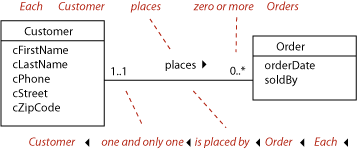
Objects and Association

<http://www.tomjewett.com/dbdesign/dbdesign.php?page=subkeys.php>

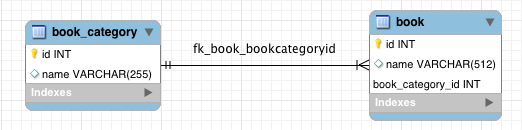
https://hellokoding.com/jpa-many-to-many-relationship-mapping-example-with-spring-boot-maven-and-mysql/



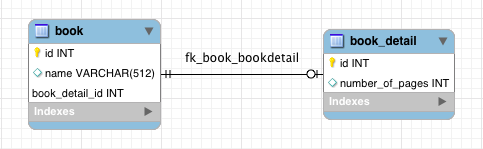
“Each customer places *zero or more* orders.” (\* in the diagram below means “many”, and any quantity more than one is the same as “many” in a database.)

“Each order is placed by *one and only one* customer.” (Bad English—passive voice—but makes sense!)

 Looking at the *maximum* multiplicity at each end of the line (1 and \* here), we call this a **one-to-many** association.



One to one foreign key



CREATE DATABASE IF NOT EXISTS `jpa\_onetoone\_foreignkey`;

USE `jpa\_onetoone\_foreignkey`;

--

-- Table structure for table `book\_detail`

--

DROP TABLE IF EXISTS `book\_detail`;

CREATE TABLE `book\_detail` (

`id` int(11) unsigned NOT NULL AUTO\_INCREMENT,

`number\_of\_pages` int(11) DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=4 DEFAULT CHARSET=utf8;

--

-- Table structure for table `book`

--

DROP TABLE IF EXISTS `book`;

CREATE TABLE `book` (

`id` int(11) unsigned NOT NULL AUTO\_INCREMENT,

`name` varchar(255) DEFAULT NULL,

`book\_detail\_id` int(11) unsigned DEFAULT NULL,

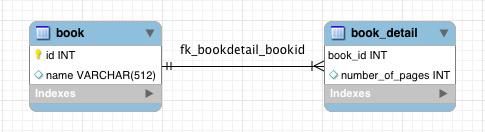
PRIMARY KEY (`id`),

KEY `fk\_book\_bookdetail` (`book\_detail\_id`),

CONSTRAINT `fk\_book\_bookdetail` FOREIGN KEY (`book\_detail\_id`) REFERENCES `book\_detail` (`id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB AUTO\_INCREMENT=4 DEFAULT CHARSET=utf8;

One to one shared primary key



CREATE DATABASE IF NOT EXISTS `jpa\_onetoone\_sharedprimarykey`;

USE `jpa\_onetoone\_sharedprimarykey`;

--

-- Table structure for table `book\_detail`

--

DROP TABLE IF EXISTS `book`;

CREATE TABLE `book` (

`id` int(11) unsigned NOT NULL AUTO\_INCREMENT,

`name` varchar(255) DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=18 DEFAULT CHARSET=utf8;

--

-- Table structure for table `book`

--

DROP TABLE IF EXISTS `book\_detail`;

CREATE TABLE `book\_detail` (

`book\_id` int(11) unsigned NOT NULL,

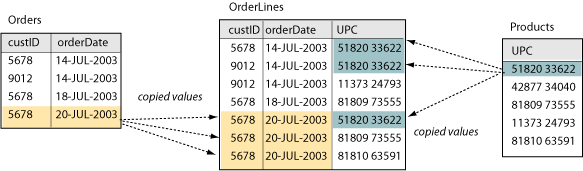
`number\_of\_pages` int(11) DEFAULT NULL,

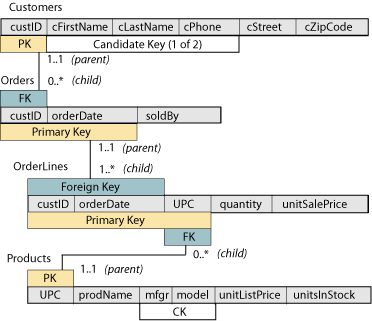
PRIMARY KEY (`book\_id`),

CONSTRAINT `fk\_bookdetail\_bookid` FOREIGN KEY (`book\_id`) REFERENCES `book` (`id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

