Spring-Hibernate-Mapping-2022

**One to One UniDirectional**

@Data

@Entity

@Table(name = "book\_o2oU")

**public** **class** Book {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

}

@Data

@Entity

@Table(name = "author\_o2oU")

**public** **class** Author {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**@OneToOne(fetch = FetchType.*LAZY*, cascade = {CascadeType.*ALL*})**

**private** Book book;

}

**Table Design**

Graphical user interface, text, application, chat or text message

Description automatically generated Graphical user interface, text, application

Description automatically generated

**Repository**

@Repository

**public** **interface** AuthorBookRepository **extends** JpaRepository<Author, Long> {

}

**Service**

@Service

**public** **class** PersistenceService {

@Autowired

**private** AuthorBookRepository repo;

**public** **void** saveInfo(Author author) {

repo.save(author);

}

}

**public** **class** TestOne2OneUniDirectional {

**public** **static** **void** saveInfo(ApplicationContext appContext) {

PersistenceService pservice = appContext.getBean(PersistenceService.**class**);

Author author = **new** Author();

author.setName("Conan Doyle");

Book book = **new** Book();

book.setName("Cat Among Pigeons");

author.setBook(book);

pservice.saveInfo(author);

}

}

@SpringBootApplication

**public** **class** StandaloneJPAHibernateMappingApp {

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

ApplicationContext applicationContext = SpringApplication.*run*(StandaloneJPAHibernateMappingApp.**class**, args);

TestOne2OneUniDirectional.*saveInfo*(applicationContext);

}

}

@Data

@Entity(name="book\_o2oB")

@Table(name="book\_o2oB")

**public** **class** Book {

@Id

@Column(name = "BOOK\_ID")

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**@OneToOne**

**@MapsId**

**private** Author author;

}

**One To One Bidirectional**

@Data

@Entity(name = "author\_o2oB")

@Table(name = "author\_o2oB")

**public** **class** Author {

@Id

@Column(name = "AUTH\_ID")

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**@OneToOne(fetch = FetchType.*LAZY*,**

**cascade = { CascadeType.*ALL* })**

**private** Book book;

**public** **void** setBook(Book book) {

**if** (book == **null**) {

**if** (**this**.book != **null**) {

**this**.book.setAuthor(**null**);

}

} **else** {

book.setAuthor(**this**);

}

**this**.book = book;

}

}

**Table Design**

Text

Description automatically generated with medium confidenceGraphical user interface, text, table

Description automatically generated with medium confidence

**Service**

@Service(value = "one2oneBService")

**public** **class** PersistenceService {

@Autowired

**private** AuthorBookRepository repo;

**public** **void** saveInfo(Author author) {

repo.save(author);

}

}

**Repository**

@Repository(value = "one2oneBRepo")

**public** **interface** AuthorBookRepository

**extends** JpaRepository<Author, Long> {

}

**public** **class** **TestOne2OneBiDirectional** {

**public** **static** **void** saveInfo(ApplicationContext applicationContext) {

Author author = **new** Author();

author.setName("Larry Colins");

Book book = **new** Book();

book.setName("Indian Freedom Movement");

author.setBook(book);

PersistenceService service = applicationContext.getBean(PersistenceService.**class**);

service.saveInfo(author);

}

}

**One To Many**

@Data

@Entity (name = "book\_o2mU")

@Table(name = "book\_o2m")

**public** **class** Book {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**public** Book() {

}

**public** Book(String name) {

**this**.name = name;

}

}

@Data

@Entity (name = "author\_o2mU")

@Table(name = "author\_o2m")

**public** **class** Author {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**@OneToMany(cascade = CascadeType.*ALL*,**

**orphanRemoval = true)**

**@JoinColumn(name = "author\_id")**

**private** List<Book> bookList =

**new** ArrayList<Book>();

}

**Table Design**

Graphical user interface

Description automatically generated Text

Description automatically generated

**Repository**

@Repository (value = "one2oneURepo")

**public** **interface** AuthorBookRepository **extends** JpaRepository<Author, Long> {

}

**Service**

@Service (value = "one2oneUService")

