文章编号：2096-1472(2017)-01-01-04

数据库同步技术的研究与实现

刘娟娟，刘 帅
（防灾科技学院，河北 三河 065201）

摘 要：为解决当前分布式系统中多数据源、多异构数据库问题，针对现有的数据库同步方案在实际应用中所暴露出的资源损耗大、可移植性差、应用范围窄等问题，在分析了现有的数据同步技术的各自优缺点的基础上，提出了采用Sql plus和Merge语句相结合的数据库同步方案，并对数据同步的系统进行了设计，最后通过功能比较和性能分析，该方案在资源损耗、可移植性、应用范围等方面更具优势。

关键词：数据同步；数据转换；数据复制
中图分类号：TP311.132 **文献标识码**：A

Research and Implementation of Database Synchronization Technology

LIU Juanjuan,LIU Shuai
(*Institute of Disaster Prevention,Sanhe* 116023,*China*)

Abstract: In order to solve the various existing problems in the distributed system,including multiple data sources,multiple heterogeneous databases,low resource efficiency exposed in the practical application of the database synchronization scheme,poor portability,and narrow application range,the paper analyzes the respective advantages and disadvantages of existing data synchronization techniques,puts forward a new database synchronization method with the combination of Sql plus and Merge statements,and designs the data synchronization system.Finally,the results of function comparison and performance analysis show that the scheme proposed in this paper is at a great advantage in several aspects,such as resource consumption,portability and application range.

Keywords:data synchronization;data conversion;data replication

1 引言(Introduction)

近年来在信息技术飞速发展的带领下，各行业对信息管理技术都逐渐步入深层次的运用阶段，各个单位面临的环境越来越复杂，由于各个行业的市场发展和扩张，越来越多的单位都迈入了跨区分布式经营的行列，而与单位分布式经营相呼应的结果就是产生了分布式数据存储环境[1]。并且现代信息技术的发展是日新月异的，当前正在使用的很多信息系统在不同的时期使用不同技术进行开发，而且又由不同的团队各自进行设计，使得这些系统的平台往往呈现出异构性，导致了现有数据的差异性。为保证当前使用的各个信息系统之间数据交换的畅通，保障各个系统之间数据传输中的安全性及一致性，真正实现数据共享，避免出现数据孤岛。因此，如何及时有效地处理这些异构数据库中的数据成为一个颇受关注的研究问题。

为解决当前多应用系统中使用Sql Server、Oracle等多异构分布式数据库的数据集成、合并等问题，结合使用异构数据库数据合并技术，实现源数据库Sql Server、Oracle和目标数据库Oracle的数据同步。另外其常常涉及还包括Access、DB2、MYSql等数据库。

2 数据库同步技术解析(Analysis of database synchronization technology)

2.1 媒介同步技术

媒介同步技术是一种20世纪后期存在的数据同步技术。其原理是把需要更新的数据拷贝到如硬件磁盘等媒介中，通过专用信息通道发送到需要数据同步的区域，再由目的区域管理员进行新增数据的添加与更新[2]。本方法由于更新的时间长效率慢等特性，已不再使用。

2.2 发布/订阅同步技术

Sql Server的发布/订阅是一种单对多的依赖关系备份的机制，是多个订阅数据库监听数据源数据库的变更，当源数据库数据发生变更时通知订阅它的数据库进行数据同步变更，通过该机制实现整个过程不用代码编写。其本质是生成源数据库的快照，是Sql Server多数据库间保持数据变更同步一种方案。在发布/订阅中，各个订阅数据库采取异步方式来

被数据同步，也就是当源数据库数据发生变更的同时，订阅数据库要无条件的服从源数据库的数据操作并同步完成数据变更。

在实际应用中，从专业的角度来分析，发布/订阅技术存在如下缺陷：

(1)同步双方的表结构不能更改，并且表结构必须一致，一旦发布方数据库表结构发生更改需要重新生成数据库快照。

(2)对于大数据量的同步缺乏可靠的保证，其中主要原因是大数据量的数据同步过程中不具备可靠的通信链路，造成订阅数据库与源数据库之间数据传输延时问题，同时会致使有效数据严重丢失，最终造成数据同步失败。

