



# Переход с СУБД MS-SQL на СУБД Firebird

Marcelo Lopez Ruiz

1 декабря 2005 - Версия документа 1.1.ru

Перевод на русский язык: Павел Меньщиков



---

# Содержание

Введение .....	3
За и против .....	3
Зачем переходить на СУБД Firebird .....	3
Почему не нужно переходить на СУБД Firebird .....	5
Установка сервера баз данных .....	5
Администрирование баз данных .....	6
Управление файлами базы данных .....	6
Администрирование пользователей .....	7
Операции резервного копирования и восстановления .....	7
Типы данных .....	8
Перенос типа данных bit .....	12
Перенос типа данных identity (идентификатор) .....	12
Перенос типа данных uniqueidentifier (глобально уникальный идентификатор) .....	13
Синтакс SQL .....	14
Основы использования баз данных .....	15
Использование переменных .....	15
Управляющие конструкции .....	16
Стандартные операторы .....	19
Использование транзакций .....	20
Использование курсоров .....	20
Трюки с SQL .....	21
Трюк: использование каскадных изменений .....	21
Трюк: использование обновляемых представлений (views) .....	22
Доступ клиентов .....	22
Утилиты со встроенным в них доступом .....	22
Клиентский доступ для разработчиков .....	22
Утилиты .....	23
Утилиты для переноса .....	23
Замена утилит СУБД MS SQL .....	23
А. История документа .....	25
В. Заметки о публичном доступе и отказ от претензий .....	26

---

Microsoft SQL Server (MS SQL) является широко распространенным сервером баз данных. В настоящее время большинство пользователей используют три версии этого программного продукта: MS SQL 6.5, MS SQL 7 и MS SQL 2000.

## Введение

Этот раздел содержит информацию о данном руководстве по переходу на использование СУБД Firebird.

Документация по переходу с СУБД MS SQL Server, прежде всего, предназначена для помощи пользователям СУБД оценить, следует ли вообще осуществлять такой переход. Также документация знакомит с деталями процесса перехода, которые были наработаны различными пользователями при переходе на СУБД Firebird с СУБД MS SQL Server.

Есть две вещи, на которые следует обратить внимание при переходе.

Во-первых, проблемность переноса данных из одной СУБД в другую зависит от сложности схемы базы данных. Существует много утилит для помощи в этом процессе. В данном руководстве показаны стандартные преобразования.

Во-вторых, возможно, Вам понадобится перенести хранимые процедуры и триггеры «вручную». Этот процесс может оказаться непростым: существует множество различий, больших и не очень, между СУБД. Данное руководство дает примеры наиболее часто встречающихся проблем и их решений.

### Внимание

Это руководство было написано в 2003 году, когда текущей версией Firebird была версия 1.0. Хотя руководство с тех пор немного корректировалось, оно нуждается в полной ревизии. Этот факт не должен останавливать Вас от использования данного руководства - просто имейте в виду, что содержание может не соответствовать текущей версии Firebird, и в случае проблем обращайтесь к Release Notes (примечания к выпущенной версии программы) СУБД Firebird с версии 1.0 до используемой Вами версии.

## «За» и «против»

Этот раздел описывает причины, по которым может понадобиться перенос существующей базы данных на СУБД Firebird, и причины, по которым этого не стоит делать.

## Зачем переходить на СУБД Firebird

Причины необходимости перехода могут зависеть от того, какую версию MS SQL и с какой це-

люю Вы используете.

Например, обычно при использовании MS SQL 6.5 причинами для перехода могут служить имеющиеся в Firebird функции и простота использования. MS SQL 6.5 работает с фиксированными устройствами (devices), а не с динамически расширяющимися файлами, что ведет к возможным проблемам при администрировании. Также в MS SQL 6.5 существуют некоторые ошибки и некорректное поведение.

Если Вы используете MS SQL 7, то Вы знаете, что многие «особенности» были исправлены, но многих функций все еще не существует (обновляемые представления, большее управление полями идентификаторов, функции пользователя, хранимые процедуры, возвращающие наборы данных). Вы также не можете использовать каскадную ссылочную целостность данных до версии 2000. То же относится к разным порядкам сортировки строк в одной базе данных.

Для операционных систем семейства UNIX Firebird способен использовать систему безопасности операционной системы. Однако эту функцию желательно не использовать для возможности портирования.

СУБД MS SQL 2000 улучшена по сравнению с MS SQL 7, но, по-прежнему, в ней нет такой ключевой функции Firebird, как многоверсионная архитектура (multi-generational architecture, MGA), которая позволяет иметь «долгие» (по времени существования) транзакции, не затрагивающие поведение «быстрых» (по времени существования) операционных транзакций. Наоборот, MS SQL убеждает пользователей приобрести еще один сервер (оборудование, операционную систему, сервер баз данных) и установить его только как хранилище данных для генерации отчетов. Можно только удивляться необходимости отдельного хранилища данных в интегрированной среде, дающей возможности создания запросов к разным базам данных (cross-database query).

Другая причина перехода - исключение привязки к определенному поставщику программного обеспечения. MS SQL работает исключительно на Windows NT/2000 (существуют так называемые персональные выпуски [personal editions], но они имеют ограничения по числу одновременных подключений и имеющимся функциям). Это означает, что Вы «привязаны» к операционным системам Microsoft. Firebird может работать на множестве платформ, включая Microsoft Windows, Linux, Solaris, MacOS X и другие.

Еще одна причина для перехода - цена. Firebird бесплатен; за использование MS SQL необходимо платить за каждый процессор, на котором работает сервер MS SQL. Например, доступ к базе данных, расположенной на компьютере с dual-Pentium, через Интернет будет стоить \$39998 (цены с [сайта Microsoft](#) по состоянию на 2001.05.03).

