Skillbox

Управление транзакциями

Андрей Гордиенков

Solution Architect

ABAX

В прошлом и в этом модуле

В прошлом модуле:

- правильное проектирование монолита
- разделения баз данных и обмена данными между ними
- транзакции в базе данных

В этом модуле:

- транзакции в базе данных и не только
- распределённые транзакции
- уровни согласованности данных
- аномалии в данных и что с ними делать

Skillbox

Транзакции

Андрей Гордиенков

Solution Architect

ABAX

В этом уроке

- Подробно поговорим про транзакции и распределённые транзакции.
- Поймём, в чём сложность и опасность распределённых транзакций.

Определение

Транзакция — группа последовательных операций, которая представляет собой логическую единицу работы с данными. Транзакция может быть выполнена либо целиком и успешно, соблюдая целостность данных и независимо от параллельно идущих других транзакций, либо не выполнена вообще, и тогда она не должна произвести никакого эффекта.

Распределённые транзакции подразумевают использование более чем одной транзакционной системы и требуют намного более сложной логики.

Почему?

- База данных или её серверы могут упасть в любое время
- Приложение может упасть в любое время
- Могут возникнуть неполадки в сети между приложением и базой или её репликами
- Несколько клиентов пишут в базу одновременно, переписывая значения друг друга
- Приложение читает частично записанные данные, которые не несут смысла в таком виде
- Клиенты могут оказаться в условиях «гонки», что может привести к непредсказуемым последствиям

Для чего?

- Приемлемое решение указанных проблем
- Упрощение модели разработки:
    - работа с ошибками делегируется работы базе данных
    - приложение не заботится о целостности данных

Идеальное решение

Не использовать транзакции, если возможно

Целостность данных
Согласованность данных

Скорость
или доступность

Немного истории

Транзакции были предложены в 1975 году IBM для System R, и с тех пор фактически принцип не изменился.

В 2000-х появились NoSQL-хранилища и сформировалось мнение, что транзакции — это то, что ограничивает масштабирование.

Некоторые производители до сих пор позиционируют свои решения с транзакциями как единственный и верный подход для «серьёзных приложений» с «ценной информацией».

Немного истории

Транзакции были предложены в 1975 году IBM для System R,
и с тех пор фактически, принцип не изменился.

В 2000-х появились NoSQL хранилища и появилось мнение,
что транзакции — это то, что ограничивает масштабирование.

Некоторые производители до сих пор позиционируют свои решения
с транзакциями как единственный и верный подход для «серьёзных
приложений» с «ценной информацией».

НЕПРАВДА

Транзакции, как каждое архитектурное решение, имеют свои плюсы и ограничения.

Skillbox

ACID

Характеристики были определены в 1983 году Тео Йёрдером (Theo Härder) и Андреасом Рейтером (Andreas Reuter) для обозначения систем, устойчивым к сбоям.

- Atomicity (Атомарность)
- Consistency (Согласованность)
- Isolation (Изолированность)
- Durability (Прочность)

Но не каждая база одинаково трактует эти атрибуты качества.

Атомарность

- Атомарность — неразделимость чего-либо. Для баз данных это означает, что изменения применяются полностью или не применяется ничего.
- Однозначность результата изменений.
- Возможно, лучшим названием для этого свойства было бы «прерываемость».

Согласованность

Consistency может означать:

- одинаковость данных
- механизм вычисления кеша данных для партицирования
- линеаризуемость в CAP-теореме
- согласованность реляционных данных
- база в хорошем состоянии

В общем случае это означает, что некоторое утверждение, записанное в базе данных, будет верно независимо от того, как мы его вычисляем.

Однако не все базы данных могут это поддерживать и гарантировать по умолчанию.

В строгом смысле это ответственность приложения.

Изолированность

Операции разных пользователей не влияют друг на друга.

Не проблема, если субъекты разные, но если один и тот же?

Большинство БД подразумевают под этим сериализуемость данных, то есть только одна транзакция действует в один момент времени.

