



O T U S

ОНЛАЙН ОБРАЗОВАНИЕ

Онлайн-образование



Меня хорошо видно && слышно?

Ставьте ☐ , если все хорошо
Напишите в чат, если есть проблемы

Не забыть включить запись!



The background of the slide is an aerial photograph of a city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue layer. On the left side of this layer, there is a network diagram consisting of white dots connected by thin white lines, forming a complex web. The text 'CAP теорема' is centered in the upper half of the blue area.

CAP теорема

Курс NoSQL

Содержание

1. Транзакции, ACID
2. Распределенные системы
3. CAP теорема, BASE
4. Алгоритмы консенсуса
5. Домашнее задание

Стрекалов Павел

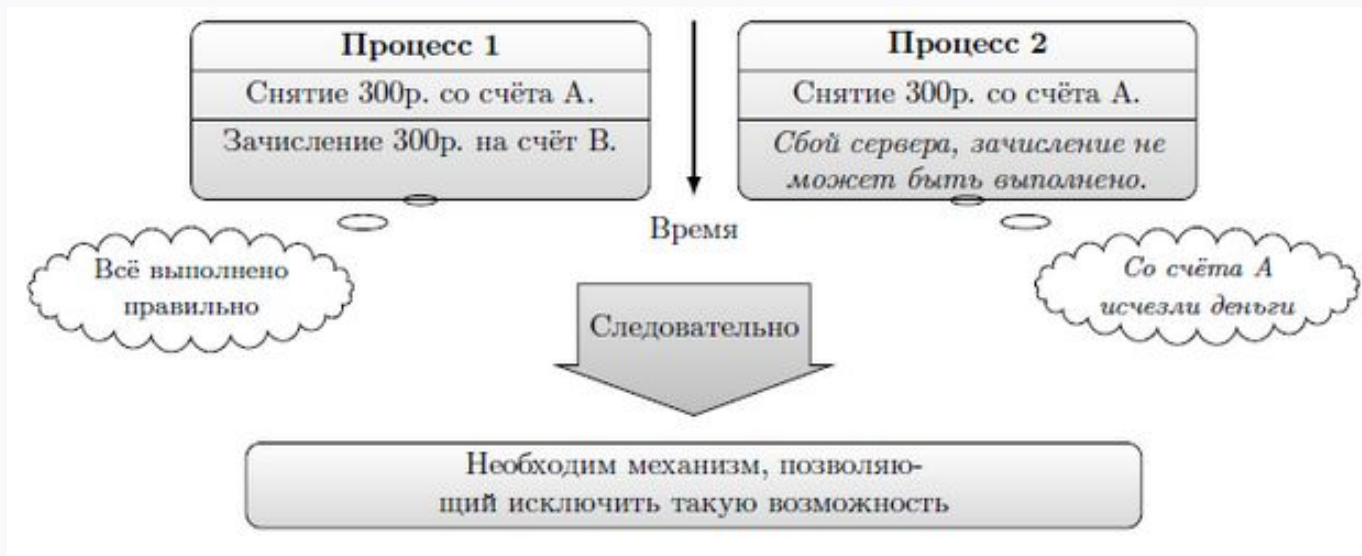
- Около 15 лет опыта профессиональной разработки
- Основной стек: C#, Java, MS SQL Server
- Окончил МИЭТ, 2006 г
- В OTUS преподаю на курсах
«MS SQL Server разработчик»,
«Разработчик Java»