Последняя, но не менее важная, причина в том, что исходные коды Firebird открыты. Это означает не только то, что в мире существует сотни разработчиков, готовых помочь Вам использовать эту СУБД, улучшать ее, находить ошибки и т.д., но и даже то, что Вы самостоятельно можете вносить изменения в исходный код и собирать Вашу версию СУБД для удовлетворения Ваших потребностей. Добавление таких функций, как система электронной почты или система ведения лога, является несложным, если Вы понимаете исходный код и имеете знания и опыт для внесения изменений. Хотя изменение исходного кода под свои нужды не является тривиальной задачей, эта задача все-таки выполнима. Также это вносит большой уровень гибкости в СУБД.

## **Почему не нужно переходить на СУБД Firebird**

Прежде всего, если Ваша система работает без каких-либо проблем, то зачем Вам переходить на что-то другое? В этом случае Вы можете рассматривать СУБД Firebird для использования в своих будущих проектах. Не ломайте то, что работает.

MS SQL 7 имеет несколько функций, которые Вы не найдете в СУБД Firebird, такие как встроенная поддержка репликации (эта поддержка доступна как дополнительный модуль к СУБД Firebird), временные таблицы и интеграция с другими СУБД посредством OLE DB. Также в MS SQL встроены сервисы анализа OLAP и полнотекстовый поисковый механизм (эта поддержка доступна как дополнительный модуль к СУБД Firebird).

В СУБД MS SQL доступны инкрементные резервные копии (и восстановление из таких копий). Firebird резервирует и восстанавливает полную базу данных (инкрементные резервные копии станут доступны пользователям СУБД Firebird, начиная с версии 2.0 - прим. перев.).

При работе с операционной системой Microsoft, СУБД MS SQL 7 и выше способна использовать учетные записи пользователей операционной системы. Однако эту функцию желательно не использовать для возможности портирования и в связи с вопросами производительности.

СУБД MS SQL 2000 также имеет встроенную поддержку работы с XML и поддержку разделенных представлений (partitioned views) для лучшей производительности при работе с таблицами, расположенными на нескольких серверах.

В общем, кажется, что СУБД MS SQL более производительна на Windows, чем СУБД Firebird на этой операционной системе. Также, СУБД MS SQL имеет более тесную интеграцию с Microsoft Visual Studio.

## **Установка сервера баз данных**

Этот раздел описывает различия при установке серверов баз данных.

Для пользователей, ранее использовавших СУБД MS SQL и являющихся новичками в использовании СУБД Firebird, вероятно, простота установки СУБД Firebird будет некоторым приятным сюрпризом. Процесс установки не представляет никаких проблем, Вы можете устанавливать соединение с базой данных сразу после установки СУБД Firebird. Обратите внимание, что по умолчанию именем администратора СУБД Firebird является SYSDBA, и пароль masterkey (пароль при установке в ОС Linux генерируется произвольный или запрашивается у пользователя при установке - прим. перев.); в СУБД MS SQL соответственно sa и пустой пароль.

Заметьте, что Вам не нужно осуществлять выбор используемого набора символов и способ их сортировки при установке СУБД Firebird. В СУБД MS SQL Вы не только должны указать эту информацию при установке, но и, в случае необходимости изменения, Вам необходимо будет переустановить СУБД и указать новые параметры. В дополнение к этому, другие программные пакеты, такие как Microsoft Commerce Server, могут отказываться работать с СУБД MS SQL,

если Вы выберете неверные значения параметров.

В отличие от СУБД MS SQL 6.5, СУБД Firebird не имеет понятия «устройства» (devices). Все данные хранятся в обычных файлах доступной файловой системы. Имейте в виду, что Вы не сможете использовать «чистые» разделы диска (raw disk partition, разделы без файловой системы) для хранения Ваших баз данных.

*Важное замечание:* СУБД MS SQL использует механизм журналирования (логи, logs) для сохранения целостности баз данных, в том числе при непредвиденных сбоях. СУБД Firebird использует многоверсионную архитектуру (multi-generation architect, MGA) для создания копии (версии) записи «на месте», если это необходимо (например, при модификации); но запись новой версии записи на диск может быть отложена, что может дать некоторый выигрыш в скорости работы СУБД. Для каждой базы данных Вы можете установить атрибут `Forced Writes` (принудительная запись) - рекомендуется устанавливать этот атрибут, чтобы быть уверенным в целостности базы данных в различных чрезвычайных ситуациях, таких как исчезновение питания и т.п. В случае уверенности в конфигурации оборудования, Вы можете не устанавливать этот атрибут (например, для выделенного сервера Linux, наличии источников бесперебойного питания и т.д.).

## Администрирование баз данных

Этот раздел описывает различия в управлении базами данных между СУБД Firebird и СУБД MS SQL.

### Управление файлами базы данных

СУБД MS SQL 6.5 использует устройства (devices), которые могут представлять собой файлы или «чистые» разделы (raw partitions, разделы без файловой системы), чтобы работать с данными. Это ведет к сложным для поддержки системам. СУБД MS SQL 7 и MS SQL 2000 используют обычные файлы вместо устройств. Для каждой базы данных создаются, как минимум, два файла: один, собственно, для хранения данных, и второй - для сохранения журнала выполненных транзакций.

СУБД Firebird не использует отдельный журнал для сохранения выполненных транзакций, и поэтому использует единственный файл для хранения всех данных.

Оператор **CREATE DATABASE** («создать базу данных») в СУБД Firebird проще, чем аналогичный оператор в СУБД MS SQL. Обратитесь к справочнику SQL для подробного описания всех возможностей.

Существенное различие в модели управления файлами - это использование СУБД групп файлов (filegroups) для деления базы данных на несколько наборов файлов по определенному принципу. СУБД Firebird так же может работать с разбитой на несколько файлов базой данных, но модель работы с файлами более проста (в смысле принципа, из которого производится раз-

биение на файлы - прим. перев.).

