
Краткое руководство по миграции на Firebird 4.0

Оглавление

Введение	2
1. Ручная установка Firebird 4.0 в Windows	2
1.1. Инициализация SYSDBA	2
1.1.1. Инициализация SYSDBA с помощью ISQL	2
1.1.2. Инициализация SYSDBA с помощью GSEC	3
1.2. Конфигурация	4
1.2.1. Режим сервера	4
1.2.2. Авторизация из клиентских библиотек Firebird 2.5	4
1.2.3. Установка часового пояса соединения по умолчанию	5
1.2.4. Одновременный запуск нескольких экземпляров Firebird	5
1.3. Установка и запуск Firebird как службы	6
1.3.1. Использование install_service.bat и uninstall_service.bat	8
1.4. Установка клиента	8
1.4.1. Утилита instclient	10
1.5. Установка embedded версии	11
2. Преобразование базы данных к новому формату	12
2.1. Список несовместимостей на уровне языка SQL	12
2.1.1. Новые зарезервированные слова	12
2.1.2. Имена столбцов в PSQL курсорах	12
2.1.3. Новые типы данных	13
2.1.4. Литералы дат и времени	14
2.2. Поддержка внешних функций (UDF) объявлена устаревшей	14
2.3. Преобразование базы данных к новой ODS	15
2.3.1. Предупреждения об отсутствии UDF	16
3. Перенос псевдонимов баз данных	17
4. Перенос списка пользователей	17
4.1. Перенос списка пользователей из Firebird 3.0	17
4.2. Перенос списка пользователей из Firebird 2.5	20
4.2.1. Копирование списка пользователей в плагин SRP	21
4.2.2. Копирование списка пользователей в плагин Legacy_UserManager	22
5. Настройка доверительной аутентификации	24
6. Несовместимости на уровне приложения	25
6.1. Новые типы данных	25
6.2. Согласованное чтение в транзакциях READ COMMITTED	26

6.3. INSERT ... RETURNING требует привилегию SELECT	27
7. Заключение	27

Введение

Эта статья предназначена прежде всего для тех, кто планирует в ближайшем будущем обновить СУБД Firebird до версии 4.0. Многие администраторы всё ещё используют Firebird 2.5, но планируют обновиться до версии 4.0 прежде всего из-за наличия в ней встроенной репликации. Именно поэтому здесь описан процесс миграции как с Firebird 2.5, так и Firebird 3.0.

Спонсором написания статьи «Руководство по миграции на Firebird 4.0» является IBSurgeon (iBase.ru) (<https://www.ib-aid.com>, <https://www.ibase.ru>): техническая поддержка и инструменты разработчика и администратора для СУБД Firebird.

1. Ручная установка Firebird 4.0 в Windows

Далее описан процесс установки из zip архива. Даже если вы устанавливаете Firebird из специального инсталляционного пакета, вам всё равно может потребоваться изменить некоторые настройки после установки. Кроме того, ручная установка позволяет установить несколько версий Firebird на одной машине.

Скачайте архив соответствующей разрядности и распакуйте его в директорию, в которой будет размещён сервер Firebird. Далее необходимо создать пользователя SYSDBA. Для тех администраторов, которые мигрируют с Firebird 3.0, это операция не нова. В Firebird 2.5 и ранее после установки пользователь SYSDBA существовал всегда и имел пароль по умолчанию *masterkey*, который надо было сразу же менять.

1.1. Инициализация SYSDBA

Начиная с Firebird 3.0 пользователь SYSDBA не инициализирован по умолчанию (для плагина управления пользователями SRP), поэтому необходимо явно создать пользователя и указать ему пароль. Это можно сделать двумя способами: с использованием консольного инструмента для выполнения интерактивных запросов *isql.exe* и консольного инструмента для управления базой данных безопасности *gssec.exe*.



Замечание

В зависимости от размещения Firebird эти утилиты могут потребовать запуска с привилегиями администратора.

1.1.1. Инициализация SYSDBA с помощью ISQL

Запустите инструмент для выполнения интерактивных запросов *isql.exe*. Соединитесь с базой данных безопасности в режиме встроенного сервера, указав при этом пользователя SYSDBA без пароля. Пользователя SYSDBA ещё не существует в базе данных безопасности, но в embedded режиме пользователь и его пароль не проверяется, и Firebird доверяет любому

указанному имени пользователя. Выполните SQL запрос для создания пользователя SYSDBA:

```
CREATE USER SYSDBA PASSWORD '<password>';
```

Пользователь SYSDBA инициализирован, можно выходить из интерактивного режима.

Пример 1. Инициализация SYSDBA через ISQL

```
c:\Firebird\4.0>isql

Use CONNECT or CREATE DATABASE to specify a database

SQL> connect security.db user SYSDBA; Database: security.db, User: SYSDBA
SQL> CREATE USER SYSDBA PASSWORD 'm8ku234pp';
SQL> exit;

c:\Firebird\4.0>
```

1.1.2. Инициализация SYSDBA с помощью GSEC

Запустите `gsec.exe` указав пользователя SYSDBA и базу данных security.db. Выполните команду для добавления пользователя SYSDBA:

```
add SYSDBA -pw <password>
```

Пример 2. Инициализация SYSDBA через GSEC

```
c:\Firebird\4.0>gsec -user SYSDBA -database security.db

* gsec is deprecated, will be removed soon *

GSEC> add SYSDBA -pw m8ku234pp
GSEC> quit

c:\Firebird\4.0>
```



Предупреждение

Инструмент `gsec.exe` является устаревшим, многие возможности доступные через SQL недоступны в нём.

1.2. Конфигурация

Перед тем как установить и запустить Firebird как службу необходимо выбрать режим работы сервера.

1.2.1. Режим сервера

По-умолчанию Firebird будет стартовать в режиме SuperServer. Если вы хотите чтобы сервер запускался в другой архитектуре, то необходимо изменить значение параметра `ServerMode` в `firebird.conf`. Раскомментируйте его (удалите решётку) и установите нужный режим: Super, SuperClassic или Classic.

```
ServerMode = Classic
```

1.2.2. Авторизация из клиентских библиотек Firebird 2.5

В Firebird 4.0 по умолчанию используется безопасная парольная аутентификация (SRP). Клиенты Firebird 2.5 и более ранние версии использовали традиционную аутентификацию (Legacy_Auth), которая отключена в Firebird 4.0 по-умолчанию, поскольку не является безопасной.

