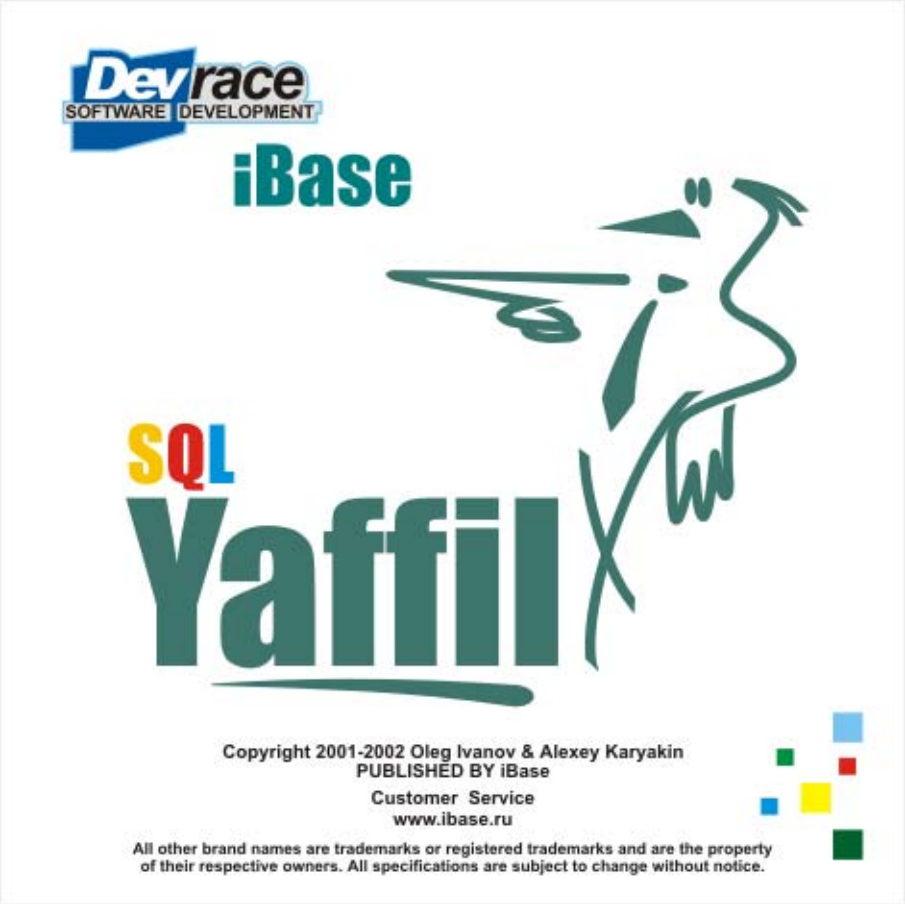
Руководство пользователя

Yaffil® SQL Server



Версия 1.3.0.878

Введение

В этой главе будет рассказано о проекте Yaffil и об отличиях и возможностях этого клона по сравнению с серверами InterBase/Firebird 1.0.

Yaffil, Yaffle (сущ.) [произошло, вероятно, от имитации крика] (зоол.) Европейский зеленый дятел (Picus, or Genius, viridis). Известен своим громким, похожим на смех, криком.-web1913

Неудивительно, что после открытия исходного кода InterBase разработчики, использующие сервер в своих проектах, стали пытаться его усовершенствовать, приспосабливая к своим нуждам и улучшая характеристики. В то же время ясно, что самостоятельная модификация кода разрозненными разработчиками исключительно в собственных целях приведет к появлению несовместимых версий и трудностям в сопровождении. В мире программного обеспечения с открытым исходным кодом (Open Source) подобная неприятная ситуация достаточно распространена и имеет название code forking.

В конце 2001 года в результате объединения усилий группы российских разработчиков, использующих InterBase на Windows NT, на свет появился проект Yaffil. За основу разработчики взяли исходный код сервера Firebird, поскольку он является единственным динамично развивающимся клоном InterBase с открытым исходным кодом. При дальнейшей разработке также планируется периодически синхронизировать исходный код Yaffil с изменениями Firebird.

Почему принято решение создавать новый клон вместо того, чтобы интегрировать изменения с проектом Firebird?

К сожалению, политика координаторов и участников проекта Firebird является достаточно жесткой относительно внедрения новых возможностей. Главным приоритетом команды Firebird является создание стабильной версии на базе существующего исходного кода при сохранении полной платформенной независимости. В то же время, разработчики Yaffil считают очень перспективным направлением интеграцию сервера с Windows NT, что потребует введения в сервер специфичных для данной ОС возможностей, что неприемлемо для участников проекта Firebird. Оптимизация производительности также реализуется с использованием зависимых от платформы (аппаратной и ОС) модулей, включая участки кода, реализованные на ассемблере.

В данных условиях представляется целесообразным создание отдельного клона InterBase/Firebird с дополнительными возможностями и лучшей производительностью на Windows NT при сохранении совместимости с другими версиями. Под совместимостью подразумеваются следующее:

1. Возможность переноса баз данных от InterBase/Firebird к Yaffil через резервную копию базы данных (backup-restore) и обратно, если в базе данных не используются возможности, специфичные для конкретной версии.
2. Возможность подключения клиентов Yaffil к серверам InterBase/Firebird по протоколу TCP/IP.
3. Возможность подключения клиентов InterBase/Firebird к серверу Yaffil по протоколу TCP/IP.
4. Выполнение сервером Yaffil инструкций диалекта SQL, используемого в Firebird v 1.0 и InterBase 6.0.

Приоритетные направления развития Yaffil

Интеграция с платформой Windows NT

Изначально InterBase разрабатывался на платформах Unix, и только в начале 90 годов, в версии 4.0, был перенесен на Windows NT. К сожалению, при переносе кода мало внимания было уделено платформо-зависимой оптимизации под Windows NT. В результате данный вариант сервера оказался менее эффективным, чем мог бы быть. В то же время число установок InterBase на Windows NT составляет значительную (а возможно и большую) часть, поэтому такая ситуация является достаточно печальной.

Кроме того, Windows NT обладает рядом возможностей, использование которых в сервере СУБД позволит добиться нового уровня функциональности и увеличить отдачу при использовании сервера в прикладных системах. Среди таких возможностей назовем интерфейс для аутентификации пользователей по протоколам NTLM или Kerberos, при использовании которого нет необходимости вводить имя пользователя и пароль при каждом соединении, встроенные криптографические средства, координатор распределенных транзакций (MS DTC), асинхронный ввод-вывод, интерфейс WinSock2 и другие. Все перечисленные возможности широко используются другими серверами баз данных, однако полностью игнорируются сервером InterBase/Firebird.

Производительность

Производительность систем на базе СУБД является одним из ключевых факторов при выборе сервера. Учитывая это, при разработке сервера Yaffil большое внимание уделяется оптимизации исходного кода, используются средства анализа и профайлинга производительности.

Надежность и безопасность

В сервере Yaffil исправлено большое число критических ошибок, многие из которых приводят к порче данных. В сервере InterBase такие опасные ситуации как порча памяти[i] (вследствие ошибки в коде сервера, некорректной UDF или испорченных метаданных БД) протекают бессимптомно до тех пор, пока база данных не будет непоправимо испорчена. В отличие от этого, ошибка при работе с памятью в Yaffil, как правило, приводит к немедленному останову сервера с потерей только последних обновляющих транзакций.

В Yaffil исправлена уязвимость, связанная с использованием внешних таблиц для получения доступа к любому файлу системы.

В Yaffil исправлена уязвимость, связанная с обработкой структурированных исключений (SEH) Win32 API[ii]

Функциональность

В настоящее время в Interbase/Firebird отсутствуют многие возможности, ставшие фактически стандартом в других серверах подобного класса. Среди них можно назвать очень скромный набор встроенных функций, отсутствие возможности выполнения пакетов SQL команд, отсутствие динамического SQL в контексте процедур и триггеров, отсутствие средств мониторинга.

Вместе с тем в исходном коде InterBase имеется большое количество функций, по неясным причинам так и не увидевших свет в выпускаемых версиях. Подобные функции будут включаться, дорабатываться и тестироваться в сервере Yaffil. Некоторые

возможности, например, протокол XNET[iii], индексы по выражениям доступны уже сейчас.

Отличительные особенности сервера Yaffil

Улучшенная производительность

Производительность является одним из ключевых факторов, определяющих пригодность сервера СУБД для использования в конкретном приложении. Производительность определяет максимальную нагрузку, которую сервер может нести на выбранном аппаратной платформе, выраженную, например, в количестве одновременно работающих пользователей или количестве операций, выполняемых в единицу времени. К сожалению, скоростные характеристи линейки серверов InterBase часто уступают показателям конкурентов, таких как MSSQL или Sybase SQL Anywhere. Причины этого лежат как в архитектурных особенностях, так и в недостаточно оптимальной реализации внутренних алгоритмов сервера от Borland.

По производительности Yaffil далеко опережает своих аналогов из клона InterBase. При выполнении тех же самых запросов Yaffil работает в 2-3 раза быстрее, чем InterBase 6.x и Firebird 1.0. Такие результаты достигнуты благодаря многочисленным улучшениям алгоритмов и оптимизацией кода, большое число критических алгоритмов переписаны на ассемблере x86, убраны ненужные системные вызовы.

Улучшенный оптимизатор запросов

Оптимизатор - своего рода "мозг" сервера, и степень его интеллектуальности может кардинально повлиять на скорость работы приложений. Неверно выбранный план может привести к увеличению времени выполнения запроса в тысячи раз. За время своей эволюции от версии InterBase 4.x, неспособной вообще оптимизировать явные соединения, и до последних версий, в оптимизатор InterBase разработчиками Borland вносились изменения, которые хотя и приводили к улучшению планов для определенных случаев, но часто непредсказуемо сказывались на других запросах. Например, в пятой версии InterBase научился корректно оптимизировать соединения в явном синтаксисе ANSI SQL, но другой способ выборки индексов в ряде случаев приводил к катастрофическому увеличению времени выполнения запросов. Подобные неприятности появились в версии InterBase 6.x, в результате многие разработчики не могли перенести свои системы на современную платформу.

Проанализировав большое число проблемных случаев с оптимизацией InterBase версий 4.x, 5.x и 6.x, разработчики Yaffil внесли ряд улучшений, в результате чего в большинстве случаев можно вообще отказаться от применения ручных планов. При этом при переходе на Yaffil с более старых версий InterBase случаев ухудшения автоматических планов практически не наблюдается.

Однако, в тех случаях, когда ручного планирования не избежать, Yaffil предоставляет больше возможностей для этого.

Перечислим некоторые улучшения оптимизатора Yaffil.

Выбор индекса с меньшим числом полей при наличии индекса с большим числом полей.

