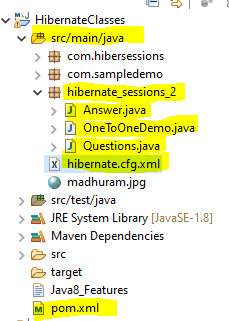
# Hibernate One to One Mapping - Annotation

Most of the times, database tables are associated with each other. There are many forms of association - **one-to-one**, **one-to-many** and **many-to-many** are at the broad level. These can be further divided into unidirectional and bidirectional mappings.



**package** hibernate\_sessions\_2;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.Transaction;

**import** org.hibernate.cfg.Configuration;

**public** **class** OneToOneDemo {

**public** **static** **void** main(String[] args) {

SessionFactory factory = **new** Configuration().configure("hibernate.cfg.xml").buildSessionFactory();

// question

Questions q1 = **new** Questions();

q1.setQuestionId(123);

q1.setQuestion("How many keywords in java?");

// answer

Answer a1 = **new** Answer();

a1.setAnswerId(123);

a1.setAnswer("In Java there are 52 keywords");

q1.setAnswer(a1);

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Questions q2 = **new** Questions();

q2.setQuestionId(456);

q2.setQuestion("How many keywords in python?");

Answer a2 = **new** Answer();

a2.setAnswerId(456);

a2.setAnswer("In Python there are 33 keywords");

q2.setAnswer(a2);

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Questions q3 = **new** Questions();

q3.setQuestionId(789);

q3.setQuestion("How many keywords in js?");

Answer a3 = **new** Answer();

a3.setAnswerId(789);

a3.setAnswer("In JS there are 48 keywords");

q3.setAnswer(a3);

Session s = factory.openSession();

Transaction tx = s.beginTransaction();

s.save(q1);

s.save(a1);

s.save(q2);

s.save(a2);

s.save(q3);

s.save(a3);

tx.commit();

s.close();

factory.close();

System.***out***.println("Tables created and");

System.***out***.println("Data inserted successfully!!!");

}

}

**package** hibernate\_sessions\_2;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

@Entity

**public** **class** Answer {

@Id

@Column(name = "answer\_id")

**private** **int** answerId;

**private** String answer;

**public** Answer() {

**super**();

}

**public** Answer(**int** answerId, String answer) {

**super**();

**this**.answerId = answerId;

**this**.answer = answer;

}

**public** **int** getAnswerId() {

**return** answerId;

}

**public** **void** setAnswerId(**int** answerId) {

**this**.answerId = answerId;

}

**public** String getAnswer() {

**return** answer;

}

**public** **void** setAnswer(String answer) {

**this**.answer = answer;

}

}

<?xml version = *"1.0"* encoding = *"utf-8"*?>

<!DOCTYPE hibernate-configuration SYSTEM

"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<property name=*"connection.driver\_class"*>com.mysql.cj.jdbc.Driver</property>

<property name=*"connection.url"*>jdbc:mysql://localhost:3306/empdb?useSSL=false</property>

<property name=*"connection.username"*>root</property>

<property name=*"connection.password"*>root</property>

<property name=*"dialect"*>org.hibernate.dialect.MySQL5Dialect</property>

<property name=*"hbm2ddl.auto"*>create</property>

<property name=*"show\_sql"*>true</property>

<!-- tag added to create a new table from the class. -->

<!-- <mapping class="com.hibersessions.Student" />

<mapping class="com.hibersessions.Address" />

<mapping class="com.hibersessions.StudentEmbed" /> -->

<mapping class=*"hibernate\_sessions\_2.Questions"* />

<mapping class=*"hibernate\_sessions\_2.Answer"* />

</session-factory>

</hibernate-configuration>

**package** hibernate\_sessions\_2;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.Transaction;

**import** org.hibernate.cfg.Configuration;

**public** **class** OneToOneDemo {

**public** **static** **void** main(String[] args) {

SessionFactory factory = **new** Configuration().configure("hibernate.cfg.xml").buildSessionFactory();

// question

Questions q1 = **new** Questions();

q1.setQuestionId(123);

q1.setQuestion("How many keywords in java?");

// answer

Answer a1 = **new** Answer();

a1.setAnswerId(123);

a1.setAnswer("In Java there are 52 keywords");

q1.setAnswer(a1);

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Questions q2 = **new** Questions();

q2.setQuestionId(456);

q2.setQuestion("How many keywords in python?");

