Использование базы данных Mnesia в чат-сервере

Юра Жлоба

Wargaming.net

Май 2019

Mnesia

Распределенная key value база данных,

встраиваемая в Erlang приложения.

Mnesia

Почему ее не рекомендуют использовать?

И почему все же используют?

Как она используется в чат-сервере?

Amnesia

1999 год

Изначально база данных называлась Amnesia.

Это название не понравилось кому-то из менеджмента.

”So we dropped the A, and the name stuck.” Joe Armstrong.

Amnesia

Традицию продолжила компания WhatsApp

Они назвали свою БД

ForgETS

Фичи

- Работает внутри эрланговской ноды
  Не нужно передавать данные по сети

Фичи

- Работает внутри эрланговской ноды
  Не нужно передавать данные по сети
- Хранит данные нативно (Erlang term)
  Не нужно сериализовать/десериализовать данные

Фичи

- Работает внутри эрланговской ноды
  Не нужно передавать данные по сети
- Хранит данные нативно (Erlang term)
  Не нужно сериализовать/десериализовать данные
- Хранит данные в ETS/DEST таблицах
  Чтение и запись работают очень быстро

Фичи

Работает внутри эрланговского кластера

Данные доступны отовсюду в кластере (сетевая прозрачность)

Полная реплика данных на каждой ноде

Фичи

- Транзакции (ACID)
- Вторичные индексы
- Миграции (структуры таблиц и данных)
- Шардинг (fragmented tables)

С точки зрения CAP теоремы

- Если с транзакциями, то CP
  И это медленно (очень)

С точки зрения CAP теоремы

- Если с транзакциями, то CP
  И это медленно (очень)
- Если в dirty режиме, то AP
  И тут никаких гарантий Consistency, даже ”eventualy”

С точки зрения CAP теоремы

- Если с транзакциями, то CP
  И это медленно (очень)
- Если в dirty режиме, то AP
  И тут никаких гарантий Consistency, даже ”eventualy”
- А если я хочу CA?
  Тогда просто бери ETS/DETS

API

- Базовые KV операции
  read, write, delete

API

- Базовые KV операции
  read, write, delete
- ETS/DETS API
  lookup, match, select

API

- Базовые KV операции
  read, write, delete
- ETS/DETS API
  lookup, match, select
- Fold
  foldl, foldr

API

- Базовые KV операции
  read, write, delete
- ETS/DETS API
  lookup, match, select
- Fold
  foldl, foldr
- QLC
  Query List Comprehension

Query List Comprehension

```
qlc:q([X || X <- mnesia:table(shop)])
```

```
qlc:q([
    Xshop.item || X <- mnesia:table(shop),
    Xshop.quantity < 250
])
```

```
qlc:q([
    Xshop.item ||
    X <- mnesia:table(shop),
    Xshop.quantity < 250,
    Y <- mnesia:table(cost),
    Xshop.item =:= Ycost.name,
    Ycost.price < 2
])
```

Транзакции

Синхронные и ”обыкновенные”

Pessimistic locking

Медленные

Но без них нет консистентности данных

Репутация Mnesia

Мнение широко известных в узких кругах авторитетов

Печальный опыт с персистентными очередями в RabbitMQ

Слухи из Стокгольма от местных разработчиков

Репутация Mnesia

Суть проблемы в том,

что если нода не была корректно остановлена, а упала,

то восстановление большой таблицы с диска может занять часы.

Репутация Mnesia

Downtime сервиса может длится несколько часов!

На этом про Mnesia можно было бы забыть и не вспоминать

но ...

Применение Mnesia

но её можно применить с пользой

Задача

Кластер из нескольких эрланг-нод.

Нужно хранить пользовательские сессии,

так, чтобы они были доступны во всех нодах кластера.

Задача

Прежнее решение:

Сессии хранятся в MySQL

Задача

- Конечно, хочется иметь эту инфу прямо в ноде.

Задача

- Конечно, хочется иметь эту инфу прямо в ноде.
- Кешировать в ETS?

Задача

- Конечно, хочется иметь эту инфу прямо в ноде.
- Кешировать в ETS?
- Хорошо, а как обновить этот кэш на всех нодах?

Задача

- Конечно, хочется иметь эту инфу прямо в ноде.
- Кешировать в ETS?
- Хорошо, а как обновить этот кэш на всех нодах?
- Вот если бы был распределенный кэш ...

Задача

- Конечно, хочется иметь эту инфу прямо в ноде.
- Кешировать в ETS?
- Хорошо, а как обновить этот кэш на всех нодах?
- Вот если бы был распределенный кэш ...
- Постойте-ка, а Mnesia – это что?

Применение Mnesia

Mnesia не вызывает проблем, если:

- не нужно персистентное хранение данных
- не нужны сложные запросы с транзакциями
- данные относительно дешево реплицируются

Применение Mnesia

Mnesia не стоит использовать, если:

- Нужно хранить много данных
- Нужно хранить их персистентно
- Объем данных постоянно растет
- Выполняются сложные запросы к данным

Применение Mnesia

Все это – типичные сценарии использования типичной БД

И все это – плохо для Mnesia

Применение Mnesia

in-memory хранение пользовательских сессий

идеальный сценарий для Mnesia

Применение Mnesia

Mnesia – это не БД, это кэш :)

Применение Mnesia

Еще раз про ключевые преимущества:

- Данные прямо в памяти ноды,
  за ними не надо ходить по сети
- Данные в нативном виде,
  их не надо сериализовать/десериализовать
- Прозрачная репликация на все ноды кластера

Применение Mnesia

Что важно для нас:

- Mnesia неплохо переживает рестарты отдельных нод в кластере
- Потому что мы именно так обновляем кластер
- Но нужно знать объем данных и время их репликации
- Это этого зависит время downtime ноды при рестарте

Вопросы?