Java Training Center

(No.1 in Training & Placement)

Hibernate-3 Doc.

Master the Content…

Author

SomPrakashRai

Version mapping

- Version mapping is used to versioning the data or record of the table.
- This allows you to track the updating happened on various records of table.
- When you insert a new record, then version value will be 0.
- Whenever you update the record then version will be increased by one automatically.
- Version number will be provided by the Hibernate system automatically so don't include the version related variable in the constructor.

A) Tables required

Customer1

cid	Cname	Email	Phone	Version
1	Som	Jtc@jtc.com	99999	1

Example with hibernate core

B) Hibernate persistence classes

Customer.java

```
public class Customer {
        private int cid;
        …..
        private int version;
}
```

C) Hibernate mapping document

Customer.hbm.xml

```
<hibernate-mapping package="com.jtcindia.hibernate">
<class name="Customer" table="customer1">
        <id name="cid" column="cid" type="int">
                <generator class="increment" />
        </id>
        <version name="version" type="integer" column="version" />
        <property name="cname" />
        <property name="email" />
        <property name="phone" type="long" />
</class>
</hibernate-mapping>
```

Jtc19: Files required

1. Jtc 19A. java	2. Jtc19B. java
3. Customer.java	4. Customer. hbm. Xml
5. CHibernate Util.java	6. Hibernate.cfg. xml

JTC19A.java

```
package com.jtcicindia.hibernate;

import org.hibernate.*;

/*
*@Author:Som Prakash Rai
*@company:java Training center
*@see   :www.jtcindia.org
**/
public class Jtc19A {
public static void main(string[]args){
Transaction tx=null;
try{
        SessionFactory sf=CHibernateUtil.getSessionFactory();
        Session session=sf.open session();
        tx=session.beginTransaction();
        Customer cust=new Customer("Jtc","Jtc@jtcindia.org",747474);
        Session.save(cust);
        tx.commit();
        session.close();
        System.out.println("record inserted");
        }catch(Exception e)
        {
                e.prinStackTrace();
                if (txl = null)
                        tx.rollback();
        }
    }
}
```

JTC19B.java

```
package com.jtcicindia.hibernate;

import org.hibernate.*;

/*
*@Author:Som Prakash Rai
*@company:java Training center
*@see   :www.jtcindia.org
**/
public class Jtc19B {
public static void main(string[]args){
        Transaction tx=null;
        try{
                SessionFactory sf=CHibernateUtil.getSessionFactory();
                Session session=sf.open session();
                tx=session.beginTransaction();
                customer cust = (customer) session.load(Customer.class,1);
                cust.setCename("Praveen");
                session.update(cust);
                tx.commit();
                session.close();
                System.out.println("record inserted");
                }catch(Exception e){
                        e.prinStackTrace();
                        if (txl = null)
                                tx.rollback();
                        }
        }
}
```

Customer.java

```
package com.jtcindia.hibernate;
public class customer{
        private int cid;
        private String cname;
```

```
	private String email;
	private long phone;
	private int version;

public Customer() {}

	public Customer(String cname, String email, long phone) {
		this.cname = cname;
		this.email = email;
		this.phone = phone;
	}
	// setters and getters
}
```

Customer.hbm.xml

```
<hibernate-mapping package="com.jtcindia.hibernate">
	<class name="Customer" table="customer1">
		<id name="cid" column="cid" type="int">
			<generator class="increment" />
		</id>
		<version name="version" type="integer" column="version" />
		<property name="cname" />
		<property name="email" />
		<property name="phone" type="long" />
	</class>
</hibernate-mapping>
```

Note : Hibernateutil.java and hibernate.cfg.xml same as Jtc1.

Example with hibernate annotation

B) Hibernate persistence classes

Customer. java

```
@Entity
@ Table (name="customer1")
public class Customer {
...
@Version
private int version;
```

```
                    }
```

Jtc20: Files required

1. **Jtc 20A. java**	2. **Jtc20B. java**
3. **Customer.java**	4. **Hibernate Util.java**
5. **Hibernate.cfg. xml**	

Jtc20A.java

```
package com.jtcicindia.hibernate;
import org.hibernate.*;
/*
*@Author:Som Prakash Rai
*@company:java Training center
*@see   :www.jtcindia.org
**/
public class Jtc20A{
public static void main(string[]args){
          Transaction tx=null;
          try{
                    SessionFactory sf=CHibernateUtil.getSessionFactory();
                    Session session=sf.open session();
                    tx=session.beginTransaction();
                    Customer cust=new customer("Prakash",Prakash@jtcindia.org,858585);
                    Session.save(cust);
                    tx.commit();
                    session.close();
                    System.out.println("record inserted");
          }catch(Exception e){
                    e.prinStackTrace();
                    if(txl=null)
                              tx.rollback();
                    }
}
```

Jtc20B.java

```
package com.jtcicindia.hibernate;
```

```
import org.hibernate.*;

/*
*@Author:Som Prakash Rai
*@company:java Training center
*@see   :www.jtcindia.org
**/
public class Jtc20B {
public static void main(string[]args){
        Transaction tx=null;
        try{
                SessionFactory sf=CHibernateUtil.getSessionFactory();
                Session session=sf.open session();
                tx=session.beginTransaction();
                Customer cust = (customer) Session.load(customer.class, 2);
                Cust.setEmail("JtcPrakash@jtcindia.org");
                session.save(cust);
                tx.commit();
                session.close();
                System.out.println("record inserted");
        }catch(Exception e) {
                e.prinStackTrace();
                if(txl=null)
                        tx.rollback();
        }
}
```

Customer.java

```
package com.jtcindia.hibernate;
import javax.persistence.*;
/*
*@Author:Som Prakash Rai
*@company:java Training center
*@see  :www.jtcindia.org
**/
@Entity
@Table(name =”customer1”)
public class customer {
        @Id
        @Column(name="cid")
        @GeneratedValue(strategy=GenerationType.AUTO)
        private int cid;

        @Column(name="cname")
        private string cname;

        @Column(name="email")
        private string email;

        @Column(name="phone")
        private string phone;

        @Version
        private int version;

        public customer() {}

        public Customer(String email, long phone, String cname) {
                this.cname = cname;
                this.email = email;
                this.phone = phone;
        }
        // Setters and Getters
}
```

Note : Hibernateutil.java and hibernate.cfg.xml same as Jtc1.

Timestamp mapping

- Timestamp mapping is used to Time stamping the data i.e. you can track when the record was updated recently.
- Hibernate System is responsible for providing the value for Timestamp variable. So don't include in constructor.

A) Tables required

1) **Customer2**

cid	Cname	Email	Phone	Tstamp
1	som	som@Jtc.com	99999	…..

Example with hibernate core

B) Hibernate persistence classes

1) Customer.java

```
public class customer {
        private int cid;
        ….
        privateTimestamp stamp; //java. sql. Timestamp
}
```

C) Hibernate mapping document

2. Customer.hbm.xml

```
<hibernate-mapping package=”com.jtcindia.hibernate”>
        <class name=”Customer” table=”customer2”>
                <id name=”cid” column=”cid” type=”int”>
                        <generator class=”increment” />
                </id>
                <timestamp name=”tstamp” column=”tstamp” />
                <property name=”cname” />
                <property name=”email” />
                <property name=”phone” type=”long” />
        </class>
</hibernate-mapping>
```

Jtc21: Files required

1. Jtc21A. java same as Jtc19A	**2. Jtc21B. java same as Jtc19B**
3. Customer.java	**4. Customer. hbm. Xml**
5. CHibernate Util.java	**6. Hibernate.cfg. xml**

Customer.java

```
package com.jtcindia.hibernate;

public class Customer{
```

```
                private int cid;
                private String cname;
                private String email;
                private long phone;
                private int version;

                public Customer() {}
                public Customer(String cname, String email, long phone){
                        this.cname=cname;
                        this.email=email;
                        this.phone=phone;
                }
  //setters and getters
}
```

Customer.hbm.xml

```
<hibernate-mapping package=”com.jtcindia.hibernate”>
        <class name=”Customer” table=”customer2”>
                <id name=”cid” column=”cid” type=”int”>
                        <generator class=”increment” />
                </id>
                <timestamp name=”tstamp” column=”tstamp” />
                <property name=”cname” />
                <property name=”email” />
                <property name=”phone” type=”long” />
        </class>
</hibernate-mapping>
```

Example with hibernate annotation

B) Hibernate persistence classes

1) Customer. java

```
                        @Entity
                        @ Table (name=”customer2”)
                        public class Customer {
                                …
                                @Temporal (Temporal Type. TIMESTAMP)
                                private java. Util. Data  tstamp;
                }
```

Jtc22: Files required

1. Jtc 22A. java same as Jtc20A	**2. Jtc20B. java same as Jtc20B**
3. Customer.java	**4. AHibernate Util.java**
5. Hibernate.cfg. xml	

Customer.java

```
package com.jtcindia.hibernate;
import javax.persistence.*;
/*
*@Author:Som Prakash Rai
*@company:java Training center
*@see :www.jtcindia.org
**/

@Entity
@Table(name=”Customer1”)
public class Customer{
        @Id
        @Column(name=”cid”)
        @GeneratedValue(strategy=GenerationType.AUTO)
        private int cid;
        @Column(name=”cname”)
        private String cname;
        @Column(name=”email”)
        private String email;
        @Column(name=”phone”)
        private String cname;
        @Version
        private int version;
        public Customer() {}
        public Customer(String email, long phone, String cname){
                this.cname=cname;
                this.email=email;
                this.phone=phone;
        }
       //Setters and Getters
```

}

Example Using All Hibernate mapping

Consider the following book store application.

Bookstore has the following Entities.

- Author
- Book
- Customer
- CreditCard

- Order
- OrderItem
- ShippingAddress

Example with hibernate core
Jtc23:Files required

1. Jtc23A.java	2. Jtc23B.java
3. Jtc23C.java	4. Jtc23.java
5. Author.java	6. Book.java
7. CreditCard.java	8. Customer.java
9. GoldCustomer.java	10. SilverCustomer.java
11. Order.java	12. OrderItem. Java
13. ShippingAddress.java	14. CHibernateUtile .java
15. Author. hbm.xml	16. Book. hbm. Xml
17. CreditCard. hbm. Xml	18. Customer. hbm. xml
19. Order. hbm.xml	20. Order Item.hbm.xml
21. ShippingAddress.hbm. xml	22. Hibernate. cfg.xml

Jtc23A.java

```
package com.jtcicindia.hibernate;
import java.util.*;
import org.hibernate.*;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See    :www.jtcindia.org
**/
public class Jtc23A{
        public static void main(string[]args){
                Transaction tx=null;
                try{
                        SessionFactory sf=CHibernateUtil.getSessionFactory();
                        Session session=sf.open session();
                        tx=session.beginTransaction();
                        List<String>quails=new ArrayList<String>();
                        quails.add("M.sc");
                        quails.add("M.C.A");
                        quails.add("M.Tech");
                        Set<String> exps=new HashSet<String>();
                        exps.add("SUN");
                        exps.add("IBM");
                        exps.add("Oracle");
                        Author a1=new Author("Jtc","Jtc@jtc",123,new data(),qualis,exps);
                        session.save(a1);
                        Author a2=new Auther("Som","Som@jtc",321,new
date(),quails,exps);
                        session.save(a2);
                        Book b1=new book("master java",99.99,1,"jtc");
                        session.save(b1);
                        Book b1=new book("master Hiber",99.99,1,"jtc");
                        session.save(b2);
                        Book b1=new book("master Spring",99.99,1,"jtc");
                        session.save(b3);
                        Set<Author>as1=new HashSet<Author>();
                        as1.add(a1);
                        Set<Author>as2=new HashSet<Author>();
                        as2.add(a1);
                        as2.add(a2);
                        b1.setAuthors(as1);
                        b2.setAuthors(as2);
                        b3.setAuthors(as2);
                        tx.commit();
                        session.close();
                        System .out.println(record inserted");
                }catch(Exception e) {
                        e.printStack Trace();
                        if(tx!=null)
                                tx.rollback();
                }
        }
}
```

Jtc23B.java

```
package com.jtcicindia.hibernate;
import java.util.*;
import org.hibernate.*;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See   :www.jtcindia.org
**/
public class Jtc23B{
        public static void main(String[]args){
                Transaction tx=null;
                try{
                        SessionFactory sf=CHibernateUtil.getSessionFactory();
                        Session session=sf.open session();
                        tx=session.beginTransaction();
                        CreditCard cc1=new creditcard(1111,"visa",999,new Data());
                        session.save(cc1);
                        SilverCustomer c1=new
SilverCustomer("jtc","jtc@jtc",1234,"Noida",10,"c@jtc");
                        c1.setCcard(cc1);
                        session.save(c1);
                        Map<String,String>refs=new HashMap<String,String>();
                        refs.put("AA","11");
                        refs.put("BB","22");
                        CreditCard cc2=new credltCard(2222,"jtcprak",999, new Date());
                        session.save(cc2);
                        GoldCustomer c2=new
GoldCustomer("Praveen","Praveen@jtc",1234,4321,refs,1000);
                        c2.setCcard(cc2);
                        session.save(c2);
                        tx.commit();
                        session.close();
                        System .out.println(record inserted");
                }catch(Exception e) {
                        e.printStack Trace();
                        if(tx!=null)
                                tx.rollback();
                }
        }
}
```

