Отчет по практике: Разработка приложений архитектуры клиент-сервер при помощи SQL

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

На протяжении последних десяти лет специалисты по вычислительной технике работают над усовершенствованием приложений клиент-сервер. В результате были построены

приложения, поддерживающие совместную работу множества пользователей с единственным источником данных в сети.

Архитектура клиент-сервер стала общераспространенной при общении с компьютером или с системой на его основе. Любой человек, подключающийся к диалоговой информационной системе с помощью телефонной связи, использует архитектуру клиент-сервер. Пользуясь автоматическим кассовым аппаратом, считывая штриховые коды своих покупок на проверочном устройстве магазина или расплачиваясь за них с помощью кредитной карточки, идет взаимодействие с компьютерной системой клиент-сервер.

Целью курсовой работы является рассмотрение структурированного языка запросов SQL, при помощи которого разрабатываются базы данных для системы клиент-сервер.

Задачами курсовой работы является рассмотрение:

архитектуры информационной системы, и в частности клиент-сервер;

языков запросов SQL и QBE, и их сравнение;

принципов разработки приложений архитектуры клиент-сервер при помощи SQL.

Система клиент-сервер является наиболее перспективной, так как поддерживает большое число пользователей и сложные приложения, кроме этого она обладает высоким уровнем защиты информации, за счет среды программирования SQLServeri все данные и прикладные средства хранятся централизованно, то есть, сосредоточены в одном месте.

1. АРХИТЕКТУРА ИНФОРМАЦИОННОЙ СИСТЕМЫ

Эффективность функционирования информационной системы во многом зависит от ее архитектуры. В настоящее время перспективной является архитектура клиент-сервер. В достаточно распространенном варианте она предполагает наличие компьютерной сети и распределенной базы данных, включающей корпоративную базу данных (КБД) и персональные базы данных (ПБД). КБД размещается на компьютере-сервере, ПБД размещаются на компьютерах сотрудников подразделений, являющихся клиентами корпоративной базы данных.

1.1 Архитектура файл-сервер

Самой простой архитектурой для реализации является архитектура «файл-сервер» (рисунок 1), но она же обладает и самым большим количеством недостатков, ограничивающих спектр решаемых ею задач. Простейшим случаем является случай, когда данные располагаются физически на том же компьютере, что и само приложение.

/>

Рисунок 1 Структура информационной системы с файл-сервером

К существенным неудобствам, возникающим при работе с системой, построенной по такой архитектуре, можно отнести следующее:

— трудности при обеспечении непротиворечивости и целостности данных;

— существенная загрузка локальной сети передаваемыми данными;

— в целом, невысокая скорость обработки и представления информации;

— высокие требования к ресурсам компьютеров. При этом возникают следующие ограничения.

— невозможность организации равноправного одновременного доступа; пользователей к одному и тому же участку базы данных;

— количество одновременно работающих с системой пользователей не превышает пяти человек для ЛВС, построенной в соответствии со спецификацией 1 OBaseT(скорость обмена данными до 10Мб/с);

При всем этом система обладает одним очень важным преимуществом — низкой стоимостью.

Архитектура «файл-сервер» предусматривает концентрацию обработки на рабочих станциях. Основным преимуществом этого варианта является простота и относительная дешевизна. Подобное решение приемлемо, пока число пользователей, одновременно работающих с базой данных, не превышает 5-10 человек. При увеличении количества пользователей система может «захлебнуться» из-за перегруженности ЛВС большими потоками необработанной информации.

Сервер, как правило, — самый мощный и самый надежный компьютер. Он обязательно подключается через источник бесперебойного питания, в нем предусматриваются системы двойного или даже тройного дублирования. В особо ответственных случаях можно подключить вместе несколько серверов так, что при выходе из строя одного из них в работу автоматически включится «дублер». Таким образом, при концентрации обработки данных на сервере надежность системы в целом ограничивается только материальными средствами, которые заказчики готовы вложить в техническое оснащение.

Решение по автоматизации учета и управления в корпоративных структурах предполагает распределенную обработку данных, организацию параллельных вычислений, глубокое разграничение уровней доступа, возможность выбора различных операционных систем и серверных платформ. Если бизнес не велик, подобное решение оптимально.

В ходе эксплуатации были выявлены общие недостатки файл-серверного подхода при обеспечении многопользовательского доступа к базе данных.

Вся тяжесть вычислительной нагрузки при доступе к базе данных ложится на приложение клиента, что является следствием принципа обработки информации в системах «файл-сервер»: при выдаче запроса на выборку информации из таблицы вся таблица базы данных копируется на клиентское место, и выборка осуществляется на клиентском месте. Локальные СУБД используют так называемый «навигационный подход», ориентированный на работу с отдельными записями.

Не оптимально расходуются ресурсы клиентского компьютера и сети; например, если в результате запроса мы должны получить 2 записи из таблицы объемом 10000 записей, все 10000 записей будут скопированы с файл-сервера на клиентский компьютер; в результате возрастает сетевой трафик и увеличиваются требования к аппаратным мощностям пользовательского компьютера.

В базе данных на файл-сервере гораздо проще вносить изменения в отдельные таблицы, минуя приложения. Эта возможность облегчается тем обстоятельством, что у локальных СУБД база данных — понятие более логическое, чем физическое, поскольку под базой данных понимается набор отдельных таблиц, сосуществующих в едином каталоге

на диске. Все это позволяет говорить о низком уровне безопасности — как с точки зрения хищения и нанесения вреда, так и с точки зрения внесения ошибочных изменений.

