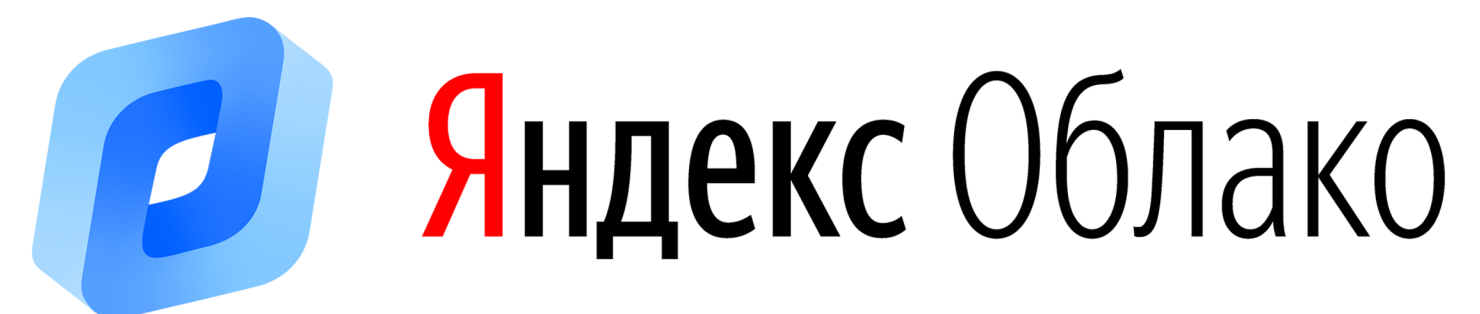


Яндекс



Распределённые транзакции в Yandex Database

Семён Чечеринда, ведущий разработчик

О чём я расскажу

- 01 | Какие распределённые транзакции нам нужны
- 02 | Как мы реализовали распределённые транзакции в YDB
- 03 | Как масштабируется наше решение
- 04 | Каков overhead у запросов в нашем решении
- 05 | Доступные уровни изоляции для транзакций

01

**Какие распределённые
транзакции нам нужны**

Зачем?



- › Высокие требования внутри Яндекса к отказоустойчивости сервисов
- › Сервисы Яндекса должны переживать выход из строя одного из ДЦ
- › Опыт разработки и эксплуатации своей кросс-ДЦ noSQL БД
- › Eventual consistency не подходит под наши задачи

ACID



- › Atomicity
- › Consistency
- › Isolation
- › Durability

ACID



- › Atomicity
- › Consistency
- › Isolation
- › Durability

ACID



- › Atomicity
- › Consistency
- › Isolation
- › Durability

ACID



- › Atomicity
- › Consistency
- › Isolation
- › Durability

ACID



- › Atomicity
- › Consistency
- › Isolation
- › Durability

Isolation levels

Anomalies

	Dirty read	Unrepeatable read	Lost updates	Phantoms	Write skew
Read uncommitted	✓	✓	✓	✓	✓
Read committed	✗	✓	✓	✓	✓
Repeatable read	✗	✗	✗	✓	✓
Snapshot isolation	✗	✗	✗	✗	✓
Serializable	✗	✗	✗	✗	✗

Strict Serializable

Serializable + Linearizable

Isolation levels

Anomalies

	Dirty read	Unrepeatable read	Lost updates	Phantoms	Write skew
Read uncommitted	✓	✓	✓	✓	✓
Read committed	✗	✓	✓	✓	✓
Repeatable read	✗	✗	✗	✓	✓
Snapshot isolation	✗	✗	✗	✗	✓
Serializable	✗	✗	✗	✗	✗
Strict Serializable	Serializable + Linearizable				

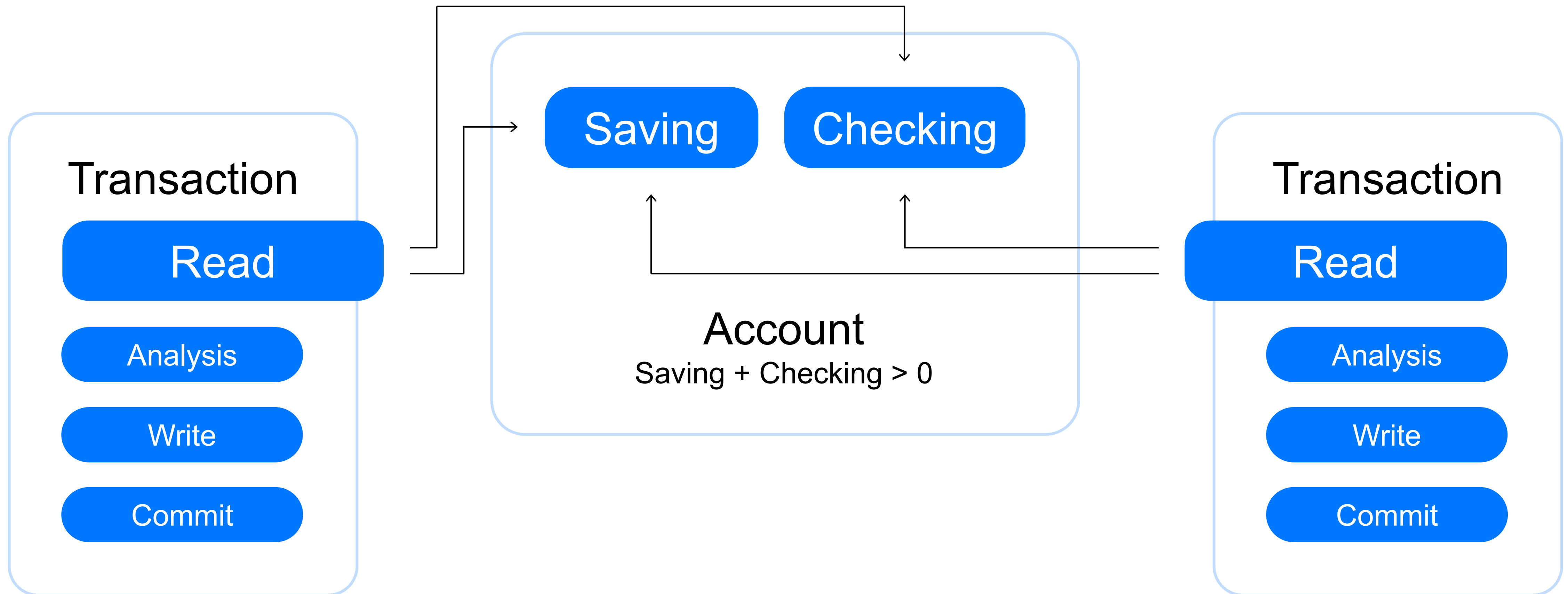
Isolation levels

Anomalies

	Dirty read	Unrepeatable read	Lost updates	Phantoms	Write skew
Read uncommitted	✓	✓	✓	✓	✓
Read committed	✗	✓	✓	✓	✓
Repeatable read	✗	✗	✗	✓	✓
Snapshot isolation	✗	✗	✗	✗	✓
Serializable	✗	✗	✗	✗	✗
Strict Serializable	Serializable + Linearizable				

