Hibernate Q&A

Q.1) What is hibernate? Explain features of Hibernate?

🡪 Hibernate is an open source and lightweight ORM tool that is used to store, manipulate and retrieve data from the database.

It was started in 2001 by Gavin King as an alternative to EJB2 style entity bean.

Hibernate is a Java framework that simplifies the development of Java application to interact with the database. It is an open source, lightweight, ORM (Object Relational Mapping) tool. Hibernate implements the specifications of JPA (Java Persistence API) for data persistence.

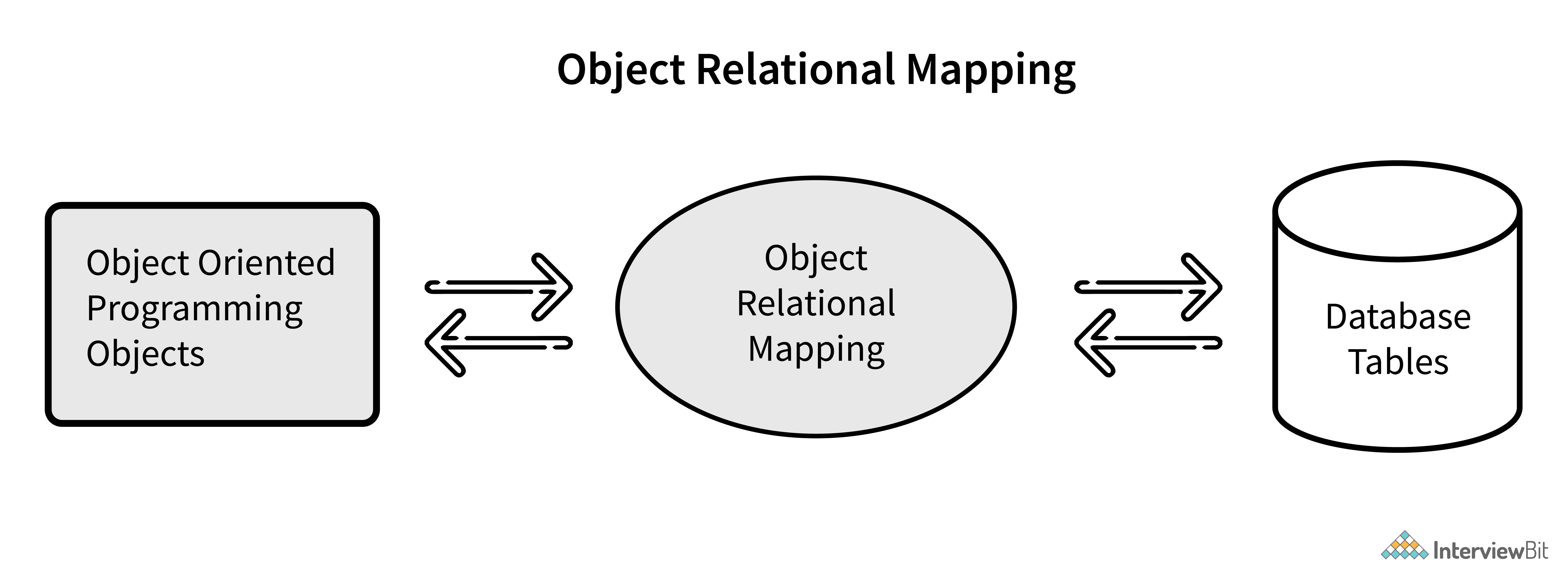
Hibernate is a java based persistence framework and an object –relational mapping(ORM) framework that basically allows a developer to map POJO –plain old java objects – to relational database tables.

The aim of hibernate framework is to free the developer form the common data persistence related complex configuration and tasks.

It does so by mapping the POJO objects with the database tables efficiently and most importantly in an abstract manner.

The developer need not know the underlying complications involved. Along with abstraction, the queries can be executed in a very efficient manner.

All these helps developers to save a lot of time involved in development.



Q.2) What is ORM tool?

ORM is an acronym for Object/Relational mapping. It is a programming strategy to map object with the data stored in the database. It simplifies data creation, data manipulation, and data access.

