System Design

Teorema CAP, ACID, BASE e PACELC

$ whoami

Matheus Fidelis

Engenheiro de $RANDOM

@fidelissauro

https://fidelissauro.dev

https://linktr.ee/fidelissauro

OBJETIVOS

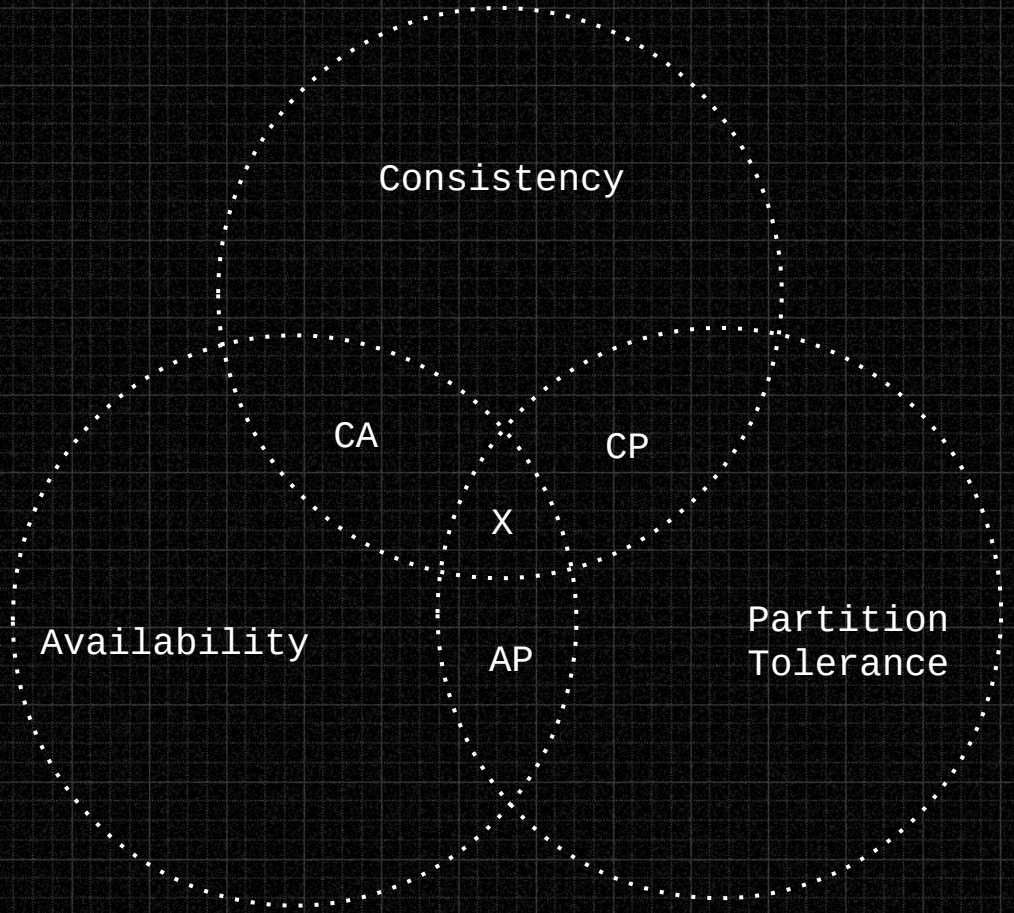
- Entender o Teorema CAP
- Entender Tradeoffs de Databases
- Entender o Modelo ACID
- Entender o Modelo BASE
- Consistência Eventual e Forte

1
INTRODUÇÃO AO TEOREMA CAP

Definições e Conceitos

Introdução ao Teorema CAP

- Consistency, Availability, and Partition Tolerance
- Consistencia, Disponibilidade e Tolerância a Partições
- Modelo Conceitual
- Eric Brewer, 2000
- Análogo a escolher apenas 2 dos 3 principios
- Escolher Tradeoffs entre databases


Consistency
CA
CP
X
Availability
AP
Partition
Tolerance

2. MODELO ACID TRANSACIONAL

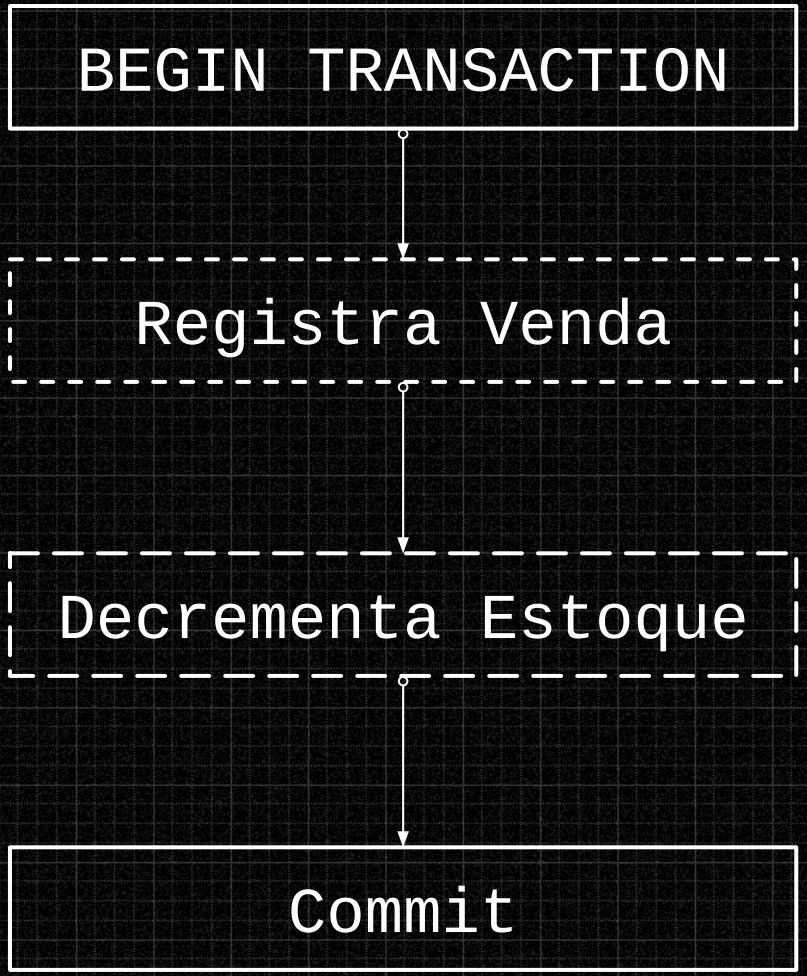
Definições e Conceitos

Modelo ACID

- Databases Transacionais
- Databases SQL
- Atomicidade, Consistência, Isolamento e Durabilidade
- Operações Atômicas e Confiáveis
- Transação e Commits são priorizados
- Detrimento da performance e disponibilidade