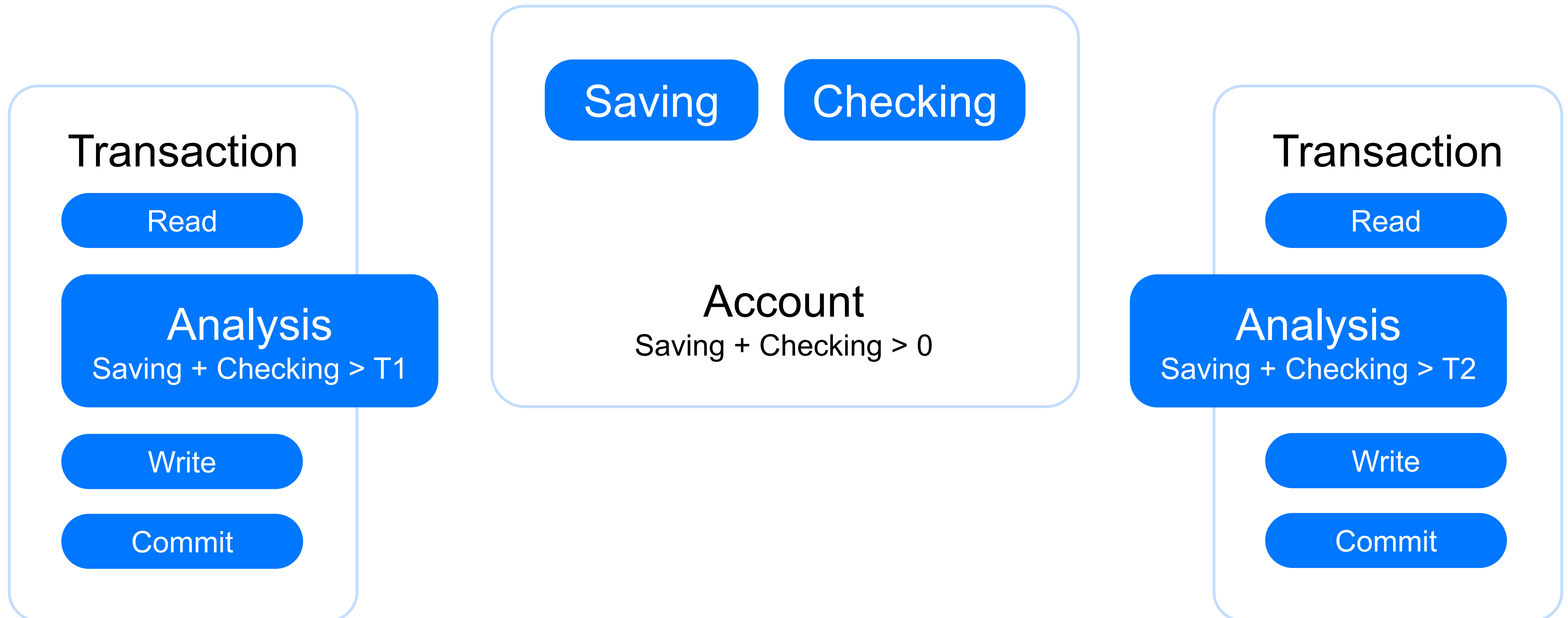
Anomalous behavior

Write skews anomalies are allowed by snapshot isolation level



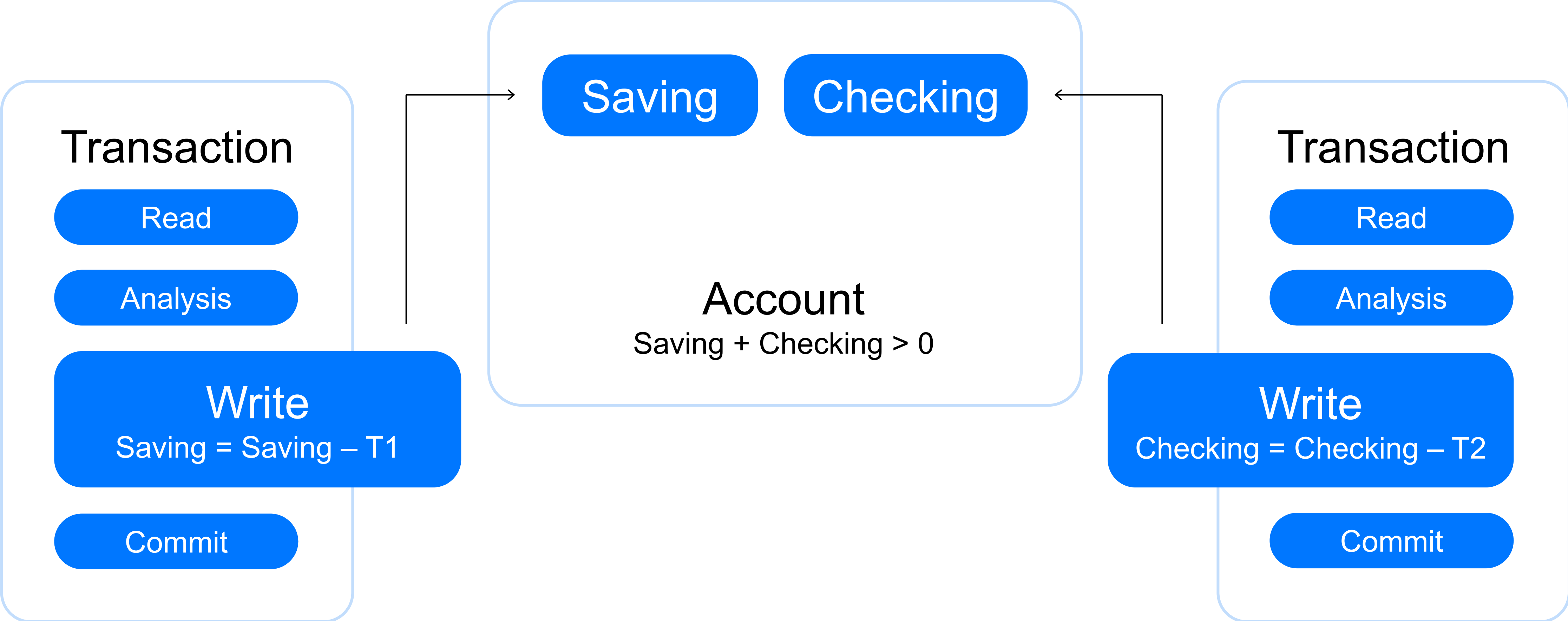
Anomalous behavior

Write skews anomalies are allowed by snapshot isolation level



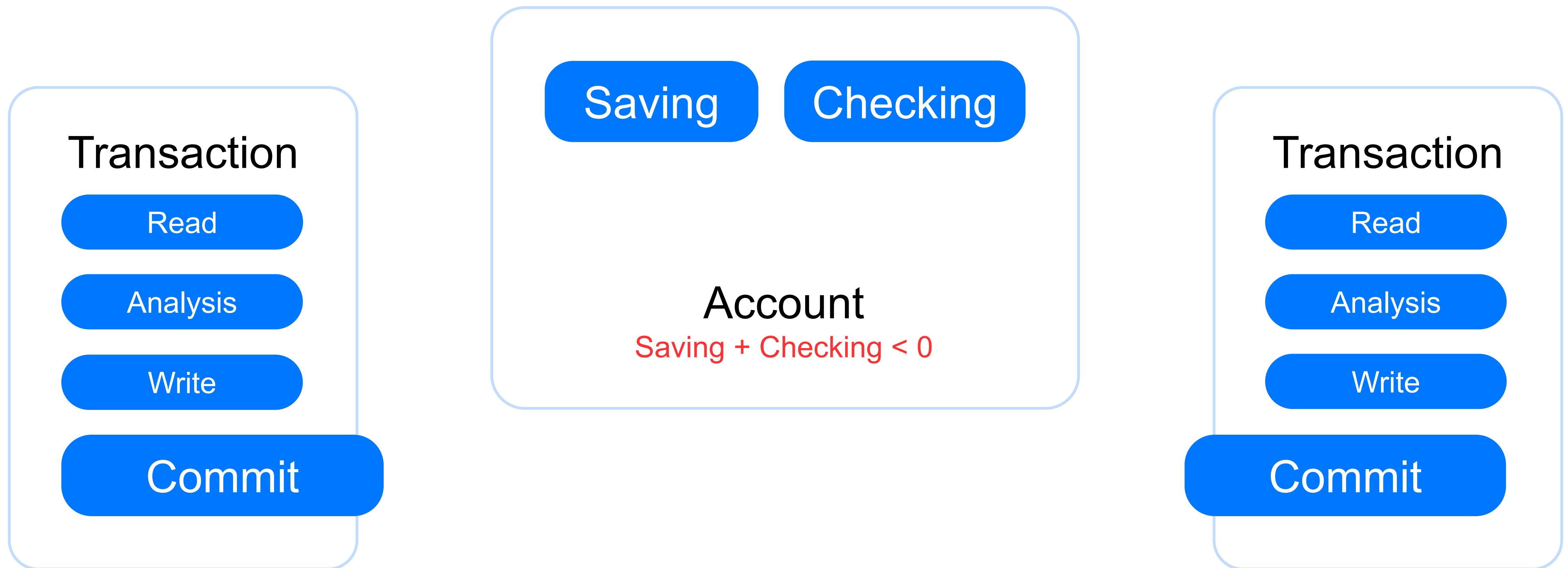
Anomalous behavior

Write skews anomalies are allowed by snapshot isolation level



Anomalous behavior

Write skews anomalies are allowed by snapshot isolation level



Anomalous behavior

Snapshot isolation

- › Обе транзакции выполняются
- › Анализ данных в каждой из транзакций не покажет нарушение инварианта
- › Инвариант нарушится после конкурентного выполнения транзакций

Serializable isolation

- › Транзакции выполняются последовательно
- › Анализ данных в каждой из транзакций будет корректным
- › Инвариант не нарушится

Yandex Database (YDB)

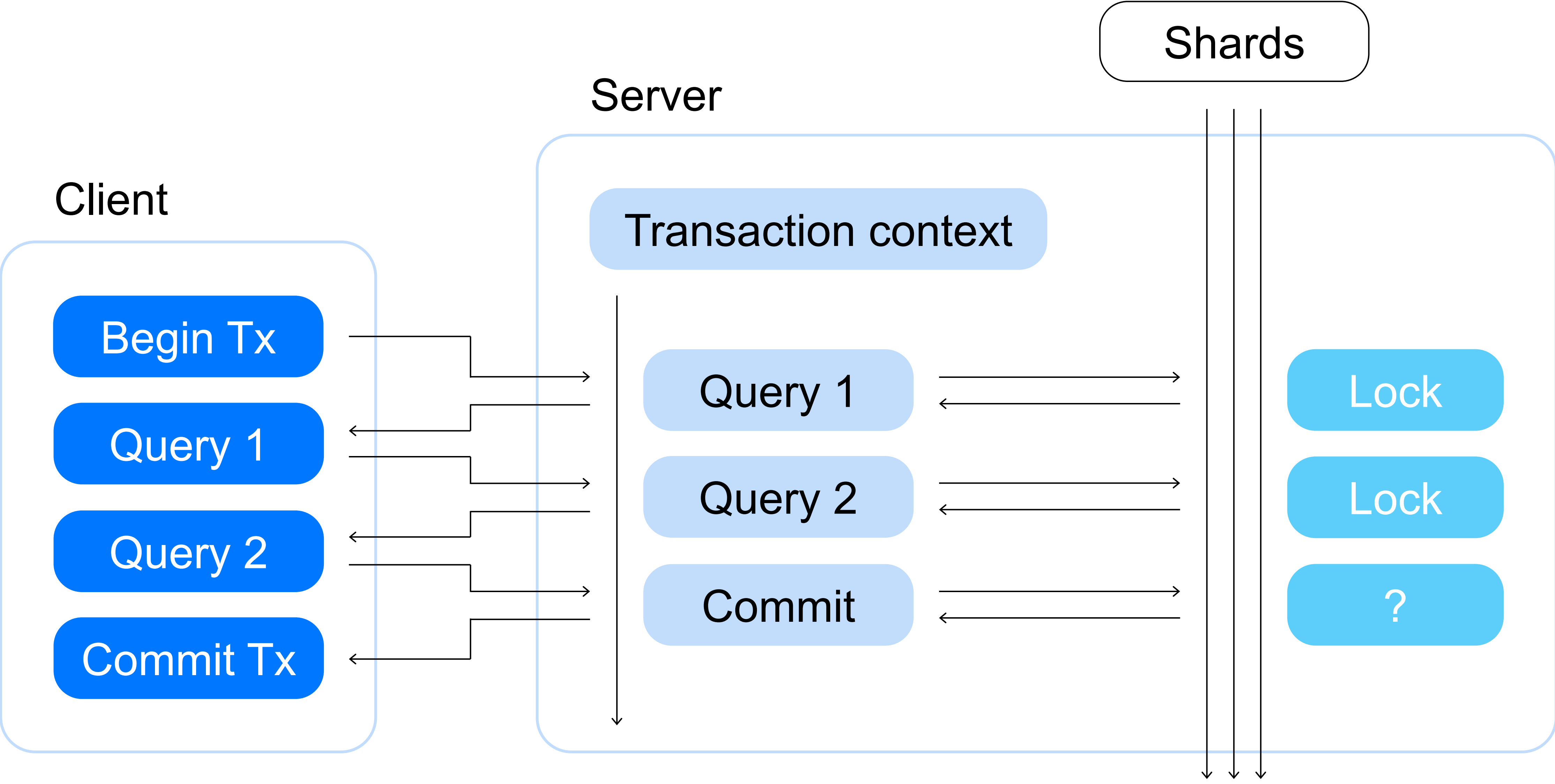


- › Надёжное хранение данных с избыточностью и автоматической репликацией
- › Отказоустойчивость, автоматическое восстановление от сбоев
- › Распределённые ACID-транзакции с serializable-уровнем изоляции транзакций
- › Горизонтальная масштабируемость до тысяч нод

02

Как мы реализовали распределённые транзакции в YDB

Общий вид

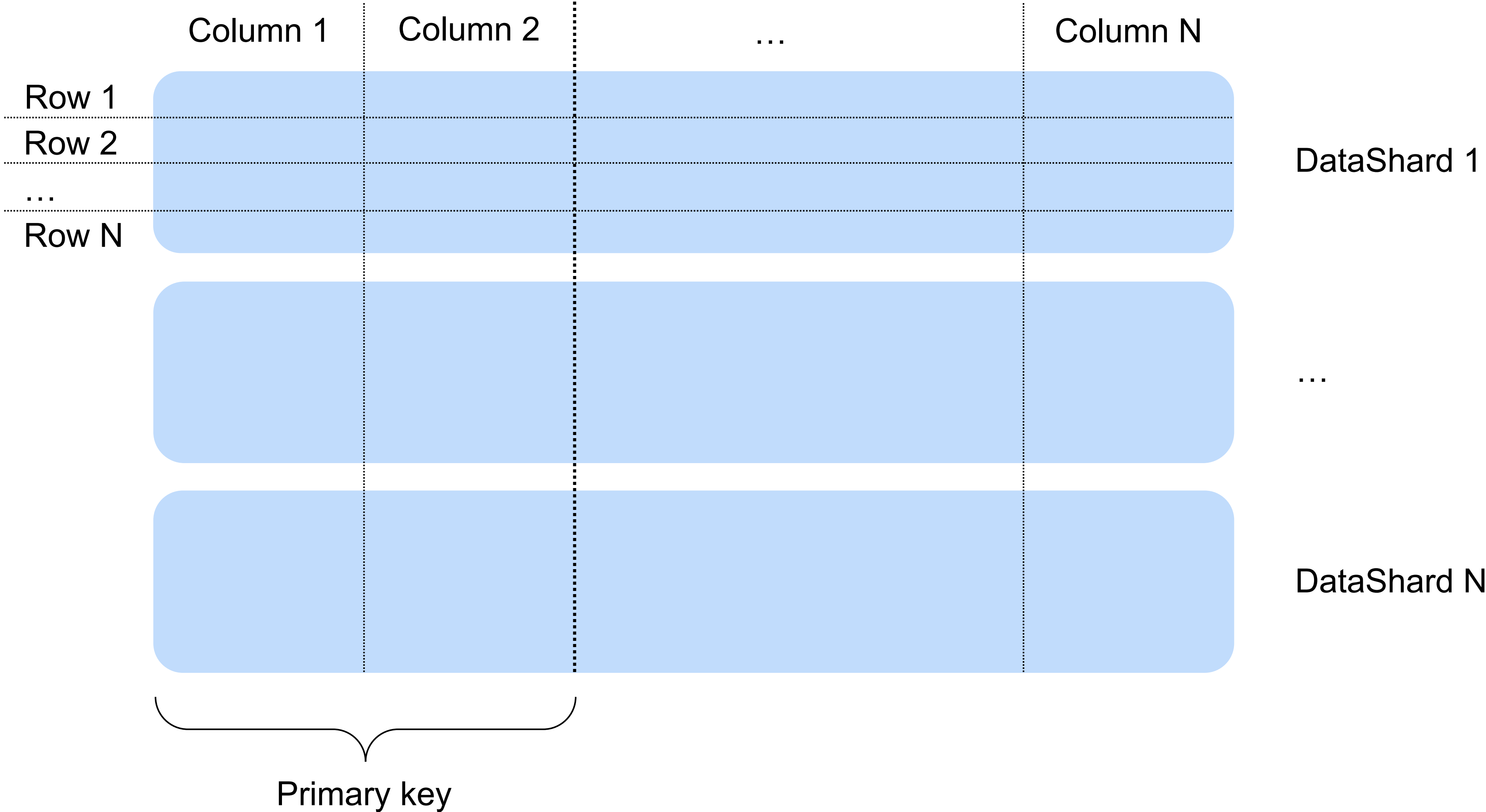


Как мы реализовали распределённые транзакции в YDB



- › Кто участвует в распределённых транзакциях
- › Свойства и ограничения атомарного распределённого запроса
- › Конкурентное выполнение распределённых транзакций