Для поддержки традиционной аутентификации необходимо изменить следующие параметры `AuthServer`, `UserManager` и `WireCrypt`.

Пример 3. Включение авторизации с предыдущими версиями клиента Firebird

```
AuthServer = Srp256, Srp, Legacy_Auth
userManager = Srp, Legacy_UserManager
WireCrypt = Enabled
```

После вышеперечисленных манипуляций у нас будет активно два менеджера пользователей, по умолчанию активен первый в списке `UserManager`.

Важно



Одноименные пользователи в разных менеджерах пользователей — это разные пользователи, и у них могут быть разные пароли. Это относится и к SYSDBA и владельцу базы данных.



Если вам не нужна поддержка безопасной парольной аутентификации (SRP), удалите из `AuthServer` плагины `Srp256` и `Srp`; из `UserManager` — `Srp`, а `WireCrypt` можете изменить на `Disabled`.

Ранее мы уже создали SYSDBA в менеджере пользователей SRP. В `Legacy_UserManager` SYSDBA уже существует, причём со стандартным паролем *masterkey*, который необходимо изменить. Сделаем это с использованием инструмента `isql`. В операторе `ALTER USER` необходимо

обязательно указать менеджер пользователей Legacy_UserNameManager.

Пример 4. Изменение пароля SYSDBA в Legacy_UserNameManager

```
c:\Firebird\4.0>isql
Use CONNECT or CREATE DATABASE to specify a database
SQL> connect security.db user sysdba;
Database: security.db, User: SYSDBA
SQL> ALTER USER SYSDBA SET PASSWORD 'er34gfde' USING PLUGIN Legacy_UserNameManager;
SQL> exit;

c:\Firebird\4.0>
```

1.2.3. Установка часового пояса соединения по умолчанию

В Firebird 4.0 введены новые типы даты и времени с поддержкой часовых поясов.

Даже если вы не собираетесь в ближайшее время использовать типы с часовыми поясами, то необходимо учитывать, что выражения CURRENT_TIMESTAMP и CURRENT_TIME теперь возвращают типы данных с часовыми поясами. Существует [режим совместимости](#), который позволяет преобразовать типы с часовыми поясами в типы без часовых поясов. Однако такое преобразование может работать неверно, если часовой пояс соединения выставлен неправильно.

Обычно часовой пояс сеанса задаётся на стороне клиента. Если часовой пояс на стороне клиента не выставлен, то по умолчанию используется часовой пояс операционной системы. Вы также можете выставить часовой пояс сеанса по умолчанию с помощью параметра конфигурации DefaultTimeZone.

```
DefaultTimeZone = Europe/Moscow
```

1.2.4. Одновременный запуск нескольких экземпляров Firebird

Здесь предполагается, что вы хотите запустить экземпляры разных версий Firebird, каждая из которых установлена в своём каталоге.

Для одновременного запуска нескольких экземпляров Firebird необходимо развести их по разным портам tcp (если, конечно, слушатель запущен в режиме прослушивания TCP/IP). Для этого необходимо изменить в firebird.conf параметр RemoteServicePort. Например, если у вас уже есть один сервер, который слушает порт 3050, то необходимо установить любой другой свободный порт, например 3051. В этом случае в строке подключения необходимо будет указывать новый порт. (кроме случая когда приложению и клиенту Firebird доступен firebird.conf с измененным номером порта по умолчанию)

```
RemoteServicePort = 3051
```

Также желательно изменить параметры *IpcName* и *RemotePipeName*. Если их оставить неизменными, то в логе того экземпляра, который будет запущен вторым, появятся ошибки. Это не является критической ошибкой, если вы не пользуетесь протоколами XNET и WNET. Однако, если они используются, то стоит учесть, что эти параметры придётся изменять и на стороне клиента через DPB.

1.3. Установка и запуск Firebird как службы

Утилита *instsvc.exe* записывает, удаляет или меняет информацию о запуске сервера в базе сервисов операционной системы. Кроме того, она позволяет управлять запуском и остановкой сервиса.

Если запустить её без параметров, то будет выведена справка по командам и параметрам.

```
instsvc
Usage:
  instsvc i[nstall]
                [ -a[uto]* | -d[emand] ]
                [ -g[uardian] ]
                [ -l[ogin] username [password] ]
                [ -n[ame] instance ]
                [ -i[nteractive] ]

  sta[rt]      [ -b[oostpriority] ]
                [ -n[ame] instance ]
  sto[p]       [ -n[ame] instance ]
  q[ue]ry
  r[emove]     [ -n[ame] instance ]

'*' denotes the default values
'-z' can be used with any other option, prints version
'username' refers by default to a local account on this machine.
Use the format 'domain\username' or 'server\username' if appropriate.
```



Важно

Утилита *instsvc* должна запускаться в консоли с административными привилегиями (запуск консоли от имени администратора).

Для установки сервиса необходимо ввести команду

```
instsvc install
```

В этом случае Firebird будет установлен в качестве службы с именем "Firebird Server – DefaultInstance". Эта служба будет запускаться автоматически при старте ОС, под учётной записью LocalSystem, предназначенной для служб.