Answer a2 = **new** Answer();

a2.setAnswerId(456);

a2.setAnswer("In Python there are 33 keywords");

q2.setAnswer(a2);

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Questions q3 = **new** Questions();

q3.setQuestionId(789);

q3.setQuestion("How many keywords in js?");

Answer a3 = **new** Answer();

a3.setAnswerId(789);

a3.setAnswer("In JS there are 48 keywords");

q3.setAnswer(a3);

Session s = factory.openSession();

Transaction tx = s.beginTransaction();

s.save(q1);

s.save(a1);

s.save(q2);

s.save(a2);

s.save(q3);

s.save(a3);

tx.commit();

s.close();

factory.close();

System.***out***.println("Tables created and");

System.***out***.println("Data inserted successfully!!!");

}

}

Oct 18, 2023 11:02:06 PM org.hibernate.Version logVersion

INFO: HHH000412: Hibernate ORM core version 5.6.15.Final

Oct 18, 2023 11:02:07 PM org.hibernate.annotations.common.reflection.java.JavaReflectionManager <clinit>

INFO: HCANN000001: Hibernate Commons Annotations {5.1.2.Final}

Oct 18, 2023 11:02:07 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl configure

WARN: HHH10001002: Using Hibernate built-in connection pool (not for production use!)

Oct 18, 2023 11:02:07 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator

INFO: HHH10001005: using driver [com.mysql.cj.jdbc.Driver] at URL [jdbc:mysql://localhost:3306/empdb?useSSL=false]

Oct 18, 2023 11:02:07 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator

INFO: HHH10001001: Connection properties: {password=\*\*\*\*, user=root}

Oct 18, 2023 11:02:07 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator

INFO: HHH10001003: Autocommit mode: false

Oct 18, 2023 11:02:07 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>

INFO: HHH000115: Hibernate connection pool size: 20 (min=1)

Oct 18, 2023 11:02:08 PM org.hibernate.dialect.Dialect <init>

INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQL5Dialect

Hibernate: drop table if exists Answer

Oct 18, 2023 11:02:09 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection

INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@76332405] for (non-JTA) DDL execution was not in auto-commit mode; the Connection 'local transaction' will be committed and the Connection will be set into auto-commit mode.

Hibernate: drop table if exists Questions

Hibernate: create table Answer (answer\_id integer not null, answer varchar(255), primary key (answer\_id)) engine=MyISAM

Oct 18, 2023 11:02:09 PM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection

INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@3bb87d36] for (non-JTA) DDL execution was not in auto-commit mode; the Connection 'local transaction' will be committed and the Connection will be set into auto-commit mode.

Hibernate: create table Questions (question\_id integer not null, question varchar(255), a\_id integer, primary key (question\_id)) engine=MyISAM

Hibernate: alter table Questions add constraint FKrq85tinm34mlrgxgqrpvgehw1 foreign key (a\_id) references Answer (answer\_id)

Oct 18, 2023 11:02:09 PM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService

INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]

Hibernate: select answer\_.answer\_id, answer\_.answer as answer2\_0\_ from Answer answer\_ where answer\_.answer\_id=?

Hibernate: select answer\_.answer\_id, answer\_.answer as answer2\_0\_ from Answer answer\_ where answer\_.answer\_id=?

Hibernate: select answer\_.answer\_id, answer\_.answer as answer2\_0\_ from Answer answer\_ where answer\_.answer\_id=?

Hibernate: insert into Questions (a\_id, question, question\_id) values (?, ?, ?)

Hibernate: insert into Answer (answer, answer\_id) values (?, ?)

Hibernate: insert into Questions (a\_id, question, question\_id) values (?, ?, ?)

Hibernate: insert into Answer (answer, answer\_id) values (?, ?)

Hibernate: insert into Questions (a\_id, question, question\_id) values (?, ?, ?)

Hibernate: insert into Answer (answer, answer\_id) values (?, ?)

Hibernate: update Questions set a\_id=?, question=? where question\_id=?

Hibernate: update Questions set a\_id=?, question=? where question\_id=?

Hibernate: update Questions set a\_id=?, question=? where question\_id=?

Oct 18, 2023 11:02:10 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PoolState stop

INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/empdb?useSSL=false]

Tables created and

Data inserted successfully!!!

