Исследование возможности автоматической передачи данных между несколькими источниками данных с гибкой настройкой

Д.П. Молчанов, В.В. Баранюк

***Аннотация* – С развитием информационных технологий возрастают объемы обрабатываемых, анализируемых, используемых данных. В связи с этим все более актуальным становится вопрос хранения информации.**

Существует множество возможностей передавать, загружать и изменять данные: от использования рукописных команд до применения готовых ETL-систем (Extract, Transform, Load). Однако часто возникает проблема согласованности, при которой не всегда возможен постоянный контроль за появлением обновленных данных и их своевременной транспортировки. Одним из путей решения этой проблемы является создание программного средства по автоматизированной передаче данных между несколькими источниками с возможностью гибкой настройки.

Разрабатываемое программное средство должно поддерживать передачу данных между несколькими источниками и иметь возможность работать с разными диалектами – реализациями языка SQL в конкретных СУБД (MySQL, PostgreSQL, Oracle Database, MS SQL Server). Программное средство является кроссплатформенным; содержит встроенный пользовательский web-интерфейс; содержит встроенную базу данных для хранения параметров настройки системы и последних передаваемых в системе данных.

Использование программного средства автоматизированной передачи данных между несколькими источниками позволит пользователю задавать точное время для совершения копирования данных, временной интервал, через который новые данные будут передаваться до тех пор, пока процесс передачи или активное соединение не будут закрыты. Кроме этого, обеспечивается возможность персонального копирования данных из отдельных таблиц баз данных различной реализации, позволяя обслуживать их в автоматическом режиме с публикацией результатов проведения операций.

***Ключевые слова* – информационные технологии, хранение информации, базы данных, системы управления базами данных, программное средство автоматизированной передачи данных, источники данных.**

I. ВВЕДЕНИЕ

В современном мире информация играет ключевую роль. С ростом численности населения Земли и развитием информационных технологий, ее объемы быстро увеличиваются, а следовательно, операции по работе с информацией приобретают большую значимость и востребованность.

Статья получена 21.04.2021 г.
Д.П. Молчанов, МИРЭА (e-mail: idmitrymolchanov@gmail.com),
К.т.н., с.н.с., В.В. Баранюк, МИРЭА (e-mail: baranyuk@mirea.ru).

Накапливаемые данные необходимо где-то хранить и уметь их получать, сохранять, использовать. Хранение информации является одним из ключевых параметров в мире информационных технологий. Ни одна крупная корпоративная система не обходится без создания информационных баз в различных реализациях и, соответственно, нуждается в методах их взаимодействия.

С целью использования получаемой информации и ее систематизации были созданы базы данных. База данных представляет собой набор информации, которая хранится постоянно и которую спустя какое-то время обновляют и пополняют новыми данными.

Следовательно, возникает проблема своевременного получения обновленной информации, а также передачи данных между несколькими базами данных.

Существует множество возможностей передавать, загружать и изменять данные: от использования рукописных команд до применения готовых ETL-систем (Extract, Transform, Load). Методы, связанные с ручным выполнением, несут в себе существенные убытки по времени и провоцируют ситуации, связанные с большим возникновением ошибок, что негативно влияет и на экономическую составляющую проектов. Именно поэтому для повышения надежности работы систем целесообразно проводить автоматизацию вышеуказанных процессов.

В настоящее время практически невозможно создать корпоративную систему без использования баз данных, следовательно, востребована оптимизация и автоматизация проведения операций для работ с ними, а значит возрастает и спрос на готовые программные продукты, способные эти операции автономно осуществлять. Использование таких программных продуктов предоставляет множество преимуществ командам разработчиков, в том числе таких как:

– сокращение времени на согласование передачи данных;

– сокращение потенциальных ошибок, связанных, в первую очередь, с человеческим фактором, при переносе данных;

– обеспечение контроля за совершенными транзакциями;

– обеспечение гибкой настройки продукта для более широкого использования;

– обеспечение планового объединения данных в рамках единого проекта.