Транзакции, ACID

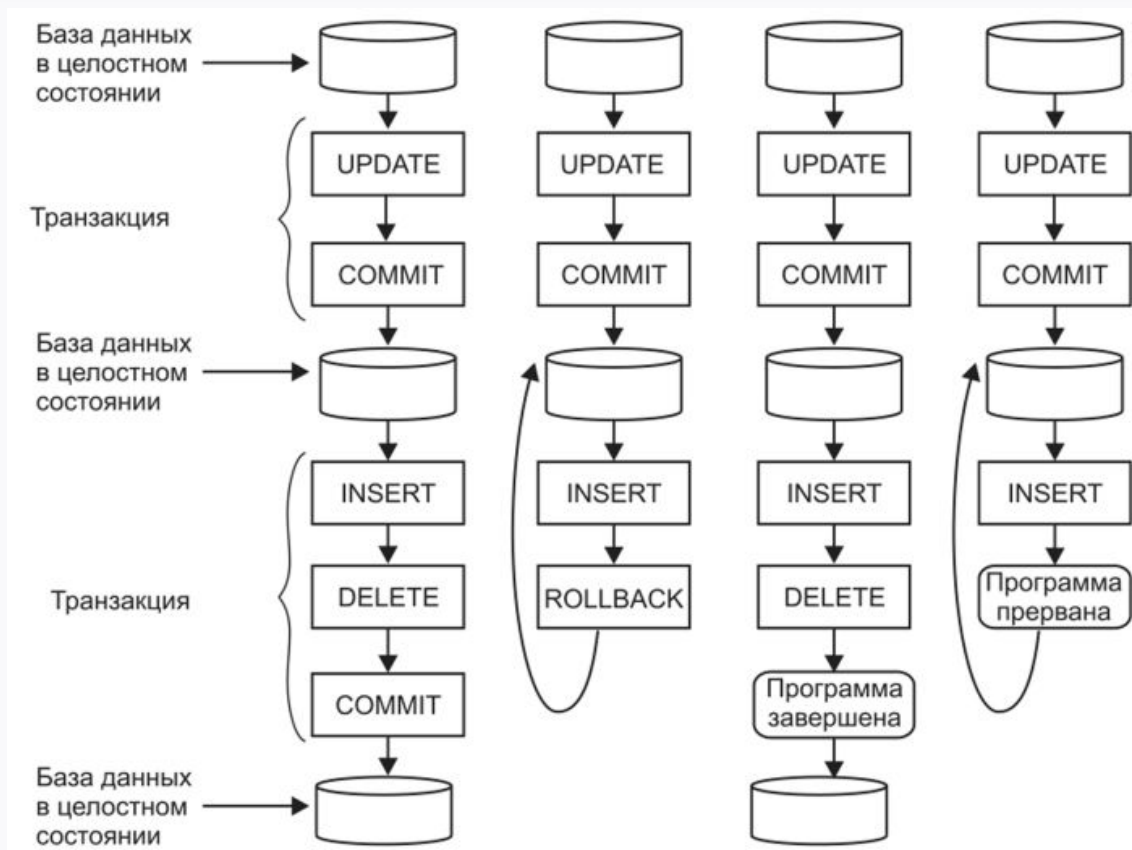
Транзакция



Транзакция — это последовательность действий, выполняющаяся как единое целое и переводящая базу данных из одного согласованного состояния в другое согласованное состояние.

Транзакция завершается подтверждением изменений (commit) либо откатом изменений (rollback).

Транзакция



Свойства транзакций ACID

- **Atomicity** — атомарность

Выполняется все (commit) или ничего (rollback)

- **Consistency** — согласованность

Данные остаются в согласованном состоянии

- **Isolation** — изолированность

Транзакции разных пользователей не мешают друг друга

- **Durability** — долговечность, стойкость

Ничего не потеряется после фиксации транзакции

Изолированность

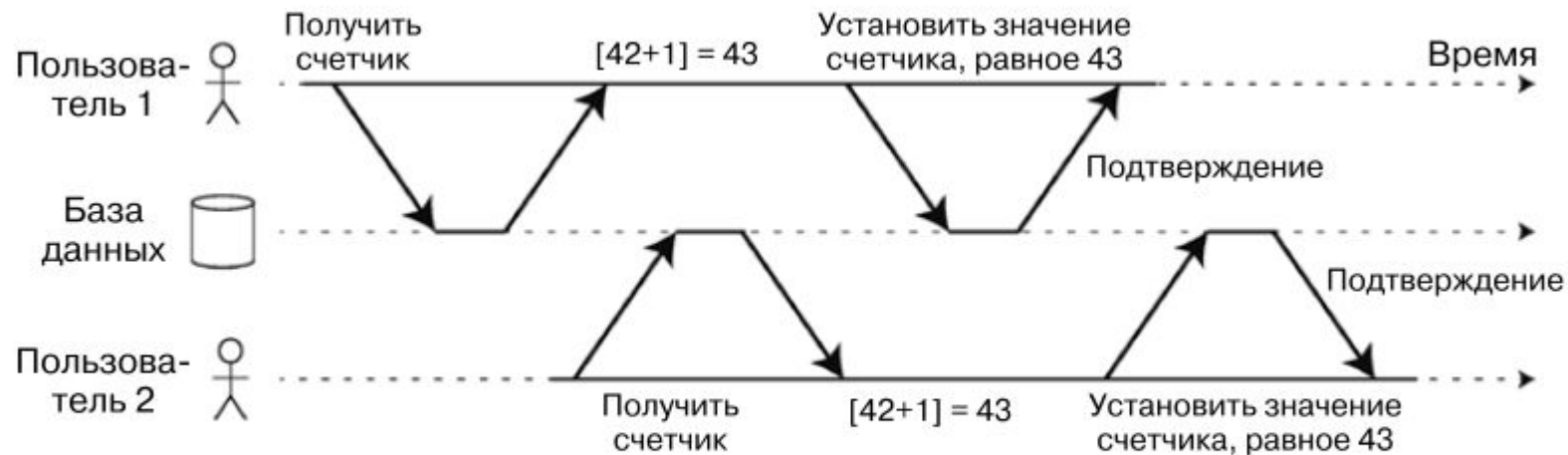


Рис. 7.1. Состояние гонки между двумя клиентами, конкурентно увеличивающими значение счетчика

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.

The background of the slide is an aerial photograph of a dense city skyline, likely New York City, with numerous skyscrapers. A semi-transparent blue overlay covers the entire image. In the center, there is a network diagram consisting of a series of interconnected nodes and lines, resembling a mesh or a web, which is also rendered in a light blue color. The title text is centered over this background.