**public** **class** PersistenceService {

@Autowired

**private** AuthorBookRepository repo;

**public** **void** saveInfo(Author author) {

repo.save(author);

}

}

**public** **class** TestOne2ManyUniDirectional {

**public** **static** **void** saveInfo(ApplicationContext appContext) {

PersistenceService pservice = appContext.getBean(PersistenceService.**class**);

Author author = **new** Author();

author.setName("Conan Doyle");

author.getBookList().add(**new** Book("Cat Among Pigeons"));

author.getBookList().add(**new** Book("Sherlock Homes"));

pservice.saveInfo(author);

}

}

**One 2 Many Bidirectional**

@Data

@Entity

@Table(name = "book\_o2mB")

**public** **class** Book {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**@ManyToOne(fetch = FetchType.*LAZY*)**

**private** Author author;

**public** Book() {

}

**public** Book(String name) {

**this**.name = name;

}

}

@Data

@Entity

@Table(name = "author\_o2mB")

**public** **class** Author {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**@OneToMany(cascade = CascadeType.*ALL*,**

**orphanRemoval = true)**

**@JoinColumn(name = "author\_id")**

**private** List<Book> bookList =

**new** ArrayList<Book>();

**public** **void** addBook(Book book) {

bookList.add(book);

book.setAuthor(**this**);

}

}

**Table Design**

Graphical user interface

Description automatically generated Text

Description automatically generated

**Repository**

@Repository

**public** **interface** AuthorBookRepository **extends** JpaRepository<Author, Long> {

}

**Service**

@Service

**public** **class** PersistenceService {

@Autowired

**private** AuthorBookRepository repo;

**public** **void** saveInfo(Author author) {

repo.save(author);

}

}

**public** **class** TestOne2ManyBiDirectional {

**public** **static** **void** saveInfo(ApplicationContext appContext) {

PersistenceService pservice = appContext.getBean(PersistenceService.**class**);

Author author = **new** Author();

author.setName("Conan Doyle");

author.getBookList().add(**new** Book("Cat Among Pigeons"));

author.getBookList().add(**new** Book("Sherlock Homes"));

pservice.saveInfo(author);

}

}

**Many to Many Unidirectional**

@Data

@Entity(name = "book\_m2mU")

@Table(name = "book\_m2mU")

**public** **class** Book {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**@ManyToMany(cascade = CascadeType.*ALL*)**

**@JoinTable(name = "author\_m2mU\_book\_m2mU",**

**joinColumns = @JoinColumn(name = "book\_id"),**

**inverseJoinColumns = @JoinColumn(name = "author\_id"))**

**private** Set<Author> authors;

**public** Book() {

}

**public** Book(String name) {

**this**.name = name;

}

}

@Data

@Entity(name = "author\_m2mU")

@Table(name = "author\_m2mU")

**public** **class** Author {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**public** Author() {

}

**public** Author(String name) {

**this**.name = name;

}

}

**Table Design**

Table

Description automatically generated with medium confidence Graphical user interface

Description automatically generated with medium confidence Graphical user interface, application

Description automatically generated

**Repository**

@Repository(value = "m2mURepo")

**public** **interface** BookRepository **extends** JpaRepository<Book, Long> {

}

**Service**

@Service(value = "m2mUService")

**public** **class** PersistenceService {

@Autowired

**private** BookRepository bookRepo;

**public** **void** saveBookInfo(Book book) {

bookRepo.save(book);

}

}

**public** **class** TestMany2ManyUniDirectional {

**public** **static** **void** saveInfo(ApplicationContext appContext) {

Book book = **new** Book("Live Happily");

Author author3 = **new** Author("Chetan Bhagat");

Author author4 = **new** Author("R. K. Narayan");

Set<Author> generalAuthors = **new** HashSet<>();

generalAuthors.add(author3);

generalAuthors.add(author4);

book.setAuthors(generalAuthors);

PersistenceService pservice = appContext.getBean(PersistenceService.**class**);

pservice.saveBookInfo(book);

}

}

**Many to Many Bidirectional**

@Data

@Entity(name = "book\_m2mB")

@Table(name = "book\_m2mB")

**public** **class** Book {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**@ManyToMany(cascade = CascadeType.*ALL*)**