Дополнительная возможность, предоставляемая СУБД Firebird - это использование теневого файлов (shadow files). Теневые файлы являются немедленной копией исходной базы данных. Обычно они используются для быстрого получения актуальной копии. В СУБД MS SQL нет такой функции, хотя СУБД MS SQL 2000 имеет похожую функцию при использовании журналов, перемещенных между серверами, и репликации.

## **Администрирование пользователей**

СУБД MS SQL 6.5 предоставляет два объекта для управления: «логины» и «пользователи». «Логин» определяет комбинацию имени пользователя и пароля для доступа к серверу базы данных; «пользователь» определяет права доступа для каждой базы данных. «Логины» отображаются на «пользователей» в базе данных.

В СУБД MS SQL 7 добавлен новый объект управления группами пользователей - «роли». «Роль» является хранилищем прав. Некоторые «роли» являются предопределенными, например, «оператор создания копии базы данных» или «администратор базы данных».

СУБД Firebird предоставляет похожую модель, но не включает в нее «логины». Пользователи сообщают серверу свое имя, пароль и «роль», с правами которой они хотят работать. Существует одна общая база данных для всего сервера баз данных, в которой хранится информация обо всех пользователях сервера (их имена и пароли); информация о «ролях» и предоставленных каждому пользователю правах хранится в каждой конкретной базе данных. (Исходный документ имеет здесь неточность: там сказано, что в базе данных безопасности/пользователей хранится так же информация о «ролях» и разрешениях/запретах для каждого пользователя. Так же версия 2.0 СУБД Firebird вносит много изменений в модель управления пользователями. - прим. перев.)

В обоих СУБД считается хорошей практикой получать доступ к ресурсам через хранимые процедуры (stored procedures), и предоставлять доступ к данным только этим хранимым процедурам. (Это утверждение спорно и не может слепо применяться во всех случаях, по крайней мере, относительно СУБД Firebird. - прим. перев.)

## **Операции резервного копирования и восстановления**

СУБД Firebird использует более простую модель создания резервных копий баз данных и восстановления из них по сравнению с СУБД MS SQL, хотя в качестве жертвы здесь принесена гибкость. Резервное копирование может выполняться с помощью консольных (из командной строки) и графических утилит, в любом случае производится резервное копирование всей базы данных. Операция восстановления также восстанавливает всю базу данных. Нет возможности создания резервной копии только изменений, произошедших с момента создания последней резервной копии, или восстановления изолированного набора транзакций. (В СУБД Firebird 2.0

появилась возможность создания инкрементных резервных копий. - прим. перев.)

Обратите внимание, что при создании резервной копии базы данных СУБД Firebird существует важный параметр, определяющий формат создаваемой резервной копии: платформозависимый или платформонезависимый (переносимый). Создание резервной копии в переносимом формате позволяет администратору базы данных создать ее резервную копию в одной операционной системе, а затем восстановить базу данных из этой копии в другой операционной системе. Обычно такая возможность используется, например, при разработке приложения на Windows, а затем переносе базы данных на более мощный сервер Linux. (Так же, это рекомендуемый способ переноса баз данных с одного компьютера на другой даже для одной операционной системы: простое копирование файла базы данных не рекомендуется. - прим. перев.)

## Типы данных

Этот раздел описывает различные типы данных, доступные в СУБД Firebird и MS SQL, а также соответствия типов для перехода с одной системы на другую.

Доступные типы данных СУБД MS SQL зависят от версии СУБД. В следующей таблице перечислены типы данных и версия СУБД, в которой они были введены.

**Таблица 1. Таблица соответствия типов данных**

Вер. MSSQL	Тип данных	Тип Firebird	Определение MSSQL и пояснения
6.5	bigint	INT64	8-байтные целочисленные данные.
6.5	binary	CHAR	Двоичные данные фиксированной длины. Максимальная длина 8000 байт. В версии 6.5 максимальная длина была 255 байт.
6.5	bit	CHAR(1)	Целочисленные данные со значениями только 1 или 0. Обычно заменяются константами 'T' и 'F'. (В СУБД Firebird так же можно использовать тип SMALLINT со значениями 0 и 1. - прим. перев.)
6.5	char	CHAR	Текстовые данные фиксированной длины (не-Unicode). Максимальная длина 8000 символов. В версии 6.5

Вер. MSSQL	Тип данных	Тип Firebird	Определение MSSQL и пояснения
			максимальная длина была 255 байт. СУБД Firebird способна хранить до 32767 символов.
6.5	cursor		Ссылка на курсор. Этот тип используется только в хранимых процедурах и триггерах; этот тип не может использоваться при объявлении структуры таблиц.
6.5	datetime	TIMESTAMP	Дата и время с 1 января 1753 года до 31 декабря 9999 года, точность 3/100 секунды (3.33 мс).
6.5	decimal	DECIMAL	Числовые данные с фиксированной точностью. Диапазон от $-10^{38}-1$ до $10^{38}-1$ .
6.5	float	FLOAT	Вещественные числовые данные. Диапазон от $-1.79E+38$ до $1.79E+38$ .
6.5	image	BLOB	Двоичные данные переменной длины. Максимальная длина $2^{31}-1$ (2147483647) байт.
6.5	int	INTEGER	4-байтные целочисленные данные. Диапазон от $-2^{31}$ (-2147483648) до $2^{31}-1$ (2147483647).
6.5	money	DECIMAL(18,4)	Денежные данные. Диапазон от $-2^{63}$ (-922337203685477.5808) до $2^{63}-1$ (+922337203685477.5807), точность до 1/10000 денежной единицы.
7	nchar	CHAR(x) CHARACTER SET UNICODE_FSS	Символьные данные фиксированной длины (Unicode). Максимальная длина 4000 символов.
7	ntext	BLOB SUB_TYPE	Символьные данные переменной дли-

