OTUS

Онлайн образование

otus.ru

Проверить, идет ли запись

Меня хорошо видно && слышно?

Тема вебинара

Теорема CAP

Коробков Виктор

Консультант команды технологического обеспечения
ООО «ИТ ИКС5 Технологии»

Telegram: @Korobkov_Viktor

Преподаватель

Виктор Коробков

более 20 лет в IT

специализация: проектирование баз данных (СУБД PostgreSQL, MS SQLServer)

В OTUS веду занятия на курсах: СУБД, PostgreSQL, SQL Server Developer, noSQL, MongoDB, Программист C

Правила вебинара

Активно
участвуем

Off-topic обсуждаем
в Telegram

Задаем вопрос
в чат или голосом

Вопросы вижу в чате,
могу ответить не сразу

Условные обозначения

Индивидуально

Время, необходимое
на активность

Пишем в чат

Говорим голосом

Документ

Ответьте себе или
задайте вопрос

Маршрут вебинара

- Транзакции, ACID
- Распределенные системы
- BASE vs ACID
- Теорема CAP
- Алгоритмы консенсуса
- Рефлексия

Цели вебинара

После занятия вы сможете

1. Понимать смысл теоремы CAP
2. Различать BASE и ACID требования к базам
3. Иметь представление об алгоритмах консенсуса

Транзакции, ACID

Транзакция - последовательность операций, рассматриваемая базой данной как атомарное действие и завершающаяся подтверждением изменений (commit) либо откатом изменений (rollback).

Обеспечивает:

- сохранение целостности данных;
- параллельную работу пользователей с базой данных;
- восстановление данных при откатах и сбоях.

ACID

Atomicity (Атомарность) - либо все изменения транзакции фиксируются (commit), либо все откатываются (rollback).
Consistency (Согласованность) - транзакции не нарушают согласованность данных, то есть они переводят базу данных из одного корректного состояния в другое.
Isolation (Изолированность) - работающие одновременно транзакции не влияют друг на друга.
Durability (Долговечность) - если транзакция была успешно завершена, то никакое внешнее событие не должно привести к потере совершенных ей изменений.

Способы реализации ACID

1. **ARIES** (Algorithms for Recovery and Isolation Exploiting Semantics) - алгоритмы восстановления систем:

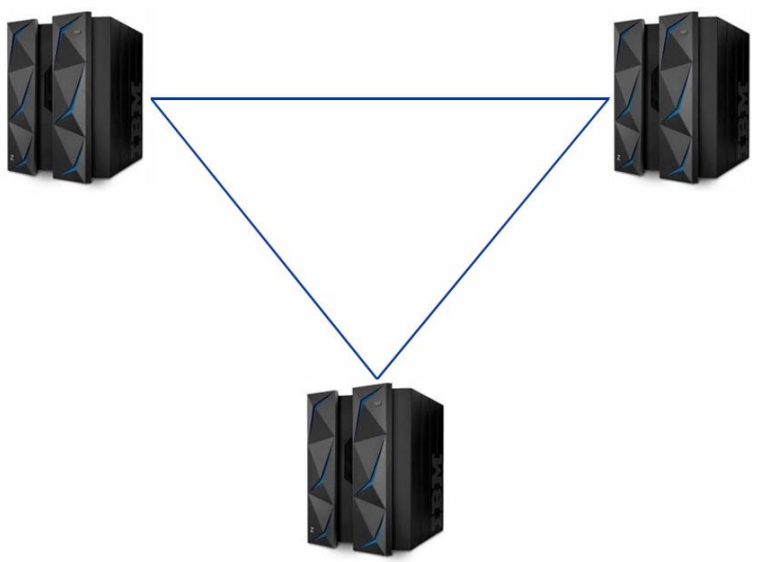
- logging - запись в журнал всех действий транзакции, которые могут изменить состояние БД;
- checkpoints - механизм контрольных точек;
- поддержка покортежных блокировок.

2. **MVCC** (MultiVersion Concurrency Control) - механизм обеспечения параллельного доступа к БД:

- каждой сессии предоставляется «снимок» БД;
- изменения в БД невидимы другим пользователям до момента фиксации транзакции.

Распределенные системы

Что такое распределенная система ?