**@JoinTable(name = "author\_m2mB\_book\_m2mB",**

**joinColumns = @JoinColumn(name = "book\_id"),**

**inverseJoinColumns = @JoinColumn(name = "author\_id"))**

**private** List<Author> authors;

**public** Book() { }

**public** Book(String name) {

**this**.name = name;

}

}

@Data

@Entity(name = "author\_m2mB")

@Table(name = "author\_m2mB")

**public** **class** Author {

@Id

@GeneratedValue

**private** **long** id;

@Column(name = "name")

**private** String name;

**@ManyToMany(cascade = CascadeType.*ALL*,**

**mappedBy = "authors")**

**private** List<Book> bookList =

**new** ArrayList<Book>();

}

**Table Design**

Table

Description automatically generated Text

Description automatically generated Table

Description automatically generated

**Repository**

@Repository(value = "Auth\_m2mBRepo")

**public** **interface** AuthorRepository **extends** JpaRepository<Author, Long> {

}

@Repository(value = "Book\_m2mBRepo")

**public** **interface** BookRepository **extends** JpaRepository<Book, Long> {

}

**Service**

@Service

**public** **class** PersistenceService {

@Autowired

**private** AuthorRepository authRepo;

@Autowired

**private** BookRepository bookRepo;

**public** **void** saveAuthorInfo(Author author) { authRepo.save(author); }

**public** **void** saveBookInfo(Book book) { bookRepo.save(book); }

}

**public** **class** TestMany2ManyBiDirectional {

**public** **static** **void** saveInfo(ApplicationContext appContext) {

PersistenceService pservice = appContext.getBean(PersistenceService.**class**);

Author author1 = **new** Author();

author1.setName("Conan Doyle"); //Has written two books

List<Author> classicAuthors = **new** ArrayList<>();

classicAuthors.add(author1);

Book book2 = **new** Book("Sherlock Homes");

Book book3 = **new** Book("Cat Among The Pigeons");

List<Book> classicBooks = **new** ArrayList<Book>();

classicBooks.add(book2);

classicBooks.add(book3);

author1.setBookList(classicBooks);

book2.setAuthors(classicAuthors);

book3.setAuthors(classicAuthors);

// ~~~~~~~~~~~~~~~~~~~~~~~~~

Book book1 = **new** Book("Live Happily"); // Written by two authors

Author author3 = **new** Author();

author3.setName("Chetan Bhagat");

Author author4 = **new** Author();

author4.setName("R. K. Narayan");

List<Author> generalAuthors = **new** ArrayList<>();

generalAuthors.add(author3);

generalAuthors.add(author4);

book1.setAuthors(generalAuthors);

List<Book> generalBooks = **new** ArrayList<>();

generalBooks.add(book1);

author3.setBookList(generalBooks);

author4.setBookList(generalBooks);

pservice.saveAuthorInfo(author1); // Working

// -- Get all the books id written by one author, just like ISBN no

// SELECT id, book\_id from author\_m2mu a, author\_m2mu\_book\_m2mu ab WHERE a.id = ab.author\_id;

pservice.saveBookInfo(book1);

/\*

which book was written by many authors

SELECT a.id, a.name, b.name FROM author\_m2mu a, book\_m2mu b WHERE b.id =

(SELECT ab.book\_id FROM author\_m2mu\_book\_m2mu ab

GROUP BY ab.book\_id HAVING count(ab.book\_id) >= 2);

\*/

}

}

@SpringBootApplication

**public** **class** StandaloneJPAHibernateMappingApp {

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

ApplicationContext applicationContext =

SpringApplication.*run*(StandaloneJPAHibernateMappingApp.**class**, args);

// TestOne2OneUniDirectional.saveInfo(applicationContext);

// TestOne2OneBiDirectional.saveInfo(applicationContext);

// One To Many

// TestOne2ManyUniDirectional.saveInfo(applicationContext);

// TestOne2ManyBiDirectional.saveInfo(applicationContext);

// Many to Many

// TestMany2ManyUniDirectional.saveInfo(applicationContext);

TestMany2ManyBiDirectional.*saveInfo*(applicationContext);

}

}