(3)源数据库在发布过程中若设置为指定时间广播发送同步消息，无论订阅数据库是否收到消息，源数据库发送后便终止尝试发送，其无法保证订阅数据库及时有效的进行数据同步。

2.3 复制同步技术

数据复制同步是指：数据必须实时，如果不是实时，那只能叫异步数据迁移；数据必须保证其准确性，同步端接受的同步数据必须保证数据的唯一性；复制的数据可实时在线查询；数据库源数据不能被更改，其具有独立性；复制过程中具有监控机制。

在具备以上条件的由两个及以上数据库平台在进行数据库复制的整个程序，将源数据库中发生变化的需要复制的数据通过网络发送给同步端需要数据变更的数据库中，使得分布式系统得以实现数据同步，数据复制的整个过程是解决分布式系统数据同步的基础组成部分。

当前数据复制技术在各个企业信息管理平台下应用较为广泛，其中Oracle数据库中的DATAGUARD完全免费并且支持断电续传，但在使用过程中需要打开归档模式并且宽带传输要求比较高，并且对目的数据库跟源数据库的操作系统跟软件版本有一定限制；还有支持Oracle、DB2等传统关系型数据库的HVR技术，支持异构数据库，可实现一对多、多对一等复杂情况处理，但是需要单独收费。

上述两种比较具体的数据复制技术在具体应用过程中都有一定的局限性，选择一种免费的安全性,以及稳定性较高的复制技术是实现数据同步的关键，本文结合Sql plus和Merge语句实现数据同步应运而生。

3 数据复制的变更数据捕捉方式(Data copy change data capture method)

变更数据捕捉是为了获取自上次数据同步后需要同步的源数据库中的数据表进行插入、更新和删除等活动需要变更的数据。变更数据是源数据库中信息发生改变的数据，变更的数据主要包括当前变更数据的映像或者变化序列，还包括详细的控制信息来保证数据差异性产生后的监测与解决[3]。

2010年以来，使用次数比较多的数据捕获方法包括基于数据日志法、基于触发器法、基于API法、影子表法和控制变化法等几种，结合不同方法实现过程中的不同原理介绍如下。

3.1 基于数据日志法

在数据库操作过程中数据库日志记录了数据库中对数据表进行增、删、改等操作信息，作为维护数据库中数据信息的依据。它包含对数据库中数据表增、删、改等操作中100%正确的具体数据变更序列和数据操作信息，因此，可以把数据库日志作为数据库恢复以及维护数据完整性的重要工具。

基于数据库日志的变更数据捕捉方法就是通过分析源数据库日志中的操作和被操作的数据来确定复制对象的变化序列[1]。基于数据日志法的捕捉流程，如图1所示。

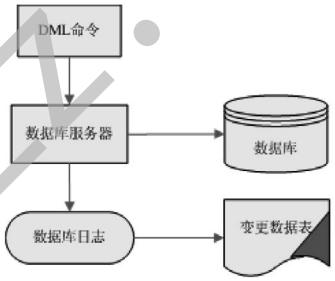

图1 基于数据日志法的捕捉流程图

Fig.1 Capture flow chart based on data log method

基于数据日志法的好处在于它对任何类型的数据库复制都适用，因为当前多个数据库平台都拥有日志文件，使用数据库日志文件作为数据同步的基本方式，其优点表现在耗费极少的系统资源，并且更加方便快捷。但本方法也存在一些缺点：在具体应用中有些数据库系统为保证其软件的安全性而没有向第三方公布其日志的存储格式，使得要开发一个基于日志的变化捕捉的程序非常困难，除非在具体开发过程中向数据库厂家索取相应日志存储格式以及日志文件相应操作接口；为解决异构数据库数据复制问题，其数据模式、数据类型等方面有着不同的操作方式，还需要对通过系统日志提取出来的变化操作信息进行一些处理。Oracle、MySql、Sql Server等主要数据库厂商都使用的是基于日志的捕捉方法进行数据复制。