Однако основными проблемами многих решений являются их жесткость и длительность разработки. Зачастую программные продукты создаются «на местах» под нужды конкретной задачи в манипулировании данными. Следовательно, продукт нуждается во внутренней переработке, в изменении программного кода даже для выполнения аналогичной задачи, но с другими параметрами, что влечет дополнительные затраты. Поэтому целесообразна разработка гибкой системы, способной выполнять одну и ту же задачу, но предоставляющей возможность не привязываться к одной внутренней структуре.

II. Системы управления базами данных

Решение вопросов, связанных с переносом информации между базами данных, невозможно без привязки к используемым системам управления базами данных (СУБД).

«Жесткая зависимость между данными и использующими их приложениями создает серьезные проблемы в ведении данных и делает использования их менее гибкими» [1]. Одним из решений этой проблемы стало появление систем управления базами данных.

Все СУБД подчинены одной цели, но обычно выделяют три вида систем управления базами данных: простейшие (первое поколение баз данных), реляционные и нереляционные.

К первым относят сетевые и иерархические СУБД, привнесшие с собой и язык манипулирования данными, и их организацию в простых структурах, где определяющим фактором являлся рост отношений между хранимыми записями [1].

Третий тип, также называемый NoSQL базами данных – это группа типов баз данных, предлагающих подходы, отличные от стандартного реляционного шаблона. NoSQL предлагает способ структуризации данных, заключающийся в избавлении от ограничений при хранении и использовании информации [2].

В рамках настоящей статьи рассматривается только вторая группа, представляющая собой реляционные системы управления базами данных (РСУБД). Суть реляционного подхода состоит в требовании четких и ясных схем для работы с данными. Несмотря на заметный рост использования нереляционных систем, реляционный подход остается очень востребованным. Во-первых, из-за удобства и простоты в использовании, во-вторых, что даже существеннее, из-за огромного количества существующих проектов и решений, в-третьих – по причине удобного описания хранимых сущностей, где каждый создаваемый объект может быть выражен объектом реального мира.

Однако, существуют минусы, которые связаны не с самими системами, а с проблемами, возникающими при их интеграции и использовании. В отличие от моделей NoSQL, операции по передаче данных в такие системы требуют больших усилий от оператора, а следовательно, других средств автоматизации. Для структурированного языка запросов SQL существуют разные стандарты, например, СУБД Microsoft SQL основан на Transact-SQL, расширяющем стандарт SQL2 [3]. Помимо множества стандартов также существуют отличия и диалектов – реализаций языка SQL в конкретной СУБД, которые имеют различные возможности для обработки данных и процедурные настройки. Так как в настоящее время не существует ни одного диалекта, полностью соответствующего выбранному стандарту, и не бывает двух идентичных диалектов, это, помимо фактора ручного проведения операций, может прибавлять ошибок в процессе использования СУБД [4].

Разрабатываемое программное средство по автоматизированной передаче данных между несколькими источниками направленно на работу с РСУБД, при его создании делается упор не только на автоматизацию работы с данными, но и на унификацию основных диалектов версий поставщиков систем управления базами данных.

III. Цели создания программного средства

Нередко при работе в крупной организации разработчики сталкиваются с проблемой доступа к данным из разных подразделений. Речь идет о разных источниках в вопросах объединения собранной информации с целью создания общего проекта, над которым трудятся разные команды. Зачастую, объединение информации из этих источников должно сопровождаться предварительной договоренностью и ручным переносом, на что тоже затрачивается время. Но главное, возникает проблема согласованности, при которой не всегда возможен постоянный контроль за появлением обновленных данных и их своевременной транспортировки. Это приводит к задержкам разработки и более высоким финансовым вложениям. В связи с этим появляется необходимость автоматизации работы с данными, организованными для нескольких источников.