Проблема:

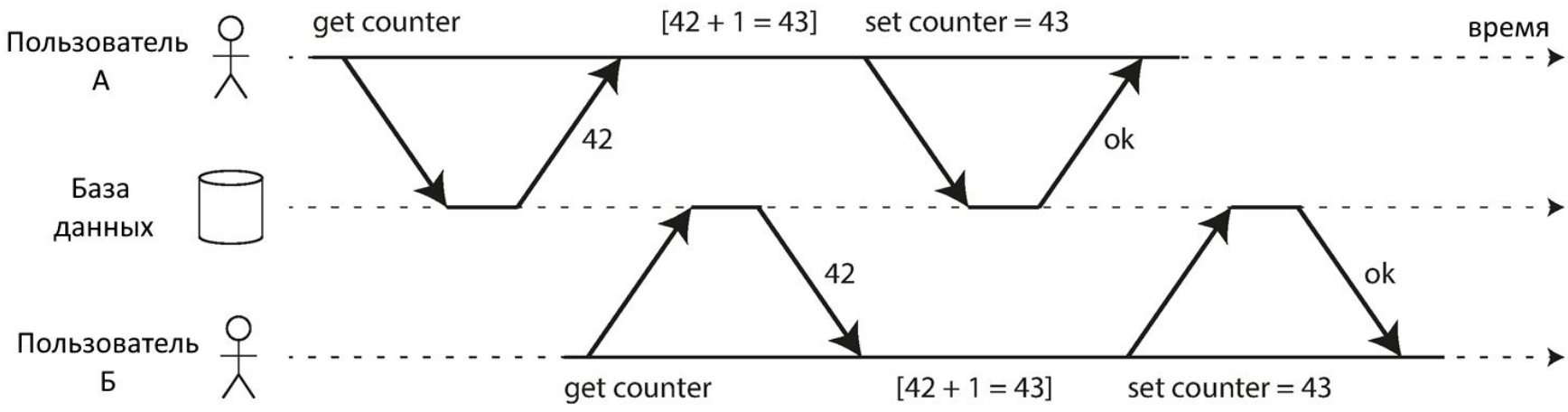
Изолированность

Операции разных пользователей не влияют друг на друга.

Не проблема, если субъекты разные, но если один и тот же?

Большинство БД подразумевают под этим сериализуемость данных, то есть только одна транзакция действует в один момент времени.

Проблема:



Изолированность

Сериализация на практике не используется, потому имеет большой штраф к производительности.

Многие БД даже не поддерживают сериализуемость.

Прочность

Прочность — гарантия от БД, что записанные данные не потеряются в результате какого-либо сбоя.

Не стоит забывать, что БД может гарантировать одно, а жёсткие диски и прочие компоненты сервера могут иметь сбои, независимо от желания БД.

Для реплицированных БД это означает, что изменения были записаны на достаточное количество реплик.

Особенности

Для единичных субъектов все БД предоставляют свойства **атомарности** и **изолированности**, остальное не гарантировано.