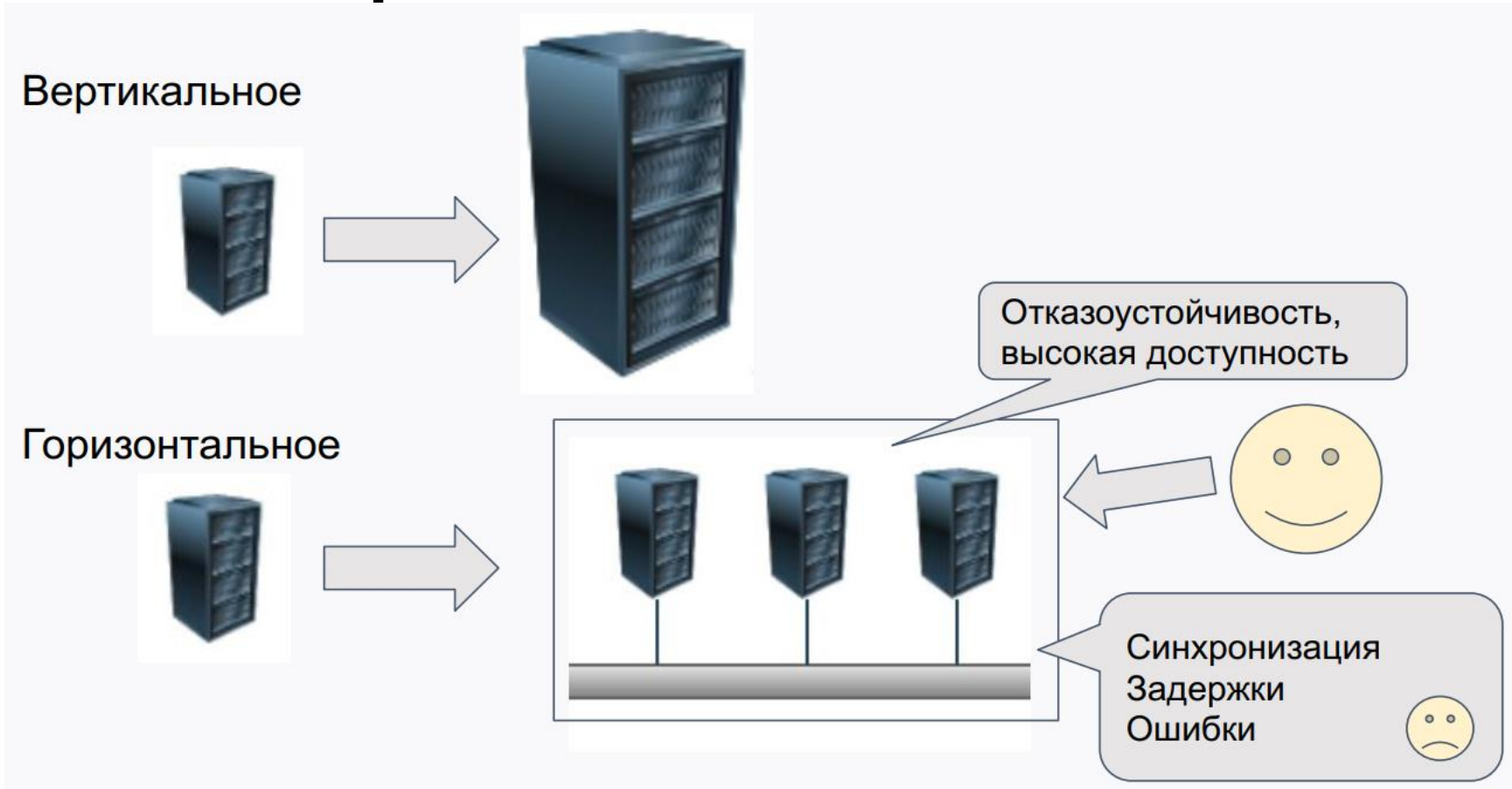


Распределенной вычислительной системой можно назвать такую систему, в которой отказ компьютера, о существовании которого вы даже не подозревали, может сделать ваш собственный компьютер непригодным к использованию.

Лесли Лампорт
первый лауреат премии Дейкстры
за достижения в области распределенных вычислений

Распределенная система — это набор компьютерных программ, использующих вычислительные ресурсы нескольких отдельных вычислительных узлов для достижения одной общей цели.

Масштабирование



8 заблуждений распределенных систем: 7 (1994 Питер Дейч) + 1 (1997 Джеймс Гослинг)