Этот случай часто возникает, когда в таблице имеется большое количество индексов, в том числе составных. Оптимизатор Interbase/Firebird всегда пытался выбрать индекс, который охватывает множество полей, в то время как существовал более быстрый и компактный индекс. Оптимизатор Yaffil автоматически разрешает данную ситуацию.

Пример:

Таблица из 3-х полей Table1 (F1, F2, F3)

Для неё созданы три индекса: IDX_F1(F1), IDX_F1_F2(F1,F2), IDX_F1_F2_F3(F1,F2,F3)

Выполняем запрос:

select * from Table1 where F1 = параметр

Получаем следующие планы, описывающие использование индексов:

Interbase/FireBird: PLAN (T INDEX (IDX_F1_F2_F3))

Yaffil: PLAN (T INDEX (IDX_F1))

Interbase/FireBird используют индекс с максимальным количеством полей, хотя очевидно, что в данном случае надо использовать единственный индекс по полю F1, как это и делает Yaffil. Это очень распространенная ошибка, приводящая к замедлению выполнения запроса, обойти которую без явного планирования достаточно сложно.

Возможность использования индекса с начальными сегментами, удовлетворяющими условию, и сортировкой по остальным

Этот случай возникает, когда выборка делается по одному полю, а сортировка - по другому полю, при этом имеется индекс по обоим полям. Пример - пусть существует таблица из трех полей: T(F1,F2,F3), есть индексы на следующие сочетания полей: на одно поле IDX_F1(F1), на два первых поля IDX_F1_F2(F1,F2), на все три поля IDX_F1_F2_F3(F1,F2,F3).

Производим запрос следующего вида: select * from T where F1=.. order by F2

Получаем планы следующего вида:

FireBird/InterBase 6.5: PLAN SORT(T INDEX (IDX_F1_F2_F3))

Yaffil: PLAN (T ORDER (IDX_F1))

Исключение из обработки индексов с сильно различной селективностью

Пример - у нас есть таблица из трех полей T(F1,F2,F3) с индексами на каждое поле IDX_F1(F1), IDX_F2(F2) и IDX_F3(F1). Здесь F1 является уникальным полем (например, первичный ключ), а F2 - не уникальным. Производим запрос:

select * from T where F1 = параметр1 and F2 = параметр2 ...

В результате получаем следующие планы:

FireBird/InterBase:

PLAN (T INDEX (IDX_F1, IDX_F2))

Yaffil: PLAN (T INDEX (IDX_F1))

Оптимизатор Yaffil всегда объединяет индексы с учётом селективности.

Отметим, что в качестве критериев отбора индексов используется не только селективность, но и категории условий, используемых в ограничениях. Так, например "вес" операции "=" больше, чем "вес" операций "<" и ">". Дополнительно оптимизатор Yaffil учитывает такие характеристики индексов как уникальность и число сегментов попадающих в условие сортировки.

Алгоритм выбора индексов

Алгоритм, реализованный в сервере Yaffil, принципиально отличается от существующего алгоритма Interbase/Firebird. Для начала рассмотрим последовательность действий, приводящих к формированию множества используемых индексов, используемых при выполнении запроса, в Interbase/Firebird:

1. Выбор индекса с максимальным количеством сегментов, для которого хотя бы один из начальных сегментов попадает под условие выборки.
2. Просмотр всех индексов таблицы в порядке обратном количеству сегментов индекса, начиная с числа сегментов индекса, выбранного на предыдущем этапе, и формирование ограничений выборки.
3. После формирования множества подходящих индексов для всех таблиц запроса происходит их сортировка в плане с учётом их селективности.

Таким образом, приоритет имеют индексы с максимальным количеством сегментов, вне зависимости от вида условий накладываемых на сегменты индекса и селективности, что и приводит к формированию неоптимальных планов.

Последовательность действий в сервере Yaffil:

1. Сортировка индексов с учётом условий выборки и селективности. Индексы сортируются по следующим условиям (в порядке приоритета).
    a. Уникальный индекс, ко всем сегментам которого применяется ограничение «=»
    b. Условный вес ограничений выборки, накладываемых на начальные сегменты индекса
    c. Условный вес ограничений выборки, определяющих возможный порядок сортировки
    d. Меньшее число сегментов индекса, для которых заданы ограничения
    e. Меньшее число сегментов индекса
    f. Селективность индекса
2. Просмотр отсортированного на первом шаге списка индексов таблицы и формирование ограничений выборки. При этом не используются индексы со значениями селективности и удельного веса существенно худшими по сравнению с соответствующими значениями уже отобранных индексов.
3. После формирования множества подходящих индексов для всех таблиц запроса происходит их сортировка в плане с учётом их селективности, а также условного веса ограничений, заданных для индекса.

Оптимизация сетевого трафика

Как правило, сервер СУБД устанавливается не на одном компьютере вместе с клиентом, а используется через локальную сеть. При этом на времени отклика сервера сказываются задержки при передаче данных по сети независимо от того, насколько мощный сервер установлен.

Объем сетевого трафика Yaffil значительно уменьшен за счет более эффективной передачи полей типа VARCHAR, так как передаются данные фактической длины, а не объявленной в описании типа. Аналогичное улучшение в других клонах есть только в InterBase начиная с версии 6.5.

Эффективная работа с временными файлами сортировки

В сервере InterBase сортировка всегда выполняется с использованием временных файлов независимо от количества доступной памяти. Операции чтения/записи временных файлов дополнительно нагружают дисковую подсистему, что может отрицательно сказаться на скорости работы сервера, особенно при наличии параллельно работающих соединений, интенсивно обращающихся к диску. Поэтому сервер Yaffil открывает временные файлы сортировки с флагом FILE_ATTRIBUTE_TEMPORARY, что предотвращает сброс кэш-буферов на диск операционной системой при наличии достаточного объема кэш-памяти.

Кроме оптимизированной работы с временными файлами, сервер Yaffil содержит улучшенный алгоритм сортировки, более эффективно использующий оперативную память. Фактически, временные файлы сортировки требуются, только при сортировке более чем 16Кб записей.[1]

Оптимальная структура хранения записей

InterBase использует эффективный способ хранения записей на страницах базы данных, используя алгоритм RLE (run length encoding - кодирование последовательностей) при размещении данных, за счет которого базы данных InterBase являются компактными. Несмотря на то, что алгоритм является очень простым, он, тем не менее, требует вычислительных ресурсов при записи и чтении данных на странице.

В зависимости от характера данных упаковка может оказаться нецелесообразной. Наибольший эффект достигается при сжатии "хвостов" из пробелов длинных текстовых строк (типы данных CHAR и VARCHAR), в то время как короткие нетекстовые данные практически несжимаемы.

Для управления сжатием в Yaffil введен параметр конфигурации SQZ_BLOCK. Этот параметр определяет минимальный размер блока[2] строки таблицы (в байтах), начиная с которого будет производиться сжатие. Блоки менее указанного размера хранятся в исходном виде. С увеличение параметра объем базы данных возрастает за счет сокращения затрат времени процессора на упаковку, в то же время, увеличившийся объём данных больше нагружает дисковую подсистему. В зависимости от характера данных[iv] (главным образом количества текстовых полей в таблицах) и определенных параметрах процессора и дисков системы существует некоторое оптимальное значение SQZ_BLOCK, обеспечивающее максимальную производительность. Алгоритм упаковки, реализованный в Yaffil, дополнительно выравнивает начало и размер сжатого блока на границу машинного слова.

Ускоренная работа с индексами

Гораздо быстрее работают операции поиска по индексам (часто используемые в соединениях), а также построения индексов. Это достигнуто за счет тщательной оптимизации кода индексирования.

[1] В сборках до 865 рекомендовалось устанавливать параметр LargeSystemCache в 1. Начиная с 865 сборки изменение указанного параметра не требуется

[2] Блок - последовательность байтов обрабатываемая за один шаг алгоритмом RLE

Улучшенная стратегия вычисления предиката IN и условий, объединенных по OR

По сравнению с InterBase и другими клонами, Yaffil выполняет предикат IN гораздо более эффективно. При этом используется только одно построение битовой карты для индекса независимо от числа параметров. В других клонах InterBase получим значительное замедление при большом числе параметров, (особенно если IN не ограничивает основной объём выборки), несмотря на *одинаковый*[3] план исполнения запроса. То же самое произойдет и для условия вида OR:

select * from T where F1 = .. or F2 = ... or F2

Более быстрое обновление данных

Наибольший выигрыш в скорости при использовании Yaffil наблюдается при обновлении данных[v].

Переработаны алгоритмы, связанные с управлением деревом "грязных страниц" буферного кэша, в результате чего устранено замедление при массивных обновлениях данных в рамках одной транзакции. Ликвидирована известная проблема InterBase, при которой задание числа буферов больше некоторого предела (около 1000) приводит не к увеличению скорости, а совсем наоборот, при этом возможны даже зависания (известная проблема "10000 буферов").

Устранена деградация скорости при обновлении большого числа рядов данных, а также при откате большого числа изменений. Быстрее выполняется массовая вставка данных, также за счет оптимизации кода.

Уменьшение времени, необходимого для резервного копирования и восстановления

В InterBase отсутствует возможность инкрементального резервного копирование, при этом сам процесс backup/restore из-за архитектурных особенностей проходит медленнее по сравнению с серверами СУБД, где запись копии происходит постранично. Резервное копирование большой базы может занимать несколько часов, при этом хоть и возможна работа пользователей, но время отклика ухудшается.

На Yaffil резервное копирование и восстановление происходит в несколько раз быстрее, особенно заметно это на больших базах данных[vi] (отсутствует деградация скорости восстановления со временем).

Индексы по выражениям

Индексы по выражениям (Expression Indexes[4]) - это еще одна интересная возможность, так и не реализованная полностью в InterBase. Индексы по выражениям используются в тех случаях, когда необходимо обеспечить быстрый поиск или сортировку по значениям, вычисляемым на основе полей таблицы. Необходимый индекс определяется следующим образом:

CREATE [UNIQUE] [ASC[ENDING] | DESC[ENDING]]

INDEX index ON table COMPUTED BY (expression)

---

[3] План для условия OR указывает не на количество используемых битовых карт индекса, а на количество операций выборки по битовой карте

[4] SF: 446253 Expression based indexes

где index - имя индекса, expression - выражение, построенное на основе полей таблицы. В выражении можно использовать арифметические операции, встроенные функции и UDF. Во время оптимизации запросов, если выражение совпадает с выражением, указанном в условии WHERE или в условии соединения, будет использован индекс.

