参考地址：

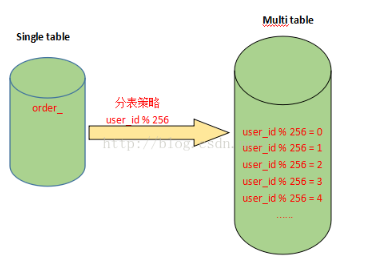
<https://blog.csdn.net/sunqingzhong44/article/details/84791074>

一. 分表

      场景：对于大型的互联网应用来说，数据库单表的记录行数可能达到千万级甚至是亿级，并且数据库面临着极高的并发访问。采用Master-Slave复制模式的MySQL架构，

只能够对数据库的读进行扩展，而对数据库的写入操作还是集中在Master上，并且单个Master挂载的Slave也不可能无限制多，Slave的数量受到Master能力和负载的限制。

因此，需要对数据库的吞吐能力进行进一步的扩展，以满足高并发访问与海量数据存储的需要！

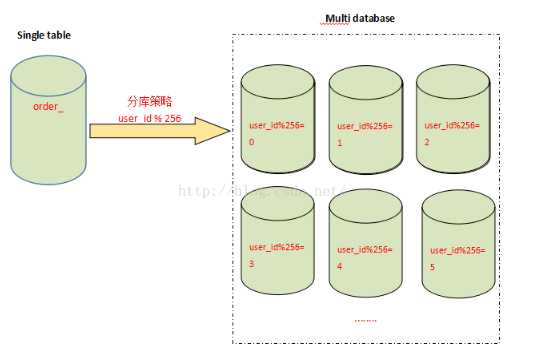


二. 分库

   场景：分表能够解决单表数据量过大带来的查询效率下降的问题，但是，却无法给数据库的并发处理能力带来质的提升。面对高并发的读写访问，当数据库master

服务器无法承载写操作压力时，不管如何扩展slave服务器，此时都没有意义了。

因此，我们必须换一种思路，对数据库进行拆分，从而提高数据库写入能力，这就是所谓的分库!



创建两个数据库order1,order2创建语句如下：

Order1:

|  |
| --- |
| CREATE DATABASE `order1` DEFAULT CHARACTER SET utf8;  USE `order1`;  DROP TABLE IF EXISTS `t\_address`;  CREATE TABLE `t\_address` (  `id` bigint(20) NOT NULL,  `code` varchar(64) DEFAULT NULL COMMENT '编码',  `name` varchar(64) DEFAULT NULL COMMENT '名称',  `pid` varchar(64) NOT NULL DEFAULT '0' COMMENT '父id',  `type` int(11) DEFAULT NULL COMMENT '1国家2省3市4县区',  `lit` int(11) DEFAULT NULL,  PRIMARY KEY (`id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8;  DROP TABLE IF EXISTS `t\_user0`;  CREATE TABLE `t\_user0` (  `id` bigint(20) NOT NULL,  `name` varchar(64) DEFAULT NULL COMMENT '名称',  `city\_id` int(12) DEFAULT NULL COMMENT '城市',  `sex` tinyint(1) DEFAULT NULL COMMENT '性别',  `phone` varchar(32) DEFAULT NULL COMMENT '电话',  `email` varchar(32) DEFAULT NULL COMMENT '邮箱',  `create\_time` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '创建时间',  `password` varchar(32) DEFAULT NULL COMMENT '密码',  PRIMARY KEY (`id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8;  DROP TABLE IF EXISTS `t\_user1`;  CREATE TABLE `t\_user1` (  `id` bigint(20) NOT NULL,  `name` varchar(64) DEFAULT NULL COMMENT '名称',  `city\_id` int(12) DEFAULT NULL COMMENT '城市',  `sex` tinyint(1) DEFAULT NULL COMMENT '性别',  `phone` varchar(32) DEFAULT NULL COMMENT '电话',  `email` varchar(32) DEFAULT NULL COMMENT '邮箱',  `create\_time` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '创建时间',  `password` varchar(32) DEFAULT NULL COMMENT '密码',  PRIMARY KEY (`id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |

