

**2.2 分库**

本文分库样例比较简单，根据数据库表中字段goods\_id的大小进行判断，如果goods\_id大于20则使用database0，否则使用database1。

**2.3 分表**

分样例比较简单，根据数据库表中字段goods\_type的数值的奇偶进行判断，奇数使用goods\_1表，偶数使用goods\_0表。

**2.4 代码流程**

流程大致是这样，在应用程序中我们操作虚拟表goods，但是当真正操作数据库的时候，会根据我们的分库分表规则进行匹配然后操作。

**3.代码实现**

本文使用SpringBoot2.0.3，SpringData-JPA，Druid连接池，和当当的sharding-jdbc。

**3.1 建表SQL**

创建表和数据库的SQL如下所示。

CREATE DATABASE database0;

USE database0;

DROP TABLE IF EXISTS `goods\_0`;

CREATE TABLE `goods\_0` (

`goods\_id` bigint(20) NOT NULL,

`goods\_name` varchar(100) COLLATE utf8\_bin NOT NULL,

`goods\_type` bigint(20) DEFAULT NULL,

PRIMARY KEY (`goods\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_bin;

DROP TABLE IF EXISTS `goods\_1`;

CREATE TABLE `goods\_1` (

`goods\_id` bigint(20) NOT NULL,

`goods\_name` varchar(100) COLLATE utf8\_bin NOT NULL,

`goods\_type` bigint(20) DEFAULT NULL,

PRIMARY KEY (`goods\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_bin;

CREATE DATABASE database1;

USE database1;

DROP TABLE IF EXISTS `goods\_0`;

CREATE TABLE `goods\_0` (

`goods\_id` bigint(20) NOT NULL,

`goods\_name` varchar(100) COLLATE utf8\_bin NOT NULL,

`goods\_type` bigint(20) DEFAULT NULL,

PRIMARY KEY (`goods\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_bin;

DROP TABLE IF EXISTS `goods\_1`;

CREATE TABLE `goods\_1` (

`goods\_id` bigint(20) NOT NULL,

`goods\_name` varchar(100) COLLATE utf8\_bin NOT NULL,

`goods\_type` bigint(20) DEFAULT NULL,

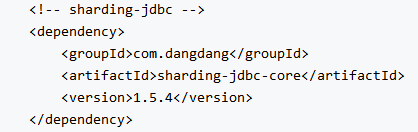
PRIMARY KEY (`goods\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8\_bin;