Таблицы



YDB Tablet



- › Является актором, но имеет персистентное состояние
- › Работает однопоточно
- › Хранит свое состояние и данные в Distributed Storage
- › Единица отказоустойчивости системы
- › Всегда работает только один экземпляр
- › Поднимается на любой ноде кластера
- › На кластере ~100K tablet

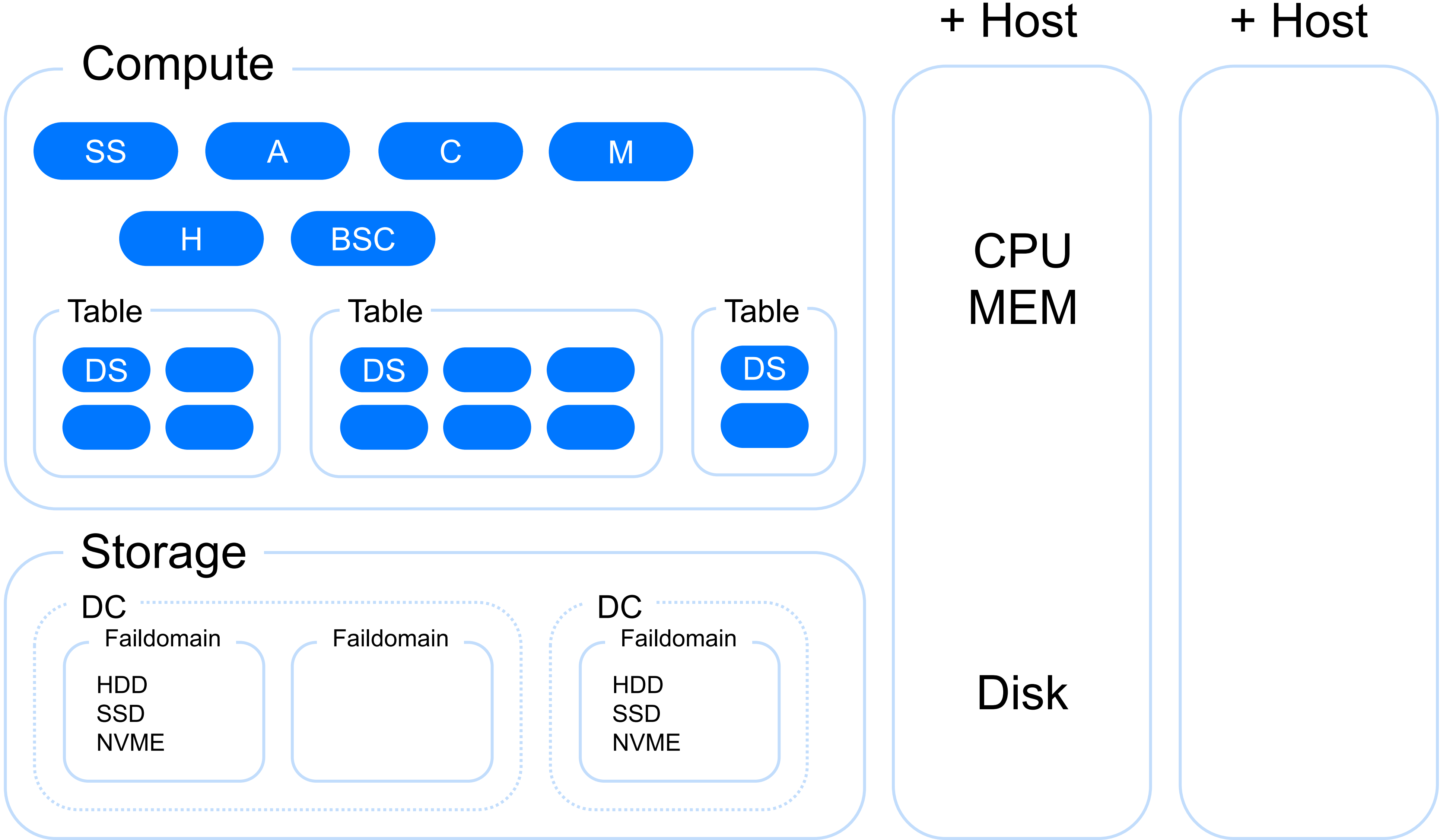
Tablet — универсальный строительный блок системы



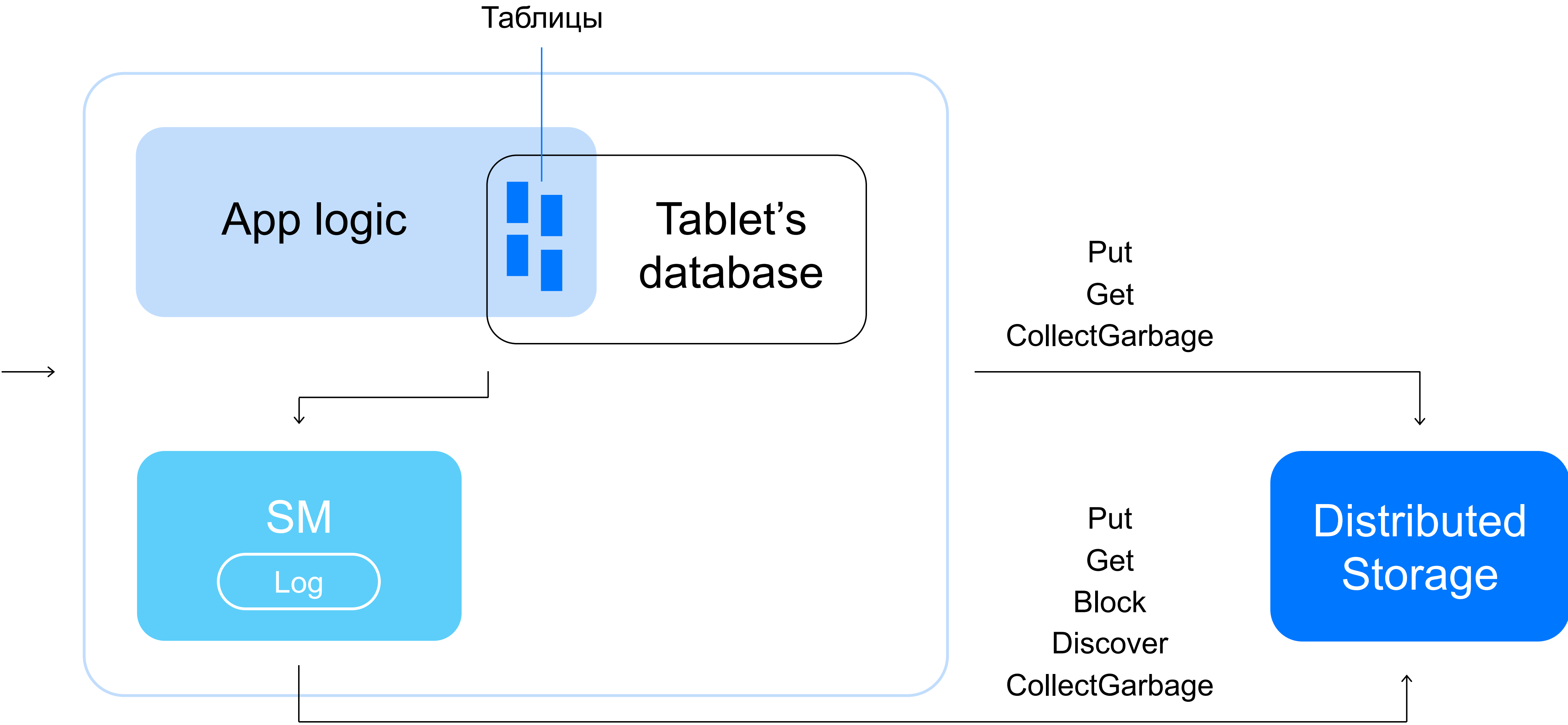
С использованием tablet построены разные логические сущности

- › Пользовательские таблицы
- › Очереди
- › Хранилище time-series данных мониторинга
- › Системные tablet, необходимые для работы YDB

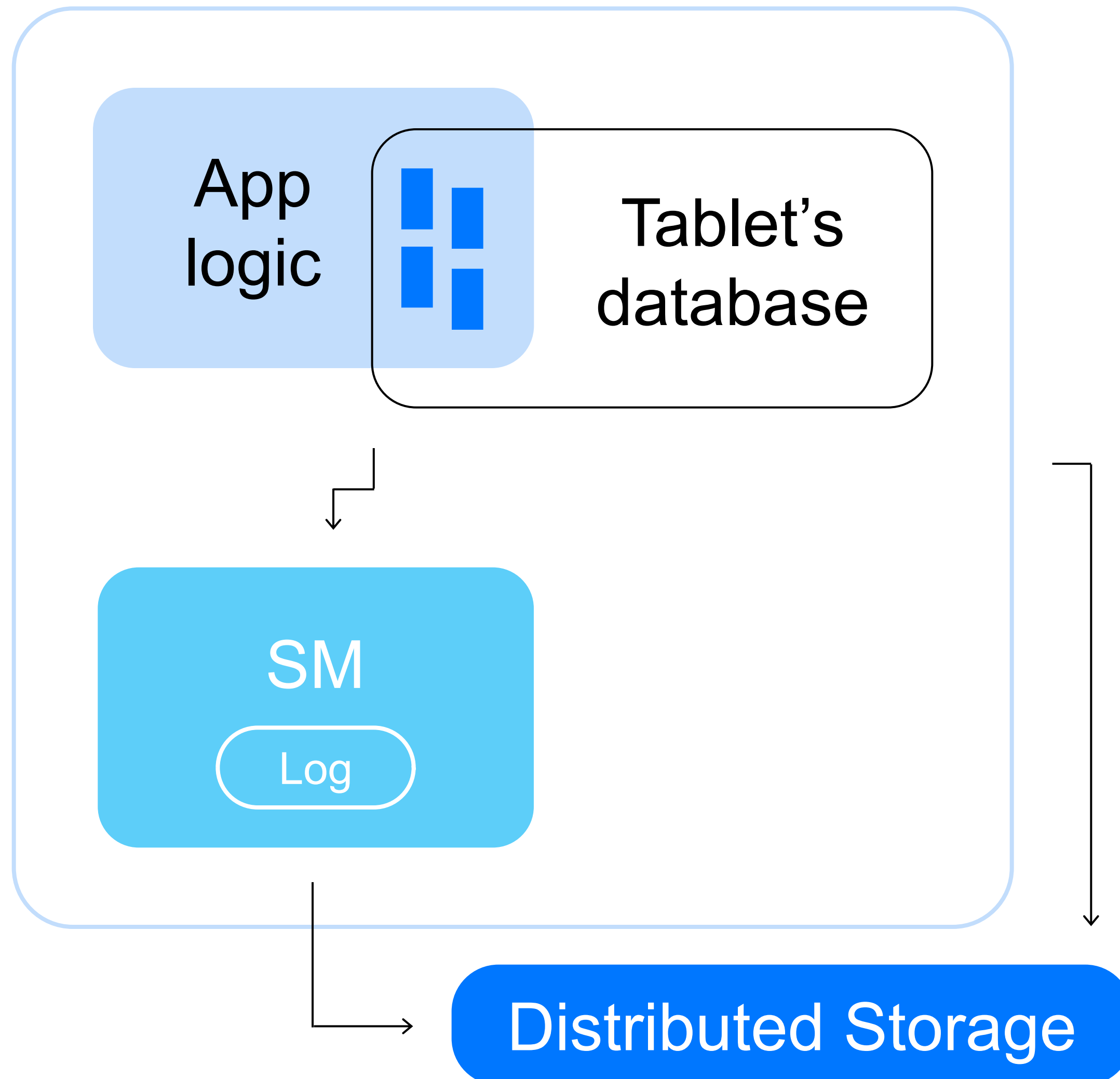
Overview



YDB Tablet



Datashard как tablet



- › Надёжность и доступность — это свойства таблетки
- › Транзакционное изменение СВОИХ ДАННЫХ
 - Однопоточная, strict serializable
 - Одношардовые запросы
- › ПрIMITIV масштабируемости таблицы
 - Количество шардов в таблице меняется в зависимости от её размера и нагрузки

