HIBERNATE NOTES

1) Drawbacks / Limitation of jdbc...
2) What is hibernate...
3) Hibernate terminology/(java persistence api)
4) What is orm & advantage of orm (object relational mapping)
5) Hibernate Architecture
6) Important component of hibernate
    a) Configuration
    b) Session Factory
    c) Session
    d) Query
    e) Criteria
    f) Transaction
7) Mapping Ways
    a) .hbm.xml file b) annotation
8) Hql
9) Hcql
10) Paging
11) Hibernate advanced mapping
        a) One to one
        b) One to many
        c) Many to many
12) Cascading types
13) Hibernate lifecycle

❖ **Difference between technology and framework**

1)Technology provides the basic platform and library to develop an application Where as Framework provides advanced platform and huge number of library to develop application

2)Technology is a semi finished product whereas a framework is a fully finished product.

HIBERNATE NOTES

❖ Drawback /Limitation of jdbc...

jdbc is database dependent technology. It means if we change databases we have to modify jdbc code and changing the database is a very difficult task in jdbc.
We have to represent every sql query as a string value.
In jdbc sql queries are database platform dependent.
Transaction management is very weak in jdbc.
Each time exception handling is mandatory in jdbc.

❖ What is hibernate...

Hibernate is an open source lightweight object relational framework which can be used to develop database applications.
In other words, hibernate is a database framework.
Open source doesn't not mean free of cost but also source code is also available.

❖ Hibernate Terminology...

- ❖ Persistence :- It is a process of storing data for a longer duration.
- ❖ Persistence storage :- It is the location where we can store data for a longer duration. For example databases can be tabular format or file format.
- ❖ Persistence data :- The actual data which can be present inside the table or file.
- ❖ Persistence operation :- The operations which can be performed on the persistence data like crud operations.
- ❖ Persistence logic :- The logic which can be developed to perform persistence operations like jdbc code.
- ❖ Persistence framework :- Framework which can be used to develop persistence logic i.e. hibernate framework.

HIBERNATE NOTES

❖ What is ORM...

- ❖ Orm stands for object relational mapping.
- ❖ It is a tool which can be used to build object oriented database applications.
- ❖ In the case of orm, the database table will be mapped with class and columns present inside the database table will be mapped with variables declared inside the class.
- ❖ We can create an object of class to store the data into the database table.

❖ Advantage of orm...

- ❖ Orm is database independent i.e. we can change the platform without changing the hibernate code.
- ❖ Changing the database is not a difficult task as compared to jdbc.
- ❖ Exception handling is not required to perform database operations.
- ❖ Hibernate provides strong support for transaction management.
- ❖ In hibernate we can perform object oriented operations.

❖ Hibernate architecture

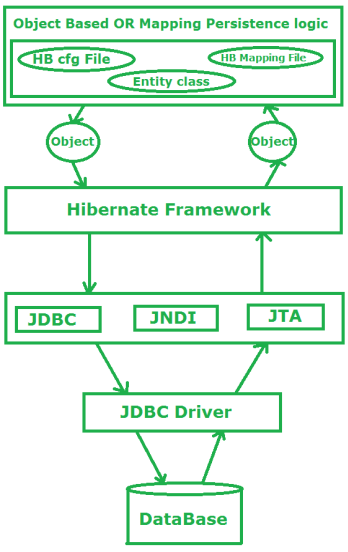

Fig: Working Flow of Hibernate framework to save/retrieve the data from the database in form of Object

HIBERNATE NOTES

- Hibernate architecture is a collection of different software & hardware or simple hibernate applications we have to create 3 files.
- .cfg.xml file -> This file contains information related to the database and in one project we have to create only one .cfg.xml file.
- .hbm.xml file -> This is a mapping file which contains information about database tables and in one project we can have multiple mapping files.
- .java file -> This file will contain a simple java class which has to be mapped with database table dependencies.