Atomicidade

- Assegura que cada transação seja tratada como uma unidade indivisível
- Todas as operações dentro da transação devem ser concluídas
- Caso contrário, nenhuma delas será efetivada
- Consistencia Forte
- Operações que precisam de interdependência
- Sistemas Financeiro


BEGIN TRANSACTION
Registra Venda
Decrementa Estoque
Commit


BEGIN TRANSACTION
Registra Transação
Aumenta/Reduz Saldo
Commit

Consistência

- Estado consistente para outro estado consistente
- Integridade dos dados
- Evitando dados corrompidos ou inválidos
- Validação das transações
- Tipos, Foreign Keys, Nullabilidade, Triggers
- "Inserir uma string em um tipo decimal"

Isolamento

- Operar independentemente de outras transações simultâneas
- Transações ocorrendo ao mesmo tempo não interfiram umas nas outras
- Dirty Reads
- Non-repeatable Reads
- Phantom Reads

Durabilidade

- Após a confirmação de uma operação de, ela não será perdida
- Confirmada, uma transação permanecerá assim permanentemente.
- Persistência dos dados em uma fonte não volátil

3. MODELO BASE E ESTADO EVENTUAL

Definições e Conceitos

Modelo BASE

- Basic Availability, Soft-State, Eventual Consistency

- Flexibilidade mais adequados para lidar com sistemas distribuídos modernos
- Disponibilidade e a tolerância a falhas são prioridades
- Também proposto por Eric Brewer
- Sistemas que lidem com consistencia eventual

Basicamente Disponível

- Maximizar a disponibilidade
- Disponibilidade total e ininterrupta
- Alguns dados ou funcionalidades podem não estar disponíveis
- Replicação e particionamento
- Múltiplos Servidores e Nós
- Se uma parte desses Nós ficarem indisponíveis, outros continuam funcionando
- Ambientes de Larga Escala e Alta Demanda

Soft State

- Sistema pode mudar a longo do tempo
- Mesmo sem um estímulo ou intervenção externa
- Não há garantia que o sistema permaneça consistente
- Dados podem se autogerenciar, auto deletar e auto atualizar
- Expiração e Tempos de Vida
- Databases Voláteis
- Memcached, Redis e Etc.

Eventualmente Consistente

- Não confirmação em todos os nodes
- Escrita pode acontecer num nó, e não se confirmar nos demais
- Replicação Assincrona
- Por um breve momento, os dados podem estar inconsistentes entre diversos nós
- Alta disponibilidade e escalabilidade
- Performance

4. COMPONENTES DO CAP (C, A, P)

Definições e Conceitos

Consistency / Consistência (C)

- Garantia de que todos os nós de um banco de dados distribuído exibam os mesmos dados simultaneamente
- Independentemente de qual nó seja consultado, todos retornarão sempre a versão mais recente
- Atomicidade
- Sistemas financeiros e registros hospitalares

Availability / Disponibilidade (A)

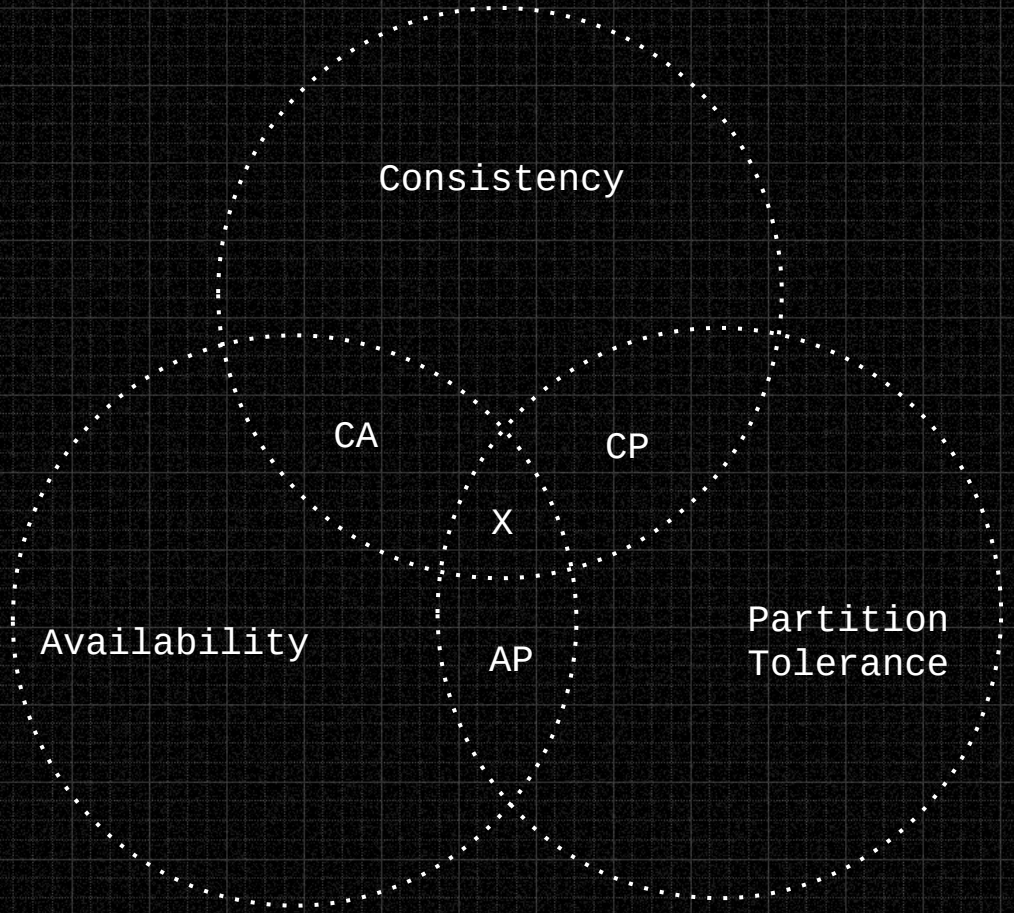
- Assegura que todas as solicitações feitas ao sistema receberão uma resposta
- Independentemente de os dados estarem atualizados no nó consultado ou não
- Presume que o dado retornado pode não ser o mais recente, desde a operação de escrita até a de leitura
- Alta Performance, Alto Volume, Tempos de Resposta Rápidos
- Detrimento da Garantia do Dado Atualizado

Partition Tolerance / Tolerância a Partições (P)