Недостаточно развитый аппарат транзакций для локальных СУБД служит потенциальным источником ошибок как с точки зрения одновременного внесения изменений в одну и ту же запись, так и с точки зрения отката результатов серий объединенных по смыслу в единое целое операций над базой, когда некоторые из них завершились неуспешно, а некоторые — нет; это может нарушать ссылочную и смысловую целостность базы данных.

Недостатки настольных СУБД обычно проявляются не сразу, а лишь в процессе длительной эксплуатации, когда объем хранимых данных и число пользователей становятся достаточно велики — это приводит к снижению производительности приложений, использующих такие СУБД.

Поскольку настольные СУБД не содержат специальных приложений и сервисов, управляющих данными, а используются для этой цели файловые сервисы операционной системы, вся реальная обработка данных в таких СУБД осуществляется в клиентском приложении, и любые библиотеки доступа к данным в этом случае также находятся в адресном пространстве клиентского приложения. Поэтому при выполнении запросов данные, на основании которых выполняется такой запрос, должны быть доставлены в то же самое адресное пространство клиентского приложения. Это и приводит к перегрузке сети при увеличении числа пользователей и объема данных, а также грозит иными неприятными последствиями, например разрушением индексов и таблиц. Недаром до сих пор популярны утилиты для «ремонта» испорченных файлов настольных СУБД.

Недостатки архитектуры «файл-сервер» решаются при переводе приложений в архитектуру «клиент-сервер», которая знаменует собой следующий этап в развитии СУБД. Характерной особенностью архитектуры «клиент-сервер» является перенос вычислительной нагрузки на сервер базы данных (SQL-сервер) и максимальная разгрузка приложения клиента от вычислительной работы, а также существенное укрепление безопасности данных — как от злонамеренных, так и просто ошибочных изменений.

БД в этом случае помещается на сетевом сервере, как и в архитектуре «файл-сервер», однако прямого доступа к базе данных (БД) из приложений не происходит. Функция прямого обращения к БД осуществляет специальная управляющая программа — сервер БД (SQL-сервер), поставляемый разработчиком СУБД.

--PAGE_BREAK--

1.2 Архитектура клиент-сервер

Сервером определенного ресурса в компьютерной сети называется компьютер (программа), управляющая этим ресурсом, клиентом — компьютер (программа), использующий этот ресурс. В качестве ресурса компьютерной сети могут выступать, базы данных, файловые системы, службы печати, почтовые службы. Тип сервера определяется видом ресурса, которым он управляет. Например, если управляемым ресурсом является база данных, то соответствующий сервер называется сервером базы данных.

Структура распределенной ИС, построенной по архитектуре клиент-сервер с использованием сервера баз данных, рассматривается на рисунке 2. При такой архитектуре сервер базы данных обеспечивает выполнение основного объема обработки данных. Формируемые пользователем или

приложением запросы поступают к серверу базы данных в виде инструкции языка SQL. Сервер базы данных выполняет поиск и извлечение нужных данных, которые затем передаются на компьютер пользователя. Достоинством такого подхода в сравнении с файл-сервером является заметно меньший объем передаваемых данных.

Для создания и управления персональными базами данных и приложений, работающих с ними, используются СУБД, такие как Accessи VisualFoxProфирмы Microsoft, Paradoxфирмы Borland.

Корпоративная база данных создается, поддерживается и функционирует под управлением сервера баз данных, например MicrosoftSQLServer. В зависимости от размеров организации и особенностей решаемых задач ИС может иметь одну из следующих конфигураций: компьютер-сервер, содержащий корпоративную и персональную базы; компьютер-сервер и персональные компьютеры с ПБД; несколько компьютеров-серверов и персональных компьютеров с ПБД.

Использование архитектуры клиент-сервер дает возможность постепенного наращивания ИС предприятия, во-первых, по мере развития предприятия; во-вторых, по мере развития самой ИС.

Разделение общей базы данных на корпоративную и персональные позволяет уменьшить сложность проектирования баз данных по сравнению с централизованным вариантом, а значит снизить вероятность ошибок при проектировании и стоимость проектирования.

Важнейшим достоинством применения базы данных в ИС является обеспечение независимости данных от прикладных программ, это дает возможность пользователям не

заниматься проблемами представления данных на физическом уровне: размещение данных в памяти, методов доступа к ним.

Такая независимость достигается поддерживаемым СУБД многоуровневым представлением данных в базе данных на логическом (пользовательском) и физическом уровнях. Благодаря СУБД и наличию логического уровня представления данных обеспечивается отделение концептуальной (понятийной) модели базы данных от ее физического представления в памяти ЭВМ. Важнейшим параметром крупной информационной системы является быстродействие при значительном количестве пользователей, а также надежность, масштабируемость и безопасность. Всё это обеспечивает архитектура «клиент-сервер». Такая архитектура позволяет оптимально распределить работу между клиентскими и серверной частями системы: теперь приложение, работающее на рабочей станции, не читает записи базы данных «напрямую», а посылает запросы на сервер, где они принимаются и последовательно отрабатываются специальными программами. В результате на рабочую станцию поступают только обработанные данные, что радикально сокращает информационные потоки в ЛВС.

/>

Рисунок 2. Структура информационной системы с клиент-сервером

1.3 Языки запросов

Хранимые в базе данные можно обрабатывать вручную, последовательно просматривая и редактируя данные в таблицах, с помощью имеющихся в СУБД соответствующих средств. Для выполнения эффективности применяют запросы, позволяющие производить множественную обработку данных,

то есть одновременно вводить, редактировать и удалять множество записей, а также выбирать данные из таблиц.