Order2:

|  |
| --- |
| CREATE DATABASE `order2` DEFAULT CHARACTER SET utf8;  USE `order2`;  DROP TABLE IF EXISTS `t\_address`;  CREATE TABLE `t\_address` (  `id` bigint(20) NOT NULL,  `code` varchar(64) DEFAULT NULL COMMENT '编码',  `name` varchar(64) DEFAULT NULL COMMENT '名称',  `pid` varchar(64) NOT NULL DEFAULT '0' COMMENT '父id',  `type` int(11) DEFAULT NULL COMMENT '1国家2省3市4县区',  `lit` int(11) DEFAULT NULL,  PRIMARY KEY (`id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8;  DROP TABLE IF EXISTS `t\_user0`;  CREATE TABLE `t\_user0` (  `id` bigint(20) NOT NULL,  `name` varchar(64) DEFAULT NULL COMMENT '名称',  `city\_id` int(12) DEFAULT NULL COMMENT '城市',  `sex` tinyint(1) DEFAULT NULL COMMENT '性别',  `phone` varchar(32) DEFAULT NULL COMMENT '电话',  `email` varchar(32) DEFAULT NULL COMMENT '邮箱',  `create\_time` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '创建时间',  `password` varchar(32) DEFAULT NULL COMMENT '密码',  PRIMARY KEY (`id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8;  DROP TABLE IF EXISTS `t\_user1`;  CREATE TABLE `t\_user1` (  `id` bigint(20) NOT NULL,  `name` varchar(64) DEFAULT NULL COMMENT '名称',  `city\_id` int(12) DEFAULT NULL COMMENT '城市',  `sex` tinyint(1) DEFAULT NULL COMMENT '性别',  `phone` varchar(32) DEFAULT NULL COMMENT '电话',  `email` varchar(32) DEFAULT NULL COMMENT '邮箱',  `create\_time` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP COMMENT '创建时间',  `password` varchar(32) DEFAULT NULL COMMENT '密码',  PRIMARY KEY (`id`)  ) ENGINE=InnoDB DEFAULT CHARSET=utf8; |

第一种 使用sharding-jdbc-spring-boot-starter整合

Pom.xml

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  <modelVersion>4.0.0</modelVersion>  <artifactId>shard-jdbc-starter</artifactId>  <packaging>jar</packaging>  <name>shard-jdbc-starter</name>  <description>shard-jdbc-starter</description>  <groupId>com.sun</groupId>  <version>0.0.1-SNAPSHOT</version>    <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <java.version>1.8</java.version>  <spring\_boot.version>2.0.3.RELEASE</spring\_boot.version>  <sharding.jdbc.version>3.0.0</sharding.jdbc.version>  <mybatis.version>1.2.0</mybatis.version>  <druid.version>1.1.6</druid.version>  </properties>    <dependencies>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-web</artifactId>  </dependency>    <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-test</artifactId>  <scope>test</scope>  </dependency>    <!-- sharding jdbc依赖 -->    <dependency>  <groupId>io.shardingsphere</groupId>  <artifactId>sharding-jdbc-spring-boot-starter</artifactId>  <version>${sharding.jdbc.version}</version>  </dependency>    <dependency>  <groupId>io.shardingsphere</groupId>  <artifactId>sharding-jdbc-spring-namespace</artifactId>  <version>${sharding.jdbc.version}</version>  </dependency>    <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  </dependency>    <dependency>  <groupId>org.mybatis.spring.boot</groupId>  <artifactId>mybatis-spring-boot-starter</artifactId>  <version>${mybatis.version}</version>  </dependency>    <dependency>  <groupId>com.alibaba</groupId>  <artifactId>druid</artifactId>  <version>${druid.version}</version>  </dependency>  </dependencies>    <!-- 引入spring boot的依赖 -->  <dependencyManagement>  <dependencies>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-dependencies</artifactId>  <version>${spring\_boot.version}</version>  <type>pom</type>  <scope>import</scope>  </dependency>  </dependencies>  </dependencyManagement>    <!-- 添加spring-boot的maven插件 -->  <build>  <plugins>  <plugin>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-maven-plugin</artifactId>  </plugin>  </plugins>  </build>  </project> |

