Firebird – планирование резервного копирования

© kdv, iBase.ru, 03.07.2025, 17.07.2025

если вы большую базу
не бэкапили ни разу
то бэкапить и не надо
значит база не нужна

Резервное копирование данных – основа устойчивости любой организации. Александрийская библиотека сгорела, а копий документов практически не было, знания утеряны.
Для организаций, конечно, речь идет не о «знаниях» а о прошедших бизнес-процессах, на которых основывается вся деятельность.

В конце этой статьи я приведу несколько вариантов провальных стратегий резервного копирования данных, но начнем мы именно с обычных действующих вариантов, от простых к сложным – то есть, как в СУБД Firebird можно делать резервные копии базы данных, и особенно – сколько на это потребуется дискового пространства.

Файловое копирование

Самый простой способ, и одновременно опасный. Любое копирование копирует файл поблочно. А база данных – файл случайного доступа. Таким образом, если копировать БД «вживую», то вы в результате получите по факту файл с «мусором» (то же самое относится к любым утилитам файлового копирования).

Поэтому копировать файл БД, можно, в общем-то, только в двух случаях

- перевести БД в full shutdown (gfix), чтобы никто не мог подключиться, и остановить службу Firebird, скопировать файл БД, и вернуть базу в online (и запустить службу Firebird).
  Но нужно учитывать, что embedded-приложения, начиная с Firebird 3.0, могут подключиться к такой базе данных в этот период.
  В общем, такой метод лучше не использовать, мало-ли что.
- заблокировать базу nbackup –L, скопировать, разблокировать базу nbackup –N, после чего можно разблокировать копию (не базу!) nbackup –f
  (о nbackup подробнее будет написано дальше)

Дисковое пространство

Поскольку в обоих случаях вы получаете полноразмерную копию БД, то места нужно будет столько, сколько «исходный размер БД» умножить на «количество копий».

Штатное резервное копирование, gbak

По штатному резервному копированию в Firebird есть множество статей. Например, самые актуальные:

- Утилита gbak, подробное описание
  https://www.ibase.ru/gbak/
- Как делать бэкап –рестор правильно, Windows и Linux, с замерами скорости
  https://www.ibase.ru/gbak-tips-and-tricks/
- Измерение скорости диска при помощи бэкапа
  https://www.ibase.ru/backupspeed3/

Эти статьи можно использовать как для проверки, оптимально ли вы делаете бэкап-рестор, и насколько медленная или быстрая ваша дисковая подсистема.
Здесь же описываем особенности стратегий. Итак, допустим, у нас есть база данных размером 100 гигабайт.
Мы должны решить, сколько времени будет занимать бэкап, и восстановление БД из бэкапа, как часто мы можем делать бэкапы, сколько дискового пространства на это потребуется, и самое главное, нужно понимание - сколько данных мы потеряем в случае повреждения БД и восстановления БД из бэкапа.

Обычно в большинстве компаний основная нагрузка на базу данных идет в течение дня, а минимум – ночью (хотя там тоже могут выполняться разные регламентные операции).
Значит, например, если бэкап базы 100гиг идет где-то 1-2 часа, мы можем это делать только ночью. Потому что если мы решим делать бэкап днем, то он будет выполняться раз в 5 дольше, и кроме того, замедлит работу пользователей.
Таким образом, мы устанавливаем, что если мы теряем базу данных в результате каких-то катастрофических ситуаций – сбой диска, сгорание сервера, и т.д. – мы гарантировано теряем где-то от 2 до 20 часов введенных за рабочее время данных (если бэкап делается в 2-3 часа ночи, а начало работы в 8 утра).

Допустим, мы настроили бэкапы на каждую ночь, тем самым штатным gbak. Что будет происходить, когда внезапно днем база будет повреждена (опять же, сбой по питанию, сбой оборудования, или еще что-то)?
Нам нужно будет

- либо попытаться отремонтировать поврежденную базу данных
- либо восстановить базу данных из последнего бэкапа

Про специфику работы с большими базами данных есть видео моего выступления на конференции FBConf 2024:
https://rutube.ru/video/1bec9498f6e257caa1412f76037beb66/
И там я говорю о том, что «ремонт» больших баз данных почти не имеет смысла, из-за непредсказуемого количества итераций и времени ремонта.
То есть, ремонт-бэкап-ремонт-бэкап-рестор-ремонт-бэкап-рестор и т.д. Это может растянуться на несколько суток, в зависимости от размера базы данных. (Собственно, «ремонт» не всегда возможен, но возможен экспорт данных в пустую БД, однако такое делается только если нет ни бэкапов, ни других копий данных).

