

```
SqlSessionFactory
事务
使用 SqlSession
注入映射器
使用 Spring Boot
使用 MyBatis API
```

使用 Spring 示例代码

maven

使用 Spring Batch

MyBatis-Spring 1.10 没布以后,提供了三个 bean 以供构建 Spring Batch 应用程序: MyBatis Batch 以供格理 Spring Batch が MyBatis Batch に MyBatis Batch Batc Reader 和 MyBatisBatchItemWriter . 而在 2.0.0 中,还提供了三个建造器(buikder)类来对 Java 配置提供支持: MyBatisPagingIt

MyBatis MyBatis•

10元 本章是关于 Spring Batch © 的,而不是关于 MyBatis 的批量 SqlSession 。要直找关于批量 session 的更多信息,请参考 使用 SqlSes

MyBatisPagingItemReader

这个 bean 是一个 ItemReader ,能够从数据库中分页地读取记录。

它执行 setQueryId 属性指定的查询来获取请求的数据。这个查询使用 setPageSize 属性指定了分页请求的大小,并被执行。其它的页面将在必要时被请求(例如调用 read()方法时),返回对应位置上的对象。

reader 还提供了一些标准的请求参数。在命名查询的 SQL 中,必须使用部分或全部的参数(视乎 SQL 方言而定)来构造指定大小的结果集。这些参数是:

- __page: 欲读取的页码 (从 0 开始)
- __pagesize: 每一页的大小,也就是返回的行数
- __skiprows: __page 和 __pagesize 的乘积

它们可以被映射成如下的 select 语句:

cselect id="getEmployee" resultMap="employeeBatchResult">
SELECT id, name, job FROM employees ORDER BY id ASC LIMIT #(_skiprows), #(_pagesize)

配合如下的配置样例:

```
cean is-"reader" class-"org.mpatis.spring.batch.hyBatsiPagingItemBeader">
corposerty name-"sqltessionFactory" ref-"sqltessionFactory" />
corposerty name-"queryis" value-"com.my.name.space.batch.fmploperBapper.getImployee" />
c/beam)
@Configuration
public class batchAppConfig (
public NatalPagingItemBoser:Cheologues reader() {
    return new NyBatchAppConfiguration
    respective new NyBatchAppConfiguration
    respective
```

让我们通过一个更复杂一点的例子来阐明一切:

```
Geom is "datelasod'riterialeader"

Class" "vg. mybuli, spring haten. "Mystisfyglightemleader"

positisesian kervine" hatendheadinglessions actory

poparasterivalues - refr'idstelfaranters'

poparafia" (nm. posice satuh. Lamplehaper, query/ber/interactions/dapecificliseS)ot'

popargia" (nm. posice satuh. Lamplehaper, query/ber/interactions/dapecificliseS)ot'

popargia" (nm. posice - No. )

coge="stary")
                      utilings id-"datesParacters" scope-"stop")

centry keys-"pesterday" values-"djobbecordoxcontext['ERTACTIOL_STATL_DATE'])*/)

centry keys-"scops" values-"djobbecordoxcontext['DATACTIOL_STATL_DATE'])*/)

centry keys-"scops" values-"djobbecordoxcontext['DATACTIOL_STATL_DATE'])*/)

centry keys-"scops_"oxt-tam_post" values-"djobbecordoxcontext['TAST_DAV_G'_DB_JONT_DATE'])*/)

centry keys-"scops_"dt_post_tam_post' values-"djobbecordoxcontext['TAST_DAV_G'_DB_JONT_DATE'])*/)

centry keys-"scops_"dt_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_post_tam_po
           @Configuration
public class BatchAppConfig {
                      Modific Class SetchAppConfig (

Bleen

Bleen

Dulli (MastiSPagingItemSenser-User) dateDasseCriteriaReader(

pRolaw(*(MasterDassectry)) NapoSeria, Dojecto dateDasseCriteriaReader(

pRolaw(*(MasterDassectry)) NapoSeria, Dojecto dateDassectry) throat Exception (

return new hybridargingItemSenseEndisTecture)()

.augrid[Cross, years, sect. achter, ficesportuper, quaryUser_InteractionsObjectificImsclin_
.augrad[Cross, years, sect. achter, ficesportuper, quaryUser_InteractionsObjectificImsclin_
.apactica(200)

.build();
                      Straticose

Stem

Straticose

Stem

Dull: ResOrting, Object: deteParameters(

Stalland (*/sjobroution/detext(CTMACTON_SIAT_DATE'))') LocalDate yesterday,

Stalland (*/sjobroution/detext(TDMACTON_SIAT_DATE'))') LocalDate firstDayOfTheMouth,

Stalland (*/sjobroution/detext(TDMACTON_SIAT_DATE'))') LocalDate firstDayOfTheMouth,

Stalland (*/sjobroution/detext(TDMACTON_SIAT_DATE'))') LocalDate firstDayOfTheMouth) (

Negotifies, Objecto ass - one NanNaspo();

ass.port('Ctmacton', visterday))

ass.port('Ctmacton', visterday);

ass.port('Ctmacton', incomplex);

ass.port('Ctmacton', inc
```