🡪

1. It is a tool for Database and Object oriented programming language.
2. It reduces efforts, time, and cost of developers.
3. It overcome mismatch between Object oriented programming language and database.
4. Java base ORM tools:

i)Hibernate

ii) iBatis

iii) TopLink



An ORM tool simplifies the data creation, data manipulation and data access. It is a programming technique that maps the object to the data stored in the database.

The ORM tool internally uses the JDBC API to interact with the database.

The object-relational mapping is a mechanism which is used to develop and maintain a relationship between an object and the relational database by mapping an object state into the database column. It converts attributes of programming code into columns of the table. It is capable of handling various database operations easily such as insertion, updation, deletion, etc.

Q.3) What are disadvantages of hibernate?

**What is disadvantage of hibernate?**

Can’t perform multiple insert operations.

Debugging is difficult as compare to JDBC.

Contain lots of boiler plate code.

it is not good for small project

Slow execution as it perform SQL queries at runtime.

Q.4) What is Session?

🡪

A session is an object that maintains the connection between java application and database. Session also has methods for storing, retrieving, modifying or deleting data from database using methods like persist(),load(),get(),update(),delete() etc.

Additionally, it has factory methods to return query, Criteria, and Transaction objects.

It holds a first-level cache (mandatory) of data.

Q.5) What are the methods present on Session?

A Session is used to get a physical connection with a database. The Session object is lightweight and designed to be instantiated each time an interaction is needed with the database. Persistent objects are saved and retrieved through a Session object.

The session objects should not be kept open for a long time because they are not usually thread safe and they should be created and destroyed them as needed. The main function of the Session is to offer, create, read, and delete operations for instances of mapped entity classes.

Instances may exist in one of the following three states at a given point in time −

* **transient** − A new instance of a persistent class, which is not associated with a Session and has no representation in the database and no identifier value is considered transient by Hibernate.
* **persistent** − You can make a transient instance persistent by associating it with a Session. A persistent instance has a representation in the database, an identifier value and is associated with a Session.
* **detached** − Once we close the Hibernate Session, the persistent instance will become a detached instance.

A Session instance is serializable if its persistent classes are serializable. A typical transaction should use the following idiom −

Session session = factory.openSession();

Transaction tx = null;

try {

tx = session.beginTransaction();

// do some work

...

tx.commit();

}

catch (Exception e) {

if (tx!=null) tx.rollback();

e.printStackTrace();

} finally {

session.close();

}

If the Session throws an exception, the transaction must be rolled back and the session must be discarded.

Session Interface Methods

There are number of methods provided by the **Session** interface, but I'm going to list down a few important methods only, which we will use in this tutorial. You can check Hibernate documentation for a complete list of methods associated with **Session** and **SessionFactory**.

|  |  |
| --- | --- |
| **Sr.No.** | **Session Methods & Description** |
| 1 | **Transaction beginTransaction()**  Begin a unit of work and return the associated Transaction object. |
| 2 | **void cancelQuery()**  Cancel the execution of the current query. |
| 3 | **void clear()**  Completely clear the session. |
| 4 | **Connection close()**  End the session by releasing the JDBC connection and cleaning up. |
| 5 | **Criteria createCriteria(Class persistentClass)**  Create a new Criteria instance, for the given entity class, or a superclass of an entity class. |
| 6 | **Criteria createCriteria(String entityName)**  Create a new Criteria instance, for the given entity name. |
| 7 | **Serializable getIdentifier(Object object)**  Return the identifier value of the given entity as associated with this session. |
| 8 | **Query createFilter(Object collection, String queryString)**  Create a new instance of Query for the given collection and filter string. |
| 9 | **Query createQuery(String queryString)**  Create a new instance of Query for the given HQL query string. |
| 10 | **SQLQuery createSQLQuery(String queryString)**  Create a new instance of SQLQuery for the given SQL query string. |
| 11 | **void delete(Object object)**  Remove a persistent instance from the datastore. |
| 12 | **void delete(String entityName, Object object)**  Remove a persistent instance from the datastore. |
| 13 | **Session get(String entityName, Serializable id)**  Return the persistent instance of the given named entity with the given identifier, or null if there is no such persistent instance. |
| 14 | **SessionFactory getSessionFactory()**  Get the session factory which created this session. |
| 15 | **void refresh(Object object)**  Re-read the state of the given instance from the underlying database. |
| 16 | **Transaction getTransaction()**  Get the Transaction instance associated with this session. |
| 17 | **boolean isConnected()**  Check if the session is currently connected. |
| 18 | **boolean isDirty()**  Does this session contain any changes which must be synchronized with the database? |
| 19 | **boolean isOpen()**  Check if the session is still open. |
| 20 | **Serializable save(Object object)**  Persist the given transient instance, first assigning a generated identifier. |
| 21 | **void saveOrUpdate(Object object)**  Either save(Object) or update(Object) the given instance. |
| 22 | **void update(Object object)**  Update the persistent instance with the identifier of the given detached instance. |
| 23 | **void update(String entityName, Object object)**  Update the persistent instance with the identifier of the given detached instance. |