Skillbox

Теорема CAP

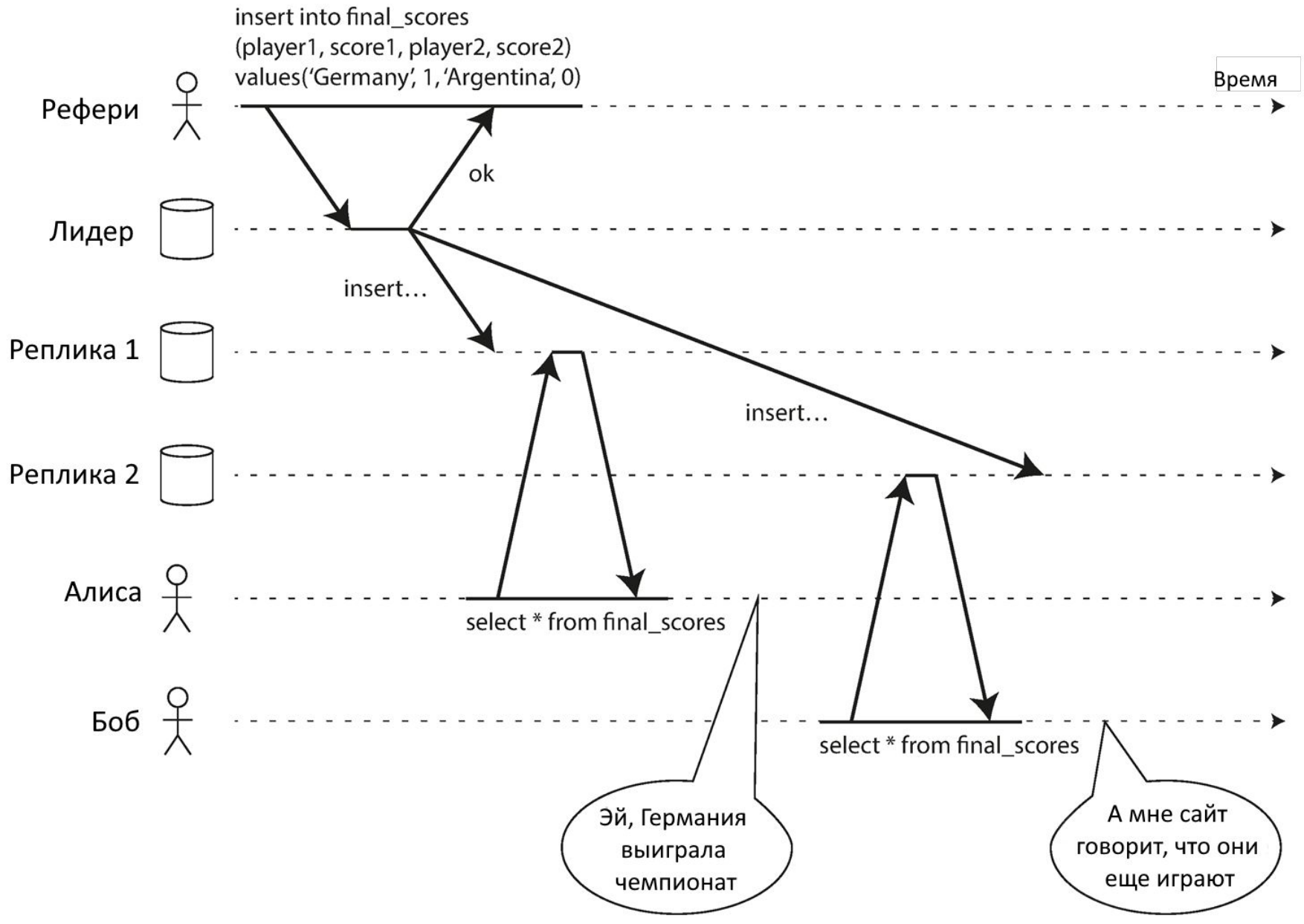
Автор — Эрик Брюер (Eric Brewer).

Джефф Ходжес (Jeff Hodges) упомянул о связи распределённых систем и CAP-теоремы в статье «Заметки о распределённых системах для новичков».