Jtc23C.java

```
package com.jtcicindia.hibernate;
import java.awt.print.Book;
import java.util.*;
import org.hibernate.*;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See    :www.jtcindia.org
**/
public class Jtc23c{
        public static void main(string[]args){
                Transaction tx=null;
                try{
                        SessionFactory sf=CHibernateUtil.getSessionFactory();
                        Session session=sf.openSession();
                        tx=session.beginTransaction();
                        Customer cust=(customer)session.load(customer.class,1);
                        Book b1=(Book)session.load(Book.class, 1);
                        OrderItem oi1=new OrderItem(1100);
                        oi1.setBook(b1);
                        session.save(oi1);

                        Book b2=(Book)session.load(Book.class, 2);
                        OrderItem oi2=new OrderItem(2200);
                        oi2.setBook(b2);
                        session.save(oi2);
                        ShippingAddress add=new
ShippingAddress("SEC2","NOIDA","UP');
                        session.save(add);
                        Set<OrderItem> orderItems=new HashSet<OrderItem>();
                        orderItems.add(oi1);
                        orderItems.add(oi2);
                        Order order =new Order(3300.0,new Date(),"New");
                        order.setAddress(add);
                        order.setOrderltems(orderltems);
                        order.setCustomer(cust);
```

```
				session.save(order);
				tx.commit();
				session.close();
				System .out.println(record inserted”);
			}catch(Exception e) {
				e.printStack Trace();
				if(tx!=null)
					tx.rollback();
				}
		}
}
```

Jtc23D.java

```
package com.jtcicindia.hibernate;

import org.hibernate.*;

public class Jtc23C {
	public static void main(string[]args){
		Transaction tx=null;
		try{
			SessionFactory sf=CHibernateUtil.getSessionFactory();
			Session session=sf.openSession();
			tx=session.beginTransaction();
			Customer cust = (Customer) session.load(customer.class,1);
			System.out.printin(“CustomerInfo”);

	System.out.println(cust.getCid()+”\t”+cust.getCname()+”\t”+cust.getEmail()+”\t”
+ cust.getPhone());
			CreditCard cc=cust.getCcard();
			System.out.println("CCinfo:");
			System.out.println(add.getCcid()+”\t”+add.getCcno()+ “\t”+
	add.getCode()+”\t”+add.getCtype();
			Set<Order> ods=cust.getOrders();
			for(Order o:ods){
				System.out.println(“order info”);

	System.out.println(o.getOrderId()+"\t"+o.getTotalQty()+”\t”+o.getTotalCost()”\t”
+o.getStatus());
```

```
                        ShippingAddress add =o.getAddress();
                        System.out.printIn("Shiping Address;#"+o.getOrderId());

            System.out.println(add.getAddid()+"\t"+add.getStreet()+"\t"+add.getCity()+"\t"+a
dd.getState());
                        Set<OrderItem> items = o.getOrderItems();
                        System.out.println("Order items:#"+"\t"+o.getOrderId());
                        for(OrderItem oit:items){

            System.out.println(oit.getOtid()+"\t"+oit.getCost()+"\t"+oit.getQty()+"\t"+ooit.get
Book().getBid());;
                        }

                    }
                    tx.commit();
                    session.close();
                    System.out.println("Record Inserted");
                }catch(Exception e){
                    e.printStackTrace();
                    if(tx!=null)
                        tx.rollback();
                }
            }
}
```

Author.java

```
package com.jtcicindia.hibernate;
import java.util.*;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See    :www.jtcindia.org
**/
public class Author{
        private int aid;
        private String aname;
        private String email;
        private Date dob;
```

```
	private long phone;
	private List<String> quails;
	private Set<String>exps;
	private Set<Book>books;

	public Author() {}

	public Author (String aname, String email, String email, Date dob,List<String>
qualis,Set<String> exps){
		this. aname = aname;
		this.email=email;
		this.phone=phone;
		this.dob=dob
		this.quails=quails;
		this.exps=exps;
	}
//seters and getters
}
```

Book.java

```
package com.jtcicindia.hibernate;
import java.util.*;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See  :www.jtcindia.org
**/
public class Book{
	private int bid;
	private String bname;
	private double cost;
	private int edition;
	private String pub;
	private Set<Author> authors;

	public Book() {}

	public Book(String bname,double cost, int edition, String pub) {
		this.bname=bname;
```

```
                        this.cost=cost;
                        this.edition=edition;
                        this.pub=pub;
            }
  //Setters and getters
}
```

CreditCard.java

```
package com.jtcindia.hibernate;
import java.util.*;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See          :www.jtcindia.org
**/
public class CreditCard{
            private int ccid;
            private int ccno;
            private String ctype;
            private int code;
            private Date expDate;
            private Customer customer;

            public CreditCard(){}
            public CreditCard(int ccno, String ctype, int code, Date expDate) {
                        this.ccno=ccno;
                        this.ctype=ctype;
                        this.code=code;
                        this.expDate=expDate;
            }
//Setters and Getters
}
```

Customer.java

```
package com.jtcicindia.hibernate;
import java.util.*;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
```

```
*@See   :www.jtcindia.org
**/
public abstract class Customer{
        private int cid;
        private String cname;
        private String email;
        private long phone;
        private CreditCard ccard;
        private Set<Order> orders;

        public Customer() {}

        public Customer(String cname, String email, String email, String phone){
                this. cname = cname;
                this.email=email;
                this.phone=phone;
        }
//seters and getters
}
```

GoldCustomer.java

```
package com.jtcindia.hibernate;
import java.utll.Map;
public class GoldCustomer extends Customer{
        private long ophone;
        private Map<String,String>refs;
        private int points;
        public gold customer() {}
        public GoldCustomer(String cname, String email, long phone,long
ophone,Map<String, String>refs, int points){
                        super(cname,email,phone);
                        this.ophone=ophone;
                        this.refs=refs;
                        this.points=points;
        }
//setters and getters
}
```

SilverCustomer.java

```
package com.jtcicindia.hibernate;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See  :www.jtcindia.org
**/
public class SilverCustomer extends Customer{
	private String city;
	private int rpoints;
	private String oemail;

	public SilverCustomer() {}

	public SilverCustomer(String cname ,double email, long phone, String city, int
rpoints, String oemail) {
		super(cname,email,phone);
		this.city=city;
		this. rpoints =rpoints;
		this.oemail= oemail;
	}
  //Setters and getters
}
```

Order.java

```
package com.jtcicindia.hibernate;
import java.util.*;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See		:www.jtcindia.org
**/
public class Order{
	private int orderId;
	private int totalQty;
	private double totalCost;
	private Date orderDate;
	private String status;
	private Customer customer;
	private ShippingAddress address;
	private Set<Orderltem>orderItems;
	public Order() {}
	public Order(int totalQty, double totalCost, Date orderDate, String status){
		this.totalQty=totalQty;
		this.totalCost=totalCost;
		this.orderDate=orderDate;
		this.status=status;
	}
//setters and getters
}
```

OrderItem.java

```
package com.jtcicindia.hibernate;
import java.util.*;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See   :www.jtcindia.org
**/
public class OrderItem{
	private int otid;
	private int qty;
	private double cost;
	private Order order;
	private Book book;

	public OrderItem() {}

	public OrderItem (int qty, double cost){
		this.qty= qty;
		this.cost= cost;
	}
//setters and getters
}
```

ShippingAddress.java

```
package com.jtcicindia.hibernate;
```

```
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See   :www.jtcindia.org
**/
public class ShippingAddress{
        private int addid;
        private String street;
        private String city;
        private Order order;
        private String state;

        public ShippingAddress() {}

        public ShippingAddress (String street, String city, String state){
                this. street = street;
                this. city = city;
                this.state=state;
        }
//setters and getters
}
```

Author.hbm.xml

```
<hibernate-mapping package=”com.jtcindia.hibrenate”>
        <class name=”Author” table=”authors”>
                <id name=”aid” column=”aid” type=”int”>
                        <generator class=”increment” />
                </id>
                <property name=”aname” />
                <property name=”email” />
                <property name=”phone” type=”long” />
                <property name=”dob” type=”date”/>
                <list name=”qualis” table="qualis" />
                        <key column=”aid” />
                        <index />
                        <element column="qualis" type=”string” />
                </list>
                <set name=”exps” table=”exps”>
                        <key column=”aid” />
```

```
                    <element column=”exp” type=”string” />
                </set>
                <set name=”books” table=”books_authors”>
                    <kay column=”aid” />
                    <many-to-many class=”Book” column=”bid” />
                </set>
        </class>
</hibernate-mapping>
```

Book.hbm.xml

```
<hibernate-mapping package=”com.jtcindia.hibernate”>
        <class name=”Book” table=”books”>
                <id name=”bid” column=”bid” type=”int”>
                    <generator class=”increment” />
                </id>
                <property name=”bname” />
                <property name=”cost” type=”double” />
                <property name=”edition” type=”int” />
                <property name=”pub” />
                <set name="authors" table="books_authors">
                    <key column="bid" />
                    <many-to-many class=”Author” column=”aid />
                </set>
        </class>
</hibernate-mapping>
```

CreditCard.hbm.xml

```
<hibernate-mapping package=”com.jtcindia.hibrenate”>

        <class name=”CreditCard” table=”ccards”>
                <id name=”ccid” column=”ccid” type=”int”>
                    <generator class=”increment” />
                </id>
                <property name=”ccno” type=”int” />
                <property name=”ctype” />
                <property name=”code” type=”int” />
                <Property name="expDate" type=”date” />
                <one-to-one name=”customer” class=”Customer” />
        </class>
</hibernate-mapping>
```

Customer.hbm.xml

```
<hibernate-mapping package=”com.jtcindia.hibernate”>
        <class name=”Customer” table=customers”>
                <id name=”cid” column=”cid” type=”int”>
                        <generator class=”increment” />
                </id>
                <property name=”cname” />
                <property name=”email” />
                <property name=”phone” type=”long” />
                <many-to-one name=”ccard” class=”CreditCard” Column=”ccid”
                        unique=”true” />
                <set name=”orders”>
                        <key column=”cid” />
                        <one-to-many class="Order" />
                </set>
                <joined-subclass name=”SilverCustomer” table=”scustomers”>
                        <property name=”cid” />
                        <property name=”city” />
                        <property name=”rpoints“ type=”int " />
                        <property name=”oemail” />
                </joined-subclass>
                <joined-subclass name=”goldCustomer” table=”gcustomers”>
                        <key column=”cid” />
                        <property name=”ophone” type=”long” />
                        <property name=”points” type=”int” />
                        <map name=”refs” table=”refs”>
                                <key column=”cid” />
                                <index column=”rname” type=”string” />
                                <element column=”remail” type=”string” />
                        </map>
                </joined-subclass>
        </class>
</hibemate-mapping>
```

Order.hbm.xml

```
<hibernate-mapping package=”com.jtcindia.hibernate”>
        <class name=”Order” table=”orders”>
                <id name=”orderId” column=”orderId ” type=”int”>
                    <generator class="increment" />
                </id>
                <property name="totalQty" type="int" />
                <property name="totalCost" type="double" />
                <property name="orderData" type="data" />
                <Property name="status" />
                <many-to-one name="customer" class="Customer" column="cid"/>
                <many-to-one name="address" class="ShippingAddress"
column="addid" />
                <set name="orderItems">
                    <key column="orderId" />
                    <one-to-many class="OrderItem />
                </set>
        </class>
</hibernate-mapping>
```

OrderItem.hbm.xml

```
<hibernate-mapping package="com.jtcindia.hibernate">
        <class name="OrderItem" table="orderItems">
            <id name="otid" column="otid " type="int">
                <generator class="increment" />
            </id>
            <property name="qty" type="int" />
            <property name="cost" type="double" />
            <many-to-one name="order" class="Order" column="orderId" />
            <many-to-one name="book" class="Book" column="bid" />
        </class>
</hibernate-mapping>
```

ShippingAddress.hbm.xml

```
<hibernate-mapping package="com.jtcindia.hibernate">
        <class name="ShippingAddress" table="addresses ">
            <id name="addid" column="addid " type="int">
                <generator class="increment" />
            </id>
            <property name="street" />
            <property name="city" />
            <property name="state" />
            <one-to-one name="order" class="Order" />
        </class>
</hibernate-mapping>
```