Вер. MSSQL	Тип данных	Тип Firebird	Определение MSSQL и пояснения
		TEXT	ны (Unicode). Максимальная длина $2^{30}-1$ (1073741823) символов.
6.5	numeric	NUMERIC	В СУБД MS SQL decimal и numeric эквивалентны.
7	nvarchar	VARCHAR(x) CHARACTER SET UNICODE_FSS	Символьные данные переменной длины. Максимальная длина 4000 символов.
6.5	real	DOUBLE PRECISION	Вещественные данные. Диапазон от $-3.40E+308$ до $3.40E+308$ .
6.5	smalldatetime	TIMESTAMP	Дата и время с 1 января 1900 года до 6 июня 2079 года, точность 1 минута. Тип данных Firebird имеет БОЛЬШИЕ диапазон и точность.
6.5	smallint	SMALLINT	2-байтные целочисленные данные. Диапазон от $-2^{15}$ (-32768) до $2^{15}-1$ (32767).
6.5	smallmoney	DECIMAL(10,4)	Денежные данные. Диапазон от $-214748.3648$ до $+214748.3647$ , точность 1/10000 денежной единицы. Диапазон СУБД Firebird больше при указанной замене.
2000	sql_variant	BLOB	Данные различных типов.
2000	table	нет эквивалента	Промежуточные результаты выполнения запроса для последующего использования.
6.5	text	BLOB SUB_TYPE TEXT	Символьные данные переменной длины (не-Unicode). Максимальная длина $2^{31}-1$ (2147483647) символов.
6.5	timestamp	INTEGER или	Уникальное для базы данных число. В

Вер. MSSQL	Тип данных	Тип Firebird	Определение MSSQL и пояснения
		BIGINT	СУБД Firebird Вам необходимо использовать механизм генераторов для этих целей.
6.5	tinyint	SMALLINT	1-байтовое целочисленное значение без знака. Диапазон от 0 до 255. В СУБД Firebird нет эквивалентного типа.
6.5	varbinary	VARCHAR	Двоичные данные переменной длины. Максимальная длина 8000 байт.
6.5	varchar	VARCHAR	Символьные данные переменной длины (не-Unicode). Максимальная длина 8000 символов. СУБД Firebird способна хранить до 32765 символов. В СУБД MS SQL 6.5 максимум был 255 символов.
7	uniqueidentifier	CHAR(36)	Глобально уникальный идентификатор (GUID). В СУБД Firebird Вам необходимо использовать функции, определяемые пользователем (UDF), для генерации значения идентификатора. (Если Вы собираетесь индексировать поле с глобально уникальными идентификаторами, то лучше использовать UUID - это другой формат представления GUID, представляемый как CHAR(22). - прим. перев.)

Небольшое различие в поведении в СУБД Firebird между типами NUMERIC и DECIMAL, которое приходит на ум, заключается в том, что определение NUMERIC означает *четко* указанную точность (общее число цифр), в то время как DECIMAL означает *как минимум* указанную точность. В СУБД MS SQL оба типа numeric и decimal эквивалентны.

Существует также квази-тип данных - identity (идентификатор), который может использоваться только для указания в определениях таблиц. Фактически, это тип данных int, значение поля автоматически генерируется при добавлении новой записи и не может быть впоследствии изменено.

## ***Перенос типа данных bit***

Тип данных bit используется для хранения булевого (логического) значения, 0 или 1. СУБД MS SQL не поддерживает присваивание значения NULL таким полям. Пользователи СУБД Firebird могут эмулировать поведение логического типа, используя типы SMALLINT или CHAR(1).

Возможные значения полей логического типа в СУБД Firebird могут быть ограничены через использование доменов (domains).

## ***Перенос типа данных identity (идентификатор)***

Существует несколько способов переноса механизма идентификаторов СУБД MS SQL. В общем, СУБД Firebird имеет большую гибкость и мощь в этом вопросе.

Самый простой способ создания аналога типа identity - создание триггера **BEFORE INSERT** (перед вставкой новой записи в базу данных) для интересующей Вас таблицы, и в этом триггере присваивать очередное значение генератора. Такой метод гарантирует уникальность (при условии, что шаг для генератора отличен от нуля и всегда имеет один и то же знак - прим. перев.).

Для дополнительной гибкости, можно использовать один генератор для всех таблиц. В этом случае получится аналог типа timestamp - уникального для всей базы данных идентификатора.

Еще одна обычная техника - создание хранимой процедуры, которая возвращает очередное значение генератора. Это так же позволяет получать значение генератора и использовать его для нескольких операций (например, при добавлении записи в основную таблицу, а затем добавлении нескольких записей в подчиненную таблицу).

```
CREATE TABLE my_table (  
    my_number integer not null primary key  
)  
  
CREATE GENERATOR my_generator  
  
CREATE TRIGGER my_before_trigger FOR my_table  
BEFORE INSERT  
AS  
BEGIN  
    IF (NEW.my_number IS NULL)  
        THEN NEW.my_number = GEN_ID(my_generator, 1);  
END  
  
CREATE PROCEDURE get_my_generator  
RETURNS (new_value INTEGER)
```

```
AS
BEGIN
    new_value = GEN_ID(my_generator, 1);
END
```

## ***Перенос типа данных `uniqueidentifier` (глобально уникальный идентификатор)***

В СУБД MS SQL тип `uniqueidentifier` используется для репликации. Также это простой способ определения уникальных глобальных идентификаторов для записей базы данных.

Для использования аналогичного типа данных в СУБД Firebird создайте триггер **BEFORE INSERT** (перед вставкой новой записи) для интересующей Вас таблицы с полем типа `uniqueidentifier`, и используйте значение функции, определяемой пользователем (UDF), для получения очередного значения GUID.

