mybatis-plus知识总结

1.mybatis-plus和mybatis的关系

mybatis-plus是mybatis的升级，只做功能的增强，不影响mybatis的使用，简而言之，就是mybatis改怎么用就怎么用，语法相同，只需要掌握mybatis-plus的一些新功能

2.mybatis-plus提供的BaseMapper接口

- 自定义mapper接口实现

```java
//我们自己自定义的mapper接口只需要实现BaseMapper
//既可以使用BaseMapper中的方法
public interface UserMapper extends BaseMapper<User> {

}

```

- baseMapper中crud的关键字：select查询，insert增加，delete删除,update修改

3.mybatis-plus提供的service接口

- 接口实现

```java
//自定义的service只需要实现提供的IService接口
//即可调用方法
public interface UserService extends IService<User> {

}

```

- 实现类的实现

```java
//注意泛型参数
@Component
public class UserServiceImpl extends ServiceImpl<UserMapper, User>
implements UserService {

}

```

- 注意userService中的方法关键字：save 添加，get代表单值查询, list代表查多行,remove代表删除 update有id则视为修改，没id则视为增加

4.mybatis-plus保持原有的用法

- 自定义方法，写mapper.xml也可以使用

- 你的mybatis咋使用，你就咋用
- 注意：plus的环境搭建，日志配置，等请查看自己的学习手册，这里不做废话

5.mybatis-plus的原理

mybatis-plus是通过实体bean，获取到你的类名，属性等，通过这些反向操作数据库的字段，所以要遵循mybatis-plus的约定写响应的bean，映射响应的数据库字段。

6.mybatis-plus的常用注解

1. @TableName :这个注解是当表名与数据库字段不一致的时候，映射用的
2. @TableId

- 作用一: 就是这个字段不是id的时候，将字段变为主键字段
- 作用二：type: 当你需要主键自增的方式就去使用
- 作用三：value: id 和数据库字段不同的时候

当然如果闲一个一个配置麻烦还可以使用全局配置

YAML | 复制代码

```yaml
spring:
  datasource:
    driver-class-name: com.mysql.jdbc.Driver
    url: jdbc:mysql://localhost:3306/seckill?characterEncoding=utf-8
    username: root
    password: 1234
    type: com.zaxxer.hikari.HikariDataSource  #springboot最快的连接池，可以使用
#配置MyBatis日志
mybatis-plus:
  configuration:
    log-impl: org.apache.ibatis.logging.stdout.StdOutImpl
  mapper-locations: classpath*:/mapper/*Mapper.xml
  type-aliases-package: com.edu.pojo
  #全局配置
  global-config:
    db-config:
      table-prefix: s_  #表前缀
      id_type: auto   #主键自增  mybatis-plus中的主键回填可以直接返回
```

3. @TableField(value = "user_name")：用于设置普通字段是与数据库对应的关系
4. @TableLogic :代表逻辑删除字段，数据库的默认值必须为0，根据字段修改字段值为1，查询的时候默认不查字段等于1的记录，约定大于配置大于编码

