    for(int i=0;i<l.size();i++)
      {
      Student st=(Student)l.get(i);
    System.out.(st.getSid()+" "+st.getSname()+" "+
                                st.getSmarks());
    }
```

Sql

Database

JPG to PDF Converter For Mac - Unregis

Criteria API

— In hibernate applications by using the methods like save(), persist(), get(), update() and so on from Session we are able to perform persistent operations with single records.

In Hbn applications to perform persistent operations with multiple records hibernate has provided HQL, native SQL, and criteria api.

— If we are using HQL and native SQL in hibernate application then we must provide query language syntaxes either with the POJO class names and properties or with the database table names and column names.

In the above context, to reduce query language dependency we have to use criteria api.

The main purpose of criteria api is to eliminate query language dependency in hibernate application; to implement the complete persistent logic in the form of Java methods.

— To implement the persistent logic by using Java methods that is criteria api, hibernate has provided very good predefined library in the form org.hibernate package and org.hibernate.criterion package.

JPG to PDF Converter For Mac - Unregis

— If we want to use criteria API in hibernate applications, then we have to use the following steps.

Step① :- create criteria object

— To get criteria object we have to use the following overloaded methods from Session

```
public Criteria createCriteria(String EntityClass)
public Criteria createCriteria(Class class);
```

ex: Criteria c = s.createCriteria(Student.class);

In hbn applications, criteria object is able to represent the complete persistent logic which we implemented through criteria API that is java methods.

Step②: provide restrictions to criteria object

In criteria API, each and every restriction is nothing but a condition, which will be represented in the form of org.hibernate.criterion.Restrictions class.

To define restrictions to criteria object, we have to use the following methods from Restrictions class.

JPG to PDF Converter For Mac - Unregis

Date: 15th may 2012 Tue.

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 lt(String property, Object o)

public static criterion lt(String property, String property)

public static criterion le(String property, Object o)

public static criterion le(String property, String property)

public static criterion gt(String property, Object o)

public static criterion gt(String property, String property)

public static criterion eq(String property, Object value)

public static criterion eq(String property, String property)

public static criterion ne(String property, Object value)

public static criterion ne(String property, String property)

public static criterion in(String property, Object[] values)

JPG to PDF Converter For Mac - Unregis

public static criterion in(String property, Collection values)
public static criterion between(String property, Object val1, Object val2)
public static criterion between(String property, Collection values)

exple:- criterion c1 = Restrictions.eq("eno", 111);
criterion c2 = Restrictions.lt("esal", 1000);

— After preparing criterion objects with the restrictions we need to add all criterion objects to criteria objects.

To achieve this we have to use the following method from criteria.

public void add(Criterion c)

ex:- c.add(c1);
c.add(c2);

Step③:- prepare projections and add projections to criteria object.

— In criteria api, to represent a single property in the persistance logic, we have to use projection.

JPG to PDF Converter For Mac - Unregis

— To prepare projections we have to use the following method.

public static PropertyProjection property(String property name)

where PropertyProjection is a sub class to Projection.

2 ⇒ After preparing all the projection objects, we have to add all the projection objects to the ProjectionList.

To prepare ProjectionList, we have to use the following method from Projections class.

```
public static ProjectionList projectionList()
```

— To add a projection to projectionlist, we have to use the following method

```
public void add(Projection p)
```

— After getting projectionlist with projection we need to add projectionlist to criteria object, for this 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"));
```

JPG to PDF Converter For Mac - Unregis

pro.add(Projections.property("esal"));
c.setProjection(pr);

specifying ordering

— In general in persistance logic we may use either ascending order or descending order over the results, to represent a particular order criteria API has provided a predefined class that is org.hibernate.criterion.Order

— To specify particular order, over the results with respect to a property, we have to use the following method

```
public static order asc(String prop)
public static order desc(String prop)
```

— To add a particular order to criteria object we have to use the following method

```
public void addOrder(Order o)
```

ex: c.addOrder(Order.desc("ename"));

Step :- execute the persistance logic which was represented by criteria object.

— To execute persistance logic represented by the criteria object, we have to use the following methods,

JPG to PDF Converter For Mac - Unregis

```
public List list()
public ScrollableResult scroll()
public Object uniqueResult()
```

smart

ex: List l = c.list();

Notes: Like Query() object in HQL, it is possible to provide customizations over the results generated from the respective criteria object.

```
public void setFetchSize(int size)
public void setFirstResult(int pos)
public void setMaxResults(int no)
public void readOnly(boolean b)
public void cacheable(boolean b)
public void setComment(String comment)
public void setTimeOut(int time)
```

exp: pgm

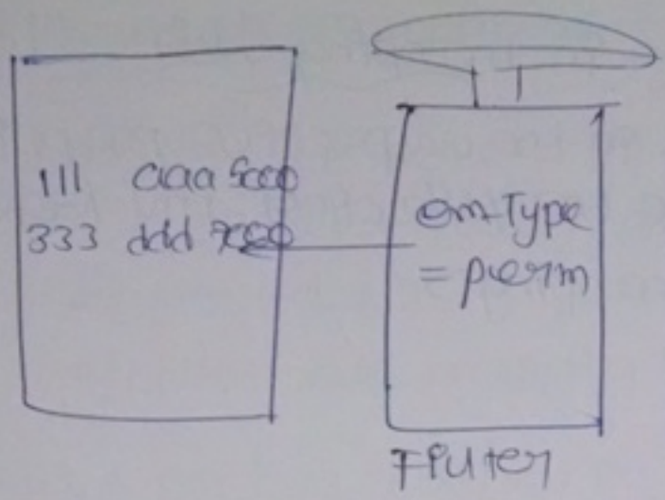
clientApplication

```
import java.util.*;
import org.hibernate.*;
import org.hibernate.cfg.*;
import org.hibernate.criterion.*;
public class ClientApp
{
public static void main(String[] args) throws Exception
{
```

JPG to PDF Converter For Mac - Unregis

```
Configuration cfg= new Configuration();
        cfg.configure();
SessionFactory sf= cfg.buildSessionFactory();
  Session s= sf.openSession();
Criteria c=s.createCriteria(Student.class)
                    cap
Criterion c1= Restrictions.lt("smarks", 90);
Criterion c2= Restrictions.gt("smarks", 60);
        c.add(c1);
        c.add(c2);
projectionList pl= projections.projectionList();
    pl.add(projections.property("sId"));
    pl.add(projections.property("sname"));
    pl.add(projections.property("smarks"));
        c.setProjection(pl);
    c.addOrder(order.desc("sname"));
        List l= c.list();
    for(int i=0; i<l.size(); i++)
    {
    Object[] o=(Object[]) l.get(i);
    sop(o[0]+" "+o[1]+" "+o[2]);
    }
  }
}
```

JPG to PDF Converter For Mac - Unregis



eno	ename	esal	emp Type
111	aaa	3000	perm
222	bbb	6000	temp
333	ccc	7000	perm
444	ddd	8000	temp

Date: 17th may, 2012 Thurs

— In general, in hibernate applications, by using HQL queries, we are able to retrieve some results from the database table.