Q.6) What is SessionFactory?

🡪

SessionFactory is an interface.

It contain all DB related property details (that it pulled from either hibernate.cfg.xml file or hibernate.properties file)

SessionFactory provides an instance of Session. It is a factory class that gives the Session objects based on the configuration parameters in order to establish the connection to the database.

As a good practice, the application generally has a single instance of SessionFactory. The internal state of a SessionFactory which includes metadata about ORM is immutable, i.e once the instance is created, it cannot be changed.

This also provides the facility to get information like statistics and metadata related to a class,or query execution, etc. It also holds second-level cache data if they enabled.

Q.7) Explain Difference between get() and load() methods:

The differences between get() and load() methods are given below.

In hibernate, get() and load() are two methods which is used to fetch data for the given identifier. They both belong to Hibernate session class. Get() method return null, If no row is available in the session cache or the database for the given identifier whereas load() method throws object not found exception.

| **Sr. No.** | **Key** | **Get()** | **Load()** |
| --- | --- | --- | --- |
| 1 | Basic | It  is used to fetch data from the database for the given identifier | It  is also used to fetch data from the database for the given identifier |
| 2 | Null Object | It object not found for the given identifier then it will return null object | It will throw object not found exception |
| 3 | Lazy or Eager loading | It returns fully initialized object so this method eager load the object | It always returns proxy object so this method is lazy load the object |
| 4 | Performance | It is slower than load() because it return fully initialized object which impact on the performance of the application | It is slightly faster. |
| 5. | Use Case | It should be used if **you are not sure** about the existence of instance. | It should be used if **you are sure** about the instance exists. |

Q.8) Explain Difference between save() and persist() methods :

Save() and persist() both methods are used for saving object in the database.

| **Sr. No.** | **Key** | **save()** | **persist()** |
| --- | --- | --- | --- |
| 1 | Basic | It stores object in database | It also stores object in database |
| 2 | Return Type | It return generated id and return type is serializable | It does not return anything. Its void return type. |
| 3 | Transaction Boundaries | It can save object within boundaries and outside boundaries | It can only save object within the transaction boundaries |
| 4. | Detached Object | It will create a new row in the table for detached object | It will throw persistence exception for detached object |
| 5. | Supported by | It is only supported by Hibernate | It is supported by Hibernate and also by JPA (Java Persistence API). |

Q.9) What are the stages of an object? Explain them?

Q.10) What is cache? How many types of cache? Which they are?

Q.11) What is 1st level cache?

Q.12) What is 2nd level cache?

Q.13) How enable 2nd level cache?

Q.14) How disable 1st level cache?

Q.15) What is purpose of hbm2ddl?

hbm2ddl. auto is a hibernate configuration property. It is used **to validate the numbers of columns and exports schema DDL to the database when the SessionFactory is created**.

Q.16) How to show sql\_query at runtime?

Q.17) In OneToMany-ManyToOne bidirectional how many tables are created by-default and if mappedby is used?

Q.18) Define NamedQueries? Explain syntax and Advantages?

Q.19) Define NativeQueries? Explain syntax and Advantage?