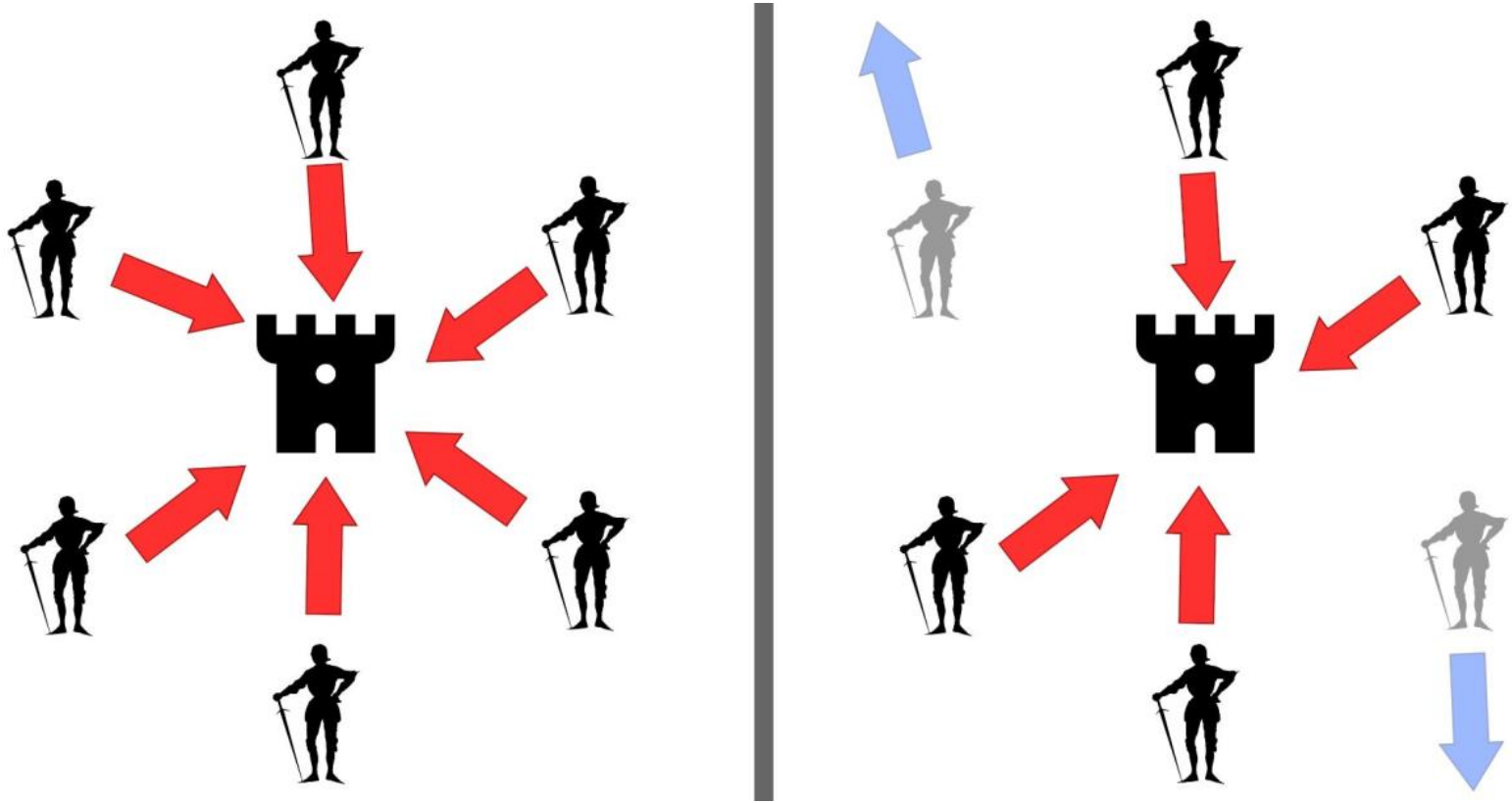
1. Сеть надежна.
2. Задержка равна нулю.
3. Пропускная способность бесконечна.
4. Сеть безопасна.
5. Топология не меняется.
6. Существует один администратор.
7. Цена передачи данных равна нулю.
8. Сеть однородна.

А еще существуют отказы узлов

Модели отказов в PBC:
• модель отказа «Остановка» - узел просто останавливается без предупреждения.
• византийская модель отказа - неисправные узлы не только останавливаются, но и могут вести себя неопределенным образом.

Византийские отказы предназначены для моделирования любого типа неисправностей. Термин «Византийский» был впервые использован в докладе Лампорта, Пиза и Шостака для такого типа сбоев, в котором в терминах византийских генералов формулируется проблема консенсуса.

Проблема византийских генералов



Чем плохо ACID в распределенных системах

Медленно – каждая транзакция применяется, только если все узлы добавили информацию о ней.

Дорого – дата центры должны быть связаны выделенным каналом.

Избыточно – такой уровень надежности не нужен, если у вас хранятся сообщения или фотографии соц сети.

BASE vs ACID

BASE вместо ACID

1. Basic Availability (базовая доступность) – каждый запрос гарантированно завершается.

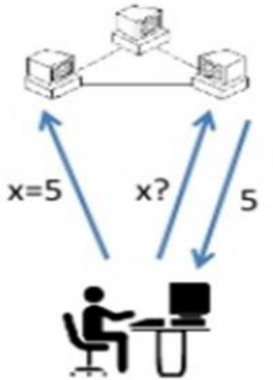
2. Soft state (гибкое состояние) – состояние системы может изменяться без ввода новых данных для достижения согласования данных.

3. Eventual consistency (согласованность в конечном счете) – данные некоторое время могут быть рассогласованные, но в итоге приходят к согласованию.

CAP теорема

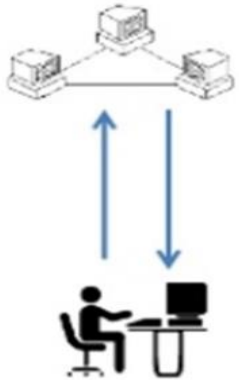
CAP теорема (Eric Brewer, 2000)

Consistency



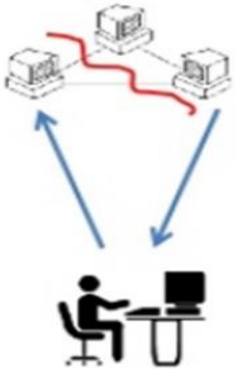
Согласованность – все рабочие узлы содержат одинаковую информацию.