Example with hibernate annotations

Jtc24: Files required

1. Jtc24A.java	**2. Jtc24B.java**
3. Jtc24C.java	**4. Jtc24.java**
5. Author.java	**6. Book.java**
7. CreditCard.java	**8. Customer.java**
9. GoldCustomer.java	**10. SilverCustomer.java**
11. Order.java	**12. OrderItem. Java**
13. ShippingAddress.java	**14. A HibernateUtil .java**
15. Hibernate.cfg. xml	

Jtc24A.java

```
package com.jtcindia.hibernate;
import java.util.*;
/*
*@Author:Som prakash Rai
*@company:java Training Center
*@see : www.jtcindia.org
**/
public class Jtc24A{
        public static vaid main(string[] args){
                Transation tx=null;
                try{
                        SessionFactory sf=Chibernate Util.getSessionFactory();
                        // Same as Jtc23A
                        tx.commit();
                        session.close();
                        System.out.println("record inserted");
                }catch(Exception e) {
                        e.printStackTrace();
                        if (tx!=null)
                                tx.rollback();
                }
        }
}
```

Jtc24B.java

```
package com.jtcindia.hibernate;
import java.util.*;
/*
*@Author:Som prakash Rai
*@company:java Training Center
*@see : www.jtcindia.org
**/
public class Jtc24B{
        public static vaid main(string[] args){
                Transation tx=null;
                try{
                        SessionFactory sf=ChibernateUtil.getSessionFactory();
                        // Same as Jtc23B
                        tx.commit();
                        session.close();
                        System.out.println("record inserted");
                }catch(Exception e) {
                        e.printStackTrace();
                        if (tx!=null)
                                tx.rollback();
                }
        }
}
```

Jtc24C.java

```
package com.jtcicindia.hibernate;
import java.util.*;
import org.hibernate.*;
/*
*@Author:Som Prakash Rai
*@Company:Java Training Center
*@See    :www.jtcindia.org
**/
public class Jtc23C{
        public static void main(string[]args){
```

```
		Transaction tx=null;
		try{
			SessionFactory sf=CHibernateUtil.getSessionFactory();
			Session session=sf.openSession();
			tx=session.beginTransaction();
			//same as Jtc23C
			tx.commit();
			session.close();
			System .out.println(record inserted”);
		}catch(Exception e) {
			e.printStack Trace();
			if(tx!=null)
				tx.rollback();
			}
	}
}
```

Jtc23D.java

```
package com.jtcicindia.hibernate;

import org.hibernate.*;

public class Jtc23D {
	public static void main(string[]args){
		Transaction tx=null;
		try{
			SessionFactory sf=CHibernateUtil.getSessionFactory();
			Session session=sf.openSession();
			tx=session.beginTransaction();
			//same as Jtc23D
			tx.commit();
			session.close();
			System.out.println("Record Inserted");
		}catch(Exception e){
			e.printStackTrace();
			if(tx!=null)
				tx.rollback();
		}
	}
}
```

Author.java

```
package com.jtcindia.hibernate;
import java.util.*;
import javax.persistence.*;
/*
*@Author:Som prakash Rai
*@company:java Training center
*@see :www.jtcindia.org
**/
@Entity
@Table(name="authors")
public class Author{

        @Id
        @GeneratedValue(strategy=GenerationType.AUTO)
        @Column(name="aid")
        private int aid;

        @Column(name="aname")
        private String aname;

        @Column(name="email")
        private String email;

        @Column(name="phone")
        private String phone;
        @Column(name=" dob")
        private String dob;
        @CollectionOfElements
        @JoinTable(name="qualls",JoinColumns=@JoinColumn(name="aid"))
        @Column(name="quails")
        private List<String>quails;

        @CollectionOfElements
        @JoinTable(name="exps",JoinColumns=@JoinColumn(name="aid"))
        @Column(name="exp")
        private List<String>exps;
        @ManyToMany
        @JoinTable(name=”books_authors”,JoinColumns=@JoinColumn(name=”aid”,ReferencedColumnName=“aid”),InverseJoinColumns=@JoinColumn(name=”bid”,ReferencedColumnName=”bid”))
        private set<Book>books;
//Constructors same as Jtc23
//setters and getters
}
```

Book.java

```
package com.jtcindia.hibernate;
import javax.persistence.*;
/*
*@Author:Som prakash Rai
*@company:java Training center
*@see       :www.jtcindia.org
**/
        @Entity
        @Table(name=”books”)
        public class Book{

                @Id
                @Column(name=”bid”)
                @GeneratedValue(strategy=GenerationType.AUTO)
                private int bid;

                @Column(name=”bname”)
                private String bname;

                @Column(name=”cost”)
                private String cost;

                @Column(name=”edition”)
                private String edition;

                @Column(name=”pub”)
                private String pub;
                @ManyToMany(mappedBy=”books”)
                private Set<Author>authors;

                public Book() {}
                public Book(String bname, String cost, long edition,){
                        this. bname = bname;
                        this. cost = cost;
                        this. edition = edition;
                        this pub = pub;
                }
        //Setters and Getters
}
```

CreditCard.java

```
package com.jtcindia.hibernate;
import java.util.data;
import javax.persistence.*;
/*
*@Author:Som prakash Rai
*@company:java Training center
*@see       :www.jtcindia.org
**/
@Entity
@Table(name=”ccards”)
public class CreditCard{

        @Id
        @Column(name=”ccid”)
        @GeneratedValue(strategy=GenerationType.AUTO)
        private int ccid;

        @Column(name=”ccno”)
        private int ccno;

        @Column(name=”ctype”)
        private String ctype;

        @Column(name=”code”)
        private int code;
        @Column(name="expDate")
        private Date expDate;

        @OneToOne
        @JoinColumn(name="ccid")
        private Customer customer;


        public CreditCard() {}
        public CreditCard(int ccno, String ctype, int code,Date expDate){
                this. ccno = ccno;
                this ctype = ctype;
                this. code = code;
                this. expDate = expDate;

        }
       //Setters and Getters
}
```

Customer.java

```
package com.jtcindia.hibernate;
Import java.util.set;
Import javax.persistence.*;
/*
*@Author:Som prakash Rai
*@company:java Training center
*@see :www.jtcindia.org
**/

@Entity
@Table(name="customers")
@Inheritance(strategy=InheritanceType.JOINED)
public abstract class Customer{

        @Id
        @Column(name="cid")
        @GeneratedValue(strategy=GenerationType.AUTO)
        private int cid;
        @Column(name=”cname”)
        private String cname;

        @Column(name=”email”)
        private String email;

        @Column(name=”phone”)
        private long phone;

        @Column(name=”ccid”)
        private CraditCard ccard;

        @OneToMany(mappedBy=”customer”)
        private Set<Order>orders;

        public Customer() {}
        public Customer(String cname, String email, long phone){
                this. cname = cname;
                this. email = email;
                this. phone = phone;
        }
       //Setters and Getters
}
```

GoldCustomer.java

```
package com.jtcindia.hibernate;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.PrimaryKeyJoinColumn;

import org.hibernate.annotations.CollectionOfElements;

/*
*@Author:Som prakash Rai
*@company:java Training center
*@see :www.jtcindia.org
**/
```

```
@Entity
@Table(name="gcustomers”)
@PrimaryKeyJoinColumn(name="cid")
public class GoldCustomer extends Customer{

        @Column(name=”ophone”)
        private long ophone;

        @CollectionOfElements
        @JoinTable(name="refs",joinColumns=@JoinColumn(name="cid"))
        @Column(name=”remail”)
        private Map<String,String> refs; //key should be mapKey

        @Column(name=”points”)
        private int points;

        public GoldCustomer() {}
        public GoldCustomer(String cname, String email, long phone, long
ophone,Map<String,String> refs,int points){
                super(cname,email,phone);
                this. ophone = ophone;
                this.refs=refs;
                this.points=points;
        }
       //Setters and Getters
}
```

SilverCustomer.java

```
package com.jtcindia.hibernate;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.PrimaryKeyJoinColumn;

import org.hibernate.annotations.~~CollectionOfElements~~;

/*
*@Author:Som prakash Rai
```

```
*@company:java Training center
*@see :www.jtcindia.org
**/
@Entity
@Table(name="scustomers”)
@PrimaryKeyJoinColumn(name="cid")
public class SilverCustomer extends Customer{

        @Column(name=”city”)
        private String city;

        @Column(name=”rpoints”)
        private int rpoints;

        @Column(name=”oemail”)
        private String oemail;

        public SilverCustomer() {}
        public SilverCustomer(String cname, String email, long phone, String city,int
rpoints,String oemail){
                super(cname,email,phone);
                this. city=city;
                this.rpoints=rpoints;
                this.oemail=oemail;
        }
       //Setters and Getters
}
```

Order.java

```
package com.jtcindia.hibernate;
import javax.persistence.*;
import org.hibernate.annotations.CollectionOfElements;

/*
*@Author:Som prakash Rai
*@company:java Training center
*@see :www.jtcindia.org
**/
```

```
@Entity
@Table(name="scustomers”)
public class Order {
        @Id
        @GeneratedValue(strategy=GenerationType.AUTO)
        @Column(name="orderId")
        private int orderId;

        @Column(name="totalQty")
        private int totalQty;

        @Column(name="totalCost")
        private double totalCost;

        @Column(name="orderDate")
        private Date orderDate;

        @Column(name="status")
        private String status;

        @ManyToOne
        @JoinColumn(name="cid",referencedColumnName="cid")
        private Customer customer;

        @OneToOne
        @JoinColumn(name=”addid”)
        private ShippingAddress address;

        @OneToMany(mappedBy="order")
        private Set<Orderitem> orderItems;

        public Order() {
        }

        public Order(int totalQty, double totalCost, Date orderDate, String status) {
                this.totalQty = totalQty;
                this.totalCost = totalCost;
                this.orderDate = orderDate;
                this.status = status;
        }
        //Setters and Getters
}
```

OrderItem.java

```
package com.jtcindia.hibernate;
import javax.persistence.*;
import org.hibernate.annotations.CollectionOfElements;

/*
*@Author:Som prakash Rai
*@company:java Training center
*@see :www.jtcindia.org
**/

@Entity
@Table(name="orderItems”)
public class OrderItem {
        @Id
        @GeneratedValue(strategy=GenerationType.AUTO)
        @Column(name="otid")
        private int otid;

        @Column(name="qty")
        private int qty;

        @Column(name="cost")
        private double cost;

        @ManyToOne
        @JoinColumn(name="orderId",referencedColumnName="orderId")
        private Order order;

        @OneToOne
        @JoinColumn(name="bid")
        private Book book;
        @OneToMany(mappedBy="order")
        private Set<Orderitem> orderItems;

        public OrderItem() {
        }

        public OrderItem(int qty, double cost) {
                this.qty = qty;
                this.cost = cost;
        }


       //Setters and Getters
}
```

ShippingAddress.java

```
package com.jtcindia.hibernate;
import javax.persistence.*;
import org.hibernate.annotations.CollectionOfElements;

/*
*@Author:Som prakash Rai
*@company:java Training center
*@see :www.jtcindia.org
**/

@Entity
@Table(name="address”)
public class ShippingAddress {
        @Id
        @GeneratedValue(strategy=GenerationType.AUTO)
        @Column(name="addid")
        private int addid;

        @Column(name="street")
        priavte String street;

        @Column(name="city")
```

```
		priavte String city;

		@Column(name="state")
		priavte String state;

		@oneToOne
		@JoinColumn(name="addid")
		private Order order;

		public ShippingAddress() {
		}

		public ShippingAddress(String street, String city,String state) {
			this.street = street;
			this.city = city;
			this.state=state;
		}
	//Setters and Getters
}
```

Hibernate core Example using DAO

Jtc25: Files required

1. Jtc25. Java	**2. DAOFactory.java**
3. Customer DAO. Java	**4. HibernateCustomer DAO.java**
5. CHibernate Template. Java	**6. CustomerTo. Java**
7. CHibernateUtil.java	**8. Customer. Java**
9. Customer. hbm. Xml	**10. Hibernate. cfg. Xml**

Jtc25.java

```
package com.jtcindia.hibernate;
public class Jtc25 {
	public static void main(String[] args) {
		CustomerDAO cdao=DAOFactory.getCustomerDAO();
		//1. add Customer
		CustomerTo cto=new CustomerTo("som", "som@jtcindia.com", 
9990399, "Noida", "Enable");
		cdao.addCustomer(cto);
		
		//2.get Customer
		CustomerTo c1=cdao.getCustomerByCid(1);
		
	System.out.println(c1.getCid()+"\t"+c1.getCname()+"\t"+c1.getEmail()+"\t"
+c1.getPhone()+"\t"+c1.getCity()+"\t"+c1.getStatus());
		
		//3. delete Customer
		cdao.deleteCustomer(4);
		
		//4. update Customer
		CustomerTo c2=cdao.getCustomerByCid(2);
		c2.setCname("SomPrakash");
		c2.setEmail("somprakash@jtcindia.com");
		c2.setPhone(6660366);
		cdao.updateCustomer(c2);
	}
}
```