- **Important component of Hibernate...**
- Configuration :- Is a class declared inside the org.hibernate package.
  It activates hibernate framework or we can say execution of hibernate framework starts from cfg.xml file.
  This class contains a configure() method which reads configuration files as well as mapping files.
  It checks whether the cfg.xml file is syntactically correct or not. If the file is not valid then it will throw an exception if it is valid then it creates an object to represent the configuration file.
- Session Factory :- Session factory is an interface which is also declared inside the org.hibernate package.
  Session factory fetch the data from cfg object and create an object of sessionFactory.
  Session factory is used to create a session object.
  Session factory is immutable.

HIBERNATE NOTES

Session factory(I) contains the buildSessionFactory() method which gathers the meta-data which is in the cfg object.
From the cfg object it takes the jdbc information & creates a jdbc connection.
Session factory is thread safe.

- Session :- Session is an interface which is also declared inside the org.hibernate package.
  Session object is created based upon session factory object.
  Session opens the connection with database software through hibernate framework so by using session objects we can perform crud operations.
  Session is not thread safe.
  The Session object should not be kept open for a long time because the session is not thread safe so they should be created and destroyed as needed.
  The main function of the session is to offer, create, read and delete operations for instance mapped entity classes.

When the session object is created it can be one of the following three states.

1) Transient
2) Persistent
3) Detached

1)Transient :- This state is the first state of an entity object whenever an object of pojo(java bean class) class is instantiated by using the new operator then the object is in transient state.

```
Employee e=new Employee();
e.setEmpId(101);
e.setEmpName("ROHAN");
e.setEmpSalary(24999.99);
```

HIBERNATE NOTES

2) Persistent :- Whenever the object is connected with the hibernate session then the object moves into the persistent state.
There are two ways to convert transient state to persistent state

a) save() :- ses.save(e);
b) load() :- ses.load(e);

3)Detached :- The session has to be closed.
To close the session we can use ses.close() method or ses.clear();

- Transaction :- Transaction object is used whenever we perform any operations .
- Based upon those operations there is some change in the database .
- Transaction gives the instructions to the database to make the permanent change that happens because of operations by using commit() method.

- **Difference between load() and get().**

get()	load()
get() is used to fetch the data from the database table.	load() method is also used to fetch the data from the database table.
get() method is slower as compared to load().	load() is faster as compared to get().
If an object is not found then the given identifier will return NullPointerException.	If an object is not found then the given identifier will return an ObjectNotFoundException.

HIBERNATE NOTES

❖ Hibernate Annotations

Annotations are declared inside the javax.persistence pkg.

1)@Entity :- This annotation is used on the model classes by using "@Entity" and tells that the classes are entity beans.Marks a Java class as an entity, which corresponds to a database table.

2) @Table :- This annotation is used on the model classes by using "@Table" and tells that the class maps to the table name in the database.

3) @Column :- "@Column" is used for defining the column name in the database table.

4) @Id :- used on the attribute in a class to indicate that attribute is the primary key in the bean entity.

5) @joinColumn :- @JoinColumn is used to specify the mapping of a foreign key column for a relationship between two entities. This annotation is typically applied to a field or property that represents an association between two entities using a relational database. Using @JoinColumn allows you to customize the mapping of the foreign key column.

6) @JoinTable :- The @JoinTable annotation in Hibernate is used to define the mapping for a many-to-many relationship. In a many-to-many relationship, entities from one side can be associated with multiple entities from the other side, and vice versa.

HIBERNATE NOTES

7) inverseJoinColumn :- inverseJoinColumns is used to specify the columns in the join table that will store the foreign key to the other table.inverseJoinColumns attribute specifies the foreign key column(s) in the join table that reference the primary key column(s) of the inverse side

- **Generation Strategy**
  A generation strategy is an approach which can be used to generate primary keys.

  GenerationType.AUTO:-
  This strategy allows the framework user to choose the approach strategy based upon the database vendor. It can use identity or sequence depending on the database vendor.

