Hibernate –

1. ORM Framework – Object Relational Mapping
2. JPA Implementation – Java Persistance API (javax.persistence package)
3. SessionFactory & Session
4. Hibernate.cfg.xml & <entity>.hbm.xml (Two configuration files of hibernate)
5. Cfg – configuration file, hbm- hibernate mapping file (optional)
6. In side config file, db properties (db url, username, password, driver & dialect), hbmddl-auto, show-sql
7. Persistence is the form of serialization where we store the state in database table.
8. Annotations can be used instead of hbm file
9. @Entity, @Table, @Id, @GenereatedValue, @Column, @OnetoMany, @ManytoMany
10. @NamedQuery, @NamedQueries, @NativeQuery
11. HQL – Hibernate Query Lang ( DB independent query)
12. Other popular JPA implementations are IBatis, EclipseLink, Generics

Criteria

### HCQL - Hibernate Criteria Query Language

1. Crietria c=session.createCriteria(Emp.**class**);//passing Class class argument
2. List list=c.list();

Example of HCQL to get the 10th to 20th record

1. Crietria c=session.createCriteria(Emp.**class**);
2. c.setFirstResult(10);
3. c.setMaxResult(20);
4. List list=c.list();

Example of HCQL to get the records whose salary is greater than 10000

1. Crietria c=session.createCriteria(Emp.**class**);
2. c.add(Restrictions.gt("salary",10000));//salary is the propertyname
3. List list=c.list();

Example of HCQL to get the records in ascending order on the basis of salary

1. Crietria c=session.createCriteria(Emp.**class**);
2. c.addOrder(Order.asc("salary"));
3. List list=c.list();

HCQL with Projection

We can fetch data of a particular column by projection such as name etc. Let's see the simple example of projection that prints data of NAME column of the table only.

1. Criteria c=session.createCriteria(Emp.**class**);
2. c.setProjection(Projections.property("name"));
3. List list=c.list();

|  |  |  |
| --- | --- | --- |
| Normal SQL | JPQL | HQL |
| Select \* from employee | From Employee | From Employee |
| Employee refers the table name | Employee refers to Entity Bean class | Employee refers to Entity Bean class |
|  |  |  |

@Entity

@Table(name = "student")

@NamedQueries({

@NamedQuery(name = "viewAllStudents", query = "FROM Student s")

@NamedQuery(name = "findStudentByID", query = "FROM Student s WHERE s.id = :id")

@NamedQuery(name = "findStudentByName", query = "FROM Student s WHERE s.name = :name")

})

public class Student {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "id")

private int id;

@Column(name = "first\_name")

private String firstName;

@Column(name = "last\_name")

private String lastName;

@Column(name = "email")

private String email;

public student(){

}

public Student(String firstName, String lastName, String email) {

this.firstName = firstName;

this.lastName = lastName;

this.email = email;

}

//getters and setters

}

Shortcut for force updating maven project ==== alt+f5

To run the project / file = Ctrl+F11

XML – eXtensible Markup Lang (Tag based lang)

XML Is case as well as space sensitive

All the tags in xml are user defined

Student

{

Int id;

String name;

String email;

}

XML

<students>

<student>

<id>100</id>

<name>abc</name>

<email>[abc@gmail.com</email](mailto:abc@gmail.com%3c/email)>

</student>

</students>

HTML – For viewing the data

XML – To store / transfer/exchange the data (data representation)

XML is OS, Platform, Architecture independent

UTF – Unicode Text Format

Valid xml

Well-formed =

XML schema is validated using -- DTD,XSD

DTD – Document Type Description

XSD – XML Schema Definition

Parser is an application which helps to extract data from xml file

<https://www.geeksforgeeks.org/how-to-reset-mysql-root-password-in-windows-using-cmd/>

Different ways of parsin xml file

**JAXP - Java API for XML Processing**

**sJAXB - Java and XML Binding**

**Marshalling**: converting XML to Java Objects

**UnMarshalling**: converting Java Objects to XML