Запрос представляет собой специальным образом описанное требование, определяющие состав производимых над базой данных операций по выборке, удалению или модификации хранимых данных. Для подготовки запросов с помощью различных СУБД чаще всего используются два основных языка описания запросов: язык QBE(QueryByExample) — язык запросов по образцу; язык SQL(StructuredQueryLanguage) - структурированный язык запросов.

В языке QBE(Query-by-Example— язык запросов по образцу) используется визуальный подход для организации доступа к информации в базе данных, основанный на применении шаблонов запросов. Применение QBEосуществляется путем задания образцов значений в шаблоне запроса, предусматривающем такой тип доступа к базе данных, который требуется в данный момент, например получение ответа на некоторый вопрос.

Язык QBEбыл разработан компанией IBMв 1970-х годах и предназначался для пользователей, заинтересованных в выборе информации из баз данных. Этот язык получил у пользователей столь широкое признание, что в настоящее время в той или иной мере он реализован практически во всех популярных СУБД, включая и MicrosoftAccess. Средства поддержки языка QBEв СУБД MicrosoftAccessвесьма просты в эксплуатации и в то же время представляют пользователям достаточно широкий спектр возможностей работы с данными. Средства языка QBEмогут использоваться для ввода запросов к информации, сохраняемой в одной или нескольких таблицах, а так же для определения набора полей, которые должны присутствовать в результирующей таблице. Отбор записей может, проводится по конкретному или общему критерию, и предусматривать выполнение необходимых

вычислений на основе информации, сохраняемой в таблицах. Кроме того, средства языка QBEможно использовать для выполнения различных операций над таблицами, например, для вставки и удаления записей, модификации значений полей или создания новых полей и таблиц.

СУБД MicrosoftAccessпри создании запроса с использованием средств QBEнеявно формирует эквивалентный оператор языка SQL, предназначенный для выполнения указанных действий.

Чаще всего используется тип запросов, который принято называть запросами на выборку. Запросы на выборку позволяют просматривать, анализировать и вносить изменения в данные, сохраняемые в одной или нескольких таблицах. При выполнении запроса на выборку СУБД MicrosoftAccessпомещает выбранные данные в динамический набор данных, который представляет собой обновляемый набор записей, зависящий от таблицы или запроса, рассматриваемый как отдельный объект. Исключением являются лишь запросы, использующие специфические возможности языка SQL, которые отсутствуют в языке QBE.

Язык SQLберет свое начало в одной из исследовательских лабораторий компании IBM. В начале 1970-х годов исследователи выполняли первые разработки реляционных систем СУБД (или РСУБД), и тогда они создали подъязык данных, предназначенный для работы в этих системах. Пробная версия этого подъязыка была названа SEQUEL(StructuredEnglishQueryLanguage— структурированный английский язык запросов). Однако когда пришло время официально выпускать их язык запросов в качестве продукта, разработчики захотели сделать так, чтобы люди понимали, что выпущенный продукт отличается от пробной системы СУБД и превосходит ее. Поэтому они решили дать выпускаемому продукту имя, хотя и отличающееся от SEQUEL, но явно принадлежащее к этому же семейству. Так что они назвали его

SQL, который стал стандартом для подъязыков данных. В результате, хотя почти все поставщики и использовали варианты одного языка SQL, платформенная совместимость была слабой.

Вскоре началось движение за создание общепризнанного стандарта SQL, которого мог бы придерживаться каждый. В 1986 году организация ANSIвыпустила официальный стандарт под названием SQL-86.3ТОТстандарт был обновлен той же организацией в 1989 году и получил название SQL-89, а затем в 1992 году, был назван SQL-92. Самой последней версией стандарта SQLявляется SQL2003.

Структурированный язык запросов SQL— это гибкий язык, являющийся самым распространенным инструментом, используемым для связи с реляционной базой данных. Этот язык предназначен для выполнения операций над таблицами (создание, удаление, изменение структуры) и над данными таблиц (выборка, изменение, добавление и удаление), а также некоторых сопутствующих операций. SQLявляется непроцедурным языком и состоит из ограниченного числа команд, специально предназначенных для управления над данными. В связи с этим SQLавтономно не используется, обычно он погружен в среду встроенного языка программирования СУБД (например, FoxPro, Access). Чтобы решить с его помощью задачу, сообщается SQL, то, что именно вам нужно, а СУБД сама решит, как лучше всего выполнить ваш запрос. Язык SQLне обладает функциями полноценного языка разработки, а ориентирован на доступ к данным, поэтому его включают в состав средств разработки программ. В этом случае его называют встроенным. В специализированных системах разработки приложений типа клиент-сервер среда программирования, кроме того, обычно дополнена коммуникационными средствами (установление и разъединение соединений с серверами баз данных, обнаружение и обработка возникающих в сети ошибок),

средствами разработки пользовательских интерфейсов, средствами проектирования и отладки.

продолжение
--PAGE_BREAK--

Различают два основных метода использования встроенного SQL: статический и динамический.

При статическом использовании языка в тексте программы имеются вызовы функций языка SQL, которые жестко включаются в выполняемый модуль после компиляции. Изменения в вызываемых функциях могут быть на уровне отдельных параметров вызовов с помощью переменных языка программирования.