Поэтому, остается (при использовании только gbak) восстановление БД из последнего бэкапа.
При этом, нужно учитывать

- длительность рестора, которая обычно составляет 2-3 раза времени бэкапа. Если бэкап идет 1 час, то рестор будет идти 2-3 часа

- Если есть какие-то логические повреждения базы, то бэкап может пройти, а вот рестор – не пройти, и в результате придется исправлять базу, делать бэкап, и опять делать рестор (с успешным или неуспешным результатом).

Как правило, для штатных бэкапов рекомендуется хотя бы на 3-5 бэкапов делать тестовый рестор, чтобы убедиться что база восстановима из бэкапа. Причины проблем с рестором изложены в статье:
https://www.ibase.ru/db_repair#norestore

Да, и вот тут может возникнуть вопрос – если мы регулярно делаем бэкап и тестовый рестор, зачем мы удаляем восстановленную из бэкапа БД и сохраняем только бэкап? Ведь бэкап «читать» никак нельзя, из него нужно сначала восстановить БД, и только после этого с ней можно работать. А это плюс те самые 2-3 интервала времени, в течение которых делался бэкап. Допустим, бэкап 1 час, рестор 2-3 часа. Зачем нам тратить 2-3 часа в случае сбоя?
Может быть, выгоднее делать бэкап, рестор, потом архивировать ресторенную БД, а бэкап удалять? Подумайте об этом.

Дисковое пространство

Наконец, необходимо запланировать определенное количество дискового пространства для бэкапов и тестового рестора. Опять же, если исходим из размера базы данных в 100 гигабайт, то:

- бэкап 100гб обычно имеет размер около 60 гигабайт (индексы не бэкапятся). Плюс-минус. Значит, если хранить бэкапы за неделю, потребуется 60*7=420гиг места для бэкапов. Плюс еще 100гиг, если делается тестовый рестор. Итого, 520 гиг (впритык), а еще лучше 700 гиг «на всякий случай», если внезапно по какой-то причине база увеличится.
  В упомянутом выше докладе я также говорил, что сейчас базы растут в 2 раза каждые 2-3 года.
- плюс, если мы хотим оставлять проверочный рестор в качестве «теста», то добавьте еще 150 гиг.
- Не забудьте, что при проверочном restore в TEMP требуется достаточно места для создания индексов в восстанавливаемой БД. Если места не хватит, то restore не завершится успешно (и хотя в восстановленной базе будут все данные, но только часть индексов).

Также, понятно, что база и бэкапы должны находиться на разных физических носителях. Не на разных разделах одного RAID, а именно на разных физических дисках. Потому что если RAID помрёт, то тогда сгинут и база, и бэкапы. Еще лучше бэкапы держать на отдельном сервере, а уж как вы туда будете перекидывать файлы бэкапов – дело десятое.
Тем не менее, самый быстрый бэкап – на локальный диск (см. статьи с замерами времени выше). А уже потом можно этот бэкап скопировать по сети в какое-нибудь хранилище.

Инкрементный бэкап, nbackup

Подробное описание механизмов и параметров nbackup здесь:
https://www.ibase.ru/files/firebird/nbackup_ru.pdf

В версиях Firebird 2.x nbackup был как бы «экспериментом». Потому что он при любом уровне инкремента всегда читал всю базу данных, что крайне напрягает дисковую подсистему с базой данных (замедляет работу пользователей, и т.д.).

С версии 3.0 стало куда легче. Первичная копия БД, понятно, сканирует всю БД (nbackup –b 0), а вот дальнейшие инкременты (-b 1, -b 2…) сканируют только измененные страницы, что гораздо быстрее. Хотя, зависит от того, сколько страниц в БД меняется между этими инкрементами.