F面的経例使用了几个在西:

MyBatisCursorItemReader

这个 bean 是一个 ItemReader ,能够通过游标从数据库中读取记录 IRES 为了使用这个 bean, 你需要使用 MyBatis 3.4.0 或更新的版本。

它执行 setQueryId 属性指定的查询来获取请求的数据(通过 selectCursor() 方法)。每次阐用 read() 方法时,将会返回游标箱间的下个元素,直到没有剩余的元素了。

这个 reader 将会使用一个单独的数据库连接,因此 select 语句将不会参与到 step 处理中创建的任何事务。 当使用游标时,只需要执行普通的查询:

cselect id="getEmployee" resultNap="employee&atchResu SELECT id, name, job FROM employees ORDER BY id ASC </select>

搭配以下的配置样例:

```
them [d=reserr class*org.mphatis.spring.batch.hphatisCursorttemReserr)
conceptry neme*slikesipOnketop* refr*slikesipOnketop* //
conceptry neme*ourpid* value*com.mp.neme.space.batch.fmployeeMopper.getfmployee* //
CMRESO
gConfiguration
poblic class SetchiopConfig (
gBean
public highstisCursorItemBeadercitpipyee) reader() (
return new highstisCursorItemBeadersLublarcitpipyee()
.silessionPeatroy(silessionPeatroy())
.oupvy20f(cro.my,name.space.batch.bubloyeeMupper_getEmployee*)
.bullE();
}
```

MyBatisBatchItemWriter

```
这是一个 Itemsfriter ,通过利用 SalSessionTemplate 中的批量处理功能来对给定的所有记录执行多个语句。 SalSessionFactory 需要被配置为 BAICH 执行类型
当调用 write() 时,将会执行 statementId 属性中指定的映射语句。一般情况下,write() 应该在一个事务中进行调用。
下面是一个配置样例:
```

```
Coom id-"mriter" class-"org.mynatis.spring.batch.Mydatidatch.tembriter">
corcoperty name-"silisesionFactory" refa-"silisesionFactory" />
corcoperty name-"sitatementia" value-"com.my.name.space.batch.EmployeeMapper.updateEmployee" />
c/Deems/
```