Many to Many



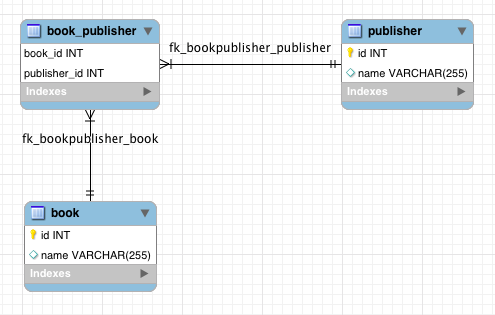


Each Order is associated with *one or more* OrderLines.”

“Each OrderLine is associated with *one and only one* Order.”

“Each OrderLine is associated with *one and only one* Product.”

“Each Product is associated with *zero or more* OrderLines.”



CREATE DATABASE IF NOT EXISTS `jpa\_manytomany`;

USE `jpa\_manytomany`;

--

-- Table structure for table `book`

--

DROP TABLE IF EXISTS `book`;

CREATE TABLE `book` (

`id` int(10) unsigned NOT NULL AUTO\_INCREMENT,

`name` varchar(255) DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=15 DEFAULT CHARSET=utf8;

--

-- Table structure for table `book\_publisher`

--

DROP TABLE IF EXISTS `book\_publisher`;

CREATE TABLE `book\_publisher` (

`book\_id` int(10) unsigned NOT NULL,

`publisher\_id` int(10) unsigned NOT NULL,

PRIMARY KEY (`book\_id`,`publisher\_id`),

KEY `fk\_bookpublisher\_publisher\_idx` (`publisher\_id`),

CONSTRAINT `fk\_bookpublisher\_book` FOREIGN KEY (`book\_id`) REFERENCES `book` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT `fk\_bookpublisher\_publisher` FOREIGN KEY (`publisher\_id`) REFERENCES `publisher` (`id`) ON DELETE CASCADE ON UPDATE CASCADE

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--

-- Table structure for table `publisher`

--

DROP TABLE IF EXISTS `publisher`;

CREATE TABLE `publisher` (

`id` int(10) unsigned NOT NULL AUTO\_INCREMENT,

`name` varchar(255) DEFAULT NULL,

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=19 DEFAULT CHARSET=utf8;

package com.hellokoding.jpa;

import com.hellokoding.jpa.model.Book;

import com.hellokoding.jpa.model.Publisher;

import com.hellokoding.jpa.repository.BookRepository;

import com.hellokoding.jpa.repository.PublisherRepository;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import javax.transaction.Transactional;

import java.util.HashSet;

@SpringBootApplication

public class HelloJpaApplication implements CommandLineRunner {

private static final Logger logger = LoggerFactory.getLogger(HelloJpaApplication.class);

@Autowired

private BookRepository bookRepository;

@Autowired

private PublisherRepository publisherRepository;

public static void main(String[] args) {

SpringApplication.run(HelloJpaApplication.class, args);

}

@Override

@Transactional

public void run(String... strings) throws Exception {

// save a couple of books

Publisher publisherA = new Publisher("Publisher A");

Publisher publisherB = new Publisher("Publisher B");

Publisher publisherC = new Publisher("Publisher C");

bookRepository.save(new HashSet<Book>(){{

add(new Book("Book A", new HashSet<Publisher>(){{

add(publisherA);

add(publisherB);

}}));

add(new Book("Book B", new HashSet<Publisher>(){{

add(publisherA);

add(publisherC);

}}));

}});

// fetch all books

for(Book book : bookRepository.findAll()) {

logger.info(book.toString());

}

// save a couple of publishers

Book bookA = new Book("Book A");

Book bookB = new Book("Book B");

publisherRepository.save(new HashSet<Publisher>() {{

add(new Publisher("Publisher A", new HashSet<Book>() {{

add(bookA);

add(bookB);

}}));

add(new Publisher("Publisher B", new HashSet<Book>() {{

add(bookA);

add(bookB);

}}));

}});

// fetch all publishers

for(Publisher publisher : publisherRepository.findAll()) {

logger.info(publisher.toString());

}

}

}