DAOFactory.java

```
package com.jtcindia.hibernate;
public class DAOFactory {
	static CustomerDAO customerDAO;
	static{
		customerDAO=new HibernateCustomerDAO();
	}
	
	public static CustomerDAO getCustomerDAO(){
		return customerDAO;
	}
	
}
```

CustomerDAO.java

```
package com.jtcindia.hibernate;

public interface CustomerDAO {
	public int addCustomer(CustomerTo cust);
	public void updateCustomer(CustomerTo cust);
	public void deleteCustomer(int cid);
	public CustomerTo getCustomerByCid(int cid);

}
```

HibernateCustomerDAO.java

```
package com.jtcindia.hibernate;
public class HibernateCustomerDAO implements CustomerDAO{

	@Override
	public int addCustomer(CustomerTo cto) {
		Customer cust=new Customer(cto.getCid(), cto.getCname(), 
cto.getEmail(), cto.getPhone(), cto.getCity(), cto.getStatus());
		Integer it=(Integer) CHibernateTemplate.saveObject(cust);
		return it.intValue();
	}

	@Override
	public void updateCustomer(CustomerTo cto) {
		Customer cust=new Customer(cto.getCid(), cto.getCname(), 
cto.getEmail(), cto.getPhone(), cto.getCity(), cto.getStatus());
		CHibernateTemplate.updateObject(cust);

	}

	@Override
	public void deleteCustomer(int cid) {
		CHibernateTemplate.deleteObject(Customer.class, cid);;
	}

	@Override
	public CustomerTo getCustomerByCid(int cid) {
		Customer 
cust=(Customer)CHibernateTemplate.loadObject(Customer.class, cid);
		CustomerTo cto=new CustomerTo(cust.getCid(), cust.getCname(), 
cust.getEmail(), cust.getPhone(), cust.getCity(), cust.getStatus());
		return cto;
	}

}
```

CHibernateTemplate.java

```
package com.jtcindia.hibernate;

import java.io.Serializable;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

public class CHibernateTemplate {
	public static Object saveObject(Object obj){
		Object id=null;
		try{
			SessionFactory sf=CHibernateUtil.getSessionFactory();
			Session session=sf.openSession();
			Transaction tx=session.beginTransaction();
			id=session.save(obj);
			tx.commit();
			session.close();
		}catch(Exception e){
			e.printStackTrace();
		}
		return id;
	}

	public static void updateObject(Object obj){
		Object id=null;
		try{
			SessionFactory sf=CHibernateUtil.getSessionFactory();
			Session session=sf.openSession();
			Transaction tx=session.beginTransaction();
			session.update(obj);
			tx.commit();
			session.close();
		}catch(Exception e){
			e.printStackTrace();
		}
	}

	public static Object loadObject(Class cls,Serializable s){
		Object o=null;
		try{
			SessionFactory sf=CHibernateUtil.getSessionFactory();
			Session session=sf.openSession();
			Transaction tx=session.beginTransaction();
			o=session.load(cls,s);
			tx.commit();
			session.close();
		}catch(Exception e){
			e.printStackTrace();
		}
		return o;
	}

	public static void deleteObject(Class cls,Serializable s){
		Object id=null;
		try{
			SessionFactory sf=CHibernateUtil.getSessionFactory();
			Session session=sf.openSession();
```

```
                    Transaction tx=session.beginTransaction();
                    Object o=session.load(cls, s);
                    tx.commit();
                    session.close();
            }catch(Exception e){
                    e.printStackTrace();
            }
    }

}
```

CustomerTo.java

```
package com.jtcindia.hibernate;

public class CustomerTo {
        private int cid;
        private String cname;
        private String email;
        private long phone;
        private String city;
        private String status;

        public CustomerTo() {
        }

        public CustomerTo(String cname, String email, long phone, String city,
String status) {
                this.cname = cname;
                this.email = email;
                this.phone = phone;
                this.city = city;
                this.status = status;
        }
        public CustomerTo(int cid, String cname, String email, long phone, String
city, String status) {
                this.cid = cid;
```

```
            this.cname = cname;
            this.email = email;
            this.phone = phone;
            this.city = city;
            this.status = status;
    }
    //Setters and Getter
    public int getCid() {
            return cid;
    }
    public void setCid(int cid) {
            this.cid = cid;
    }
    public String getCname() {
            return cname;
    }
    public void setCname(String cname) {
            this.cname = cname;
    }
    public String getEmail() {
            return email;
    }
    public void setEmail(String email) {
            this.email = email;
    }
    public long getPhone() {
            return phone;
    }
    public void setPhone(long phone) {
            this.phone = phone;
    }
    public String getCity() {
            return city;
    }
    public void setCity(String city) {
            this.city = city;
```

```
	}
	public String getStatus() {
		return status;
	}
	public void setStatus(String status) {
		this.status = status;
	}
}
```

CHibernateUtil.java

```
package com.jtcindia.hibermate;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class CHibernateUtil {
	static SessionFactory factory = null;
	static {
		try {
			Configuration configuration = new Configuration();
			configuration = configuration.configure("hibernate.cfg.xml");
			factory = configuration.buildSessionFactory();
		} catch (Exception e) {
			e.printStackTrace();
		}
	}
	public static SessionFactory getSessionFactory() {
		return factory;
	}
}
```

Customer.java

```
package com.jtcindia.hibernate;
public class Customer {
	private int cid;
	private String cname;
	private String email;
	private long phone;
	private String city;
	private String status;

	public Customer() {
	}
	public Customer(String cname, String email, long phone, String city, String
status) {
		this.cname = cname;
		this.email = email;
		this.phone = phone;
		this.city = city;
		this.status = status;
	}
	public Customer(int cid, String cname, String email, long phone, String
city, String status) {
		this.cid = cid;
		this.cname = cname;
		this.email = email;
		this.phone = phone;
		this.city = city;
		this.status = status;
	}
	//Setters and Getter
	public int getCid() {
		return cid;
	}
	public void setCid(int cid) {
		this.cid = cid;
	}
	public String getCname() {
		return cname;
	}
	public void setCname(String cname) {
		this.cname = cname;
	}
	public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public long getPhone() {
        return phone;
    }
    public void setPhone(long phone) {
        this.phone = phone;
    }
    public String getCity() {
        return city;
    }
    public void setCity(String city) {
        this.city = city;
    }
    public String getStatus() {
        return status;
    }
    public void setStatus(String status) {
        this.status = status;
    }
}
```

Customer.hbm.xml

```
<hibernate-mapping package="com.jtcindia.hibernate">
    <class name="Customer" table="customers" lazy="false">
        <id name="cid" column="cid" type="int">
            <generator class="increment" />
        </id>
        <property name="cname" />
        <property name="email" type="string" />
        <property name="phone" column="phone" type="long" />
        <property name="city" column="city" />
        <property name="status" column="status" type="string" />
        </class>
</hibernate-mapping>
```

Hibernate Annotations Example using DAO

Jtc: Files required

1. Jtc26. Java (Same As JTC25)	2. DAOFactory.java Same as JTC25
3. Customer DAO. Java(Same as Jtc25)	4. HibernateCustomerDAO.java (same as JTC25)
5. AHibernateTemplate.java (Same as Jtc25)	6. CustomerTo.Java Same Jtc25
7. AHibernateUtil.java (same as Jtc25)	8. Customer.java
9. Hibernate.cfg. xml	

Customer.java

```
package com.jtcindia.hibernate;

import javax.persistence.*;

@Entity
@Table(name = "customer")
public class Customer {
        @Column(name = "cid")
        @Id
        private int cid;
        @Column(name = "phone")
        private long phone;

        @Column(name = "cname")
        private String cname;
        @Column(name = "email")
        private String email;
        @Column(name = "city")
        private String city;

        @Column(name = "status")
        private String status;

        public Customer() {
        }

        public Customer(String cname, String email, long phone, String city,String
status) {
                this.cname = cname;
                this.email = email;
                this.phone = phone;
                this.city = city;
                this.status = status;

        }
        public Customer(int cid,String cname, String email, long phone, String
city,String status) {
                super();
                this.cid=cid;
                this.cname = cname;
                this.email = email;
                this.phone = phone;
                this.city = city;
                this.status = status;
        }
        //setters and Getters
}
```

Hibernate Query Languages

There are 4 methods defined in session for doing CURD operations.

1. **session. save ();**
2. **session. load();**
3. **session. update();**
4. **session. delete()**

- Hibernate supports various Query languages to select the records based on various criteria's.

1. HQL (Hibernate Query language).
2. QBC(Query by criteria)
3. Native queries
4. Named Queries.

- Consider the table called students which is mapped with persistence class called Student.

table Students ->**Sid, sname, email, branch1** (Cloumn name in the table Student)

class Student ->**Sid, sname, email, branch** (Variable name in persistent class)

Q1. Display all the students

SQL: In Sql we directly use column name and table name query.

select sid, sname, branch1 from student's stu;

HOL: It is also called Object Oriented Query Language (OQL)

String hql ="from student stu;
Query q = session. Create Query (hql);
List<Student>list = q.list ();

QBC: **Criteria ct = session. Create criteria (student. class);**
List<student>list= ct.list ()

Native Queries:

String sql =" select (stu.*} from students stu"

```
SQLQuery sq = session. Create SQLQuery (sql);
Sq.addsEntity ("stu",Student.class);
List<Student> List = sq. list ();
```

Named Queries:

In student. hbm. xml

```
<hibernate-mapping>
        <class>
                ….
        </class>
        <spl-query name = "AllStudents">
        <return alias = "stu" class ="Student"/>
        SELECT stu1. sid AS stu. sid,
        stu1. sname AS stu. Sname,
        stu1. email AS stu. email,
        stu1. branch1  AS stu. branch,
        FROM students stu1;
        </sql-query>
</hibernate-mapping>
```

In client code

```
Query q=session. getNameQuery ("All Students");
List<Student> list =q. list ();
```

Q2. Display all the students belongs to Noida

SQL: select * from students stu where stu.branch1 =?;

HOL:

```
String hql ="from Student stu where stu. branch =?";
Query q = session. create Query (hql);
q=q.setString (0,"Noida")
List<Student>list = q.list ();
```

QBC:

```
Criteria ct = session. createCriteria (Student. class);
```

```
Ct=ct. add (Expression. req (“branch”,”Noida”));
List<student>list=ct.list ()
```

Native Queries:

```
String sql =” select (stu.*} from students stu where stu.branch1=?”;
SQL Query sq = session. createSQL Query (sql);
q. set string (0,”Noida”)
sq. add Entity(“stu”, Student. class);
List<Student> List = sq. list();
```

Named Queries:

In student.hbm. xml

```
<hibernate-mapping>
        <class>
                ….
        </class>
        <sql-query name = “StudentsByBranch”/>
                <return alias = “stu” class =”Student”/>
                SELECT stu1. sid AS stu. sid,
                stu1. sname AS stu. Sname,
                stu1. email AS stu. email,
                stu1. branch1  AS stu. branch,
                FROM students stu1;
                WHERE stu1. branch1 =?
        </sql-query>
</hibernate-mapping>
```

In client code

```
Query q=session. Get name Query (“StudentByBranch”);
q. setString (0,”Noida”)
List<Student> list =q. list ();
```

More about HQL and QBC

1. **Display all the customers**.