Вы можете использовать библиотеку `uuidUDF` (прим. перев.):

```
/******
UUID_CREATE
Returns a UUID (cstring(22), compressed, reversed,
    URL compatible UUID)
Parameters: cstring(22) dummy to allow Interbase to perform
    memory management
Returns:    cstring(22)
*****/
DECLARE EXTERNAL FUNCTION UUID_CREATE
    CSTRING(22)
    RETURNS PARAMETER 1
    ENTRY_POINT 'fn_uuid_create' MODULE_NAME 'uuidlib';

/******
GUID_CREATE
Returns a GUID (cstring(36), standard readable representation
of a UUID in the xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx format)
Parameters: cstring(36) dummy to allow Interbase to perform
    memory management
Returns:    cstring(36)
*****/
DECLARE EXTERNAL FUNCTION GUID_CREATE
    CSTRING(36)
    RETURNS PARAMETER 1
    ENTRY_POINT 'fn_guid_create' MODULE_NAME 'uuidlib';
```

## Синтакс SQL

Этот раздел описывает основные различия в синтаксе языка SQL, используемого СУБД Firebird и MS SQL.

СУБД Firebird и MS SQL могут ссылаться на объекты базы данных (таблицы, поля и т.д.) по их именам напрямую, если имена объектов не содержат пробелы и другие недопустимые в прямой ссылке символы (например, нелатинские буквы). Для использования пробелов и других символов СУБД MS SQL использует квадратные скобки, [ и ], а СУБД Firebird использует двойные кавычки, ". Еще одно различие - возможность использования в СУБД MS SQL схемы для ссылки на объект: база\_данных.владелец\_объекта.объект. СУБД Firebird не допускает такой нотации.

### Внимание

СУБД MS SQL использует регистро-зависимые имена объектов, если при установке Вы выбрали использование различения регистра символов; в противном случае, имена объектов регистро-независимы. Весело? Не очень...

### Подсказка

СУБД MS SQL способна работать с идентификаторами, имена которых заключены в двойные кавычки, но по умолчанию эта возможность доступна только при доступе через OLE DB и ODBC, но не при доступе через DB-Library. По этой причине такую практику работы следует избегать.

СУБД MS SQL 7 и выше поддерживает обновляемые соединения (joins) (обновление, удаление, вставка). СУБД Firebird не распознает такой синтакс.

Типы данных, конечно, различаются. Хотя обе СУБД имеют общее подмножество наиболее часто используемых типов. Этот вопрос редко вызывает проблемы при переносе базы данных.

Различаются встроенные функции. Большинство из встроенных функций СУБД MS SQL можно заменить в СУБД Firebird использованием функций, определяемых пользователем (UDFs).

Различаются форматы указания строковых констант для дат. СУБД Firebird принимает строки различных форматов, вне зависимости от используемой платформы. СУБД MS SQL, в свою очередь, использует совмещение серверо-независимых, серверо-платформенных форматов и формата настройки клиентского соединения. Дополнительно, методы доступа СУБД MS SQL обычно вводят один или два уровня, в которых строковая константа может быть преобразована в дату тем или иным образом.

В СУБД MS SQL можно определять большее количество переменных окружения, чем в СУБД Firebird, но наиболее общие можно найти и в СУБД Firebird (извлечение идентификатора и имени пользователя). Единственная важная переменная, которая отсутствует в СУБД Firebird, - это переменная, возвращающая количество строк последней операции (с версии 1.5 СУБД Firebird такая переменная введена - прим. перев.).

Важное различие было в том, что СУБД Firebird 1.0 не поддерживала оператор CASE СУБД MS SQL. Иногда можно было заменить его функциональность использованием хранимой процедуры. Начиная с версии 1.5, СУБД Firebird поддерживает использование оператора CASE.

Небольшое различие между СУБД еще и в том, что СУБД MS SQL не использует разделителей для операторов, что может служить источником трудно обнаруживаемых ошибок при переходе, особенно при использовании множества скобок. СУБД Firebird в скриптах требует завершать каждый оператор точкой с запятой (если не определен другой разделитель - прим. перев.), поэтому ошибки легче обнаружить.

Обе СУБД MS SQL и Firebird поддерживают комментарии, заключенные между разделителями /\* и \*/. СУБД MS SQL также поддерживает синтакс «два дефиса» -- для однострочного комментария. Некоторые утилиты для СУБД Firebird также поддерживают такой синтакс.

## Основы использования баз данных

СУБД MS SQL позволяет клиентам обращаться к нескольким базам данных через одно и то же соединение. Для этого при обращении к объектам баз данных необходимо использовать нотацию база\_данных.владелец\_объекта.объект, либо явно использовать оператор USE.

СУБД Firebird не допускает использование нескольких баз данных в одном операторе SQL. Однако эта СУБД позволяет клиенту работать с транзакциями, охватывающими несколько баз данных.

Существует много утилит для ввода и выполнения операторов SQL для обеих платформ. Имейте в виду, что для СУБД Firebird не нужно вводить команду GO для разделения пакетов команд, как для пакетов T-SQL - Вы обычно управляете транзакциями явно. Также, Вы можете для обеих СУБД использовать настройку «подтверждать (commit) каждый оператор».

### Внимание

Если Вы установили СУБД MS SQL и Firebird на один компьютер, будьте внимательны при использовании утилиты `isql`. Если Вы не указываете при запуске этой утилиты полный путь, будет запущена та, что первой будет найдена по путям для поиска файлов, а обе СУБД имеют в своем составе программу с именем `isql`.

## Использование переменных

Управление переменными схоже на обеих платформах. Переменные необходимо объявлять до их использования, а также указать их тип. В СУБД Firebird нет необходимости добавлять к имени переменной префикс @, а также переменные обязательно должны быть объявлены до тела хранимой процедуры или триггера.

Например, сравните следующие части кода.

```
/* MS-SQL */
```

```
CREATE PROCEDURE my_procedure
AS
DECLARE @my_variable int
SET @my_variable = 5

/* Firebird */
CREATE PROCEDURE my_procedure
AS
DECLARE VARIABLE my_variable int;
BEGIN
    my_variable = 5;
END
```

В обоих СУБД параметры считаются обычными переменными с установленным начальным значением.