- Capacidade de um banco de dados distribuído continuar operacional
- Falhas que “particionem” a rede e duas ou mais partes que não conseguem mais se comunicar
- Oferecer um certo nível de continuidade do serviço em caso de falhas parciais
- Databases Geodistribuídos, Redes Sociais, Agregadores de Logs, Brokers, Filas e etc

5. COMBINAÇÕES

Definições e Conceitos


Consistency
CA
CP
X
Availability
AP
Partition
Tolerance

CP (Consistência e Tolerância a Partições)

- Prioriza a consistência e a tolerância a partições
- Sacrificando a disponibilidade
- Mantém a consistência através de todos os nós que continuam operando em caso de falhas de rede ou partições
- Capacidade de desativar os nós inconsistentes, tornando-os indisponíveis até que a consistência seja restaurada

CP (Consistência e Tolerância a Partições)

- Precisão dos dados
- atomicidade transacional
- MongoDB
- Cassandra*
- Coachbase
- Etcd
- Consul

AP (Disponibilidade e Tolerância a Partições)

- Prioriza a alta disponibilidade e tolerância a partições
- Sacrificando a consistência
- Todos os nós permanecem disponíveis para consultas, independentemente do seu nível de atualização
- Durante processos de ressincronização, todos os nós continuarão respondendo

AP (Disponibilidade e Tolerância a Partições)

- Continuidade da operação
- Mais importante do que a manutenção de dados consistentes
- DynamoDB
- CouchDB
- Cassandra*
- SimpleDB

CA (Consistência e Disponibilidade)

- Prioriza a consistência e a disponibilidade das solicitações
- Sensível a partições de dados
- Falha de rede ou partição, o sistema pode ficar completamente inoperante

CA (Consistência e Disponibilidade)

- Bancos de Dados Centralizados
- Bancos de Dados SQL Tradicionais
- MySQL/MariaDB
- PostgreSQL
- Oracle
- SQL Server
- Redis Standalone
- Memcached Standalone

6. CRÍTICAS AO MODELO DO CAP

Definições e Conceitos

CAP no dia de hoje

- Eric Brewer,2012
- Twelve Years Latter: How the "Rules" Have Changed
- 2/3 não é altamente exclusiva
- Simplificação excessiva não é real em tecnologias modernas
- Consistência e disponibilidade (on/off)
- Níveis de Consistência - não binário

7. Teorema PACELC

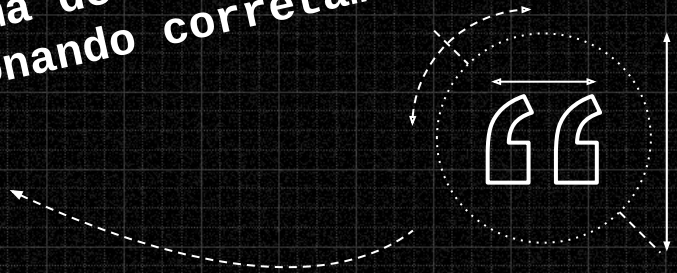
Particionamento Moderno

Teorema Pacelc

- Daniel Abadi, 2010 (Yale)
- Extensão do Teorema CAP
- Nos modelos CP e AP
- Partition Tolerance
- Lacunas do CAP (Nem sempre há partições)

"O que aconteceria com um sistema quando não houver falhas de rede?"

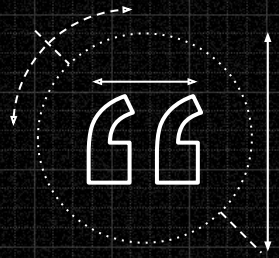
"O que o sistema deve priorizar quando está funcionando corretamente?"



"o que ele deve priorizar quando ocorre um particionamento entre os nós?"

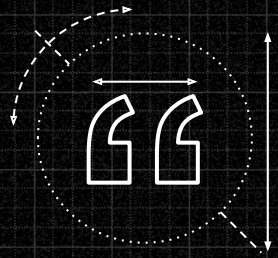
"É possível operar em mais de um nível de consistência?"

Teorema CAP vs PACELC - CAP



"Quando ocorre uma Partição de Rede (P) entre os nós do sistema, é necessário escolher entre Consistência (C) ou Disponibilidade (A)"