在application.yml中配置数据源

|  |
| --- |
| server:  port: 9901  spring:  application:  name: shard-jdbc-starter  mybatis:  mapper-locations: classpath:mybatis/mapper/\*.xml  type-aliases-package: com.sun.shard.bean  sharding:  jdbc:  datasource:  names: ds0,ds1  ds0:  type: com.alibaba.druid.pool.DruidDataSource  driver-class-name: com.mysql.jdbc.Driver  url: jdbc:mysql://localhost:3306/order1  username: root  password: root  ds1:  type: com.alibaba.druid.pool.DruidDataSource  driver-class-name: com.mysql.jdbc.Driver  url: jdbc:mysql://localhost:3306/order2  username: root  password: root  config:  sharding:  props:  sql.show: **true**  tables:  t\_user:  key-generator-column-name: id  actual-data-nodes: ds${0..1}.t\_user${0..1}  #根据city\_id进行分库  database-strategy:  inline:  sharding-column: city\_id  algorithm-expression: ds${city\_id % 2}  #根据sex进行分表  table-strategy:  inline:  shardingColumn: sex  algorithm-expression: t\_user${sex % 2}  t\_address:  key-generator-column-name: id  actual-data-nodes: ds${0..1}.t\_address  #分库  database-strategy:  inline:  shardingColumn: lit  algorithm-expression: ds${lit % 2} |

第二种  硬编码方式：

这种方式使用Java config的方式，数据源，分片策略都要编码，这种方式比较灵活，但是所有策略都要硬编码，不方便维护

Pom.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>  <modelVersion>4.0.0</modelVersion>  <artifactId>shard-jdbc-yb</artifactId>  <packaging>jar</packaging>  <name>shard-jdbc-yb</name>  <description>shard-jdbc-yb</description>  <groupId>com.sun</groupId>  <version>0.0.1-SNAPSHOT</version>    <properties>  <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  <java.version>1.8</java.version>  <spring\_boot.version>2.0.3.RELEASE</spring\_boot.version>  <sharding.jdbc.version>2.0.3</sharding.jdbc.version>  <mybatis.version>1.2.0</mybatis.version>  <druid.version>1.1.6</druid.version>  </properties>    <dependencies>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-web</artifactId>  </dependency>    <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-test</artifactId>  <scope>test</scope>  </dependency>    <!-- sharding jdbc -->  <dependency>  <groupId>io.shardingjdbc</groupId>  <artifactId>sharding-jdbc-core</artifactId>  <version>${sharding.jdbc.version}</version>  </dependency>    <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  </dependency>    <dependency>  <groupId>org.mybatis.spring.boot</groupId>  <artifactId>mybatis-spring-boot-starter</artifactId>  <version>${mybatis.version}</version>  </dependency>    <dependency>  <groupId>com.alibaba</groupId>  <artifactId>druid</artifactId>  <version>${druid.version}</version>  </dependency>    <dependency>  <groupId>commons-dbcp</groupId>  <artifactId>commons-dbcp</artifactId>  <version>1.4</version>  </dependency>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-configuration-processor</artifactId>  <optional>true</optional>  </dependency>      </dependencies>    <!-- 引入spring boot的依赖 -->  <dependencyManagement>  <dependencies>  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-dependencies</artifactId>  <version>${spring\_boot.version}</version>  <type>pom</type>  <scope>import</scope>  </dependency>  </dependencies>  </dependencyManagement>  <!-- 添加spring-boot的maven插件 -->  <build>  <plugins>  <plugin>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-maven-plugin</artifactId>  </plugin>  </plugins>  </build>    </project> |

Yml文件

|  |
| --- |
| server:  port: 9902  spring:  application:  name: shard-jdbc-yb  mybatis:  mapper-locations: classpath:mybatis/mapper/\*.xml  type-aliases-package: com.sun.shard.bean |
| jdbc.className1=com.mysql.jdbc.Driver  jdbc.url1=jdbc:mysql://127.0.0.1:3306/order1?characterEncoding=UTF-8  jdbc.user1=root  jdbc.password1=root  jdbc.className2=com.mysql.jdbc.Driver  jdbc.url2=jdbc:mysql://127.0.0.1:3306/order2?characterEncoding=UTF-8  jdbc.user2=root  jdbc.password2=root |