## Управляющие конструкции

### BEGIN..END

В обоих СУБД ключевые слова BEGIN и END могут использоваться для объединения нескольких операторов в один блок, например, при использовании в условном операторе IF.

### GOTO

СУБД Firebird не имеет оператора GOTO. Однако, как правило, это к лучшему. Оператор GOTO обычно используется СУБД MS SQL, потому что по умолчанию возникающие при выполнении операторов ошибки не откатывают транзакции, необходимо после выполнения каждого оператора проверять значение переменной @@ERROR; оператор GOTO используется для объединения в группы операторов управления ошибками. В СУБД Firebird используется более удачный механизм управления ошибками - оператор WHEN...DO.

Конечно, оператор GOTO может использоваться и для других целей. В таком случае, правильное использование хранимых процедур обычно улучшает дизайн базы данных.

### IF

Оператор IF..ELSE в СУБД Firebird имеет ту же семантику, что и в СУБД MS SQL, однако синтакс СУБД Firebird требует наличия ключевого слова THEN после условия.

```
IF (something = 'unknown')
    THEN something = 'uuhhh.....';
ELSE something = 'I know! I know!';
```

## CASE

СУБД Firebird 1.0 не поддерживает оператор CASE, поэтому Вам необходимо будет сделать дополнительную работу по переносу, если Вы использовали этот оператор в базе данных СУБД MS-SQL.

Вы можете [пропустить этот раздел](#), если Вы используете СУБД Firebird 1.5 или более новую версию, так как, начиная с версии 1.5, СУБД Firebird имеет поддержку оператора CASE.

Оператор CASE используется аналогично оператору `switch` языка C или оператору `case` языка Pascal для замены одного значения на другое. Обычно в СУБД Firebird 1.0 эту операцию можно эмулировать хранимой процедурой, возвращающей некоторое значение.

```
/* Это исходный оператор MS SQL,
   используя традиционную базу данных pubs. */
CREATE PROCEDURE list_states
AS
SELECT
CASE state
WHEN 'CA' THEN 'California'
WHEN 'UT' THEN 'Utah'
ELSE 'unknown'
END
FROM authors

/* Это вариант перевода в СУБД Firebird 1.0. */
/* Выделить оператор CASE. */
CREATE PROCEDURE get_state_name ( state_code char(2) )
RETURNS ( state_name varchar(64) )
AS
BEGIN
IF (state_code = 'CA') THEN state_name = 'California';
ELSE IF (state_code = 'UT') THEN state_name = 'Utah';
ELSE state_name = 'unknown';
END

/* Это процедура-выборка. */
CREATE PROCEDURE list_states
RETURNS (state varchar(64))
AS
DECLARE VARIABLE short_state CHAR(2);
BEGIN
FOR SELECT state FROM authors INTO :short_state DO
BEGIN
EXECUTE PROCEDURE get_state_name :short_state
RETURNING_VALUES :state;
SUSPEND;
END
```

END

Три замечания по приведенному примеру. Во-первых, перенос является тривиальным. Во-вторых, текст получается довольно ясным. В-третьих, использование хранимой процедуры вносит большую гибкость и простоту внесения изменений логики. Предположим, оператор CASE для поля state встречается в 12 различных процедурах, и был добавлен новый штат, или что Вы неправильно написали название штата... Совершенно ясно, что лучше выполнить указанное преобразование для использования хранимой процедуры.

Еще раз: с версии 1.5 СУБД Firebird полностью поддерживает оператор CASE, поэтому, в принципе, необходимости перехода на использование хранимых процедур нет.

## WHILE

Оператор WHILE существует в обоих СУБД Firebird и MS SQL, но с некоторыми различиями. Не существует операторов BREAK или CONTINUE, но их можно эмулировать при помощи дополнительных конструкций. Так же есть небольшое различие в используемом синтаксисе: СУБД Firebird требует наличия ключевого слова DO после условия цикла. Сравните следующие эквивалентные части кода.

```
/* Синтакс Firebird. */
WHILE (i < 3) DO
BEGIN
  i = i + 1;
  j = j * 2;
END
```

```
/* Синтакс MS SQL. */
WHILE (i < 3)
BEGIN
  SET @i = @i + 1
  SET @j = @j * 2
END
```

## RETURN

Оператор RETURN в СУБД MS SQL возвращает значение целочисленной переменной и прекращает выполнение. В СУБД Firebird существует оператор EXIT, который передает управление на заключительный END хранимой процедуры. Однако здесь нет скрытой возвращаемой переменной, поэтому, если Вам необходимо вернуть некоторое значение (что опционально в СУБД MS SQL), Вы должны явно объявить возвращаемую переменную в процедуре.

## WAITFOR

Оператор WAITFOR в СУБД MS SQL приостанавливает выполнение на некоторое время или до наступления указанного времени. Что-то подобное можно осуществить при помощи функ-

ций, определяемых пользователем (UDFs), в СУБД Firebird. Но в обоих СУБД использование такого оператора следует исключить, поскольку взаимодействие с клиентом полностью приостанавливается (в СУБД Firebird это может привести и к приостановке обслуживания всех соединений, а не только соединения, вызвавшего аналог указанного оператора - прим. перев.).

## Стандартные операторы

Стандартными операторами, имеющимися в обоих СУБД, являются SELECT, INSERT, UPDATE и DELETE. СУБД Firebird и MS SQL поддерживают их, но в СУБД MS SQL есть несколько нестандартных расширений этих операторов, о которых необходимо рассказать на случай их использования.