— In the above context, as per the application requirements, we want to retrieve all the results which are satisfying a particular condition.

— To achieve the above requirement, we have to use hibernate filters.

The main advantage of filters in hibernate applications is to retrieve the results as per a condition and it is possible to enable and disable the filters at run time. That is retrieving either all the results or the results satisfying the conditions.

— If we want to use filters in hibernate applications, we have to use the following steps.

JPG to PDF Converter For Mac - Unregis

Step①: Define filter in mapping file.

To define a filter in the respective mapping file, we have to use the following xml tags.

```
<hibernate-mapping>
    <class ->
    =
    </class>
<filter-def name="_">
    <filter-param name="_" type="_"/>
</filter-def>
</hibernate-mapping>
```

— where <filter-def> tag can be used to declare a single filter in mapping file.

— where name attribute in <filter-def> tag can be used to specify the logical name of the filter.

— where <filter-param> tag can be used to declare a single parameter to the filter.

— where name attribute in <filter-param> tag can be used to ~~kk~~ provide name of the parameter.

— where type attribute in <filter-param> tag can be used to specify java type to the filter-parameter.

JPG to PDF Converter For Mac - Unregis

```
<hibernate-mapping>
    <filter-def name="my_filter">
        <filter-param name="type" type="string"/>
    </filter-def>
</hibernate-mapping>
```

Small

Step ①: Declare a filter in class configuration.

— To declare a filter in class configuration, we have to use the following some tags.

```
<hibernate-mapping>
    <class --->
        <filter name="__" condition="__"/>
    </class>
</hibernate-mapping>
```

— where <filter--> tag can be used to declare a single filter in class configuration.

— where name attribute will take logical name of the filter which we defined in the mapping file.

— where condition attribute will take a conditional expression.

Ex:

```
<hibernate-mapping>
    <class name="Employee" table="emp">
        <filter name="my_filter" condition="type=emp_type"/>
    </class>
</hibernate-mapping>
```

JPG to PDF Converter For Mac - Unregis

Step (4): Enable filter in client application.

- To enable filter in client application, we have to use the following method from Query.

```
public Filter enableFilter(String logical-name)
```

- If we declare any parameter inside the filter then we must pass parameter value, for this we have to use the following method.

```
public void setParameter(String name,
                         Object value)
```

- If we want to disable the filter then we have to use the following method

```
public Filter disableFilter(String logical-name)
```

Eg

```
Query q = s.createQuery("from Employee");
Filter f = q.enableFilter("myFilter");
f.setParameter("type", "perm");
List l = q.list();
```

JPG to PDF Converter For Mac - Unregis

In database table

SQL> create table account(accno varchar2(5),

firstapp.

- Account.java
- Account.hbm.dtd.
- hibernate.cfg.dtd.
- ClientApp.java

Account.java

```
public class Account
  { private String accNo;
    private String accName;
    private String accBranch;
    private String accType;
  public void setAccNo(String accNo)
      { this.accNo=accNo;
      }
  public String getAccNo()
      { return accNo;
      }
  public void setAccName(String accName)
      { this.accName=accName;
      }
```

JPG to PDF Converter For Mac - Unregis

```
public String getAccName()
        { return AccName;
        }
public void setAccount Branch(String accBranch)
        { this.accBranch = accBranch;
        }
public String getAccBranch()
        { return accBranch;
        }
public void setAccType(String accType)
        { this.accType = accType;
        }
public String getACCType()
        { return accType;
        }
}
```

Account.hbm.xml

```
<DOCTYPE-->
<hibernate-mapping>
<class name="Account" table="account">
  <id name="accNo"/>
  <property name="accName"/>
  <property name="accBranch"/>
  <property name="accType"/>
<filter name="acc-type-filter"
    condition=":type = accType"/>
</class>
```

JPG to PDF Converter For Mac - Unregis

```
<filter-def name="acc_type_filter">
<filter-param name="type" type="string"/>
</filter-def>
</hibernate-mapping>
```

— hibernate.cfg.xml

```
<!DOCTYPE ---->
<hibernate-configuration>
<session-factory>
<property name="connection.driver_class">
oracle.jdbc.driver.OracleDriver </property>
<property name="connection.url">
jdbc:oracle:thin:@localhost:1521:xe
<property name="connection.user">
system </property>
<property name="connection.password">
durga </property>
<property name="hibernate.dialect">
org.hibernate.dialect.OracleDialect
</property>
<property name="show_sql">true
</property>
<mapping resource="Account.
hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

JPG to PDF Converter For Mac - Unregis

ClientApp.java

```
import org.hibernate.cfg.*;
import org.hibernate.*;
import java.util.*;
public class ClientApp
{
public static void main(String[] args) throws
                                        Exception
{
Configuration cfg=new Configuration();
        cfg.configure();
SessionFactory sf = cfg.buildSessionFactory();
Session s= sf.openSession();
Query q = s.createQuery("from Account");
Filter f = s.enableFilter("acc_type_filter");
Scanner sc new Scanner(System.in);
s.o.p("select Account Type");
sop("1.Savings");
sop("2.current"); sop("3.all");
String option = sc.nextLine();
if(option.equals("Savings"))
    {
      f.setparameter("type","Savings");
    }
if(option.equals("Current"))
    {
f.setparameter("type","Current");
    }
if(option.equals("All"))
    {
s.disableFilter("acc_type_filter")
```

JPG to PDF Converter For Mac - Unregis

```
List d = e.list();
 for(int i=0; i<d.size(); i++)
      {  Account a = (Account)d.get(i);
         System.out.println(a.getACCNO()+" "+
a.getACCName()+" "+a.getACCBranch+" "+
a.getACCType());
      }
     e.close();
   }
 }
```

D:\hbmy\ejetterapp> javac AirportApp.java

— In SQL

