Hibernate

Stable Release (6.5)

Hibernate

Hibernate is an **O**bject-**R**elational **M**apping (ORM) solution for JAVA.

It provides a framework for mapping an object-oriented domain model to a relational database.

It is an open source persistent framework created by Gavin King in 2001.

JDBC

- JDBC stands for **Java Database Connectivity**.
- It provides a set of Java API for accessing the relational databases from Java program.
- These Java APIs enables Java programs to execute SQL statements and interact with any SQL compliant database.

Why Object Relational Mapping (ORM)?

-> what if we need to modify the design of our database after having developed the application?

-> Loading and storing objects in a relational database

Java Class

```
public class Employee
{
   private int id;
   private String first_name;
   private String last_name;
   private String email;
}
```

Table in a database

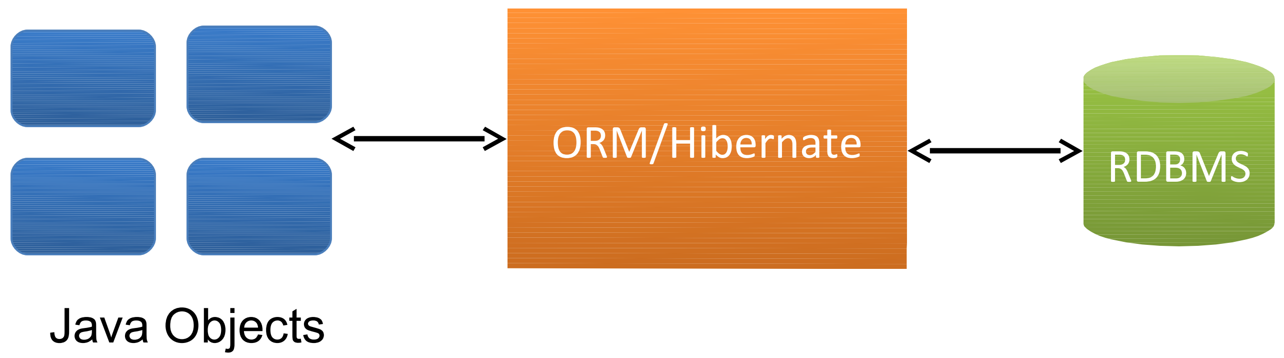
```
create table EMPLOYEE (
    id INT NOT NULL auto_increment,
    first_name VARCHAR(20) default NULL,
    last_name VARCHAR(20) default NULL,
    email VARCHAR(80) default NULL,
    PRIMARY KEY (id)
);
```

ORM:

ORM stands for **O**bject-**R**elational **M**apping (ORM) is a programming technique for converting data between relational databases and object oriented programming languages

JPA:

Java Persistence API (JPA) is a Java specification that provides certain functionality and standard to ORM tools. The **javax.persistence** package contains the JPA classes and interfaces.



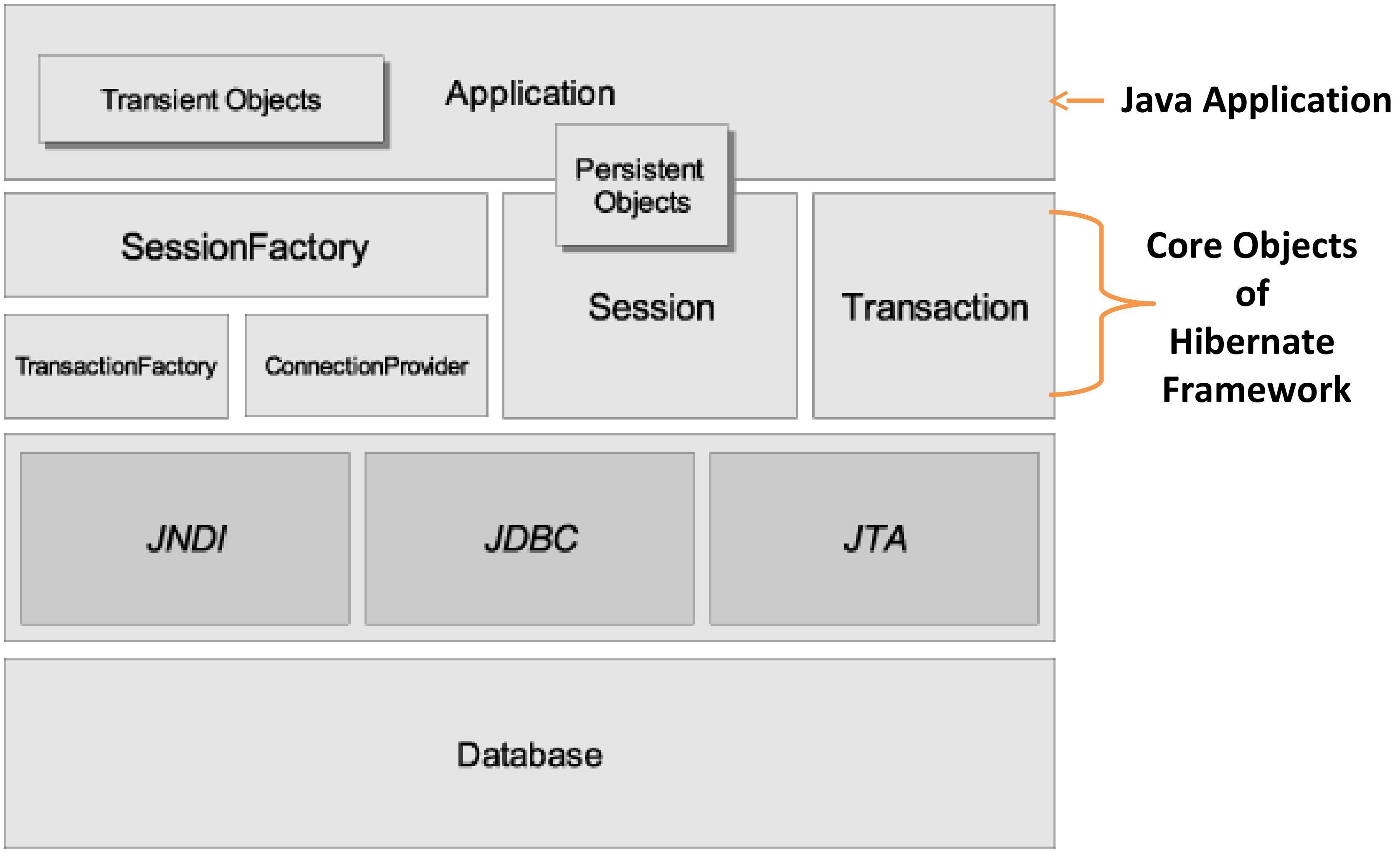
Hibernate sits between traditional Java objects and database server to handle all the works in persisting those objects based on the appropriate O/R mechanisms.