Распределенные системы

Масштабирование

Вертикальное



Горизонтальное



Отказоустойчивость,
высокая доступность



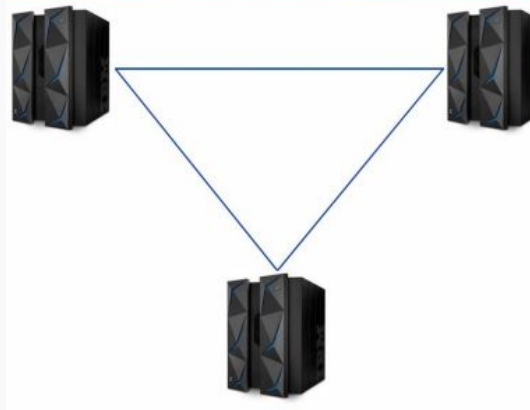
Синхронизация
Задержки
Ошибки



Распределенные системы

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

Нет центрального узла - если вырубить любой компонент системы, система не должна упасть.

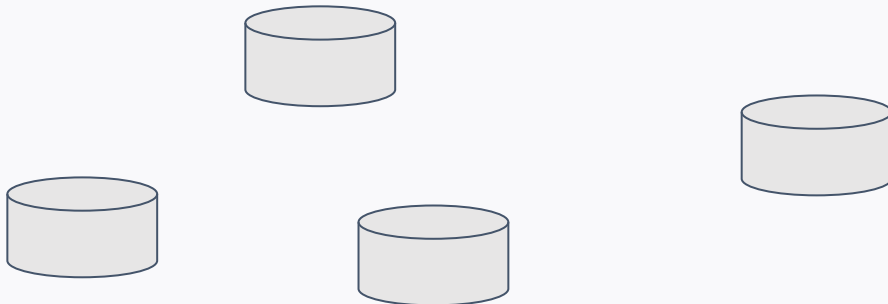


Распределенной вычислительной системой можно назвать такую систему, в которой отказ компьютера, о существовании которого вы даже не подозревали, может сделать ваш собственный компьютер непригодным к использованию.

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

Виды распределенных систем

- Без состояния (stateless)
 - Это нас не интересует (в контексте данного курса)
- С состоянием (stateful)
 - Участники сети делятся общим состоянием, то каким-то образом синхронизируют его.
 - Говорим об этом



Заблуждения о распределенных системах

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

А еще существуют отказы узлов

Модели отказов в распределенных системах:

- 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.

Проблемы ACID

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

Дорого – дата центры должны быть связаны выделенным каналом.

Избыточно – такой уровень надежности нужен не всегда (например, сообщения или фотографии соц. сети).

Чем плох ACID?

- ACID дорогой и не всегда нужен
- если нам надо всего лишь:
 - кэшировать значения
 - построить аналитическую модель
 - поставить лайк
 - загрузить фотку
 - и др



CAP теорема, BASE

CAP – свойства распределенных систем

Consistency (согласованность)

на всех узлах одинаковые данные.

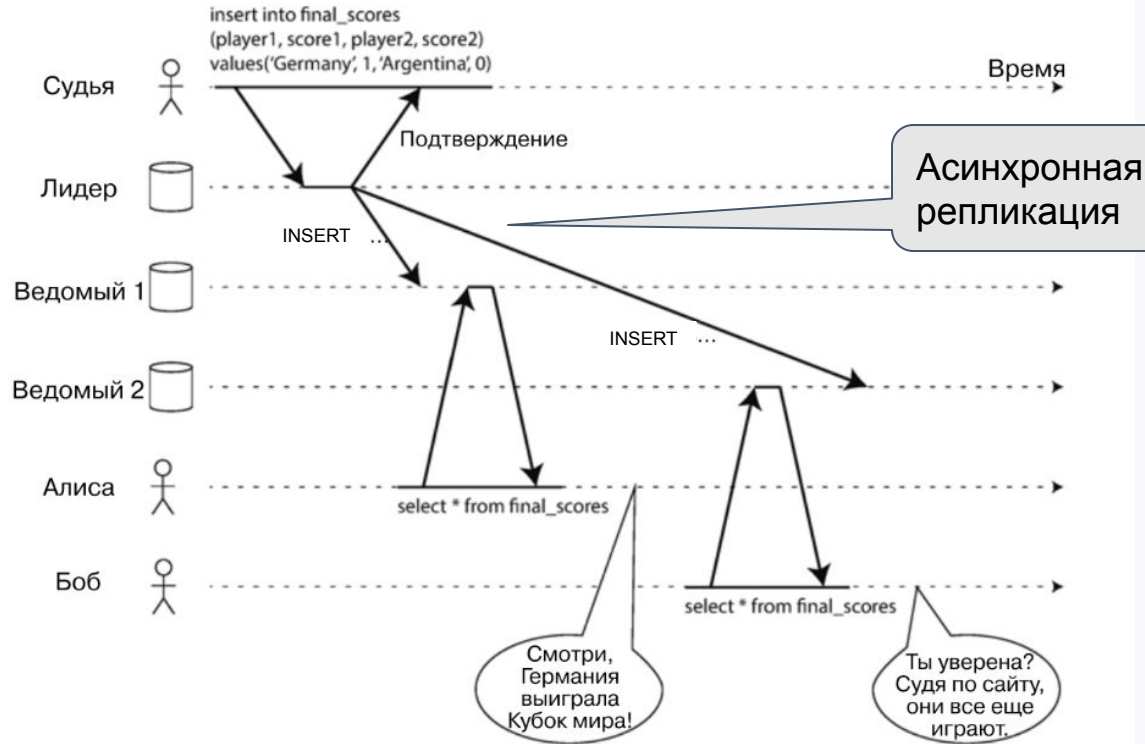
Availability (доступность)

