一、分页插件

MyBatis Plus自带分页插件，只要简单的配置即可实现分页功能

1、添加配置类

创建config包，创建MybatisPlusConfig类

```
package com.atguigu.mybatisplus.config;

@Configuration
@MapperScan("com.atguigu.mybatisplus.mapper")  //可以将主类中的注解移到此处
public class MybatisPlusConfig {

}
```

2、添加分页插件

配置类中添加@Bean配置

```
@Bean
public MybatisPlusInterceptor mybatisPlusInterceptor() {
    MybatisPlusInterceptor interceptor = new MybatisPlusInterceptor();
    interceptor.addInnerInterceptor(new PaginationInnerInterceptor(DbType.MYS
    return interceptor;
}
```

3、测试分页

创建类InterceptorTests

```

```

```java
package com.atguigu.mybatisplus;

@SpringBootTest
public class InterceptorTests {

    @Resource
    private UserMapper userMapper;

    @Test
    public void testSelectPage(){

        //创建分页参数
        Page<User> pageParam = new Page<>(1,5);
        //执行分页查询
        userMapper.selectPage(pageParam, null);
        //查看分页参数的成员
        System.out.println(pageParam);
    }
}

```

二、XML自定义分页

1、UserMapper中定义接口方法

```java
/**
     * 查询 : 根据年龄查询用户列表，分页显示
     *
     * @param page 分页对象,xml中可以从里面进行取值,传递参数 Page 即自动分页,必须
     * @param age 年龄
     * @return 分页对象
     */
IPage<User> selectPageByPage(Page<?> page, Integer age);
```

2、定义XML

```xml
<select id="selectPageByPage" resultType="com.atguigu.mybatisplus.entity.User
    SELECT <include refid="Base_Column_List"/> FROM user WHERE age > #{age}
</select>
```

3、测试

```
@Test
public void testSelectPageVo(){
    Page<User> pageParam = new Page<>(1,5);
    userMapper.selectPageByPage(pageParam, 18);
    List<User> users = pageParam.getRecords();
    users.forEach(System.out::println);
}
```

三、乐观锁

1、场景

一件商品，成本价是80元，售价是100元。老板先是通知小李，说你去把商品价格增加50元。小李正在玩游戏，耽搁了一个小时。正好一个小时后，老板觉得商品价格增加到150元，价格太高，可能会影响销量。又通知小王，你把商品价格降低30元。

此时，小李和小王同时操作商品后台系统。小李操作的时候，系统先取出商品价格100元；小王也在操作，取出的商品价格也是100元。小李将价格加了50元，并将100+50=150元存入了数据库；小王将商品减了30元，并将100-30=70元存入了数据库。是的，如果没有锁，小李的操作就完全被小王的覆盖了。

现在商品价格是70元，比成本价低10元。几分钟后，这个商品很快出售了1千多件商品，老板亏1万多。

接下来将我们演示这一过程：

step1：数据库中增加商品表

```
CREATE TABLE product
```

```
(
    id BIGINT(20) NOT NULL AUTO_INCREMENT COMMENT '主键ID',
    name VARCHAR(30) NULL DEFAULT NULL COMMENT '商品名称',
    price INT(11) DEFAULT 0 COMMENT '价格',
    version INT(11) DEFAULT 0 COMMENT '乐观锁版本号',
    PRIMARY KEY (id)
);

INSERT INTO product (id, NAME, price) VALUES (1, '笔记本', 100);
```

step2：创建实体类

```
package com.atguigu.mybatisplus.entity;
@Data
public class Product {
    private Long id;
    private String name;
    private Integer price;
    private Integer version;
}
```

step3：创建Mapper

```
package com.atguigu.mybatisplus.mapper;

public interface ProductMapper extends BaseMapper<Product> {

}
```

step4：测试

```
@Resource
private ProductMapper productMapper;

@Test
public void testConcurrentUpdate() {

    //1、小李
```

```
    Product p1 = productMapper.selectById(1L);

    //2、小王
    Product p2 = productMapper.selectById(1L);

    //3、小李将价格加了50元，存入了数据库
    p1.setPrice(p1.getPrice() + 50);
    int result1 = productMapper.updateById(p1);
    System.out.println("小李修改结果：" + result1);

    //4、小王将商品减了30元，存入了数据库
    p2.setPrice(p2.getPrice() - 30);
    int result2 = productMapper.updateById(p2);
    System.out.println("小王修改结果：" + result2);

    //最后的结果
    Product p3 = productMapper.selectById(1L);
    System.out.println("最后的结果：" + p3.getPrice());
}
```

2、乐观锁方案

数据库中添加version字段：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 `ve
```

接下来介绍如何在Mybatis-Plus项目中，使用乐观锁：

3、乐观锁实现流程

step1：修改实体类

添加 @Version 注解

```
@Version
```

```
2 private Integer version;
```

step2：添加乐观锁插件

```
1 interceptor.addInnerInterceptor(new OptimisticLockerInnerInterceptor());//乐观
```

step3：重新执行测试

小王的修改失败!

4、优化流程

失败后重试

```
1 if(result2 == 0){//更新失败，重试
2     System.out.println("小王重试");
3     //重新获取数据
4     p2 = productMapper.selectById(1L);
5     //更新
6     p2.setPrice(p2.getPrice() - 30);
7     productMapper.updateById(p2);
8 }
```