Arenadata™ Grid

Версия - v2.3.1-nsud

Основные концепции Arenadata Grid

2020 Arenadata™

Оглавление

1 Сервер приложений + СУБД **3**

2 Возможности СУБД **4**

3 Служебные компоненты **6**

В документации приведены основные факты об **Arenadata Grid** (**ADG**).

Глава 1

Сервер приложений + СУБД

В основе **Arenadata Grid** лежит ПО **Tarantool**. **Tarantool** -– это open-source проект. Исходный код открыт для всех и распространяется бесплатно согласно лицензии **BSD license**. Поддерживаемые платформы: **GNU** / **Linux**, **Mac OS** и **FreeBSD**.

Tarantool представляет собой сервер приложений на языке **Lua**, интегрированный с СУБД. В основе **Tarantool** лежат файберы (fibers), что означает, что несколько Tarantool-приложений могут работать в одном потоке (thread), при этом каждый экземпляр Tarantool-сервера может одновременно запускать несколько потоков для обработки ввода-вывода данных и фоновых задач. **Tarantool** включает в себя **LuaJIT** (Just In Time) – Lua-компилятор, Lua-библиотеки для наиболее распространенных приложений, а также сервер базы данных **Tarantool**, который представляет собой широко признанную СУБД NoSQL. Таким образом, **Tarantool** используется для всех тех целей, которые принесли популярность **node.js** и **Twisted**, и более того – поддерживает персистентность данных.

Пользователи **Tarantool** могут создавать, изменять и удалять Lua-функции прямо во время исполнения кода. Также они могут указывать Lua-программы, которые будут загружаться во время запуска **Tarantool**. Такие программы могут служить триггерами, выполнять фоновые задачи и взаимодействовать с другими узлами по сети. В отличие от многих популярных сред разработки приложений, которые используют “реактивный” принцип, сетевое взаимодействие в **Lua** устроено последовательно, но очень эффективно, так как оно использует среду кооперативной многозадачности самого **Tarantool**.

Один из встраиваемых Lua-пакетов – это API для функций СУБД. Таким образом, некоторые разработчики рассматривают **Tarantool** как СУБД с популярным языком для написания хранимых процедур, другие рассматривают его как Lua-интерпретатор, а третьи – как вариант замены сразу нескольких компонентов в многозвенных веб-приложениях. Производительность **Tarantool** может достигать сотен тысяч транзакций в секунду на ноутбуке, и ее можно наращивать “вверх” или “вширь” за счет новых серверных ферм.

Глава 2

Возможности СУБД

Компонент “box” – серверная часть с функциями СУБД – это важная часть **Tarantool**, хотя он может работать и без данного компонента.

API для функций СУБД позволяет хранить Lua-объекты, управлять коллекциями объектов, создавать и удалять вторичные ключи, делать атомарные изменения, конфигурировать и мониторить репликацию, производить контролируемое переключение при отказе (failover), а также исполнять код на **Lua**, который вызывается событиями в базе. А для прозрачного доступа к удаленным (remote) экземплярам баз данных разработан API для вызова удаленных процедур.

В архитектуре серверной части СУБД **Tarantool** реализована концепция “движков” базы данных (storage engines), где в разных ситуациях используются разные наборы алгоритмов и структуры данных. В **Tarantool** есть два встроенных движка: in-memory движок, который держит все данные и индексы в оперативной памяти, и двухуровневый движок для B-деревьев, который обрабатывает данные размером в *10-1000* раз больше того, что может поместиться в оперативной памяти. Все движки в **Tarantool** поддерживают транзакции и репликацию, поскольку они используют единый механизм упреждающей записи (WAL = write ahead log). Этот механизм обеспечивает согласованность и сохранность данных при сбоях. Таким образом, изменения не считаются завершенными, пока не проходит запись в лог WAL. Подсистема записи в журнал также поддерживает групповые коммиты.

In-memory движок базы данных **Tarantool** (*memtx*) хранит все данные в оперативной памяти, поэтому у него низкое значение задержки чтения. Кроме того, когда пользователи запрашивают снимки данных (snapshots), этот движок создает персистентные копии данных в энергонезависимой памяти, например на диске. Если экземпляр сервера прекращает работать и данные в оперативной памяти теряются, то при следующем запуске сервер загружает в память самый свежий снимок и воспроизводит все транзакции из журнала. Таким образом, данные не теряются.

В штатных ситуациях in-memory движок работает без блокировок. Вместо многопоточных примитивов, которые предлагает операционная система (таких как *mutex*), **Tarantool** использует кооперативную многозадачность для работы с тысячами соединений одновременно. В **Tarantool** есть фиксированное количество независимых потоков управления (thread), и у них нет общего состояния. Для обмена данными между потоками используются очереди сообщений с малой перегрузкой. Хотя такой подход накладывает ограничение на количество процессорных ядер, которые может использовать экземпляр, в то же время он позволяет избежать борьбы за шину памяти, а также дает запас масштабируемости по скорости доступа к памяти и производительности сети. В результате даже при большой нагрузке экземпляр **Tarantool** в среднем использует процессор менее чем на *10%*. Кроме того, **Tarantool** поддерживает поиск как по первичным, так и по внешним ключам в индексах.

Дисковый движок базы данных **Tarantool** (*vinyl*) совмещает в себе подходы, заимствованные из современных файловых систем, журнально-структурированных деревьев со слиянием (log-structured merge trees) и классических B-деревьев. Все данные разбиты на диапазоны. Каждый диапазон представлен файлом на

диске. Размер диапазона можно изменять, обычно он равен *64МБ*. Каждый диапазон – это набор страниц, которые служат разным целям. После полного слияния диапазона ключи на его страницах не пересекаются. Если диапазоны ключей недавно сильно изменялись, можно провести частичное слияние диапазона. В этом случае на некоторых страницах появляются новые ключи и значения. Дисковый движок обновляет данные по принципу дописывания в конец: новые данные никогда не затирают старые.

Tarantool поддерживает работу с составными ключами в индексах. Возможные типы ключей: *HASH*, *TREE*, *BITSET* и *RTREE*.

Tarantool также поддерживает асинхронную репликацию – как локальную, так и на удаленных серверах. При этом репликацию можно настроить по принципу мастер-мастер, когда несколько узлов могут не только обрабатывать входящую нагрузку, но и получать данные от других узлов.

Глава 3

Служебные компоненты

Помимо **Tarantool** в **Arenadata Grid** входят:

- Службы мониторинга хостов и метрик экземпляров и приложений Tarantool для интеграции с *Arenadata Monitoring*;
- Служба сбора событий с экземпляров Tarantool для аггрегации и индексирования в кластере *ElasticSearch*;
- HTTP-балансировщик вызовов REST-API.