Parul® University

NAAC A++

Enterprise Programming using JAVA
Chapter-4: Hibernet (ORM)

Prof. ARNIKA PATEL
Assistant Professor
Department of CSE

Parul® University

NAAC A++

Content

INDEX

HQL

- HQL stands for Hibernate Query Language, a language used within the Hibernate Framework for querying and manipulating data in relational databases.
- Hibernate converts HQL queries into SQL queries, which are used to perform database actions.
- Although Native SQL may be used directly with Hibernate, it is encouraged to utilize HQL wherever feasible to prevent database portability issues.
- HQL has many benefits. Some benefits are:

  HQL is database-independent.

  polymorphic queries supported which are type-safe.

  It is portable and easy to learn for Java programmers.

HQL

Hibernate Query Language (HQL) Clauses

There are many HQL clauses available to interact with relational databases, and several of them are listed below:

1. FROM Clause
2. SELECT Clause
3. WHERE Clause
4. ORDER BY Clause
5. UPDATE Clause
6. DELETE Clause
7. INSERT Clause

HQL

1. FROM Clause

To load a whole persistent object into memory, the FROM clause is used.

```
String hib = "FROM Student";

Query query = session.createQuery(hib);
List results = query.list()
```

HQL

2. SELECT Clause

The SELECT clause is used when only a *few attributes of an object* are required rather than the entire object.

```
String hib = "SELECT S.roll FROM Student S";
Query query = session.createQuery(hib);
List results = query.list();
```

HQL

3. WHERE Clause

Filtering records is done with the WHERE clause. It’s used to *retrieve only the records that meet a set of criteria*.

```
String hib = "FROM Student S WHERE S.id = 5";

Query query = session.createQuery(hib);
List results = query.list();
```

HQL

4. ORDER BY Clause

The ORDER BY clause is used to *sort the results* of an HQL query.

String hib = "FROM Student S WHERE S.id > 5 **ORDER BY** S.id **DESC**";

Query query = session.createQuery(hib);
List results = query.list();

HQL

5. UPDATE Clause

The UPDATE clause is required to *update the value* of an attribute.

```
String hib = "UPDATE Student set name=:n WHERE roll=:i";

Query q=session.createQuery(hib);
q.setParameter("n","John");
q.setParameter("i",23);

int status=q.executeUpdate();
System.out.println(status);
```

HQL

6. DELETE Clause

It is required to *delete a value* of an attribute.

String hib = "**DELETE** FROM Student WHERE id=10";

```
Query query=session.createQuery(hib);
query.executeUpdate();
```

HQL

7. INSERT Clause

It is required to *Insert values* into the relation.

String hib = "**INSERT** INTO Student(first_name, last_name)"
+
"SELECT first_name, last_name FROM backup_student";

Query query = session.createQuery(hib);
int result = query.executeUpdate();

HCQL

Hibernate Criteria Query Language (HCQL) provides a type-safe, object-oriented approach to querying database entities in Hibernate.

It's built using Java code instead of string-based queries like HQL, offering better readability, maintainability, and reduced risk of errors.

HCQL uses the Criteria interface, Restrictions class, and Order class to define query conditions, filters, and sorting.

HCQL

Criteria Interface

The **Criteria** interface is the main part of HCQL. It provides methods to add conditions, set limits, and define how results should be ordered.

You create a **Criteria** object by calling the **createCriteria()** method from the Hibernate **Session**.

Example of creating a Criteria object:

```
Criteria c = session.createCriteria(Emp.class);
```

HCQL

Advantages of HCQL

- HCQL provides built-in methods to add conditions, making it simple for Java developers to use.
- You can easily combine multiple conditions in a single query without worrying about query syntax.
- Since queries are built using Java code, it’s easier to spot mistakes and fix them.

HCQL

Commonly used methods of Criteria:

add(Criterion c): Adds a condition to the query.
addOrder(Order o): Specifies how the results should be ordered.
setFirstResult(int firstResult): Sets where to start fetching records (useful for pagination).
setMaxResults(int totalResult): Sets how many records to fetch.
list(): Executes the query and returns the results as a list.
setProjection(Projection p): Allows fetching specific columns instead of full objects.

HCQL

Order Class

The **Order** class is used to sort the results in ascending or descending order based on a specific property.
asc(property): Sorts in ascending order.
desc(property): Sorts in descending order.

PPT Content Resources Reference Sample:

1. **Book Reference**

   Jim Farley, William Crawford, David Flanagan. Java Enterprise in a Nutshell, O'Reilly

2. **Book Reference**

   Rocha, R., Purificação, J. (2018). Java EE 8 Design Patterns and Best Practices: Build Enterprise-ready Scalable Applications with Architectural Design Patterns. Germany: Packt Publishing..

3. **Website Reference**

   https://www.scribd.com/document/268349254/Java-8-Programming-Black-Book .

4. **Sources**

   https://developers.redhat.com/topics/enterprise-java

5. **Article**

   https://www.researchgate.net/publication/276412369_Advanced_Java_Programming

Parul® University | NAAC GRADE A++

https://paruluniversity.ac.in/