Например, необходимо сделать выборку записей, имеющих поле типа DATE, для определенного месяца по всем годам (на примере БД Employee из комплекта поставки InterBase 6):

CREATE INDEX employee_hire_date_month_idx ON employee COMPUTED BY (EXTRACT(MONTH FROM hire_date))

Попробуем выполнить запрос и посмотрим на выбранный оптимизатором план:

SELECT * FROM employee

WHERE EXTRACT(MONTH FROM hire_date) = 1

PLAN (EMPLOYEE INDEX (EMPLOYEE_HIRE_DATE_MONTH_IDX))

Вычисляемое выражение должно полностью определяться значениями полей таблицы. Индекс на основе, например, функции CURRENT_TIMESTAMP, скорее всего, будет бесполезен, хотя Yaffil не запрещает использование подобных выражений. В вычисляемых выражениях нельзя использовать подзапросы. Очень интересные возможности появляются при использовании User-Defined Functions (UDF) в вычисляемых выражениях. С их помощью можно выполнять эффективный поиск по практически любому условию.

Уменьшение размера занимаемого индексами

Индексы, построенные по текстовым полям с национальным порядком сортировки, занимают в среднем на одну треть меньше места на диске по сравнению с Interbase.

Удобная операция объединения строк

Yaffil позволяет объединять строки с использованием оператора "||" при превышении размера результата максимальной длинны строки. При этом результирующая строка будет иметь максимально допустимый сервером размер, а ошибка переполнения возникает, только если действительные данные пользователя не могут быть размещены в результирующей строке. Такое поведение оператора "||" соответствует привычному поведению обычных арифметических операторов над числами.

В качестве примера приведём вариант с объединением двух полей: CREATE TABLE T(V1 VARCHAR(20000), V2 VARCHAR(20000)). При попытке написать V1 || V2 Interbase/Firebird выдадут ошибку переполнения ещё на этапе компиляции. Сервер Yaffil в качестве результата сформирует строку VARCHAR(32765) . Ошибка переполнения возникнет, только если количество символов объединения V1 и V2, исключая концевые пробелы, превысит 32765.

Широкие возможности указания пользовательских планов

Не всегда встроенный оптимизатор может выбрать оптимальный план. Причиной этого может быть отсутствие подробной статистики по индексам и полям, без которой трудно оценить стоимость выполнения варианта или слишком большое число вариантов соединений. В таких случаях приходится применять явные планы в запросах.

Yaffil расширяет возможности использования явных планов, тем самым, предоставляя возможность дальнейшего ускорения работы приложений.

В Yaffil появилась возможность указывать явные планы в не курсорных операторах обновления данных, таких как UPDATE и DELETE. Дополнительно, этим можно добиться обновления или удаления строк данных в заданном порядке, указав использование индекса в условии ORDER.

Как известно, в InterBase не разрешается использование явных планов в тексте триггеров. Yaffil снимает это ограничение.

Имена индексов ограничений

Использование явных планов в Yaffil в триггерах и процедурах существенно упрощается благодаря возможности именования индексов, автоматически создаваемых сервером для ограничений первичных, внешних ключей и ограничений уникальности. В версиях InterBase такие индексы приобретают системные имена в формате RDB$PRIMARYX для индексов первичных ключей, RDB$FOREIGNX для индексов внешних ключей и RDB$X для индексов ограничений уникальности. X обозначает номер индекса данного типа по порядку с момента создания базы данных (или восстановления из резервной копии). Так как эти номера могут измениться при следующем восстановлении базы данных, фиксировать такие индексы в плане становится опасным. Возникает тупиковая ситуация: автоматический план неэффективен, явный план записать нельзя. Разработчикам приходится создавать собственные индексы, целиком дублирующие системные. Однако появление новых индексов на таблице влечет к увеличению дискового пространства, занимаемого базой, замедлению операций обновления, и кроме того к повышению вероятности выбора оптимизатором неверного плана из-за увеличения числа вариантов соединений таблиц.

В Yaffil индексы для ограничений могут принимать имена соответствующих ограничений[5]. Нужное поведение включается параметром CONSTRAINT_INDEX_NAME в конфигурационном файле. Например:

```
SQL> create table T (id into not null, constraint PK_T primary key (id))

SQL> show index

SQL> PK_T UNIQUE INDEX ON T(ID)
```

Улучшенное время отклика для версии Superserver

В архитектуре Superserver одновременное обслуживание нескольких клиентов реализовано по схеме многопоточного сервера. Однако переключение процессора между потоками (диспетчеризация) происходит не по требованию операционной системы, а в моменты времени, выбираемые активным потоком "добровольно". Такая схема очень похожа на реализацию многозадачности в Windows 3.1 и называется не вытесняющей многозадачностью. С каждым потоком связано числовое значение, первоначально равное величине кванта времени. При прохождении потоком через определенные точки в коде это значение уменьшается на единицу, пока не достигнет нуля. В этот момент квант времени считается исчерпанным, и активный поток передает управление другому. Ясно, что обеспечить равный доступ каждого потока к процессору при таком подходе невозможно. На практике эта проблема проявляется в резком увеличении времени выполнения оперативных запросов при одновременном выполнении долгих и тяжелых запросов, например, аналитики.

В сервере Yaffil данная проблема ослаблена за счет введения дополнительных точек переключения в наиболее часто повторяющихся участках кода сервера. Для рабочих

[5] SF: 451925 User names for PK and FK indexes

потоков сервера и отдельно для потока сборки мусора есть возможность задавать величину кванта времени с помощью параметров конфигурации THREAD_QUANTUM и SWEEP_THREAD_QUANTUM. Параметр FORCE_RESHEDULE активизирует дополнительные точки переключения. В результате распределение процессорного времени между рабочими потоками происходит более равномерно.

Улучшенный протокол локальных соединений (XNET)

Локальное соединение в InterBase выполняется с использованием буферов разделяемой памяти и обеспечивает более высокую производительность по сравнению с другими протоколами (TCP и Named Pipes). В документации это называется локальным протоколом или IP Server (IPS) по внутренней терминологии Borland. Вместе с тем локальное соединение в InterBase обладает двумя серьезными недостатками, часто делающими его использование невозможным. Прежде всего, локальное соединение не допускает одновременной параллельной (многопоточной) работы нескольких соединений. Второе клиентское соединение будет заблокировано до момента завершения первого.

Кроме того, для установления соединения используется оконное сообщение, посылаемое клиентом специальному скрытому окну сервера. Если сервер InterBase работает как служба NT и клиентское приложение также работает как служба, они могут использовать разные desktops, при этом посылка оконных сообщений между ними невозможна.

Типичный пример подобной конфигурации - веб-сервер IIS, выполняющий приложения, обращающиеся к базам данных InterBase. Традиционно в таких случаях рекомендуется использовать протокол TCP для установления соединений из сервисов NT, что отрицательно сказывается на производительности.

XNET также является реализацией локального соединения с использованием буферов разделяемой памяти, поэтому его скорость передачи данных идентична "старому" локальному протоколу (IPS). Отличие состоит в том, что его реализация допускает одновременное использование на разных потоках без взаимного блокирования. Для установления соединения больше не используются окна и оконные сообщение, поэтому соединения надежно устанавливаются из служб Windows NT.

Оригинально XNET был разработан специалистами фирмы Борланд и планировался как замена ненадежному старому локальному протоколу. Начало разработки датировано в исходном коде 1995 годом. Однако до сих пор не было принято решение о включении XNET в текущие версии. Возможно, что XNET планировался для включения в версию InterBase 7.0 при условии реализации на платформах UNIX.

В сервере Yaffil по сравнению с оригинальным кодом внесены изменения, направленные на повышение надежности работы в случае неожиданного "падения" одной из сторон. Возможность соединения из служб NT также впервые была реализована в Yaffil.

Строка соединения по XNET аналогична обычной строке локального соединения, при этом необходимо указать имя файла базы данных. Протокол XNET не совместим по локальному подключению с другими версиями InterBase/Firebird, поэтому необходимо использовать клиентскую библиотеку (GDS32.DLL), соответствующую версии сервера. Протокол XNET не доступен в Yaffil Classic Server.

Ограничение времени ожидания для транзакций (Lock timeout)

При возникновении конфликта обновления записи в InterBase возможны два варианта поведения транзакции, задаваемых параметром WAIT (isc_tpb_wait / isc_tpb_no_wait) - бесконечное ожидание разрешения конфликта или немедленная выдача ошибки. Режим с ожиданием часто удобнее, так как нет необходимости повторять операцию в случае конфликта, но такой режим является очень опасным из-за возможности бесконечной блокировки приложения.

В сервере Yaffil добавлена возможность ограничивать время ожидания разрешения конфликта заданным интервалом времени. Для этого служит параметр конфигурации LOCK_TIMEOUT, задающий время в секундах. Положительное значение от 1 до 32767 определяет время ожидания WAIT транзакций. Отрицательное число определяет бесконечное время ожидания. Нулевое значение эффективно превращает WAIT транзакции в NOWAIT. Значение по умолчанию -1 (минус один), что обеспечивает совместимость с другими версиями. Параметр не оказывает влияния на транзакции, запущенные в режиме NOWAIT.

В следующих версиях планируется ввести константу блока TPB (transaction parameter block), управляющую временем ожидания при запуске каждой транзакции индивидуально.

Расширения SQL

В сервере Yaffil реализовано несколько дополнительных языковых конструкций SQL по сравнению с Interbase/Firebird.

Инструкция IIF

Инструкция IIF позволяет реализовать дополнительную логику в запросах.

Синтаксис: IIF '(' search_condition ',' value_if_true ',' value_if_false ')'. Выполняя инструкцию IIF, сервер вычисляет выражение search_condition. Если search_condition, то результатом IIF является выражение value_if_true, в противном случае value_if_false.

Пример:

```
select iif(rc.rdb$collation_id = 0, 'ДА', 'НЕТ') from rdb$collations rc
where rc.rdb$collation_name = 'WIN1251'
```

Выполнив запрос, получим - «ДА»

Инструкцию IIF можно применять и при вычислении выражений.

Пример:

```
a = b + iif(c is null, 0, c);
```

Инструкция CASE

Инструкция CASE позволяет реализовать дополнительную логику в запросах.

Синтаксис:

```
<case expression> ::= <case abbreviation> | <case specification>
<case abbreviation> ::=
    NULLIF <left paren> <value expression> <comma>
        <value expression> <right paren>
```


| COALESCE <left paren> <value expression>
{ <comma> <value expression> }... <right paren>

<case specification> ::=
<simple case>
| <searched case>

<simple case> ::=
CASE <case operand>
<simple when clause>...
[<else clause>]
END

<searched case> ::=
CASE
<searched when clause>...
[<else clause>]
END

<simple when clause> ::= WHEN <when operand> THEN <result>
<searched when clause> ::= WHEN <search condition> THEN <result>
<else clause> ::= ELSE <result>
<case operand> ::= <value expression>
<when operand> ::= <value expression>
<result> ::= <result expression> | NULL
<result expression> ::= <value expression>

Инструкция NULLIF

Инструкция NULLIF позволяет реализовать дополнительную логику в запросах.

Синтаксис: NULLIF (V1, V2) эквивалентно CASE WHEN V1=V2 THEN NULL ELSE V1 END, или с использованием инструкции IIF – IIF(V1=V2, NULL, V1)

Инструкция COALESCE

Инструкция COALESCE позволяет реализовать дополнительную логику в запросах.