Если необходимо чтобы было установлено несколько экземпляров Firebird работающих как службы, то необходимо задать им разные имена с помощью опции -n

```
instsvc install -n fb40
```

Для запуска службы воспользуйтесь командой

```
instsvc start
```

Если служба была установлена с именем отличным от умолчательного, то необходимо воспользоваться переключателем -n

```
instsvc start -n fb40
```

Для остановки службы воспользуйтесь командой

```
instsvc stop
```

Если служба была установлена с именем отличным от умолчательного, то необходимо воспользоваться переключателем -n

```
instsvc stop -n fb40
```

Для удаления сервиса необходимо ввести команду

```
instsvc remove
```

Если служба была установлена с именем отличным от умолчательного, то необходимо воспользоваться переключателем -n

```
instsvc remove -n fb40
```

Для просмотра всех служб Firebird установленных в системе воспользуйтесь командой

```
instsvc query
```

```
Firebird Server - fb30 IS installed.  
Status   : running  
Path     : C:\Firebird\3.0\firebird.exe -s fb30  
Startup  : automatic  
Run as   : LocalSystem
```

```
Firebird Server - fb40 IS installed.  
Status   : running  
Path     : C:\Firebird\4.0\firebird.exe -s fb40  
Startup  : automatic  
Run as   : LocalSystem
```

1.3.1. Использование `install_service.bat` и `uninstall_service.bat`

Для упрощения процедуры установки и удаления служб в ZIP архиве в комплекте с Firebird поставляются два BAT файла: `install_service.bat` и `uninstall_service.bat`.

В этом случае процедура установки Firebird в качестве сервиса выглядит следующим образом

```
install_service.bat
```

В этом случае процедура удаления службы Firebird выглядит следующим образом

```
uninstall_service.bat
```

Если необходимо задать службе имя отличное от умолчательного, то указываем это имя в качестве аргумента

```
install_service.bat fb40
```

Если служба была установлена с именем отличным от умолчательного, то указываем это имя в качестве аргумента

```
uninstall_service.bat fb40
```

1.4. Установка клиента

Если речь идёт об установке только клиентской части, то обязательно требуется файл `fbclient.dll`. Клиент Firebird 4.0 обязательно требует наличия установленного Microsoft Runtime C++ 2017 соответствующей разрядности. Если данная библиотека не установлена, то можно скопировать дополнительные библиотеки, которые поставляются в ZIP архиве под Windows `msvcpr140.dll` и `vcruntime140.dll`.

Желательно, чтобы рядом с `fbclient.dll` был расположен файл сообщений `firebird.msg`. Большинство сообщений об ошибках уже содержатся в `fbclient.dll`, однако если вы собираетесь пользоваться консольными утилитами файл `firebird.msg` обязательно должен присутствовать.

В отличие от Firebird 2.5 и Firebird 3.0 клиентской библиотеки так же требуются файлы ICU (`icudt63.dll`, `icuin63.dll`, `icuuc63.dll` и `icudt63l.dat`). Ранее ICU библиотека требовалась только серверу. Теперь она может потребоваться клиентской части, если вы собираетесь работать с типами данных `TIMESTAMP WITH TIME ZONE` и `TIME WITH TIME ZONE`. ICU библиотека также требуется при вызове функций `UtilInterface::decodeTimeTz()` и `UtilInterface::decodeTimestampTz()`.



Замечание

В Windows 10 может использоваться ICU библиотека поставляемая вместе с операционной системой.

Если необходимо сжатие трафика при работе по TCP/IP, то потребуется библиотека `zlib1.dll`.

Вам может потребоваться библиотека `plugins/chacha.dll`, если вы собираетесь использовать плагин шифрования трафика ChaCha. Этот плагин, используется по умолчанию в Firebird 4.0, поскольку он находится первым списке значений в параметре конфигурации `WireCryptPlugin = ChaCha, Arc4`.

Замечание о загрузке плагинов

fbclient.dll версии 3.0 по умолчанию не загружал плагины из динамических библиотек из каталога plugins. fbclient.dll версии 4.0 использует plugins/chacha.dll по умолчанию, если этот плагин присутствует. Отсутствующие плагины игнорируются.

Однако, есть важная особенность. fbclient.dll ищет в своём каталоге файл firebird.conf, и если он отсутствует, то пытается найти его на каталог выше. Каталог где будет найден firebird.conf является корневым каталогом — от которого отсчитываются все остальные известные относительные пути (plugins, intl).



Такое поведение может сыграть с вами злую шутку. Дело в том что 64-разрядный инсталлятор располагает в папке \$(fbroot)/WOW64 32-разрядную библиотеку fbclient.dll. Если захотите использовать библиотеку из данного каталога, то можете получить следующее сообщений об ошибке

```
Error loading plugin ChaCha.  
Module C:\Firebird\4.0\plugins\ChaCha exists but can not be loaded.  
unknown Win32 error 193.
```

В данном случае 32-разрядный fbclient.dll загрузить 64-разрядный плагин ChaCha.

Для исправления данной ошибки достаточно поместить в папку \$(fbroot)/WOW64 файл firebird.conf.

Для того чтобы клиентское приложение могло загрузить библиотеку fbclient.dll, а также другие файлы клиентской библиотеки, должны располагаться либо рядом с приложением, либо в одной из директорий в которой производится поиск, например добавленной в PATH или системной директории для размещения общедоступных библиотек (system32 или SysWOW64).

Важно



Размещение клиентской библиотеки в PATH может помешать другим приложениям, которым требуется клиентская библиотека другой версии или другого сервера. Поэтому, если предполагается, что приложение должно работать независимо от других приложений с конкретной версией клиента, то файлы клиента требуется разместить в папке приложения, и не прописывать этот путь в PATH.

1.4.1. Утилита instclient

Для развёртывания клиентской библиотеки Firebird в системном каталоге Windows воспользуйтесь командой

```
instclient install fbclient
```

**Важно**

Утилита `instclient` не копирует в системный каталог никаких файлов кроме `fbclient.dll`.

1.5. Установка embedded версии

Embedded версия теперь требует большего количества файлов. Минимальная структура файлов и каталогов для Firebird 4.0 embedded следующая:

- intl
 - fbintl.conf
 - fbintl.dll
- plugins
 - engine13.dll
- firebird.conf
- icudt63l.dat
- fbclient.dll
- ib_util.dll
- icudt63.dll
- icuin63.dll
- icuuc63.dll
- msvcp140.dll
- vcruntime140.dll
- firebird.msg

При необходимости вы также можете скопировать исполняемые файлы утилит `fbsvcmgr.exe`, `fbtracemgr.exe`, `gbak.exe`, `gfix.exe`, `gstat.exe`, `isql.exe`, `nbackup.exe`.