Nbackup выполняет копирование БД гораздо быстрее чем gbak, а с версии Firebird 3.0 делает инкременты совсем быстро. Всё это потому, что gbak читает данные и копирует их, а nbackup копирует целиком страницы данных без чтения внутренней структуры.

Типичная схема (пример) инкрементных бэкапов выглядит так:

- делаем исходную копию базы опцией –b 0
- каждый день делаем инкремент уровня 1, -b 1
- каждый час делаем инкремент уровня 2, -b2

Инкременты уровня 2 «действуют» только между очередными инкрементами уровня 1, и т.д.
Можно было бы еще делать инкременты уровня 3, но это перебор.
Дальше, допустим, мы складываем все эти файлы, и получаем возможность восстановить базу на конкретный день и конкретный час путем склеивания соответствующих файлов уровней 0, 1 и 2.

Минусом такой схемы является то, что nbackup при всех этих операциях (и вообще) не проверяет содержимое страниц БД на целостность (gbak напротив, читает только данные). В результате при обнаружении повреждения в БД вам придется искать (и перебирать) инкременты до неповрежденного состояния. Поэтому лучше всего, если уж используете nbackup для инкрементов, не забывать периодически делать обычный бэкап-рестор при помощи gbak.

Дисковое пространство

Реальный пример: база 400 гигабайт, nbackup –b 0 делается раз в месяц, -b 1 каждый день. Инкремент уровня 1 имеет размер около 25 гигабайт. Для хранения инкрементов за месяц нужно 400гб (-b 0) плюс 30*25 = 750гб. Итого в месяц 1125гб.

Другой пример: база 10 гигабайт, инкрементов 4 уровня - раз в месяц, раз в неделю, раз в день, и каждый час. –b 1 занимает 1гиг, уровень 2 – 800мб, уровень 3 – 300мб.
В сумме хранимый за месяц объем, с возможностью восстановления на каждый час каждого дня месяца – 136 гигабайт.

Репликация

Раньше встроенной репликации в Firebird не было. Поэтому люди делали свою, программную репликацию, на триггерах. Такая репликация могла копировать данные как всей БД, так и части таблиц, не все столбцы, и т.д.
Тут известная проблема, что для реализации такой репликации нужны специальные таблицы, которые в основной бизнес-логике не участвуют, но замедляют ее дополнительными операциями вставки и удаления.

Репликация, основанная на передаче сегментов логических (по записям) изменений мастер-БД в реплику появилась в HQBird, для Firebird версий 2.5 и 3.0. В штатном Firebird эта же репликация появилась в версии 4.0 (и есть в 5.0, разумеется).

Смысл в следующем – мастер-сервер накапливает сегменты репликации с изменениями в БД, и затем эти сегменты каким-то способом передаются на сервер реплики. Реплика их принимает в БД реплики, и в результате все действия, которые произошли на мастере, применяются и на реплике.
В случае синхронной репликации отставание реплики от мастера составляет несколько секунд, в случае асинхронной – несколько минут.
Синхронная репликация неудобна тем, что для неё требуется постоянная связь серверов реплики и мастера, иначе при сбое связи реплику нужно будет переинициализировать (т.е. по факту останавливать мастер, копировать БД на реплику и опять стартовать оба сервера, сначала реплику, потом мастер).
Асинхронная реплика выгодна тем, что при обрыве связи между серверами просто на мастере будут накапливаться сегменты, которые отправятся на реплику при восстановлении связи.

Итого, в случае сбоя мастера восстановить работу можно за несколько минут, просто переключившись на БД реплики. Не нужны никакие бэкапы-ресторы, и т.д.
Кстати, это еще и возможность масштабирования – бэкапы можно делать на реплике, а не на мастере, и также можно направлять на реплику (которая для приложений работает в режиме «только чтение») разнообразные отчетные приложения, которые не будут грузить этой работой мастер-базу.

! на текущий момент убедиться, что реплика не сильно отстает от мастера можно либо проверкой количества накопившихся на мастере сегментов, или не примененных репликой сегментов. Еще проще – проверкой разницы в генераторах (реплика вполне доступна для чтения). Но служба такой проверки должна иметь возможность коннекта одновременно и к мастеру, и к реплике.