Q.20) what is the process for automatic id generation for any random number?

Q.21) What is dirty checking?

Q.22)What is 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.

## Advantages of Hibernate Framework

Following are the advantages of hibernate framework:

### 1) Open Source and Lightweight

Hibernate framework is open source under the LGPL license and lightweight.

### 2) Fast Performance

The performance of hibernate framework is fast because cache is internally used in hibernate framework. There are two types of cache in hibernate framework first level cache and second level cache. First level cache is enabled by default.

### 3) Database Independent Query

HQL (Hibernate Query Language) is the object-oriented version of SQL. It generates the database independent queries. So you don't need to write database specific queries. Before Hibernate, if database is changed for the project, we need to change the SQL query as well that leads to the maintenance problem.

### 4) Automatic Table Creation

Hibernate framework provides the facility to create the tables of the database automatically. So there is no need to create tables in the database manually.

### 5) Simplifies Complex Join

Fetching data from multiple tables is easy in hibernate framework.

### 6) Provides Query Statistics and Database Status

Hibernate supports Query cache and provide statistics about query and database status.

### 5) Mention some of the advantages of using ORM over JDBC.

ORM has the following advantages over JDBC:

* Application development is fast.
* Management of transaction.
* Generates key automatically.
* Details of SQL queries are hidden.

Q.23) What are the advantages of Hibernate over JDBC?

🡪

1) Clean Readable Code:

Using Hibernate, helps in eliminating a lot of JDBC API-based boiler-plate codes, thereby making the code look cleaner and readable.

2) HQL(Hibernate Query Language)

🡪

Hibernate provides HQL which is closer to java and is object-oriented in nature. This helps in reducing the burden on developers for writing database independent queries. In JDBC, this is not the case, A developer has to know the database-specific codes.

3) Transaction Management

JDBC doesn’t support implicit transaction management. It is upon the developer to write transaction management code using commit and rollback methods. Whereas, Hibernate implicitly provides this feature.

4) Exception Handling

Hibernate wrap the JDBC exception and throws unchecked exception like

JDBCException or HibernateException. This along with the built-in transaction management system helps developers to avoid writing multiple try-catch blocks to handle exceptions. In the case of JDBC, it throws a checked exception called SQLException thereby mandating the developer to write try-catch blocks to handle this exception at compile time.

5) Special features

Hibernate supports OOPs features like inheritance, associations and also support collections. These are not available in JDBC.

24) what are some of the important interfaces of Hibernate framework?

🡪

Hibernate core interfaces are:

\*Configuration

\*SessionFactory

\*Session

\*Criteria

\*Query

\*Transaction

25) What do you think about the statement-“session being a thread safe object”?

🡪

No, Session is not a thread-safe object which means that any number of threads can access data from it simultaneously.

26) Can you explain what is lazy loading in hibernate?

🡪Lazy loading is mainly used for improving the application performance by helping to load the child object on demand.

It is to be noted that, since Hibernate 3 version, this feature has been enabled by default. This signifies that child objects are not loaded until the parent gets loaded.

Q.27) What is the difference between first level cache and second level cache?

🡪

Hibernate has 2 types of cache.

1. First level cache
2. Second level cache

|  |  |
| --- | --- |
| First level cache | Second level cache |
| 1. This is local to the Session object and cannot be shared between multiple sessions. | 1. This cache is maintained at the SessionFactory level and shared among all sessions in Hibernate. |
| 1. This cache is enabled by default and there is no way to disable it. | 1. This is disabled by default, but we can enable it through configuration. |
| 1. The first level cache is available only until the session is open, once the session is closed, the first level cache is destroyed. | 1. The second level cache is available through the applications life cycle, it is only destroyed and recreate when an application is closed. |

If an entity or object is loaded by calling the get() method then Hibernate first checked the first level cache. If it doesn’t find the object, then goes to the second level cache if configured. If the object is not found then it finally goes to the database and returns the object, if there is no corresponding row in the table then it returns null.

Q.28) What can you tell about Hibernate configuration file?

🡪

Hibernate Configuration File or hibernate.cfg.xml is one of the most required configuration files in Hibernate. By default, this file is placed under the src/main/resource folder.