Teorema CAP vs PACELC - PACELC

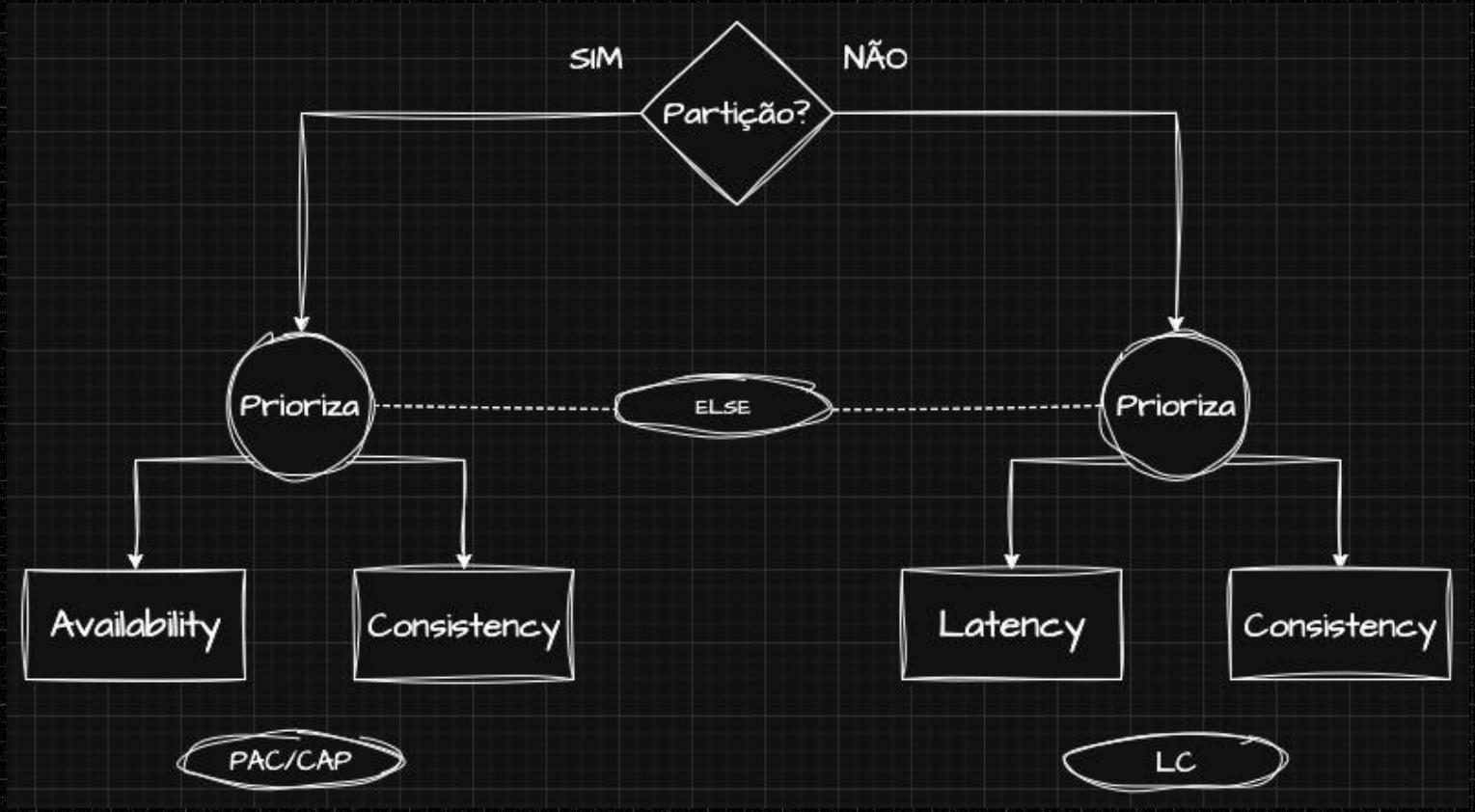


"Se houver partição (P), devemos escolher entre Disponibilidade (A) e Consistência (C)"

ELSE*

"Se não houver partição, escolhemos entre Latência (L) e Consistência (C)"

Teorema CAP vs PACELC - PACELC



PACELC

ON

(P)artition

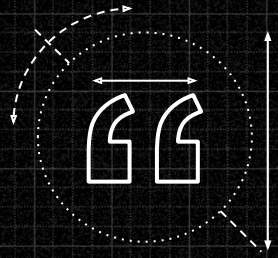
(A)vailability

(C)onsitency

(E)lse

(L)atency

(C)consistency



Teorema Pacelc

- Mesmo em condições normais
- Ainda é preciso tomar decisões difíceis
- Maior garantia em troca de Tempo de Resposta
- Tempo de Resposta em troca de Consistência
- Consistência Forte
- Consistência Eventual
- Os dois teoremas não são excludentes, mas complementares

8. Aplicação do PACELC

Particionamento Moderno

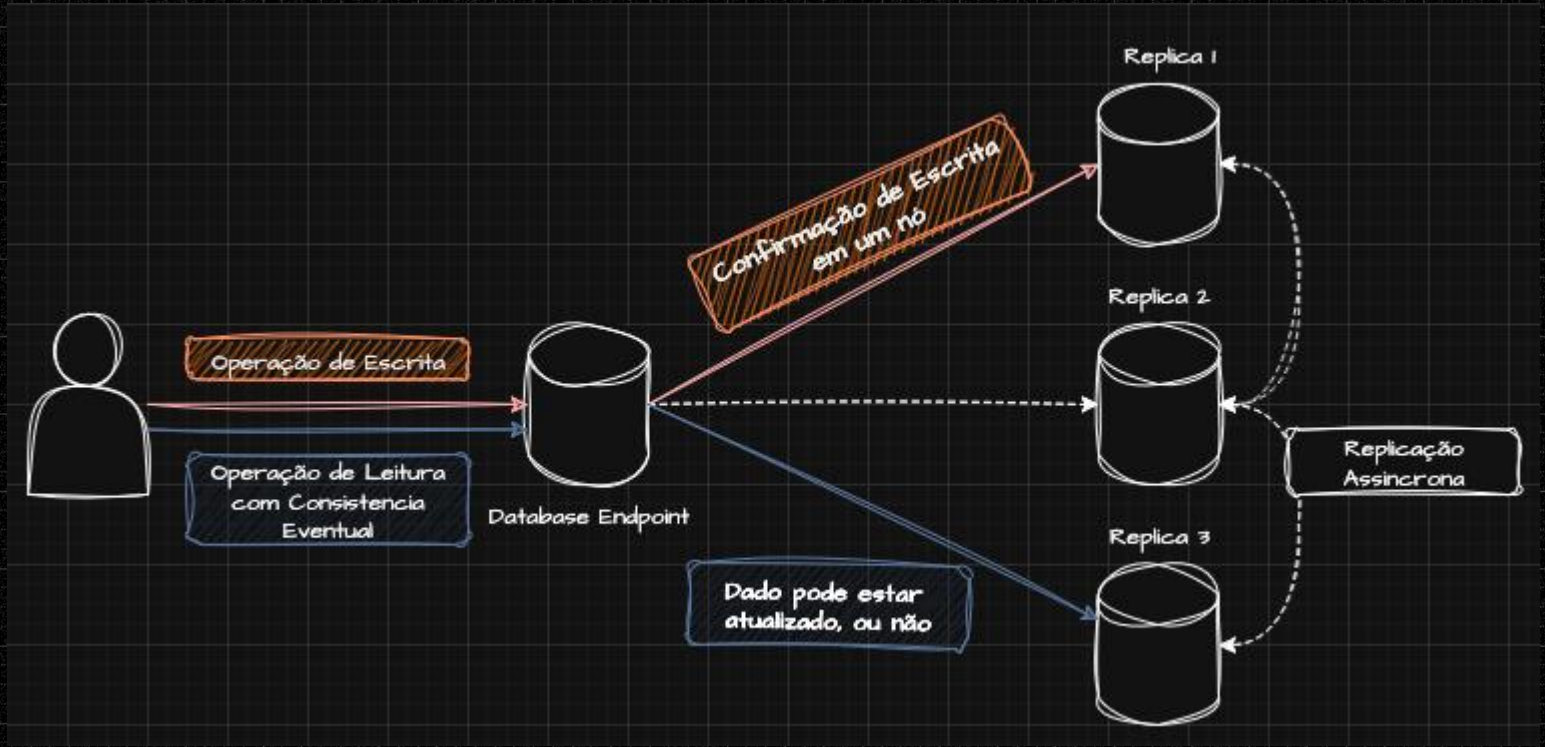
Aplicações Pacelc

- PA/EL (On Partition, Availability; Else, Latency)
- PC/EL (On Partition, Consistency; Else, Latency)
- PA/EC (On Partition, Availability; Else, Consistency)
- PC/EC (On Partition, Consistency; Else, Consistency)