```
SQL> insert into account values ('a1', 'aaa', 'xxx',
                                 'savings');
  > insert into account values ('a2', 'bbb', 'yyy',
                                 'current');
  >                             ('a3', 'ccc', 'zzz',
                                 'savings');
  > insert into account values ('a4', 'ddd', 'xxx',
                                 'current');
  > insert
```

JPG to PDF Converter For Mac - Unregis

SQL> select * from

> java clientAPPd

select Account Type:
1. Savings
2. Current 30 AU
Savings

> java clientAPPd

current

JPG to PDF Converter For Mac - Unregis

Hibernate mappings

Date: 18th may 2012 Fri

As part of the enterprise application development, it is essential to use object oriented data model and relational data model.

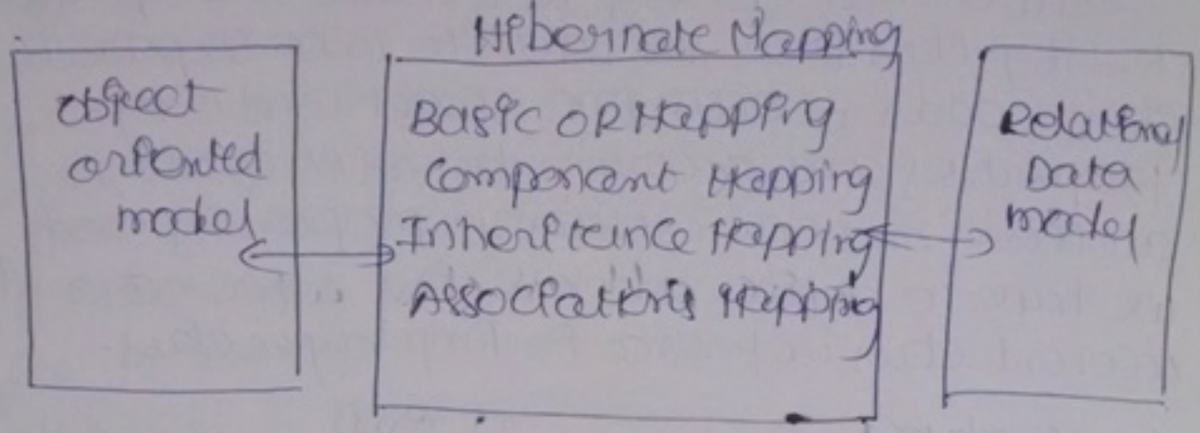
- In object oriented model, the data will be represented in the form of objects, it will prepare the applications by using the object oriented features like inheritance, polymorphism and so on.
- In relational data model, data will be represented in the form of tables. Here to achieve the application requirements like inheritance, associations, we will provide primary key foreign key relationships between tables.
- In enterprise application development, if we use both object oriented data model and relation data model then they may provide some mismatches like inheritance [subtypes mismatch], Associations mismatch, granularity mismatch and so on--

In the above context, due to the mismatches between object oriented data model and relational data model, data persistency is difficult in enterprise applications.

In the above situation, to simplify the data persistency hibernate has provided the following four mappings.

JPG to PDF Converter For Mac - Unregis

1. Basic OR Mapping
2. Component mapping
3. Inheritance mapping
4. Associations mapping



Basic OR Mapping :-

It is very common mapping between object oriented model and relational model, it will include the mapping between a class from object oriented model and a table from relational model and the respective properties and columns.

Component mapping :- In general in object oriented model, to represent a particular entity we will define a class with the properties.

Ex: To declare employee entity, we will define class employee with the properties eno and ename.

— As per the application requirement if we want to add some extra details associative to that entity then we need to define

JPG to PDF Converter For Mac - Unregis

another class with the required properties and we will define the reference of new class in the entity class.

Ex: If we want to provide address details and account details to the above employee entity class then we have to take separate classes address with pno, street and city properties and account class with accno, accname, and accountBranch properties and we have to define address class reference and account class reference in Employee class.

```
class Employee
{
int eno;
String ename;
Address addr;
Account acc;
}
class Address
{ String pno
  String street;
  String city;
}
class Account
{
String accNo;
String accName;
String accBranch;
}
```

```
class:
  Employee-emp
Id:
  eno → eno
Component: name:addr,
  class Address
    pno → pno
    street → street
    city → city
Component: name:acc,
  class: Account
    accNo → accNo
    accName → accName
    accBranch
      ↳ accBranch
property:
  ename → ename
```

emp

eno	ename	addr	
		pno	street
111	abc	b1	15 road

city	accNo	accName
hyd	a1	aaa

accBranch
SR nagar

acc

eno: 11
ename: abc
addr
acc
e

pno: b1
street: 15 road
city: hyd

s.save(e)

accNo: a1
accName: aaa
accBranch: SR nagar

JPG to PDF Converter For Mac - Unregis

— In relational model, if we want to represent an entity then we have to define a table with the columns.

Ex: To define Employee Entity in relational model, we will declare a table with the name emp and with the colums eno, and ename. As per the new requirement, if we want to provide some extra data to the same entity then we have to add some extra columns to the entity table.

Ex: If we want to add address and account details to the employee entity then we have to add pno, street, city with respect to address and accNo, accName and accBranch with respect to Account to the emp table.

2 → In the above context, we are able to identify your properties in Employee entity class in object oriented model and eight columns at emp table in relational model, it will give granularity mismatch between object oriented model and relational model.

→ To solve this granularity mismatch hibernate has provided component mapping as a solution.

In component mapping, we are able to resolve granularity mismatch by declaring component properties.

JPG to PDF Converter For Mac - Unregis

- Component property is a property in pojo class, it able to represent multiple number of columns in the database table.

Date: 19th may, 2012 Sat

- To provide component mapping in hibernate application, we have to use the following xml tags in hibernate mapping file.

```
<hibernate-mapping>
<class name="_" table="_">
<id name="_" column="_"/>
<component name="_" class="_">
<property name="_" column="_"/>
</component>
<property name="_" column="_"/>
</class>
</hibernate-mapping>
```

- where component tag can be used to represent or configure a single component property.
- where name attribute in component tag can be used to provide component property name defined in pojo class.
- where class attribute in components tag can be used to provide fully qualified name of the class required by the component property.

JPG to PDF Converter For Mac - Unregis

- In component configuration we have to configure all the respective properties by using properties tag.

- In hibernate mapping file, we have to provide all the component property configuration immediately after the id property configuration. component property configuration should not be provided after the normal properties configuration.

componentMappingApp

- Employee.java
- Account.java
- Address.java ✓
- Employee.hbm.xml
- hibernate.cfg.xml
- clientApp.java ✓

JPG to PDF Converter For Mac - Unregis

In database