В СУБД Firebird оператор SELECT не позволяет использовать ключевое слово INTO для создания новой таблицы «на лету». Вместо этого, ключевое слово INTO используется для связи результата запроса с переменной.

```
/* Синтаксис MS SQL для присваивания значения поля переменной. */
SELECT @my_state = state
FROM authors
WHERE auth_name = 'John'

/* Синтаксис Firebird. */
SELECT state INTO :state /* --> обратите внимание на
                           ":" перед именем переменной */
FROM authors
WHERE auth_name = 'John'
```

В СУБД MS SQL 7 и выше в операторе SELECT можно указывать спецификатор TOP для ограничения возвращаемого набора данных. Эта функция в настоящий момент находится в стадии разработки для СУБД Firebird. (Спецификаторы FIRST и SKIP в СУБД Firebird введены, начиная с версии 1.5. - прим. перев.)

Обе СУБД MS SQL и Firebird поддерживают обычный синтаксис оператора INSERT и оператора INSERT..SELECT.

Обе СУБД MS SQL и Firebird поддерживают обычный синтаксис оператора UPDATE. СУБД MS SQL также поддерживает синтаксис оператора UPDATE, в котором выполняется соединение (join) и производится обновление одной из таблиц соединения. Можно думать об этом как об условии WHERE на стероидах. Если такая функция очень нужна, то ее можно реализовать в СУБД Firebird с использованием представлений (views).

Обе СУБД MS SQL и Firebird поддерживают обычный синтаксис оператора DELETE. СУБД MS SQL также поддерживает оператор TRUNCATE TABLE, который более эффективен (но и опаснее), чем оператор DELETE. (СУБД MS SQL также поддерживает синтаксис оператора DELETE, в котором выполняется соединение. - прим. перев.)

```
/* Синтаксис MS SQL для удаления всех записей my_table. */
TRUNCATE TABLE my_table /* ...или... */
```

```
DELETE FROM my_table

/* Синтаксис Firebird. */
DELETE FROM my_table
```

Самая большая угроза - действовать «методом тыка». Больше всего данных было утеряно командами "delete from xxx" и последующей "ой!", чем продуманной командой "delete from rdb\$pages".

—Jim Starkey

## Использование транзакций

В СУБД Firebird в DSQL (dynamic SQL) транзакции «напрямую» не используются. Именованные транзакции в этом случае недоступны совсем. (Операторы DSQL выполняются в контексте транзакций, запущенных и контролируемых клиентским приложением.- прим. перев.) Синтаксис обоих СУБД поддерживает ключевое слово WORK для совместимости.

В большинстве случаев проблем с транзакциями не должно возникать: явное управление транзакциями «на месте» в СУБД MS SQL используется обычно из-за отсутствия поддержки управлением через исключения (exceptions).

### Подсказка

В СУБД MS SQL есть глобальная переменная XACT\_ABORT, которая управляет откатом транзакции при возникновении ошибки времени выполнения (run-time error). В противном случае Вам необходимо проверять значение переменной @@ERROR после выполнения каждого оператора.

В общем, большинство проблем, связанных с уровнями изоляции транзакций в СУБД MS SQL, пропадают при переходе на СУБД Firebird. Соперничество между «читателями» и «писателями» минимально за счет использования многоверсионной архитектуры (multigeneration architecture, MGA).

## Использование курсоров

В СУБД MS SQL курсоры используются, в основном, для перемещения по результатам запросов, чтобы выполнить с этими результатами некоторые действия. Кроме синтаксиса, нет большой разницы для выполнения одной и той же задачи. Хотя существуют опции для перемещения вперед и назад, на практике используются, по большей части, однонаправленные курсоры.

```
/* Синтакс MS SQL. */
DECLARE my_cursor CURSOR
FOR SELECT au_lname FROM authors ORDER BY au_lname
  DECLARE @au_lname varchar(40)
  OPEN my_cursor
  FETCH NEXT FROM my_cursor INTO @au_lname
```

```
WHILE @@FETCH_STATUS = 0
BEGIN
    /* Сделать что-то интересное с @au_lname. */
    FETCH NEXT FROM my_cursor
END
CLOSE my_cursor
DEALLOCATE my_cursor

/* Синтакс Firebird. */
DECLARE VARIABLE au_lname VARCHAR(40);
...
FOR SELECT au_lname FROM authors
ORDER BY au_lname INTO :au_lname DO
BEGIN
    /* Сделать что-то интересное с au_lname. */
END
```

Заметьте, что СУБД MS SQL может размещать курсоры в переменных, и передавать эти переменные как параметры; это невозможно в СУБД Firebird.

#### **Внимание**

Различные версии СУБД MS SQL имеют различные области видимости для переменных-курсов. Будьте осторожны при их использовании и при переносе кода.

## Трюки с SQL

Этот раздел описывает некоторые трюки СУБД Firebird, используемые для эмуляции поведения СУБД MS SQL и для исключения «обходных путей» СУБД MS SQL.

### ***Трюк: использование каскадных изменений***

Версии СУБД MS SQL до 2000 не поддерживают каскадные обновления и удаления. Ссылочные ключи при изменениях всегда откатывались (rolled back).

В СУБД MS SQL указанная проблема решается комбинированием хранимых процедур и триггеров, и вместе с тем исключается явная декларация ссылочной целостности (foreign key). В свою очередь, такое поведение делает ссылочную целостность неявной вместо наличия явно декларируемой ссылочной целостности, а это означает, что различные утилиты не смогут распознавать наличие отношений между таблицами и работать с этими отношениями.

При переносе базы данных все такие «обходные пути» могут быть опущены - СУБД Firebird полностью поддерживает каскадные обновления и удаления через декларативную ссылочную целостность. (СУБД Firebird до версии 2.0 имеет проблемы с индексами с низкой селективностью, которые часто образуются и используются в реализации декларативной ссылочной це-

лостности, и ситуация с которыми усугубляется нежелательностью создания параллельных индексов с хорошей селективностью во избежание введения оптимизатора в заблуждение. Версия 2.0 решает эту проблему. - прим. перев.)

