электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 004.658.2

СРАВНИТЕЛЬНЫЙ АНАЛИЗ МЕТОДОВ ИЗВЛЕЧЕНИЯ ДАННЫХ ИЗ ИСТОЧНИКОВ ДАННЫХ ИНФОРМАЦИОННОЙ СИСТЕМЫ ПРИМЕНИТЕЛЬНО К ЗАДАЧЕ АГРЕГАЦИИ ДАННЫХ

Саврасов С.Б., *студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана
кафедра «Системы обработки информации и управления»

Научный руководитель: Ревунков Г. И., к.т.н., доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана
savrasovs@bmstu.ru

Введение:

При проектировании распределенной информационной системы перед разработчиками встает задача выбора оптимального способа передачи данных между элементами системы. Частным случаем задачи передачи данных является задача агрегации данных. Под агрегацией данных подразумевается процесс объединения данных из различных источников данных системы в её подсистеме. Для реализации данного процесса требуется подобрать наиболее подходящий способ извлечения данных из источников данных и технические средства для его реализации. Источники данных в системе могут быть представлены как простой СУБД, так и более сложным программным комплексом. В работе рассматривается реализация агрегации из источников, представленных простой СУБД.

Для реализации автоматизированной агрегации данных требуется реализовать следующие задачи:

1. Установка соединения с СУБД и открытие баз данных для работы.
2. Выполнение запросов к базам данных на соответствующем языке запросов и извлечение данных из баз.
3. Объединение данных в соответствии с условием поставленной задачи.

Каждая из данных задач может быть решена различными способами. Различные Комбинации вариантов решения данных задач формируют множество *методов извлечения данных*. В работе рассмотрены четыре метода извлечения данных:

1. Использование средств языка программирования.

2. Использование расширенных средств СУБД.
3. Использование ODBC-драйверов.
4. Использование веб-сервисов.

Рассмотрим каждый из методов, применительно к решению задачи агрегации данных, более подробно.

1. Метод «Использование средств языка программирования».

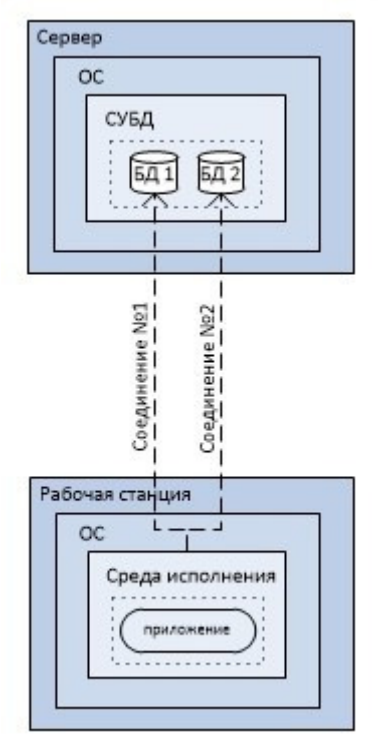

Рис. 1. Метод извлечения данных из СУБД — «Использование средств языка программирования»

Предположим, что данные для агрегации располагаются в различных базах данных одной реляционной СУБД на удаленной рабочей станции (сервере). Для того чтобы произвести агрегацию потребуется установить соединение с СУБД, в определенной последовательности выбирать для работы конкретную БД и извлекать из нее данные, сохраняя их на стороне клиента. Затем закрыть соединение с СУБД и завершить агрегацию путем обработки данных программными средствами.

Рассмотрим каждый из этапов более подробно. Подключение к СУБД в различных языках программирования реализовано по-разному. Если для реализации мы используем современные объектно-ориентированные языки, такие как: PHP, JAVA, C#, то подключение реализуется путем создания экземпляра класса, обеспечивающего данное подключение. В указанных языках данный механизм реализован по-разному, но общий принцип одинаков для всех. Для создания объекта конструктору передаются параметры соединения: host – адрес сервера, port – порт соединения, user – имя учетной записи пользователя с правами доступа к СУБД и к конкретной БД, pass – пароль для учетной записи, dbname – имя базы данных для работы. После создания объекта, соединение устанавливается либо сразу, либо требуется вызвать соответствующий метод. Если соединение установлено, то можно перейти к следующему этапу [1].