The mapping Java classes to database tables is accomplished through the configuration of an XML file or by using Java annotations.

Hibernate Advantages

- It takes care of mapping Java classes to database tables using XML files and without writing any line of code.
- Provides simple APIs for storing and retrieving Java objects directly to and from the database.
- If there is change in the database or in any table, then you need to change the XML file properties only.
- Hibernate does not require an application server to operate.
- Minimizes database access with smart fetching strategies.
- Provides simple querying of data.

Architecture



Architecture

Configuration Object - is the first Hibernate object you create in any Hibernate application. It is usually created only once during application initialization. It represents a configuration or properties file required by the Hibernate.

SessionFactory - is a factory of session and client of ConnectionProvider. The **org.hibernate.SessionFactory** interface provides factory method to get the object of Session.

- It is usually created during application start up and kept for later use.
- You would need one SessionFactory object per database using a separate configuration file.
- So, if you are using multiple databases, then you would have to create multiple SessionFactory objects.
- It holds second level cache (optional) of data.

Session - session object provides an interface between the application and data stored in the database. The **org.hibernate.Session** interface provides methods to insert, update and delete the object.

- A Session is used to get a physical connection with a database.
- It is factory of Transaction, Query and Criteria.
- It holds a first-level cache (mandatory) of data.

Architecture

Transaction

- The transaction object specifies the atomic unit of work. It is optional. The **org.hibernate.Transaction** interface provides methods for transaction management.

Query

- Query objects use SQL or Hibernate Query Language (HQL) string to retrieve data from the database and create objects.

Criteria

- Criteria objects are used to create and execute object oriented criteria queries to retrieve objects

ConnectionProvider

- It is a factory of JDBC connections. It abstracts the application from DriverManager or DataSource. It is optional.

TransactionFactory

- It is a factory of Transaction. It is optional.

Hibernate Configuration

- Hibernate requires to know in advance — where to find the mapping information that defines how your Java classes relate to the database tables.
- It also requires a set of configuration settings related to database and other related parameters.
- Such information is usually supplied as an XML file named **hibernate.cfg.xml**.

Hello World Application

- **hibernate.cfg.xml:** contains the database connection and schema details
- **Employee:** refers to a POJO (Plain Old Java Object) (hibernate annotations)
- **Employee.hbm.xml:** a mapping file for the Employee class
- **HibernateUtil:** user to creating the SessionFactory and Session Objects
- **TestClass:** test the code

Collections Mappings

- If an entity or class has collection of values for a particular variable, then we can map those values using any one of the collection interfaces available in java.
- Hibernate can persist instances of
  - **java.util.Map**
  - **java.util.Set**
  - **java.util.SortedMap**
  - **java.util.SortedSet**
  - **java.util.List**

Association Mappings

- Many-to-One
  - Mapping many-to-one relationship using Hibernate
- One-to-One
  - Mapping one-to-one relationship using Hibernate
- One-to-Many
  - Mapping one-to-many relationship using Hibernate
- Many-to-Many
  - Mapping many-to-many relationship using Hibernate

Component Mapping

- **Component** mapping is a mapping for a class having a reference to another class as a member variable.
  - An component is an object that is stored as an value rather than entity reference.
  - This is mainly used if the dependent object doesn't have primary key.
  - It is used in case of composition (HAS-A relation), that is why it is termed as component.

HQL

- Hibernate Query Language (HQL) is an object-oriented query language, similar to SQL, but instead of operating on tables and columns, HQL works with persistent objects and their properties.

```
Query query = session.createQuery(“from UserDetails”);
List results = query.list();
```

- HQL queries are translated by Hibernate into conventional SQL queries, which in turns perform action on database.

```
Query query = session.createQuery(“update UserDetails set name=:newName where
email=:emailID”);
query.setParameter(“newName”,”abcd”);
query.setParameter(“emailID”,”abcd@cdac.in”);
query.executeUpdate();
```

Named Query

- Named Query is way to use any query by some meaningful name.
  - It is like using alias names.
- There are two ways to define the named query in hibernate:
  - by annotation
  - by mapping file

```
@NamedQueries(

{
  @NamedQuery(
      name = “findUserByName”,
      query = “from UserDetails  ud  where ud.name=:name”
)
})
```

HCQL (Hibernate Criteria Query Language)

- The Hibernate Criteria Query Language (HCQL) is used to fetch the records based on the specific criteria.
- **Session** interface provides createCriteria() method, which can be used to create a Criteria object that returns instances of the persistence object's class when your application executes a criteria query

```
Criteria ct = session.createCriteria("UserDetails.class");
List list = ct.ist();
```