Теорема CAP

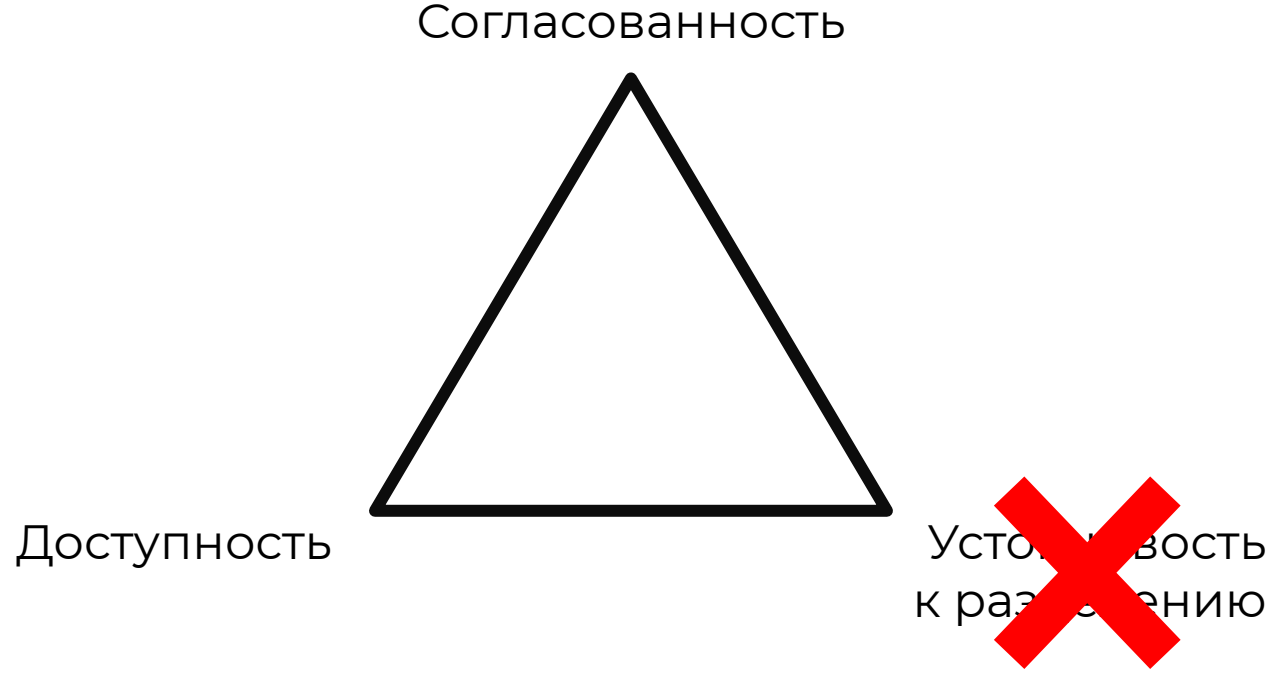
- Согласованность (Consistency) в CAP — линеаризуемость, является принципом согласованности (и очень сильным).
  Не имеет ничего общего с «C» из ACID, даже если эта «C» также означает «согласованность».
- Доступность (Availability) — в CAP определена как «каждый запрос, полученный работающим узлом [базой данных], в системе должен приводить к ответу [не содержащему ошибок]». Недостаточно, чтобы некоторые узлы могли обработать запрос — **любой** работающий узел должен быть способен обработать запрос.
- Устойчивость к разделению (Partition tolerance) — ужасное название, в общих словах означает, что для связи вы используете асинхронную сеть, которая может терять или задерживать сообщения. Интернет и все дата-центры обладают этим свойством, так что в реальности у вас нет выбора в этом контексте.

Линеаризуемость



Skillbox

Теорема CAP



Skillbox

Теорема CAP

И даже теперь теорема всё равно не применима к распределённым системам и базам данных.

BASE

Определение было дано Эриком Брюером (Eric Brewer), автором теоремы CAP.

- **B**asically **A**vailable
- **S**oft state
- **E**ventual consistency

Ещё более расплывчатое определение, нежели ACID, обычно используется для NoSQL БД.

Свойства

- **Basic Availability**
  Большинство реплик БД доступно для работы.
- **Soft state**
  БД не обязана быть согласованной относительно операций записи. Реплики не обязаны быть согласованными относительно друг друга в плане имеющихся данных.
- **Eventual consistency**
  Данные не согласованы и могут приходить в согласованное (одинаковое) состояние «когда-нибудь».

В основном используется в хранилищах типа «ключ — значение» (Redis) или документоориентированных (Mongo).

Особенности

- Более высокая скорость операций
- Высокая доступность реплик
- Контроль целостности данных полностью в ответственности приложения

Skillbox

ACID vs BASE

ACID ориентирована на согласованность и целостность данных:

- В основном для финансовых учреждений и там, где целостность данных критична
- Контроль целостности данных — забота базы
- Стабильная модель данных

BASE ориентирована на доступность данных:

- Маркетинг, работа с клиентами — когда нужна согласованность в рамках одного агрегата, который будет записан, скорее всего, как один объект
- Контроль целостности — забота клиента
- Модель данных активно эволюционирует