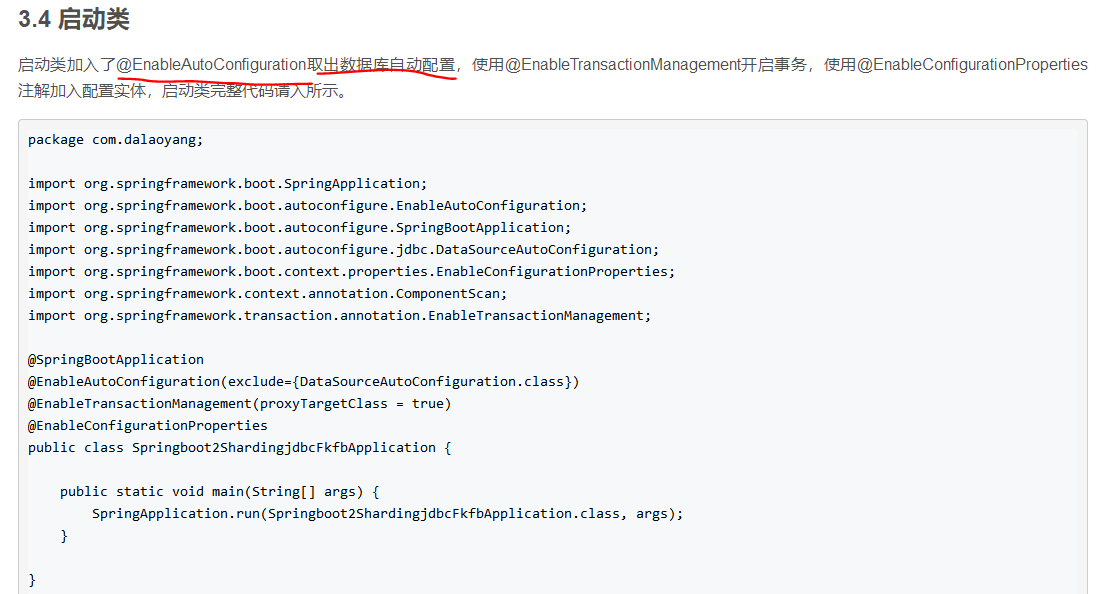
## 3.2 依赖文件

新建项目，加入当当的sharding-jdbc-core依赖和druid连接池，完整pom如下所示。













## 3.6 数据库配置

本文使用了两个实体来接收数据库信息，并且创建数据源，也可以采用别的方式。首先看一下Database0Config和Database1Config两个类的代码。

Database0Config类。



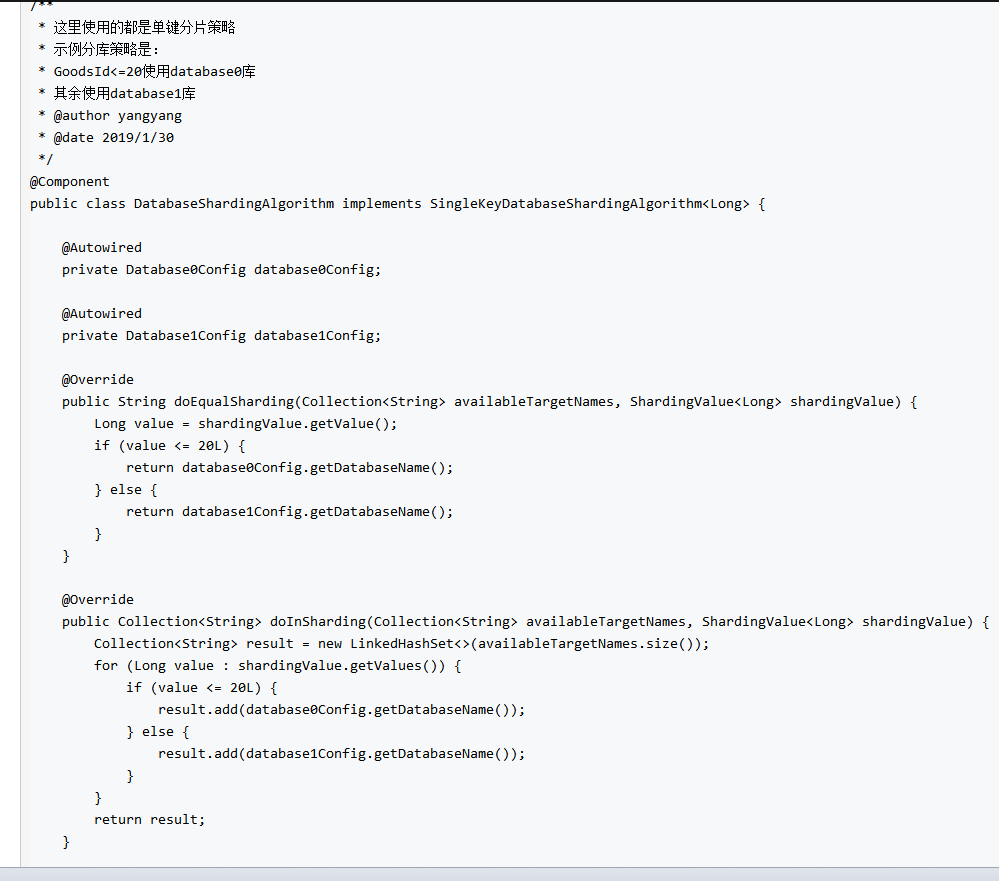


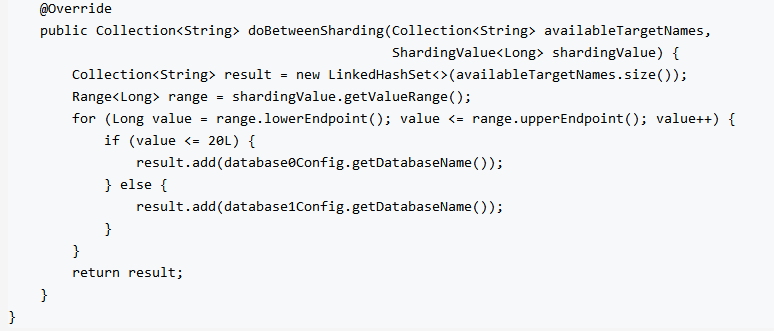
## 3.7 分库分表算法

由于这里只是简单的分库分表样例，所以分库类这里实现SingleKeyDatabaseShardingAlgorithm类，采用了单分片键数据源分片算法，需要重写三个方法，分别是：

* doEqualSharding：SQL中==的规则。
* doInSharding：SQL中in的规则。
* doBetweenSharding：SQL中between的规则。