Можно выделить основные цели создания программного средства, обеспечивающего автоматизированную передачу данных между несколькими источниками:

– сокращение временных задержек при передаче данных;

– уменьшение материальных и временных затрат на проведение автоматизируемых операций;

– повышение качества проведения операций, минимизация и исключение ошибок, связанных с человеческим фактором при проведении операций;

– предоставление более дружественных пользователю возможностей контроля за работой системы.

IV. Существующие решения

В настоящее время уже существуют популярные программные продукты для частичного осуществления поставленных целей.

В первую очередь, это проекты жестких решений – ETL-системы, созданные с помощью технологий Spring Boot, Spring Cloud и т. д. Использование подобных проектов предполагает решение одной четко поставленной задачи в узких рамках разработки.

Целью инструмента миграции базы данных с открытым исходным кодом pgloader является упрощение процесса миграции на PostgreSQL. Он поддерживает миграцию из нескольких типов файлов и РСУБД, включая MySQL и SQLite в PostgreSQL. Данный инструмент представляет собой легкий способ переноса данных из одной БД в другую [5].

Как правило, в пределах одной системы, но в разных базах данных существуют стандартные средства переноса данных. Основой для одной из таких технологи является метод Log Shipping, поддерживаемый Microsoft SQL Server, MySQL и PostgreSQL. Суть Log Shipping состоит в передаче копий журналов транзакций с одного сервера на другой, что при определенных дополнениях позволяет создать и поддерживать в актуальном состоянии резервный сервер. При этом нужно помнить, что данные на резервный сервер попадают не сразу, а через некоторый промежуток времени, зависящий от настроек, и нельзя заставить работать Log Shipping в непрерывном режиме, как например, репликацию [6].

Транзакционная репликация используется для копирования данных в режиме реального времени с возможностью репликации отдельных объектов. К плюсам использования можно отнести производительность и масштабируемость**,** отказоустойчивость и резервирование данных. Профессиональным программным решением, осуществляющим репликацию данных между несколькими источниками в автоматическом режиме, является программа для резервного копирования баз данных Handy Backup. При работе для бэкапа базы используется универсальный плагин Database, способный работать как с SQL, так и с NoSQL ориентированными СУБД [7].

Разрабатываемое программное средство по внутренней структуре ориентируется на проекты жесткой реализации, которые показывают хорошую надежность, гарантированно справляются с поставленной задачей. Но такие системы оказываются совершенно непригодными для малейших их изменений, ибо для каждого нового требования приходится заново пересобирать программный продукт. Следовательно, требуется разработать программное средство по автоматизированной передаче данных между несколькими источниками с возможностью гибкой настройки.

V. Требования к программному средству

В процессе формирования технического задания на разработку программного средства автоматизированной передачи данных между несколькими источниками определены требования к нему. Указанное программное средство должно:

– быть реализовано в виде исполняемого файла (war/jar) и быть кроссплатформенным;

– содержать встроенный пользовательский web-интерфейс для настройки и контроля процесса передачи данных и формирования отчетностей;

– содержать встроенную базу данных для хранения параметров настройки системы и последних передаваемых в системе данных;

– предоставлять пользователю базовый набор операций ETL-систем: загрузка, преобразование, выгрузка данных;

– иметь возможность отображать успешность/неуспешность проведения операций;

– быть доступно для развертывания в docker-контейнере для участия в общей архитектуре проекта. Для этого должна обеспечиваться возможность подключения посредством JNDI (Java Naming and Directory Interface).

Для реализации гибкой настройки целесообразно использовать пользовательский интерфейс. Таким образом можно обойтись без последующего программного вмешательства в продукт, а задавать специфику передачи данных посредством выполнения команд оператора в графической оболочке, представленной веб-сервисом в локальной сети.