Извлечение данных происходит путем выполнения соответствующего метода или функции, применительно к установленному соединению с передачей ему запроса на соответствующем языке запросов. Сервер принимает запрос, анализирует его. Если запрос составлен правильно и при его выполнении не возникает ошибки, сервер возвращает данные.

Поскольку агрегируемые данные находятся в различных базах данных, а для каждой БД требуются отличные права доступа, процедуру подключения к СУБД и проведение запросов требуется повторять многократно.

Извлечение данных из разных БД будет происходить последовательно в одном соединении. Агрегация данных займет относительно много времени, если учесть тот факт, что после последовательного извлечения данных завершение их агрегации (обработки) будет производиться средствами языка на стороне клиента. Полученные данные будут представлены соответствующими объектами языка, а, следовательно, увеличится расход памяти.

При возможности установки нескольких параллельных соединений с одной СУБД и выбор для работы сразу нескольких БД запросы можно проводить параллельно, т.е. независимо друг от друга. После получения данных необходимо закрыть соединения и выполнить обработку данных на стороне клиента средствами языка программирования. Нужно иметь ввиду, что если занять и не использовать установленные соединения, то остальные пользователи не смогут работать с СУБД. Данная ситуация критична для СУБД поддерживающих малое количество соединений. В реальной рабочей ситуации (для работы клиенту) обычно доступно только одно соединение.

Достоинства:

1) Метод имеет простую реализацию.

2) Для реализации используются встроенные средства языка, разработчик свободен в выборе инструмента, подходящего для имеющейся СУБД.

Недостатки:

1) Если запросы выполняются в параллельных соединениях, то нужно следить за количеством открытых соединений.

2) Нужно принимать во внимание, что у некоторых СУБД ограничено количество соединений и, если занять и не использовать часть из них, то остальные пользователи не смогут работать с СУБД.

3) При последовательном выполнении запросов в рамках одного соединения процесс агрегации данных замедляется.

Итог:

Данный метод применим в случае, если баз данных немного, а время сбора данных не критично и количество запросов также ограничено.

Вывод:

Требуется метод, уменьшающий количество соединений и запросов. Таким методом является метод «Использование расширенных средств СУБД».

2. Метод «Использование расширенных средств СУБД».

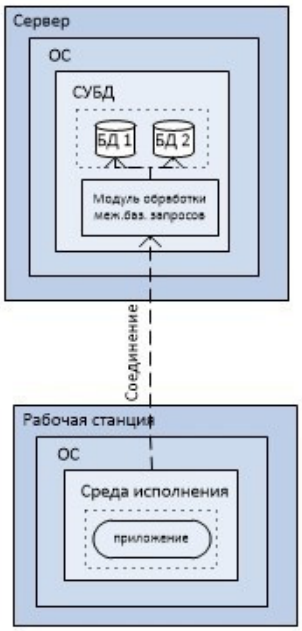

Рис. 2. Метод извлечения данных из СУБД — «Использование расширенных средств СУБД»

Некоторые СУБД допускают расширение своих функциональных возможностей путем установки определенных модулей-расширений. Суть метода заключается в установке и настройке расширения СУБД, которое позволит пользователю устанавливать соединение с СУБД и с использованием расширенного синтаксиса SQL в одном запросе задействовать данные из нескольких баз данных СУБД. То есть с данным модулем СУБД обладает расширенной возможностью принимать SQL запрос, в котором будет задействована не одна, а сразу несколько БД. Пример: модуль dblink для СУБД «PostgreSQL» [2].

По сравнению с методом «Использование средств языка программирования» мы значительно сокращаем количество открытых соединений с СУБД для выполнения агрегации. Если базы содержатся в одной СУБД, то для выполнения агрегации может понадобиться только одно соединение и возможно один запрос. В данном случае агрегация данных будет происходить на стороне сервера, что значительно ускорит выполнение данного процесса, поскольку с данными будет работать СУБД. Обработка данных на стороне клиента предполагает работу с объектами, что при большом объеме данных потребует большого объема памяти и вычислительных мощностей системы. Механизм работы современных СУБД за десятилетия был доведен до совершенства и является максимально быстрым и оптимальным для работы с реляционными данными, поэтому, выполняя агрегацию на стороне сервера, мы получим значительный выигрыш в скорости.