HQL

```
String hql = "from Customer c";
Query q = session.createQuery (hql);
List<Customer> list = q.list ();
for (Customer c: list) {
        System. out.println(c);
}
```

QBC

```
Criteria ct = session. createCriteria (Customer. class);
List < Customer> list = ct. list ();
for (customer c: list) {
        System.out.println (c);
}
```

2. **Display all the Customers staying in Noida.**

HQL

```
String hql = "from customer c where c.city=?";
Query q = session.createQuery (hql);
q=q. setString (0,"Noida")'
List<Customer> list = q.list ();
for (Customer c: list) {
        System. out.println(c);
}
```

QBC

```
Criteria ct = session. createCriteria (Customer. class);
ct=ct. add (Expression. eq ("city"," delhi"));
List = ct. list ();
for (Customer c: list) {
        System. out. Println(c);
}
```

3. **Display all the customers staying in Noida with BAL> 10000.**

HQL

```
String hql = “from Customer c where c. city =? and c. cardBAL>?”
Query q = session. createQuery (hql);
q=q. setparameter (0,”Noida”)
q=q.setparameter (1, 10000.0);
list =q. list ();
```

QBC

```
Criteria ct = session. createCriteria (Customer. class);
ct=ct. add (Expression. and (Expression. eq (“city”,”delhi”),Expression. gt
(“cardBal”, 1,10000.0));
List =q. list ();
```

4. **Display all the customers with Visa card, bal between 10K and 30K and status true.**

HQL

```
String hql =  “from Customer c where c. cardType = ? and c. cardBal
between ? and ? c. status =?”
Query q=session. createQuery (hql);
q=q. setparameter (0,”Visa”);
q=q. setparameter (1, 10000.0);
q=q setparameter (2, 30000.0);
q=q. se parameter (3, true);
list = q. list ();
```

QBC

```
Criteria ct = session. createCriteria (Customer. class);
ct=ct. add (Expression. and (Expression. and (Expression.eq (“card Type “,
“Visa”) Expression eq (“status”, true)),Expression.between(“cardType”,
”10000.0, 300000.0)));
list=ct. list
```

5. **Display all the customers who are in Noida, pune, and Delhi.**

HQL

```
String hql = “from Customer c where c. city in (???)”;
Query q=session. createQuery (hql);
q=q.setparameter (0,”Noida”);
```

```
q=q. setparameter (1,"pune");
q=q. setparameter (3," Delhi");
list = q. list ();
```

QBC

```
Criteria ct = session.createCriteria (Customer. class);
ct=ct. add (Expression. eq ("city",list));
list =q.list ();
```

6. Display all the Customers who are in Noida, in desc order by cname

HQL

```
String hql = "from Customer c where c.city=? order by c.cname desc";
Query q=session.createQuery (hql);
q=q. setparameter (0,"Noida");
list = q.list ();
```

QBC

```
Criteria ct = session.createCriteria (Customer. class);
ct=ct. add (Expression. eq ("city"," pune"));
ct=ct. addOrder (Order.desc ("cname"));
list =ct.list ();
```

7. Display all the Customers whose email is Jtc email accorder by email

HQL

```
String hql=" from Customer c where c.email like ? order by c.email";
Query q=session.createQuery (hql);
q=q. setparameter (0," %Jtc.com");
list=ct.list ();
```

QBC

```
Criteria ct=session.createCriteria (Customer. class);
ct=ct. add (Expression.like ("email", "%Jtc.com"));
ct=ct.addOrder (Order.asc ("email"));
list=ct.list ();
```

8. Display all the Customers whose email is jtc email accorder by email

HQL

```
String hql=" from Customer c where c.email like ? order by c.email";
Query q=session.createQuery (hql);
q=q. setparameter (0," %Jtc.com");
list=ct.list ();
```

QBC

```
Criteria ct=session.createCriteria (Customer. class);
ct=ct. add (Expression.like ("email", "%Jtc.com"));
ct=ct.addOrder (Order.asc ("email"));
list=ct.list ();
```

9. Display Nth highest customer.

```
Object getNthHighestCustomer (int n) {
        String hql="select distinct * from Customer c where order by 
c.cardBal";
        Query q = session.createQuery (hql);
        list = q.list ();
        Object o = list.get (n-1);
        return 0;
}
```

Pagination

10.Display all the customers staying in Noida.

HQL

```
getCustomersBy City (String ci, int start, int total) {
        String hql="from Customer c where c. city =?";
        Query q=session.createQuery (hql);
        q=q.setString (0, ci);
        q=q.setFirstResult (start);
        q=q.setMaxResults (total);
        list<Customer>list =q.list ();
}
```

QBC

```
getCustomersBy City (String ci, int start, int total) {
        Criteria ct= session.createCriteria (Customer. class);
        ct=ct add(Expression. eq ("city", ci));
        ct=ct.setFirstResult (start);
        ct=ct.setMaxResults (total);
        list =ct.list ();
}
```

Polymorphic Queries.

1. **Display the Regular students**

```
String hql = "from RegularStudent stu";
Query q = session. create Query (hql);
list<Regular Student> list = q.list ();
```

2. **Display all types of students**

```
String hql =" from student stu";
Query q = session.createQuery (hql);
List<Student> list = q.list ();
for (Student s: list) {
        if(s instance of RegularStudent){
                RegularStudent rest = (Regular student) s;
        } else if (s instance of WeekendStudent)
                WeekendStudent ws= (WeekendStudent) s;
        }
}
```

3. **Display all records from all the tables**

```
String hql = "from Object obj";
```

4. **Display all Serializable**

```
String hql = "from Serializable s";
```

Parameter types

1. **Positional parameters**
2. **Named parameters**

Positional parameters

Ex 1) String hql = “from Customer c where c.city =?”;
Query q = session. createQuery (hql);
q=q. setString (0,”Noida”);

Ex 2) String hql = “from Customer c where c.city =? And c.cardBal>?”;
Query q = session.createQuery (hql);
q=q. setparameter (0,”Noida”);
q=q.setparameter(1,10000.0);
list = q.list ();

Named parameters

Ex 1) String hql = “from Customer c where c.city= :cit”;
Query q = session.createQuery (hql);
q=q. setString (“cit,” Noida”);

Ex2) String hql = “from Customer c where c.city=:cit and c.cardBAL>: cbal”;

Query q = session. createQuery (hql);
q=q. setparameter (“cit”,”Noida”);
q=q. set parameter (“cbal”, 10000.0);
list = q. list ();

Example Using HQL

Jtc27: Files required

1. Jtc27. Java	**2. DAOFacttory.java**
3. CustomerDAO.java	**4. HibernateCustomerDAO.java**
5. HibernateTemplate.java	**6. CHibernateUtil.java**
7. Customer.java	**8. Customer. hbm. Xml**
9. Hibernate. cfg. xml	

Jtc27.java

```
package com.jtcindia.hibernate;
import java.util.List;
public class Jtc27 {
	public static void main(String[] args) {
		CustomerDAO cdao = DAOFactory.getCustomerDAO();
		System.out.println("All Customers");
		List<Customer> list = cdao.getAllCustomers();
		for (Customer c : list) {
			System.out.println(c);
		}
		System.out.println("Customer By City");
		list = cdao.getCustomersByCity("Noida");
		for (Customer c : list) {
			System.out.println(c);
		}
		System.out.println("Get Customer by city 0 3");
		list = cdao.getCustomersByCity("noida", 0, 3);
		for (Customer c : list) {
			System.out.println(c);
		}
		System.out.println("Get Customer by cardType");
		list = cdao.getCustomersByCardType("MASTER");
		for (Customer c : list) {
			System.out.println(c);
		}
		System.out.println("Get Customer by cardType 0 2");
		list = cdao.getCustomersByCardType("MASTER", 0, 2);
		for (Customer c : list) {
			System.out.println(c);
		}
		System.out.println("Get Customer by  bal");
		list = cdao.getCustomersBal(15200.0);
		for (Customer c : list) {
			System.out.println(c);
		}
```

```
                    System.out.println("Get Customer by  bal  0 1");
                    list = cdao.getCustomersBal(15200.0, 0, 1);
                    for (Customer c : list) {
                              System.out.println(c);
                    }

                    System.out.println("Get Customer by status ");
                    list = cdao.getCustomersByStatus("Active");
                    for (Customer c : list) {
                              System.out.println(c);
                    }

                    System.out.println("Get Customer by status 0 3");
                    list = cdao.getCustomersByStatus("Active", 0, 3);
                    for (Customer c : list) {
                              System.out.println(c);
                    }

                    System.out.println("Get Customer by name ");
                    list = cdao.getCustomersByName("som1");
                    for (Customer c : list) {
                              System.out.println(c);
                    }

                    System.out.println("Get Customer by CardType ");
          /*        list = cdao.getCustomers("noida", "VISA");
                    for (Customer c : list) {
                              System.out.println(c);
                    }
*/
                    System.out.println("Get Customer by CardType status  ");
                    list = cdao.getCustomers("noida", "VISA", "Active");
                    for (Customer c : list) {
                              System.out.println(c);
                    }
```

```
                System.out.println("Get Customer by email  ");
                Customer c = (Customer)
cdao.getCustomerByEmail("Som@jtc.org");

                System.out.println(c);
                System.out.println("Get All Customer By CCNO");
                c = (Customer) cdao.getCustomerCardNo(12345);
                System.out.println(c);
        }
}
```

DAOFactory.java

```
package com.jtcindia.hibernate;
public class DAOFactory {
        static CustomerDAO customerDAO = null;
        static {
                customerDAO = new HibernateCustomerDAO();
        }
        public static CustomerDAO getCustomerDAO(){
                return customerDAO;
        }
}
```

CustomerDAO.java

```
package com.jtcindia.hibernate;
import java.util.*;
public interface CustomerDAO {
        public int addCustomer(Customer cust);
        public void updateCustomer(Customer cust);
        public void deleteCustomer(int cid);
        public Customer getCustomerByCid(int cid);
        public Customer getCustomerByEmail(String email);
        public Customer getCustomerCardNo(int ccno);
        public List<Customer> getAllCustomers();
```

```
	public List<Customer> getAllCustomers(int start, int total);
	public List<Customer> getCustomersByCity(String city);
	public List<Customer> getCustomersByCity(String city, int start,
				int total);
	public List<Customer> getCustomersByName(String name);
	public List<Customer> getCustomersByName(String name, int start, int
total);
	public List<Customer> getCustomersByCardType(String cardType);
	public List<Customer> getCustomersByCardType(String cardType, int
start,
				int total);
	public List<Customer> getCustomersByStatus(String status);
	public List<Customer> getCustomersByStatus(String status, int start,
				int total);
	public List<Customer> getCustomersBal(double bal);
	public List<Customer> getCustomersBal(double bal, int start, int total);
/*
	public List<Customer> getCustomers(String city, String cardType);*/
	public List<Customer> getCustomers(String city, String cardType,String
status);
/*	public List<Customer> getCustomers(String city, String cardType, int start,
				int total);
	public List<Customer> getCustomers(String city, String cardType,
				String status, int start, int total);
*/
}
```

Customer.java

```
package com.jtcindia.hibernate;

public class Customer {
	private int cid;
	private String cname;
	private String email;
	private String city;
	private double cardBal;
	private String cardType;
	private int cardNo;
	private String status;

	public Customer() {	}

	public Customer(String cname, String email, String city, double cardBal,
			String cardType, int cardNo, String status) {
		this.cname = cname;
		this.email = email;
		this.city = city;
		this.cardNo = cardNo;
		this.cardBal = cardBal;
		this.status = status;
	}
	//Seters and Getter

	public String toString() {
		return this.cid + "\t" + this.cname + "\t " + this.email + "\t "
				+ this.city + "\t" + this.cardBal + " \t" + this.cardNo +
"\t"
				+ this.cardType + "\t" + this.status;
	}
}
```

HibernateCustomerDAO.java

```
package com.jtcindia.hibernate;

import java.util.*;

public class HibernateCustomerDAO implements CustomerDAO {
	public int addCustomer(Customer cust) {
		Integer it = (Integer) HibernateTemplate.save(cust);
		return it.intValue();
	}

	public void updateCustomer(Customer cust) {
		HibernateTemplate.update(cust);
	}

	public void deleteCustomer(int cid) {
		HibernateTemplate.delete(Customer.class, cid);

	}

	public Customer getCustomerByCid(int cid) {
		Customer c = (Customer) HibernateTemplate.load(Customer.class,
cid);
		return c;
	}

	public Customer getCustomerByEmail(String email) {
		String hql = "from customer c where c.email=?";
		Customer c = (Customer) HibernateTemplate.findObject(hql, email);
		return c;
	}

	public Customer getCustomerCardNo(int ccno) {
		String hql = "from customer c where c.cardNo=?";
		Customer c = (Customer) HibernateTemplate.findObject(hql, ccno);
		return c;
	}

	public List<Customer> getAllCustomers() {
		String hql = "from customer c";
		List<Customer> list = HibernateTemplate.getList(hql);
		return list;

	}
	public List<Customer> getAllCustomers(int start, int total) {
		String hql = "from customer c";
		List<Customer> list = HibernateTemplate.getList(hql, start, total);
		return list;

	}

	public List<Customer> getCustomersByCity(String city) {
		String hql = "from customer c where c.city=?";
		List<Customer> list = HibernateTemplate.getList(hql, city);
		return list;

	}

	public List<Customer> getCustomersByCity(String city, int start, int total)
{
		String hql = "from customer c where c.city=?";
		List<Customer> list = HibernateTemplate.findList(hql, start, total,
				city);

		return list;

	}

	public List<Customer> getCustomersByName(String name) {
		String hql = "from customer c where c.cardType=?";
		List<Customer> list = HibernateTemplate.getList(hql, name);
		return list;
	}

	public List<Customer> getCustomersByName(String name, int start, int
total) {
		String hql = "from customer c where c.cardType=?";
		List<Customer> list = HibernateTemplate.findList(hql, start, total,
				name);
		return list;
	}

	public List<Customer> getCustomersByCardType(String cardType) {

		String hql = "from customer c where c.cardType=?";
		List<Customer> list = HibernateTemplate.getList(hql, cardType);
		return list;

	}

	public List<Customer> getCustomersByCardType(String cardType, int
start,
			int total) {
		String hql = "from customer c where c.cardType=?";
		List<Customer> list = HibernateTemplate.findList(hql, start, total,
				cardType);
		return list;
	}

	public List<Customer> getCustomersByStatus(String status) {
		String hql = "from customer c where c.status=?";
		List<Customer> list = HibernateTemplate.getList(hql, status);
		return list;
	}

	public List<Customer> getCustomersByStatus(String status, int start,
			int total) {
		String hql = "from customer c where c.status=?";
		List<Customer> list = HibernateTemplate.findList(hql, start, total,
				status);
		return list;
	}

	public List<Customer> getCustomersBal(double bal) {
		String hql = "from customer c where c.cardBal=?";
		List<Customer> list = HibernateTemplate.getList(hql, bal);
		return list;
	}

	public List<Customer> getCustomersBal(double bal, int start, int total) {
		String hql = "from customer c where c.cardBal=?";
		List<Customer> list = HibernateTemplate
				.findList(hql, start, total, bal);
		return list;

	}

	/*
	 * public List<Customer> getCustomers(String city, String cardType) {
	 *
	 * }
	 */

	public List<Customer> getCustomers(String city, String cardType,
			String status) {
		String hql = "from customer c where c.city=? and c.cardType=? and
c.status=?";
		List<Customer> list = HibernateTemplate.getList(hql, city, cardType,
				status);
		return list;
	}

	/*
	 * public List<Customer> getCustomers(String city, String cardType, int
	 * start, int total) {
	 *
	 * String hql =
	 * "from customer c where c.city=? and c.cardType=? and c.status=?";
	 * List<Customer> list = HibernateTemplate.findList(hql, start, total, city,
	 * cardType, status); return list;
	 *
	 * }
	 */

	/*
	 * public List<Customer> getCustomers(String city, String cardType, String
	 * status, int start, int total){
	 *
	 * }
	 */
}
```

