LanguageDriver默认实现 SqlSessionFactory XMLScriptBuilder XMLConfigBuilder XMLMapperBuilder XMLStatementBuilder XMLLanguageDriver Builder 创建SqlSessionFactoryBuild(配置文件)的过程就是初始化过程 New 输入是配置文件 XMLConfigBuilder parse parseCon<mark>f</mark>iguration 解析配置文件中的各元素 1解析sett<mark>i</mark>ng放到configuration中 2解析prop<mark>e</mark>rties放到configuration中 还有XPat<mark>h</mark>Parser中,用于替换\${} 3解析type<mark>A</mark>liases 4解析Plug<mark>i</mark>ns 5解析obje<mark>c</mark>tFactory 6解析obje<mark>c</mark>tWrapperFactory 7解析refl<mark>e</mark>ctionFactory 8解析envi<mark>r</mark>onment 9解析data<mark>b</mark>aseIdProvider 10解析typ<mark>e</mark>Handlers 11解析mapper文件 parse 解析入口mapper节点: 1解析Namespace/cacheref/cach<mark>e</mark>属性 2解析resultMap 3解析sql节点 parseStatementNode 4解析select update节点 解析入口<mark>se</mark>lect|update 1解析id/d<mark>a</mark>tabaseId等参数 2sq1节点<mark>替</mark>换 3解析sele<mark>ct</mark>Key createSqlSource 4解析sele<mark>ct</mark>创建SqlSource parseScriptNode 解析各个子节点创建 List<Sq<mark>lN</mark>ode>,根据 SqlNode创建SqlSource 5解析useG<mark>en</mark>eratedKeys 6根据以上信息创建MappedStatement加入 到configuration MapperRegistry configuration addMapper knownMappers.put(type, new MapperProxyFactory<T>(type)) bindMappe<mark>r</mark>ForNamespace Configuration创建完成,作为构造参数创建 new DefaultSqlSessionFactory(config); Sql执行流程 分3步 DefaultSqlSessionF configuration __openSession openSess<mark>io</mark>nFromDataSource 1从confi<mark>g</mark>中获取Environment 2从Environment中获得TransactionFactory 3使用transactionFactory创的Transaction new 4根据Tra<mark>ns</mark>action创建Executor— 1根据E<mark>xe</mark>cutorType选择创建Executor 的具体<mark>类</mark>型,根据cacheEnabled 选择是<mark>否</mark>使用CachingExecutor 二级缓存装饰 2inter<mark>ce</mark>ptorChain插件拦截 5根据executor创建DefaultSqlSession并返回 MapperProxyFactory DefaultSqlSession mapperRegistry MapperProxy configuration getMapper(interface) getMapper getMapper MapperProxyFactory < T > mapperProxyFactory = (MapperProxyFactory<T>) knownMappers.get(type); mapperProxyFactory.newInstance(sqlSession); 返回inerface的动态代理 MapperProxy是InvocationHandler的子类,利用动态代理_ 创建期望接口的代理类 上面获取的动态代理的接口实例,执行方法时回走 MapperProxy这个InvocationHandler中的逻辑,执行 真正的方法 sqlSource以 SqlSourceBuilder DynamicSqlSource MapperMethod DefaultSqlSession MappedStatement MapperProxy CachingExecutor BaseExecutor Invoke

