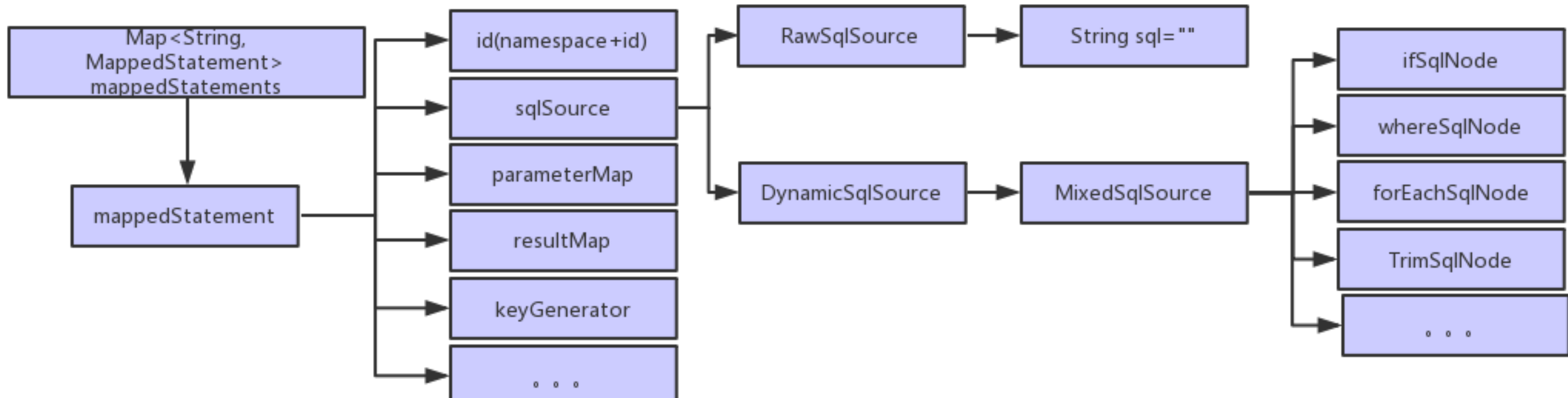


## 一、配置解析



### Configuration

InterceptorChain interceptorChain  
解析并实例化插件然后保存到InterceptorChain中

HashSet<String> loadedResources  
注册xml文件，文件夹+文件名比如：  
sqlmapper/UserMapper.xml表示对应的xml已经解析  
并且存储接口注册信息，以namespace+接口全限定名  
比如：namespace.com.cradymapper.UserMapper

Map<String, ResultMap> resultMap  
保存<sql>节点，ID为namespace+\"+id

Map<String, XNode> sqlFragments  
保存<sql>节点，ID为namespace+\"+id

### MapperRegistry mapperRegistry

Map<Class<?>, MapperProxyFactory<?>> knownMappers  
存储mapper工厂对象，key为interface+mapper接口全限定名，比如：  
interface com.cradymapper.UserMapper，value为mapper工厂对象。

MapperProxyFactory  
主要用来创建mapper接口的实际代理调用sql语  
句时实际调用的是这个代理对象

### Map<String, MappedStatement> mappedStatements

#### MappedStatement

id(namespace+\"+id)

