

第六篇 主键生成策略

Mybatis Plus 为我们提供了三种设置 主键生成策略的方式。

它们的优先级顺序是：局部注解 > 全局 > 默认(雪花算法)。

一、默认主键生成策略：雪花算法

Mybatis Plus如果不做任何主键策略配置，默认使用的是雪花算法。

该策略会根据雪花算法生成主键ID，主键类型为Long或String（具体到MySQL数据库就是BIGINT和VARCHAR），该策略使用接口IdentifierGenerator的方法nextId（默认实现类为DefaultIdentifierGenerator雪花算法）

snowflake算法（雪花算法）是Twitter开源的分布式ID生成算法，结果是一个long类型的ID。

其核心思想：使用41bit作为毫秒数，10bit作为机器的ID（5bit数据中心，5bit的机器ID），12bit作为毫秒内的流水号（意味着每个节点在每个毫秒可以产生4096个ID），最后还有一个符号位，永远是0。

二、自定义主键策略

mybatis-plus3.3.0以后，主要有五种主键生成策略。

```
public enum IdType {  
    /**  
     * 数据库ID自增,数据库需要支持主键自增(如MySQL), 并设置主键自增  
     */  
    AUTO(0),  
  
    /**  
     * 该类型为未设置主键类型,默认使用雪花算法生成  
     */  
}
```

```

    */
    NONE(1),

    /**
     * 用户输入ID,数据类型和数据库保持一致就行
     * <p>该类型可以通过自己注册自动填充插件进行填充</p>
     */
    INPUT(2),

    /** 以下3种类型、只有当插入对象ID 为空,才自动填充。 */
    /**
     * 全局唯一ID (idWorker),数值类型 数据库中也是必须是数值类型
     否则会报错
     */
    ID_WORKER(3),
    /**
     * 全局唯一ID (UUID, 不含中划线)
     */
    UUID(4),
    /**
     * 字符串全局唯一ID (idWorker 的字符串表示), 数据库也要保证一
     样字符类型
     */
    ID_WORKER_STR(5);
}

```

三、局部注解配置策略

我们针对主键设置主键策略使用注解方式为

```

@TableId(type = IdType.AUTO)
private Long userId;

```

注意要将数据库的user表主键也设置成自增。



四、全局配置策略

在application.yml的mybatis-plus节点添加配置：

```
mybatis-plus:
  global-config:
    db-config:
      id-type: auto
```

然后自行去[测试一下添加用户](#)，看自增策略的生效情况。

五、扩展使用（看一下即可）

5.1.Input用户输入ID策略的用法

其中需要特殊介绍的是：Input（用户输入ID），这个ID来源可以有两种

- 用户自己设置ID，并在insert之前SET主键的值
- 一些有序列的数据库，比如Oracle，SQLServer等，针对这些数据库我们可以通过序列填充ID字段

Mybatis-Plus 内置了如下数据库主键序列（如果内置支持不满足你的需求，可实现 IKeyGenerator 接口来进行扩展）：

- DB2KeyGenerator
- H2KeyGenerator
- KingbaseKeyGenerator
- OracleKeyGenerator
- PostgreKeyGenerator

以Oracle 的Sequence使用方法为例，使用方法如下：
首先添加@Bean

```
@Bean
public OracleKeyGenerator oracleKeyGenerator(){
    return new OracleKeyGenerator();
}
```

然后实体类配置主键 Sequence，指定主键策略为 IdType.INPUT 即可，

```
@Data
@KeySequence(value = "SEQ_USER" , clazz = Long.class)
public class User {

    @TableId(value = "ID",type = IdType.INPUT)
    private Integer id;
```