Availability



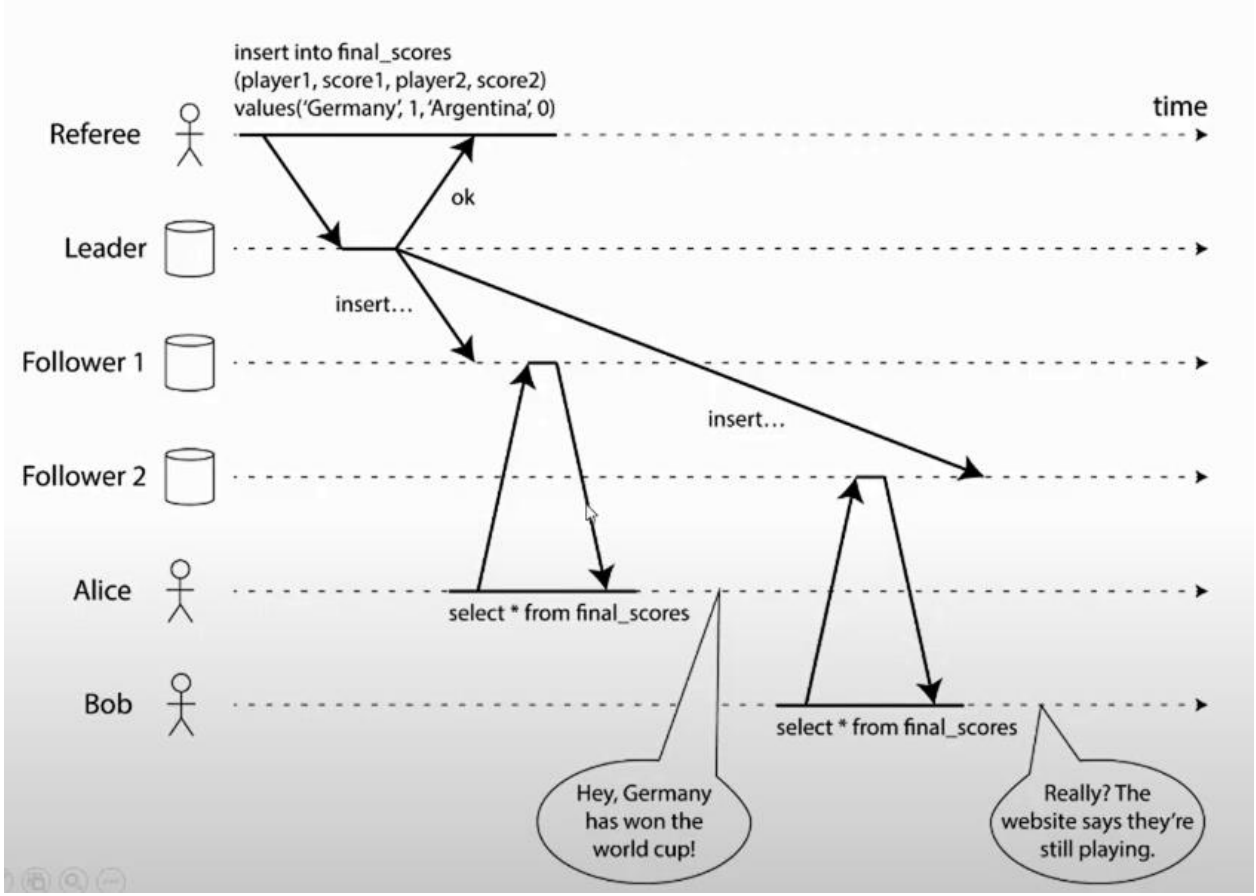
Доступность – возможность доступа к кластеру, даже если узел в кластере выходит из строя

Partition tolerance



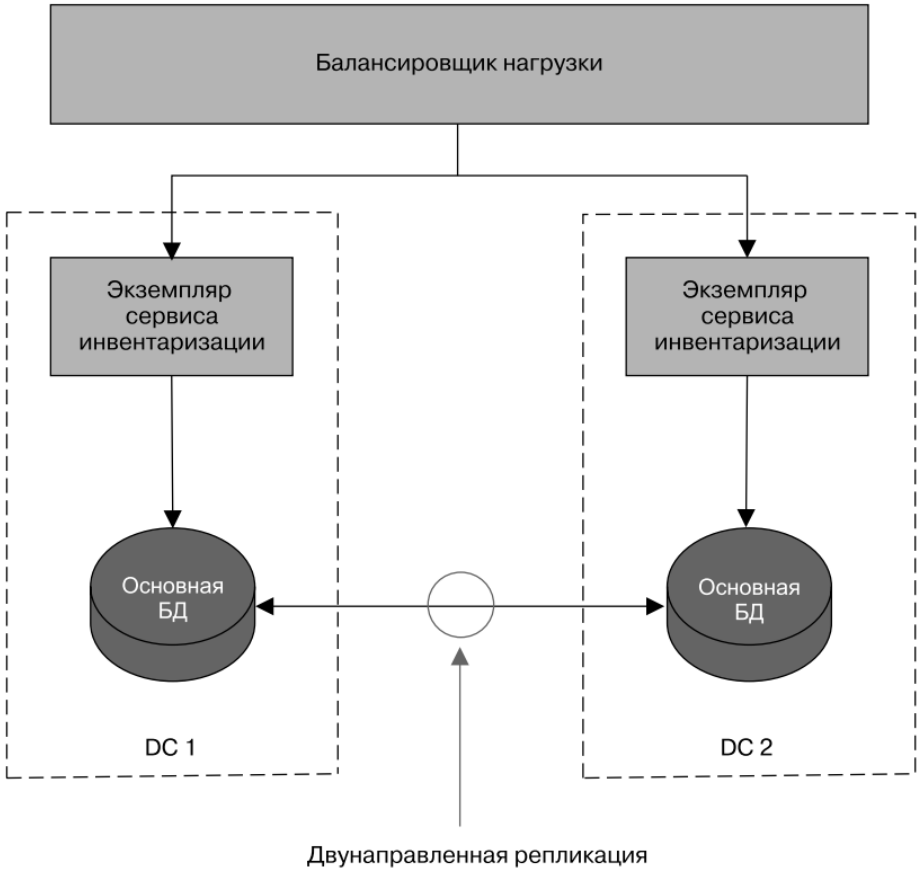
Терпимость к разделению сети – независимо от сбоев в работе сети узлы продолжают работать.