#### SqlSource

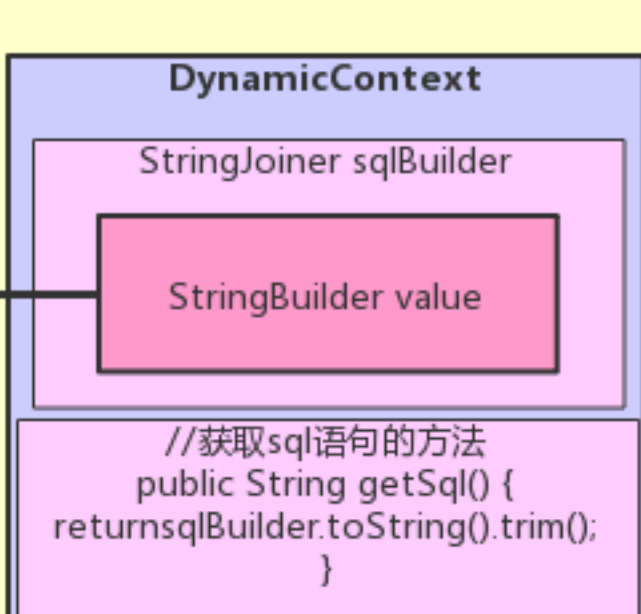
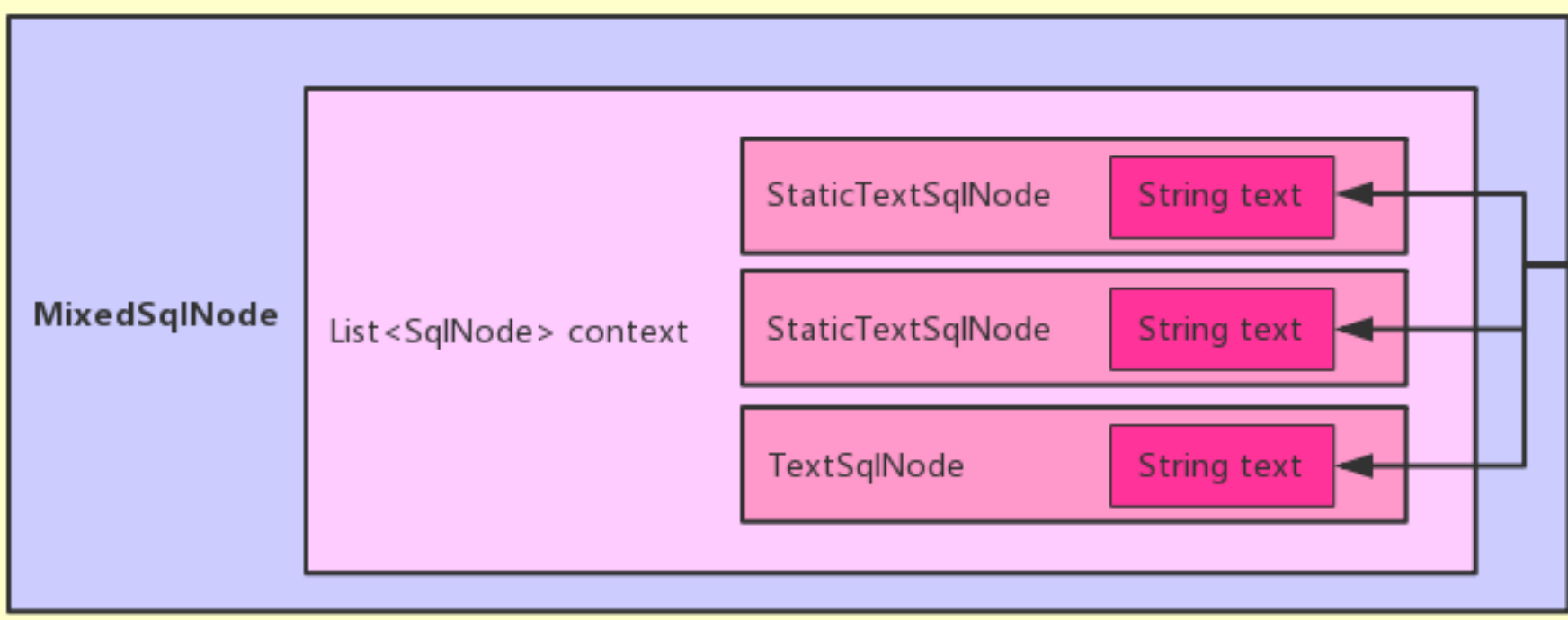
sql语句中是否包含#{}