Однако встречаются ситуации, в которых данный подход будет неприемлем. Данные для агрегации могут содержаться в разных БД и при этом могут быть представлены в разных форматах или же иметь различные структуры, неудобные для проведения агрегации в одном запросе. Для проведения агрегации будет требоваться выполнение промежуточного преобразования, невозможного для реализации в рамках SQL запроса.

Может возникнуть ситуация, когда данные для агрегации будут расположены в базах данных, находящихся на различных серверах. Если модуль позволит проводить межбазовые запросы в пределах одной СУБД, то возможность связи с базами на другом сервере может отсутствовать.

Также нужно учесть, что не все современные СУБД обладают такими расширениями, то есть поддерживают соответствующий функционал. К тому же, как уже было сказано, могут возникнуть различные технологические ограничения, например невозможность транзитивно обратиться к базе на удаленном сервере.

Метод не может быть применен в случае, если требуется провести агрегацию данных из разнотипных СУБД (реляционных, постреляционных). Несмотря на то, что многие постреляционные (объектные) СУБД позволяют работать с данными в рамках реляционной модели (пример Cache`), использование подобной надстройки для связи с реляционной базой данных может просто не существовать.

Достоинства:

1) Сокращение количества открытых соединений с СУБД для выполнения агрегации данных.

2) Значительное ускорение выполнения агрегации за счет проведения агрегации на стороне сервера средствами СУБД.

Недостатки:

1) Метод становится непригоден в случае, если данные в разных БД содержатся в форматах или имеют структуры, неудобные для проведения агрегации в одном запросе.

2) Не все современные СУБД поддерживают данный функционал.

3) Отсутствует возможность транзитивного обращения к двум базам на различных серверах в одном запросе.

4) Неприменим для агрегации данных из разнотипных СУБД (реляционные, постреляционные - объектные)

Итог:

Применяя метод, мы выигрываем, уменьшая количество соединений, но становимся зависимыми от выбора СУБД. Также возникают проблемы, если СУБД отличаются по типу хранения данных и находятся на различных серверах.

Вывод:

Метод дает ограничение в выборе СУБД: если СУБД не поддерживает данную технологию или же нужно выполнить агрегацию данных из разнотипных СУБД, то требуется другой метод. Таким методом является метод «Использование ODBC-драйверов».

3. Метод «Использование ODBC-драйверов».

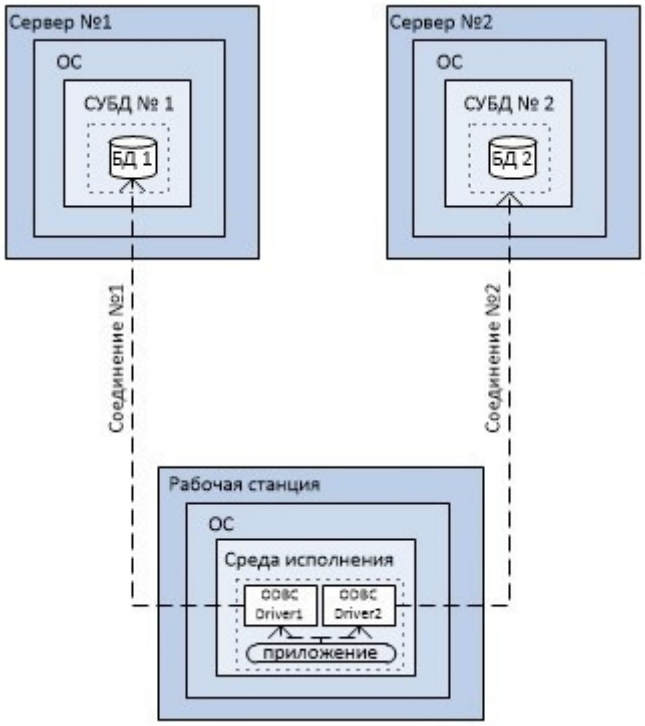

Рис. 3. Метод извлечения данных из СУБД — «Использование ODBC-драйверов»

Каждая СУБД, разработанная определенной компанией, имеет свои архитектурные особенности. Несмотря на то, что рассматриваемые БД могут относиться к классу реляционных, даже в пределах одного класса различия могут быть огромными. Простым примером являются различия в синтаксисе языков запросов, не считая интерфейсы взаимодействия и различные функциональные особенности каждой из СУБД. Изучение уникальных особенностей СУБД требует много времени и для упрощения задачи можно воспользоваться технологией ODBC - Object Database Connectivity — Общий интерфейс взаимодействия [3].