```
SQL> create table employee (eno number, ename
 varchar2(5), accNo varchar2(3), accName varchar2(5),
 accBranch varchar2(10), pro varchar2(5),
 street varchar2(10), city varchar2(10));
SQL> desc employee;
```

Employee.java

```
public class Employee
    {
        private int eno;
        private String ename;
        private Account acc;
        private Address addr;
    public void setEno(int eno)
        {  this.eno = eno;
        }
    public int getEno()
        {  return eno;
        }
    public void setEname(String ename)
        {  this.ename = ename;
        }
    public String getEname()
        {  return ename;
        }
    public void setAcc(Account acc)
        {  this.acc = acc;
        }
```

JPG to PDF Converter For Mac - Unregis

```
public Account getAcc()
        { return acc;
        }
public void setAddr(Address addr)
        { this.addr = addr;
        }
public Address getAddr()
        { return addr;
        }
}
```

Account.java

```
public class Account
{
    private String accNO;
    private String accName;
    private String accBranch;
public void setAccNo(String accNO)
        { this.accNO = accNO;
        }
public String getAccNO()
        { return accNO;
        }
public void setAccName(String accName)
        { this.accName = accName;
        }
public String getAccName()
        { return accName;
        }
```

JPG to PDF Converter For Mac - Unregis

```
public void setAccBranch(String accBranch)
        { this.accBranch=accBranch;
        }
public String getAccBranch()
        { return accBranch;
        }
    }
```

Address.java

```
public class Address
    { private String pno;
                     street;
                     city;
    public void setpno(String pno)
        { this.pno=pno;
        }
    public String getpno()
        { return pno;
        }
    public void setStreet(String street)
        { this.street=street;
        }
    public String getStreet()
        { return street;
        }
    public void setCity(String city)
        { this.city=city;
        }
```

JPG to PDF Converter For Mac - Unregis

```
public String getCity()
        {return city;
        }
    }
```

Employee.hbm.xml

```
<DOCTYPE--->
<hibernate-mapping>
<class name="Employee" table="employee">
  <id name="eno"/>
<component name="acc" class="Account">
<property name="accNo"/>
<property name="accName"/>
<property name="accBranch"/>
</component>
<component name="addr" class="Address">
    <property name="pno"/>
    <property name="street"/>
    <property name="city"/>
    </component>
<property name="enamd"/>
  </class>
</hibernate-mapping>
```

JPG to PDF Converter For Mac - Unregis

hibernate.cfg.xml

```
<!DOCTYPE ---1">
<hibernate-configuration>
<session-factory>
<property name="connection.driver_class">
oracle.jdbc.driver.OracleDriver</property>
<property name="connection.url">
jdbc:oracle:thin:@localhost:1521:xe</property>
<property name="connection.user">
system</property>
<property name="connection.password">
durga</property>.
<property name="hibernate.dialect">
org.hibernate.dialect.OracleDialect</property>
<property name="show_sql">true</property>
<mapping resource="Employee.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

ClientApp.java

```
import org.hibernate.*;
import org.hibernate.cfg.*;
public class ClientApp
{
public static void main(String args[]) throws
exception
{
```

JPG to PDF Converter For Mac - Unregis

```
Configuration cfg=new Configuration();
        cfg.configure();
SessionFactory sf=cfg.buildSessionFactory();
  Session s=sf.openSession();
   Account acc=new Account();
      acc.setAccNo("a1");
      acc.setAccName("aaa");
      acc.setAccBranch("sonagar");
     Address addr=new Address();
        addr.setPno("p1");
        addr.setStreet("mgroad");
        addr.setCity("secbad");
      Employee e=new Employee();
          e.setEno(111);
          e.setEname("aaa"); e.setAcc(acc);
                              e.setAddr(addr);
   Transaction tx=s.beginTransaction();
          s.save(e);
          tx.commit(); sop("employee inserted
                                successfully");
      }
  }
```

2 → Compilation

DSI hbm4 | Component mappings → hbr set Enos.bat
→ javac *.java

DSI

JPG to PDF Converter For Mac - Unregis

> java ClientApp

employee inserted successfully

SQL

SQL> select * from employee;

ENO	ENAME	ACCID	ACCNAME	ACCBRANCH	DNO
111	aaa	a1	aaa	secunderabad	D1

STREET	CITY
mgroad	hyderabad

SQL>

2 → In hibernate applications, to represent component mapping EJB3 has provided the following two annotations along with the basic annotations in javax.persistence package.

- @Embedded
- @Embeddable.

where @Embedded annotation can be used to represent component property.

where @Embeddable annotation can be used to represent a pojo class referred by component property.

— The following hibernate application uses annotations to retrieve the data from employee table (this pgm uses the modifications made on the previous program)

JPG to PDF Converter For Mac - Unregis

hibernate-cfg.xml <DocType —>

```
<hibernate-configuration>
<session-factory>
    ==
<mapping class="Employee"/>
</session-factory>
</hibernate-configuration>
```

Employee.java

```
import javax.persistence.*;
  @Entity
  @Table(name="employee")
public class Employee
  {  @Id
     @Column(name="eno")
       cap
     private int eno;
     @Colum(name="ename").
      private String ename;
     @Embedded
      private Account acc;
     @Embedded
      private Address addr;
    public void setEno(int eno)
        {
          =
        }
      ==
  } //Employee class.
```

JPG to PDF Converter For Mac - Unregis

Account.java

```
import javax.persistence.*;
@Embeddable
public class Account
{
    @Column(name="accNo")
    private String accNo;
    @Column(name="accName")
    private String accName;
    @Column(name="accBranch")
    private String accBranch;
    public void setAccNo(String accNo)
    {
    }
}//Account class.
```

Address.java

```
import javax.persistence.*;
@Embeddable
public class Address
{
    @Column(name="pno")
    private String pno;
    @Column(name="street")
    private String street;
    @Column(name="city")
    private String city;
```

JPG to PDF Converter For Mac - Unregis

```
public void setPro(String pro)
{ =
}

}// class Address.
```

ClientApp.java

```
import org.hibernate.*;
        .cfg.*;

public class ClientApp
{
public static void main(String args[]) throws
                                Exception
{
Configuration cfg=new AnnotationConfiguration
                                ();
    cfg.configure();
SessionFactory sf=cfg.buildSessionFactory();
  Session s=sf.openSession();
Employee e=(Employee) s.get(Employee.class,
                                1L);
    SOP("Employee Details");
      SOP("----------");
  SOP("employee Number--"+e.getEno());
  SOP("employee Name--"+e.getEname());
    SOP("Account Details");
    SOP("----------");
      Account acc=e.getAcc();
  SOP("ACCNO--"+acc.getAccNO());
```

JPG to PDF Converter For Mac - Unregis

```
sop("AccName--"+acc.getAccName());
sop("AccBranch--"+acc.getAccBranch());
    Address addr=e.getAddr();
  sop("Address Details");
    sop("----------");
sop("pno:--"+addr.getpno());
sop("street"+addr.getStreet());
sop("city--"+addr.getCity());
  }
}
```

JPG to PDF Converter For Mac - Unregis

Inheritance mapping

Date: 20th May, 2012 Sun

— In general in enterprise application development it is essential to use object oriented model and relational model to represents the data.