Дисковое пространство

Реплика занимает столько же места, сколько и оригинальная база. Однако, если по какой-то причине сервер реплики станет недоступен на неопределенное время, то на мастере начнут накапливаться сегменты репликации, которые нужно будет передать на реплику. И их объем может быть крайне большим, в зависимости от количества изменений, которые производятся в БД. Это могут быть сотни гигабайт в сутки.
Предсказать конкретные объемы невозможно, нужно просто попробовать – включить репликацию на мастере, не передавать сегменты на реплику, и посмотреть, сколько по размеру накопится сегментов.

Кстати, бэкапы gbak-ом можно вместо мастера делать на реплике, если отставание и правда не больше 3-5 минут. Поэтому, для дискового пространства реплики плюсуйте еще и размеры бэкапов gbak-ом или nbackup (для nbackup есть специфика, т.к. реплика находится в режиме read-only, что не дает изменять информацию об инкрементах в БД).

Репликация через nbackup

Начиная с версии Firebird 4.0 у nbackup появился интересный функционал, когда можно осуществить репликацию (физическую, страницами данных а не записями) при помощи nbackup

1. исходно делаем полную копию базы данных, nbackup –b 0 …
2. копируем эту копию на другой компьютер (это будет реплика)

3. на мастере делаем nbackup –b с параметром GUID базы вместо номера инкремента.
4. полученный инкремент переносим на реплику и применяем к копии БД
5. повторяем пункты 3 и 4.

В данном случае вы сами выбираете, как часто вы будете делать пункт 3, в зависимости от того, сколько он времени занимает, плюс время копирования файла на реплику, плюс применение его на реплике.
Прочитайте статью, стр. 19, там этот механизм описан подробно. Недостаток – нет «истории» инкрементов, кроме того, nbackup не проверяет целостность данных на страницах, таким образом можно при повреждении БД получить и поврежденный инкремент и копию БД.
(А значит нужно регулярно делать backup/restore штатным gbak).

Размер инкремента, как уже говорилось выше, будет зависеть от количества измененных страниц с момента предыдущего инкремента (частота изменения одних и тех же страниц в промежутке между инкрементами не влияет).

Копирование виртуальных машин и дисков

Можно вернуться к самому первому блоку, про «Файловое копирование». Только тут копируется не база, а целиком ОС или конкретный диск. Причем, с тем же результатом.
Конечно, внутри ОС в этот момент включается, например, VSS (на Windows) или что-то вроде, но внутри виртуальной машины база данных будет находиться в состоянии «как после reset».
То есть, с высокой вероятностью базу надо будет проверять утилитой gfix на наличие «повреждений».

Кстати, VSS и прочие технологии для копирования виртуальных машин – это практически то же самое, что делает nbackup. Только nbackup умеет корректно переводить БД в состояние «блокировки», а все эти внешние технологии – нет.

В лучшем случае, можно было бы делать так:

- блокируем базу nbackup –L
- делаем бэкап виртуалки
- разблокируем базу nbackup –N

В этом случае даже при всяких ресетах или ошибках база будет целой.

Но заменять gbak и nbackup копированием виртуальных машин нельзя.

Про требуемое дисковое пространство в этой части я говорить ничего не буду, т.к. это копирование не просто БД, а всей ВМ, и здесь есть свои инкременты, и т.д.

Ужасные схемы резервного копирования

12 типичных ошибок при бэкапе баз данных
https://habr.com/ru/articles/267881/

Если не хотите читать ссылку выше, то, например:

- Бэкап делается на тот же диск, где и база данных. Диск портится, нет ни базы, ни бэкапов.

- Регулярно делается бэкап-рестор, с перезаписью оригинальной базы данных. Место на диске кончается, бэкап завершается с ошибкой по нехватке места (и не проверяется), рестор из половины бэкапа затирает оригинальную базу – в результате нет ни полноценного бэкапа, ни базы

И т.д. Всё-таки, прочитайте статью выше про системные ошибки организации бэкапов.

Вопросы? support@ibase.ru

p.s. спасибо всем, кто помог с правками – Павел Зотов, Василий Сидоров, Денис Симонов !