любой работающий узел всегда
выполнит запрос и на чтение и на запись
(без гарантии актуальных данных)

Partition Tolerance (устойчивость к разделению сети)

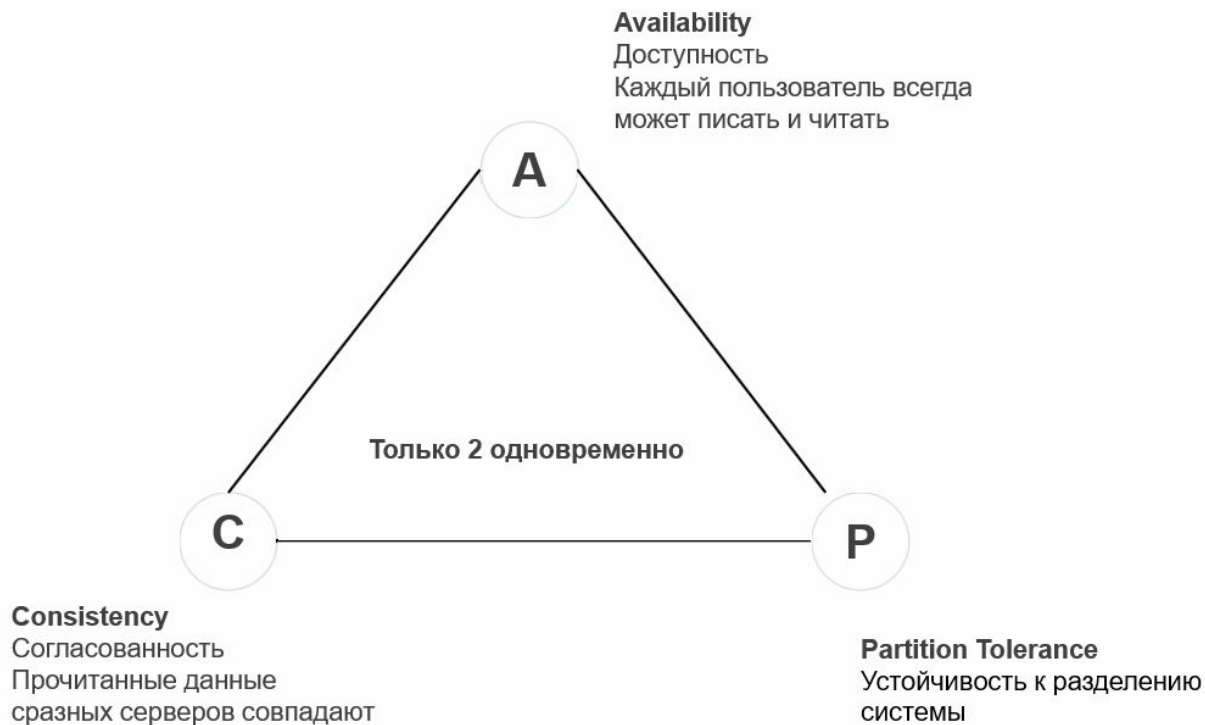
система продолжает работать, несмотря на
отсутствие связи между узлами