## **Трюк: использование обновляемых представлений (views)**

Версии СУБД MS SQL до 2000 не поддерживали полностью обновления для представлений, основанных на соединениях. Это было основным вопросом, так как представления обычно считались необновляемыми («только для чтения»); существовало некоторое количество ограничений, чтобы сделать такие представления обновляемыми.

В СУБД Firebird также существуют некоторые ограничения, но эти ограничения рассматриваются только для *автоматического* обновления. Если СУБД не в состоянии самостоятельно произвести обновления, Вы можете создать триггеры для представления, чтобы реализовать необходимую логику работы.

## **Доступ клиентов**

Этот раздел описывает различия в способах доступа клиентских приложений к базам данных, управляемых СУБД Firebird и MS SQL.

### **Утилиты со «встроенным» в них доступом**

Стандартной утилитой для работы из командной строки является программа isql. Обычно она используется для выполнения больших скриптов или при написании пакетных файлов.

Когда имеется возможность работать с графическим интерфейсом пользователя (GUI), для СУБД Firebird рекомендуется использовать утилиты IBOConsole, IB\_SQL, IB Expert, DataBase Workbench, FlameRobin (платформонезависимая утилита). Большая часть из указанных утилит схожа с утилитой управления для СУБД MS SQL Enterprise Manager.

### **Клиентский доступ для разработчиков**

Существует несколько механизмов доступа к базам данных СУБД Firebird. Среди них использование низкоуровневого API на языке C, драйвера ODBC (Open Database Connectivity), драйвера OLE DB (он также используется для доступа в технологии ActiveX Data Objects, ADO), и др.

Использование низкоуровневого API на языке C позволяет создавать переносимый код. На

всех платформах есть поддержка этого API. Это также является базисом для компонент доступа сред разработки Delphi и C++ Builder, таких как InterBase Objects (IBO), FIBPlus, InterBase Express (IBX), UnifiedIB (UIB) и др.

Использование драйвера ODBC позволяет разработчикам создавать код, который можно будет использовать с различными базами данных (и серверами баз данных) в случае, если разработчик ограничит себя использованием некоторого (общего) подмножества операторов языка SQL. Существует много утилит, использующих базы данных СУБД Firebird через драйверы ODBC.

Использование драйвера OLE DB позволяет разработчикам использовать популярный интерфейс доступа к данным от Microsoft ADO API. Это дает возможность получать доступ к базам данных СУБД Firebird из таких сред, как Visual Basic или ActiveX Server Pages (ASP). В настоящее время наиболее популярной является собственная связка Microsoft - OLE DB->ODBC.

## Утилиты

Этот раздел описывает утилиты, которые могут помочь в переносе баз данных из одной СУБД в другую, а также способные заменить стандартные утилиты СУБД MS SQL.

### Утилиты для переноса

Этот раздел описывает утилиты, которые можно использовать в процессе перехода с СУБД MS SQL на СУБД Firebird.

**Таблица 2. Утилиты для переноса**

Название	Кем предоставляется
Microsoft Access and Microsoft SQL Server to InterBase Wizard (Помощник перехода с Microsoft Access и Microsoft SQL Server на InterBase)	<a href="#">Marcelo Lopez Ruiz</a>
IBDataPump	<a href="#">Alex Polozouk</a>

### Замена утилит СУБД MS SQL

В следующей таблице перечислены утилиты, которые можно использовать в СУБД Firebird по аналогии со стандартными утилитами управления, поставляемыми с СУБД MS SQL.

К Вашему сведению, в основном, все приложения, которые имеют функции, аналогичные Enterprise Manager, также способны обслуживать (выполнять) и запросы.

**Таблица 3. Альтернативные утилиты**

Название	Замена
Электронная документация	Электронная документация PDF
Утилита Client Network	Нет необходимости; строка подключения к базе данных говорит сама за себя. Проверьте правильность с помощью isql или других утилит.
Enterprise Manager	IBOConsole
Enterprise Manager	<a href="#">IBExpert</a> (free/commercial)
Enterprise Manager	<a href="#">DataBase Workbench</a> (commercial)
Import and Export Data	-
Performance Monitor	<a href="#">InterBase Heartbeat</a> (commercial)
Profiler	<a href="#">InterBase Observer SQL Profiler</a> (commercial)
Query Analyzer	Planalyzer
Утилита Server Network	IBOConsole
Service Manager	Control Panel Applet

Указанный список является далеко не полным, и служит для примера того, что и чем можно заменить. Для более полного и обновленного списка обратитесь к [IBPhoenix Contributed Downloads](#).

## История документа

Точная история изменения исходного файла фиксируется в модуле `manual` нашего дерева CVS; обратитесь к [http://sourceforge.net/cvs/?group\\_id=9028](http://sourceforge.net/cvs/?group_id=9028).

### История переиздания

1.0	2003	MLR	Написан, опубликован и размещен в общем доступе автором Marcelo Lopez Ruiz.
1.1	1 окт 2005	PV	Добавлено предупреждение о вероятности устаревания этого документа. Раздел об управляющих конструкциях разбит на подразделы. Явно указано, что замена оператора CASE нужна только при использовании версий СУБД Firebird до 1.5. Добавлена история документа. Заметки о публичном доступе перемещены в приложение.
1.1.ru	1 дек 2005	PM	Документ переведен на русский язык. Исправлены некоторые неточности.

## **Заметки о публичном доступе и отказ от претензий**

Автор размещает эту статью в публичном доступе, оставляя за собой все авторские права. Любой вправе использовать, изменять, повторно публиковать, продавать, дарить эту работу без предварительного согласия кого бы то ни было.

Эта документация предоставляется на основе «как есть», без каких бы то ни было гарантий. Вы используете документацию на свой собственный риск! Ни при каких обстоятельствах ни автор(ы), ни распространитель(ли) не будут нести ответственность за возможные повреждения, появившиеся из-за использования или невозможности использования этой документации.