Замечание

Для тех кто мигрирует с Firebird 2.5 следует учитывать 2 момента:



- Вместо единой библиотеки `fbembed.dll` требуется несколько файлов, причём файл `fbclient.dll` переименовывать нельзя. Компоненты доступа должны использовать в качестве точки входа именно библиотеку `fbclient.dll`.
- В файле конфигурации `firebird.conf` следует изменить значение параметра `ServerMode` на `SuperClassic` или `Classic` для того чтобы на одном компьютере можно было подключаться к одной и той же базе данных из разных приложений, использующих `embedded` (поведения Firebird 2.5 `embedded` по умолчанию).

2. Преобразование базы данных к новому формату

Базы данных Firebird 4.0 имеют ODS (On-Disk Structure) 13.0. Для того чтобы Firebird 4.0 мог работать с вашей базой данных её необходимо привести к родной ODS. Обычно это осуществляется с помощью инструмента *gbak*. Однако не торопитесь делать резервное копирование своей БД и её восстановление с новой ODS — сначала необходимо устранить возможные проблемы совместимости.

2.1. Список несовместимостей на уровне языка SQL

Проблемы совместимости языка SQL возможны как для объектов самой базы данных (PSQL процедуры и функции), так и в DSQL запросах, используемых в вашем приложении.

Перечислим некоторые наиболее часто встречающиеся проблемы совместимости на уровне SQL которые вы можете исправить ещё до перехода на новую ODS. Полный список несовместимостей вы можете прочитать в Release Notes 3.0 (для тех кто переходит с Firebird 2.5) и Release Notes 4.0 в главе "Compatibility Issues".

2.1.1. Новые зарезервированные слова

Проверьте вашу базу данных на наличие новых зарезервированных слов в идентификаторах, столбцах и переменных. В первом SQL диалекте такие слова не могут применяться в принципе (надо будет переименовать), в третьем — могут применяться, но должны обрамляться двойными кавычками. Список ключевых и зарезервированных слов вы можете найти в Release Notes 3.0 и 4.0 в главе "Reserved Words and Changes". Ключевые слова могут применяться в качестве идентификаторов, хотя это не рекомендуется.

2.1.2. Имена столбцов в PSQL курсорах

Этот пункт актуален для тех кто переходит с Firebird 2.5. Все выходные столбцы в PSQL курсорах объявленных как `DECLARE CURSOR` должны иметь явное имя или псевдоним. То же

самое касается PSQL курсоров используемых как FOR SELECT ... AS CURSOR <cursor name> DO

Пример 5. Проблема с неименованными столбцами в курсорах

```
create procedure sp_test
returns (n int)
as
  declare c cursor for (select 1 /* as a */ from rdb$database);
begin
  open c;
  fetch c into n;
  close c;
  suspend;
end
```

```
Statement failed, SQLSTATE = 42000
unsuccessful metadata update
-ALTER PROCEDURE SP_TEST failed
-Dynamic SQL Error
-SQL error code = -104
-Invalid command
-no column name specified for column number 1 in derived table C
```

2.1.3. Новые типы данных

В Firebird 4.0 введены новые типы данных:

- TIMESTAMP WITH TIME ZONE
- TIME WITH TIME ZONE
- INT128
- NUMERIC(38, x) и DECIMAL(38, x)
- DECFLOAT(16) и DECFLOAT(34)

Последние два типа не вызывают особых проблем, поскольку раньше вы их не использовали, и обычно выражения их не возвращают.

Некоторые выражения теперь могут возвращать типы NUMERIC(38, x), DECIMAL(38, x) и INT128. О решении этой проблемы мы поговорим позже, поскольку на этапе изменения ODS они обычно не проявляются.

Выражения CURRENT_TIMESTAMP и CURRENT_TIME теперь возвращают типы TIMESTAMP WITH TIME ZONE и TIME WITH TIME ZONE. Это может стать серьезной проблемой.

Для старых клиентских библиотек и приложений вы можете установить [режим совместимости типов](#), однако это не поможет внутри хранимых процедур, функций и

триггеров. Вам необходимо использовать выражения LOCALTIMESTAMP и LOCALTIME вместо CURRENT_TIMESTAMP и CURRENT_TIME там где вы не хотите получить типы данных с часовыми поясами. Данные выражения специально были введены в корректирующих релизах Firebird 2.5.9 и Firebird 3.0.4, чтобы вы заранее могли подготовить свои базы данных для миграции на Firebird 4.0.

2.1.4. Литералы дат и времени

В Firebird 4.0 ужесточён синтаксис литералов дат и времени.

Литералы 'NOW', 'TODAY', 'TOMORROW', 'YESTERDAY' с неявным приведением типа (с префиксами TIMESTAMP, DATE, TIME) теперь запрещены. Дело в том, что значение таких литералов вычислялось во время подготовки DSQL запроса или компиляции PSQL модулей, что приводило к неожиданным результатам.

Если что-то вроде TIMESTAMP 'NOW' использовалось в запросах DSQL в коде приложения или в перенесенном PSQL, возникнет проблема совместимости с Firebird 4.

Пример 6. Следующий код не будет скомпилирован

```
..
DECLARE VARIABLE moment TIMESTAMP;
..
SELECT TIMESTAMP 'NOW' FROM RDB$DATABASE INTO :moment;

/* здесь переменная: moment будет "заморожена" как отметка времени
в момент последней компиляции процедуры или функции */
..
```

Необходимо вычистить такие литералы (например заменить на явное преобразование типа CAST('NOW' AS TIMESTAMP)) в коде ваших процедур и функций до преобразования вашей базы данных в новую ODS.

Кроме того, необходимо проверить другие литералы дат и времени с явным заданием известной даты (времени). Ранее в таких литералах позволялись разделители частей даты и времени не соответствующие стандарту, теперь такие разделители запрещены. Подробнее о разрешённых форматах литералов даты и времени вы можете прочитать в "Руководство по языку SQL СУБД Firebird 4.0" в главе "Литералы даты и времени".

