СИЛЬНЫЕ СТОРОНЫ MySQL ДЛЯ ВЫСОКОНАГРУЖЕННЫХ ПРОЕКТОВ

Дмитрий ДРЕГЛЯ

Технический Университет Молдовы, Департамент Программной Инженерии и Автоматики

Аннотация: *Статья посвящена системе управления базами данных MySQL. Описаны ее сильные стороны в контексте сравнения с PostgreSQL. Рассмотрены механизмы репликации и компрессии данных.*

Ключевые слова*: база данных, СУБД, MySQL, PostgreSQL, репликация, компрессия данных.*

Введение

Базы данных - это специально разработанное хранилище для различных типов данных. Каждая база данных имеет определённую модель (реляционная, документно-ориентированная), которая обеспечивает удобный доступ к данным. Системы управления базами данных (СУБД) – это специальные приложения (или библиотеки) для управления базами данных различных размеров и форм. СУБД – это, наверное, самый сложный класс программ.

В статье пойдет речь о двух СУБД проектах - *MySQL и PostgreSQL,* с более чем 20-ти летней историей, с огромным набором функционала, и в таком случае детали - это главная проблема. Ни в один доклад эта информация не влезет, однако мы постараемся построить общую картину.

Речь пойдет о камне преткновения при сравнении этих двух СУБД, а именно о репликации, а также о компрессии данных.

1. Репликация

Репликация – это основа любых нагруженных проектов. Для таких проектов очень важна высокая доступность данных, а также их сохранность. Репликация позволяет разгрузить один сервер и распределять запросы между множеством серверов. Это требуется по ряду причин:

- Если мастер сервер становится недоступен, то мы переключаем клиентов на резервные сервера;
- Географическая распределёность клиентов – если у нас большой, масштабный проект, и клиент находится в США, а дата-центр в Азии , то работать им будет не очень комфортно просто в силу законов физики.

Существует репликация двух видов: физическая и логическая.

Физическая – самый простой вид репликации, она работает на низком уровне и передает на реплику точную побайтовую копию всех данных. Физическая репликация отсутствует в MySQL, но встроена в PostgreSQL с 2010 года.

Логическая – передает только изменение данных. К примеру, можно передать SQL запрос на slave-сервер и там проигрывать их заново. В PostgreSQL все аналоги решений не покрывают функционал логической репликации в MySQL.

У логической репликации следующие плюсы:

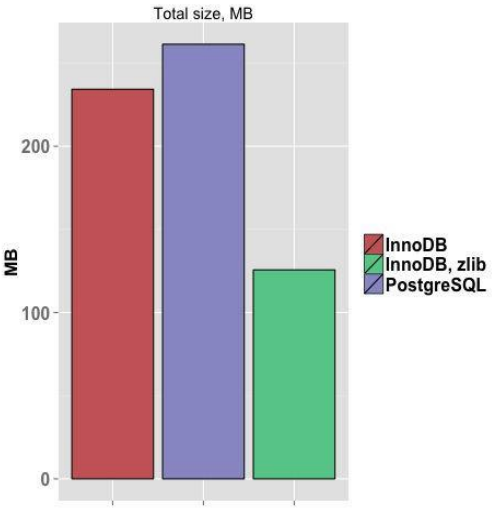
- Не создает точную побайтовую копию данных, следственно если мы на реплике перестроим таблицу, перестроим данные и индексы, то она продолжит работать, физическая нет;
- Позволяет иметь различную структуру данных на мастере и на реплике. Можно, например, добавить колонку, назначить для нее дефолтное значение и логическая репликация продолжит работать, несмотря на то, что на мастере этой колонки ещё нет. Это свойство как раз используется в высоконагруженных проектах, когда нужно изменить схему в большом распределённом кластере, сначала мы делаем каскадный апгрейд. Изменяем схемы на каждой реплике, переключаем клиентов на одну из реплик и старый мастер тоже апгрейдим;
- При логической репликации нет ограничения на чтение, с физической репликацией есть некоторые ограничения, связанные с работой механизма мульти-версионности.
- Можно делать частичную репликацию, то есть, передавать на slave не весь набор данных, а только его часть, это также используется в высоконагруженных проектах;
- Компактность – обычно набор данных передаваемых по сети намного меньше чем при физической репликации. Пример с индексами, если у нас есть много индексов в таблице, и мы изменяем какую-то строчку, то при физической репликации на реплику пойдет изменение не только данных, но и всех индексов, а при логической только данных.

2. Компрессия данных

Компрессия данных тоже достаточно важная функциональность для высоко нагруженных проектов. Любой проект, который работает с каким-то не тривиальным объемом данных заинтересован в том, чтобы размер этих данных на диске уменьшить. Для интернет гигантов вроде Facebook – это вопрос выживаемости.

InnoDB – это одна из подсистем низкого уровня для СУБД MySQL. InnoDB поддерживает постраничную компрессию данных, т.е. компрессируются как данные, так и индексы, и результаты компрессии и декомпрессии кэшируются, то есть в памяти хранятся как компрессированные, так и декомпрессированные образы страниц.

В PostgreSQL компрессии нет как таковой, есть нечто под названием TOAST, но компрессируются только отдельные записи. причем не все, а только те, которые длиннее 2 КБ.



Компрессия работает только для полей переменной длины.

Компрессируются только данные, индексы не компрессируются и результат компрессии и декомпрессии никак не кэшируется. То есть, если мы будем тысячу раз читать одну и ту же запись, и она скомпрессирована, то PosgtreSQL будет тысячу раз её декомпрессировать.

К примеру, была создана некая таблица, туда было загружено миллион записей и красным мы видим объем данных InnoDB без компрессии, синий это PostgreSQL, а зеленый это InnoDB с компрессией.

Для этой таблицы TOAST компрессия даже не включилась, т.к. средний размер записи был около 180 байт, а PostgreSQL требует 2 КБ, и то, для того чтобы решить компрессировать её или нет.

Заключение

Есть много причин использовать MySQL в высоко нагруженных проектах. Некоторые важные для крупных проектов возможности отсутствуют в PostgreSQL и не реализованы до сих пор. Однако, это не значит, что MySQL должна использоваться абсолютно везде.

Выбор между данными СУБД сложный вопрос, если кто-то вам предлагает простой ответ на этот вопрос, то этот человек либо не объективен, либо не компетентен.

Библиография

1. Алексей Копытов. Сильные стороны MySQL, OSSDEVCONF-2016. – [Электронный ресурс]. – Режим доступа: http://catcut.net/zNzA
2. Сергей Яковлев. MySQL и PostgreSQL. Сравнительный анализ. – [Электронный ресурс]. – Режим доступа: https://www.ibm.com/developerworks/ru/library/os-mysql-postgresql/01/index.html
3. Devacademy.ru. SQLite vs MySQL vs PostgreSQL: сравнение систем управления базами данных. – [Электронный ресурс]. – Режим доступа: http://devacademy.ru/posts/sqlite-vs-mysql-vs-postgresql/