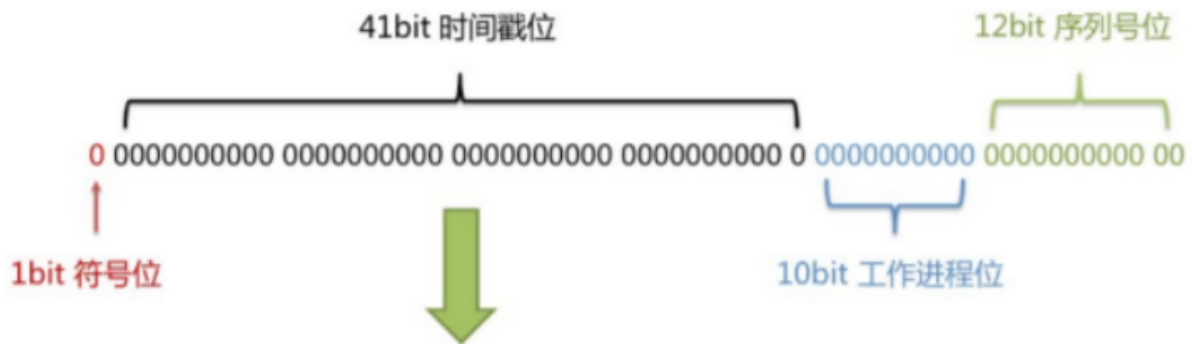
7.雪花算法

1. 雪花算法的概念：分布式环境下的主键生成算法，生成的主键不重复，而且能保证主键是递增的；
2. 雪花算法的背景：

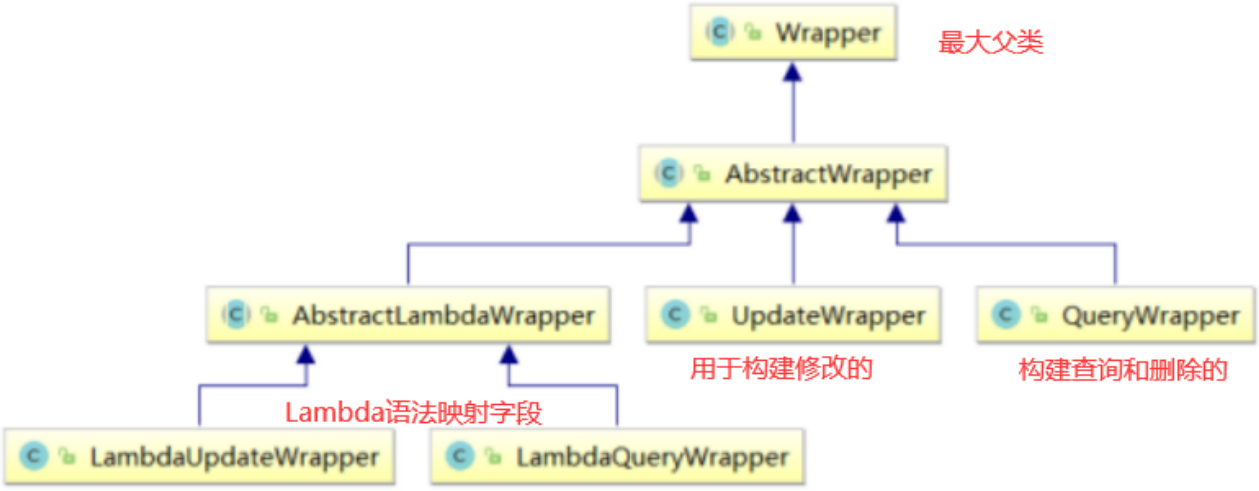
- 数据库的分库分表，当我们的数据库的数据达到一定量的时候，举个列子，数据库中的一页数据存储16kb，我们一条数据存储1kb,一页数据就是16条记录,如果数据库是一个两层的b+tree，第二层存的不在是数据而是冗余索引，主键大整型的，8字节，指针4字节，那么16kb大小能存储16kb/12字节==1770个页面,所以两层b+tree能存储1770*16，三层就是1770*1770*16=50,126,400多数据.
- 在这种数据存储情况下我们要进行分表，分库就是按照字段拆分，当然前提是业务拆分，拆分后必然有一个列的字段相同，用于多表查询,这叫垂直拆分。
- 水平拆分就是数据将记录存放到不同的表当中，可以根据多少条分组，也可以hash分组，缺点就是当我们要增加表的时候，相当于全部重新运算。

注意，分布式情况下生成主键的最好方法就是雪花算法

3. 雪花算法的原理：是一个long型数据的位图结构64bit位 8字节 8bit，最高位用于标识符号位，下面41位表示时间戳，10表示工作进程位，12位标识序号位。



8.mybatis-plus的crud的wrapper玩法



1. Wrapper： 条件构造抽象类，最顶端父类

2. AbstractWrapper： 用于查询条件封装，生成 sql 的 where 条件

- QueryWrapper： 查询条件封装
- UpdateWrapper： Update 条件封装

3. AbstractLambdaWrapper： 使用Lambda 语法

- LambdaQueryWrapper： 用于Lambda语法使用的查询Wrapper
- LambdaUpdateWrapper： Lambda 更新封装Wrapper