DynamicSqlSource

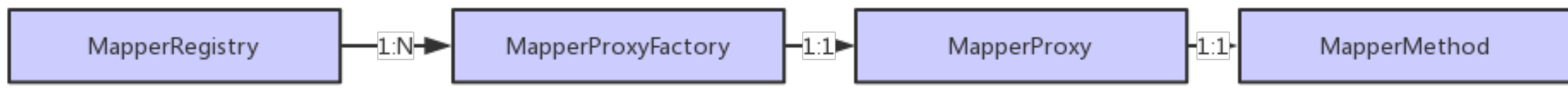
getBoundSql()  
获取(sql语句#{})都已经解析成?  
{}都需要替换成实际参数

RawSqlSource

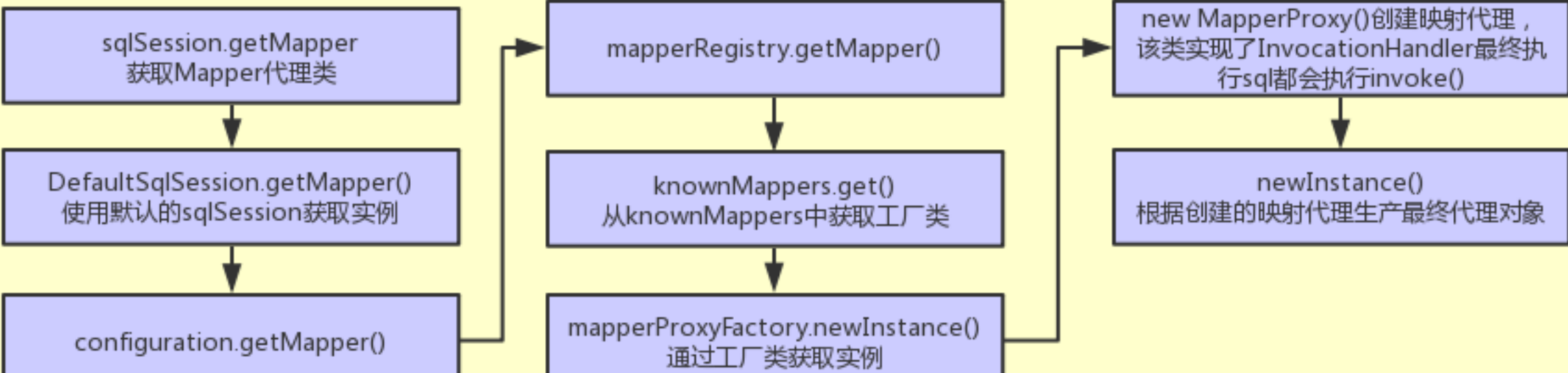
getBoundSql()  
获取(sql语句#{})都已经解析成?