2.2. Поддержка внешних функций (UDF) объявлена устаревшей

Поддержка внешних функций (UDF) в Firebird 4 объявлена устаревшей. Эффект от этого заключается в том, что UDF нельзя использовать с конфигурацией по умолчанию, поскольку для параметра UdfAccess в firebird.conf значение по умолчанию теперь None. Библиотеки UDF ib_udf и fbudf изъяты из дистрибутива.

Большинство функций в этих библиотеках уже устарели в предыдущих версиях Firebird и были заменены встроенными аналогами. Теперь доступны безопасные замены для некоторых из оставшихся функций либо в новой библиотеке определяемых пользователем подпрограмм (UDR) с именем [lib]udf_compat.[dll/so/dylib] (это делается после смены ODS), либо в виде преобразований по сценарию в сохраненные функции PSQL.

Рекомендуем заранее (до перехода на новую ODS) заменить UDF функции на их встроенные аналоги. Если вы делаете миграцию с Firebird 3.0, вы также можете переписать часть функций на PSQL.

Если после этих шагов у вас остались UDF функции, то необходимо изменить параметр конфигурации

```
UdfAccess = Restrict UDF
```

2.3. Преобразование базы данных к новой ODS

После предварительной подготовки, вы можете попробовать преобразовать базу данных к новой ODS с помощью инструмента *gbak*.

В данном примере предполагается, что на одной машине стоят Firebird 3.0 и Firebird 4.0. Firebird 3.0 работает используя TCP порт 3053, а Firebird 4.0 — 3054.

Прежде всего необходимо создать резервную копию вашей базы данных на текущей версии Firebird с помощью следующей команды.

```
gbak -b -g -V -user <username> -pas <password> -se <service> <database> <backup_file>
-Y <log_file>
```

Пример 7. Создание резервной копии на текущей версии Firebird

```
gbak -b -g -V -user SYSDBA -pas 8kej712 -se server/3053:service_mgr my_db
d:\fb30_backup\my_db.fbk -Y d:\fb30_backup\backup.log
```

Далее необходимо восстановить вашу копию на Firebird 4.0 с помощью команды

```
gbak -c -v -user <username> -pas <password> -se <service> <backup_file>
<database_file> -Y <log_file>
```

```
gbak -c -v -user SYSDBA -pas 8kej712 -se server/3054:service_mgr  
d:\fb30_backup\my_db.fbk d:\fb40_data\my_db.fdb -Y d:\fb40_data\restore.log
```



Важно

Обратите внимание, на переключатели -V и -Y, они обязательно должны использоваться, чтобы вы могли просмотреть в лог файле, что в процессе восстановления пошло не так.

После восстановления внимательно изучите restore.log на предмет ошибок. Однако, в этом логе не будет ошибок несовместимости уровня SQL, поскольку объекты БД при restore не перекомпилируются. Если какая-то процедура или триггер содержат несовместимые конструкции, то впоследствии при alter такого объекта будет выдана ошибка.

Полностью очистить БД от таких ошибок можно только если извлечь скрипт из БД операцией

```
isql -x <database> > script.sql
```

в предыдущей версии Firebird, и создать пустую БД в Firebird 4.0 из этого скрипта, исправляя возникающие ошибки создания метаданных по очереди.

2.3.1. Предупреждения об отсутствии UDF

После восстановления в файле restore.log вы можете увидеть следующие предупреждения

```
gbak: WARNING:function UDF_FRAC is not defined  
gbak: WARNING: module name or entrypoint could not be found
```

Это означает, что у вас есть UDF, которые объявлены в базе данных, но их библиотека отсутствует. Выше уже было описано, что надо делать в этом случае. Но это в основном касалось ваших UDF библиотек. Однако если вы использовали UDF из комплекта поставляемого с Firebird, а именно ib_udf и fbudf, то вы можете заменить их на встроенные функции или на безопасные аналоги UDR расположенные в библиотеке udf_compat.dll. Для этого необходимо запустить SQL скрипт миграции, поставляемый в комплекте с Firebird 4.0, который расположен в misc/upgrade/v4.0/udf_replace.sql. Это делается следующей командой

```
isql -user sysdba -pas masterkey -i udf_replace.sql {your-database}
```


Этот сценарий не повлияет на объявления UDF из сторонних библиотек!

3. Перенос псевдонимов баз данных

Этот раздел актуален для тех кто мигрирует с Firebird 2.5. Файл `aliases.conf` в котором настраивались псевдонимы баз данных переименован в `databases.conf`. Он полностью обратно совместим по синтаксису, однако его назначение значительно расширено. Теперь в нём можно задавать некоторые индивидуальные параметры для каждой базы данных. Настоятельно рекомендуем воспользоваться этой возможностью, если ваш сервер обслуживает более 1 базы данных.

Параметры, которые можно задавать на уровне базы данных, помечены в файле `firebird.conf` надписью 'Per-database configurable'.

4. Перенос списка пользователей

Перенос списка пользователей из Firebird 2.5 и Firebird 3.0 осуществляется по-разному. Перенос пользователей из Firebird 3.0 производится значительно проще, поэтому сначала опишем этот случай.

4.1. Перенос списка пользователей из Firebird 3.0

Самый простой способ перенести пользователей из базы безопасности Firebird 3.0 в базу данных безопасности Firebird 4.0 заключается в резервной копии `security3.fdb` с помощью *gbak* и восстановлении её в Firebird 4.0. Однако учтите, что в этом случае вы потеряете некоторые новые возможности. Мы пойдём более сложным способом:

1. Сделайте резервную копию базы данных безопасности на Firebird 3.0

```
c:\Firebird\3.0>gbak -b -g -user SYSDBA security.db d:\fb30_backup\security.fbk
```

2. Восстановите резервную копию на Firebird 4.0 под новым именем

```
c:\Firebird\4.0>gbak -c -user SYSDBA -pas 8kej712 -se localhost/3054:service_mgr  
d:\fb30_backup\security.fbk d:\fb40_data\security_30.fdb
```