Согласованность vs доступность

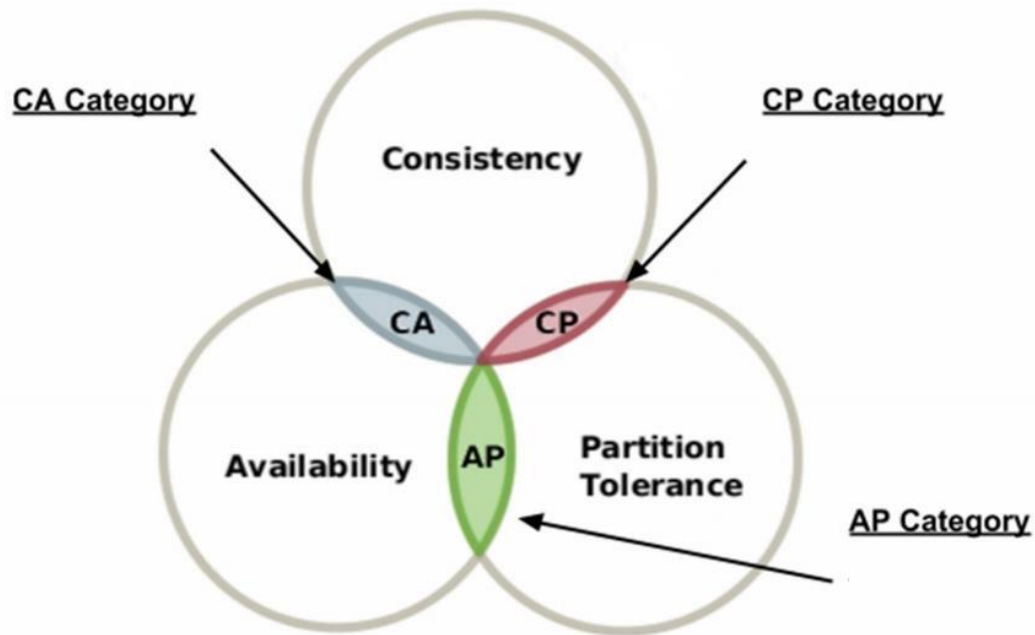


Доступность есть,
Согласованности нет

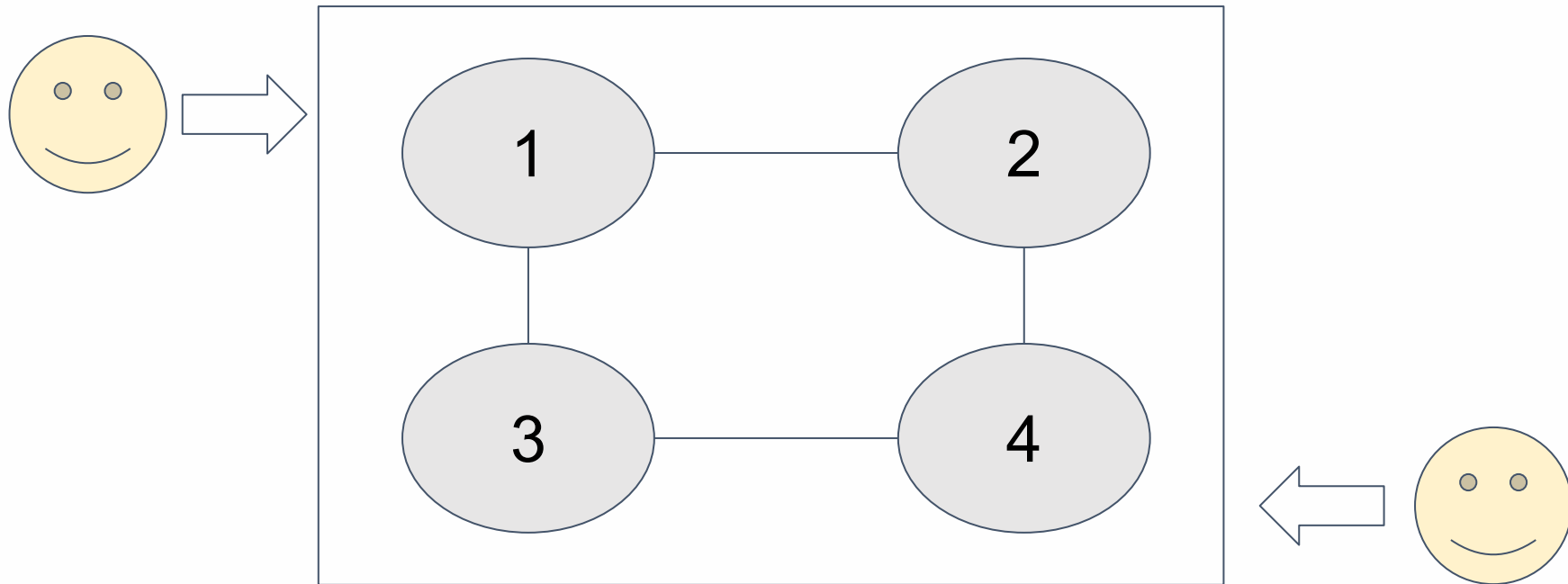
CAP теорема



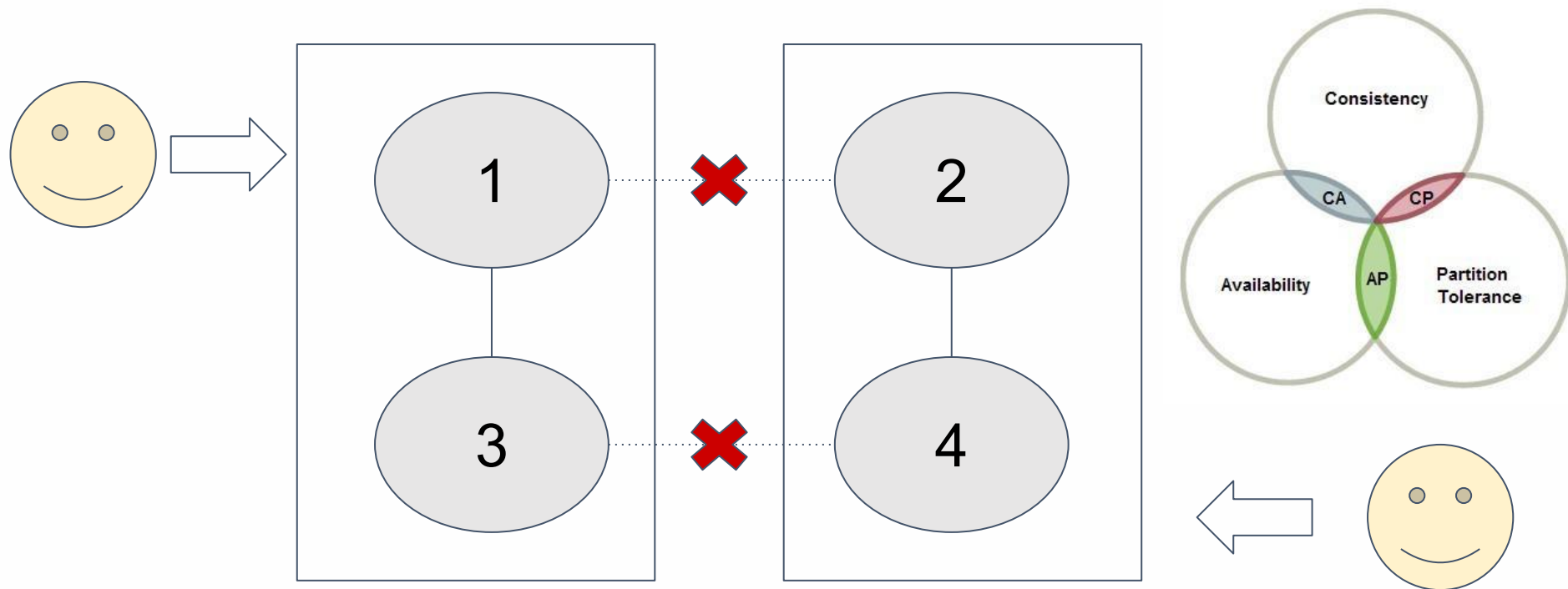
Категории систем



Все работает



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

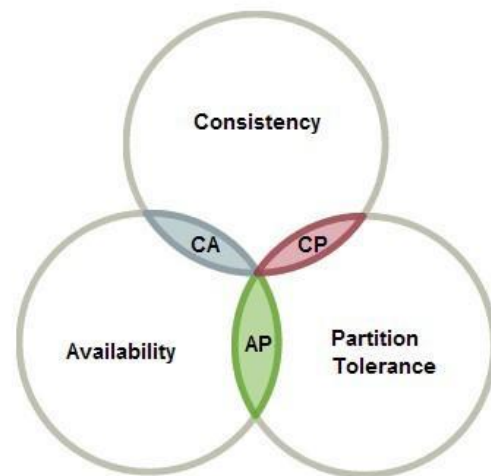
DNS

К какому классу систем относится DNS?

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

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

AP — разрешаем и чтение и запись (есть **доступность**)



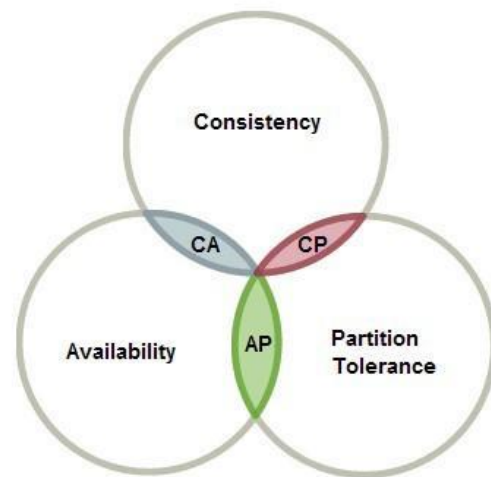
DNS

К какому классу систем относится DNS?