Атомарный распределённый запрос



- › Один запрос к системе
- › Выполнение транзакции детерминировано
- › Известны read / write set'ы ключей транзакции

Конкурентное выполнение

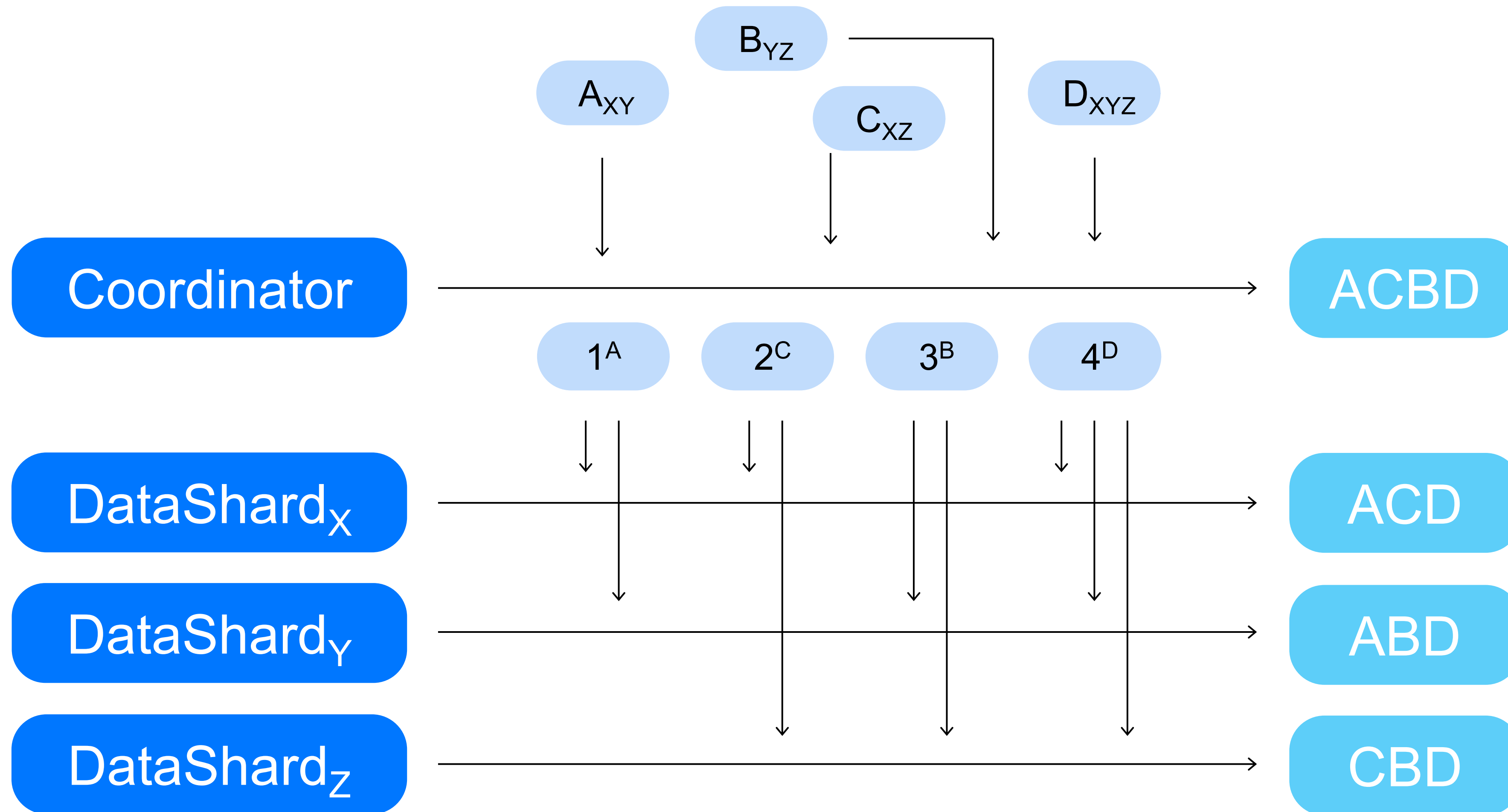
Calvin: Fast Distributed Transactions for Partitioned Database Systems by Daniel J. Abadi, Alexander Thomson

Если каждая транзакция выполняется детерминировано, то их упорядоченное выполнение также детерминировано

Детерминированное выполнение распределённых транзакций

- › Определить глобальный порядок выполнения транзакций
- › Выполнить на шардах релевантные им транзакции в соответствии с их глобальным порядком выполнения

Координация транзакции

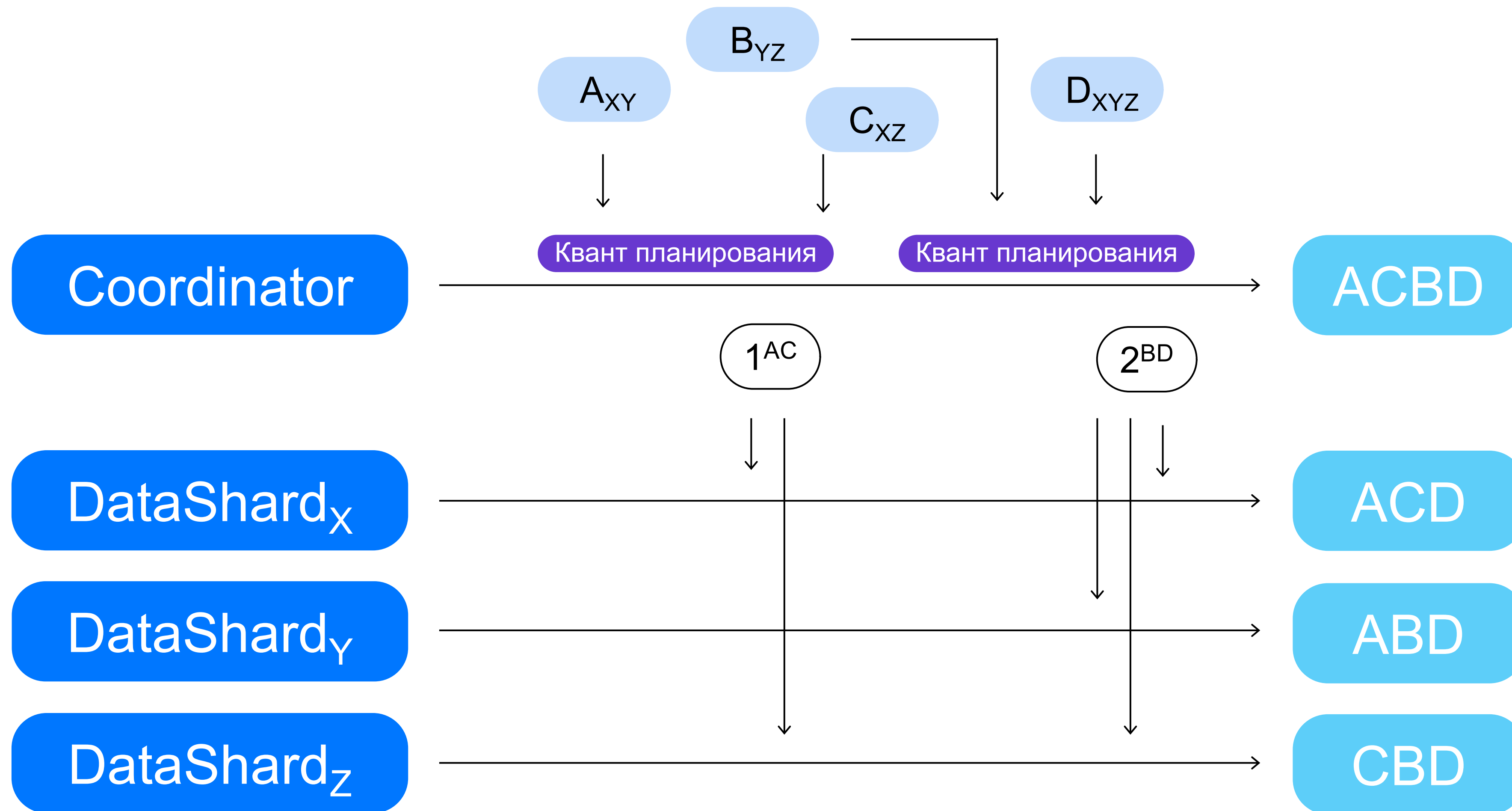


A, B, C, D — транзакции в системе
X, Y, Z — даташарды таблиц
1, 2, 3, 4 — порядок выполнения транзакций