  GenerationType.IDENTITY:
  This strategy is used for an auto-incremented database column to generate primary key values. It is commonly used with databases that support auto-incrementing columns, such as MySQL and PostgreSQL.

  GenerationType.SEQUENCE:
  This strategy involves using a database sequence to generate unique identifiers. It is suitable for databases that support sequences, such as Oracle.

- **Hql (hibernate query language)**
- HQL Stand for Hibernate Query Language.
- HQL is the own query language of Hibernate Framework.
- HQL can be used to perform bulk operations. example display Multiple records,update multiple records etc.

HIBERNATE NOTES

- HQL is purely object oriented in nature.
- HQL is platform independent.
- While writing HQL query we have to use class name instead of table name and variable name instead of column name.
- Keywords like SELECT, FROM, and WHERE, etc., are not case sensitive, but properties like table and column names are case sensitive in HQL.

- **Hcql (hibernate criteria query language)**
- The Hibernate Criteria Query Language (HCQL) is used to fetch the records based on the specific criteria. The Criteria interface provides methods to apply criteria such as retrieving all the records of tables whose salary is greater than 50000 etc.

Criteria (I) Methods :-

1) list() :- is used to fetch all records.

2) add(Criterion c) :- is used for all restrictions to criteria class.

3) addOrder(Order o):- Used to add Order.

4) setMaxResult(int n):- is used to fetch first n records;

5) setFirstResult(int n):- Exclude first n records & display rest of records

Projections © Methods :-

1) Projections.property(String pNAme) :- is used to fetch particular column.
2) Projections.projectionList() :- returns the ProjectionList object & used to fetch Multiple columns.

Order © Methods :-

HIBERNATE NOTES

1)Order.asc(String pName):- is used to fetch particular column in ascending order.
2)Order.desc(String pName):- is used to fetch particular column in descending order.

Restrictions © Methods :-

1) lt(String pNAme,Object value)
2) le(String pNAme,Object value)
3) gt(String pNAme,Object value)
4) ge(String pNAme,Object value)
5) ne(String pNAme,Object value)
6) eq(String pNAme,Object value)
7) between(String pName,object low,Object high)
8) like(String pNAme,String object)

❖ Cascading

Cascading is nothing but if we apply some operation on the parent object then this operation is automatically reflected on the child object.
Cascading refers to the propagation of operations (Such as insert,update,delete etc.) from one entity to another entity. In Other words when you perform operations on one entity cascading ensures that the same operations are applied to related entities as well.

CascadeType.persist :- This means that when you save the parent entity then the associated entity also saves.

CascadeType.remove :-This means that when you remove the parent entity then the associated entity also saves.

HIBERNATE NOTES

CascadeType.refresh :-This means that when you select the parent entity then the associated entity also selects.

CascadeType.detached :- This means that when you detach the parent entity then the associated entity also detaches.
A detached entity in hibernate is an object which is associated with the hibernate session.

CascadeType.merge :- This means that when you merge the parent entity then the associated entity also merges.
Merge is an operation in hibernate which is used to reattach an entity to a new hibernate session.

CascadeType.All :- We can perform all operations

- **ADVANCED MAPPING**

  One-To-One mapping
- One to one represents that a single entity is associated with a single instance of the other entity.
- One person has one passport, a passport is associated with a single person.
- We have one college ID, a college ID is uniquely associated with a person.

  One-To-Many mapping

  One To Many Mapping in Hibernate. In simple terms, one to many mapping means that one row in a table can be mapped to multiple rows in another table. For example, think of a Cart

system where we have another table for Items. A cart can have multiple items, so here we have one to many mapping.

Many-To-Many mapping

Many-to-Many mapping is usually implemented in a database using a Join Table. For example we can have a Cart and Item table and Cart_Items table for many-to-many mapping. Every cart can have multiple items and every item can be part of multiple carts, so we have a many to many mapping here.