Суть метода заключается в использовании ODBC-драйверов для соответствующей СУБД. При использовании ODBC становится возможным проводить запросы в некоторой промежуточной среде, условно говоря, прослойке внутри языка. У различных СУБД можно выделить общий интерфейс и работать с ним. При использовании ODBC работа с СУБД сводится к работе с абстрактным источником, реализующим набор ясных и доступных возможностей. Пример: ODBC или JDBC.

Однако в случае, как и с модулем-надстройкой над СУБД в методе «Использование расширенных средств СУБД», не для всех баз данных существуют ODBC-драйвера, поддерживаемые соответствующими средами разработки. Кроме того, используя такое промежуточное решение, мы ограничиваем себя в средствах разработки.

В зависимости от платформы появляется возможность экспериментировать с выбором языков реализации, но приходится пользоваться только теми виртуальными классами, которые предоставляет виртуальная машина, мы ограничены этой платформой.

Источники могут быть разнотипными (реляционными, постреляционными) и располагаться на разных серверах, но агрегация будет производиться на стороне клиента. Извлечение данных происходит из каждого источника по отдельности.

Достоинства:

1) Агрегация данных может быть выполнена из разнотипных СУБД, расположенных на разных серверах.

2) Сокращается время изучения работы интерфейсов СУБД.

Недостатки:

1) Не для всех баз данных существуют ODBC — драйвера (интерфейсы).

2) Появляются ограничения в использовании средств разработки.

3) Агрегация данных происходит на стороне клиента.

Итог:

Способ позволяет сократить время изучения работы интерфейсов СУБД, но ограничивает нас в использовании языков. Не все языки могут поддерживать ODBC-драйвера для определенных СУБД.

Агрегация данных из разнотипных СУБД становится возможной, но как быть, если нам требуется для обработки данных определенный функционал среды или языка, которые в свою очередь не поддерживают соответствующий ODBC драйвер?

Вывод:

Нужен метод, который позволит произвести агрегацию из разнотипных СУБД и использовать максимальное количество комбинаций языков и СУБД. Таким методом является метод «Использование веб-сервисов».

4. Метод «Использование веб-сервисов».

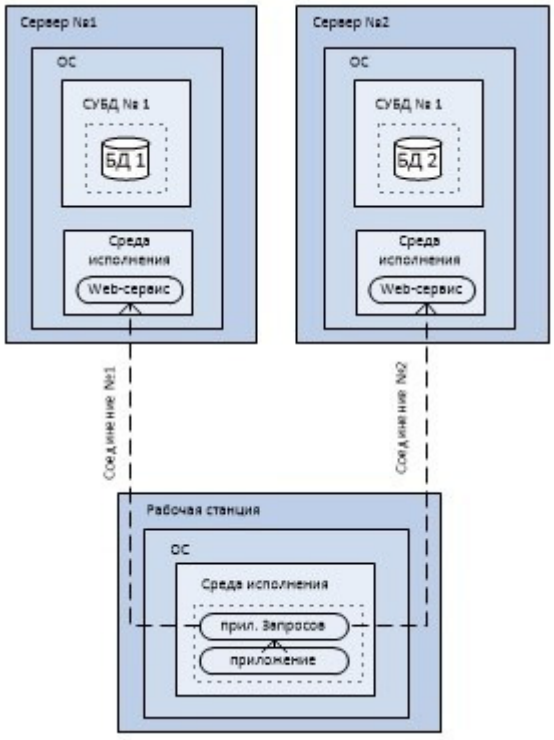

Рис. 4. Метод извлечения данных из СУБД — «Использование веб-сервисов»

Суть метода заключается в присоединении к каждому источнику данных (СУБД) программной системы со стандартизированным интерфейсом и идентифицируемой веб-адресом. Данная программная система называется веб-сервисом. Веб-сервисы могут взаимодействовать друг с другом и со сторонними приложениями посредством сообщений, основанных на определенных протоколах (SOAP, XML-RPC и т. д).