3. Сохраните следующий скрипт для переноса пользователей в файл `copy_user.sql`

```
set term ^;  
  
EXECUTE BLOCK  
AS
```

```

-- замените на параметры вашей копии БД безопасности
DECLARE SRC_SEC_DB    VARCHAR(255) = 'd:\fb40_data\security_30.fdb';
DECLARE SRC_SEC_USER  VARCHAR(63) = 'SYSDBA';

-----

DECLARE PLG$USER_NAME SEC$USER_NAME;
DECLARE PLG$VERIFIER  VARCHAR(128) CHARACTER SET OCTETS;
DECLARE PLG$SALT      VARCHAR(32) CHARACTER SET OCTETS;
DECLARE PLG$COMMENT   BLOB SUB_TYPE TEXT CHARACTER SET UTF8;
DECLARE PLG$FIRST     SEC$NAME_PART;
DECLARE PLG$MIDDLE    SEC$NAME_PART;
DECLARE PLG$LAST      SEC$NAME_PART;
DECLARE PLG$ATTRIBUTES BLOB SUB_TYPE TEXT CHARACTER SET UTF8;
DECLARE PLG$ACTIVE    BOOLEAN;
DECLARE PLG$GROUP_NAME SEC$USER_NAME;
DECLARE PLG$UID       PLG$ID;
DECLARE PLG$GID       PLG$ID;
DECLARE PLG$PASSWD    PLG$PASSWD;
BEGIN
  -- перемещаем пользователей из плагина SRP
  FOR EXECUTE STATEMENT Q'!
    SELECT
      PLG$USER_NAME,
      PLG$VERIFIER,
      PLG$SALT,
      PLG$COMMENT,
      PLG$FIRST,
      PLG$MIDDLE,
      PLG$LAST,
      PLG$ATTRIBUTES,
      PLG$ACTIVE
    FROM PLG$SRP
    WHERE PLG$USER_NAME <> 'SYSDBA'
  !'
    ON EXTERNAL :SRC_SEC_DB
    AS USER :SRC_SEC_USER
    INTO :PLG$USER_NAME,
        :PLG$VERIFIER,
        :PLG$SALT,
        :PLG$COMMENT,
        :PLG$FIRST,
        :PLG$MIDDLE,
        :PLG$LAST,
        :PLG$ATTRIBUTES,
        :PLG$ACTIVE
  DO
  BEGIN
    INSERT INTO PLG$SRP (
      PLG$USER_NAME,
      PLG$VERIFIER,
      PLG$SALT,
      PLG$COMMENT,

```

```

        PLG$FIRST,
        PLG$MIDDLE,
        PLG$LAST,
        PLG$ATTRIBUTES,
        PLG$ACTIVE)
VALUES (
    :PLG$USER_NAME,
    :PLG$VERIFIER,
    :PLG$SALT,
    :PLG$COMMENT,
    :PLG$FIRST,
    :PLG$MIDDLE,
    :PLG$LAST,
    :PLG$ATTRIBUTES,
    :PLG$ACTIVE);
END
-- перемещаем пользователей из плагина Legacy_UserManager
FOR EXECUTE STATEMENT Q'!
    SELECT
        PLG$USER_NAME,
        PLG$GROUP_NAME,
        PLG$UID,
        PLG$GID,
        PLG$PASSWD,
        PLG$COMMENT,
        PLG$FIRST_NAME,
        PLG$MIDDLE_NAME,
        PLG$LAST_NAME
    FROM PLG$USERS
    WHERE PLG$USER_NAME <> 'SYSDBA'
!'

    ON EXTERNAL :SRC_SEC_DB
    AS USER :SRC_SEC_USER
    INTO :PLG$USER_NAME,
        :PLG$GROUP_NAME,
        :PLG$UID,
        :PLG$GID,
        :PLG$PASSWD,
        :PLG$COMMENT,
        :PLG$FIRST,
        :PLG$MIDDLE,
        :PLG$LAST
DO
BEGIN
    INSERT INTO PLG$USERS (
        PLG$USER_NAME,
        PLG$GROUP_NAME,
        PLG$UID,
        PLG$GID,
        PLG$PASSWD,
        PLG$COMMENT,

```

```

        PLG$FIRST_NAME,
        PLG$MIDDLE_NAME,
        PLG$LAST_NAME)
VALUES (
    :PLG$USER_NAME,
    :PLG$GROUP_NAME,
    :PLG$UID,
    :PLG$GID,
    :PLG$PASSWD,
    :PLG$COMMENT,
    :PLG$FIRST,
    :PLG$MIDDLE,
    :PLG$LAST);

END
END^

set term ;^

commit;

exit;

```



Важно

Не забудьте заменить значение переменной SRC_SEC_DB на путь к копии вашей БД безопасности.



Замечание

Мы исключили копию пользователя SYSDBA, поскольку инициализировали его при установке.

4. Выполните скрипт на Firebird 4.0 подключившись к БД безопасности в embedded режиме

```

c:\Firebird\4.0>isql -i "d:\fb40_data\copy_users.sql" -u SYSDBA -ch UTF8
security.db

```

Поздравляем ваши пользователи перенесены с сохранением всех атрибутов и паролей.

4.2. Перенос списка пользователей из Firebird 2.5

Перенос пользователей из Firebird 2.5 более сложен. В Firebird 3.0 ввели новый способ аутентификации SRP - Secure Remote Password Protocol. Старый способ аутентификации также доступен, но выключен по умолчанию поскольку считается недостаточно безопасным. В Release Notes 3.0 описан способ переноса пользователей из Legacy_UserManager в SRP, однако в этом случае вы не сможете подключаться через fbclient версии 2.5. Кроме того, перенести пароли из Legacy_UserManager в SRP невозможно.

Предлагаемый скрипт перенесёт список пользователей, но будут сгенерированы случайные пароли. Если вы хотите восстановить прежние пароли, то это придётся делать вручную. Я написал альтернативный скрипт, который позволяет перенести пользователей из security2.fdb в security3.fdb в плагин Legacy_UserManager. Здесь я опишу оба варианта.