Синтаксис:

COALESCE (V1, V2) эквивалентно CASE WHEN V1 IS NOT NULL THEN V1 ELSE V2 END, или с использованием инструкции IIF – IIF(V1 IS NOT NULL, V1, V2)

COALESCE (V1, V2, . . . ,n) эквивалентно CASE WHEN V1 IS NOT NULL THEN V1 ELSE COALESCE (V2, . . . ,n) END

Инструкция INSERT INTO … FROM … UNION …

Сервер Yaffil, в отличие от InterBase/Firebird, позволяет использовать объединения UNION для формирования данных на вставку.

Пример.

insert into t_a (id) select b.id from b union select c.id from c

Инструкция LEAVE

Инструкция LEAVE[6] вызывает выход из циклов WHILE и FOR SELECT. Использование и синтаксис аналогичен оператору Break в Delphi.

Выражения в EXCEPTION

Сервер Yaffil расширяет синтаксис инструкции EXCEPTION[7]. Допускаются три варианта использования:

1. EXCEPTION ‘исключение’;

   Этот вариант соответствует синтаксису Interbase/Firebird – сервер выбрасывает исключение заданное соответствующим идентификатором.

2. EXCEPTION ‘исключение’ ‘выражение’;

   В этом случае сервер также выбрасывает исключение, но его текст заменяется на результат вычисления выражения.

3. EXCEPTION;

   Сервер выбрасывает последнее сформированное исключение. Если до выполнения этой инструкции исключений выброшено не было, то инструкция игнорируется. Этот вариант используется для повторного выбрасывания исключения в обработчиках ошибок WHEN. Инструкция может быть использована для формирования секций защиты ресурсов, выделяемых в UDF.

Системные переменные ROW_COUNT, GDSCODE, SQLCODE, CURRENT_TRANSACTION, CURRENT_CONNECTION, CURRENT_DATABASE, CURRENT_POOL, INSERTING, UPDATING, DELETING

- Переменная ROW_COUNT[8] содержит количество записей модифицированных записей в результате выполнения последнего запроса.
- Переменная GDSCODE содержит значения инструкции GDSCODE в обработчике WHEN, однако может использоваться вне контекста WHEN
- Переменная sqlcode содержит значения инструкции sqlcode в обработчике WHEN, однако может использоваться вне контекста WHEN
- Переменная CURRENT_TRANSACTION[9] содержит номер транзакции
- Переменная CURRENT_CONNECTION содержит номер подключения

[6] Leave и Exit недоступны в коде триггеров

[7] SF: 446240 Exception code and text at runtime

[8] SF: 451927 ROWSAFFECTED system variable

[9] SF: 446243 Transaction ID

- Переменная CURRENT_DATABASE содержит имя файла БД. Доступна только в коде тригеров и процедур.
- Переменная CURRENT_POOL содержит рукоятку[10] менеджерам памяти и может быть использована для выделения памяти в UDF, автоматически освобождаемой по окончании выполнения запроса.
- Логические переменные INSERTING, UPDATING, DELETING указывают на тип исполняемого триггера.

Группировка по номеру столбца

Сервер Yaffil расширяет синтаксис инструкции GROUP BY. Допускается указывать номера столбцов для группировки как в инструкции OREDER BY.

Пример:

```
select count(a), b from t group by 2
```

Значения переменных по умолчанию

Сервер Yaffil расширяет синтаксис инструкции DECLARE VARIABLE. Можно не указывать ключевое слово VARIABLE и дополнительно указать инициализирующее значение переменной.

Пример:

```
declare k = 0;
```

Тип данных BIGINT

Дополнительный тип данных BIGINT является аналогом типа данных NUMERIC(18,0)[11], но предлагает более лаконичное и понятное название для 64-битного целого.

Тип данных __MONEY (__NMONEY)

Дополнительные тип данных __MONEY(n)/__NMONEY(n) являются «денежными» типами. Внутренне представление эквивалентно NUMERIC(18,N), однако при арифметических операциях точность вычисляется как максимум из точностей операндов. При этом для типа __MONEY действует правило бухгалтерского округления.

[10] Первый аргумент вункций HeapAlloc, HeapFree, HeapSize, HeapReAlloc

[11] SF: 446206 INT64 datatype alias for Numeric(18,0)

Дополнительные национальные кодовые страницы и порядки сортировки

В Yaffil добавлены следующие национальные кодовые страницы и порядки сортировки:

- CS_WIN1257 Страны Прибалтики, кодовая страница Windows - 1257

  Порядки сортировки:

  - WIN1257_EE Эстония
  - WIN1257_LV Литва
  - WIN1257_LT Латвия
- CS_KOI8R Россия KOI8, кодовая страница Windows - 20866

  Порядки сортировки:

  - KOI8R
  - KOI8R_RU
- CS_KOI8U Украина KOI8-U, кодовая страница Windows - 21866

  Порядки сортировки:

  - KOI8U
  - KOI8U_UA

Так же для кодовой страницы WIN1251 добавлены порядки сортировки:

- WIN1251_UA
- WIN1251_RU

Группировка по встроенным функциям и UDF

Разрешена группировка и использование встроенных функций и UDF[12].

Пример.

select sum(vent) from sales group by extract(year from sale date)

Ограничение результатов выборки FIRST/SKIP

Результат выборки может быть ограничен с использованием инструкций FIRST/SKIP. При значении аргумента FIRST, равном 0, результат выборки пустое-множество, в отличие от Firebird, выбрасывающего исключение с сообщением о неверном параметре.

Увеличение глубины рекурсии процедур и триггеров

Количество рекурсивных вызовов процедур и триггеров увеличено до 1000.

Выражения в значениях по умолчанию для доменов

Yaffil позволяет использовать сложные выражения для значений по умолчанию в доменах: CREATE DOMAIN NEW_DOMAIN AS INTEGER DEFAULT (GEN_ID(NEW_GENERATOR, 1))

[12] SF: 546274 group by EXTRACT function doesn't work, SF: 555839 group by needs UDF's name

Подобная возможность позволяет не писать дополнительные триггеры для выполнения схожих действий.

SET GENERATOR, SET STATISTICS в PSQL

Инструкции SET GENERATOR, SET STATISTICS можно использовать в коде процедур.

Расширения выражения IN

В выражении IN допускается использование параметров и подзапросов.

Универсальные триггеры INSERT, UPDATE, DELETE

Yaffil позволяет создавать универсальный триггеры[13], заданные одновременно для нескольких операций.

Пример:

```
CREATE TRIGGER ADDRESS_ADU FOR ADDRESS AFTER DELETE OR
UPDATE
```

Сортировка подзапросов

Сервер позволяет указать инструкцию ORDER BY в подзапросах[14]. Сортировка подзапроса позволяет более эффективно использовать инструкции FIRST/SKIP в подзапросах.

Выражения в аргументах процедур, функции substring

Yaffil позволяет использовать выражения в качестве аргументов[15] при вызове процедур, функции substring.

Арифметические операции над типами DATE, TIME, TIMESTAMP

Yaffil расширяет стандартный набор операций над типами данных DATE, TIME, TIMESTAMP при помощи взаимообратного преобразования к типам INTEGER и DOUBLE. Возможные преобразования: DATE из/в INTEGER, TIME из/в INTEGER, TIMESTAMP из/в DOUBLE.

Поддержка NULL для ограничений UNIQUE

В соответствии с требованиями стандарта SQL92 Yaffil позволяет использовать NULL для ограничений UNIQUE, а также в уникальных индексах.

Поддержка функцией UPPER порядка сортировки WIN151

Yaffil позволяет корректно обрабатывает порядок сортировки WIN1251 в функции UPPER. Единственно ограничение, накладываемое на порядок сортировки WIN1251 это положение буквы «Ё» при сортировке.

---

[13] SF: 451922 Universal Insert/Delete/Update triggers

[14] SF: 523452 Allow ORDER BY in subquery

[15] SF: 437859 Execute procedure and string concat

Сортировка по выражениям

Yaffil расширяет возможности инструкции ORDER BY. Кроме стандартных возможностей сортировать по столбцу или его номеру, допускается использовать выражения (выражения должно быть заключено в скобки).

Сокращённые вычисления логических условий

Как известно в языке С используется сокращённый вариант вычисления логических выражений. Т.е. если при вычислении части выражения уже однозначно известен результат, то оставшаяся часть не вычисляется. Подобный вариант расчёта логического выражения доступен в сервере Yaffil. Режим расчёта логических выражений задаётся параметром -bool утилиты gfix и действует в рамках заданной БД.

Задание типа параметра в DSQL

В DSQL Yaffil позволяет принудительно указать тип параметра, используя инструкцию CASE.

Пример:

```
SELECT * FROM RDB$DATAVASE WHERE (:MY_PARAM AS INTEGER) IS NULL
```

Встроенные функции

Yaffil содержит значительный список встроенных функций, позволяющих отказаться от использования UDF. Это позволяет более эффективно решать прикладные задачи, а так же упростить создание приложений.

Строковые функции TRIM, PAD, POSITION, LEFT, RIGHT, REVERSE, REPLACE, STUFF, TRANSLATE, SIMILAR

Функция TRIM удаляет ведущие и/или конечные символы из строки. Её синтаксис таков:

TRIM ({ {LEADING|TRALING|BOTH} <string expression> FROM }<string expression>)

Если не один из параметров LEADING|TRALING|BOTH не задан, то принимается параметр по умолчанию BOTH. Если задан только один параметр <string expression>, то первый параметр считается равным BOTH, а второй единичному символу пробел.

Пример:

TRIM(‘A’ FROM ‘A134FA’) результат ‘134F’

TRIM(‘ A134FA ’) результат ‘A134FA’

TRIM(LEADING ‘A’ FROM ‘A134FA’) результат ‘A134F’

TRIM(TRAILING ‘ABC’ FROM ‘A134FABC’) результат ‘A134F’

Функция PAD противоположна функции TRIM. Она дополняет строку заданными символами до необходимого размера. Также как и функция TRIM, она может дополнять ведущими символами и/или конечными символами. Синтаксис:

PAD ({ {LEADING|TRALING|BOTH} <string expression> FROM }<string expression> FOR <numeric expression>)

Если не один из параметров LEADING|TRALING|BOTH не задан, то принимается параметр по умолчанию BOTH. Если задан только один параметр <string expression>, то

первый параметр считается равным BOTH, а второй единичному символу пробел. Параметр <numeric expression> задаёт необходимое количество символов новой строки.