— In object oriented model, we will represent all the entities in the form of classes but in the case of relational data model, we will represent all the entities in the form of tables.

— In object oriented data model to improve code reusability we will provide inheritance relation between entity classes. To achieve the same at relational data model, we will provide primary key, foreign key relationships b/w the tables and some other mechanisms.

Due to the above subtypes mismatch between object oriented data model and relational data model data persistency is very much complex in enterprise applications.

In the above contexts to simplify data persistency hibernate has provided ~~the following~~ ~~phase~~ Inheritance mapping as an alternative.

— As part of inheritance mapping, hibernate has provided the following 3 types of solutions

1. Table per class hierarchy.
2. Table per sub class
3. Table per concrete class.

JPG to PDF Converter For Mac - Unregis

```
class Account
{ String accNo;
  String accName;
}
```

extends

```
class Employee Account
{
  String eId;
  String eName;
}
```

extends

```
class Student Account
{
  String sId;
  String sBranch;
}
```

accNo: a1
accName: aaa
eId: e1
eName: Vijay

ea

accNo: a2
accName: bbb
sId: s1
sBranch: CS

sa

s.save(ea);
s.save(sa);

① Table per class hierarchy
② Table per sub class
③ Table per concrete class

① Account

accNo	accName	eId	eName	sId	sBranch	type
a1	aaa	e1	Vijay	null	null	Emp
a2	bbb	null	null	s1	CS	Std

discriminator column

② Account

accNo	accName
a1	aaa
a2	bbb

join column

join column

Employee Account

accNo	eId	eName
a1	e1	Vijay

Student Account

accNo	sId	sBranch
a2	s1	CS

③ employeeAccount

accNo	accName	eId	eName
a1	aaa	e1	Vijay

StudentAccount

accNo	accName	sId	sBranch
a2	bbb	s1	CS

JPG to PDF Converter For Mac - Unregis

Table per class hierarchy:

In table per class hierarchy inheritance mapping, we will define a single table for the entire inheritance class hierarchy. ~~that is~~ In table per class hierarchy inheritance mapping we will define a separate column in the database table for each and every property from super class and from all the sub classes.

2→ In table per class hierarchy inheritance mapping we will define a separate column in database table to represents the type of records called as descriminator column.

— In table per class hierarchy inheritance mapping when we store a sub class object on to the database table hibernate software will store the respective properties data in the respective columns and it will store the respective descriminator value in the descriminator column as per the mapping file configuration.

2→ If we want to provide table per class hierarchy inheritance mapping in hibernate application then we have to use the following xml tags in mapping file.

JPG to PDF Converter For Mac - Unregis

Table per class hierarchy

Class:

Account → Account

Id: accNo ← → accNo

discriminator:

column ← → Type.

property:

accName ← → accName.

Subclass:

name → Employee Account

discriminator-value: Employee.

property: eId ← → eId

property: edept ← → edept

Sub class: name → Student Account

discriminator-value: Student

property: sId ← → sId

property: sbranch ← → sbranch

JPG to PDF Converter For Mac - Unregis

```
<hibernate-mapping>
  <class name="__" table="__">
  <id name="__" column="__"/>
  <discriminator column="__"/>
  <property name="__" column="__"/>
  ==========

  <subclass name="__" discriminator-value="__">
  <property name="__" column="__"/>
  ==========
    </subclass>
  ==========
  </class>
</hibernate-mapping>
```

2⇒ where <discriminator> tag can be used to configure discriminator column which we defined in the respective database table.

where column attribute in discriminator tag will take discriminator column name.

where <subclass> tag can be used to configure a subclass and its properties in mapping file.

name attribute in <subclass> tag will take the fully qualified name of the respective sub class.

- where discriminator-value attribute in <subclass> tag will take the respective discriminator value.

JPG to PDF Converter For Mac - Unregis

SQL> create table account (accNo varchar2(5), accName varchar2(7), eid varchar2(3), edept varchar2(5), sid varchar2(8), sbranch varchar2(5), type varchar2(10));

Table created.

SQL> commit;

Commit completed.

SQL> desc account;

accNo
accName
eid
edept
sid
sbranch
type

JPG to PDF Converter For Mac - Unregis

tableperclassapp

- Account.java
- EmployeeAccount.java
- StudentAccount.java
- Account.hbm.xml
- hibernate.cfg.xml
- ClientApp.java ✓

Account.java

```
public class Account
    { private String accNO;
      private String accName;
    public void setAccNo(String accNO)
        { this.accNO = accNO;
        }
    public String getAccNo()
        { return accNO;
        }
    public void setAccName(String accName)
        { this.accName = accName;
        }
    public String getAccName()
        { return accName;
        }
    }
```

JPG to PDF Converter For Mac - Unregis

EmployeeAccount.java

```
public class EmployeeAccount extends Account
{ private String eid;
  private String edept;
public void setEid(String eid)
    { this.eid = eid;
    }
public String getEid()
    { return eid;
    }
public void setEdept(String edept)
    { this.edept = edept;
    }
public String getEdept()
    { return edept;
    }
}
```

StudentAccount.java

```
public class StudentAccount extends Account
{ private String sid;
  private String sbranch;
public void setSid(String sid)
    { this.sid = sid;
    }
public String getSid()
    { return sid;
    }
```

JPG to PDF Converter For Mac - Unregis

```
public void setBranch(String sbranch)
        {  this.sbranch = sbranch;
        }
public String getBranch()
        {  return sbranch;
        }
}
```

Account.hbm.xml

```
<!DOCTYPE — —>
<hibernate-mapping>
<class name="Account" table="account"
 <id name="accNo"/>
 <discriminator column="type"/>
 <property name="accName"/>
 <subclass name="EmployeeAccount"
    discriminator-value="Employee"/>
  <property name="eId"/>
  <property name="edept"/>
  </subclass>
 <subclass name="StudentAccount"
    discriminator-value="Student"/>
    <property name="sId"/>
    <property name="sbranch"/>
    </subclass>
  </class>
  </hibernate-mapping>
```