3.2 基于触发器法

基于触发器法是在源数据库中建立触发器，实时监控源数据库中数据表的insert、delete、update等操作，当对源数据对象进行insert、delete、update操作时，触发器将变化的数据表中的数据序列提取出来。其捕捉流程，如图2所示。

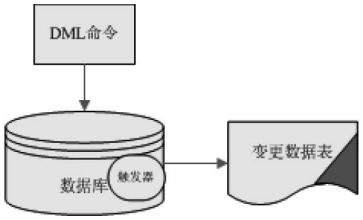

图2 基于触发器法的捕捉

Fig.2 Capture based on trigger method

基于触发器法的优点是Sql Server、Oracle等主流数据库软件都可以使用触发器，并且在捕获数据变化过程中耗时少有较高的应用效率。但是使用触发器捕捉变更数据的缺点是消耗操作系统物理资源比较大，在具体项目实现过程中相对较为复杂的逻辑业务，使用触发器过程中管理不便，并且当前项目使用的数据库系统若不支持触发器，则触发器复制方法失效。

3.3 基于API法

基于API法是解决当前开发系统中使用的有些不具有触发器接口和日志管理机制的非关系型等数据库不能同步问题[3]。其在系统平台和数据库之间引入中间件，通过API中间件实现具有类似触发器或者日志管理机制相似功能[4]。在项目具体开发过程中中间件的具体使用包括记录需要复制的数据对象的变化序列，并且根据变化序列操作相应数据库的数据表从而完成数据复制的功能，基于API法的捕捉流程，如图3所示。

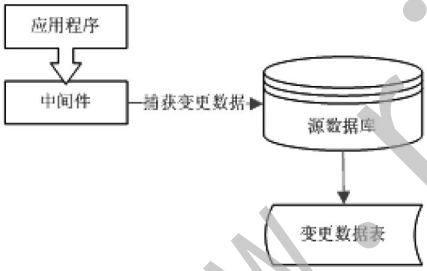

图3 基于触发器法的捕捉

Fig.3 Capture based on trigger method

在具体使用过程中，基于API法的优点是使用的系统资源基本不足1%，并且可以根据需求与绝大多数复制数据类型相互匹配；它还是有一定的不足，其表现在不能记录源数据库的操作过程，例如使用数据操纵语言在CMD命令行中进行新增，更新，删除操作时所造成的数据变更，API无法监控到，除此之外在使用API法操作海量级数据复制过程中，由于数据处理的逻辑性较高，可能会造成程序运行效率低下[4]。

3.4 影子表法

影子表法是在数据库初始化时为需要操作的对象表(T)建立与之相对应的影子表(S)，其原理是在时间节点A时刻记录一份对象表(T)的数据并且将其存储进影子表(S)，对象表(T)操作完成后的时间节点B时刻将对象表(T)与影子表(S)进行内容对比获取增量数据内容用于目的数据库进行数据复制[5]。基于影子表法的捕捉流程，如图4所示。

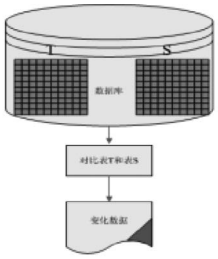

图4 基于影子表法的捕捉

Fig.4 Based on the capture of the shadow table method

基于影子表法能基于任何一个数据库完成，并且程序只需编写一次便可以在多种系统平台下运行。其程序运行消耗是使用很低的管理成本和与对象表一样的存储空间即可，具体操作过程中其传输的数据只是增量，具有较高的工作效率。但是这种方式的缺陷是得到的仅仅是发生变化的增量数据，而没有记录下中间操作过程的信息，故不能提供足够的控制信息用以完成数据回滚。

以上四种变化捕捉方式机制和资源等的比较，详见表1。

表1 四种变化捕捉方法之间的比较

Tab.1 A comparison between four kinds of change capture methods

性能 / 捕捉方式	依赖机制	资源损耗	可移植性	实用性	同步/异步
基于数据日志	日志	消耗一般	低	范围一般	同步
基于触发器	触发器	消耗较大	低	范围广	同步
基于API	中间件	消耗一般	低	范围窄	同步
影子表法	无	基本不消耗	高	范围广	异步

4 跨平台Oracle数据库同步设计方案(Cross platform Oracle database synchronization design)

本论文研究的是在分布式异构数据库环境下，由系统管理员启动此同步程序，从而将省局的Linux服务器中下的Oracle数据库中的数据单向同步到Windows服务器下的Oracle数据库中。若数据有变更，只更新变更数据，如图5所示。

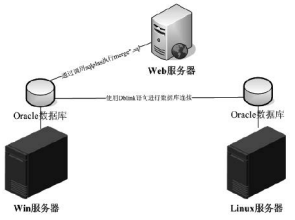

图5 同步模块部署图

Fig.5 Synchronization module deployment diagram

结合此系统的背景和需求，该系统要实现的同步过程的目标包括：

(1)高效快捷：系统中涉及险情等信息，在紧急情况下，省级数据库数据要能通过系统立即更新到Windows数据库

中，故该同步应能保证在即时状态下实现省中心数据库与Windows数据库的数据的一致。

(2)查看方便：能反映出同步状况，提供一个便于查看和分析的方式。

(3)准确无误：应能确保数据传输结果的正确性。

(4)安全可靠：此系统是数据管理部门对整个数据管理过程中所有资料数据进行管理的重要工具，因此在系统的设计、开发中必须重视系统的安全性和可靠性。

具体实施步骤如下：

步骤一：在Linux服务器上的Oracle数据库创建用户并赋予权限，如图6所示。

```
CREATE USER "ORACLE_MERGE" PROFILE "DEFAULT" IDENTIFIED BY "fzgood*" DEFAULT TABLESPACE "FZDATA" TEMPORARY TABLESPACE "TEMP01"
ACCOUNT UNLOCK
GRANT ALTER ANY MATERIALIZED VIEW TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT CREATE ANY MATERIALIZED VIEW TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT CREATE ANY TABLE TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT CREATE ANY TRIGGER TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT CREATE DATABASE LINK TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT CREATE PUBLIC DATABASE LINK TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT DELETE ANY TABLE TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT DROP ANY MATERIALIZED VIEW TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT DROP ANY TRIGGER TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT DROP PUBLIC DATABASE LINK TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT EXECUTE ANY PROCEDURE TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT GLOBAL QUERY REWRITE TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT INSERT ANY TABLE TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT UNLIMITED TABLESPACE TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT UPDATE ANY TABLE TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT "AQ_USER_ROLE" TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT "CONNECT" TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT "DBA" TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT "EXP_FULL_DATABASE" TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT "IMP_FULL_DATABASE" TO "ORACLE_MERGE" WITH ADMIN OPTION
GRANT "RESOURCE" TO "ORACLE_MERGE" WITH ADMIN OPTION
```

图6 创建用户并赋予权限图

Fig.6 Create user and assign permissions

步骤二：在Windows服务器上的Oracle数据库创建DBlink，如图7所示。

```
CDBlink.sql - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
ALTER USER "QZ_DATA_BAK" IDENTIFIED BY "fzgood*";
commit;
conn QZ_DATA_BAK/fzgood*;
DROP DATABASE LINK fzlink;
create database link fzlink
connect to ORACLE_MERGE identified by "fzgood*"
using '*.*.*.*/FZMPU';
```

图7 创建DBlink

Fig.7 Create DBlink

步骤三：调用web服务器相应数据同步函数进行同步操作，如图8所示。

```
private void Start_Oracle_Merge() //开始数据合并
{
    ctrInformation_Id=StringUtils.getId();
    ctrInformation.setId(ctrInformation_Id);
    ctrInformation.setSumNum("0");
    ctrInformation.setCompleteNum("0");
    ctrInformation.setStartTime(inputDate);
    ctrInformation.setRemark1("0");
    this.baseBizService.save(ctrInformation);
    mergeTime=ctrInformation.getStartTime();
    String DICT_methodName="";
    String Sqlcommand="sqlplus FZ_DATA_BAK/fzgood*  @";
    string _strcontextPath = EnvManager.getRealPath();
    String _strBasePath = _strContextPath+ "application"+File.separator+"m
}
private void Oracle_Merge(String _strBasePath,
                          String Sqlcommand,
                          String DICT_methods ,
                          String DICT_methodName)
                          //完成 主要数据 合并 函数
{
    new Thread(new RunSql(Sqlcommand+"FZ_"+DICT_methods+"_Merge_01.sql",_s
    new Thread(new RunSql(Sqlcommand+"FZ_"+DICT_methods+"_Merge_02.sql",_s
    new Thread(new RunSql(Sqlcommand+"FZ_"+DICT_methods+"_Merge_10.sql",_s
    new Thread(new RunSql(Sqlcommand+"FZ_"+DICT_methods+"_Merge_11.sql",_s

    new Thread(new RunSql(Sqlcommand+"FZ_"+DICT_methods+"_Merge_01.sql",_s
    new Thread(new RunSql(Sqlcommand+"FZ_"+DICT_methods+"_Merge_02.sql",_s
    new Thread(new RunSql(Sqlcommand+"FZ_"+DICT_methods+"_Merge_10.sql",_s
    new Thread(new RunSql(Sqlcommand+"FZ_"+DICT_methods+"_Merge_11.sql",_s
}
```

图8 数据同步函数图

Fig.8 Data synchronization function diagram

步骤四：web服务器通过调用Sql plus程序处理相应表的*.sql文件，如图9所示。

```
set arraysize 100
merge into FZ_DATA_BAK.FZ_DEBN b using fzdata.FZ_FZ_DEBN@fzlink c on
(b.STARTDATE=c.STARTDATE AND b.STATIONID=c.STATIONID AND b.POINTID=c.POINTII
b.ITEMID=c.ITEMID)
when matched then update set
b.SAMPLERATE=c.SAMPLERATE, b.OBSVALUE=c.OBSVALUE, b.DATE_INDEX=c.DATE_INDEX wh
(b.SAMPLERATE!=c.SAMPLERATE  or b.DATE_INDEX!=c.DATE_INDEX)
when not matched then
insert  values
(c.STARTDATE, c.STATIONID, c.POINTID, c.ITEMID, c.SAMPLERATE, c.OBSVALUE, c.DATE_I
commit;
```

图9 *.sql文件图

Fig.9 *.sql file

通过以上步骤实现了跨平台Orale数据库的数据同步功能其在数据捕获上的具体性能，详见表2。

表2 性能描述

Tab.2 Performance description

依赖机制	资源损耗	可移植性	实用性	同步/异步
Sqlplus	消耗一般	高	范围较广	异步

5 结论(Conclusion)

本文描述了以J2EE为平台对跨平台Oracle数据库的同步模块的具体实现方法。其采用Sql plus动态调用Merge语句来作为变更数据的捕捉方式。这种方式灵活易实现，对模块进行测试后得出该模块能保证数据的准确传输，并且对复制对象的表可进行自定义，但是也存在一定的局限性就是web服务器上必须安装有Sql plus客户端。

参考文献(References)

[1] 覃章荣,张军洲,诸葛隽.基于Web Service的异构数据库同步系统设计与实现[J].计算机技术与发展,2009,19(12):221-224.

[2] 刘潇,邵定宏.基于多代理的异构数据库转换系统的实现[J].化工自动化及仪表,2012(04):518-520.

[3] 傅颖勋,罗圣美,舒继武.一种云存储环境下的安全网盘系统[J].软件学报,2014,25(08):1831-1843.

[4] 李林,等.田间数据传输同步策略与中间件研究[J].农业机械学报,2016,47(1):279-288.

[5] 鲍爱华,等.基于哈希树的分布式目录同步方法[J].解放军理工大学学报,2013,14(6):608-616.

作者简介：

刘娟娟(1983-)，女，本科，助理工程师.研究领域：软件开发.

刘　帅(1983-)，男，本科，副高级工程师.研究领域：软件开发.