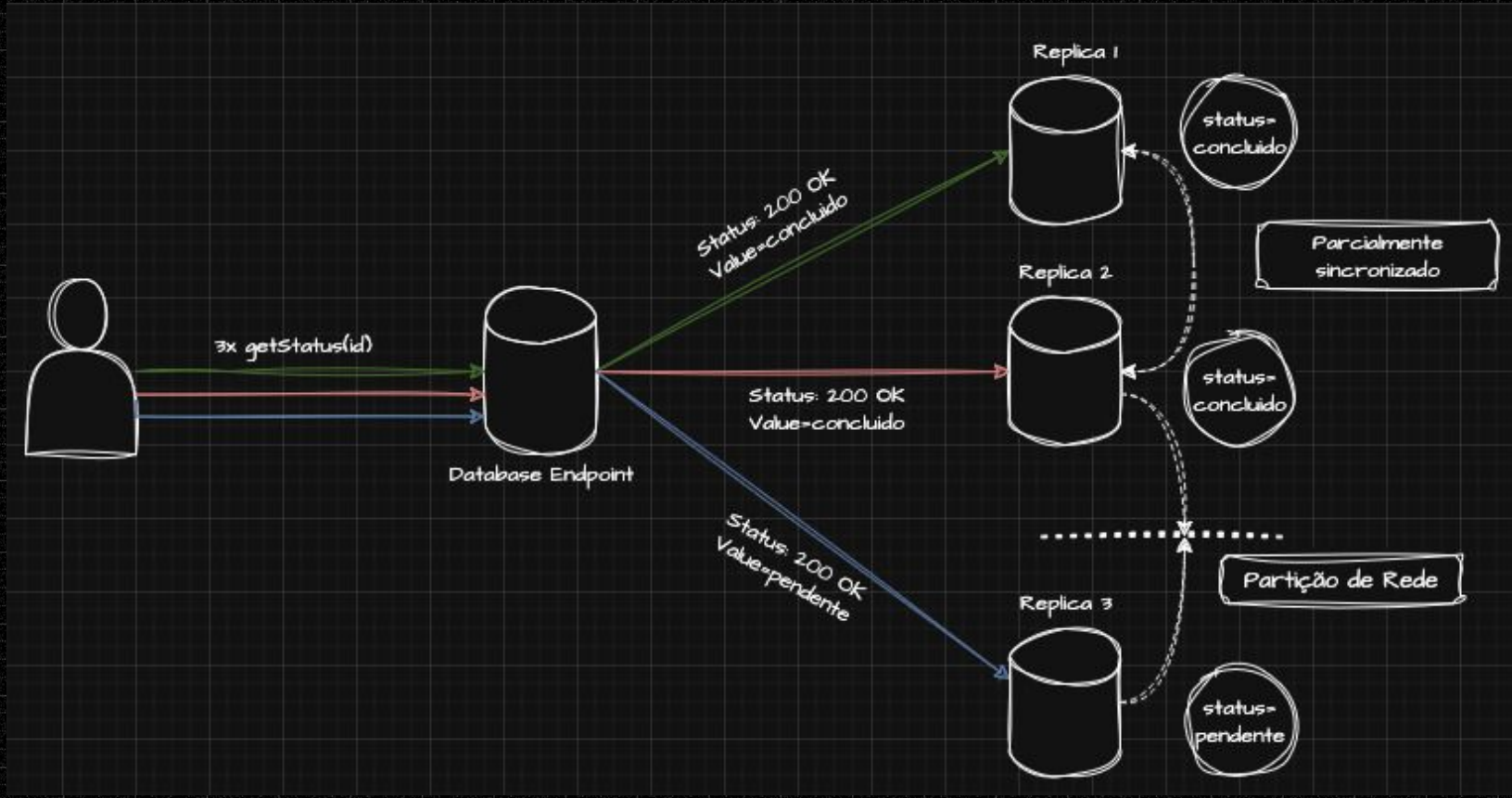
PA/EL (On Partition, Availability; Else, Latency)

- Normal: Prioriza latência ao invés da consistência
- Em partição: Prioriza disponibilidade ao invés de consistência forte
- Modelo de consistência eventual puro
- Alta performance de escrita
- Aceitam versões ligeiramente diferentes dos dados

PA/EL (On Partition, Availability; Else, Latency)



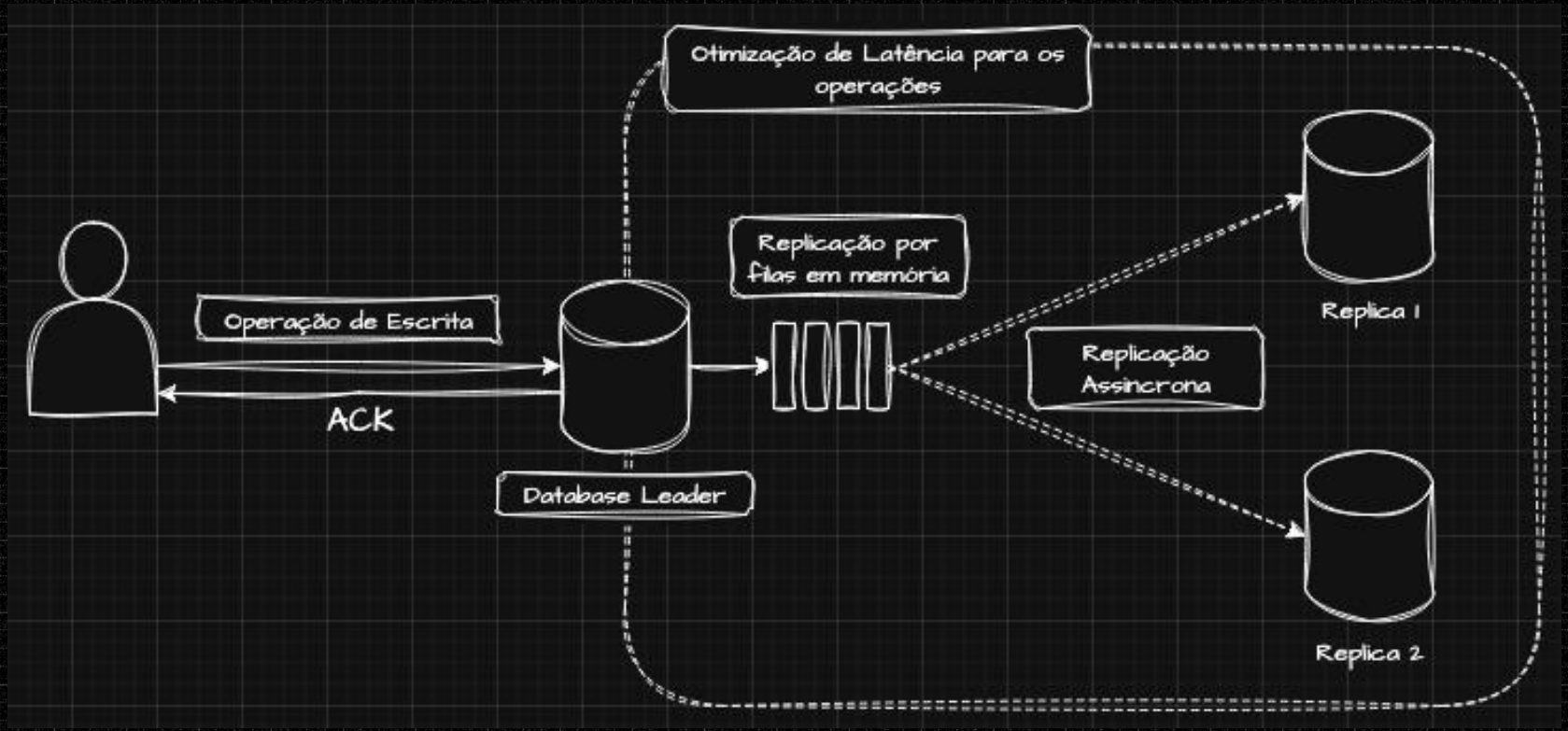
PA/EL (On Partition, Availability; Else, Latency)