JPG to PDF Converter For Mac - Unregis

hibernate.cfg.xml :-

```
<!DOCTYPE ---">
<hibernate-configuration>
<session-factory>
<property name="connection.driver_class">
oracle.jdbc.driver.OracleDriver </property>
<property name="connection.url">jdbc:oracle:
thin:@localhost:1521:xe </property>
<property name="connection.user">system</property>
<property name="connection.password">durga
</property>
<property name="hibernate.dialect">
org.hibernate.dialect.OracleDialect </property>
<property name="show_sql">true</property>
<mapping resource="Account.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

CreateApp.java :-

```
import org.hibernate.*;
import org.hibernate.cfg.*;
public class CreateApp
{
public static void main(String[] args) throws
                                Exception
{
Configuration cfg = new Configuration();
cfg.configure();
```

JPG to PDF Converter For Mac - Unregis

```
SessionFactory sf = cfg.buildSessionFactory();
Session s = sf.openSession();
EmployeeAccount ea = new EmployeeAccount();
        ea.setAccNo("a1");
        ea.setAccName("aaaa");
        ea.setFld("e1");
        ea.setFdept("Admin");
StudentAccount sa = new StudentAccount();
        sa.setAccNo("a2");
        sa.setAccName("bbbb");
        sa.setSfld("s1");
        sa.setSbranch("CS");
        Transaction tx = s.beginTransaction();
            s.save(ea);
            s.save(sa);
            tx.commit();
SOP("Employee Account Persisted Successfully");
SOP("Student Account Persisted Successfully");
    }
}
```

JPG to PDF Converter For Mac - Unregis

Ob/ bbnu\ tabledescribeaccount> java * -java
> java clientApp

Employee account inserted successfully
student account inserted successfully

SQL> select * from account;

accno	accname	eid	ecdept	sid	sbranch	Type
a1	aaa	e1	Admin			Employee.
a2	bbb			s1	CS	student

JPG to PDF Converter For Mac - Unregis

— To represent table per class hierarchy inheritance mapping EJB3 has provided the following

① @Inheritance(-)
② @DiscriminatorColumn(-)
③ @DiscriminatorValue(-)

— where @Inheritance(-) annotation will represent a particular inheritance mapping mechanism. To specify, table per class hierarchy, inheritance mapping we have to use SINGLE_TABLE constant from InheritanceType enum.

— @DescriminatorColumn(-) annotation will represent descriminator column.

— where @DescriminatorValue(-) annotation can be used to specify descriminator value at the the respective corresponding sub class.

2 → Now, I will retrieve the data from account table by using annotations in java programs.

hibernate.cfg.xml

```
<hibernate-configuration>
    <session-factory>
==
<mapping class="StudentAccount"/>
<mapping class="EmployeeAccount"/>
</session-factory>
</hibernate-configuration>
```

JPG to PDF Converter For Mac - Unregis

Account.java

```
import javax.persistence.*;
@Entity
@Table(name="account")
@Inheritance(strategy=InheritanceType.
                        SINGLE_TABLE)
@DiscriminatorColumn(name="type")
public class Account
  {
    @Id
    @Column(name="accNo")
    private String accNo;
    @Column(name="accName")
    private String accName;
    public void setAccNo(String accNo)
        { this.accNo= accNo;
        }
  }
```

setter and getter
manually

Employee.java

```
import javax.persistence.*;
@Entity
@DiscriminatorValue("employee")
public class EmployeeAccount extends Account
  {
    @Column(name="eid")
    private String eid;
```

JPG to PDF Converter For Mac - Unregis

```
@Column(name="edept")
private String edept;
public void setEid(String eid)
        { this.eid=eid
        }
        ==== setter and getter method
             of property
}
```

Student Account.java

```
import javax.persistence.*;
@Entity
@DescriminatorValue("Student")
public class StudentAccount extends Account
{
    @Column(name="sid")
    private String sid;
    @Column(name="sbranch")
      private String sbranch;
    public void setSid(String sid)
        { this.sid=sid;
        }
    public String getSid()
        { return sid;
        }
    public void setBranch(String sbranch)
        { this.sbranch=sbranch;
        }
    public String getBranch()
        { return sbranch;
        }
}
```

JPG to PDF Converter For Mac - Unregis

ClientApp.java

```
import org.hibernate.*;
import org.hibernate.cfg.*;
public class ClientApp
{
public static void main(String[] args) throws
                                        Exception
{
Configuration cfg = new AnnotationConfiguration();
    cfg.configure();
SessionFactory sf = cfg.buildSessionFactory();
    Session s = sf.openSession();
EmployeeAccount ea = (EmployeeAccount)
        s.get("EmployeeAccount", "a1");
  SOP("Employee Account Details");
    SOP("------------------------");
  SOP("AccNo-" + ea.getAccNo());
  SOP("AccName--" + ea.getAccName());
  SOP("EId----" + ea.getEId());
  SOP("EDept--" + ea.getEdept());

  ~~SOP(~~
StudentAccount sa = (StudentAccount) s.get(
                    "StudentAccount", "a2"),
    SOP("Student Account details");
    SOP("------------------------");
```

JPG to PDF Converter For Mac - Unregis

```
sop("Acc No --" + sa.getAccNo());
sop("Acc Name --" + sa.getAccName());
sop("Bal --" + sa.getBal());
sop("Branch --" + sa.getBranch());
  }
}
```

JPG to PDF Converter For Mac - Unregis

Table per Sub class Inheritance mapping

In case of table per subclass inheritance mapping, we will prepare a separate table of each and every class available in Inheritance Hierarchy.

— In case of table per subclass inheritance mapping we will provide join column between parent table and the respective child table.

JPG to PDF Converter For Mac - Unregis

class Employee
{ String eId;
String ename;
}

extends extends

class Manager
{
String addr;
String qual;
}

class Accountant
{
String mobile;
String email;
}

eId: e1
ename: aaa
addr: Hyd
qual: MBA
m

s.save(m);
s.save(a);

eId: e2
ename: bbb
mobile: 91-96
email: abc@gmail.com
a

class: Employee ↔ employee
Id: eId ↔ eId
Property: ename ↔ ename

defined-subclass:
name = "Manager"
table = "manager"
key: column: eId
property: addr ↔ addr
qual ↔ qual.

defined-subclass:
name = "Accountant"
table = "accountant"
key: column = "eId"
property: mobile ↔ mobile
email ↔ email.

employee

eId	ename
e1	aaa
e2	bbb

manager

eId	addr	qual
e1	Hyd	MBA

accountant

eId	mobile	email
e2	91-96 96534	abc@gmail.

JPG to PDF Converter For Mac - Unregis

— If we want to implement table per subclass inheritance mapping in hibernate application we have to use the following tags in the respective mapping file.