Пример:

PAD('A' FOR 11) результат ' A '

Функция POSITION производит поиск позиции первого вхождения подстроки в строке. Синтаксис:
POSITION (<substring> IN <string>)

Функция LEFT[16] возвращает заданное количество символов с начала строки. Синтаксис:
LEFT (<string>, <length>)

Функция RIGHT[17] возвращает заданное количество символов с конца строки. Синтаксис:
RIGHT (<string>, <length>)

Функция REVERSE[18] возвращает "перевёрнутую" строку. Т.е. символы строки возвращаются в обратном порядке. Синтаксис:
REVERSE (<string>)

Функция REPLACE возвращает строку, в которой все вхождения заданной подстроки заменены указанной замещающей строкой. Синтаксис:
REPLACE (<string>, <old substring>, <new substring>)

Функция STUFF[19] является разновидностью функции REPLACE. Возвращает строку в которой подстрока с заданной позиции и длины заменена на новую подстроку. Синтаксис:
STUFF (<string>, <start>, <length>, <substring>)

Функция TRANSLATE[20] является разновидностью функции REPLACE. Напомним, что последняя заменяет все вхождения заданной подстроки другой подстрокой, т.е. она обрабатывает не отдельные символы, а последовательности. В отличии он неё функция TRANSLATE выполняет указанное действие посимвольно, заменяя n-й символ искомой последовательности m-м символом замещающей последовательности. Синтаксиси:
TRANSLATE (<string>, <substring>, <substring>)

Пример:

TRANSLATE('SLKRJTKSBDV', 'RKLSB', '12345') результат '4321JT245DV'

Функция SIMILAR определяет коэффициент похожести двух строк в диапазоне 0..100. Если две строки идентичны, то результат равен 100. Синтаксис:
SIMILAR (<string1>, <string2>)

---

[16] Аналог функции LEFT в реализации MSSQL

[17] Аналог функции RIGHT в реализации MSSQL

[18] Аналог функции REVERSE в реализации MSSQL

[19] Аналог функции STUFF в реализации MSSQL

[20] Аналог функции TRANSLATE в реализации PL/SQL

Математические функции ABS, ACOS, ASIN, ATAN, ATAN2, COS, COSH, COT, LN, LOG, LOG10, CEILING, FLOOR, PI, SQRT, EXP, POWER, TRUNC, ROUND, MOD

- ABS – абсолютное значение заданного числа
- ACOS – арккосинус
- ASIN – арксинус
- ATAN – арктангенс
- ATAN2 – арктангенс, через отношение первого аргумента ко второму
- COS – косинус
- COSH – гиперболический косинус
- COT – возвращает 1 / TAN
- LN – натуральный логарифм
- LOG – логарифм числа по заданному основанию
- LOG10 – десятичный логарифм
- CEILING – наименьшее целое число, которое больше или равно заданному значению
- FLOOR – наибольшее целое число, которое меньше или равно заданному
- PI – число pi
- SQRT – квадратный корень
- EXP – число *e* в заданной степени
- POWER – возведение числа в заданную степень
- TRUNC – усечение числа до целого
- ROUND – округление числа до заданного количества знаков после десятичной запятой
- MOD – остаток от деления

Числовые функции BIN_AND, BIN_OR, BIN_XOR, BIN_SHR, BIN_SHL, RAND

- BIN_AND – побитовая операция AND
- BIN_OR – побитовая операция OR
- BIN_XOR – побитовая операция XOR
- BIN_SHR – двоичный сдвиг вправо
- BIN_SHL – двоичный сдвиг влево
- RAND – случайное число в диапазоне 0..1

Функции обработки даты-времени DATEADD, DATEDIFF[21]

Функция DATEADD изменяет дату на заданный интервал.

Функция DATEDIFF вычисляет разницу между датами в указанных еденицах измерения.

Функции HASH, GEN_UUID, MINVALUE, MAXVALUE

Функция HASH возвращает 64-битный хеш алгоритма ELF. Если входной параметр не является строкой, то предварительно выполняется преобразование сервером в строку аналогично оператору CAST.

[21] Аналоги функции DATEADD, DATEDIFF в реализации MSSQL

Функция GEN_UUID возвращает UUID в формате 38 символьной строки. Формат совместим с типом TGuidField.

Функция MINVALUE возвращает минимальное из заданных значений. Тип результата вычисляется аналогично инструкции COALESCE.

Функция MINVALUE возвращает максимальное из заданных значений. Тип результата вычисляется аналогично инструкции COALESCE.

Использование переменной окружения ISC_PATH

В Yaffil расширены возможности использования переменной окружения ISC_PATH для задания префикса к пути базы данных. Переменная ISC_PATH используется, если в имени базы данных, указываемом клиентом, не содержит каталогов или имени сервера. Переменная ISC_PATH может использоваться как на клиенте, так и на сервере.

Использование ISC_PATH на сервере

Пусть базы данных на сервере находятся в каталоге c:\database. Определим переменную ISC_PATH=c:\database. Далее можно использовать на клиентском компьютере строку соединения вида servername:database.gdb для открытия базы данных c:\database\database.gdb.

Использование ISC_PATH на клиенте

Пусть базы данных располагаются на сервере servername в каталоге c:\database. Определим на клиенте переменную ISC_PATH как servername:c:\database. После этого клиент сможет обращаться к базам данных только по короткому имени файла БД, например, CONNECT sales.gdb.

Используем внешний файл в базе sales.gdb: CREATE TABLE customers EXTERNAL FILE "sales_files/customers.txt" (...);

Безопасная работа с внешними таблицами

Внешние файлы могут располагаться ТОЛЬКО в одном из каталогов, разрешенных конфигурационным параметром EXTERNAL_FILE_DIRECTORY[22], а также в их подкаталогах. По умолчанию конфигурационный файл не содержит параметров EXTERNAL_FILE_DIRECTORY, тем самым использовать внешние файлы вообще не разрешается. Если нужно полностью снять ограничения на размещение внешних файлов, следует задать корневые каталоги дисков в EXTERNAL_FILE_DIRECTORY, например:

EXTERNAL_FILE_DIRECTORY "c:\"

EXTERNAL_FILE_DIRECTORY "d:\"

Таким образом, любой файл на диске становится доступным через внешние таблицы, что соответствует прежнему поведению Interbase\Firebird. Из-за серьезных проблем с безопасностью такой подход крайне НЕ РЕКОМЕНДУЕТСЯ. Каждому каталогу, сконфигурированному под внешние файлы, может быть присвоено логическое имя.

Логическое имя каталога необязательно и задается вторым аргументом:

EXTERNAL_FILE_DIRECTORY "c:\databases\files" myfiles

[22] SF: 451950 External table paths

На логическое имя можно ссылаться при задании внешней таблицы. Для этого нужно указать логическое имя каталога после символа $ в начале имени внешнего файла. Например:

CREATE TABLE customers EXTERNAL FILE "$myfiles/customers.txt" (...);

Другой способ задания внешних файлов состоит в указании абсолютного пути к файлу. При этом файл также может располагаться только в разрешенных каталогах. Например:

CREATE TABLE customers EXTERNAL FILE "c:/databases/files/customers.txt" (...);

И, наконец, можно задавать файл с указанием имени, относительно каталога базы данных:

CREATE TABLE customers EXTERNAL FILE "files/customers.txt" (...);

При этом каталог, в котором находится файл, должен быть разрешен параметром EXTERNAL_FILE_DIRECTORY. НЕ РЕКОМЕНДУЕТСЯ разрешать для размещения внешних файлов каталоги с базами данных, поскольку в этом случае пользователи получают доступ к файлам этих баз данных. Если желательно располагать внешние файлы рядом с БД, следует создать подкаталог для внешних файлов для каждой базы данных и разрешить его в EXTERNAL_FILE_DIRECTORY. Допустим, все БД лежат в c:\databases, среди которых sales.gdb. Создаем каталог sales_files, разрешаем его для использования:

EXTERNAL_FILE_DIRECTORY "c:\databases\sales_files"

Режим точности при операций над типом NUMERIC

По умолчанию, при умножении и делении переменных типа NUMERIC масштаб результата вычисляется как сумма масштабов исходных аргументов. Это позволяет избежать потери точности, однако при последовательных операциях умножения, деления в одном выражении, такой алгоритм вычисления масштаба результата, легко приводит к переполнению.

В сервере Yaffil существует специальный параметр подключения к БД - isc_dpb_numeric_scale_reduction[23]. С использованием этого параметра масштаб вычисляется как максимум масштабов + 1.

Создание пользовательских кодовых страниц и порядков сортировки

Сервер Yaffil не поддерживает расширение предопределённых порядков сортировки посредством интерфейса gdsintl2.dll. Вместо этого предлагается альтернативный способ, основанный на UDF.

Для создания новой кодовой страницы необходимо создать зарегистрировать следующие UDF:

```
cdecl USHORT user_text_type_xxx(TEXTTYPE, USHORT, USHORT);
DECLARE EXTERNAL FUNCTION USER_TEXTTYPE_XXX
    SMALLINT, CHAR(1)
    RETURNS PARAMETER 2
```

[23] Значение константы isc_dpb_numeric_scale_reduction равно 70

```
cdecl USHORT user_translate_xxx_xxx (CSCONVERT, USHORT, USHORT);
DECLARE EXTERNAL FUNCTION USER_TRANSLATE_XXX_XXX
    SMALLINT, SMALLINT,  CHAR(1)
    RETURNS PARAMETER 3

cdecl USHORT user_char_set_xxx(CHARSET, USHORT, USHORT);
DECLARE EXTERNAL FUNCTION USER_CHARSET_XXX
    SMALLINT,  CHAR(1)
    RETURNS PARAMETER 2
```

где XXX – трёхзначный номер кодовой страницы/порядка сортировки.

Ограничение максимального времени выполнения запроса (Max_Request_Time)

Yaffil поддерживает возможность ограничить максимальное время выполнения запроса с клиента. Для этого служит параметр конфигурации MAX_REQUEST_TIME, задающий время в секундах. Положительное значение от 1 до 32767 определяет максимальное время выполнения запроса. Отрицательное число отменяет ограничение.

Сравнение сервера Yaffil с другими клонами Intebase

Не стоит, однако, думать, что переход на Yaffil обязательно приведет к значительному ускорению работы приложения. Архитектура доступа к базам данных состоит из нескольких уровней, каждый из которых требует определенных расходов на передачу и обработку данных и приводит к дополнительным задержкам. Если ваше приложение работает с базой данных через сетевое соединение, использует дополнительные интерфейсы высокого уровня, такие как ODBC и ADO, разница во времени ответа, возможно, будет не такой заметной.

Тем не менее, важно то, что оптимизация кода СУБД эффективно снижает загрузку серверной стороны системы, тем самым, увеличивая количество операций, которые может параллельно обработать сервер на данной аппаратуре. Особенно это касается "тяжелых" запросов или процедур.