При динамическом использовании языка предполагается динамическое построение вызовов SQL-функций и интерпретация этих вызовов, например, обращение к данным удаленной базы, в ходе выполнения программы. Динамический метод обычно применяется в случаях, когда в приложении заранее неизвестен вид SQL-вызова, и он стремится в диалоге с пользователем. Основным назначением языка SQLявляется подготовка и выполнение запросов.

По возможностям манипулирования данными при описании запросов указанные языки практически эквивалентны. Главное отличие между ними, заключается в способе формирования запросов: язык QBEпредполагает ручное или визуальное формирование запроса, в то время как использование SQLозначает программирование запроса.

2. РАЗРАБОТКА ПРИЛОЖЕНИЙ АРХИТЕКТУРЫ КЛИЕНТ-СЕРВЕР ПРИ ПОМОЩИ SQL

SQL— это язык манипулирования данными, который работает в одно- или многопользовательской системе. Особенно

хорошо SQLработает в системе клиент-сервер. В такой системе пользователи работают на множестве клиентских машин, соединенных с серверным компьютером. В прикладной программе (DBLIB), работающей на клиентском компьютере, создаются команды SQL. Та часть системы СУБД, которая находится на клиентском компьютере, передает эти команды на сервер по каналу связи, соединяющему сервер с клиентом. А та часть СУБД, которая находится на сервере, интерпретирует и выполняет полученную команду SQL, а затем по каналу связи отправляет результаты назад к клиенту. В виде SQLможно закодировать сложные операции, а затем на сервере декодировать их и выполнить. Такого рода система позволяет эффективнее использовать пропускную способность канала связи.

Архитектура клиент-сервер, дополняя характеристики SQL, дает возможность в малых, средних и больших сетях получать хорошую производительность при умеренных расходах.

2.1 Обеспечение безопасности

Система безопасности SQLусловно делится на два уровня: сервера и базы данных. На уровне сервера определяется возможность доступа пользователей к серверу. На уровне базы данных для пользователей, получивших доступ к серверу, устанавливаются права доступа к объектам базы данных.

На сервере система защиты SQLможет быть реализована в двух режимах: стандартном — комбинацией средств защиты и интегрированном -использованием только средств защиты.

В стандартном режиме защиты контроль и управление учетными записями, используемыми для доступа к серверу, осуществляет SQL. Кроме того, SQLсамостоятельно выполняет проверку подлинности пользователя с помощью пароля (т.е.

аутентификация), хранит данные о правах доступа, именах и паролях. Стандартный режим используется наиболее часто. Его рекомендуется применять в случаях, когда в сети не используются средства WindowsNT/2kдля аутентификации пользователей и при использовании подключения к серверу с помощью различных протоколов.

В интегрированном режиме защиты контроль над устанавливаемыми пользователями соединениями осуществляет операционная система WindowsNT/2k. Достоинствами интегрированного режима защиты является то, что после регистрации пользователя в домене (т.е. ввода своего имени и пароля) он сразу получает соответствующие права доступа ко всем ресурсам домена WindowsNT/2k, в том числе и к данным SQL, а также использование передачи по сети. Такой метод автоматического предоставления доступа называется установлением доверительного соединения. Считается, что режим является более защищенным по сравнению с предыдущим, так как аутентификация средствами WindowsNT/2kявляется гораздо более защищенной, чем аутентификация SQL.

2.2 Язык SQL

Все языки манипулирования данными (ЯМД), созданные до появления реляционных баз данных и разработанные для многих СУБД персональных компьютеров, были ориентированы на операции с данными, представленными в виде логических записей файлов. Это требовало от пользователей детального знания организации хранения данных и достаточных усилий для указания не только того, какие данные нужны, но и того, где они размещены и как шаг за шагом получить их.

SQLже (StructuredQueryLanguage— структурированный язык запросов) ориентирован на операции с данными,

представленными в виде логически взаимосвязанных совокупностей таблиц. Особенность предложений этого языка состоит в том, что они ориентированы в большей степени на конечный результат обработки данных, чем на процедуру этой обработки. SQLcam определяет, где находятся данные, какие индексы и даже наиболее эффективные последовательности операций следует использовать для их получения: не надо указывать эти детали в запросе к базе данных.

Для иллюстрации различий между ЯМД рассматривается следующая ситуация. Пусть, например, вы собираетесь посмотреть кинофильм и хотите воспользоваться для поездки в кинотеатр услугами такси. Одному шоферу такси достаточно сказать название фильма — и он сам найдет вам кинотеатр, в котором показывают нужный фильм. (Подобным же образом, самостоятельно, отыскивает запрошенные данные SQL.)

Для другого шофера такси вам, возможно, потребуется самому узнать, где демонстрируется нужный фильм и назвать кинотеатр. Тогда водитель должен найти адрес этого кинотеатра. Может случиться и так, что вам придется самому узнать адрес кинотеатра и предложить водителю проехать к нему по таким-то и таким-то улицам. В самом худшем случае вам, может быть, даже придется по дороге давать указания: «Повернуть налево... проехать пять кварталов... повернуть направо...». (Аналогично больший или меньший уровень детализации запроса приходится создавать пользователю в разных СУБД, не имеющих языка SQL.)

Разработка, в основном, шла в отделениях фирмы IBM(языки ISBL, SQL, QBE) и университетах США (PIQUE, QUEL). Последний создавался для СУБД INGRES(InteractiveGraphicsandRetrievalSystem), которая была разработана в начале 70-х годов в Университете шт. Калифорния и сегодня входит в пятерку лучших профессиональных СУБД. Сегодня из всех этих языков

полностью сохранились и развиваются QBE(Query-By-Example — запрос по образцу) и SQL, а из остальных взяты в расширение внутренних языков СУБД только наиболее интересные конструкции.