VI. Возможности программного средства автоматизированной передачи данных между несколькими источниками

В программном средстве по автоматизированной передаче данных между несколькими источниками реализуется следующий функционал:

– получение конфигурационных данных от пользователя по источникам данных;

– добавление конфигурационных данных во встроенную базу данных приложения;

– получение данных от пользователя по частоте отслеживания изменений в базе данных-источнике;

– осуществление выборки записей по указанному параметру, либо по всей совокупности полей по срабатыванию расписания;

– автономная передача данных из баз данных источников в базы данных назначения;

– вывод пользователю активных процессов, параметров процессов, отчетов по проведенным транзакционным операциям;

– предоставление возможности остановки процесса, удаления конфигураций, просмотра реализованных операций.

Помимо основного функционала, рассчитанного на непосредственную работу с данными, не менее важную роль играет фактор кастомизируемости. Программное средство обеспечивает возможность внесения изменений, позволяет пользователю производить настройки под решение конкретной задачи, но в рамках определенного класса. В данном случае кастомизируемость можно трактовать как гибкость и возможность изменять источник данных для переноса информации в другие базы данных.

Осуществимость автономной работы программы обуславливается использованием фреймворка Spring Boot. Для получения возможности динамически устанавливать соединения с базами данных во время выполнения программы был реализован метод,

позволяющий перезагрузить главный интерфейс в Spring-приложении – ApplicationContext. Связь между веб-интерфейсом и основной программой осуществляется посредством протокола RESTful API. Интерфейс для осуществления настройки программы доступен посредством веб-страницы на локальном сервере. Базы данных, участвующие в обмене, могут быть как локальными, так и глобальными.

Программное средство поддерживает передачу данных между несколькими разными источниками и способно работать с основными диалектами: MySQL, PostgreSQL, Oracle Database, MS SQL Server.

Программное средство содержит встраиваемую базу данных и web-интерфейс для осуществления основного алгоритма функционирования. В качестве интегрированной СУБД используется SQLite. База данных в приложении требуется в основном для хранения данных, вводимых пользователем в веб-интерфейсе, для обеспечения их сохранности и получения точных настроек для выполнения дальнейших операций. К примеру, когда пользователю нужно выгружать данные не из всех полей выбранной таблицы, он помечает нужные при настройке, а выбранные им пункты будут сохранены во внутренней БД, к чему и обратится система при непосредственном осуществлении передачи данных. После получения всех необходимых записей при настройке запускается основной алгоритм, осуществляющий автономную работу, для чего сначала выполняется поиск активных соединений, в которых указаны ресурс-источник и конечный ресурс. После определения идентификатора соединения проверяются заполненные связи таблиц, содержимое которых необходимо переносить. И первый, и второй этап добавляются в два вложенных друг в друга пула, по которым двигается программа. Первый содержит список подключений баз, второй – таблиц для каждой базы, пересоздаваясь с каждой итерацией по первому. При непосредственной выгрузке проверяются последние загруженные данные в конечном ресурсе, производится их поиск в источнике и определяется наличие новых записей, в случае подтверждения которых и проводится операция копирования. По указываемым в конфигурациях данным определяется требуемый диалект, который и используется для получения-внесения записей. Следовательно, осуществляется возможность копирования содержимого отдельных таблиц, задаваемых пользователем, получаемых из нескольких подключений. Вся настройка процесса определяется в присутствующем в системе web-интерфейсе, где пользователь способен задавать строки конфигураций источников, соединения, наименования таблиц, их полей, частоту передачи, время и дату первой транзакции.

Таким образом, разработано программное средство, реализующее ряд положительных возможностей, перечисленных выше аналогов и имеющее дополнительные особенности и достоинства.

Заключение

Использование программного средства автоматизированной передачи данных между несколькими источниками позволит пользователю задавать точное время для совершения копирования данных, временной интервал, через который новые данные будут передаваться до тех пор, пока процесс передачи или активное соединение не будут закрыты. В настройках системы пользователь сможет выбрать нужные ему таблицы, а в них требуемые поля, указать по какому из них следует определять новые записи и добавить для выбранных простую фильтрацию.