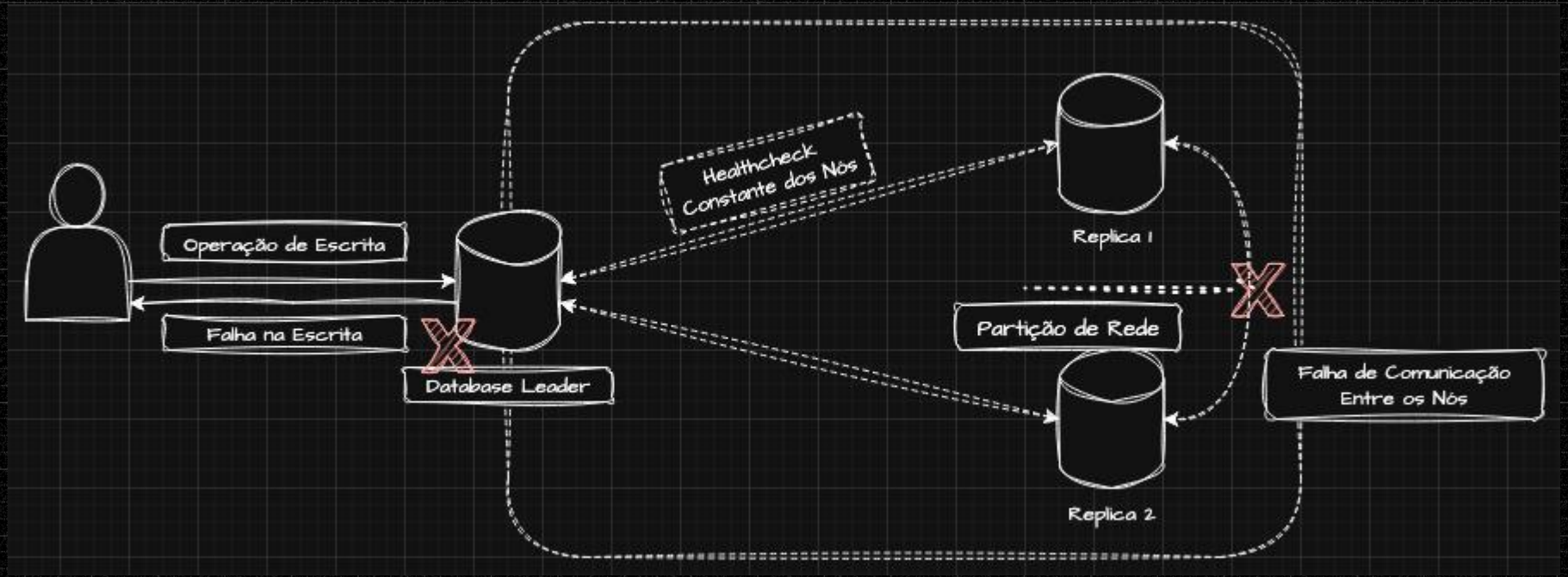
PC/EL (On Partition, Consistency; Else, Latency)

- Normal: Priorizam a latência e o alto throughput ao custo da consistência
- Em partição: Passa a priorizar consistência
- Pode ficar indisponível até que o cluster recupere o consenso e volte a operar
- Consistência mínima durante falhas é inegociável
- Aceita consistência eventual, mas apenas quando todos os nós estão disponíveis

PC/EL (On Partition, Consistency; Else, Latency)



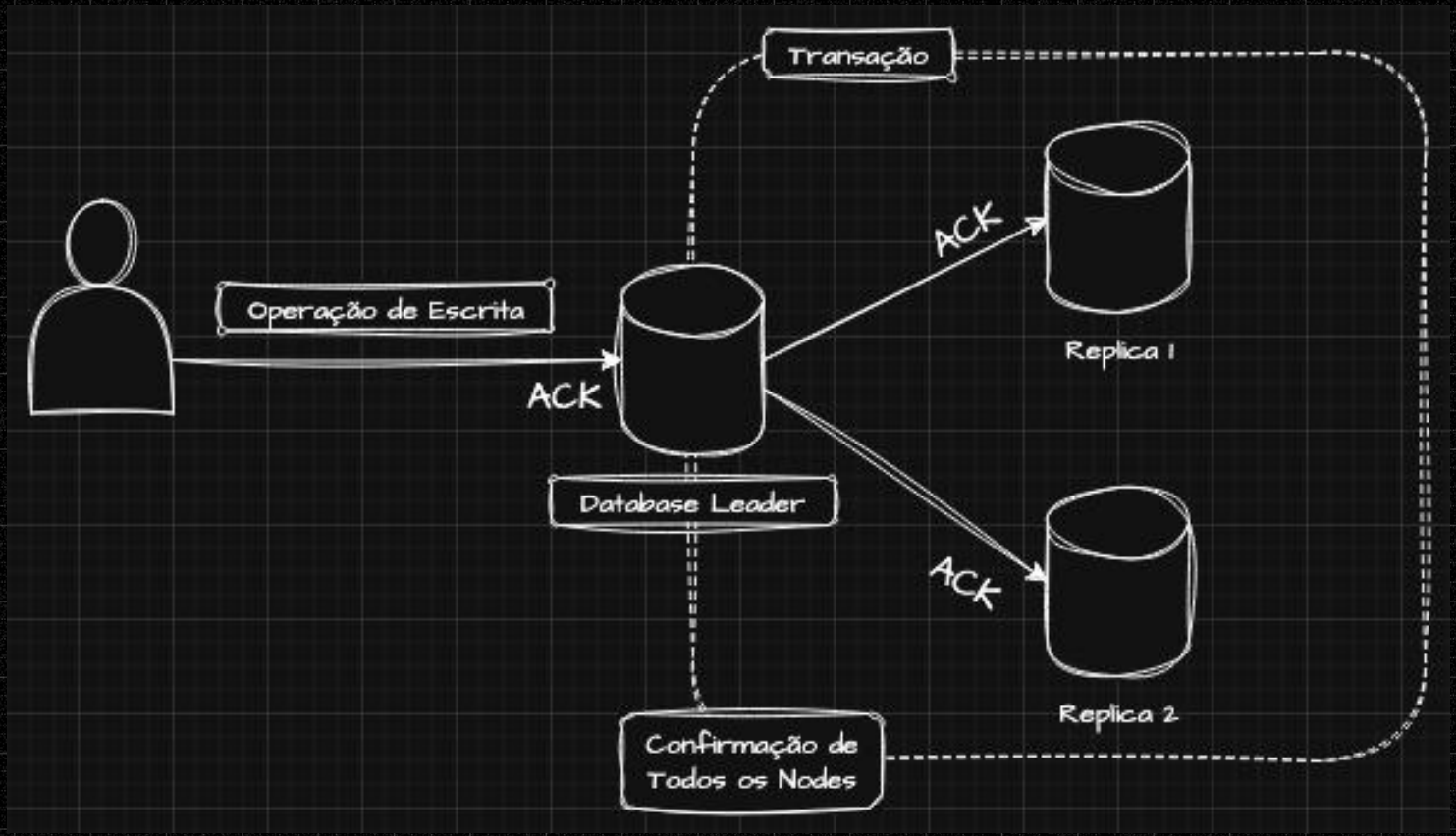
PC/EL (On Partition, Consistency; Else, Latency)