Классическая архитектура на Windows NT (Yaffil CS)

Реализация классической архитектуры Yaffil CS на платформе Windows NT является значительным преимуществом сервера Yaffil по сравнению с другими вариантами InterBase/Firebird, существующими на сегодняшний день. Классическая ветвь InterBase для Windows NT прекратила развитие в 1994 году в связи с началом разработки варианта SuperServer для версии 4.0. Предполагалось, что SuperServer быстро заменит классическую архитектуру, однако реализация SuperServer, имеющая приемлемые характеристики при использовании с большой нагрузкой, не была удачной. В связи с этим до недавнего времени версии InterBase для платформ Solaris и Linux поставлялись в двух вариантах - SuperServer и Classic Server.

Классическая архитектура обладает следующими преимуществами:

- Равномерное распределение нагрузки между выполняющимися запросами на разных соединениях.
- Эффективное использование многопроцессорных систем (SMP).
- Более полное использование оперативной памяти.
- Встраиваемый (embedded или in-process) сервер.

Другими словами, сферой использования Classic Server являются высокопроизводительные системы, обслуживающие одновременно большое число подключений, в то время как SuperServer более эффективен на системах небольшого размера.

Не будем здесь останавливаться на отличиях архитектур Classic и SuperServer, подробно описанных выше. Рассмотрим лишь особенности реализации Yaffil Classic Server.

В отличие от сервера Interbase 4.0 для Windows NT, Yaffil CS способен запускаться не только как служба Windows NT, но и как приложение. Во втором случае в области system tray панели задач Windows появляется красная иконка Yaffil Server. В строке версии вместо пары букв WI (платформа Windows Intel) используются буквы NI (платформа NT Intel), поскольку именно так обозначала себя первая версия InterBase CS 4.0 для Windows NT.

Количество соединений, поддерживаемых Yaffil Classic Server, ограничено только ресурсами системы, в основном оперативной памятью. Ограничение для InterBase CS 4.0 в 90-120 соединений при запуске на не интерактивном десктопе, снято в Yaffil CS.

При установке Yaffil CS следует обратить внимание на настройку протокола TCP/IP для обеспечения остановки процессов ibremote.exe в случае обрыва соединения с клиентом.

Встраиваемый сервер

Как известно, первые версии сервера InterBase, работающие под операционной системой UNIX и другими, более экзотическими ОС, использовали прямое связывание кода сервера с клиентским приложением. Строго говоря, термины клиент и сервер здесь не очень уместны, так как существует всего один процесс.

Можно провести аналогию с технологией COM (Component Object Model), в которой используется термин "in-process server" (сервер внутри процесса) для обозначения компонентов, загружаемых в адресное пространство клиентского приложения. Такой способ загрузки обеспечивает максимальную эффективность за счет отсутствия накладных расходов, связанных с упаковкой параметров вызова (маршалинг), передачей упакованного блока данных в адресное пространство сервера с помощью некоторого транспортного механизма (сетевого протокола или буферов разделяемой памяти) и диспетчеризацией вызова обработчика запроса на серверной стороне.

Архитектура Yaffil Classic дает возможность приложениям использовать внутрипроцессный сервер Yaffil. Такое использование часто называют встраиваемым (embedded), подразумевая легковесность и упрощение тиражирования прикладных систем. По сравнению с традиционным сервером, внутрипроцессный Yaffil не требует запуска дополнительного процесса сервера или инсталляции служб NT. Приложению для работы требуется всего лишь одна библиотека динамической загрузки (DLL). Общий объем исполнимых модулей при этом также сокращается.

Однако встраиваемое использование подразумевает некоторые (возможно, значительные для вашего приложения) ограничения.

Встраиваемый сервер может использоваться только в однопоточных приложениях. Существующее ядро InterBase/Firebird/Yaffil не является безопасным для использования из нескольких потоков (thread-safe). Глобальные структуры данных сервера не защищены от одновременного изменения. Кроме того, внутри ядра широко используется локальное состояние потока. Таким образом, поведение сервера будет непредсказуемым при вызове функций сервера с нескольких потоков одновременно, а также при использовании соединений, первоначально открытых в другом потоке.

Если вы разрабатываете программы, работающие в среде сервера приложений или веб-сервера, таких как COM+ или IIS, то встраиваемый сервер для вас также непригоден, поскольку подобные среды используют собственное управление потоками.

Рассмотрим вопрос безопасности для базы данных. При нахождении кода сервера в составе клиентского приложения необходим полный доступ к файлу базы данных. В то же время нельзя гарантировать разграничение доступа на основе разрешений SQL к объектам БД. Поскольку код приложения имеет физическую возможность обращаться к любой области базы данных, ограничения SQL являются всего лишь джентльменскими соглашениями.

С другой стороны, между приложением и базой данных нет посредников, таких как сетевые устройства и средства межпроцессного обмена данными. Поэтому нет необходимости использовать средства защиты данных при клиент-серверном взаимодействии.

Использование сервера Yaffil внутри процесса.

С точки зрения прикладной программы, различие между встраиваемым сервером и обычным удаленным клиентом заключается в имени библиотеки динамической загрузки (DLL), связываемой с программой. Как известно, обычные приложения используют библиотеку GDS32.DLL, как правило, устанавливаемую в системный каталог Windows. Существование нескольких разных библиотек с одним именем может привести к путанице, особенно если подобная библиотека находится в пути доступа, общего для всех приложений. Версии 4.x InterBase CS, выпущенные фирмой Борланд, используют библиотеку сервера, которая также имеет имя GDS32.DLL.

Yaffil CS реализован в библиотеке YAENG32.DLL, имеющей интерфейс, идентичный GDS32.DLL. Поэтому использовать встраиваемый Yaffil CS можно в приложениях, написанных на IB API или Embedded SQL с помощью указания библиотеки импорта YAENG32.LIB на этапе связывания (linking).

Другая возможность использования нужной библиотеке состоит в динамической загрузке ее во время выполнения приложения.

Если же вы пользуетесь компонентами доступа или драйверами, не позволяющими указывать имя используемой библиотеки, вам остается только один выход - скопировать модуль YAENG32.DLL под именем GDS32.DLL. Не забудьте поместить этот файл в каталог, в котором находится исполнимый (.exe) модуль программы.

Эффективное взаимодействие процессов архитектуры Classic Server

В архитектуре Classic Server несколько серверных процессов совместно работают с одной базой данных, осуществляя координацию своих действий через разделяемую таблицу блокировок. Взаимодействие процессов на версиях InterBase, работающих под операционными системами Unix, осуществляется с использованием механизма сигналов, при этом сигнал передается не напрямую, а через специальный процесс менеджера блокировок. Такой механизм быстро становится неэффективным при увеличении числа

процессов, все большая часть времени процессора тратится на доставку и обработку сигналов.

В отличие от сигналов, в Yaffil на платформе Windows NT/2000 для обмена сообщениями о блокировках используются объекты синхронизации без обращений к ядру ОС. Во время активной работы сервера между процессами передается большое число таких сообщений, в результате заметно снижаются расходы на межпроцессное взаимодействие.

Изменения оптимизатора, направленные на совместимость

Благодаря улучшениям оптимизатора исключена ситуация, когда автоматически сгенерированный план запроса оказывается неверным, в то время как на более старых версиях он был правильным.

Yaffil Classic Server - замена InterBase Classic 4.0

InterBase CS 4.0 для операционной системы Windows NT до сих пор используется в системах, несущих большую нагрузку. Переход на более новые версии был невозможен в связи с тем, что архитектура Super Server недостаточно пригодна для работы с большим числом одновременных подключений, в то время как версии сервера архитектуры Classic Server перестали выпускаться Borland, начиная с версии 4.2.

Yaffil Classic Server - лучший вариант обновления сервера. Сохраняя все преимущества архитектуры Classic, с лучшей производительностью и надежностью, обладающий новыми возможностями последних версий линейки Interbase.

Миграция баз данных на Yaffil и обратно

При разработке сервера Yaffil большое внимание было уделено безболезненному переходу со всех версий линейки InterBase, начиная с версии 4.0. Это вызвано тем, что в настоящее время существует большое число инсталляций устаревших версий (4.2, 5.6) по причине затрудненного переноса приложений на новые версии InterBase. Проблемы состоят в недостаточной обратной совместимости более новых версий InterBase.

В то же время, при использовании сервера Yaffil как обновления для InterBase версий 4.x, 5.x, возможные проблемы сведены к минимуму. Как правило, приложение сразу успешно работает без переделок.

Перечислим изменения Yaffil, направленные на обратную совместимость:

- Поддержка баз данных, созданных в предыдущих версиях InterBase
- Режим обратной совместимости

Режим обратной совместимости

Данный режим включается установкой параметра конфигурационного фала LEGACY_DIALECT1 в 1. После этого для клиентов диалекта 1 компилятор SQL выражений будет поддерживать только возможности, существовавшие в InterBase версии 5.x. Поведение сервера при работе с клиентами диалектов 2 и 3 не меняется. Режим совместимости приводит к "освобождению" следующих ключевых слов:

COLUMN	DESCRIPTOR
CURRENT_USER	SKIP
ROW_COUNT	CURRENT_DATE
CURRENT_CONNECTION	EXTRACT

SUBSTRING	TYPE
CURRENT_ROLE	CURRENT_TIMESTAMP
FIRST	INSERTING
CURRENT_TRANSACTION	UPDATING
CURRENT_TIME	DELETING
LEAVE	RECREATE
CURRENT_DATABASE	

Уменьшенное количество зарезервированных слов

Следующие ключевые слова не являются зарезервированными в Yaffil, независимо от диалекта и режима совместимости (жирным шрифтом выделены слова, новые для InterBase версий 5.x и 6.x):

BIGINT		
FLOAT	INT	**MONTH**
NUMERIC	SMALLINT	**WEEKDAY**
BLOB	CSTRING	DEC
HOUR	INTEGER	NCHAR
REAL	**TIME**	**YEAR**
CHAR	DATE	DECIMAL
IIF	**MINUTE**	NUM
SECOND	**TIMESTAMP**	
CHARACTER	**DAY**	

Перенос базы данных с предыдущих версий на Yaffil

Рекомендуемый способ переноса баз данных на Yaffil состоит в резервном копировании на исходном сервере и восстановлении полученного файла на Yaffil. Рассмотрим этот процесс по шагам.

Резервное копирование. Производится на работающем сервере предыдущей версии. В качестве клиента используется InterBase Client, соответствующий предыдущей версии сервера, включая утилиту gbak и клиентскую библиотеку gds32.dll.

В случае, если перенос происходит на одной машине, необходимо остановить старый сервер и запустить Yaffil. Необходимо также обеспечить корректную загрузку клиентской библиотеки gds32.dll от Yaffil.

Восстановление базы данных на Yaffil. Должны использоваться клиентские утилиты (gbak.exe) и библиотека gds32.dll от Yaffil.

Возврат баз данных на предыдущие версии

При необходимости всегда есть возможность вернуться к более старой версии после работы с базой данных под Yaffil. Это выполняется резервным копированием базы данных утилитой gbak.exe предыдущей версии с соответствующей клиентской библиотекой с сервера Yaffil и последующим восстановлением на сервере предыдущей версии. Рассмотрим этот процесс подробно при работе на одном компьютере.