```
<hibernate-mapping>
  <class name="_" table="_">
  <id name="_" column="_">
  <property name="_" column="_"/>
  ==
  <joined-subclass name="_" table="_">
    <key column="_"/>
    <property name="_" column="_"/>
    ==
    </joined-subclass>
    ==
  </class>
</hibernate-mapping>
```

— where <joined-subclass> tag can be used to represent subclass.

— where name attribute in joined-subclass tag will take the fully qualified name of the respective sub class.

— where table attribute in <joined-subclass> tag will take the respective table name.

— where <key> tag can be used to configure join-column.

— where column attribute in <key> tag will take join column name.

JPG to PDF Converter For Mac - Unregis

SQL> create table employee(eid varchar2(5)
primary key, ename varchar2(10));
Table created.

SQL> create table manager(eid varchar(5) primary key,
addr varchar(10), qual varchar2(10));
Table created.

SQL> create table accountent(eid varchar(5)
primary key, mobile varchar2(15), email
varchar2(15));
Table created.

SQL> commit;
Commit completed.

table persistenceclassapp
- Employee.java
- Manager.java
- Accountent.java
- Employee.hbm.xml
- hibernate.cfg.xml
- ClientApp.java

Employee.java

```
public class Employee
{ private String eid;
  private String ename;
  public void setEid(String eid)
      { this.eid=eid;
      }
  public String getEid()
      { return eid;
      }
```

JPG to PDF Converter For Mac - Unregis

```
public void setEname(String ename)
        { this.ename=ename;
        }
public String getEname()
        { return ename;
        }
}
```

Manager.java

```
public class Manager extends Employee
{ private String addr;
  private String qual;
public void setAddr(String addr)
        { this.addr=addr;
        }
public String getAddr()
        { return addr;
        }
public void setQual(String qual)
        { this.qual=qual;
        }
public String getQual()
        { return qual;
        }
}
```

JPG to PDF Converter For Mac - Unregis

Accountant.java

```
public class Accountant extends Employee
    {
        private String mobile;
        private String email;
public void setMobile(String mobile)
        {
            this.mobile = mobile;
        }
public String getMobile()
        {
            return mobile;
        }
public void setEmail(String email)
        {
            this.email = email;
        }
public String getEmail()
        {
            return email;
        }
    }
```

Employee.hbm.xml

```
<!DOCTYPE --->
<hibernate-mapping>
<class name="Employee" table="employee">
  <id name="eid"/>
  <property name="ename"/>
  <joined-subclass name="Manager"
                   table="manager">
    <key column="eid"/>
    <property name="addr"/>
    <property name="qual"/>
```

JPG to PDF Converter For Mac - Unregis

```
</joined-subclass>
<joined-subclass name="AccountDetail"
                 table="accountdetail">
  <key column="eid"/>
  <property name="mobile"/>
  <property name="email"/>
  </joined-subclass>
  </class>
  </hibernate-mapping>
```

hibernate.cfg.xml

```
<!DOCTYPE ->
<hibernate-configuration>
<session-factory>
  <property name="connection.driver_class">
oracle.jdbc.driver.OracleDriver </connection>
<property name="connection.url">jdbc:oracle:
thin:@localhost:1521:xe </property>
<property name="connection.user">system
<property name="connection.password">durga </property>
<property name="hibernate.dialect">
org.hibernate.dialect.OracleDialect </property>
<mapping resource="employee.hbm.xml"/>
</session-factory>
</hibernate-configuration>
```

JPG to PDF Converter For Mac - Unregis

ClientApp.java

```
import org.hibernate.*;
import org.hibernate.cfg.*;
public class ClientApp
{
public static void main(String[] args) throws
                                    Exception
{
Configuration cfg=new Configuration();
        cfg.configure();
SessionFactory sf=cfg.buildSessionFactory();
 Session s=sf.openSession();
  Manager m=new Manager();
      m.setEid("e1");
      m.setEname("aaa");
      m.setAddr("hyd");
      m.setQual("MBA");
    Accountent a=new Accountent();
        a.setEid("5")
        a.setEname("bob");
      a.setMobile("91-9676534531");
     a.setEmail("abc@gmail.com");
   Transaction tx=s.beginTransaction();
          s.save(m);
          s.save(a);
          tx.commit();
```

JPG to PDF Converter For Mac - Unregis

```
SOP("Manager Inserted Successfully");
SOP("Accountant Inserted Successfully");
    }
}
```

Output

```
sql> select * from employee;
```

eid	ename
e1	aaa
e2	bbb

```
sql> select * from manager;
```

JPG to PDF Converter For Mac - Unregis

2→ To represent tablepersubclass inheritance mapping EJB3 has provided the following two annotations.

① @Inheritance

② @PrimaryKeyJoinColumn(--)

— To represent tablepersubclass inheritance mapping, we have to use JOINED constant from InheritanceType enum as value to strategy parameter in @Inheritance annotation.

— where @PrimaryKeyJoinColumn(-) annotation can be used to configure joincolumn at each and every subclass.

hibernate.cfg.xml

```
<hibernate-configuration>
<session-factory>
=
=
=
<mapping class="Manager"/>
<mapping class="Accountent"/>
</session-factory>
</hibernate-configuration>
```

Employee.java

```
import javax.persistence.*;
@Entity
@Table(name="employee")
@Inheritance(strategy=InheritanceType.
                        JOINED)
```

JPG to PDF Converter For Mac - Unregis

```
public class employee
{ @Id
  @Column(name="eid")
  private String eid;
  @Column(name="ename")
  private String ename;
  public void setEid(String eid)
        ==== setter and getter
              methods of properties.
}
```

Manager.java

```
import javax.persistence.*;
@Entity
@PrimaryKeyJoinColumn(name="eid")
public class Manager extends employee
{ @Column(name="adder")
  private String adder;
  @Column(name="qual")
  private String qual;
  public void setAdder(String adder)
      { this.adder=adder;
      }
  ==========
  ==========  setter and getter methods
              of properties
}
```

JPG to PDF Converter For Mac - Unregis

Accountent.java

```
import javax.persistence.*;
@Entity
@PrimaryKeyJoinColumn(name="eid")
public class Accountent extends Employee
{
    @Column(name="mobile")
    private String mobile;
    @Column(name="email")
    private String email;
    public void setMobile(String mobile)
    {
        this.mobile=mobile;
    }
    (setter and getter methods of properties)
}
```

CallonApp.java