Кейс «Чемпионат по футболу»



Доступность есть,
Согласованности нет

Кейс «Инвентаризация»

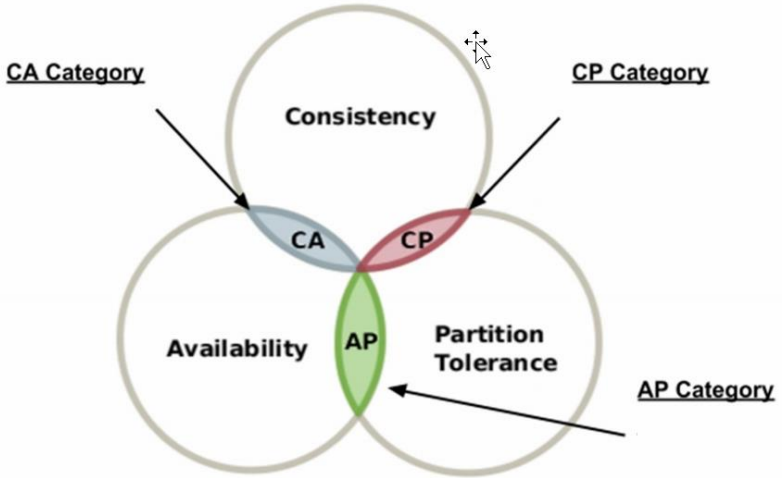


Availability
Доступность
Каждый пользователь всегда
может писать и читать

Consistency
Согласованность
Прочитанные данные
сразных серверов совпадают

Partition Tolerance
Устойчивость к разделению
системы

Eric Brewer, 2000 г. Теорема Брюера

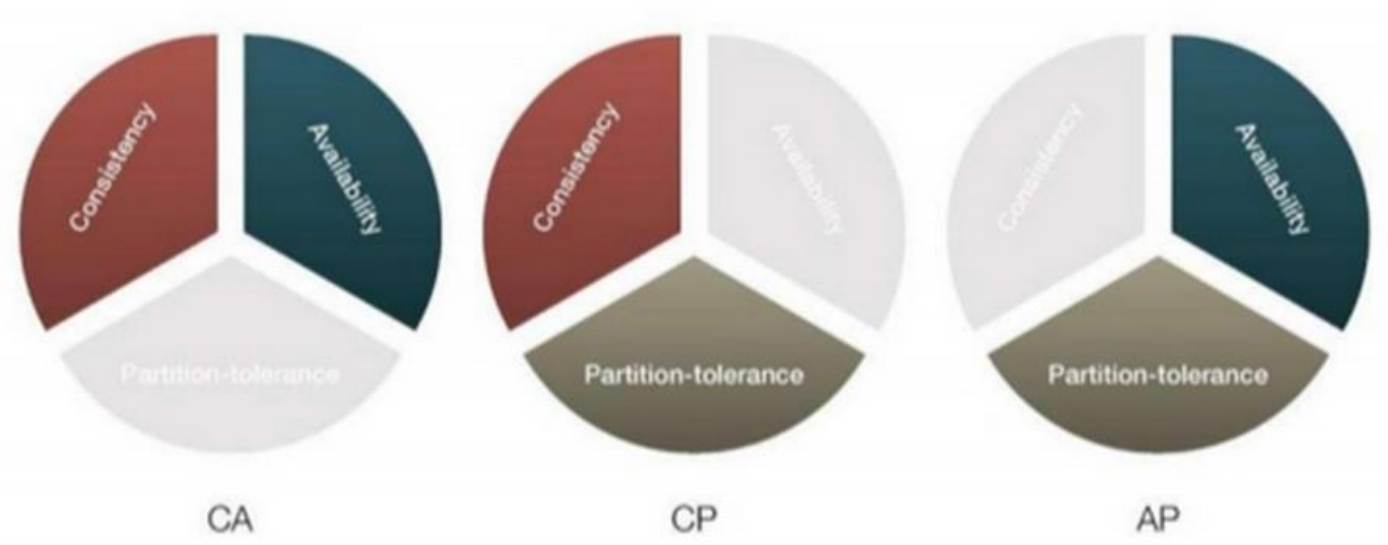


CAP комбинации

CA – система доступна и консистентна. Нежизнепособна в ненадежной сети.

CP – не будет доступна пока нет полной синхронизации между всеми узлами.

AP – данные на разных работающих узлах могут отличаться.