Skillbox

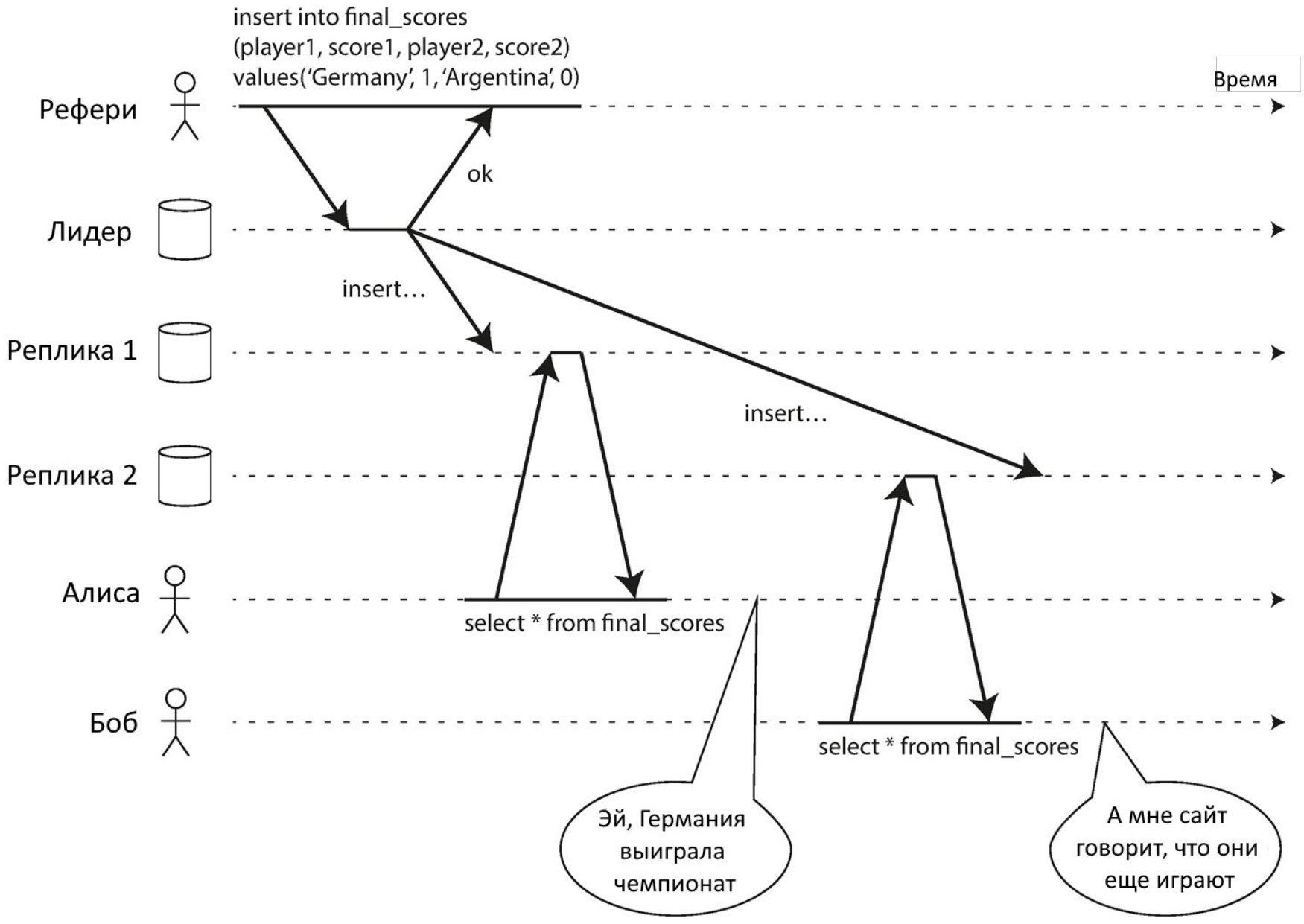
Распределённые транзакции

Распределённые транзакции подразумевают использование более чем одной транзакционной системы и требуют намного более сложной логики.

Два типа:

- **Внутренние** —происходят в рамках одной базы данных, участие реплик. Могут использовать проприетарные протоколы и оптимизации.
- **Гетерогенные** — участвуют базы от разных поставщиков или даже не базы данных, например брокеры сообщений.