То есть к каждому источнику данных (СУБД на сервере) добавляется приложение, реализующее работу веб-сервиса, тем самым превращая реляционный источник данных в XML-источник данных, который принимает HTTP-запросы и возвращает данные в XML-формате. Если все источники данных системы будут реализованы с использованием веб-сервисов, то передача данных между модулями будет происходить в едином формате — XML, а веб-сервис, в свою очередь, будет написан на языке, который позволит лучшим образом извлекать данные для их отправки.

К положительным особенностям данного метода можно отнести возможность установки ограничений доступа к данным для определенных групп пользователей или определенных типов запросов. Данное ограничение нельзя наложить средствами СУБД,

используя права доступа к БД. Пример: данные содержатся в одной базе данных, но в разных таблицах. К одним таблицам доступ должен быть открыт, к другим закрыт.

На стыке между приложением, реализующим веб-сервис и СУБД, запросы из приложения можно будет писать на любом подходящем языке, располагающим нужным функционалом для работы с СУБД, так как на сегодняшний день поддержка XML обеспечена практически во всех современных ООП языках общего назначения: C#, JAVA, PHP, Phython, Delphi и др.

Достоинства:

1) Запросы к источнику данных можно выполнять через программы, написанные на любом языке, так как поддержка XML обеспечена во всех современных ООП языках.

2) Метод обеспечивает «гибкость» и «масштабируемость»: источники данных могут быть распределены на большом количестве серверов. Веб-сервис является единицей модульности при использовании сервис-ориентированных технологий [4].

3) Метод обеспечивает «кроссплатформенность»— независимость от конкретной операционной системы, подразумевающая возможность подобрать ПО для реализации системы на любой (популярной) ОС.

Недостатки:

Агрегация производится на стороне клиента.

Итог:

Основная идея заключается в добавлении к каждому источнику данных веб-сервиса, позволяя тем самым преобразовать его в XML-источник данных, что обеспечит единый формат передачи данных в системе. Это предоставит огромный выбор программных средств для реализации запросов. Система становится «гибкой» и «масштабируемой», появляется «кроссплатформенность» - реализация становится независимой от конкретной операционной системы. Агрегацию данных можно будет производить с разнотипных СУБД и не испытывать проблем с подбором технических средств для реализации запросов.

Вывод:

Метод «Использование веб-сервисов» является оптимальным методом для выполнения запросов и передачи данных между системами, а, следовательно, выполнения агрегации данных в данной системе.

Заключение:

В работе рассмотрены четыре типа методов реализации передачи данных между различными источниками данных информационной системы применительно к задачи агрегации данных.

Было приведено подробное описание каждого из методов с выделением достоинств и недостатков каждого из них. На основе проведенного анализа сделаны соответствующие выводы о применимости методов в конкретных условиях, зависящих от множества факторов, таких как количество доступных параллельных соединений с СУБД, распределение данных между источниками данных, типы источников данных, завершение агрегации на стороне клиента или сервера и т. п.

Все четыре метода рассмотрены в определенном порядке, в котором каждый последующий метод логически следует из предыдущего. Самым эффективным методом, не имеющим существенных недостатков, оказался последний из рассмотренных - метод «Использование веб-сервисов».

Список литературы

1. Работа с базой данных на PHP на примере MySQL//codingcraft.ru: Школа программирования coding-craft. URL.http://codingcraft.ru/php/mysql.php (дата обращения 13.04.2013)
2. Реляционныя БД: PostgreSQL — дополнение dblink//doc.prototypes.ru: Статьи по веб-разработке. URL.http://doc.prototypes.ru/database/postgresql/contrib/dblink (дата обращения 14.04.2013)
3. ODBC — общие сведения о подключениях открытой базы данных//support.microsoft.com: Служба поддержки Microsoft. URL.http://support.microsoft.com/kb/110093 (дата обращения: 14.04.2013)
4. «Веб-служба» - Википедия//ru.wikipedia.org: Википедия – свободная энциклопедия. URL.http://ru.wikipedia.org/wiki/Web-сервис (дата обращения:16.04.2013).