03

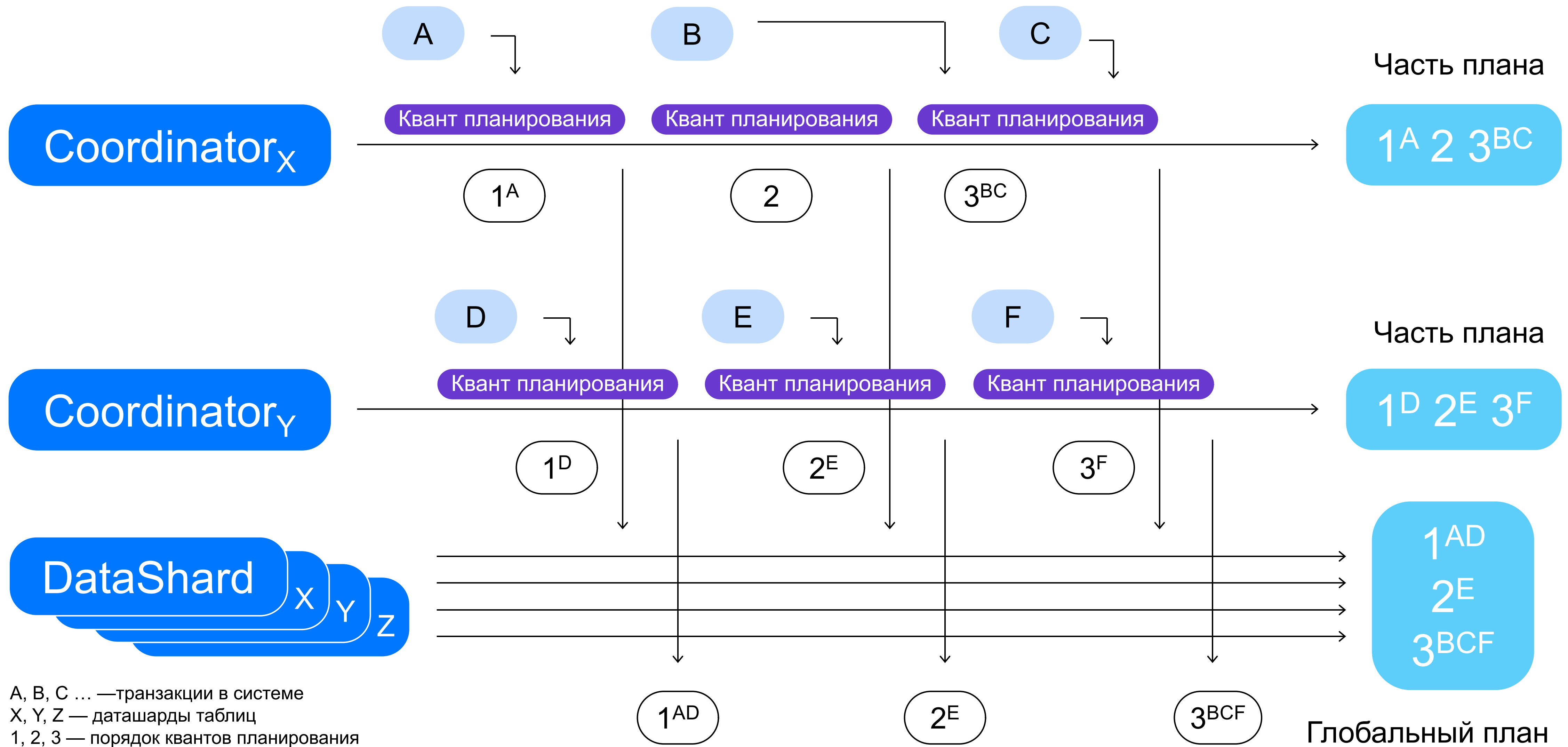
Как масштабировать наше решение

Кванты планирования



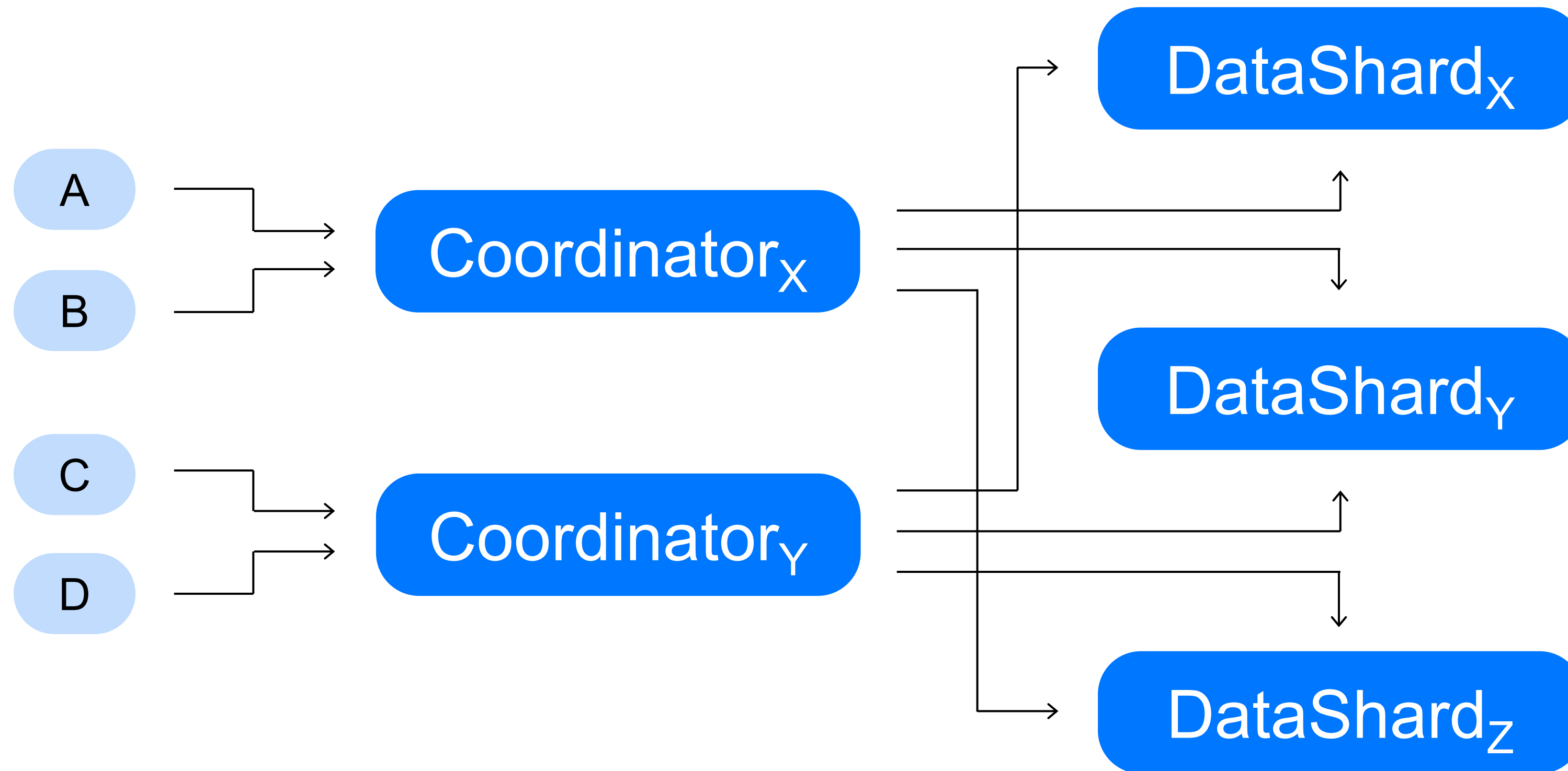
A, B, C, D — транзакции в системе
X, Y, Z — даташарды таблиц
1, 2, 3, 4 — порядок выполнения транзакций

Шардированный координатор

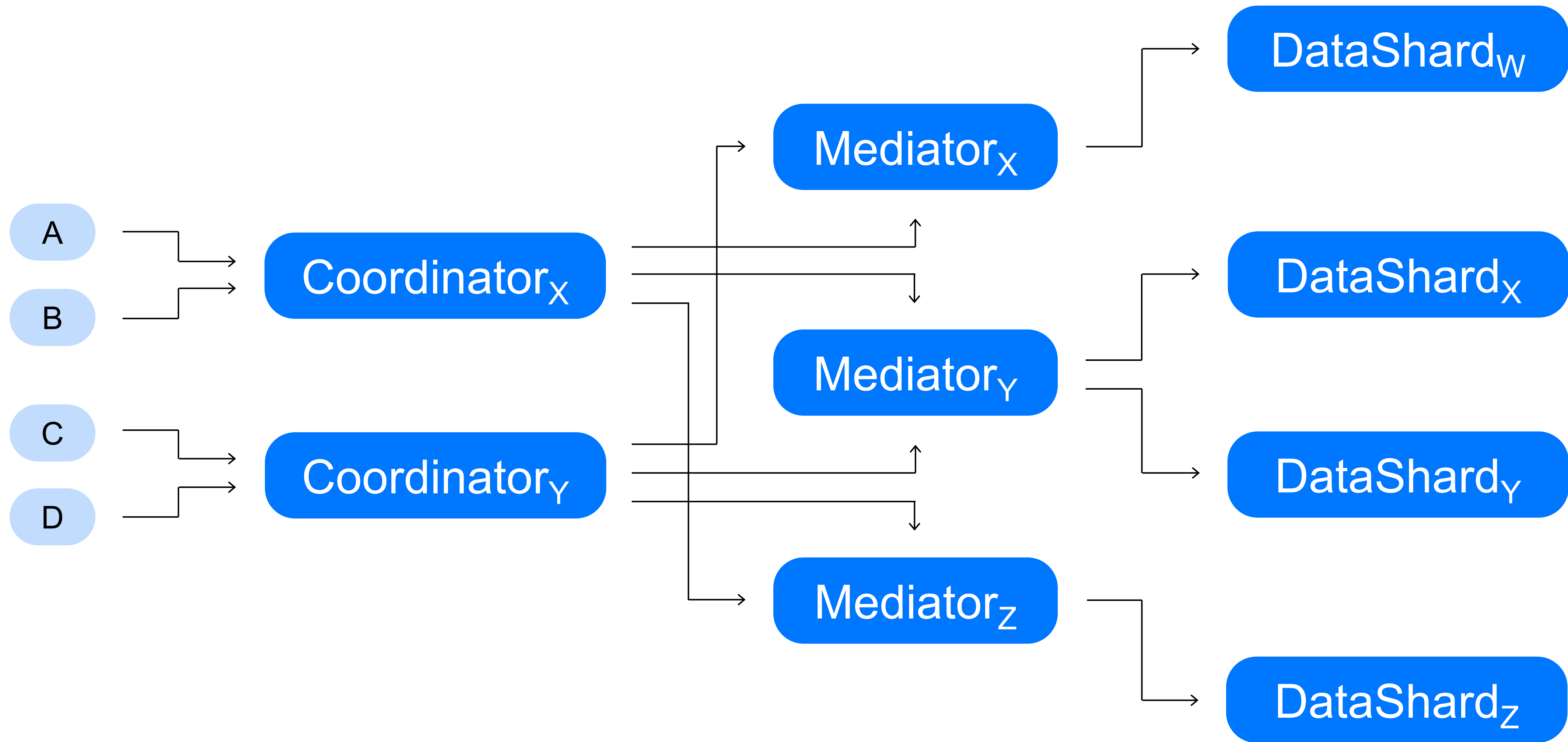


A, B, C ... —транзакции в системе
X, Y, Z — даташарды таблиц
1, 2, 3 — порядок квантов планирования