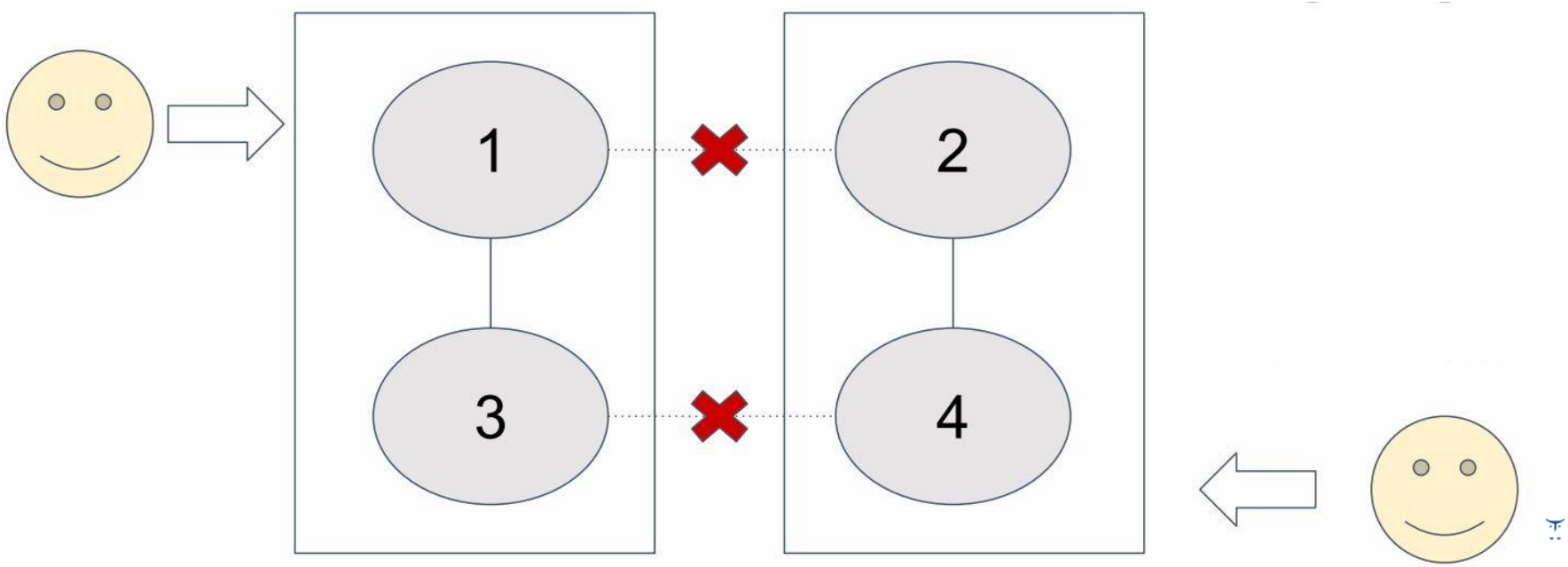


Разделение узлов

CA — не принимаем запросы (плохо, получается **не распределенная**)

CP — разрешаем чтение, запрещаем запись (есть **консистентность**)

AP — разрешаем и чтение и запись (есть **доступность**), данные на разных работающих узлах могут отличаться



Применительно к распределенным системам

Разделяемость + …

Целостность
CP

Доступность
AP

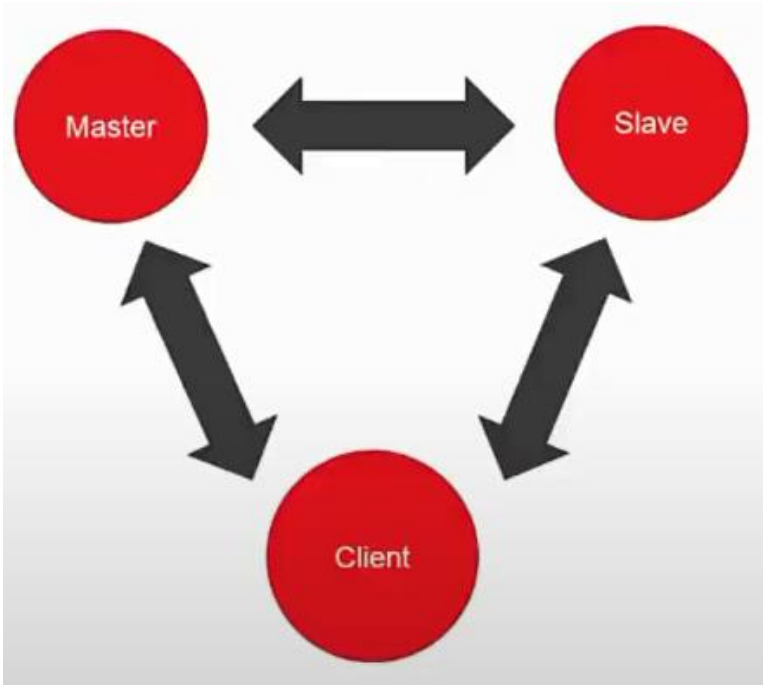
Минусы CAP теоремы

Теорема описывает системы слишком упрощенно.