## 二、创建代理



### 获取mapper代理对象

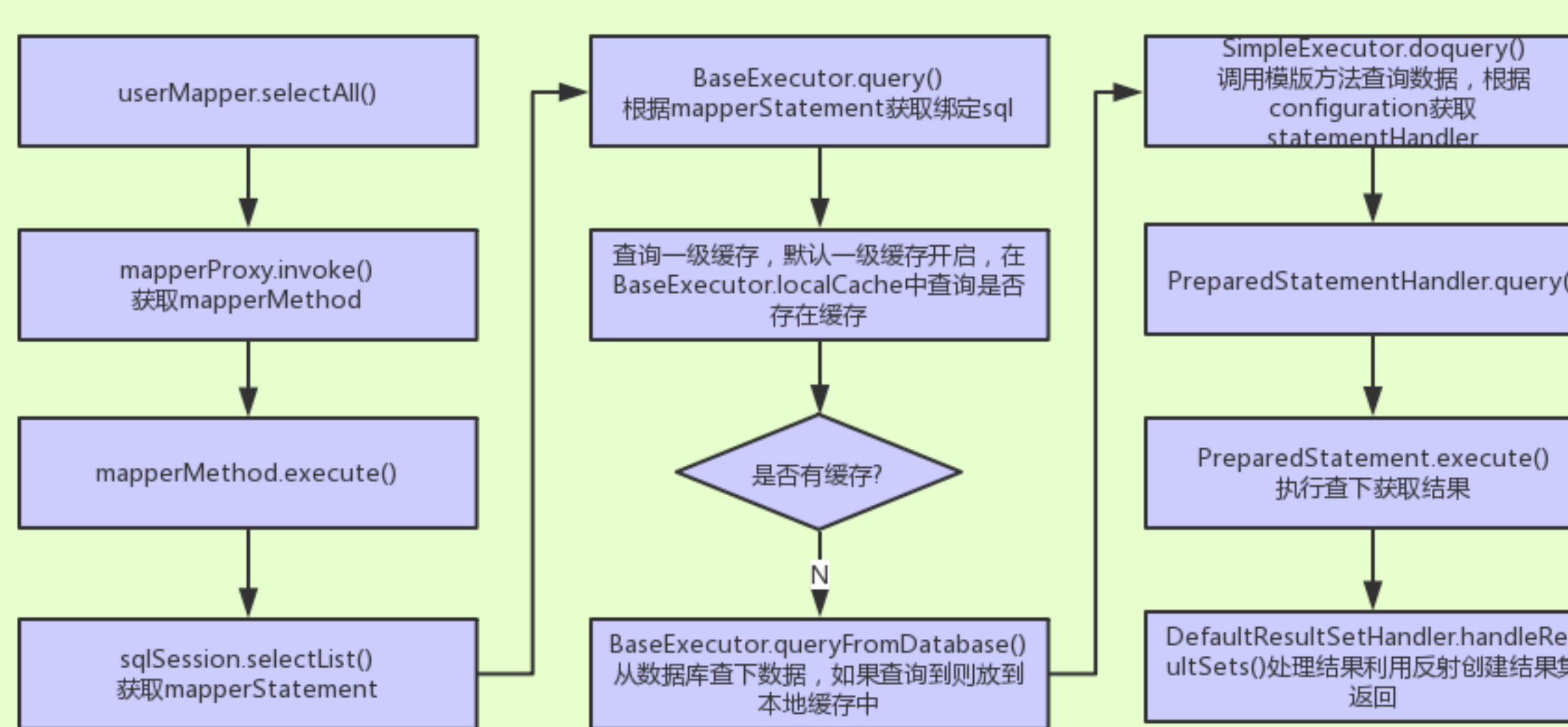


## 三、执行过程

### 总流程

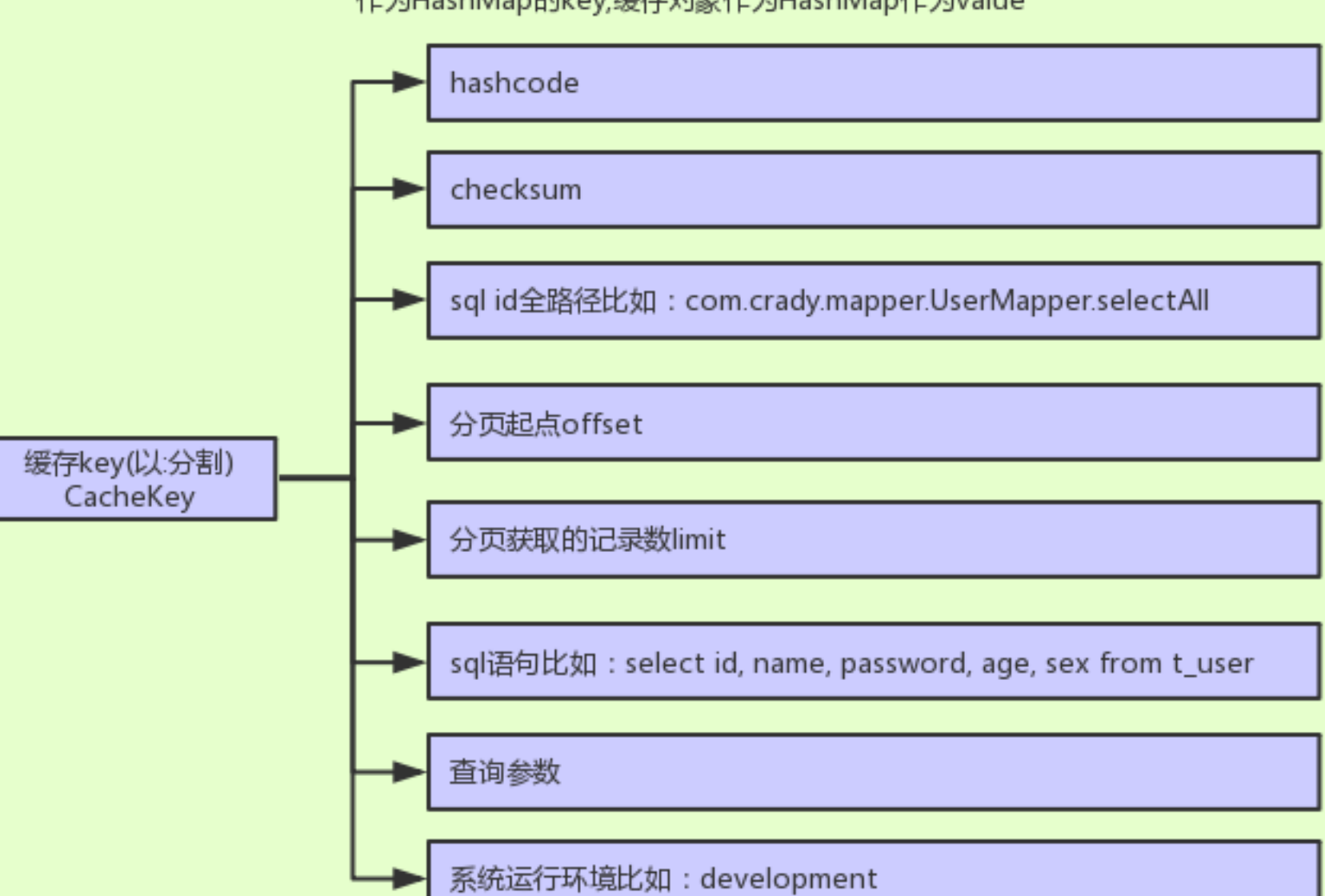


### 详细流程



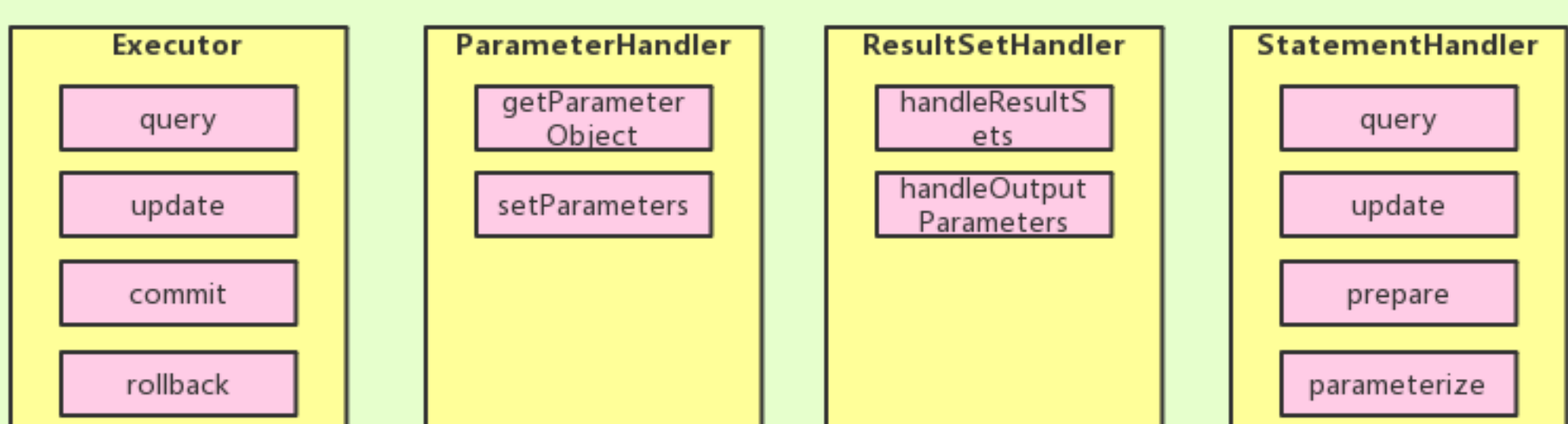