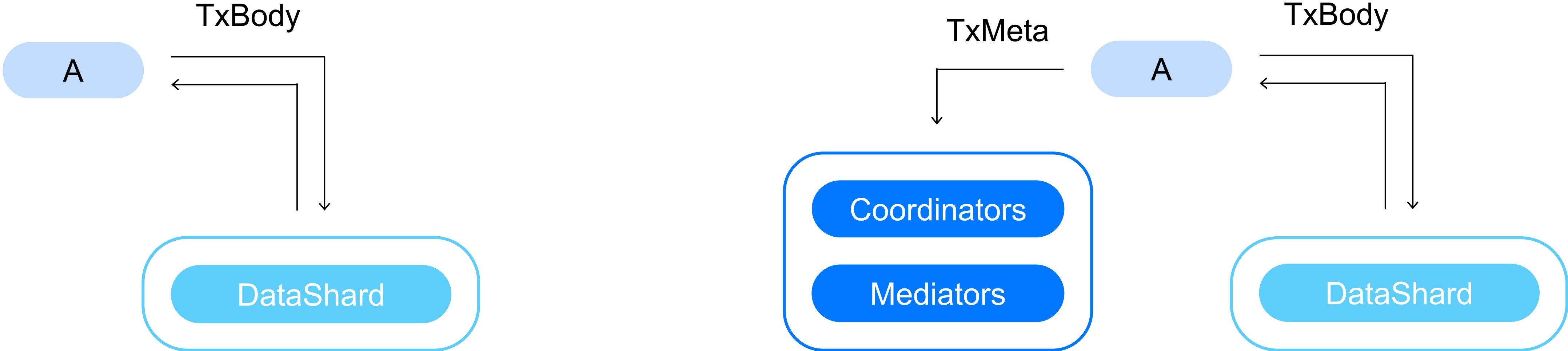
Количество связей Coordinator–DataShard



Медиатор



Разделить data и meta-информацию



04

**Каков overhead запросов
у нашего решения**

Overhead



Игнорируем все, что связано с самим запросом, считаем только overhead

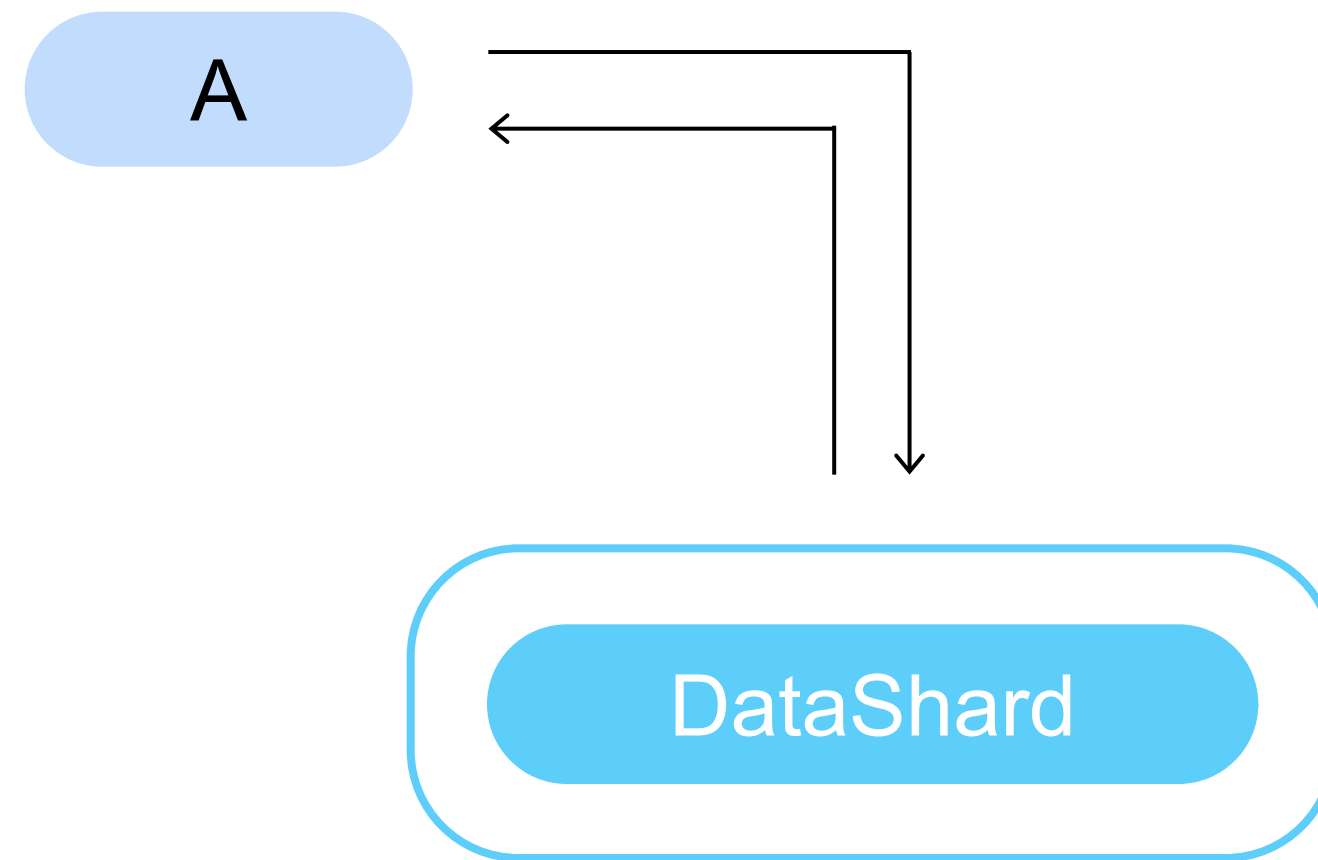
- › Все данные в памяти, чтение бесплатно
- › Отправка сообщения 0.5 RTT
- › Write \sim 1 RTT + diskIO
- › Одновременно выполняющиеся запросы считаем как один

Классы запросов



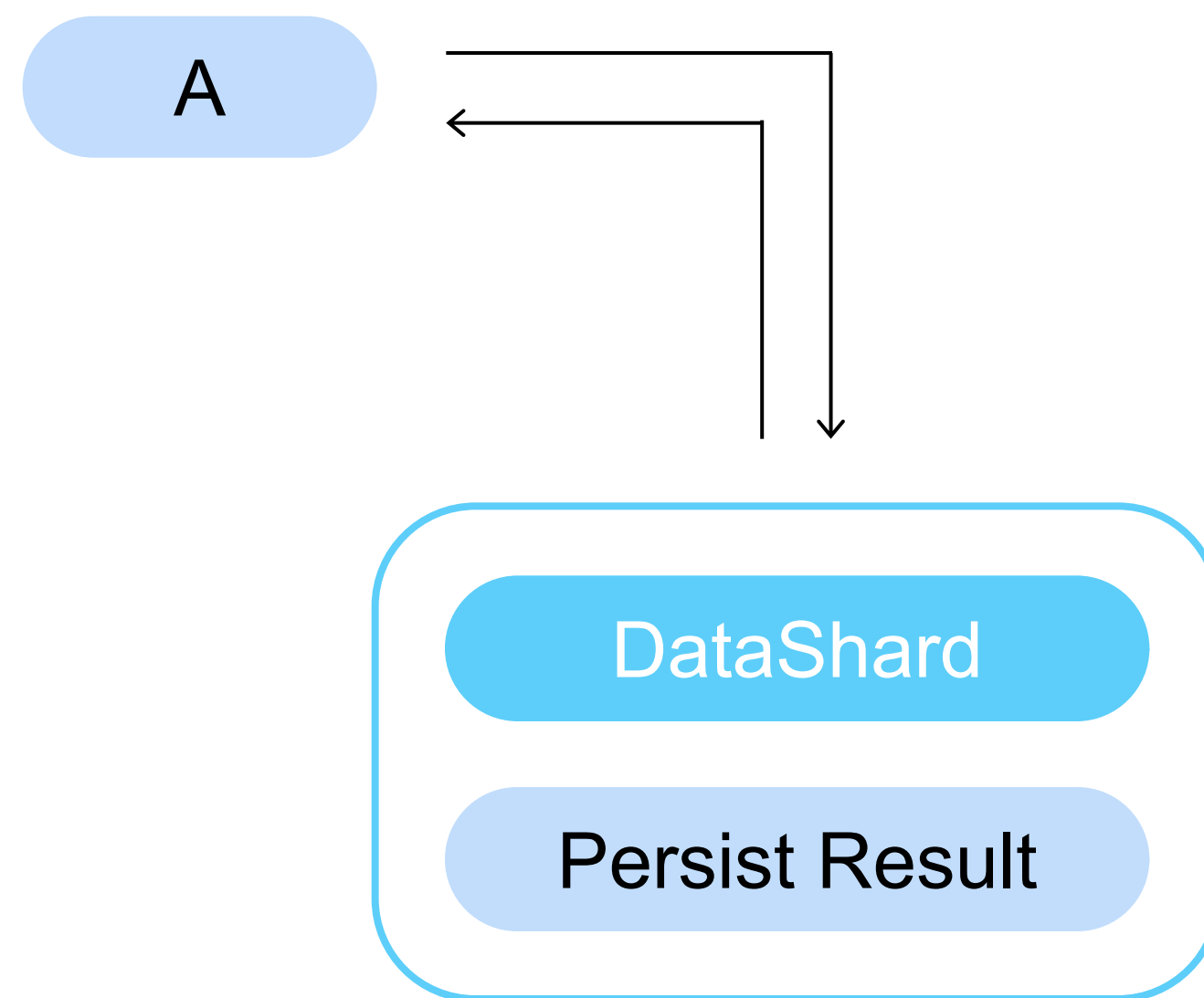
1. Read only, one shard — RO Immediate
2. Write only, one shard — WO Immediate
3. Read only / write only, multi shard — RO/WO
4. Read write, multi shard — RW