HibernateTemplate.java

```
package com.jtcindia.hibernate;
import java.io.Serializable;
import java.util.List;
import org.hibernate.Query;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.classic.Session;

public class HibernateTemplate {
	public static Object save(Object obj) {
		Transaction tx = null;
		Object ob = null;
		try {
			SessionFactory sf = CHibernateUtil.getSessionFactory();
			Session session = sf.openSession();
			tx = session.beginTransaction();
			ob = session.save(obj);
			tx.commit();
		} catch (Exception e) {
			if (tx != null) {
				tx.rollback();

			}
			e.printStackTrace();
		}
		return ob;
	}

	public static Object load(Class cls, Serializable id) {
		Transaction tx = null;
		Object obj = null;
		try {
			SessionFactory sf = CHibernateUtil.getSessionFactory();
			Session session = sf.openSession();
			tx = session.beginTransaction();
			obj = session.load(cls, id);
			tx.commit();
			session.close();
		} catch (Exception e) {
			if (tx != null) {
				tx.rollback();
			}
			e.printStackTrace();
		}
		return obj;
	}

	public static void delete(Class cls, Serializable id) {
		Transaction tx = null;
		Object obj = null;
		try {
			SessionFactory sf = CHibernateUtil.getSessionFactory();
			Session session = sf.openSession();
			tx = session.beginTransaction();
			session.load(cls, id);
			tx.commit();
			session.close();
		} catch (Exception e) {
			if (tx != null) {
				tx.rollback();
```

```
                }
                e.printStackTrace();
            }

    }

    public static void update(Object obj) {
            Transaction tx = null;
            try {
                SessionFactory sf = CHibernateUtil.getSessionFactory();
                Session session = sf.openSession();
                tx = session.beginTransaction();
                session.update(obj);
                tx.commit();
                session.close();
            } catch (Exception e) {
                if (tx != null) {
                        tx.rollback();
                }
                e.printStackTrace();
            }
    }

    public static List getList(String hql, Object... args) {
            Transaction tx = null;
            List list = null;
            try {
                SessionFactory sf = CHibernateUtil.getSessionFactory();
                Session session = sf.openSession();
                tx = session.beginTransaction();
                org.hibernate.Query q = session.createQuery(hql);

                for (int i = 0; i < args.length; i++) {
                        q = q.setParameter(i, args[i]);
                }
                list = q.list();
                tx.commit();
                session.close();
            } catch (Exception e) {
                if (tx != null) {
                    tx.rollback();
                }
                e.printStackTrace();
            }
            return list;
        }

        public static List findList(String hql, int start, int total,
                Object... args) {
            Transaction tx = null;
            List list = null;
            try {
                SessionFactory sf = CHibernateUtil.getSessionFactory();
                Session session = sf.openSession();
                tx = session.beginTransaction();
                org.hibernate.Query q = session.createQuery(hql);
                q = q.setFirstResult(start);
                q = q.setMaxResults(total);
                for (int i = 0; i < args.length; i++) {
                    q = q.setParameter(i, args[i]);

                }
                list = q.list();
                tx.commit();
                session.close();

            } catch (Exception e) {

                if (tx != null) {
                    tx.rollback();
                }
                e.printStackTrace();
```

```
		}
		return list;
	}

	public static Object findObject(String hql, Object... args) {

		Transaction tx = null;
		Object obj = null;
		try {
			SessionFactory sf = CHibernateUtil.getSessionFactory();
			Session session = sf.openSession();
			tx = session.beginTransaction();
			Query q = session.createQuery(hql);

			for (int i = 0; i < args.length; i++) {
				q = q.setParameter(i, args[i]);
			}
			obj = q.uniqueResult();
			tx.commit();
			session.close();
		} catch (Exception e) {
			if (tx != null) {
				tx.rollback();
			}
			e.printStackTrace();
		}
		return obj;
	}
}
```

Customer.hbm.xml

```
<hibernate-mapping package="com.jtcindia.hibernate">
	<class name="Customer" table="customer">
		<id name="cid" column="cid" type="int">
			<generator class="increment" />
```

```
                </id>
                <property name="cname" column="cname" />
                <property name="email" column="email" />
                <property name="city" column="city" />
                <property name="cardBal" column="double" />
                <property name="cardType" />
                <property name="cardNo" type="int" />
                <property name="status" />
        </class>
</hibernate-mapping>
```

Example Using QBC

Jtc28: files required

1. Jtc28. Java (same as Jtc 27)	**2. DAOFacttory.java (same as Jtc 27)**
3. CustomerDAO.java (same as Jtc 27)	**4. HibernateCustomer DAO.java**
5. HibernateTemplate.java	**6. CHibernateUtil.java**
7. Customer.java (same as Jtc 27)	**8. Customer. hbm.xml (same as Jtc 27)**
9. Hibernate. cfg.xml	

HibernateTemplate.java

```
package com.jtcindia.hibernate;
import java.io.Serializable;
import java.util.List;
import org.hibernate.Query;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.classic.Session;

public class HibernateTemplate {
        public static Object save(Object obj) {
                Transaction tx = null;
                Object ob = null;
                try {
                        SessionFactory sf = CHibernateUtil.getSessionFactory();
                        Session session = sf.openSession();
```

```
            tx = session.beginTransaction();
            ob = session.save(obj);
            tx.commit();
        } catch (Exception e) {
            if (tx != null) {
                tx.rollback();

            }
            e.printStackTrace();
        }
        return ob;
    }

    public static Object load(Class cls, Serializable id) {
        Transaction tx = null;
        Object obj = null;
        try {
            SessionFactory sf = CHibernateUtil.getSessionFactory();
            Session session = sf.openSession();
            tx = session.beginTransaction();
            obj = session.load(cls, id);
            tx.commit();
            session.close();
        } catch (Exception e) {
            if (tx != null) {
                tx.rollback();
            }
            e.printStackTrace();
        }
        return obj;
    }

    public static void delete(Class cls, Serializable id) {
        Transaction tx = null;
        Object obj = null;
        try {
            SessionFactory sf = CHibernateUtil.getSessionFactory();
            Session session = sf.openSession();
            tx = session.beginTransaction();
            session.load(cls, id);
            tx.commit();
            session.close();
        } catch (Exception e) {
            if (tx != null) {
                tx.rollback();
            }
            e.printStackTrace();
        }

    }

    public static void update(Object obj) {
        Transaction tx = null;
        try {
            SessionFactory sf = CHibernateUtil.getSessionFactory();
            Session session = sf.openSession();
            tx = session.beginTransaction();
            session.update(obj);
            tx.commit();
            session.close();
        } catch (Exception e) {
            if (tx != null) {
                tx.rollback();
            }
            e.printStackTrace();
        }
    }

    public static List getList(String hql, Object... args) {
        Transaction tx = null;
        List list = null;
        try {
            SessionFactory sf = CHibernateUtil.getSessionFactory();
            Session session = sf.openSession();
            tx = session.beginTransaction();
            org.hibernate.Query q = session.createQuery(hql);

            for (int i = 0; i < args.length; i++) {
                q = q.setParameter(i, args[i]);
            }
            list = q.list();
            tx.commit();
            session.close();
        } catch (Exception e) {
            if (tx != null) {
                tx.rollback();
            }
            e.printStackTrace();
        }
        return list;
    }

    public static List findList(String hql, int start, int total,
            Object... args) {
        Transaction tx = null;
        List list = null;
        try {
            SessionFactory sf = CHibernateUtil.getSessionFactory();
            Session session = sf.openSession();
            tx = session.beginTransaction();
            org.hibernate.Query q = session.createQuery(hql);
            q = q.setFirstResult(start);
            q = q.setMaxResults(total);
            for (int i = 0; i < args.length; i++) {
                q = q.setParameter(i, args[i]);

            }
            list = q.list();
```

```
			tx.commit();
			session.close();

		} catch (Exception e) {

			if (tx != null) {
				tx.rollback();
			}
			e.printStackTrace();
		}
		return list;
	}

	public static Object findObject(String hql, Object... args) {

		Transaction tx = null;
		Object obj = null;
		try {
			SessionFactory sf = HibernateUtil.getSessionFactory();
			Session session = sf.openSession();
			tx = session.beginTransaction();
			Query q = session.createQuery(hql);

			for (int i = 0; i < args.length; i++) {
				q = q.setParameter(i, args[i]);
			}
			obj = q.uniqueResult();
			tx.commit();
			session.close();
		} catch (Exception e) {

			if (tx != null) {
				tx.rollback();
			}
			e.printStackTrace();
		}
```

```
                return obj;
        }
}
```

Jtc29: files required

1. **Jtc29. Java**	2. **Jtc29B.java**
3. **DAOFacttory.java (same as Jtc27)**	4. **CustomerDAO.java**
5. **Hibernate customer DAO.java**	6. **CHibernateUtil.java**
7. **Customer.java (same as Jtc27)**	8. **Customer.hbm. xml (same as Jtc27)**
9. **Hibernate.cfg. Xml**	

Jtc29A.java

```
package com.jtcindia.hibernate;

import java.util.List;
public class Jtc29A {
        public static void main(String[] args) {
                CoustomerDAO cdao = DAOFactory.getCustomerDAO();
                System.out.println("All Customers");
                List<Customer> list = cdao.getAllCustomers();
                for (Customer c : list) {
                        System.out.println(c);
                }
                System.out.println("\nCustomer by Status");
                list = cdao.getCustomersByStatus("Active");
                for (Customer c : list) {
                        System.out.println(c);
                }
                System.out.println("\nCustomers by email");
                Customer c = (Customer)
cdao.getCustomersByEmail("som@jtcindia.com");
                System.out.println(c);

        }
}
```

Jtc29B.java

```
package com.jtcindia.hibernate;

import java.util.List;
import org.hibernate.Hibernate;
import org.hibernate.SQLQuery;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

public class Jtc29B {

	public static void main(String[] args) {
		Transaction tx=null;
		try{
		SessionFactory sf=CHibernateUtil.getSessionFactory();
		Session session=sf.openSession();
		tx=session.beginTransaction();
		System.out.println("\n1. All the Customers");
		String SQL1="select * from Customers cts";
		SQLQuery sq=session.createSQLQuery(SQL1);
		sq=sq.addScalar("cid",Hibernate.INTEGER);
		sq=sq.addScalar("cname",Hibernate.STRING);
		sq=sq.addScalar("email",Hibernate.STRING);
		sq=sq.addScalar("city",Hibernate.STRING);
		sq=sq.addScalar("status",Hibernate.STRING);

		List<Object[]> clist=sq.list();
		for(Object obj[]:clist){
			for(Object o:obj){
				System.out.println(o+"\t");
			}
			System.out.println();
		}
```

```
                System.out.println("\n2. Emails of all the contacts");
                String SQL4="select cts.email from customers cts";
                SQLQuery sq1=session.createSQLQuery(SQL4);
                List<String> ems=sq1.addScalar("email",Hibernate.STRING).list();
                for(String str:ems){
                        System.out.println(str);
                }
                tx.commit();
                session.close();
                }catch(Exception e){
                        e.printStackTrace();
                        if(tx!=null){
                                tx.rollback();
                        }
                }
        }
}
```