```
import org.hibernate.*;
import org.hibernate.cfg.*;

public class CallonApp
{
    Configuration cfg= new AnnotationConfiguration();
        cfg.configure();
    SessionFactory sf=cfg.buildSessionFactory();
        Session s = sf.openSession();
}

    Manager m =(Manager) s.getManager(
                    "Manager","e1");
    sop("Manager details");
    sop("----------");
```

JPG to PDF Converter For Mac - Unregis

```
System.out.println("Eid--"+m.getEid());
System.out.println("Ename--"+m.getEname());
System.out.println("Addr--"+m.getAddr());
System.out.println("Qual--"+m.getQual());
        System.out.println();
Accountant a=(Accountant)s.get("Accountant",id);
System.out.println("Account Details");
System.out.println("--------");
System.out.println("Eid--"+a.getEid());
System.out.println("Ename--"+a.getEname());
System.out.println("mobile--"+a.getMobile());
System.out.println("email--"+a.getEmail());
}
}
```

JPG to PDF Converter For Mac - Unregis

Table per concrete class

Date: 22nd may, 2012 Tue

Incase of table per concrete class inheritance mapping we will define a separate table for each and every sub class with the parent class properties and the respective sub class properties.

- To implement table per concrete class inheritance mapping not required to provide any extra mappings, it is possible to use directly class configuration with respect to the sub classes.
- To represent table per concrete class inheritance mapping hibernate has provided a separate configuration for sub classes that is union sub class.
- To implement table per concrete class inheritance mapping we have to use the following xml tags.

```
<hibernate-mapping>
    <class name="_" />
     <id name="_" column="_" />
     <property name="_" column="_" />
     ==
<union-subclass name="_" table="_" />
    <property name="_" column="_" />
    ==
     </union-subclass>

<class>
</hibernate-mapping>
```

JPG to PDF Converter For Mac - Unregis

- Where union-subclass tag can be used to configure a subclass in the table per concrete class inheritance mapping.
- Where name attribute in <union-subclass> tag will take the fully qualified name of the sub class.
- Where table attribute in union-subclass tag will take the respective table name.

```
class person
{
String pnme;
String paddr;
}
extends
class Customer
{
String cid;
String cmobile;
}
extends
class Employee
{
String eid;
String esal;
}
```

Table per concrete class

```
class:
    name = person
    Id:
      pname <-> pname
    property:
      paddr <-> paddr
union-subclass:
    name = "Customer"
          table = "Customer"
    property: cid <-> cid
      cmobile <-> cmobile
union-subclass:
    name = "Employee"
          table = "employee"
    property: eid <-> eid
      esal <-> esal
```

Customer

pname	paddr
abc	aaa.

cid	cmobile
9	91-9870

employee

pname	paddr
xyz	xyx.

eid	esal
9	900

pnme: abc
paddr: aaa
cid: 9
cmob: 98-76

pnme: xyz
paddr: xyx
eid: 9
esal: 900

s.save(c);
s.save(e);

JPG to PDF Converter For Mac - Unregis

table per concrete class app.

- person.java
- customer.java.
- employee.java.
- person.hbm.xml.
- clientapp.java.
- hibernate.cfg.xml ✓

SQL> create table customer (pname varchar2(5), paddr varchar2(15), cid varchar2(5), cmobile varchar2(15));

SQL> commit;

SQL> create table employee (pname varchar2(5), paddr varchar2(15), eid varchar2(10), esal number(5));

SQL> commit;

personojava:

```
public class person
{
    private String pname;
    private String paddr;
    public void setpname(String pname)
        {  this.pname = pname;
        }
    ========== (setter and getter methods of properties)
}
```

JPG to PDF Converter For Mac - Unregis

· Customer.java

```
public class Customer extends person
    { private String cid;
      private String cmobile;
      public void setCid(String cid)
          { this.cid = cid;
          }
      public String getCid()
          { return cid;
          }
      public void setCmobile(String cmobile)
          { this.cmobile = cmobile;
          }
      public String getCmobile()
          { return cmobile;
          }
    } //class
```

Employee.java

```
public class Employee extends person
    { private String eid;
      private int esal;
      public void setEid(String eid)
          { this.eid = eid;
          }
      public String getEid()
          { return eid;
          }
```

JPG to PDF Converter For Mac - Unregis

```
public void setEsal(int esal)
    { this.esal=esal;
    }
public int getEsal()
    { return esal;
    }
}//class
```

personal.hbm.mapping

```
<!DOCTYPE ---->
<hibernate-mapping>
<class name="person">
    <id name="pname"/>
    <property name="paddr"/>
<union-subclass name="Customer".
                table="customer">
<property name="cid"/>
<property name="cmobile"/>
</union-subclass>
<union-subclass name="Employee" table="employee">
    <property name="eid"/>
    <property name="esal"/>
    </union-subclass>
    </class>
    </hibernate-mapping>
```

JPG to PDF Converter For Mac - Unregis

hibernate.cfg.xml

```
<DOCTYPE --->
<hibernate configuration>
<session-factory>
<property name="connection.driver class>
oracle.jdbc.driver.OracleDriver </property>
<property name="connection.url">
jdbc:oracle:thin:@localhost:1521:xe </property>
<property name="connection.user"> system
<property name="connection.        </property>
                  password"> durga </property>
<property name="hibernate.dialect">
org.hibernate.dialect.Oracle9Dialect </property>
<mapping resource="person.hbm.xml"/>
</session-factory>
</hibernate configuration>
```

ClientApp.java

```
import org.hibernate.*;
                    .cfg.*;

public class ClientApp
{
  p s v m(SC) a) throws Exception
  {
    Configuration cfg = new Configuration();
                  cfg.configure();
```

JPG to PDF Converter For Mac - Unregis

```
SessionFactory sf = cfg.buildSessionFactory();
    Session s = sf.openSession();
      Customer c = new Customer();
            c.setname("aaa");
            c.setaddr("secunderabad");
            c.setId("c1")
            c.setmobile("91-9676534531");
      Employee e = new Employee();
            e.setname("bbb");
            e.setaddr("hyderabad");
            e.setId("e1");
            e.setsal(9000);
    Transaction tx = s.beginTransaction();
            s.save(c);
            s.save(e);
            tx.commit();
    sop("Customer inserted successfully");
    sop("Employee inserted successfully");
    }
}
```

Dsl hbn4\

> hbrsettings.bat
> javac *.java
> java CLIENTAPP

JPG to PDF Converter For Mac - Unregis