В начале 80-х годов SQL«победил» другие языки запросов и стал фактическим стандартом таких языков для профессиональных реляционных СУБД. В 1987 году он стал международным стандартом языка баз данных и начал внедряться во все распространенные СУБД персональных компьютеров. Почему же это произошло?

Непрерывный рост быстродействия, а также снижение энергопотребления, размеров и стоимости компьютеров привели к резкому расширению возможных рынков их сбыта, круга пользователей, разнообразия типов и цен. Естественно, что расширился спрос на разнообразное программное обеспечение.

Борясь за покупателя, фирмы, производящие программное обеспечение, стали выпускать на рынок все более и более интеллектуальные и, следовательно, объемные программные комплексы. Приобретая (желая приобрести) такие комплексы, многие организации и отдельные пользователи часто не могли разместить их на собственных ЭВМ, однако не хотели, и отказываться от нового сервиса. Для обмена информацией и ее обобществления были созданы сети ЭВМ, где обобществляемые программы и данные стали размещать на специальных обслуживающих устройствах — файловых серверах.

СУБД, работающие с файловыми серверами, позволяют множеству пользователей разных ЭВМ (иногда расположенных достаточно далеко друг от друга) получать доступ к одним и тем же базам данных. При этом упрощается разработка различных автоматизированных систем

управления организациями, учебных комплексов, информационных и других систем, где множество сотрудников (учащихся) должны использовать общие данные и обмениваться создаваемыми в процессе работы (обучения). Однако при такой идеологии вся обработка запросов из программ или с терминалов пользовательских ЭВМ выполняется на этих же ЭВМ. Поэтому для реализации даже простого запроса ЭВМ часто должна считывать из файлового сервера и (или) записывать на сервер целые файлы, что ведет к конфликтным ситуациям и перегрузке сети.

продолжение
--PAGE_BREAK--

Для исключения указанных и некоторых других недостатков была предложена технология «Клиент-Сервер», по которой запросы пользовательских ЭВМ (Клиент) обрабатываются на специальных серверах баз данных (Сервер), а на ЭВМ возвращаются лишь результаты обработки запроса. При этом, естественно, нужен единый язык общения с Сервером и в качестве такого языка выбран SQL. Поэтому все современные версии профессиональных реляционных СУБД (DB2, Oracle, Ingres, Informix, Sybase, Progress, Rdb) и даже нереляционных СУБД (например, Adabas) используют технологию «Клиент-Сервер» и язык SQL. К тому же приходят разработчики СУБД персональных ЭВМ, многие из которых уже сегодня снабжены языком SQL.

Бытует мнение: Поскольку большая часть запросов формулируется на SQL, практически безразлично, что это за СУБД — был бы SQL.

Реализация в SQLконцепции операций, ориентированных на табличное представление данных, позволило создать компактный язык с небольшим (менее 30) набором предложений. SQLможет использоваться как интерактивный

(для выполнения запросов) и как встроенный (для построения прикладных программ).

Ориентированный на работу с таблицами SQLне имеет достаточных средств для создания сложных прикладных программ. Поэтому в разных СУБД он либо используется вместе с языками программирования высокого уровня (например, такими как Си или Паскаль), либо включен в состав команд специально разработанного языка СУБД (язык систем dBASE, R:BASEи т.п.).

2.3 Организация взаимодействия клиент-сервер при помощи SQL

При использовании технологии клиент-сервер приложение разделяется на две части. Клиентская часть обеспечивает удобный графический интерфейс и размещается на компьютере пользователя. Серверная часть осуществляет управление данными, разделение информации, администрирование и обеспечивает безопасность информации. Клиентское приложение формирует запросы к серверу базы данных, на котором выполняются соответствующие команды. Результаты выполнения запросов пересылаются клиенту.

При разработке распределенных информационных систем в организации взаимодействия клиентской и серверной части выделяются следующие важные в практическом смысле задачи:

Перенос персональной базы данных на сервер для последующего ее коллективного использования как корпоративной базы данных;

Организация запросов к корпоративной базе данных, размещенной на сервере, со стороны компьютера-клиента;

Разработка клиентского приложения для удаленного доступа к корпоративной базе данных со стороны компьютера- клиента.

Задача переноса персональной базы на сервер может возникать в ситуациях, когда требуется обеспечить коллективный доступ к базе данных, разработанной с помощью персональной СУБД (FoxPro, Access). Для решения этой задачи, в составе названных персональных СУБД имеются соответствующие средства, предназначенные для преобразования баз данных в формат SQL.

Подготовка запросов к базе данных на сервере (на языке SQL) со стороны клиентской части может выполняться с помощью специально предназначенной утилиты. Для предоставления пользователю больших возможностей и удобства в подготовке и выполнении запросов создаются клиентские приложения.

Для организации запросов к серверной базе данных на языке SQLили с помощью клиентского приложения возможны различные способы взаимодействия, заметно влияющие на эффективность. К числу основных способов такого взаимодействия относятся:

Интерфейс DB-LIB (библиотек баз данных);

Технологии ODBC (совместимости открытых баз данных);

Интерфейса OLE DB (связывания и встраивания объектов баз данных);

Технологии DAO (объектов доступа к данным);

Технологии ADO (объектов данных).