4.2.1. Копирование списка пользователей в плагин SRP

Из-за новой модели аутентификации в Firebird 3 обновление базы данных безопасности версии 2.5 (security2.fdb) напрямую для использования в Firebird 4 невозможно. Однако существует процедура обновления, позволяющая сохранить данные учетной записи пользователя — имя пользователя, имя и т. Д., Но не пароли — из базы данных security2.fdb, которая использовалась на серверах версии 2.x.

Процедура требует запуска сценария security_database.sql, который находится в каталоге misc/upgrade вашей установки Firebird 3. Эти инструкции предполагают, что у вас есть временная копия этого сценария в том же каталоге, что и исполняемый файл isql.



Замечание

- В Firebird 4.0 файл сценария обновления БД безопасности security_database.sql отсутствует в каталоге misc/upgrade, поэтому вам необходимо скачать zip архив с дистрибутивом Firebird 3.0.
- В приведенных ниже командах замените *masterkey* фактическим паролем SYSDBA для вашего сервера, если это необходимо.

1. Сделайте резервную копию БД безопасности security2.fdb на Firebird 2.5

```
c:\Firebird\2.5>bin\gbak -b -g -user SYSDBA -password masterkey -se service_mgr
c:\Firebird\2.5\security2.fdb d:
\fb25_backup\security2.fbk
```

2. Разверните резервную копию на Firebird 4.0

```
c:\Firebird\4.0>gbak -c -user SYSDBA -password masterkey -se
localhost/3054:service_mgr d:\fbdata\4.0\security2.fbk d:\f
bdata\4.0\security2db.fdb -v
```

3. На сервере Firebird 4.0 перейдите в каталог, в котором находится утилита isql, и запустите сценарий обновления:

```
isql -user sysdba -pas masterkey -i security_database.sql
{host/path}security2db.fdb
```

security2db.fdb - это просто пример имени базы данных: это может быть любое предпочтительное имя.

4. Процедура генерирует новые случайные пароли и затем выводит их на экран. Скопируйте вывод и уведомите пользователей об их новых паролях.

4.2.2. Копирование списка пользователей в плагин Legacy_UserManager

В отличие от предыдущего варианта, данный скрипт сохранит ваши исходные пароли. Однако, мы советуем вам в будущем всё равно перейти на плагин Srp.

1. Сделайте резервную копию БД безопасности security2.fdb на Firebird 2.5

```
c:\Firebird\2.5>bin\gbak -b -g -user SYSDBA -password masterkey -se service_mgr
c:\Firebird\2.5\security2.fdb d:
\fb25_backup\security2.fbk
```

2. Разверните резервную копию на Firebird 4.0

```
c:\Firebird\4.0>gbak -c -user SYSDBA -password masterkey -se
localhost/3054:service_mgr d:\fbdata\4.0\security2.fbk d:\f
bdata\4.0\security2db.fdb -v
```

3. Сохраните следующий скрипт для переноса пользователей в файл copy_security2.sql

```
set term ^;

EXECUTE BLOCK
AS
    -- замените на параметры вашей копии БД безопасности
    DECLARE SRC_SEC_DB      VARCHAR(255) = 'd:\fbdata\4.0\security2.fdb';
    DECLARE SRC_SEC_USER    VARCHAR(63) = 'SYSDBA';
    -----
    DECLARE PLG$USER_NAME   SEC$USER_NAME;
    DECLARE PLG$COMMENT     BLOB SUB_TYPE TEXT CHARACTER SET UTF8;
    DECLARE PLG$FIRST       SEC$NAME_PART;
    DECLARE PLG$MIDDLE      SEC$NAME_PART;
    DECLARE PLG$LAST        SEC$NAME_PART;
    DECLARE PLG$GROUP_NAME  SEC$USER_NAME;
    DECLARE PLG$UID         INT;
    DECLARE PLG$GID         INT;
    DECLARE PLG$PASSWD      VARBINARY(64);
BEGIN
    FOR EXECUTE STATEMENT q'!
        SELECT
            RDB$USER_NAME,
            RDB$GROUP_NAME,
            RDB$UID,
            RDB$GID,
            RDB$PASSWD,
            RDB$COMMENT,
```

```

        RDB$FIRST_NAME,
        RDB$MIDDLE_NAME,
        RDB$LAST_NAME
    FROM RDB$USERS
    WHERE RDB$USER_NAME <> 'SYSDBA'
!'
    ON EXTERNAL :SRC_SEC_DB
    AS USER :SRC_SEC_USER
    INTO
        :PLG$USER_NAME,
        :PLG$GROUP_NAME,
        :PLG$UID,
        :PLG$GID,
        :PLG$PASSWD,
        :PLG$COMMENT,
        :PLG$FIRST,
        :PLG$MIDDLE,
        :PLG$LAST
DO
BEGIN
    INSERT INTO PLG$USERS (
        PLG$USER_NAME,
        PLG$GROUP_NAME,
        PLG$UID,
        PLG$GID,
        PLG$PASSWD,
        PLG$COMMENT,
        PLG$FIRST_NAME,
        PLG$MIDDLE_NAME,
        PLG$LAST_NAME)
    VALUES (
        :PLG$USER_NAME,
        :PLG$GROUP_NAME,
        :PLG$UID,
        :PLG$GID,
        :PLG$PASSWD,
        :PLG$COMMENT,
        :PLG$FIRST,
        :PLG$MIDDLE,
        :PLG$LAST);
END
END^

set term ;^

commit;

exit;

```



Важно

Не забудьте заменить значение переменной SRC_SEC_DB на путь к копии вашей БД безопасности.



Замечание

Мы исключили копию пользователя SYSDBA, поскольку инициализировали его при установке.

4. Выполните скрипт на Firebird 4.0 подключившись к БД безопасности в embedded режиме

```
c:\Firebird\4.0>isql -i "d:\fb40_data\copy_security2.sql" -u SYSDBA -ch UTF8
security.db
```

Поздравляем! Ваши пользователи перенесены с сохранением всех атрибутов и паролей.