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

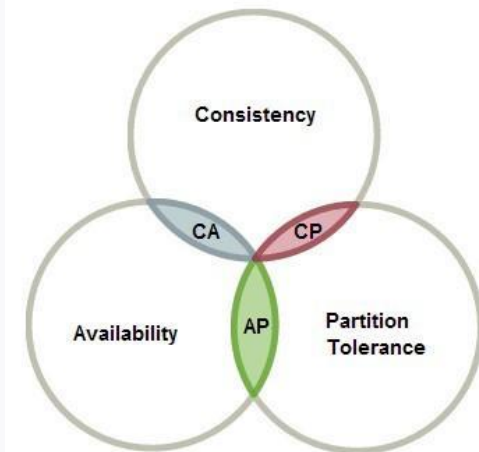
CP — разрешаем чтение, запрещаем запись (есть **консистентность**)

AP — разрешаем и чтение и запись (есть **доступность**)



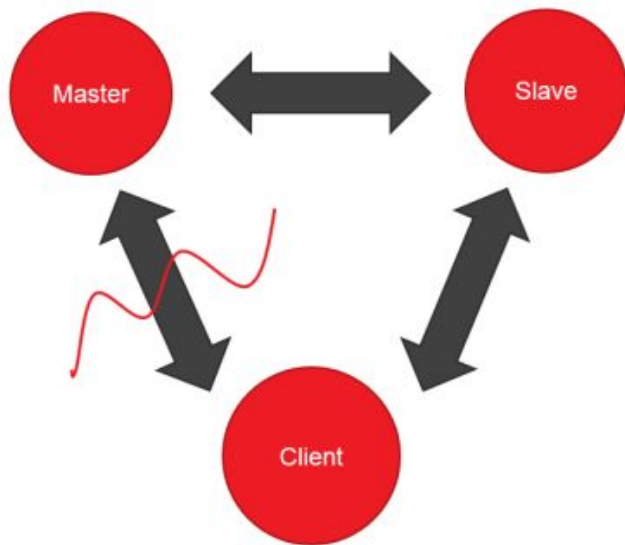
Недостатки, проблема, критика CAP теоремы

- Теорема описывает системы слишком упрощенно (AP vs CP).
- Каждое понятие возведено в абсолют.
 - Консистентность полная, на всех узлах
 - Доступность не подразумевает частичную доступность узлов
- Невозможно достичь идеально CAP для всех операций, но можно выбрать, где какой параметр важнее.