Интерфейс DB-LIBпредставляет собой специально предназначенный для SQLинтерфейс прикладных программ. Поэтому он является наименее мобильным из числа рассматриваемых в смысле возможностей переноса

приложений в другую среду. С точки зрения производительности этот способ позволяет осуществить самый быстрый доступ к информации. Причиной этого является то, что он представляет оптимизированный интерфейс прикладного программирования и непосредственно использует язык запросов системы SQL.

Технологии ODBCпредназначены для обеспечения возможности взаимосвязи между различными СУБД и получения от приложения запросов на выборку информации, перевод их на язык ядра адресуемой базы данных для доступа хранимой в ней информации.

Основное назначение ODBCсостоит в абстрагировании приложения от особенностей ядра серверной базы данных, с которой оно осуществляет взаимодействие, поэтому серверная база данных становится как бы прозрачной для любого клиентского приложения.

Достоинством этой технологии является простота разработки приложений, обусловленная высоким уровнем абстрактности интерфейса доступа к данным практически любых существующих типов СУБД. Используя эту технологию, можно создавать клиент-серверные приложения, причем средствами персональных СУБД целесообразно разрабатывать клиентскую часть приложения, а средствами SQL— серверную часть.

Основной недостаток технологии ODBCсвязан с необходимостью трансляции запросов, что снижает скорость доступа к данным. В системах клиент-сервер этот недостаток устраняется путем перемещения запроса с компьютера-клиента на компьютер-сервер. При этом устраняются промежуточные звенья, являющиеся основной причиной снижения скорости обработки информации с использованием средств рассматриваемой технологии.

При использовании в клиентском приложении средств ODBCосуществляется обращение к определенному источнику данных, а через него — к СУБД, которую он представляет. При установке средств ODBC устанавливается общая подсистема ODBCи определяются пары «драйвер-база данных», которым задаются имена, используемые при установке соединения с. базой данных. Соответствующие пары называются поименованными источниками данных.

Каждый поименованный источник данных описывает собственно источник данных и информацию о доступе к этим данным. В качестве данных могут выступать базы данных, электронные таблицы и текстовые файлы. Информация о доступе, например, к серверу баз данных, обычно включает в себя сведения о размещении сервера, имя базы данных, идентификатор учетной записи и пароль, а также различные параметры драйвера, описывающие как устанавливать соединение с источником данных.

При обработке данных на сервере с использованием технологии ODBCи применением клиентского приложения выделяются два основных этапа: задание источника данных — создание и настройка соединения, а также собственно обработка данных с помощью запросов.

Интерфейс OLEDBрекомендуется использовать для создания средств и утилит, или разработок системного уровня, нуждающихся в высокой производительности или доступе к SQLсвойствам, недоступные с помощью технологии ADO. Основные возможности спецификации OLEDBобеспечивают полную функциональность доступа к данным. В SQLпроцессор баз данных сервера использует это интерфейс для связи: между внутренними компонентами, таким как процессор хранения и процессор отношений; между установками SQLпри использовании удаленных хранимых процедур; как интерфейс к другим источникам данных для распределенных запросов.

При использовании технологии OAO работа с базами данных, таблицами ведется с использованием коллекций объектов. При этом обеспечиваются большие удобства в работе с объектами баз данных.

В настоящее время технология OAO постепенно вытесняется технологией ADO, которая позволяет разрабатывать приложения Webдля работы с базами данных. В целом технологию ADOможно охарактеризовать как наиболее современную технологию разработки приложений для работы с распределенными базами технологии клиент-сервер.

продолжение
--PAGE_BREAK--

2.4 Среды программирования на языке SQL

На сегодняшний день известно более двух десятков серверных СУБД, однако наиболее популярными, исходя из числа продаж и инсталляций, следует признать Oracle, MicrosoftSQLServer.

Oracleбыла первой коммерческой реляционной СУБД, поддерживающей ставший ныне индустриальным стандартом язык SQL; ее первая версия появилась в 1979 году. Фактически все это время Oracleявляется бессменным лидером на рынке производителей коммерческих СУБД и второй (после Microsoft) по величине компанией, производящей программное обеспечение.

Ранние версии этой СУБД были предназначены для мэйнфреймов, а в качестве рабочих мест использовались <неинтеллектуальные> терминалы. Однако со временем появились версии Oracle, предназначенные для использования в архитектуре <клиент-сервер> (первой такой версией была Oracle5, выпущенная в 1985 году). Первоначально эти версии были предназначены для различных серверных платформ — различных версий UNIX,

VMSи др. Позже были выпущены версии сервера Oracleдля NovellNetWare. Первые версии этого сервера для персональных компьютеров появились в середине 90-х (PersonalOracle7 forWindows3.1, PersonalOracle7 forWindows95, PersonalOracleLite, OracleWorkgroupServer7 forWindowsNT). До появления этих версий персональные компьютеры могли использоваться исключительно в качестве клиентских рабочих станций — в состав Oracleдля серверных платформ обычно входила клиентская часть для DOS.

Отметим, что Oraclebыла первой компанией, создавшей СУБД, использовавшую предоставляемые некоторыми серверными платформами средства параллельных вычислений — OracleParallelServer(до его появления параллельные вычисления использовались только для решения научных задач). При использовании параллельных вычислений OracleParallelServerдает возможность нескольким процессорам обращаться к одной базе данных, что позволяет обеспечить высокую скорость обработки транзакций, а более поздние его версии дают возможность осуществить декомпозицию операций с большими объемами данных с целью параллельного выполнения их на нескольких процессорах.

