1.mybatis常用的标签

insert ,delete,update,select,

resultmap,parametermap,sql,include,selectKey

动态sql:(9)

trim,where,when,if,otherwise,foreach,bind,choose,set

2.${}propertis配置文件的配置属性#{} mybatissql的占位符

3.mybatis的分页

mybatis使用rowbounds对象分页,针对resultset的内存分页

使用方言,执行物理分页

4.mapper接口的原理是jdk动态代理

延迟加载的原理是cglib动态代理

5.mybatis的一级缓存和二级缓存(内部存储缓存和查询缓存)

mybatis首先去缓存中查询结果集,如果没有则查询查询互数据库,如果有则从缓存中取出数据集,mybatis内部查询缓存使用一个hashmap,key为hashcode+sqlId+sql语句,value是查询出来映射生成的java对象;

二级缓存即查询缓存,他的作用域是一个mapper的namespace,在同一个namespace中查询sql可以从缓存中获取数据,二级缓存是可以跨sqlSession.

6.mapper编写的三种方式.

1).接口实现类继承sqlsessiondaosupport

需要编写mapper接口,mapper接口实现类,mapper.xml

在sqlMapConfig.xml中,配置mapper.xml的位置

定义mapper接口

实现类继承sqlsessiondaosupport,通过this.getSqlsession进行数据增删改查

spring配置

2).使用mapperfactorybean(在配置配置文件中配置,)

在sqlMapConfig.xml中,配置mapper.xml的位置

如果.xml和接口的名称相同并且在同一个目录,就不用配置.

定义mapper接口,namespace是mapper接口的地址,

3).使用mapper扫描器

mapperscannerconfigurer

使用扫描器后,扫描器将接口通过代理方法生成实现对象,要spring自动注册,名称为mapper接口的名称.

7.mybatis架构的对象

configuration:配置信息都保存在这个类中.

sqlSession:主要api,表示和数据库交互的会话,完成对数据库的操作

executor:mybatis的执行器,sql的生成,查询缓存的维护.

statementhandler:封装了jdbcStatement操作,负责对JDBC statement的操作,设置参数

parameterhandler:负责对用户传递的餐护士转换成jdbc statement所对应的数据类型,

resultsethandler:负责将返回的resultset结果转换成list类型的集合,

typehangdler:java数据类型和jdbc类型之间的映射和转换

mappedstatement:维护sql节点的封装

sqlsource:负责根据用户传递的参数对象,动态生成sql语句,并将信息封装到boundsql对象,

boundsql:表示动态生成的生气了语句及相应的参数信息.

8.mybatis配置:

sqlMapConfig.xml 核心配置文件,配置了mybatis的运行环境,

dataSource: type="POOLED" 配置数据库连接池,

通过mapper resource来引进\*.xml配置文件

构建mybatis连接

Resource.getResourceAsStream(sqlMapConfig.xml); 获得InPutStream 流

通过new SQLSessionFactoryBuilder().bulid(inputStream);获得sqlSessionFactory

由sqlSessionFactory会话工厂创建sqlSession会话,操作数据库通过SQLSession进行,SQLSession是线程不安全的,每个线程都应该有他独自的sqlSession,使用完后关闭.

myBatis底层自定义了Executor执行器接口操作数据库,Executor接口有两个实现,一个是基本执行器,一个是缓存执行器,Executor是真正操作数据库的.

正常的dao层开发,SQLSessionfactorybuilder当做一个工具类,一旦使用完毕,就应该销毁,最佳使用范围是在方法内部.

SQLSessionFactory要单例存在,一旦创建就应该在整个程序运行期使用.整合spring,交给spring来管理.

SQLSession是多例的,是线程不安全的,不能被共享的,使用范围是方法的内部,一旦使用完成,必须在finally中关闭.

9.#{}是一个占位符,自动进行java类型到jdbc类型的转换,

${}表示拼接字符串,不进行转换,有可能导致sql注入.

10.数据库池,sql与java分离,传统sql参数不确定问题,返回结果问题

11.分页,内存分页和物理分页

12.缓存

一级缓存:sqlsession默认

二级缓存:一个namespace下的mapper

开启二级缓存:

13.resultType:返回字段和java对象的属性一致使用

resultmap:返回字段和java对象属性不一致情况比较多的情况使用.(可以定义多表查询的结果集)

14.尽量少使用if,choose,使用case,when,decode

大量使用if,会增加大量的空格,增加网络传输时间,

大量使用if,choose语句,不可避免每次生成的sql不一致,导致oracle硬解析

15.

case,when,decode会需要Oracle函数解析,影响sql执行时间,正常不会出现函数解析导致sql变慢的问题

影响sql执行效率的一般是 join,order by,distinct,partition by

16.mybatis设计模式

builder模式:创建工厂时sessionfactory

工厂模式:创建sessionfactory

单例模式:errorContext,logFactory

代理模式:MapperProxy,ConnectionLogger,用的jdk动态代理,

executor.loader包使用cglib或者javassist达到延迟加载的效果

组合模式:sql节点和各个子类choosesqlNode

模板方法模式:

适配器模式:mybatis接口和jdbc和log4j等日志框架的适配实现

装饰者模式:缓存

迭代器模式:

17.in的使用

大量in的使用会造成数据大量的硬解析,和共享池sql碎片

子查询绑定变量实现方式