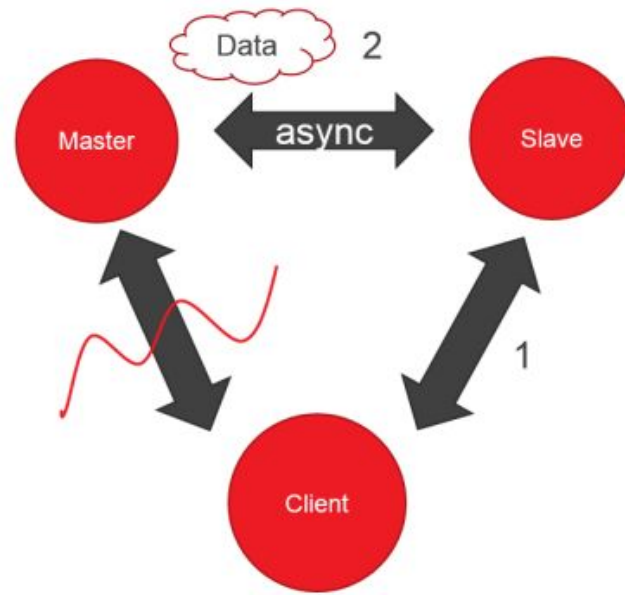


Множество систем просто "P"

Not CAP-available



Not CAP-consistent



Всё, что вы не знали о CAP теореме <https://habr.com/ru/post/328792/>

BASE-архитектура vs ACID

- **Basic Availability**

Система всегда отвечает на запрос, но могут быть несогласованные данные.

- **Soft-state**

Состояние системы может меняться со временем даже в отсутствие запросов клиентов на изменение данных. Потому что в любой момент времени данные могут приводиться в согласованное состояние.

- **Eventual consistency** (конечная согласованность)

Система, в конечном итоге, придет в согласованное состояние, если данные не будут изменяться.

Почитать про CAP-теорему 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](#)

The background of the slide is an aerial photograph of a dense city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue layer. In the center of this layer, there is a white network diagram consisting of interconnected nodes and lines, resembling a blockchain or distributed ledger structure. The title text is centered over this graphic.

Алгоритмы консенсуса (согласования)

Алгоритмы консенсуса

Консѐнсус (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?

СР. 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.

Не leader-follower (принимать изменения могут разные ноды).
Двухфазный коммит.

[Wikipedia](#)

PAXOS demo - <http://harry.me/blog/2014/12/27/neat-algorithms-paxos/>

Raft

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**.

- **Leader (лидер)** – обрабатывает все клиентские запросы, является source of truth всех данных в логе, поддерживает лог фоловеров.
- **Follower (фоловер)** – пассивный сервер, который только «слушает» новые записи в лог от лидера и редиректит все входящие запросы от клиентов на лидера. По сути, является hot-standby репликой лидера.
- **Candidate (кандидат)** – специальное состояние сервера, возможное только во время выбора нового лидера.

"Как сервера договариваются друг с другом: алгоритм распределённого консенсуса Raft"

<https://habr.com/ru/company/dododev/blog/469999/>

СР. Алгоритм Raft

- Raft разрабатывался с учётом недостатков PAXOS.
- 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.

[wikipedia](#)

RAFT demo <https://thesecretlivesofdata.com/raft/>

АР. Протокол Gossip

Протокол Gossip — это протокол для распределенных систем, состоящих из равноправных узлов.

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

Gossip-протоколы обеспечивают eventual consistency в распределённой системе. В CAP-теореме они жертвуют согласованностью, получают доступность и устойчивость к разделению.

Основная идея: узлы распространяют информацию об изменениях друг другу по мере возможности, причём не только самостоятельно добавленные изменения, но и то, что слышали от других узлов (слухи, gossip). Тогда при отсутствии сбоев рано или поздно все узлы обо всём узнают.

Обычно реализуется в форме случайного «однорангового выбора»: с заданной частотой каждый узел одноранговой сети случайным образом выбирает другой известный ему узел и передаёт тому обновлённую информацию.

[wikipedia](#)

Scuttlebutt gossip demo <https://awinterman.github.io/simple-scuttle>

The background of the slide is an aerial photograph of a dense city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue layer. On the left side of this layer, there is a network diagram consisting of light blue dots connected by thin lines, forming a web-like structure. The text "Домашнее задание" is centered in the middle of the slide in a large, white, sans-serif font.

Домашнее задание

Домашнее задание

Необходимо написать к каким системам по CAP теореме относятся перечисленные БД и почему: MongoDB, MSSQL, Cassandra.