Транзакция RO immediate



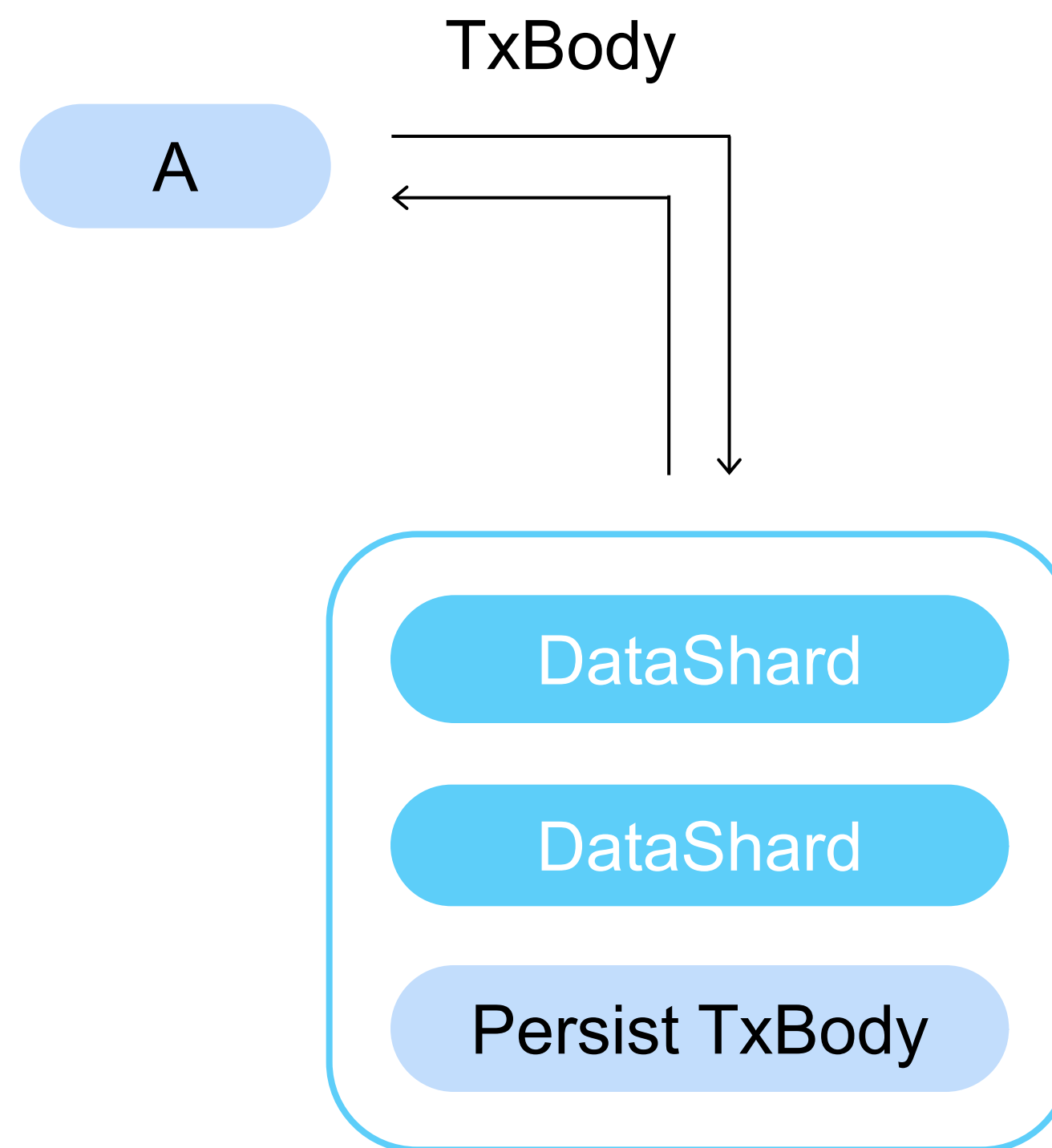
| Итого
> 1 RTT

Транзакция WO immediate



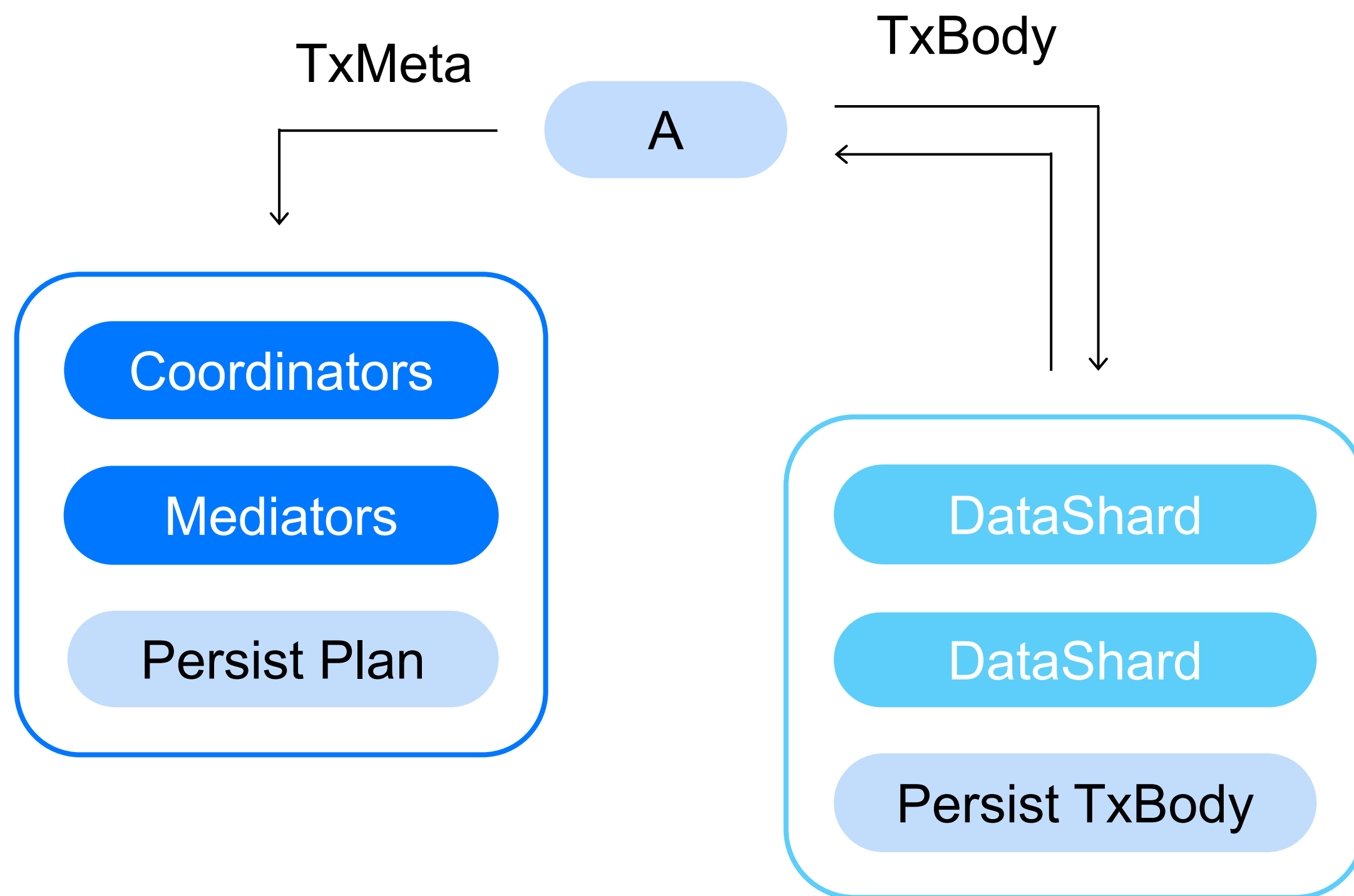
- | Итого
- > 1 RTT
- > 1 write

Транзакция RO / WO



- > 1 RTT
- > 1 write

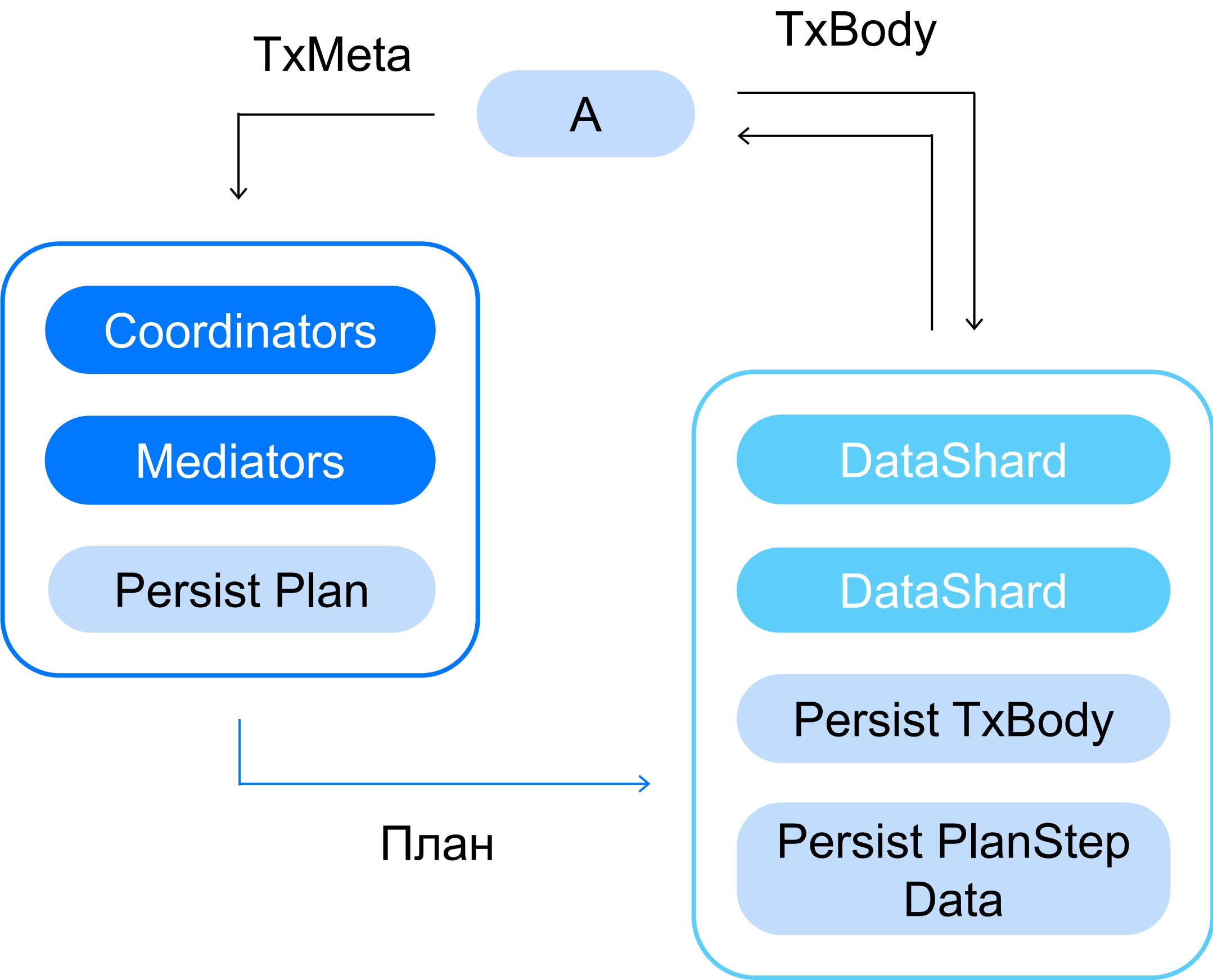
Транзакция RO / WO



- > 1 RTT
- > 1 write

- > 0.5 RTT
- > 1 plan batch time
- > 1 write
- > 0.5 RTT

Транзакция RO / WO



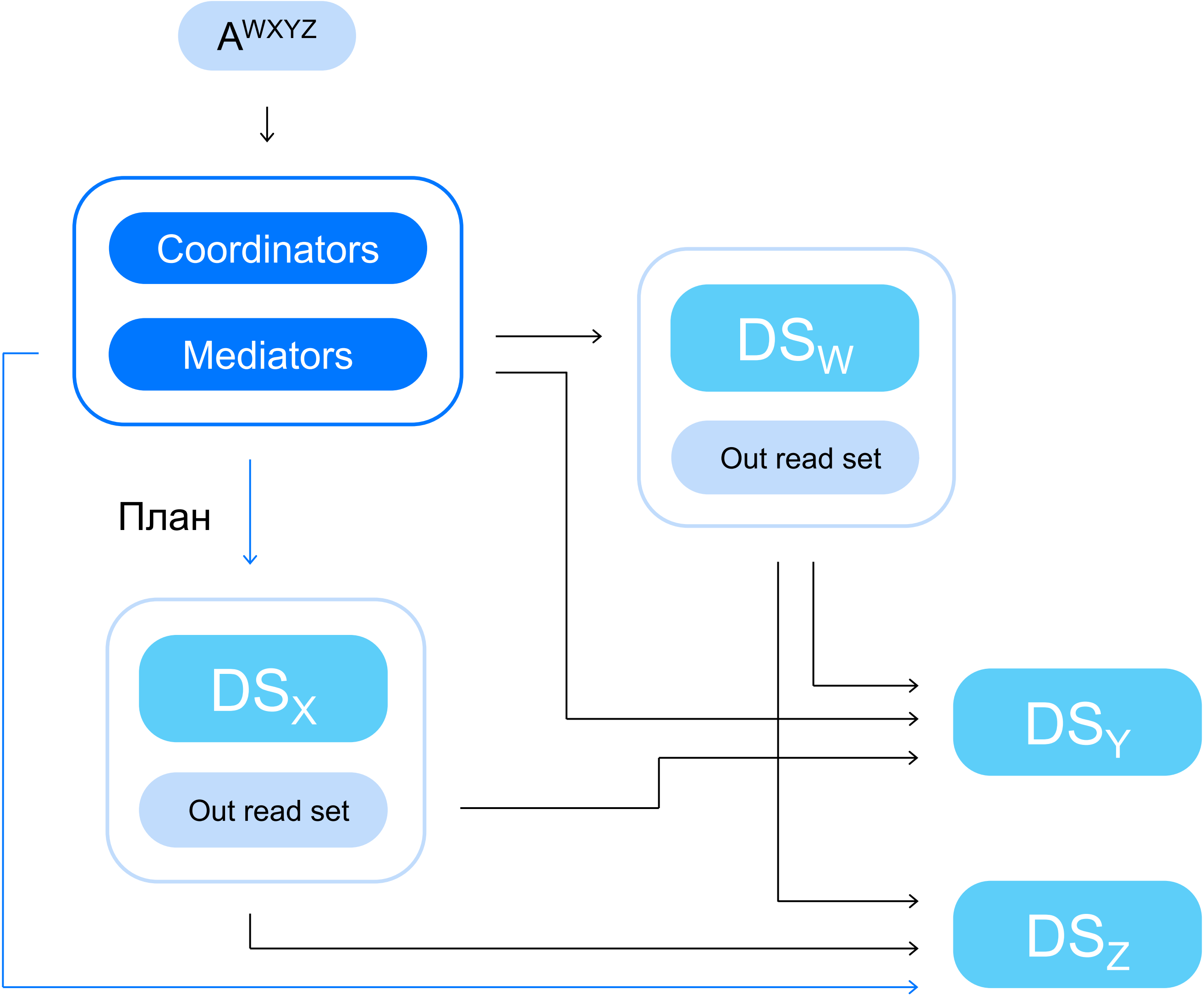
- > 1 RTT
- > 1 write

- > 0.5 RTT
- > 1 plan batch time
- > 1 write