Кроме этого, обеспечивается возможность персонального копирования данных из отдельных таблиц баз данных различной реализации, позволяя обслуживать их в автоматическом режиме с публикацией результатов проведения операций.

Библиография

[1] Зрюмов Е. А., Зрюмова А. Г., «Базы данных для инженеров». – Барнаул: Изд-во АлтГТУ, 2010. – С. 8-11.

[2] SQL и NoSQL, URL: https://tproger.ru/translations/sql-nosql-database-models/ (дата обращения: 01.03.2021).

[3] Стандарты языка SQL, URL: http://www.wiscorp.com/SQLStandards.html (дата обращения: 02.04.2021).

[4] Диалекты языка SQL в СУБД, URL: http://www.bseu.by/it/tohod/lekcii7_4.htm (дата обращения: 03.04.2021).

[5] pgloader’s documentation, URL: https://pgloader.readthedocs.io/en/latest/ (дата обращения: 27.03.2021).

[6] Как работает Log Shipping, URL: https://www.sql.ru/articles/mssql/2005/073001logshipping.shtml (дата обращения: 02.04.2021).

[7] Утилита для бэкапа баз данных Handy Backup, URL: https://www.handybackup.ru/ (дата обращения: 04.04.2021).

Investigation of the possibility of automatic data transfer between several data sources with flexible configuration

D.P. Molchanov, V.V. Baranyuk

Abstract - **The volumes of processed, analyzed and used data are increasing with the development of information technologies. In this regard, the issue of storing information is becoming more and more urgent.**

There are many possibilities for transferring, loading and modifying data, from using handwritten commands to using off-the-shelf ETL systems (Extract, Transform, Load). However, there is often a problem of consistency, in which it is not always possible to constantly monitor the appearance of updated data and their timely transportation. One of the ways to solve this problem is to create a software tool for the automated transfer of data between several sources with the possibility of flexible configuration.

The developed software tool should support the transfer of data between several different sources and be able to work with different dialects - implementations of the SQL language in specific DBMS (MySQL, PostgreSQL, Oracle Database, MS SQL Server). The software is cross-platform; contains a built-in web user interface; contains a built-in database for storing system settings and the last data transferred in the system.

Using a software tool for automated data transfer between multiple sources will allow the user to set the exact time for copying data, the time interval through which new data will be transmitted until the transfer process or an active connection is closed. In addition, the possibility of personal copying of data from individual tables of databases of various implementations is provided, allowing it to be serviced in automatic mode with the publication of the results of operations.

Keywords - **information technology, information storage, databases, database management systems, software for automated data transmission, data sources.**

REFERENCES

[1] Zrjumov E. A., Zrjumova A. G., «Bazy dannyh dlja inzhenerov». – Barnaul: Izd-vo AltGTU, 2010. – S. 8-11.

[2] SQL i NoSQL, URL: https://tproger.ru/translations/sql-nosql-database-models/ (data obrashhenija: 01.03.2021).

[3] Standarty jazyka SQL, URL: http://www.wiscorp.com/SQLStandards.html (data obrashhenija: 02.04.2021).

[4] Dialekty jazyka SQL v SUBD, URL: http://www.bseu.by/it/tohod/lekcii7_4.htm (data obrashhenija: 03.04.2021).

[5] pgloader's documentation, URL: https://pgloader.readthedocs.io/en/latest/ (data obrashhenija: 27.03.2021).

[6] Kak rabotaet Log Shipping, URL: https://www.sql.ru/articles/mssql/2005/073001logshipping.shtml (data obrashhenija: 02.04.2021).

[7] Utilita dlja bjekapa baz dannyh Handy Backup, URL: https://www.handybackup.ru/ (data obrashhenija: 04.04.2021).