8.1基础用法

Java | 复制代码

```java
//----------------------------querywrapper用法-------------------------------
//////查询用户名包含a，年龄在20到30之间，并且邮箱不为null的用户信息
QueryWrapper<User> queryWrapper = new QueryWrapper<>();
queryWrapper.like("user_name", "a")
    .between("age", 20, 30)
    .isNotNull("email");
List<User> userList = userMapper.selectList(queryWrapper);
//----------------------------updatewrapper第一种基础用法-------------------------------
//1.构造修改条件
//将（年龄大于20并且用户名中包含有a）或邮箱为null的用户信息修改
UpdateWrapper<User> updateWrapper = new UpdateWrapper<>();
updateWrapper.gt("age", 20)
                .like("user_name", "a")
                .or()
                .isNull("email");
//2.修改的值
//将用户名中包含有a并且（年龄大于20或邮箱为null）的用户信息修改
User user = new User();
user.setName("guiGeGe");
user.setEmail("201996084006@QQ.com");
System.out.println(userMapper.update(user, updateWrapper));
//----------------------------updatewrapper第二种基础用法-------------------------------
//将用户名中包含有a并且（年龄大于20或邮箱为null）的用户信息修改
UpdateWrapper<User> updateWrapper = new UpdateWrapper<>();
updateWrapper.like("user_name", "a")
                .and(wrapper -> wrapper.gt("age",
20).or().isNull("email"));
updateWrapper.set("user_name", "java开发")
                .set("email", "2033443140@qq.com");
userMapper.update(null, updateWrapper);
```

8.2子查询和查询指定字段

Java 复制代码

```
        //子查询
        ////查询id小于等于10的用户信息
        QueryWrapper<User> queryWrapper = new QueryWrapper<>();
        queryWrapper.inSql("uid", "select uid from s_user where uid <=
10");
        userMapper.selectList(queryWrapper);//注意这里可以是其他mapper
        //查询指定字段
        //查询用户信息的username和age和email字段
        QueryWrapper<User> queryWrapper = new QueryWrapper<>();
        queryWrapper.select("user_name", "age", "email");
```

8.3条件优先级和字段判空操作

Java 复制代码

```
//条件优先级
//将用户名中包含有a并且（年龄大于20或邮箱为null）的用户信息修改
UpdateWrapper<User> updateWrapper = new UpdateWrapper<>();
updateWrapper.like("user_name", "a")
             .and(wrapper -> wrapper.gt("age",
20).or().isNull("email"));
//字段判空
// isNotBlank代表  "" 和 null
// age!=null只有这个
UpdateWrapper<User> updateWrapper = new UpdateWrapper<>();
updateWrapper.like(StringUtils.isNotBlank(username), "user_name",
username)
            .and(wrapper -> wrapper.gt(age != null, "age",
age).or().isNull("email"));
updateWrapper.set(StringUtils.isNotBlank(email), "email", email);
```

8.4lambda做字段映射

```java
//("user_name", username)------------------>User::getName无需关注数据库字段了
LambdaQueryWrapper<User> lambdaQueryWrapper = new LambdaQueryWrapper<>();
lambdaQueryWrapper.like(StringUtils.isNotBlank(username),User::getName,username)
                  .ge(User::getAge,ageStart)
                  .lt(User::getAge,ageEnd);
//Function<User, String> function =(user)-> user.getName();
```

8.5mybatis-plus的分页插件

1. 使用自带的分页插件

- 添加配置类--分页插件拦截器

```java
    @Bean
    public MybatisPlusInterceptor mybatisPlusInterceptor() {
        MybatisPlusInterceptor interceptor = new MybatisPlusInterceptor();
        //分页的拦截器插件
        interceptor.addInnerInterceptor(new PaginationInnerInterceptor(DbType.MYSQL));
        //开启乐观锁的插件
        interceptor.addInnerInterceptor(new OptimisticLockerInnerInterceptor());
        return interceptor;
    }
```

- 使用基本分页

```java
Page<User> page = new Page<>(1, 3);
userMapper.selectPageLimit(page, 20);
//page中有上一页,下一页，总条数，总页数等等
```

2. 自定义分页

- 接口定义

必须使用page<User>

Java | 复制代码

```java
//按照自定义的条件分页
Page<User> selectPageLimit(@Param("page") Page<User> page,@Param("age") Integer age);
//调用自定义分页
Page<User> page = new Page<>(1, 3);
userMapper.selectPageLimit(page, 20);
```

- mapper.xml定义

XML | 复制代码

```xml
//注意用mybatis的xml时，mybatis-plus的字段映射是没效果的，需要我们按照mybatis的配置做
<sql id="BaseSql">uid as id,user_name as name, age,email </sql>
<select id="selectPageLimit" resultType="user">
    select <include refid="BaseSql"></include> from s_user where age >= #{age}
</select>
```