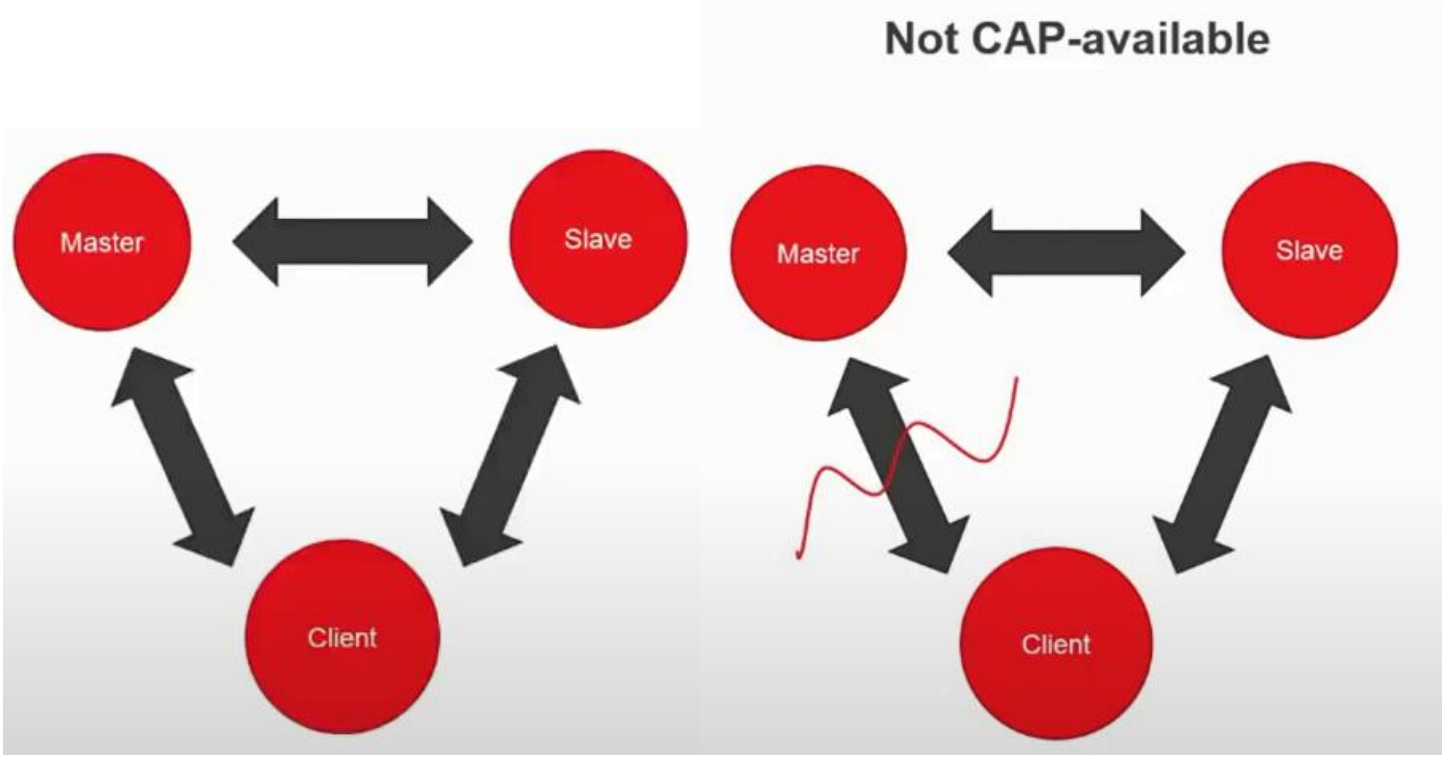
Каждое понятие возведено в абсолют.

Невозможно достичь идеально CAP для всех операций, но можно выбрать, где какой параметр важнее.

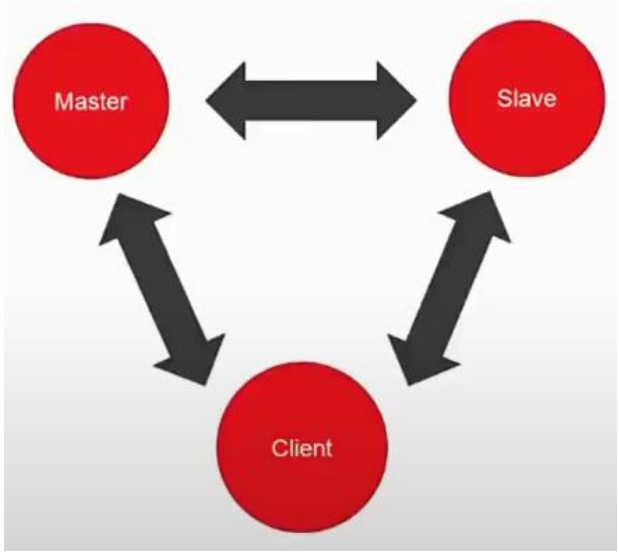
Многие «мелкие и современные» NoSQL – просто P.