The file contains database related configurations and session-related configurations.

Hibernate facilitates providing the configuration either in an XML file(like hibernate.cfg.xml) or a properties file(like hibernate.properties).

This file is used to define the below information:

\*Database connection details: Driver class, URL, username, and Password.

\*There must be one configuration file for each database used in the application, suppose if we want to connect with 2 databases, then we must create 2 configuration files with different names.

\*Hibernate properties: Dialect, show\_sql, second\_level\_cache, and mapping file names.

29) How do you create an immutable class in hibernate?

🡪

Immutable class in hibernate creation could be in the following way.

If we are using the XML form of configuration, then a class can be made immutable by making mutable=”false”;

The default value is true there which indicating that the class was not created by default.

In the case of using annotation, immutable classes in hibernate can also be created by using @Immutable annotation.

30) Can you explain the concept behind Hibernate Inheritance Mapping?

🡪

Java is an Object-Oriented Programming Language and Inheritance is one of the most important pillars of object-oriented principles.

To represent any models in Java, inheritance is most commonly used to simplify and simplify the relationship. But, there is a catch. Relational databases do not support inheritance. They have a flat structure.

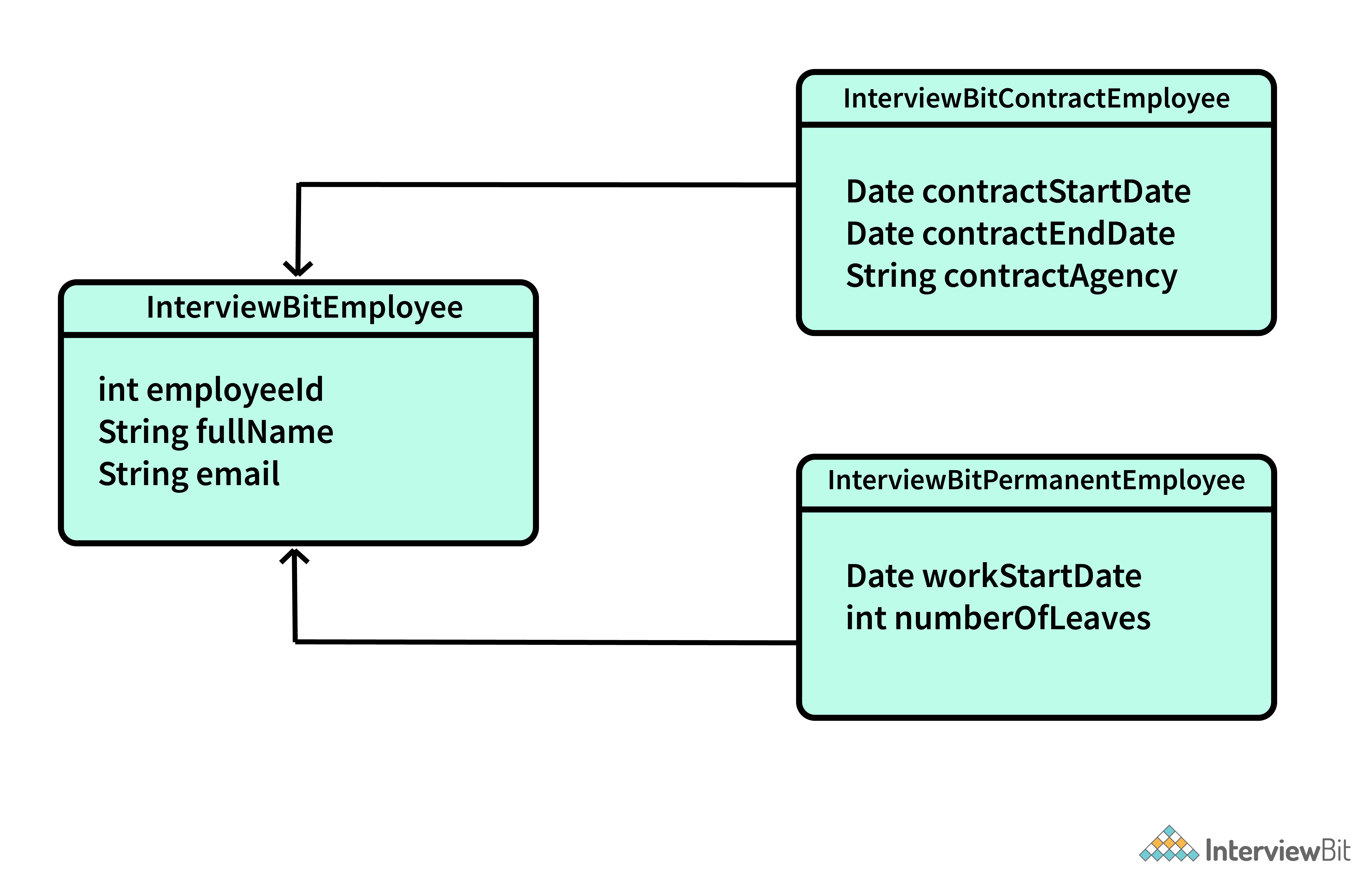
Hibernate’s Inheritance Mapping Strategies deal with solving how to hibernate being an ORM tries to map this problem between the inheritance of Java and flat structure of Databases.