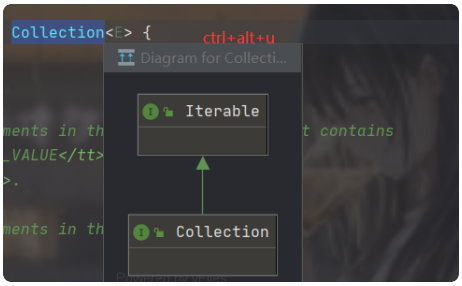
9.科普小知识

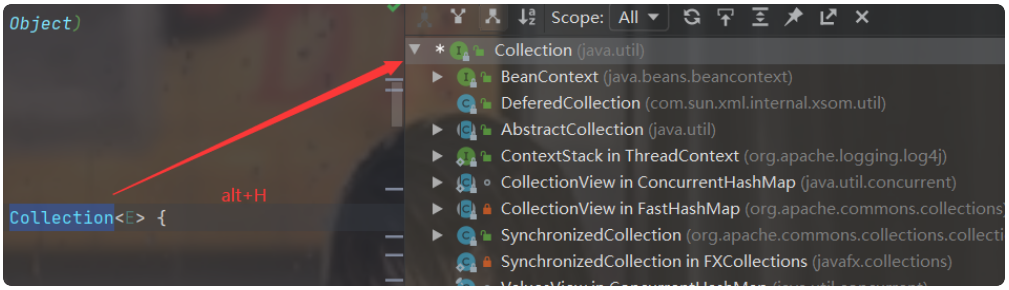
1. 函数含义

Java | 复制代码

```
lt: less than 小于
le: less than or equal to 小于等于
eq: equal to 等于
ne: not equal to 不等于
ge: greater than or equal to 大于等于
gt: greater than 大于
```

2. idea快捷键





10.悲观锁和乐观锁

- 悲观锁的概念：阻塞加锁，1.5jdksychronized的加锁方式----效率很高，最后锁升级后效率还不错
- 乐观锁：mybatis的乐观锁实现理念，我自己感觉是源于cas锁机制，比较在交换，当一个线程来读数据，复制一份原来的数据：准备修改时，会先去将原来copy的数据与现在的数据做对比，如果相同，ok更改，不同则认为数据被改过，不做任何操作

mybatis-plus的乐观锁原理，就是先去数据库查询一份原来的数据，这个数据中有一个verison字段版本号信息，在修改的时候，同样去先将原来的数据与修改的version做对比，相同则修改，并且verison+1，不同不做操作返回受影响行数0；

- 原理代码

数据库中添加version字段 SQL 复制代码

```sql
# 取出记录时，获取当前version
SELECT id,`name`,price,`version` FROM product WHERE id=1

# 更新时，version + 1，如果where语句中的version版本不对，则更新失败
UPDATE product SET price=price+50, `version`=`version` + 1 WHERE id=1
AND`version`=1
```

- 乐观锁实现插件

实现步骤 Java 复制代码

```java
//1.实体类标识版本字段
@Version
private Integer version;
//配置插件
interceptor.addInnerInterceptor(new OptimisticLockerInnerInterceptor());
```

11.mybatis-plus支持的枚举添加

- 实现步骤

好好掌握枚举 Java 复制代码

```java
//1.实体属性----使用枚举类型
private SexEmen sex;
//2.定义枚举类
@Getter
public enum SexEmen {
    MEN(1, "男"),
    WOMEN(0, "女");
    @EnumValue
    private Integer sex;
    private String sexName;

    SexEmen(Integer sex, String sexName) {
        this.sex = sex;
        this.sexName = sexName;
    }
}
//3.调用方式
 User user=new User(null,"guihaole",18,
SexEmen.MEN,"20168893699@qq.com",0);
```

12.多数据源和代码生成器

- 资料官网 :https://baomidou.com/pages/a61e1b/#%E6%96%87%E6%A1%A3-documentation
- 配置学习视频地址:https://www.bilibili.com/video/BV12R4y157Be?p=53
- 文档看,我的日积月累下的:mybatis-plus下的mybatis-plus的开发手册

13.mybatisX插件快速生成crud

- 安装插件
- 逆向工程
- 快速生成crud

教学视频地址：https://www.bilibili.com/video/BV12R4y157Be?p=57&spm_id_from=pageDriver

文档参考地址：https://baomidou.com/pages/ba5b24/