### 一级缓存key

一级缓存存储在BaseExecutor对象中的localCache属性中,localCache的实现类是perpetualCache,其底层是用HashMap存储缓存对象,CacheKey对象作为HashMap的key,缓存对象作为HashMap作为value

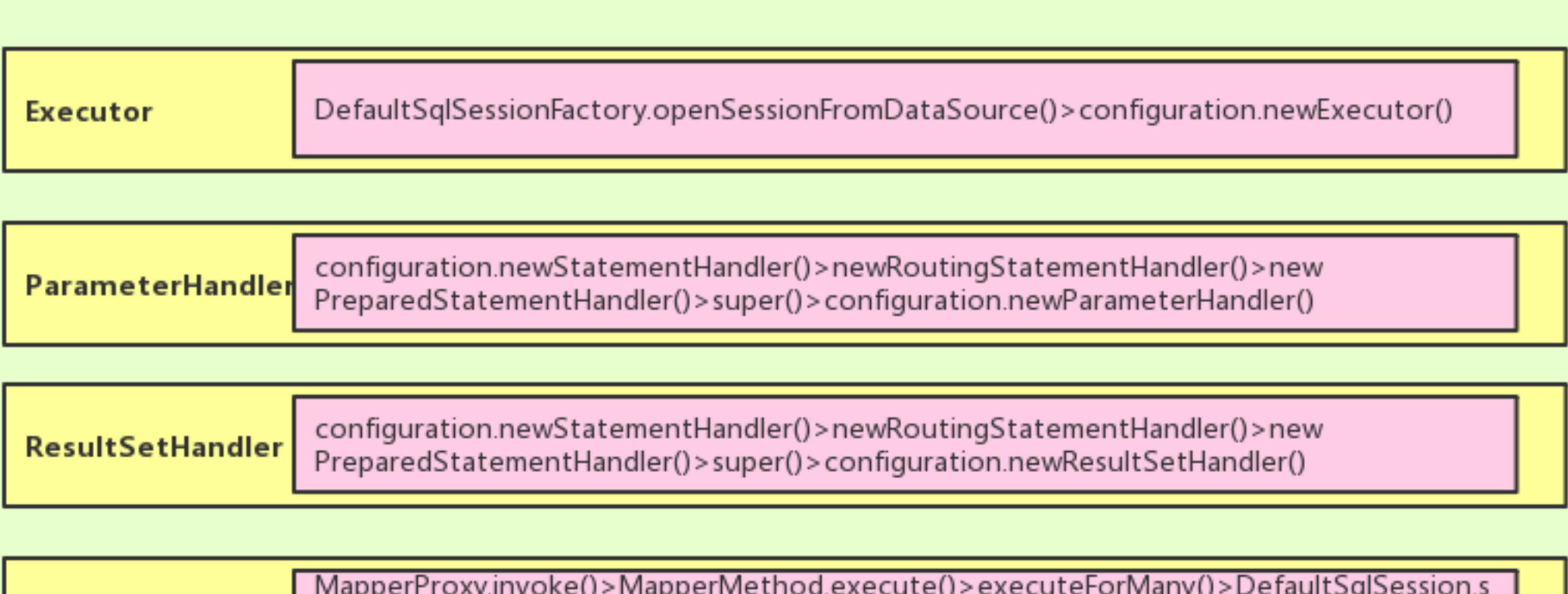


完整例  
子-1524961915.4209224624:com.cradymapper.UserMapper.selectAll:0:2147483647:select

## Mybatis拦截器拦截点



## Mybatis拦截器作用点



## Mybatis拦截器详细流程 (StatementHandler为例)