5. Настройка доверительной аутентификации

Настройка доверительной аутентификации (если она необходима) в Firebird 4.0 делается точно так же как она делалась в Firebird 3.0. Для тех производит миграцию с Firebird 2.5 опишем этот процесс подробнее.

1. Первым делом необходимо подключить плагин доверительной аутентификации в файле конфигурации `firebird.conf` или `databases.conf` в параметре `AuthServer` (по умолчанию он отключен). Для этого необходимо добавить плагин с именем `Win_Sspi`, и будем использовать его совместно с `Srp256`.

```
AuthServer = Srp256, Win_Sspi
```

2. Следующим шагом необходимо включить отображение пользователей из `Win_Sspi` на `CURRENT_USER`. Для этого необходимо создать отображение в целевой базе данных с помощью следующего запроса

```
CREATE MAPPING TRUSTED_AUTH
USING PLUGIN WIN_SSPI
FROM ANY USER
TO USER;
```

Данный SQL запрос создаёт отображение только на уровне текущей базе данных. Отображение не будет применяться к другим базам данных расположенных на том же сервере. Если вы хотите создать общее отображение для всех баз данных, то добавьте ключевое слово `GLOBAL`.


```
CREATE GLOBAL MAPPING TRUSTED_AUTH
USING PLUGIN WIN_SSPI
FROM ANY USER
TO USER;
```

3. Включение SYSDBA-подобного доступа для администраторов Windows (если он нужен).

Для включения такого доступа необходимо создать следующее отображение

```
CREATE MAPPING WIN_ADMINS
USING PLUGIN WIN_SSPI
FROM Predefined_Group
DOMAIN_ANY_RID_ADMINS
TO ROLE RDB$ADMIN;
```

Вместо включения SYSDBA-подобного доступа для всех администраторов Windows, вы можете дать административные привилегии конкретному пользователю с помощью следующего отображения

```
create global mapping cto_sysdba
using plugin win_sspi
from user "STATION9\DEVELOPER"
to user SYSDBA;
```

6. Несовместимости на уровне приложения

На уровне API клиентская библиотека fbclient 4.0 совместима с предыдущими версиями. Однако могут возникнуть проблемы совместимости на уровне некоторых SQL запросов. Большинство из них мы уже описывали ранее в разделе [Список не совместимостей на уровне языка SQL](#). Далее опишем некоторые другие проблемы, которые могут возникнуть на уровне языка SQL.

6.1. Новые типы данных

Как уже говорилось ранее, некоторые выражения могут возвращать новые типы данных, которые не могут быть интерпретированы вашим приложением без его доработки. Такая доработка может занять существенное время или оказаться вам не по силам. Для упрощения миграции на новые версии вы можете установить параметр *DataTypeCompatibility* в режим совместимости с необходимой версией в `firebird.conf` или `databases.conf`.

```
DataTypeCompatibility = 3.0
```

Это самый быстрый путь добиться совместимости с новыми типами данных. Однако со временем вы можете начать внедрять поддержку новых типов в своё приложение. Естественно, это будет происходить постепенно - сначала один тип, потом другой и так далее. В этом случае вам надо настроить отображение тех типов, поддержку которых вы ещё не доделали, на другие типы данных. Для этого используется оператор SET BIND OF.

Синтаксис

```
SET BIND OF { <type-from> | TIME ZONE } TO { <type-to> | LEGACY | NATIVE | EXTENDED }
```

Подробное описание этого оператора есть в "Firebird 4.0 Release Notes" и "Руководство по языку SQL СУБД Firebird 4.0". С помощью него вы можете управлять отображением новых типов в вашем приложении выполнив соответствующий запрос сразу после подключения, и даже написать AFTER CONNECT триггер в котором использовать несколько таких операторов.

Например, предположим, что вы добавили в ваше приложение поддержку даты и времени с часовыми поясами, но у вас до сих пор не поддерживаются типы INT128 и DECFLOAT. В этом случае вы можете написать следующий триггер.

```
create or alter trigger tr_ac_set_bind
on connect
as
begin
    set bind of int128 to legacy;
    set bind of decfloat to legacy;
end
```

6.2. Согласованное чтение в транзакциях READ COMMITTED

Firebird 4 не только вводит согласованность чтения (READ CONSISTENCY) для запросов в транзакциях READ COMMITTED, но также делает его режимом по умолчанию для всех транзакций READ COMMITTED, независимо от их свойств RECORD VERSION или NO RECORD VERSION. Это сделано для того, чтобы обеспечить пользователям лучшее поведение — как соответствующее спецификации SQL, так и менее подверженное конфликтам. Однако это новое поведение может также иметь неожиданные побочные эффекты. Пожалуй самый важный из них это так называемые рестарты при обработке конфликтов обновления. Это может привести к тому, что некоторый код, не подверженный транзакционному контролю, может выполняться многократно в рамках PSQL. Примерами такого кода может быть:

- использование внешних таблиц, последовательностей или контекстных переменных;
- отправка электронных писем с использованием UDF;

-
- использование автономных транзакций или внешних запросов.

Подробнее о согласованном чтении в транзакциях READ COMMITTED вы можете прочитать "Firebird 4.0 Release Notes".

Другим важным эффектом является то, что недофетченные курсоры в транзакциях READ COMMITTED READ CONSISTENCY в Read Only режиме теперь удерживают сборку мусора. Рекомендуем вам отказаться от использования в приложении единой длинной READ COMMITTED READ ONLY транзакции, и заменить её на несколько таких транзакций, каждая из которых активна ровно столько времени сколько это необходимо.

Если особенности режима READ CONSISTENCY по каким-либо причинам нежелательны, то параметр конфигурации *ReadConsistency* можно изменить, чтобы разрешить устаревшее поведение.

6.3. INSERT ... RETURNING требует привилегию SELECT

Если какой-либо оператор INSERT содержит предложение RETURNING, которое ссылается на столбцы базовой таблицы, то вызывающей стороне должна быть предоставлена соответствующая привилегия SELECT.

7. Заключение

На этом всё. Надеемся, что данная статья поможет вам перевести ваши базы данных и приложения на Firebird 4.0 и воспользоваться всеми преимуществами новой версии.