**1什么是 MyBatis-Plus？**

MyBatis-Plus 是 MyBatis 的一个增强工具库，简化了 MyBatis 的开发，提供了很多实用的增强功能，如代码生成器、分页插件、条件构造器等。

**2 MyBatis-Plus 的优点有哪些？**

简化开发：提供了很多实用的增强功能，减少了开发人员的工作量。

内置 CRUD 方法：通过继承 BaseMapper 接口，可以直接使用 MyBatis-Plus 提供的 CRUD 方法。

支持代码生成器：可以根据数据库表自动生成 Entity 类、Mapper 接口以及 Service 层代码，提高开发效率。

内置分页插件：集成了分页功能，支持多种数据库的分页查询。

**3如何在 Spring Boot 项目中集成 MyBatis-Plus？**

在 Spring Boot 项目中集成 MyBatis-Plus 需要添加依赖，配置数据源，创建实体类和 Mapper 接口，并配置 MyBatis-Plus 的配置类。

**4 MyBatis-Plus 的主要组件有哪些？**

BaseMapper：提供了常用的增删改查方法。

条件构造器：通过 QueryWrapper 可以方便地构建查询条件。

分页插件：支持多种数据库的分页查询。

代码生成器：根据数据库表自动生成代码。

**5 如何使用 MyBatis-Plus 的条件构造器进行复杂查询？**

使用 QueryWrapper 可以进行复杂的条件查询，支持链式调用，例如：

QueryWrapper<User> queryWrapper = new QueryWrapper<>();

queryWrapper.like("name", "张").ge("age", 18);

List<User> userList = userMapper.selectList(queryWrapper);

**6 MyBatis-Plus 中的乐观锁如何使用？**

MyBatis-Plus 提供了 @Version 注解来支持乐观锁，用于控制并发访问冲突，示例：

@Version

private Integer version;

**7 MyBatis-Plus 的分页插件如何配置和使用？**

配置分页插件可以通过在配置类中注入 PaginationInterceptor 或者在配置文件中配置 MyBatisPlusInterceptor，使用方法：

Page<User> page = new Page<>(1, 10);

IPage<User> userPage = userMapper.selectPage(page, null);

**以下是 MyBatis-Plus 中 BaseMapper 提供的常见 CRUD 方法，简洁描述如下：**

1. 插入操作

insert(T entity)：插入一条记录，返回受影响行数。

userMapper.insert(user);

2. 删除操作

deleteById(Serializable id)：根据主键删除记录。

userMapper.deleteById(1L);

deleteByMap(Map<String, Object> columnMap)：根据条件删除。

Map<String, Object> map = new HashMap<>();

map.put("name", "Tom");

userMapper.deleteByMap(map);

delete(Wrapper<T> wrapper)：使用条件构造器删除。

QueryWrapper<User> wrapper = new QueryWrapper<>();

wrapper.eq("age", 25);

userMapper.delete(wrapper);

3. 更新操作

updateById(T entity)：根据主键更新记录

user.setAge(30);

userMapper.updateById(user);

4. 查询操作

selectById(Serializable id)：根据主键查询记录。

User user = userMapper.selectById(1L);

**QueryWrapper 是 MyBatis-Plus 提供的条件查询构造器，用于构造查询条件。它允许使用链式方法动态地拼接 SQL 查询语句。常用方法：**

eq(String column, Object val)：等于条件。

示例：wrapper.eq("name", "Tom") 表示 name = 'Tom'。

ne(String column, Object val)：不等于条件。

示例：wrapper.ne("age", 25) 表示 age != 25。

gt(String column, Object val)：大于条件。

示例：wrapper.gt("age", 18) 表示 age > 18。

ge(String column, Object val)：大于等于条件。

示例：wrapper.ge("age", 18) 表示 age >= 18。

lt(String column, Object val)：小于条件。

示例：wrapper.lt("age", 30) 表示 age < 30。

le(String column, Object val)：小于等于条件。

示例：wrapper.le("age", 30) 表示 age <= 30。

between(String column, Object val1, Object val2)：区间查询。

示例：wrapper.between("age", 18, 30) 表示 age 在 18 到 30 之间。

like(String column, Object val)：模糊查询。

示例：wrapper.like("name", "Tom") 表示 name 包含 'Tom'。

orderByAsc(String... columns)：升序排序。

示例：wrapper.orderByAsc("age") 表示按 age 升序排序。

MyBatis-Plus 提供了 IPage 接口和 Page 实现类来封装分页信息。分页查询需要通过 selectPage 方法来执行。

// 创建分页对象，参数为：当前页、每页记录数

Page<User> page = new Page<>(1, 10); // 第1页，每页显示10条记录

// 查询条件构造器

QueryWrapper<User> wrapper = new QueryWrapper<>();

wrapper.ge("age", 18); // 年龄大于等于18

// 分页查询

IPage<User> userPage = userMapper.selectPage(page, wrapper);