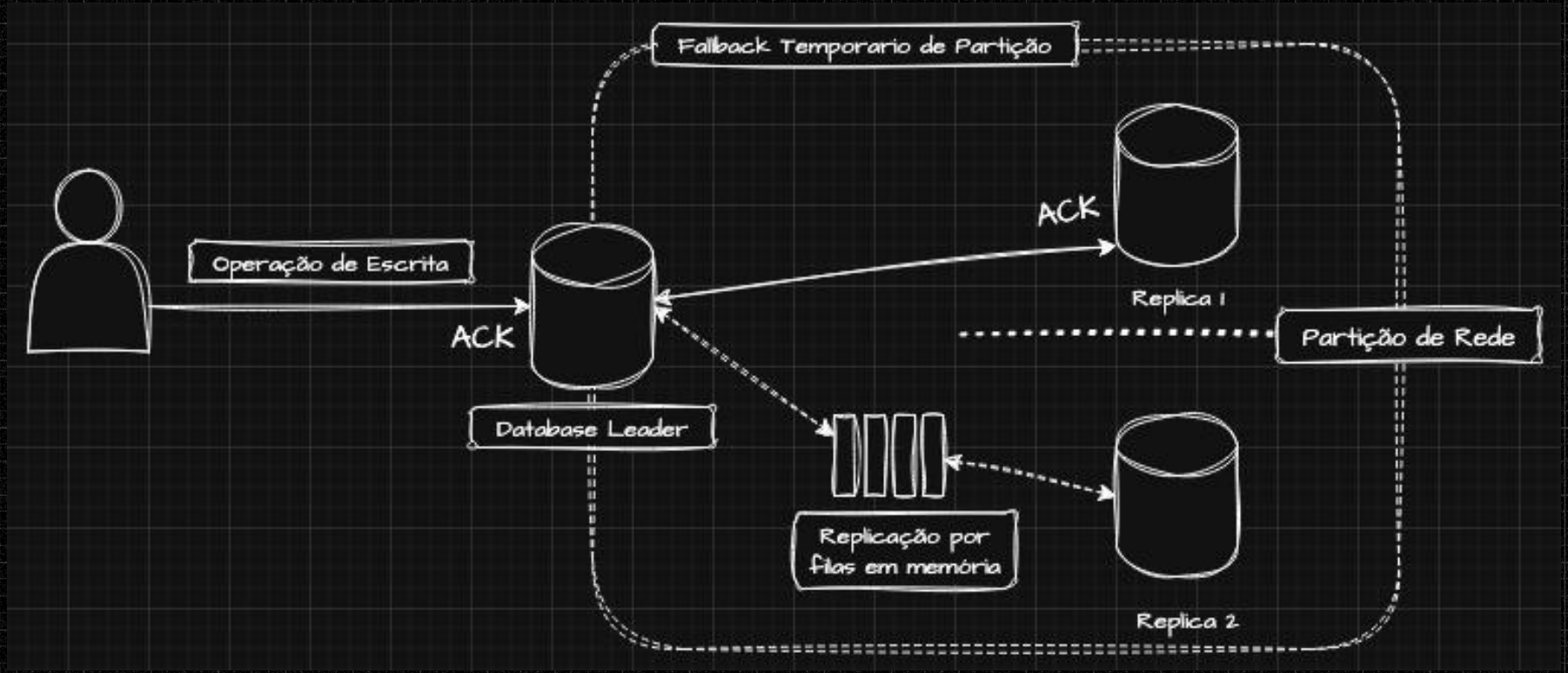
PA/EC (On Partition, Availability; Else, Consistency)

- Normal: Priorizam Consistência Forte
- Em partição: Priorizam Disponibilidade
- Assume consistência eventual em último caso
- CRDT's - Conflict-Free Replicated Data Types
- Contextos híbridos de microserviços

PA/EC (On Partition, Availability; Else, Consistency)