Исходное положение: на компьютере в разных каталогах имеется две инсталляции - одна Yaffil и одна InterBase. Чтобы обеспечить правильную загрузку gds32.dll соответствующими утилитами, для каждой инсталляции необходимо переписать файл gds32.dll соответствующей версии в каталог bin. При этом других файлов gds32.dll в системе быть не должно!

Резервное копирование. При запущенном сервере Yaffil утилитой gbak.exe от версии InterBase (при этом должен использоваться клиент от той же версии) производим бэкап базы данных по протоколу TCP:

"c:\Program files\Borland\IntrBase\bin\gbak.exe" -b localhost:c:\database\database.gdb c:\database\database.gbak

Остановка сервера Yaffil, запуск InterBase предыдущей версии.

Восстановление базы данных тем же клиентом от предыдущей версии:

"c:\Program files\Borland\IntrBase\bin\gbak.exe" -c c:\database\database.gbak c:\database\database.gdb

Исправленные ошибки

Сервер Yaffil содержит значительное исправление ошибок существующих в коде Interbase/Firebird 1.0. Ниже приводится список исправленных ошибок:

- долгое подключение под Windows XP
- серверный поток, работающий по Named Pipes, больше не переключается под учетную запись клиента, произведшего подключение
- малая глубина рекурсии триггеров
- неверный минимальный размер индекса при использовании collate
- неверное объединение строк с результатом >32765
- повторное построение битовой маски оптимизатором в обработке условия IN(список констант) или OR на один индекс[24]
- distinct внутри view влияет на внешний запрос[25]
- неверная нумерация параметров в подзапросах при вызове с клиента[26]
- ошибка группировки по встроенным функциям
- ошибка объединения пустой строки и агрегирующей функции[27]
- остановка сервера при выполнении инструкции WHERE cast (подзапрос) BETWEEN[28]
- остановка сервера при выполнении инструкции extract(month from cast(null as date))[29]

---

[24] SF: 222376 Horrible plan with a lot of OR conditions

[25] SF: 224810 DISTINCT propagates outside a VIEW

[26] SF: 496787 Parameters for inner SELECT mismatched, SF: 597769 Parameter order is wrong, SF: 496787 Parameters missmatch

[27] SF: 444763 Empty column names with aggregate funcs, SF: 562417 Aggregate concatenated empty char

[28] SF: 521947 server crash on simple query

- остановка сервера при выполнении инструкций в планах SORT/MERGE
- случайные остановки сервера при модификации процедуры, используемой в представлении
- неверная передача значений в триггеры с VIEW UNION[30]
- остановка сервера при выполнении запроса из [31]view с union
- долгая обработка точек отката (savepoint) в случае значительной модификации данных
- замедление работы при восстановлении больших БД
- замедление работы при обновлении большего числа строк в рамках одной транзакции
- неверная обработка сетевых ошибок при 2PC
- долгая обработка транзакции, модифицирующей значительное количество страниц
- неверные записи в RDB$DEPENDENS
- неверная реализация EXTERNAL_FILE_DIRECTORY
- неполный список параметров GFIX
- при указании IP адреса ошибочно используется вызов gethostbyname
- сбой клиента или остановка сервера при использовании сообщений (events)
- не освобождение памяти в коде сервера
- не освобождение памяти в коде клиента
- неверная обработка left join и join на view[32]
- неверная передача параметров типа float в UDF
- случайный сбой слова состояния сопроцессора
- неверная обработка исключений внутри циклов в триггерах
- ошибка в обработке 2PC
- невозможность работы с метаданными, ссылающимися на отсутствующие UDF[33]
- невозможность изменения SP с неверным BLR
- неверная обработка подзапросов с параметрами[34]
- ограничение “request size limit exceeded”

---

[29] SF: 538201 crash with extract from null as date

[30] SF: 489762 trigger on view with union receive nulls

[31] SF: 483795 Query using VIEW with UNION causes crash

[32] SF: 488343 Problem joining views

[33] SF: 509991 Drop objects with bad UDF references SF: 429594 GBAK UDF in 'COMPUTED BY' field

[34] SF: 627057 Variable subselects marked invariant

- удаление записи в триггере befor update[35]
- повторное обновление записи в триггере befor update[36]
- остановка сервера при выполнении группировки по некоррелированному подзапросу
- неверная обработка инструкции PLAN в выражении DELETE[37]
- неверная обработка инструкции PLAN в выражении CREATE VIEW[38]
- неверная обработка инструкции PLAN в триггерах[39]
- возможная обработка неверных выражений create table[40]
- неверная обработка индексированного выражения starting with с пустым параметром[41]
- неверная обработка индексированного выражения :param starting with field[42]
- остановка сервера при выполнении подзапроса из VIEW с UNION в выражении IN[43]
- остановка сервера при переполнении ресурсов оптимизатора[44]
- остановка сервера при регистрации событий[45]
- неверная обработка ORDER BY в выражениях с объединением процедуры и таблицы[46]
- возможность создания доменов идентичным уже существующим системным доменам[47]
- неоптимальный план при объединении таблиц через условие starting with
- неверная оптимизация подзапросов в выражении IN[48]
- неверная оптимизация left join и view[49]

---

[35] SF: 582425 DB crashes if trigger BU deletes own row

[36] SF: 625899 Bugcheck 291

[37] SF: 233644 cannot specify PLAN in UPDATE statement. (Сервер Yaffil позволяет указать план и в выражении DELETE)

[38] SF: 452120 CREATE VIEW ignores PLAN (Сервер Yaffil не допускает указание плана при создании представления)

[39] SF: 558364 Triggers fail to compile if PLAN used

[40] SF: 217042 IB doesn't validate weird constructions

[41] SF: 426607 'Starting with' and empty parameter

[42] SF: 736318 value STARTING WITH field fails when using indices

[43] SF: 660298 Server shuts down with following SQL

[44] SF: 576067 Lost connexion with Big request

[45] SF: 213460 Registering Events w/certain configuration crashes IB Server

[46] SF: 221921 ORDER BY has no effect, SF: 414420 wrong order by in table join storedproc

[47] SF: 638521 dsql allows creation of rdb$user

[48] SF: 213859 Subquery connected with 'IN' clause

[49] SF: 508594 LEFT JOIN with VIEWs not optimized

- неверная обработка инструкции WHERE CURRENT OF в триггерах
- неверная обработка подзапросов с view в выражении с IN[50]
- возможность создания уникального индекса на поле допускающем NULL[51]
- остановка сервера при передачи строковых предстаений DATE/TIME аргументов UDF в 3-м диалекте
- неверная обработка extract в операциях left join[52]
- неверная обработка UDF log[53]
- неверная обработка ltrim, rtrim[54]
- неверная обработка UDF при операциях left join[55]
- неверная обработка NULL-blob в UDF[56]
- остановка сервера при вычислении null в инструкции between
- неэффективное использование памяти при обработке точек отката внутри процедур
- случайный сбой сервера при операциях сортировки
- неверная обработка операций с BLOB полями и генераторов в транзакциях только для чтения[57]
- невозможно выполнить более 1000 раз процедуры с неполностью выбранным циклом for select в одной транзакции

Известные ошибки

Ниже приводится список наиболее серьёзных ошибок в коде Interbase/Firebird 1.0

- неверная обработка if(exists(...)), где в запросе есть group by или union
- неверная обработка строк OCTECTS с UDF
- утечка памяти в размере 40 байт на сервере при разрыве соединения на каждую активную дополнительную транзакцию[58]
- утечка ресурсов на клиенте при разрыве соединения
- неверная обработка left join[59]

---

[50] SF: 523589 View is affecting the result of a query

[51] SF: 221649 Unique index allowed on NULLABLE field

[52] SF: 784121 Expression evaluation not supported on LEFT JOIN

[53] SF: 775003 log(x, y) in fact returns log(y, x)

[54] SF: 774987 ltrim('') and rtrim('') return NULL; rtrim forgets 1st char

[55] SF: 728839 left join defeats udf by mangling a null descriptor

[56] SF: 544132 UDF with NULL input parameter

[57] SF: 750664 READ_ONLY database and generators

[58] Ошибка возникает в момент разрыва соединения вследствие сетевых ошибок, когда активны несколько транзакции

[59] SF: 495339 Left join fails with SQLCODE = -508

- неверная оптимизация подзапросов в выражении IN[60]
- неверная обработка пользовательских триггеров на системных таблицах[61]
- неверная оптимизация подзапроса в ANY/ALL[62]
- бесконечный цикл в обработке insert into A select * from A[63]
- неверная обработка OLD-значений в триггерах на view с агрегатными функциями[64]
- неверная обработка привилегий для таблиц с именами более 27[vii] символов[65]
- неверная оптимизация некоррелированных запросов в выражении SELECT[66]
- невозможно получить список подключенных пользователей в Yaffil Classic
- неполная остановка сервера Yaffil Classic
- возможность создания в запросе псевдонима, совпадающего с именем таблицы, также существующей в запросе
- неверная обработка внешнего объединения процедуры и таблицы[67]
- неверная обработка значений дат ранее 17.11.1858[viii] с использованием индексов[68] и сортировкой[69]
- отсутствие обработки зависимостей для UDF и генераторов в значениях по умолчанию доменов и полей
- неверная обработка индексов[ix] на условиях > и >=[70]
- парсер допускает вызов процедуры без указания параметров, даже если параметры существуют
- повторное использование индекса в выражении like или starting with при наличии одинаковых индексов
- медленная обработка NULL в индексах[71]
- возможность изменения генераторов в БД только для чтения[72]

---

[60] SF: 447002 Optimisation of subselects, SF: 479483 Bad treatment of FIRST/SKIP in subselect, SF: 447002 Optimisation on subselects

[61] SF: 497032 User Triggers on System Tables fail

[62] SF: 459059 index breaks = ANY result, SF: 543106 bug with ALL keyword

[63] SF: 408272 infinite insertion cycle

[64] SF: 513296 no old TRIGGER values in AGGREGATE VIEWS

[65] SF: 222375 Grants overwrite previous rdb$security_classes entries

[66] Ошибка не приводит к неправильному результату

[67] SF: 486370 left outer join with sp is bad

[68] SF: 619635 Field date index no return line date

[69] SF: 609661 invalid sort in indexed date fields

[70] SF: 223060 Slow processing of GREATER-THEN operator. Ошибка приводила к полному сканированию индекса и снижению быстродействия.

[71] SF: 750686 Wrong internal processing of NULLs in index

[72] SF: 750664 READ_ONLY database and generators

- ошибка восстановления gbak view с udf[73]

Возможности, планируемые к реализации в следующих версиях

Разработчики Yaffil не желают останавливаться на достигнутом. Кратко перечислим изменения, планируемые к реализации в следующих версиях Yaffil. По понятным причинам гарантировать наличие какой-либо конкретной функциональности или сроков реализации разработчики не могут.