Помимо различных версий сервера баз данных среди продуктов Oracleимеется также Designer/2000 — ориентированное на эту СУБД CASE-средство для анализа бизнес-процессов и проектирования данных, а также средства разработки клиентских приложений. Одно из них — Developer/2000 (называвшееся ранее Oracle*Forms) — весьма популярно среди пользователей Oracle; были и другие средства разработки (например, OraclePowerObjects). Отметим, что приложения, созданные с помощью Developer/2000, могут выполняться на различных платформах. Язык SQL, используемый в этом средстве разработки, является интерпретируемым и представляет собой тот же самый язык,

что используется в Oracleдля написания серверного кода. Это позволяет отлаживать с помощью Developer/2000 серверный код.

Производя собственные средства разработки, Oracleпредоставляет своим пользователям возможность создавать клиентские приложения с помощью других средств. В частности, помимо стандартного в таких случаях клиентского API(OracleCallInterface) клиентская часть Oraclecодержит также объектную модель (OracleObjectsforOLE), позволяющую использовать клиентскую часть Oraclекак набор COM-объектов для доступа к данным. Кроме того, обычно клиентская часть Oraclecодержит также ODBC-драйвер для доступа к данным этой СУБД.

Отметим, что и многие другие компании производят ODBC-драйверы и OLEDB-провайдеры для доступа к Oracle(в частности, Microsoft). Компании, производящие средства разработки, использующие собственные библиотеки доступа к данным (такие как Inpriseили Gupta/Centura), также включают библиотеки доступа к Oracleв состав наиболее дорогих версий своих продуктов.

Из готовых информационных систем на базе Oraclecледует особо отметить несколько крупных систем управления предприятием, в частности SAP/К3. На Западе также нередко используются готовые решения от самой OracleCorporation, объединенные под общим названием OracleApplications, такие как OracleFinancials, OracleHumanResources, OracleMarketManagement, OracleProjectSystemsи др.

MicrosoftSQLServer6.0 — одна из наиболее мощных СУБД архитектуры клиент-сервер. Эта СУБД позволяет удовлетворять такие требования, предъявляемые к системам распределенной обработке данных, как тиражирование данных, параллельная обработка, поддержка больших баз

данных на относительно недорогих аппаратных платформах при сохранении простоты управления и использования.

MicrosoftSQLServerпредставляет собой систему, выполняющую функции управления базой данных. Для пользовательского приложения SQLServerявляется мощным источником генерации и управления нужными данными.

Сервер имеет средства удаленного администрирования и управления операциями, организованными на базах объектно-ориентированной распределенной сред управления. MicrosoftSQLServerвходит в состав семейства MicrosoftBackOffice, объединяющего пять серверных приложений, разработанных для совместного функционирования в качестве интегрированной системы.

MicrosoftSQLServerпредназначен исключительно для поддержки систем, работающих в среде клиент-сервер. Он поддерживает широкий спектр среды разработки и максимально прост в интеграции с приложениями, работающими на персональном компьютере. Данная версия превосходит предыдущую с точки зрения использования многопоточной параллельной архитектуры операционной системы для повышения производительности и масштабируемости, то есть очень эффективно использует возможность ускорения работы в том случае, если на компьютере установлено несколько процессоров.

продолжение
--PAGE_BREAK--

MicrosoftSQLServer6.0 имеет новую масштабируемую архитектуру блокировок, называемую динамической блокировкой (DynamicLocking), которая комбинирует блокировку на уровне страницы и записи для достижения

максимальной производительности и подключения максимального числа пользователей.

MicrosoftSQLServerможет тиражировать информацию в базы данных иных форматов, включая Oracle, IBMDB2, Sybase, MicrosoftAccessи другие СУБД при наличии ODBCдрайвера, отвечающего определенным требованиям (ODBC— OpenDataBaseConnectivity, стандарт Microsoft, разрешающий программам работать с различными серверами баз данных, используя один общий интерфейс).

Хранимые процедуры, поддерживающие OLEAutomation, позволяют разработчику применять практически любой инструмент из тех, что поддерживают OLE, в целях создания хранимых процедур для SQLServer. VisualBasic4.0 поддерживается посредством новой 32-разрядной DB-Library(OCX). Многочисленные расширения языка Transact-SQLвключают расширенную поддержку курсоров, возможность использования команд определения данных внутри транзакций.

MicrosoftSQLServer6.0. содержит Ассистент администратора. Этот инструмент позволяет назначать основные процедуры сопровождения базы данных и определять для них график выполнения. Операции по сопровождению баз данных включают проверку распределения страниц, целостности указателей в таблицах (включая системные) и индексах, обновление информации, необходимой оптимизатору, реорганизацию страниц в таблицах и индексах, создание страховочных копий таблиц и журналов транзакций. Все эти операции могут быть установлены для автоматического выполнения по заданному администратором графику. Пакет EnterpriseManagerвключает утилиту позволяющую переносить некоторые или все объекты из одной базы данных в другую.

Сервер, который получает объекты, должен быть MicrosoftSQLServerверсии 6.0. Сервер источник может быть MicrosoftSQLServer4.хили сервер Sybase. SQLServerпредоставляет возможность создания страховочных копий и восстановления индивидуальных таблиц. Загрузка таблица может быть выполнена либо из копии индивидуальной таблицы, либо из копии базы данных. Загрузка индивидуальных таблиц может оказаться хорошим решением при необходимости восстановления данных после сбоя, когда загрузка всей базы данных неэффективна. Тем не менее создание страховочных копий всей базы данных и журнала транзакций остаются основой стратегии резервного копирования.