CustomerDAO.java

```
package com.jtcindia.hibernate;

import java.util.List;

public interface CustomerDAO {
        public Customer getCustomersByEmail(String email);
        public List<Customer> getAllCustomers();
        public List<Customer> getCustomerByStatus(String status);
}
```

HibernateDAO.java

```
package com.jtcindia.hibernate;

import java.util.List;
import org.hibernate.SQLQuery;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

public class HibernateDAO implements CustomerDAO {

	@Override
	public Customer getCustomersByEmail(String email) {
		Transaction tx=null;
		Customer cust=null;
		try{
			SessionFactory sf=CHibernateUtil.getSessionFactory();
			Session session=sf.openSession();
			tx=session.beginTransaction();
			String SQL2="select {cts.*} from customers cts where 
cts.email=?";
			SQLQuery sq2=session.createSQLQuery(SQL2);
			sq2.setString(0, email);
			List<Customer> list=sq2.addEntity("cts",Customer.class).list();
			if(list.size()>0)
				cust=list.get(0);
			tx.commit();
			session.close();
		}catch(Exception e){
			e.printStackTrace();
		}
		return cust;
	}

	@Override
	public List<Customer> getAllCustomers() {
		Transaction tx=null;
		List<Customer> list=null;
		try{
			SessionFactory sf=CHibernateUtil.getSessionFactory();
			Session session=sf.openSession();
			tx=session.beginTransaction();
```

```
                String SQL1="select {cts.*} from customers cts";
                SQLQuery sq1=session.createSQLQuery(SQL1);
                list=sq1.addEntity("cts",Customer.class).list();
                tx.commit();
                session.close();
            }catch(Exception e){
                e.printStackTrace();
            }
            return list;
        }

        @Override
        public List<Customer> getCustomerByStatus(String status) {
            Transaction tx=null;
            List<Customer> list=null;
            try{
                SessionFactory sf=CHibernateUtil.getSessionFactory();
                Session session=sf.openSession();
                tx=session.beginTransaction();
                String SQL3="select {cts.*} from customers cts where
cts.status= :sts";
                SQLQuery sq3=session.createSQLQuery(SQL3);
                sq3.setString("sts", "Active");

                list=sq3.addEntity("cts",Customer.class).list();
                tx.commit();
                session.close();
            }catch(Exception e){
                e.printStackTrace();
            }
            return list;
        }

}
```

Jtc30: files required

1. **Jtc30. Java**	2. **Customer.java** (same as Jtc27)
3. **Customer.hbm.xml**	4. **CHibernateUtil.java**
5. **Hibernate.cfg.xml**	

Jtc30.java

```
package com.jtcindian.hibernate;
import java.util.List;
import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;

public class Jtc30 {
        public static void main(String[] args) {
                Transaction tx=null;
                try{
                        SessionFactory sf=CHibernateUtil.getSessionFactory();
                        Session session=sf.openSession();
                        tx=session.beginTransaction();
                        System.out.println("1.All the Customers");
                        List<Customer>
clist1=session.getNamedQuery("JTCAllCustomers").list();
                        for(Customer c:clist1){
                                System.out.println(c);
                        }
                        System.out.println("2.All Active the customers");
                        List<Customer>
clist2=session.getNamedQuery("JTCActiveCustomers").list();
                        for(Customer c:clist2){
                                System.out.println(c);
                        }
                        System.out.println("3.Emails of All the Customers");
                        List<String>
elist=session.getNamedQuery("EmailsOfAllCustomers").list();
```

```
			for(String str:elist){
				System.out.println(str);
			}
			System.out.println("3.Emails and Status of All the 
Customers");
			List<Object[]> 
eplist=session.getNamedQuery("EmailsAndstatusOfAllCustomers").list();
			for(Object obj[]:eplist){
				for(Object o:obj){
					System.out.println(o+"\t");
				}
				System.out.println("");
			}
			tx.commit();
			session.close();
		}catch(Exception e){
			e.printStackTrace();
			if(tx!=null)
				tx.rollback();
		}
	}
}
```

Customer.hbm.xml

```
<hibernate-mapping package="com.jtcindian.hibernate">
	<class name="Customer" table="customers">
		<id name="cid" column="cid" type="int">
			<generator class="increment" />
		</id>
		<property name="cname" />
		<property name="email" />
		<property name="city" />
		<property name="cardBal" type="double" />
		<property name="cardType" />
		<property name="cardNo" type="int" />
```

```
        <property name="status" />
    </class>
    <sql-query name="JtcAllCustomers">
        <return alias="cts" class="Customer" />
        SELECT cts.cid AS {cts.cid},
        cts.cname AS {cts.cname},
        cts.email AS {cts.email},
        cts.cardBal AS {cts.cardBal},
        cts.cardType AS {cts.cardType},
        cts.cardNo AS {cts.cardNo},
        cts.status AS {cts.status}
        FROM customers cts
    </sql-query>

    <sql-query name="JtcActiveCustomers">
        <return alias="cts" class="Customer" />
        SELECT cts.cid AS {cts.cid},
        cts.cname AS {cts.cname},
        cts.email AS {cts.email},
        cts.status AS {cts.status}
        FROM customers cts
        WHERE cts.status=:st
    </sql-query>

    <sql-query name="EmailsOfAllCustomers">
        <return-scalar column="email" type="string" />
        SELECT cts.email AS {cts.email}
        FROM customers cts
    </sql-query>

    <sql-query name="EmailsAndStatusOfAllCustomers">
        <return-scalar column="email" type="string" />
        <return-scalar column="status" type="string" />
        SELECT cts.email AS {cts.email}.
        cts.status AS status
        FROM customers cts
        </sql-query>
</hibernate-mapping>
```

Working with primary keys

- Hibernate supports to configure both the types of primary keys.
  1. Simple primary keys
  2. Composite primary keys
- When you use single column as primary key then it is called as simple primary key.
- Use<id> tag or @ Id annotation to configure simple primary key.
- When you use combination of two or more columns as primary key then it is called as composite primary key.
- Use <composite-id> tag or @ Embedded and @ Embeddable to configure composite primary key.

Simple primary keys

- Hibernate core provides various Built in simple primary key generators follows.
  1. **increment- Increment Generator(org.hibernate.id)**
  2. **hilo**
  3. **sequence (**) - Oracle -Sequence Generator**
  4. **seqhilo - Oracle sequence HiLoGenerator**
  5. **uuid**
  6. **guid   GUIDGenerator**
  7. **identity**
  8. **native**
  9. **assigned**
  10. **select**
- Above Names are called short-cut names of Generator Classes.

- For every built-In generator, one generator classes is implemented and provided in the package called org. hibernate id and all the generator classes are sub classes of Identifier generator and overriding generate () method.

Using increment:

```
<id name=”sid” column=”sid” type =”int”>
        <generator class =increment/>
</id>
<id name=”sid” column=”sid” type =”int”>
        <generator class =”org. hibernate.id.IncrementGenerator”/>
</id>
```

Note: Use only for integer type columns.
Note: Starts with 1 and increment by 1.

Using uuid:

```
<id name=”sid” column=”sid” type =”String”>
        <generator class = “uuid”/>
</id>
```

Note: Use only for integer type columns.
Note :Collects the IP address, timestamp and converts to hexadecimal and converts to String.

Using sequence:

```
Create or replace sequence SID_SEQ starts with 101 increments by 1;
Insert into students values (SID_SEQ. NEXTVAL, ‘Jtc’,’som’)
<id name=”sid” column=”sid” type=”int”>
        <generator class=”sequence”>
                <param name=”sequence”>SID_SEQ</param>
        </generator>
</id>
```

Note: Use only for integer type columns.

Using hilo:

```
<id name=”cid” column=”cid” type=”int”>
```

```
                <generator class="hilo">
                        <param name="table">hi_value</param>
                        <param name="column">next_value</param>
                        <param name="max_lo">10</param>
                </generator>
        </id>
```

Note: Use only for integer type columns

Using seqhilo:

```
        <id name="sid" type="long" column="sid">
                <generator class="seqhilo">
                        <param name="sequence">SID_SEQ</param>
                        <param name="max_lo">100</param>
                </generator>
        <id>
```

Note : Use only for integer type columns.

Example using Hilo algorithm

Jtc31: files required

1. Jtc31. java	**2. Customer.java**
3. Customer. Hbm. xml	**4. CHibernate Util.java**
5. Hibernate. Cfg. xml	

Jtc31.java

```
package com.jtcindia.hibernate;

public class Jtc31 {

        public static void main(String[] args) {
                try {
                        SessionFactory sf = CHibernateutil.getSessionFactory();
                        Session session = sf.openSession();
                        Transaction tx = session.beginTransaction();
                        Customer cust = new Customer("som", "som@jtcindai.com",
123456);
                        Integer it = (Integer) session.save(cust);
                        System.out.println(it.intValue());
                        tx.commit();
                        session.close();
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
}
```

Customer.java

```
package com.jtcindia.hibernate;

public class Customer {
        private String cid;
        private String cname;
        private String email;
        private long phone;
        public Customer() {
        }
        public Customer(String cname, String email, long phone) {
                super();
                this.cname = cname;
                this.email = email;
                this.phone = phone;
        }
        //Setters and Getters
}
```

Customer.hbm.xml

```
<hibernate-mapping package="com.jtcindian.hibernate">
        <class name="Customer" table="customers">
                <id name="cid" column="cid" type="int">
                        <generator class="hilo">
                <param name="table">hi_value</param>
                <param name="column">next_value</param>
                <param name="max_lo">10</param>
            </generator>
        </id>
        <property name="cname" />
        <property name="email" type="string" />
        <property name="phone" column="phone" type="long" />
    </class>
</hibernate-mapping>
```

Example using quid algorithm

Jtc32: files required

1. Jtc32. java	**2. Customer.java**
3. Customer.hbm. xml	**4. CHibernateUtil.java**
5. Hibernate.cfg. xml	

Jtc32.java

```
package com.jtcindia.hibernate;

public class Jtc32 {

    public static void main(String[] args) {
        try {
            SessionFactory sf = CHibernateutil.getSessionFactory();
            Session session = sf.openSession();
            Transaction tx = session.beginTransaction();
            Customer cust = new Customer("C-101","som",
"som@jtcindai.com", 123456);
            String customerId=session.save(cust).toString();
            System.out.println(customerId);
```

```
                        Customer
cu=(Customer)session.load(Customer.class,"40288184488e20a501488e20a766000
1");

        System.out.println(cu.getCid()+"\t"+cu.getCname()+"\t"+cu.getEmail()+"\t"
+cu.getPhone());
                        tx.commit();
                        session.close();
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
}
```

Customer.java

```
package com.jtcindia.hibernate;

public class Customer {
        private String customerId;
        private String cid;
        private String cname;
        private String email;
        private long phone;
        public Customer() {
        }
        public Customer(String cid,String cname, String email, long phone) {
                super();
                this.cid=cid;
                this.cname = cname;
                this.email = email;
                this.phone = phone;
        }
        //Setters and Getters
}
```

Customer.hbm.xml

```
<hibernate-mapping package="com.jtcindian.hibernate">
        <class name="Customer" table="customers">
                <id name="customerId" column="customerId" type="string">
                        <generator class="uuid"/>
                </id>
                <property name="cid"/>
                <property name="cname" />
                <property name="email" type="string" />
                <property name="phone" column="phone" type="long" />
        </class>
</hibernate-mapping>
```

Custom primary keys

- When Built-In Id Generators are not suitable for your requirements then you can write your OWN custom ID Generators.

Steps to write custom Id generators.

- Write your generator class by implementing IdentifierGenerator interface which is in org. hibernates. id package.
- Override the **generate ()** method as follows
  a. **public Serializable generate (SessionImplementor si, Object obj) throws HibernateException**
- Write the required id generation logic in generate() method.
- Register Custom ID Generator in hibernate mapping document

```
<id column ="sid" name="sid" type = "string">
        <generator class ="com.Jtcindia.id.SIDGenerator"/>
</id>
```

Composite primary keys:

- Hibernate does not provide any Built-In ID Generators for Composite Primary keys, you have to write your own Custom ID Generators.