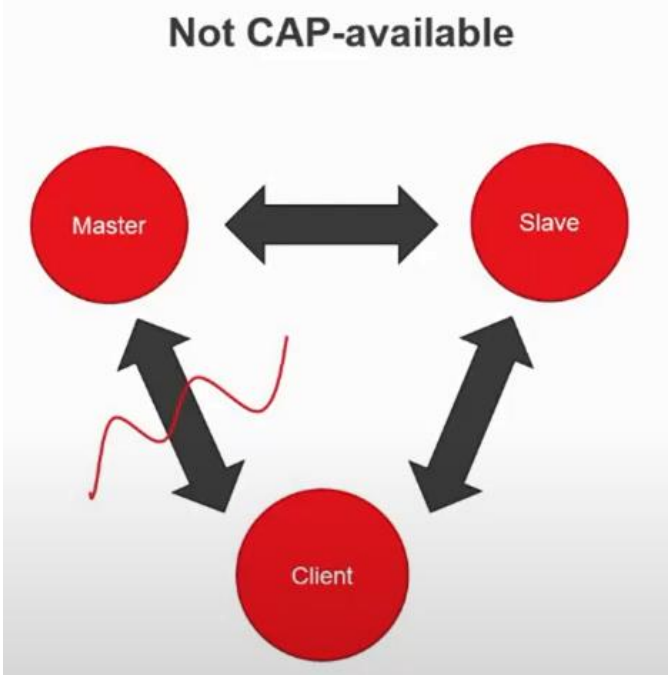


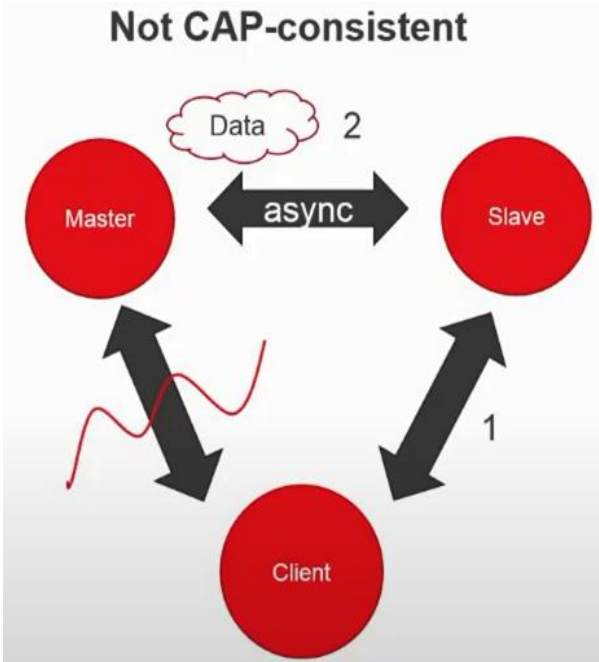
Многие «мелкие и современные» NoSQL – просто P.



Многие «мелкие и современные» NoSQL – просто P.







Расширение теоремы CAP

Теорема PACELC — в случае разделения сети (P) в распределённой компьютерной системе необходимо выбирать между доступностью (A) и согласованностью (C) (согласно теореме CAP), но в любом случае, даже если система работает нормально в отсутствии разделения (E), нужно выбирать между задержками (L) и согласованностью (C).

Даниэль Дж. Абади (Йельский университет) 2010 г.

О CAP теореме на Хабре

Замечания о распределенных системах для начинающих

CAP-теорема простым, доступным языком

Всё, что вы не знали о CAP теореме

Забудьте CAP теорему как более не актуальную

Алгоритмы консенсуса

Алгоритмы консенсуса

Консе́нсус (лат. consensus — «согласие, сочувствие, единодушие») — согласованность между всеми узлами сети по какому-то вопросу.

Как распределить изменения по всем узлам?

Если есть ведущий узел, но он упал. Как выбрать нового?

CP. Алгоритм PAXOS

Паксос (англ. *Paxos*) — семейство протоколов для решения задачи консенсуса в сети ненадёжных вычислителей.

Основная проблема — наличие помех в среде передачи данных.

Используется для утверждения транзакций в распределённых системах.

PAXOS demo: [Neat Algorithms - Paxos - Will You Harry Me]

CP. Алгоритм Raft

Raft разрабатывался с учётом недостатков более старого алгоритма PAXOS.

Для обеспечения консенсуса в Raft сначала выбирается **лидер**, на котором будет лежать ответственность за управление распределённым логом.
Лидер **принимает запросы от клиентов и реплицирует их на остальные сервера** в кластере.
В случае выхода лидера из строя, в кластере будет **выбран новый лидер**.

Состояния:

- **Leader (лидер)** – обрабатывает все клиентские запросы.
- **Follower (фоловер)** – пассивный сервер, который только «слушает» новые записи в лог от лидера и редиректит все входящие запросы от клиентов на лидера.
- **Candidate (кандидат)** – специальное состояние сервера, возможное только во время выбора нового лидера.

CP. Алгоритм Raft

Если обычный узел долго не получает сообщений от лидера, то он переходит в состояние «кандидат» и посылает запрос на голосование. Другие узлы голосуют за того кандидата, от которого они получили первый запрос.

Если кандидат получает сообщение от лидера, то он снимает свою кандидатуру и возвращается в обычное состояние.

Если кандидат получает большинство голосов, то он становится лидером. Если же он не получил большинства (возникло сразу несколько кандидатов и голоса разделились), то кандидат ждёт случайное время и инициирует новую процедуру голосования.

Процедура голосования повторяется, пока не будет выбран лидер.

Raft (thesecretlivesofdata.com)

AP. Алгоритм Gossip

Gossip (англ. *сплетник*) — это группа протоколов, в которых распространение информации идёт способом, схожим с распространения эпидемий (эпидемический протокол).

Каждый или некоторые из узлов могут передавать обновляемые данные известным соседям.

Используются для обеспечения распространения данных между всеми узлами распределённой системы.

gossip demo: [Scuttlebutt (awinterman.github.io)]

Что делать?

- помнить про теорему CAP и ее ограничения;
- проектирование системы стоит начинать с согласования компромиссов;
- помнить про ACID / BASE, насколько они применимы к системе;
- все зависит от проекта, над которым вы работаете.

Домашнее задание

Домашнее задание

Необходимо написать к каким системам по CAP теореме относится MongoDB.

ДЗ сдается ссылкой на гит, где расположен миниотчет в маркдауне.

Рефлексия

Рефлексия

С какими впечатлениями уходите с вебинара?

Заполните, пожалуйста, опрос о занятии по ссылке в чате

Спасибо за внимание!

Приходите на следующие вебинары

Коробков Виктор