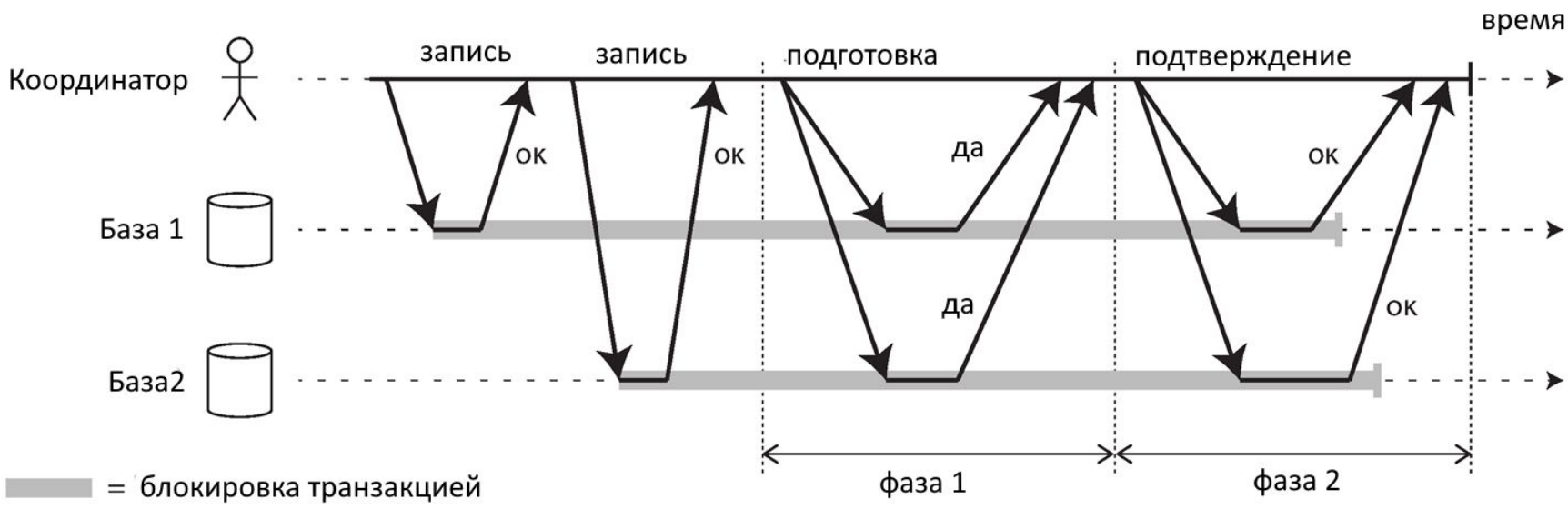
Пример


insert into final_scores
(player1, score1, player2, score2)
values('Germany', 1, 'Argentina', 0)
Время
Рефери
ok
Лидер
insert…
Реплика 1
insert…
Реплика 2
Алиса
select * from final_scores
Боб
select * from final_scores
Эй, Германия выиграла чемпионат
А мне сайт говорит, что они еще играют

2 Phase Commit (2PC)

Требуется координатор распределённых транзакций.

1. Базы/реплики делают запись как обычно, без подтверждения, но с блокировкой данных.
2. Координатор просит подтвердить, что участники готовы к подтверждению.
3. Подтверждение операции.



Ограничения

- Если координатор транзакций работает в единственном экземпляре, тогда это чувствительная точка отказа. Редко, когда такие координаторы реплицируются.
- Если координатор реализован в логике сервера приложений, тогда возникают сложности с репликацией и поддержкой лога координатора в согласованном состоянии.
- Невозможно в общем случае определить замыкания (deadlock) в базах для освобождения блокировок, а также обработать специализированные ошибки от разных БД.
- Для внутренних распределённых транзакций остаётся проблема скорости работы для 2PC, ответ должен прийти от всех реплик.

Skillbox

Распределённые транзакции на практике

Плюсы:

- гарантии безопасности выполнения транзакции (сложно реализовать)

Минусы:

- операционные проблемы (ошибки координатора)
- снижение производительности (иногда в 10 раз)

Выводы

- Транзакция — логическая структура, помогающая в управлении согласованностью данных.
- Распределённые транзакции имеют сильные штрафы и риски; использовать стоит, когда нет другого выхода и согласованность данных превыше скорости.
- CAP-теорема не применима к распределённым хранилищам.

Что дальше?

- Шаблон «Сага» для распределённых транзакций
- Компенсационные транзакции

Skillbox

Спасибо за внимание!