Интегрированная безопасность (NT Integrated Security)

Предполагается использовать принцип single sign-on, при котором при соединении пользователя к базе данных используется учетная запись домена NT, под которой он зарегистрирован в системе. Для передачи информации о пользователе используется протокол аутентификации выбранного модуля безопасности (Security Package), например, NTLM в системах с доменами типа NT4 или Kerberos для доменов Windows 2000. Далее на уровне сервера учетная запись операционной системы отображается на имя пользователя.

Для использования этой возможности необходимо вхождение серверного и клиентского компьютера в домен NT. При использовании интегрированной безопасности пароль пользователя не передается в открытом виде по сети, возможно шифрование и подпись (confidentiality и privacy) сетевого трафика.

Подобные принципы организации безопасности используется в MS SQL, Sybase SQL Studio Anywhere и других СУБД.

Асинхронный сервер и отмена выполняющихся запросов

Текущие версии InterBase/Firebird выполняют клиентские запросы последовательно. Коммуникационное соединение при этом блокируется, не разрешая клиенту передавать дополнительные запросы до окончания выполнения. Доступная в настоящий момент возможность отмены выполняющегося запроса с клиента в InterBase 6.5 с использованием дополнительного (вторичного) соединение, по которому и передается запрос на отмену, не является очень хорошим решением, поскольку усложняет протокол взаимодействия клиента и сервера и требует дополнительного потока, запускаемого на сервере.

В то же время, используемые транспортные протоколы (TCP, Named Pipe и объекты разделяемой памяти) позволяют проводить асинхронную передачу пакетов данных.

В сервере Yaffil предполагается реализовать асинхронную модель сервера. При этом для передачи запросов на отмену операции будет использоваться основное соединение.

Клиентский интерфейс отмены запросов предполагается совместимым с аналогичным расширением API в версии InterBase 6.5.

Одновременный запуск нескольких копий сервера (multi-instancing)

Существующие версии InterBase/Firebird не допускают одновременный запуск нескольких процессов сервера. Причина этого в том, что сервер использует глобальные именованные объекты и структуры данных. При запуске сервера происходит проверка

[73] SF: 586397 GBAK doesn't restore Views with UDF's

на наличие уже запущенного экземпляра, однако, иногда такая проверка может дать сбой, например, в случае запуска в терминальной сессии Windows NT/2000.

При эксплуатации СУБД часто необходимо иметь несколько экземпляров сервера на одном компьютере для целей тестирования новой версии системы без влияния на основную инсталляцию или для разграничения задач администрирования.

Для решения такой задачи планируется разделить конфигурацию серверов и пространства имен глобальных объектов. Каждый экземпляр сервера будет использовать отдельный порт протокола TCP или имя Named Pipe.

Отметим, что сервер Yaffil использует отличные от InterBase/Firebird именованные глобальные объекты. Таким образом, появляется возможность одновременного запуска и Yaffil и InterBase/Firebird.

Хранение конфигурации в системном реестре

Для платформы Windows системный реестр - основное средство хранения конфигурационной информации. Текстовый файл IBCONFIG, используемый сейчас, имеет недостаток, связанный с невозможностью задания сложной конфигурации, имеющей иерархическую структуру. Кроме того, не так то просто приложениям анализировать этот файл и вносить изменения в него.

Следующие версии Yaffil будут использовать системный реестр для хранения конфигурации.

Процедурный язык в SQL запросах

Традиционно InterBase использует разные диалекты SQL для написания текста запросов, принимаемых с клиента и для описания текста хранимых процедур и триггеров. Некоторые типы операторов доступны только в одном из этих диалектов, в то время как другие конструкции имеют разный синтаксис в каждом случае.

В сервере Yaffil предполагается ввести возможность выполнения процедурных команд SQL в запросах, передаваемых с клиента, включая использование локальных переменных, операторы IF, WHILE, конструкции SELECT INTO и FOR SELECT.

Большие индексы

Планируется значительно увеличить максимальный размер ключа индекса с нынешних 128-256 (в зависимости от типов данных) байт в InterBase/Firebird. Как известно, при создании индексов по текстовым полям с национальным порядком сортировки (COLLATION), на каждый символ отводится 2-3 байта. Тем самым реальная максимальная длина индекса в настоящее время может составить 83 символа, что недостаточно для многих приложений.

Запуск сервера и описание ключей командной строки

Запуск сервера Yaffil не многим отличается от запуска других клонов InterBase/Firebird. Отличия состоят в дополнительных ключах командной строки, позволяющих настраивать параметры сетевых протоколов, поддерживаемых сервером.

Ниже приведен синтаксис командной строки, используемый при запуске сервера. Этот синтаксис относится как к "ручному" запуску сервера, так и при запуске как служба Windows NT/2000.

Запуск сервера архитектуры SuperServer:

ibserver.exe [-a] [-s] [-i [[address]/[port]]] [-n] [-x]

Запуск сервера архитектуры Classic:

ibremote.exe [-a] [-s] [-i [[address]/[port]]] [-n]

Описание ключей:

- a - запуск в режиме приложения. Если ключ не указан, предполагается запуск как служба Windows NT/2000.
- s - не показывать иконку с System Tray. Используется при запуске как служба Windows NT/2000, а также во всех остальных случаях, когда иконка нежелательна.
- i - включить поддержку протокола TCP. Данный ключ имеет необязательный параметр, определяющий привязку сервера к IP адресу интерфейса (сетевой карты) и номер порта TCP протокола, разделенных косой чертой (слэшем). Как IP адрес, так и номер порта может опускаться, по умолчанию действует привязка к любому адресу (INADDR_ANY) и порт, присвоенный сервису gds_db в файле services. Если сервис gds_db отсутствует в файле services, используется порт 3050.
- n - включить поддержку протокола Named Pipes. Данный ключ не поддерживается на операционных системах Windows 9x/Me.
- x - включить поддержку протокола XNET для локальных соединений. Данный ключ поддерживается только сервером архитектуры SuperServer, поскольку сервер Classic использует Named Pipes для локальных соединений.

Если ни один из ключей -i, -n, -x не указан в командной строке, включаются все поддерживаемые протоколы. Если сервер запущен как приложение и в области System Tray присутствует иконка Yaffil, вы можете увидеть список поддерживаемых протоколов в окне свойств сервера.

---

[i] В сервере Yaffil применён качественно отличный от InterBase/Firebird подход к управлению памяти. Ситуация с ошибкой менеджера памяти считается критической и приводит к немедленной остановки сервера без записи на диск кэша БД. Подобный подход позволяет избежать возможных катастрофических последствий для БД и максимально защищает данные пользователя. Менеджер памяти, используемый Yaffil, не имеет ограничений как по объёму выделяемых блоков памяти, так и по количеству выделенных блоков в одном пуле.

[ii] Сервер обрабатывает следующие исключительные ситуации:

Исключение Win32 API	*Исключение Interbase*
Пользовательское исключение (установлен 21 бит)	`"Win32 User SEH"`
`EXCEPTION_FLT_DENORMAL_OPERAND`	`isc_exception_float_denormal_operand`
`EXCEPTION_FLT_DIVIDE_BY_ZERO`	`isc_exception_float_divide_by_zero`
`EXCEPTION_FLT_INEXACT_RESULT`	`isc_exception_float_inexact_result`
`EXCEPTION_FLT_OVERFLOW`	`isc_exception_float_overflow`
`EXCEPTION_FLT_STACK_CHECK`	`isc_exception_float_stack_check`
`EXCEPTION_FLT_UNDERFLOW`	`isc_exception_float_underflow`
`EXCEPTION_INT_DIVIDE_BY_ZERO`	`isc_exception_integer_divide_by_zero`
`EXCEPTION_INT_OVERFLOW`	`isc_exception_integer_overflow`
`EXCEPTION_FLT_INVALID_OPERATION`	`isc_exception_float_invalid_operand`
`EXCEPTION_STACK_OVERFLOW`	`isc_exception_stack_overflow`

[iii] В отличие от локального прежнего протокола, XNET обеспечивает возможность подключения к серверу из сервисов, а также эффективную параллельную обработку запросов от разных клиентов. Недоступ-

ность XNET как локального протокола для классической архитектуры компенсируется автоматическим подключением к серверу по Named Pipes.

[iv] Следует отметить, что в настоящее время, не предусмотрена возможность управлять размещением полей типа BLOB на страницах с данными. Таким образом, поля типа BLOB могут размещаться и на страницах с полями основных типов, что иногда приводит к значительному снижению быстродействия.

[v] К сожалению, существующая архитектура ODS не позволяет эффективно оптимизировать «сборку мусора» в индексах. Элементы индекса располагаются на страницах данных в непрерывном массиве. При удалении элемента массива необходимо сдвигать оставшиеся элементы, таким образом невозможно реализовать эффективный алгоритм массового удаления элементов индекса. Отметим, что представление записей таблиц на страницах с данными имеет принципиально иной характер, позволяя освобождать место на странице с данными без необходимости последующего упорядочивания записей на странице. Пример описывающий проблему сборки мусора – SF: 585624 IB server stalled by simple script

[vi] Для серверов InterBase/Firebird предлагается использовать ключ –O (restore one table at a time), что позволяет, за счёт вставки данных таблиц в отдельных транзакциях, более эффективно восстанавливать базы. Однако в случае наличия таблиц со значительным количеством записей, подобные ухищрения не позволяют достигнуть желаемого. Если же возникает потребность в обработке (вставка, удаления, изменение) большого количества данных в рамках одной транзакции, то ситуация становится неразрешимой. Пользователь сервера Yaffil полностью избавлен от подобных проблем.

[vii] Как известно в Interbase идентификаторы метаданных не могут быть длиннее 31 символа. А с использованием привилегий, фактическая длинна идентификатора сокращается до 27 символов. К сожалению, указанные ограничения заложены как в ODS, так и в архитектуру клиентского API. Разработчики не рекомендуют создавать таблицы, имена которых имеют более чем 27 символов.

[viii] Ошибка вызвана неверным представлением полей типа DATE в бинарном формате индекса. К сожалению, несмотря на очевидное решение проблемы, исправление потребует внести изменения в формат индексов типа DATE, что сделает несовместимыми указанные индексы с IB/FB.

[ix] Ошибка существует ещё в IB4.x. При использовании составного индекса и наличии хотя бы одного условия, отличного от равенства, на один из сегментов индекса, происходит частичное сравнение ключей индекса. Т.е. фактически это эквивалентно сканированию односегментного индекса при условии отличном от равенства. При этом иногда возникают коллизии, основанные на бинарном представлении ключей индекса, приводящие к большему сканированию индекса, чем требуется. Для исправления ошибки необходимо при сравнении ключей учитывать вид операции на сегментами составного индекса.