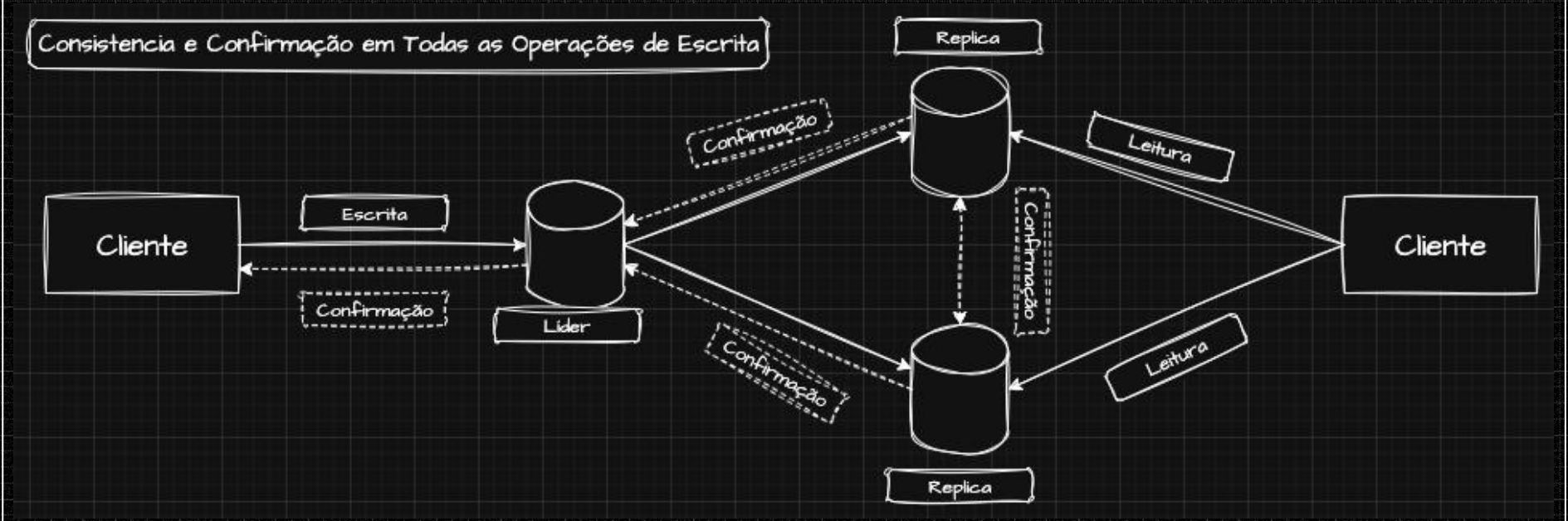
PA/EC (On Partition, Availability; Else, Consistency)



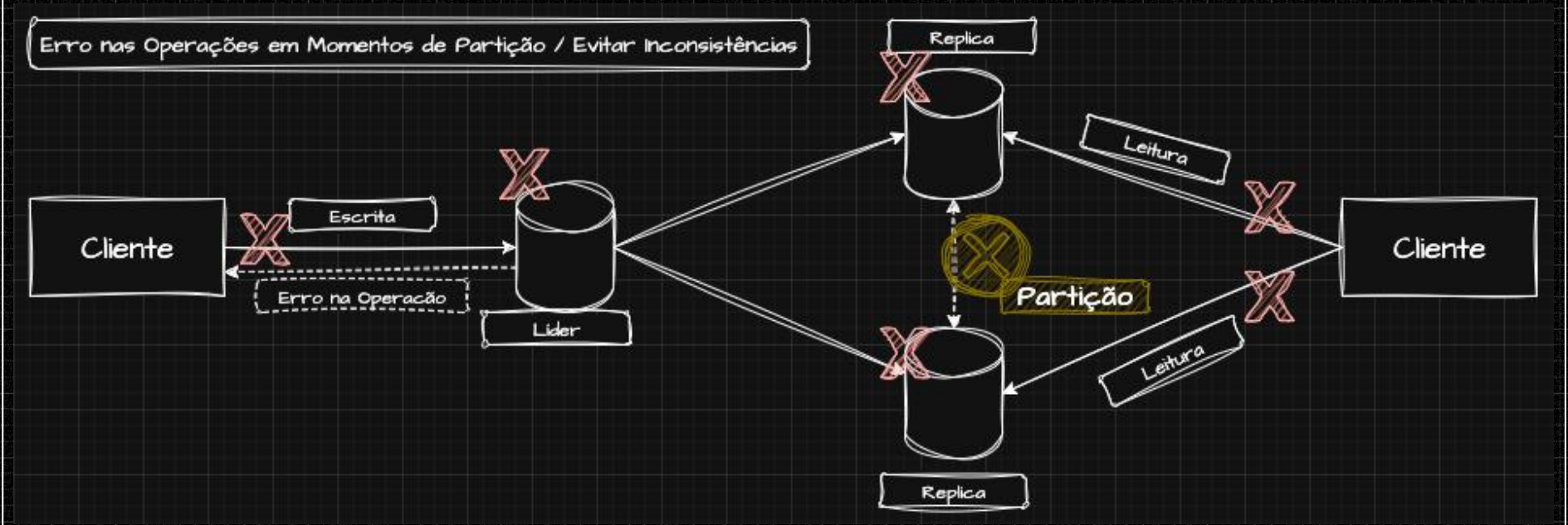
PC/EC (On Partition, Consistency; Else, Consistency)

- Normal: Prioriza Consistência ao invés de latência
- Em partição: Prioriza Consistência ao invés de Disponibilidade.
- Modelo Conservador de consistência
- Precisão do dado é a qualidade mais importante

PC/EC (On Partition, Consistency; Else, Consistency)



PC/EC (On Partition, Consistency; Else, Consistency)



Sistema / Banco de Dados	PAC (durante partição)	ELC (sem partição)	Classificação	Observação
Amazon DynamoDB	**A** (disponibilidade)	**L** (baixa latência, consistência eventual por padrão)	**PA/EL**	Eventual consistency como default, mas suporta "strong reads" opcionais.
Cassandra	**A** (disponibilidade)	**L** (baixa latência, consistência eventual por padrão)	**PA/EL**	Modelo baseado no Dynamo, otimizado para disponibilidade e baixa latência global.
MongoDB	**A** (se configurado com `w=1`) ou **C** (com majority write concern)	**L** (eventual consistency em réplicas secundárias)	**PA/EL** ou **PC/EL**	Flexível; o trade-off depende do write concern e read concern.
Google Spanner	**C** (consistência forte global)	**C** (mesmo sem partição, prioriza consistência)	**PC/EC**	Usa TrueTime para garantir consistência serializável global, com custo de latência.
Azure Cosmos DB	**A** (disponibilidade)	**L/C** (configurável: eventual, bounded staleness, session, consistent prefix, strong)	**PA/ELC**	Oferece 5 níveis de consistência configuráveis.
Apache Kafka	**A** (disponibilidade)	**L** (prioriza throughput e baixa latência)	**PA/EL**	Garantias de consistência são fracas; foco em disponibilidade e velocidade.
Etcd	**C** (consistência forte)	**C** (consistência forte)	**PC/EC**	Voltado para consistência forte, usado em sistemas críticos de coordenação.

SlidesCarnival icons are editable shapes.

This means that you can:

- Resize them without losing quality.
- Change fill color and opacity.

Isn’t that nice? :)

Examples: