Программа DW.Loader

# ВВЕДЕНИЕ

Программа DW.Loader предназначена для передачи данных между различными базами данных. Программа может быть использована для решения следующих задач:

* Репликация справочников
* Обмен документами
* Обработка хранилищ данных

Программа написана в виде приложения командной строки Microsoft® Windows® с использованием Microsoft® .NET Framework 4.0.

Программа отличается от подсистемы репликации A2 тем, что она работает без транспортного уровня. И сервер и клиент должны быть доступны по сети.

## требования К СИСТЕМЕ

На компьютере должен быть установлен Microsoft® .NET Framework 4.0.

Серверы баз данных должны быть доступны по сети.

# Принцип работы

После запуска, программа выполняет задания, указанные в стандартном файле конфигурации DW.Loader.exe.config. Каждое задание имеет один из трех типов.

* Catalog ‑ репликация каталогов
* Document ‑ репликация документов
* Journal ‑ загрузка журналов

Задания выполняются последовательно. Если в процессе выполнения задания возникли ошибки, то задание прерывается и программа переходит к выполнению следующего задания. Алгоритм работы определяется типом задания.

# Настройка КОНФИГУРАЦИИ

## Файл конфигурации программы

Файл конфигурации DW.Loader.exe.config представляет собой стандартный xml-файл конфигурации для приложений Microsoft.NET.

Структура файла:

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<!--описание обработчиков раздела jobs. Нужно всегда-->

<configSections>

<section name="jobs" type="DW.Loader.Jobs, DW.Loader" />

</configSections>

<!--настройки программы-->

<appSettings>

<add key="autorestart" value="false"/>

</appSettings>

<!--описание строк подключения-->

<connectionStrings>

<add name="Name1"   
 connectionString="connectionString1"/>

<add name="Name2"   
 connectionString="connectionString2"/>

</connectionStrings>

<!--описание задач-->

<jobs>

<add name="jobname" type="jobtype" dataKey="dataKey" useGuid="False"  
 source="SourceConnectionStringName"   
 target="targetConnectionStringName"   
 sourceProcedurePrefix="sourceProcedurePrefix"   
 targetProcedurePrefix="targetProcedurePrefix" />

</jobs>

</configuration>

Файл содержит следующие разделы:

|  |  |
| --- | --- |
| configSections | Раздел, подключающий тип-обработчик раздела <jobs>. Нужен всегда. |
| appSettings | Общие настройки приложения. Формат раздела - словарь. Возможные ключи словаря: autorestart - булевское значение. Если установлено в true, то программа автоматически перезапускается после критических сбоев. |
| connectionStrings | Описание строк подключения к базам данных. Формат раздела - стандартный словарь строк подключения. |
| jobs | Описание задач, выполняемых программой. Формат раздела - словарь. Каждый элемент словаря имеет следующие атрибуты:   |  |  | | --- | --- | | name | Обязательно. Наименование задания. | | type | Обязательно. Тип задания. Возможны следующие значения:  Catalog ‑ репликация каталогов  Document ‑ репликация документов  Journal - загрузка журналов. | | dataKey | Необязательно. Ключ данных. | | useGuid | Необязательно. Признак того, что идентификаторы документов представлены в виде guid. По умолчанию - false. Используется только для задания типа Document. | | source | Обязательно. Имя источника данных (из раздела connectionStrings). | | target | Обязательно. Имя приемника данных (из раздела connectionStrings). | | sourceProcedurePrefix | Обязательно. Префикс хранимой процедуры на источнике данных. | | targetProcedurePrefix | Обязательно. Префикс хранимой процедуры на приемнике данных. | |

# РЕПЛИКАЦИЯ КАТАЛОГОВ

Репликация каталогов основана на концепции пакетов. В базе данных источнике имеется две специальных таблицы ­‑ PACKAGES и PACKAGE\_CONTENTS. В таблице PACKAGE\_CONTENTS накапливаются изменения реплицируемых таблиц каталогов.

Репликация каталогов выполняется заданием типа Catalog. В описании задания используются имена строк подключения для источника и приемника (source и target) и префиксы хранимых процедур для источника и приемника (sourceProcedurePrefix и targetProcedurePrefix).

При редактировании таблиц каталогов в таблицу PACKAGE\_CONTENTS последовательно записываются имя таблицы и первичный ключ (идентификатор) измененной записи.

# ОБМЕН ДОКУМЕНТАМИ

Обмен документами выполняется заданием типа Document. В описании задания указываются имена строк подключения для источника и приемника (source и target), ключ данных (dataKey), признак использования глобально уникальных идентификаторов (useGuid) и префиксы хранимых процедур для источника и приемника (sourceProcedurePrefix и targetProcedurePrefix).

Документы передаются по одному, независимо друг от друга.

Задание выполняется следующим образом.