Consider the example where we have to divide InterviewBitEmployee into Contract and Permanent Employees represented by IBContractEmployee and IBPermanentEmployee classes respectively. Now the task of hibernate is to represent these 2 employee types by considering the below restrictions:

The general employee details are defined in the parent InterviewBitEmployee class.

Contract and Permanent employee-specific details are stored in IBContractEmployee and IBPermanentEmployee classes respectively

The class diagram of this system is as shown below:



Hibernate’s Inheritance Mapping

There are different inheritance mapping strategies available:

* Single Table Strategy
* Table Per Class Strategy
* Mapped Super Class Strategy
* Joined Table Strategy

### 12. Is hibernate prone to SQL injection attack?

**SQL injection** **attack** is a serious vulnerability in terms of web security wherein an attacker can interfere with the queries made by an application/website to its database thereby allowing the attacker to view sensitive data which are generally irretrievable. It can also give the attacker to modify/ remove the data resulting in damages to the application behavior.

Hibernate does not provide immunity to SQL Injection. However, following good practices avoids SQL injection attacks. It is always advisable to follow any of the below options:

* Incorporate Prepared Statements that use Parameterized Queries.
* Use Stored Procedures.
* Ensure data sanity by doing input validation.

### 5) Mention some of the advantages of using ORM over JDBC.

ORM has the following advantages over JDBC:

* Application development is fast.
* Management of transaction.
* Generates key automatically.
* Details of SQL queries are hidden.

### 6) Define criteria in terms of Hibernate.

The objects of criteria are used for the creation and execution of the object-oriented criteria queries.

### 7) List some of the databases supported by Hibernate.

Some of the databases supported by Hibernate are:

* DB2
* MySQL
* Oracle
* Sybase SQL Server
* Informix Dynamic Server
* HSQL
* PostgreSQL
* FrontBase

### 9) Mention two components of Hibernate configuration object.

Database Connection

Class Mapping Setup

### 10) How is SQL query created in Hibernate?

The SQL query is created with the help of the following syntax:

Session.createSQLQuery

### 2) How is HQL query created?

The HQL query is created with the help of the following syntax:

Session.createQuery

### 13) How can we add criteria to a SQL query?

A criterion is added to a SQL query by using the Session.createCriteria.

### 14) Define persistent classes.

Classes whose objects are stored in a database table are called as persistent classes.

### 6) How many types of association mapping are possible in hibernate?

There can be 4 types of association mapping in hibernate.

1. One to One
2. One to Many
3. Many to One
4. Many to Many

### 7) Is it possible to perform collection mapping with One-to-One and Many-to-One?

No, collection mapping can only be performed with One-to-Many and Many-to-Many.

### 29) What is HQL (Hibernate Query Language)?

Hibernate Query Language is known as an object-oriented query language. It is like a structured query language (SQL).

The main advantage of HQL over SQL is:

1. You don't need to learn SQL
2. Database independent
3. Simple to write a query