With Hibernate core

Steps to configure composite primary keys

1. Write Custom Composite Primary key class by implementing Serializable interface and declare the required fields.

   Ex.

```
public class SID implements Serializable{
        int SID;
        String Bid.
        …
}
```

2. Declare the primary key filed type as custom Composite Primary key class.

```
public class Student {
        private SID studentId;
        private String sname;
        private String email.
        private String phone;
        public Student () {}
        public student (SID  studentId, String sname, String email,
String phone) {
                this. studentId=studentId;
                this.sname=sname;
                this. email=email.
                this. phone=phone;
}}
```

3. **Table students**

bid	sid	sname	email	phone

4. **Client code**

```
SID id = SIDGenerator.getNextSId (“30”);
Student stu= new Student (id, “Som”,”Som@Jtc”,”999”);
session. save (stu);
```

5. Configure in the mapping doc as follows.

```
<class name=”Student“  table=students”>
        <composite-id name=”studentId” class=”SID”>
                <key-property name=”bid” column=”bid” type=”string”/>
                <key-property name=”sid” column=”sid ‘type=”string”/>
        </composite-id>
        <property name=”sname”/>
        <property name=”email”/>
        <property name=”phone”/>
</class>
```

With Hibernate Annotations

Steps to configure Composite Primary keys

1. Write Custom Composite Primary key class by implementing Serializable interface and declare the required fields. Mark the class with **@Embeddable**

   Ex:
   ```
           @Embeddable
           public class SID implements Serializable {
                   int sid;
                   String bid;
                   ….
           }
   ```
2. Declare the primary key field type as custom composite primary key class.

   ```
   @Entity
   @Table (name=”students”)
   public class Student {
           @Id
           @Embedded
           @AttributeOverrides ({
           @AttributeOverride (name=”bid”, column=@Column (name = “bid”)),
           @AttributeOverride (name=”sid”, column=@column (name = “sid”))})
           private SID studentId;

           @Column (name =”sname”)
           private string sname;

           @Column (name =”email”)
           private String email;

           @Column (name =”phone”)
   ```

```
        private String phone;

        public Student () {
                }
        public Student (SID studentId, String sname, String email, String phone) {
                super ();
                this.studentId = studentId;
                this. sname = sname;
                this.email = email;
                this. phone = phone;
        }
    }
```

3. **Table students**

bid	sid	sname	email	phone

4. **Client code**

```
SID id = SIDGenerator. getNextSId (“30”);
Student stu= new Student (id, “Som”,”Som@Jtc”,”999”);
session. save (stu);
```

Custom primary key generation with hibernate core

Jtc33: files required

1. Jtc33. Java	2. Customer.java
3. Customer.hbm. xml	4. CHibernateUtil.java
5. Hibernate.cfg. xml	6. CIDGenerator.java

Jtc33.java

```
package com.jtcindai.hibernate;

public class Jtc33 {
        public static void main(String[] args) {
                try {
                        SessionFactory sf = ChibernateUtil.getSessionFactory();
                        Session session = sf.openSession();
                        Transaction tx = session.beginTransaction();
                        Customer cust = new Customer("som", "som@jtcindai.com",
123456);
                        String customerId = session.save(cust).toString();
                        System.out.println(customerId);
                        tx.commit();
                        session.close();
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
}
```

Customer.java

```
package com.jtcindai.hibernate;

public class Customer {
        private String cid;
        private String cname;
        private String email;
        private long phone;
        public Customer() {
        }
        public Customer(String cid,String cname, String email, long phone) {
                super();
                this.cid=cid;
                this.cname = cname;
                this.email = email;
                this.phone = phone;
        }
        //Setters and Getters
}
```

Customer.hbm.xml

```
<hibernate-mapping package="com.jtcindian.hibernate">
        <class name="Customer" table="customers">
                <id name="cid" column="cid" type="string">
                        <generator class="com.jtcindian.hibernate.CIDGenerator" />
                </id>
                <property name="cname" />
                <property name="email" type="string" />
                <property name="phone" column="phone" type="long" />
        </class>
</hibernate-mapping>
```

CIDGenerator.java

```
package com.jtcindai.hibernate;

import java.util.List;

public class CIDGenerator implements IdentifierGenerator {
        public Serializable generate(SessionImplementor si,Object obj)throws
HibernateException{
                String sid="C-001";
                try{
                        Configuration cfg=new Configuration.configure();
                        SessionFactory sf=cfg.buildSessionFactory();
                        Session s=sf.openSession();
                        Transaction tx=s.beginTransaction();
                        Query q1=s.createQuery("from Customer cust");
                        int size=q1.list().size();
                        if(size!=0){
                                Query query=s.createQuery("select max(cust.cid) from
Customer cust");
                                List list=query.list();
                                System.out.println(list.size());
                                Object o=list.get(0);
                                System.out.println(o);
                                String id="";
                                id=o.toString();
                                String p2=id.substring(2);
                                int x=Integer.parseInt(p2);
                                x=x+1;
                                if(x<=9){
                                        sid="C-00"+x;
                                }else if(x<=99){
                                        sid="C-0"+x;
                                }else if(x<=999){
                                        sid="C-"+x;
                                }
                        }
                }catch(Exception e){
                        e.printStackTrace();
                }
                return sid;
        }
}
```

Custom primary key generation with Hibernate annotation

Jtc34: files required

1. Jtc34. java	**2. Customer.java**
3. Customer. Hbm. xml	**4. CHibernate Util.java**
5. Hibernate. Cfg. xml	

Jtc34.java

```
package com.jtcindai.hibernate;

public class Jtc34 {

        public static void main(String[] args) {
                try {
                        SessionFactory sf = AHibernateUtil.getSessionFactory();
                        Session session = sf.openSession();
                        Transaction tx = session.beginTransaction();
                        Customer cust = new Customer("som", "som@jtcindai.com",
123456);
                        cust.setCid(CIDGenerator.getNextCid());
                        String cid = session.save(cust).toString();
                        System.out.println(cid);
                        tx.commit();
                        session.close();
                } catch (Exception e) {
                        e.printStackTrace();
                }
        }
}
```

Customer.java

```
package com.jtcindai.hibernate;
@Entity
@Table(name="customers")
public class Customer {
        @Id
        @Column(name="cid")
        private String cid;
        @Column(name="cname")
        private String cname;
        @Column(name="email")
        private String email;
        @Column(name="phone")
        private long phone;
        public Customer() {
        }
        public Customer(String cid,String cname, String email, long phone) {
                this.cid=cid;
                this.cname = cname;
                this.email = email;
                this.phone = phone;
        }
        //Setters and Getters
}
```

CIDGenerator.java

```
package com.jtcindai.hibernate;

import java.util.List;

public class CIDGenerator {
	public static String getNextCid(){
		String sid="C-001";
		try{
			SessionFactory sf=AHibernateUtil.getSessionFactory();
			Session s=sf.openSession();
			Transaction tx=s.beginTransaction();
			Query q1=s.createQuery("from Customer cust");
			int size=q1.list().size();
			if(size!=0){
				Query query=s.createQuery("select max(cust.cid) from 
Customer cust");
				List list=query.list();
				System.out.println(list.size());
				Object o=list.get(0);
				System.out.println(o);
				String id="";
				id=o.toString();
				String p2=id.substring(2);
				int x=Integer.parseInt(p2);
				x=x+1;
				if(x<=9){
					sid="C-00"+x;
				}else if(x<=99){
					sid="C-0"+x;
				}else if(x<=999){
					sid="C-"+x;
				}
```

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

Custom primary key generation with Hibernate cur

Jtc35: files required

1. Jtc35A. java	**2. Jtc35B.java**
3. SID.java	**4. Student.java**
5. Student. hbm.xml	**6. SIDGenerator. java**
7. Hibernate.cfg. xml	**8. CHibernateUtil.java**

Jtc35A.java

```
package com.jtcindia.hibernate;

public class Jtc35A {

        public static void main(String[] args) {
                try{
                        SessionFactory sf=CHibernateUtil.getSessionFactory();
                        Session session=sf.openSession();
                        Transaction tx=session.beginTransaction();
                        SID id=SIDGenerator.getNextSid("30");
                        Student stu=new Student(id,"som","som@jtcindia","9999");
                        session.save(stu);
                        tx.commit();
                        session.close();
                }catch(Exception e){
                        e.printStackTrace();
                }
```

```
	}
}
```

Jtc35B.java

```
package com.jtcindia.hibernate;

public class Jtc35B {

	public static void main(String[] args) {
		try{
			SessionFactory sf=CHibernateUtil.getSessionFactory();
			Session session=sf.openSession();
			Transaction tx=session.beginTransaction();
			Object obj=session.get(Student.class,new SID("30","001"));
			if(obj==null){
				System.out.println("Student Not Found");
			}else{
				Student stu1=(Student)obj;
				System.out.println(stu1.getStudentId().getBid());
				System.out.println(stu1.getStudentId().getSid());
				System.out.println(stu1.getSname());
				System.out.println(stu1.getEmail());
				System.out.println(stu1.getPhone());
			}
			tx.commit();
			session.close();
		}catch(Exception e){
			e.printStackTrace();
		}
	}
}
```

SID.java

```
package com.jtcindia.hibernate;
```

```
import java.io.Serializable;

public class SID implements Serializable{
	private String bid;
	private String sid;

	public SID(){}
	public SID(String bid,String sid){
		this.bid=bid;
		this.sid=sid;
	}
	//Setters and Getters
}
```

Student.java

```
package com.jtcindia.hibernate;

public class Student {
	private SID studentId;
	private String sname;
	private String email;
	private String phone;

	public Student(){}

	public Student(SID studentId, String sname, String email, String phone) {
		this.studentId = studentId;
		this.sname = sname;
		this.email = email;
		this.phone = phone;
	}
	//Setters and Getters
}
```

Student.hbm.java

```
<hibernate-mapping package="com.jtcindian.hibernate">
```

```
        <class name="Student" table="students">
                <composite-id name="studentId" class="SID">
                        <key-property name="bid" column="bid" type="string" />
                        <key-property name="sid" column="sid" type="string" />
                </composite-id>
                <property name="sname" />
                <property name="email" type="string" />
                <property name="phone" column="phone" type="string" />
        </class>
</hibernate-mapping>
```

SIDGenerator.java

```
package com.jtcindia.hibernate;

import java.util.List;

public class SIDGenerator {
        public static SID getNextSid(String bid) {
                SID sid = null;
                Transaction tx = null;
                try {
                        SessionFactory sf = CHibernateUtil.getSessionFactory();
                        Session session = sf.openSession();
                        tx = session.beginTransaction();
                        String hql1 = "from Student stu where stu.studentId.bid=?";
                        Query q1 = session.creatQuery(hql1);
                        q1.setString(0, bid);
                        List l1 = q1.list();
                        if (l1.size() == 0) {
                                sid = new SID(bid, "001");
                        } else {
                                String hql = "select max(stu.studentId.sid) from Student
stu where stu.studentId.bid=?";
                                Query q = session.createQuery(hql);
                                q.seString(0, bid);
                                String id = q.list().get(0).toString();
                                int x = Integer.parseInt(id);
                                x = x + 1;
                                x = x + 1;
                                if (x <= 9) {
                                        sid = new SID(bid, "00" + x);
                                } else if (x <= 99) {
                                        sid = new SID(bid, "0" + x);
                                } else if (x <= 999) {
                                        sid = new SID(bid, "" + x);
                                }
                        }
                } catch (Exception e) {
                        if(tx!=null){
                                tx.rollback();
                        }
                        e.printStackTrace();
                }
                return sid;

        }
}
```

Custom primary key generation with Hibernate Annotation

Jtc36: files required

1. Jtc36A. java	**2. Jtc36B.java**
3. SID.java	**4. Student.java**
5. SID Generator.java	**6. Hibernate.cfg.xml**
7. CHibernate Util.java	

SID.java

```
package com.jtcindia.hibernate;

import java.io.Serializable;
import javax.persistence.Embeddable;
```

```
@Embeddable
public class SID implements Serializable{
	private String bid;
	private String sid;
	public SID() {
	}
	public SID(String bid, String sid) {
		this.bid = bid;
		this.sid = sid;
	}
	//Setters and Getters
}
```

Student.java

```
package com.jtcindia.hibernate;

import javax.persistence.AttributeOverride;
import javax.persistence.AttributeOverrides;
import javax.persistence.Column;
import javax.persistence.Embedded;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "students")
public class Student {
	@Id
	@Embedded
	@AttributeOverrides({ @AttributeOverride(name = "bid", column =
@Column(name = "bid")),
				@AttributeOverride(name = "sid", column = @Column(name =
"sid")) })
	private SID studentId;
```

```
        @Column(name = "sname")
        private String sname;

        @Column(name = "email")
        private String email;

        @Column(name = "phone")
        private String phone;

        public Student() {
        }

        public Student(SID studentId, String sname, String email, String phone) {
                super();
                this.studentId = studentId;
                this.sname = sname;
                this.email = email;
                this.phone = phone;
        }
        // Sttters and Getters
}
```