Для эффективной работы с данными SQLServerимеет целый набор специальных инструментов.

Характеристика основного инструмента MicrosoftSQLServer6.0.

SQLSetup— используется для установки нового, модификации установленного программного обеспечения и удаления SQLServerс диска. Программа Setupтакже может быть использована для изменения опций сетевой поддержки, подключения языка, перестройка базы данных Masterи установки опций доступа к данным.

SQLService— используется для старта и остановки служб SQLServerManager(SQLServerи SQLExecutive).

SQL/w— позволяет вводить выражения и хранимые процедуры Transact-SQLв графическом интерфейсе запросов.

SQLSecurity— позволяет управлять бюджетами пользователей серверов ManagerSQL.

SQLClient— устанавливает информацию соединения Serverдля утилиты конфигурирования клиентов.

SQLTransfer— обеспечивает легкий графический способ переноса Managerобъектов и данных с одного Serverна другой.

SQLTrace— графическая утилита, позволяющая администраторам и разработчикам отслеживать и фиксировать активность клиентских приложений, обращающихся к MicrosoftSQLServer6.0. SQLTraceможет в реальном времени отображать все аспекты обращений к серверу или использовать фильтры, отображающие информацию о действиях конкретных пользователей, приложений или машин.

MicrosoftSQLServer6.0 отличается быстродействием, надежностью от Oracle, позволяет удовлетворить более широкие потребности клиентов по развертыванию крупномасштабных распределенных систем информации. SQLServer6.0 обеспечивает мощные инструментальные средства для предприятий -широкой администрации, копирования данных, параллельного DBMSисполнения, и поиск в очень больших базах данных. MicrosoftSQLServer6.0 также обеспечивает плотную интеграцию OLEтехнологии.

SQLServer6.0 продолжает придерживаться промышленных стандартов, с улучшенной ANSISQLподдержкой и языковыми расширениями, которые включают декларативную справочную целостность, и мощную поддержку сервер курсора, что значительно превышает стандарт ANSI.

ЗАКЛЮЧЕНИЕ

Изучив и проанализировав архитектуру информационной системы, в структуру которой входят файл-сервер и клиент-сервер мною был сделан вывод, что файл-сервер во многом уступает клиент-серверу.

Вся тяжесть вычислительной нагрузки при доступе к базе данных ложится на приложение клиента, что является следствием принципа обработки информации в системах файл-сервер при выдаче запроса на выборку информации из таблицы вся таблица базы данных копируется на клиентское место, и выборка осуществляется на клиентском месте.

продолжение
--PAGE_BREAK--

При этом возникают следующие ограничения:

— невозможность организации равноправного одновременного доступа пользователей к одному и тому же участку базы данных;

— количество одновременно работающих с системой пользователей не превышает пяти человек для ЛВС;

— невысокая скорость обработки и представления информации;

— высокие требования к ресурсам компьютеров.

При всем этом система обладает одним очень важным преимуществом - низкой стоимостью.

Недостатки архитектуры файл-сервер решаются при переводе приложений в архитектуру клиент-сервер, достоинствами которой, является то, что вся вычислительная' нагрузка переносится на сервер базы данных, осуществляется высокая защита данных, поддерживается большое количество пользователей и сложных приложений.

Рассмотрев языки запросов SQLи QBE, был сделан вывод, что SQLявляется наиболее гибким, динамичным, а также он поддерживает высокий уровень безопасности данных, их

централизованное хранение и он ориентирован на конечный результат обработки данных.

В принципы разработки приложений архитектуры клиент-сервер входит обеспечение безопасности данных, организация взаимодействия клиента и сервера, все это достигается при использовании языка SQL.

Из сравнения сред программирования MicrosoftSQLServeri Oracle, я сделала вывод, что MicrosoftSQLServerотличается быстродействием, надежностью от Oracle, позволяет удовлетворить более широкие потребности клиентов по развертыванию крупномасштабных распределенных систем информации. SQLServer6.0 обеспечивает мощные инструментальные средства для предприятий — широкой администрации, копирования данных, параллельного DBMSисполнения, и поиск в очень больших базах данных. MicrosoftSQLServer6.0 также обеспечивает плотную интеграцию OLEтехнологии.

SQLServer6.0 продолжает придерживаться промышленных стандартов, с улучшенной ANSISQLподдержкой и языковыми расширениями, которые включают декларативную справочную целостность, и мощную поддержку сервер курсора, что значительно превышает стандарт ANSI.

Следующая курсовая работа будет направлена на разработку информационной системы, обеспечивающей электронный документооборот села Бобровки.

СПИСОК ЛИТЕРАТУРЫ

Тейлор А.Дж. SQL для «чайников» /А.Дж. Тейлор.- Москва: Вильяме, 2005.

Дейт К.Дж. Введение в системы баз данных /К.Дж. Дейт — Москва: ДМК, 2000.

Хомоненко А.Д. Базы данных /А.Д. Хомоненко, В.М. Цыганков — Санкт-Петербург: БХВ-Петербург, 2004.

ВескесЛ.Дж. Access иSQL Server. Руководство разработчика /Дж.Л. Вескес — Москва: Лори, 1997.

Конноли Т. Базы данных. Проектирование, реализация и сопровождение /Т. Конноли, К. Бегг. — Москва: Вильяме, 2003.