将 ItemReader 读取的记录转换为任意的参数对象

默认情况下,MyBatisBatchItemWriter 会将 ItemReader 读取的对象(或 ItemPr ssor 转换过的对象)以参数对象的形式传递的 MyBalis(SqlSession和pdate())。 如果你想自定义传递给 MyBalis 的参数对象,可以使用 itemToParameterConverter 选项。使用该连项后,可以传递任意对象给 MyBalis、举个例子:

```
noblic class ItemDarasetrmpcConverters {
  public static Ch Converter(), NupiString, Object) creatsItemDaraseterMspConverter(String operationBy, LocalDateIlms operationAt) {
  return item > {
    NupiString, Object) parameter = new MasNMspC()};
    parameter.put("item", item);
    parameter.put("item", item);
    parameter.put("powerisobly", operationBy);
    parameter.put("powerisobly", operationBy);
    return parameter.put("operationBut", operationBut);
    return parameter.put("powerisobly", operationBut);
    return parameter.put("powerisobly", operationBut);
    return parameter.put("powerisobly", operationBut);
    return parameter.put("powerisobly");
    return parameter.put("
接下来,编写 SQL 映射。
            select idm"createPerson" resultTypem"org.mybatis.spring.sample.domain.Person">
  insert into persons (first_name, last_name, operation_by, operation_at)
  values(#(item.firstName), #(item.lastName), #(operationBy), #(operationAt))
 最后,配置 MyBatisBatchItemWriter。
   @Configuration
public lass MatchAppConfig {
    @Rean
    paolic hybridistaticthemirter-Oresons witer() throat Exception {
    return now hybridistaticthemirter-Multer-Oreson()
        subtestion/actory(callsetion/actory())
        istatementify org. public ison/actory()
        istatementify org. public ison/actory()
        istatementify org. public ison/actory()
        istatementify org. public ison/actory()
        inter/oresonactor(converter(createltemicAppconverter("natch_bea_config_user", localDatetime.now()))
        bet/if():
them is-"writer" class-"org.mybetis.spring.absch.mybetishtchitemirther")
cyroperty name-"sulfession/mattury" ref-"sulfession/mattury"/;
cyroperty name-"stetementia" value" mys.mybetis.spring.ammole.mapper.ResonMapper.createPerson"/)
cyroperty name-"stetementia" value" mys.mybetis.spring.ammole.mapper.ResonMapper.createPerson"/)
cyroperty name-"stetementia" value" mys.mybetis.spring.ammole.mapper.resonMapper.createPerson"/)
cyroperty name-"stetementia value" value" and configurer"/)
cyroperty name-"stetementia value" and cyroperty)
cyroperty
cyroperty
cyroperty
cyroperty
cyroperty
cyroperty
使用复合 writer 对多个表进行写入(但带有问题);
 这个小技巧只能在 MyBatis 3.2+ 以上的版本中使用,因为之前的版本中含有导致 writer 行为不正确的 问题 🕼
 如果就量处理时需要写入复杂的数据,例如含有关联的记录,甚至是何多个数据车写人数据,你可能就需要一种办法来绕开 insert 语句只能插入到一个事中的限制。为了绕过此限制,我处理必须难备好要通过 writer 写入的项。 忽而,基于对维处理的数据的观察,可以尝试使用下面的方法来解决此问题。下面的方法来解决此问题。
 在这种方法中,处理 Spring Batch 源的处理器中将会持有各种不同的记录。假设每个项部有一个与 InteractionAletadata 俄关联的 Interaction,并且还有两个不相关的行 VisitorInteraction 和 CustomerInteraction,这时被持有器(holder)看起来像这样:
   public class InteractionRecordToWartieInMultipleTables {
    private final VisitorInteraction visitorInteraction;
    private final CustomerInteraction customerInteraction;
    private final Interaction interaction;
 在 Spring 配置中要配置一个 (compositeItenderiter ,它将会将写入银作委托到特定种类的 willer 上面去,注意 InteractionAbleIadala 在例子里面是一个关联,它需要简先被写入,这样 Interaction 才能获得更新之后的建。
   teem id="interactionstressiriter" (lass="org.springfreecont.batch.item.support.CompositeItem
corpusery name="delegater")
(list)
cer# seas="visiterInteractionsWriter)
cer# seas="visiterInteractionsWriter)
   gConfiguration
public Lass MatchAppConfig {
    @Rean
    public Cases MatchAppConfig {
    @Rean
    public CompositeIntermiter() {
        CompositeIntermiter() public CompositeIntermiter - new CompositeIntermiter();
        List(Intermiter()) writers - new ArrayList(A);
        writers. add(cuttoreIntermiter());
        writers. add(cuttoreIntermiter());
        writers. add(cuttoreIntermiter());
        writers. add(cuttoreIntermiter());
        writers. add(cuttoreIntermiter());
        compositeIntermiter();
        compositeIntermiter();
        return compositeIntermiter();
    }
}
 接下来需要配置每一个被委托的 writer; 例如 Interaction 和 InteractionMetadata 对应的 write
 (been id="interactionMetadatairiter"
class="0g.nymain.soring.beto.hlydatiodatohleamiter"
prosidessionOmiet-rer="0ratofassionMemplate"
prosidessionOmiet-rer="0ratofassionMemplate"
prosidessionOmiet-rer="0ratofassionMemplate"
prosidesemolid="0cn.ny.name.spec.boto.hloreractionMeconfloiriteInMultipleTablesMapper.insertInteractionMetadata"))
         (bean id="interactioninter"
class="org.mybatis.spring.batch.PyBatisBatchItemWriter"
p:sqlSessionTemplate-ref="batchSessionTemplate"
p:statementId="com.my.name.space.batch.InteractionRecordToWr
p:statementId="com.my.name.space.batch.InteractionRecordToWr
p:statementId="com.my.name.space.batch.InteractionRecordToWr
p:statementId="com.my.name.space.batch.InteractionRecordToWr
p:statementId="com.my.name.space.batch."
 和 reader 中的一样,通过 statementId 属性规定对应命名空间前缀的查询。
而在映射器配置文件中,应该根据每种特定的记录编写特定的语句,如下所示
 Closert is "insertiteration related"

parameter/per "(n.m., batch.interation.liten.Interation Record object in the Interation Record object in the Interatio
```