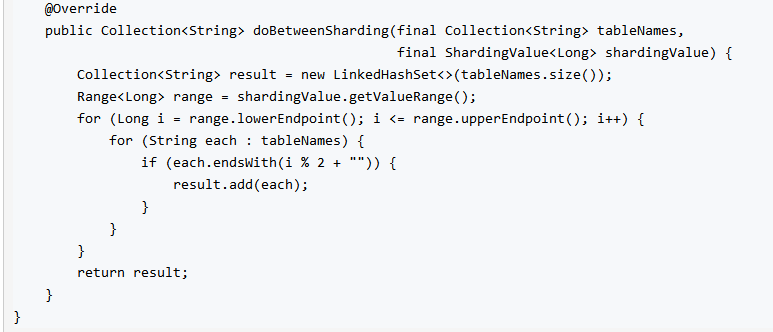
本文分库规则是基于值大于20则使用database0，其余使用database1，所以简单if，else就搞定了，分库算法类DatabaseShardingAlgorithm代码如下所示。



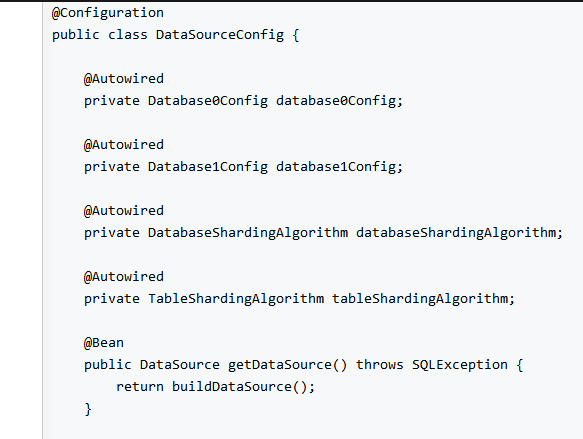


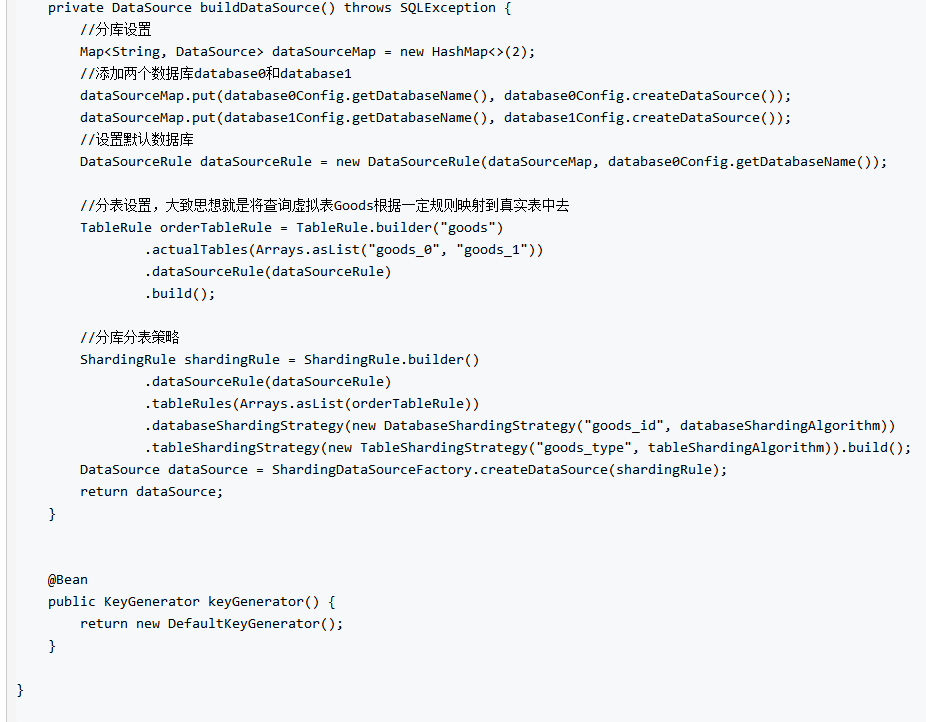
分表和分库类似，无非就是实现的类不一样，实现了SingleKeyTableShardingAlgorithm类，策略使用值奇偶分表，分表算法类TableShardingAlgorithm如代码清单所示。





接下来新建DataSourceConfig用于创建数据源和使用分库分表策略，其中分库分表策略会调用分库算法类和分表算法类，DataSourceConfig类代码如下所示。





## 3.8 Controller

接下来创建一个Controller进行测试，保存方法使用了插入40条数据，根据我们的规则，会每个库插入20条，同时我这里还创建了三个查询方法，分别是查询全部，between查询，in查询，还有删除全部方法。Controller类代码如下所示。