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

The image features a background of a dense city skyline, likely New York City, viewed from an elevated perspective. The entire image is tinted with a blue and teal color palette. A network of glowing teal lines and dots is overlaid on the image, creating a digital or technological feel. The word "Рефлексия" is centered in the middle of the image in a large, white, sans-serif font.


Рефлексия

Что делать?

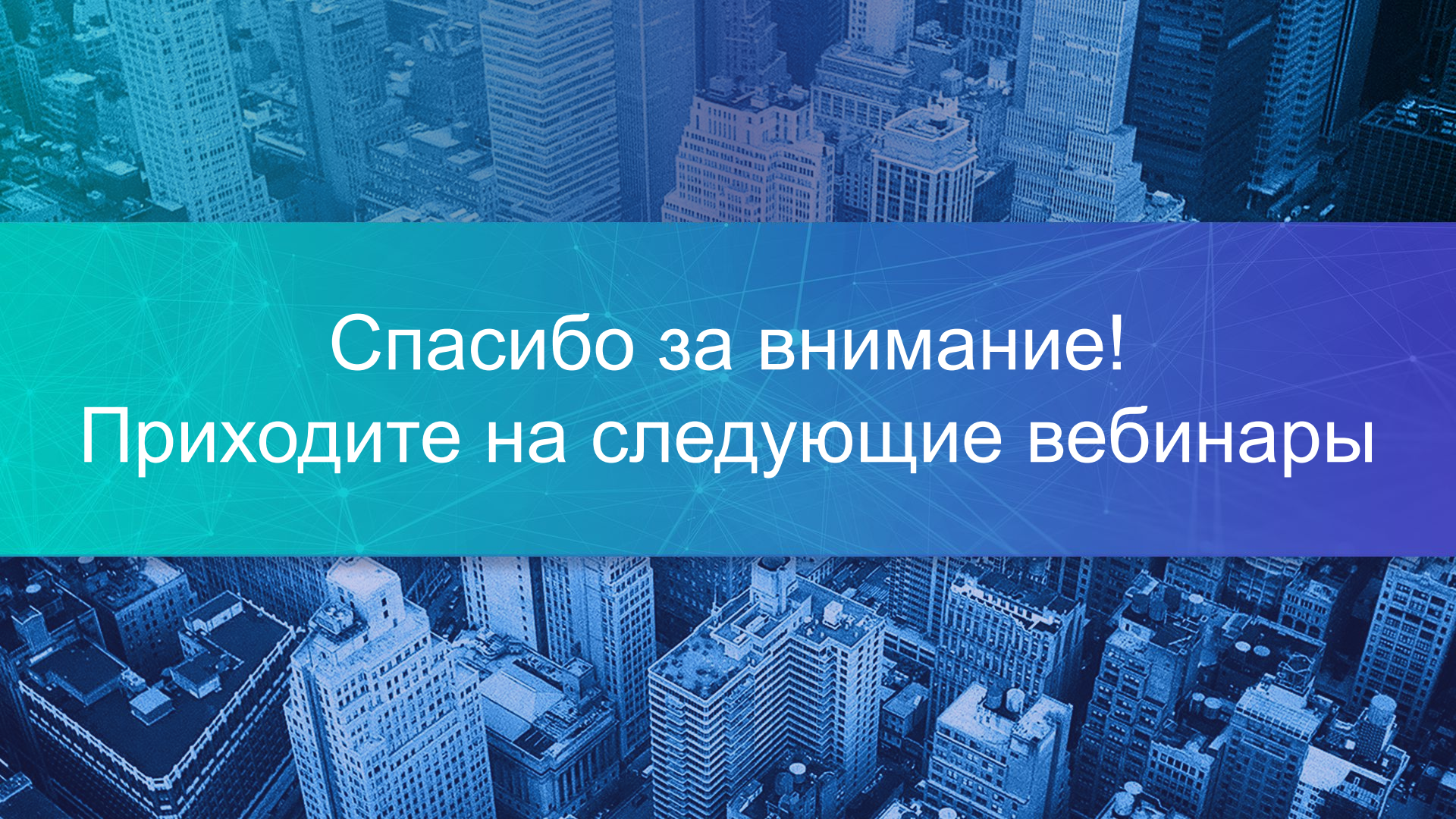
- помнить про теорему CAP и ее ограничения;
- проектирование системы стоит начинать с согласования компромиссов;
- помнить про ACID / BASE, насколько они применимы к системе;
- все зависит от проекта, над которым вы работаете.

Выбор СУБД

1. Объем данных.
2. Размер транзакций: длительность, количество одновременных транзакций.
3. Количество одновременных соединений/сессий.
4. Соотношение операций чтения/записи.
5. Масштабирование: вертикальное, горизонтальное.
6. ...



Заполните, пожалуйста,
опрос о занятии по ссылке в чате

The background of the slide is an aerial photograph of a city skyline, likely New York City, with numerous skyscrapers. The image is overlaid with a semi-transparent blue layer. A network of white lines connects various points across the blue area, creating a digital or technological aesthetic. The text is centered in this blue area.

Спасибо за внимание!
Приходите на следующие вебинары