1. Программа устанавливает соединения с источником и приемником.
2. Если признак useGuid не указан или равен False, то на источнике выполняется процедура sourceProcedurePrefix\_getid. Процедура получает ключ данных как параметр @key и должна вернуть идентификатор документа в выходном параметре @retid. Если процедура вернула 0 или null, то задание завершается.
3. Если признак useGuid равен True, то на источнике выполняется процедура sourceProcedurePrefix\_getguid. Процедура получает ключ данных как параметр @key и должна вернуть идентификатор документа в выходном параметре @retguid. Если процедура вернула нулевой идентификатор или null, то задание завершается.
4. На источнике выполняется процедура sourceProcedurePrefix\_load. Процедура получает идентификатор документа в параметре @docid или @docguid (в зависимости от значения атрибута useGuid) и ключ данных в параметре @key. Процедура может вернуть произвольное количество наборов записей произвольной структуры с одним условием: первым полем каждого набора записей должна быть текстовая строка TABLE\_NAME, которая будет использоваться при конструировании имени хранимой процедуры на стороне приемника. Обратите внимание, что неименованные поля не допустимы.
5. Для каждой записи каждого возвращенного на шаге 3 набора данных конструируется имя хранимой процедуры как targetProcedurePrefix\_TABLE\_NAME\_write. В качестве параметров этой процедуры передаются все значения полей исходного набора записей с полным сохранением типов.  
   Обратите внимание, что если записываемое поле может иметь значение null, то в соответствующем параметре процедуры обязательно нужно указывать значение по умолчанию.
6. После успешного завершения шага 4 на источнике вызывается процедура sourceProcedurePrefix\_written с параметрами @docid или @docguid (в зависимости от значения атрибута useGuid) и @key, которая должна пометить документ, как отправленный.
7. Шаги 2-6 выполняются до тех пор, пока процедура sourceProcedurePrefix\_getid (или sourceProcedurePrefix\_getguid) не вернет 0.

## Пример ОБМЕНА ДОКУМЕНТАМИ

Предположим, что нам нужно передавать "шапки" документов из одной базы данных в другую. Для определения того, что документ уже передан в другую базу, будем использовать поле D\_SENT в таблице a2doc.DOCUMENTS.

### Файл конфигурации

Пример файла конфигурации:

<?xml version="1.0" encoding="utf-8" ?>

<configuration>

<configSections>

<section name="jobs" type="DW.Loader.Jobs, DW.Loader" />

</configSections>

<appSettings>

<add key="autorestart" value="false"/>

</appSettings>

<connectionStrings>

<add name="SRC" connectionString="Server=MAIN;Database=test;Integrated Security=true"/>

<add name="DST" connectionString="Server=MAIN;Database=test;Integrated Security=true"/>

</connectionStrings>

<jobs>

<add name="Документы" type="Document"   
 dataKey="KEY1" useGuid="KEY1" source="SRC" target="DST"   
 sourceProcedurePrefix="dbo.DOCUMENT"   
 targetProcedurePrefix="dbo.DOCUMENT" />

</jobs>

</configuration>

### Процедуры на источнике

Процедура получения идентификатора:

create procedure dbo.DOCUMENT\_getid

@key nvarchar(32),

@retid bigint output

as

begin

set nocount on;

select top 1 @retid = D\_ID from a2doc.DOCUMENTS where D\_SENT=0 and D\_ID<>0;

end

Процедура загрузки:

create procedure dbo.DOCUMENT\_load

@key nvarchar(32),

@docid bigint

as

begin

set nocount on;

select TABLE\_NAME=N'DOCUMENT', D\_ID, D\_DATE, D\_SUM, D\_MEMO   
 from a2doc.DOCUMENTS where D\_ID=@docid;

end

Процедура завершения передачи:

create procedure dbo.DOCUMENT\_written

@key nvarchar(32),

@docid bigint

as

begin

set nocount on;

update a2doc.DOCUMENTS set D\_SENT=1 where D\_ID=@docid;

end

### Процедуры на ПРИЕМНИКЕ

Процедура записи шапки документа

create procedure dbo.DOCUMENT\_DOCUMENT\_write

@TABLE\_NAME nvarchar(32),

@D\_ID bigint,

@D\_DATE datetime,

@D\_SUM money,

@D\_MEMO nvarchar(255) = null

as

begin

set nocount on;

if exists(select \* from a2doc.DOCUMENTS where D\_ID=@D\_ID)

update a2doc.DOCUMENTS set D\_DATE=@D\_DATE, D\_SUM=@D\_SUM, D\_MEMO=@D\_MEMO where D\_ID=@D\_ID;

else

insert into a2doc.DOCUMENTS (D\_ID, D\_DATE, D\_SUM, D\_MEMO)

values (@D\_ID, @D\_DATE, @D\_SUM, @D\_MEMO)

end

### Процедуры на источнике для РАБОТЫ С GUID

Процедура получения идентификатора:

create procedure dbo.DOCUMENT\_getid

@key nvarchar(32),

@retguid uniqueidentifier output

as

begin

set nocount on;

select top 1 @retguid = D\_GUID from a2doc.DOCUMENTS where D\_SENT=0 and D\_ID<>0;

end

Процедура загрузки:

create procedure dbo.DOCUMENT\_load

@key nvarchar(32),

@docguid uniqueidentifier

as

begin

set nocount on;

select TABLE\_NAME=N'DOCUMENT', D\_ID, D\_DATE, D\_SUM, D\_MEMO   
 from a2doc.DOCUMENTS where D\_GUID=@docguid;

end

Процедура завершения передачи:

create procedure dbo.DOCUMENT\_written

@key nvarchar(32),

@docguid uniqueidentifier

as

begin

set nocount on;

update a2doc.DOCUMENTS set D\_SENT=1 where D\_GUID=@docguid;

end

# ЗАГРУЗКА ЖУРНАЛОВ В ХРАНИЛИЩЕ ДАННЫХ

Далі буде...