- > 0.5 RTT

- > 0.5 RTT
- > 1 write
- > 0.5 RTT

- Итого**
- > 6 RTT + 1 plan batch time + 3 diskIO

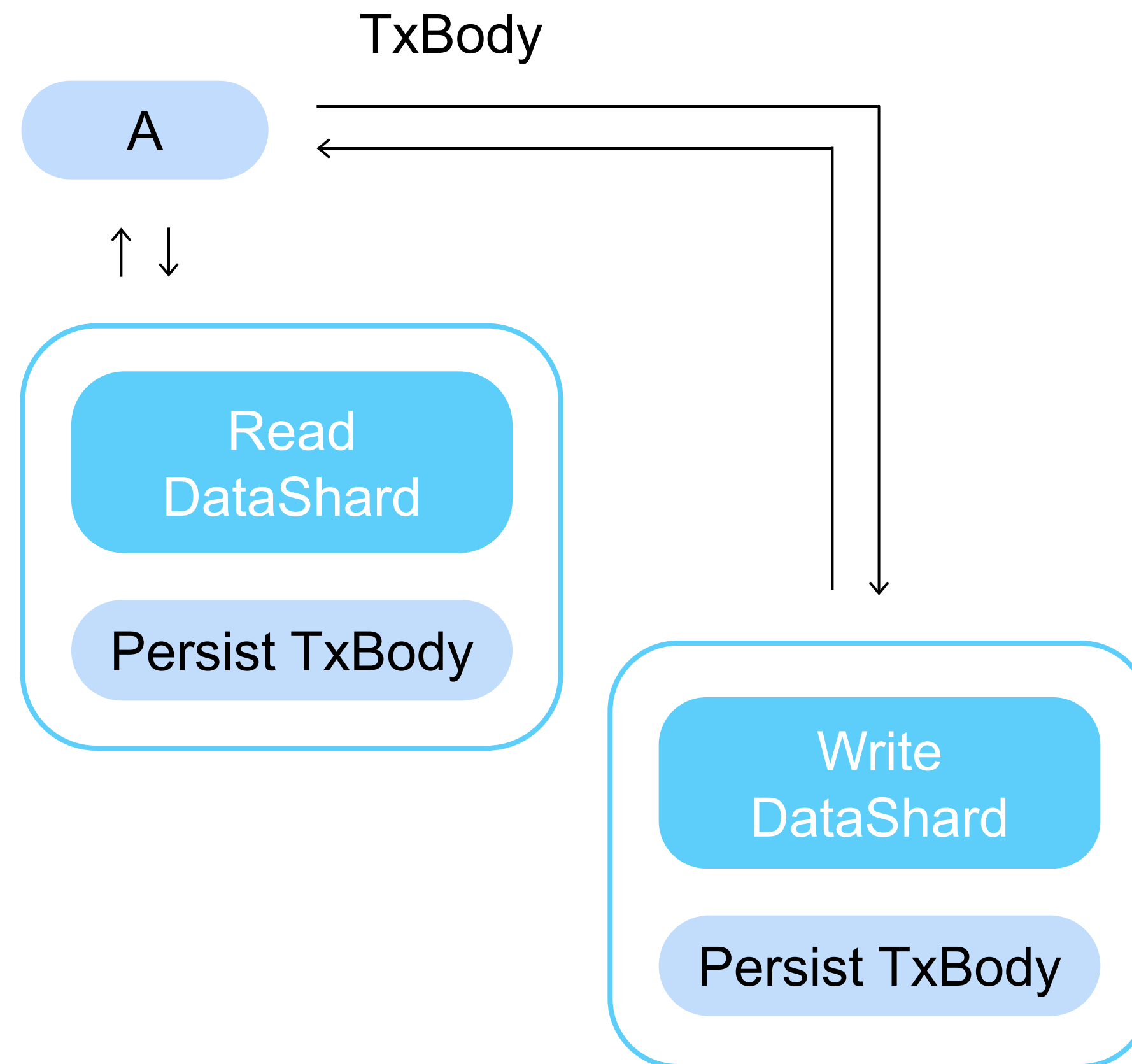
Транзакция Read-Write



Область применения

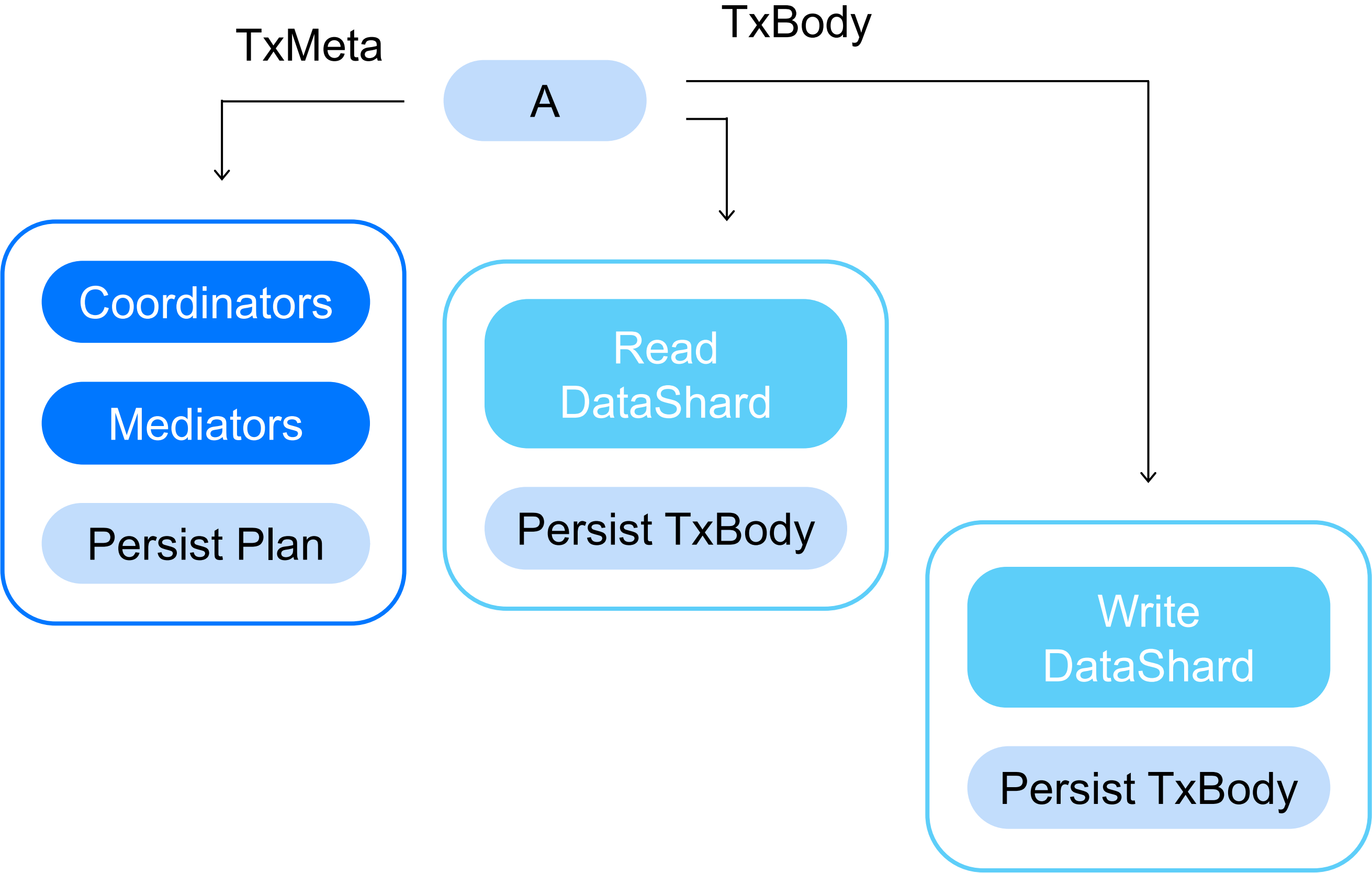
- › Высокая конкуренция на данных
- › Пропускная способность предпочтительнее быстрого отклика

Транзакция RW



- > 1 RTT
- > 1 write

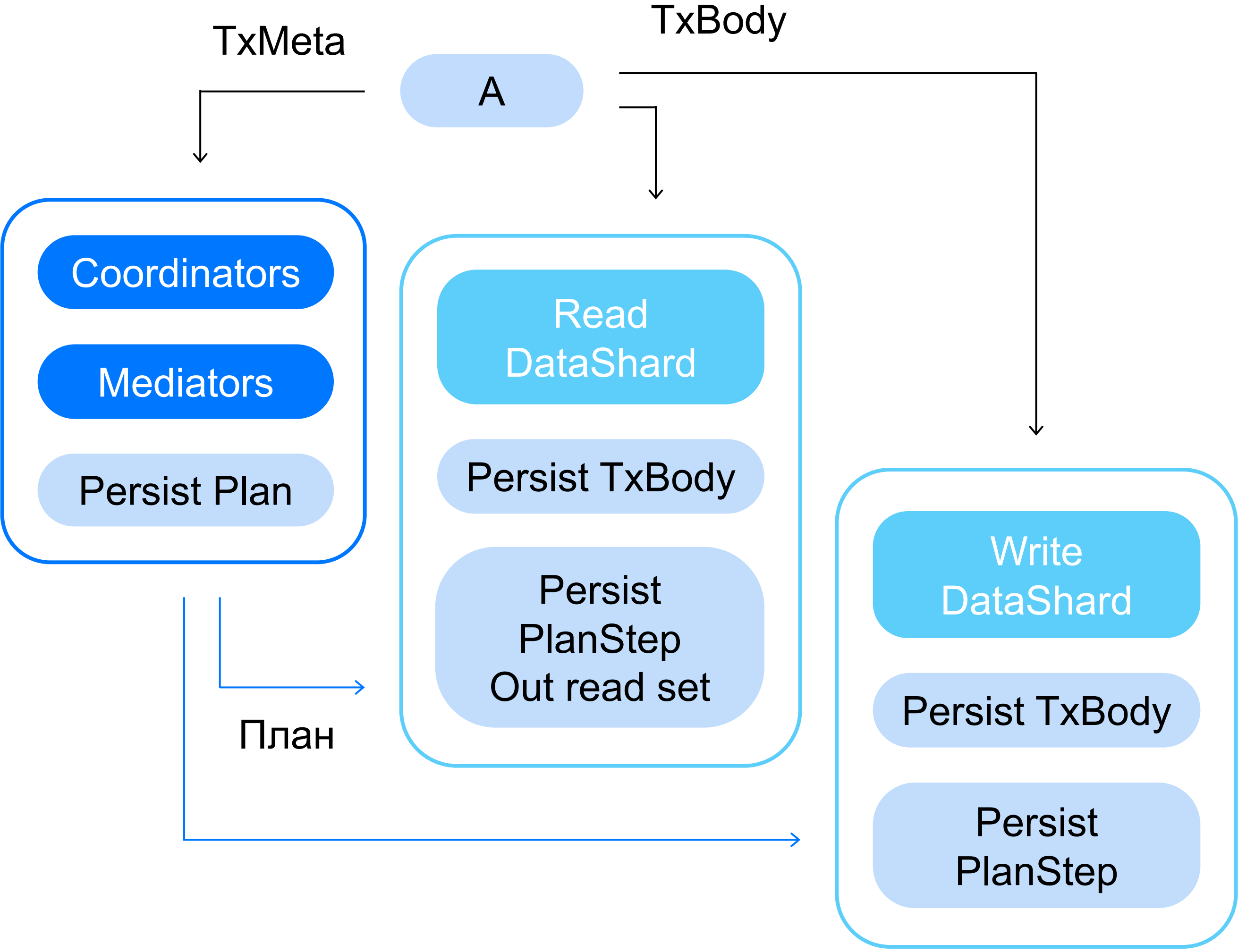
Транзакция RW



- > 1 RTT
- > 1 write

- > 0.5 RTT
- > 1 plan batch time
- > 1 write
- > 0.5 RTT

Транзакция RW

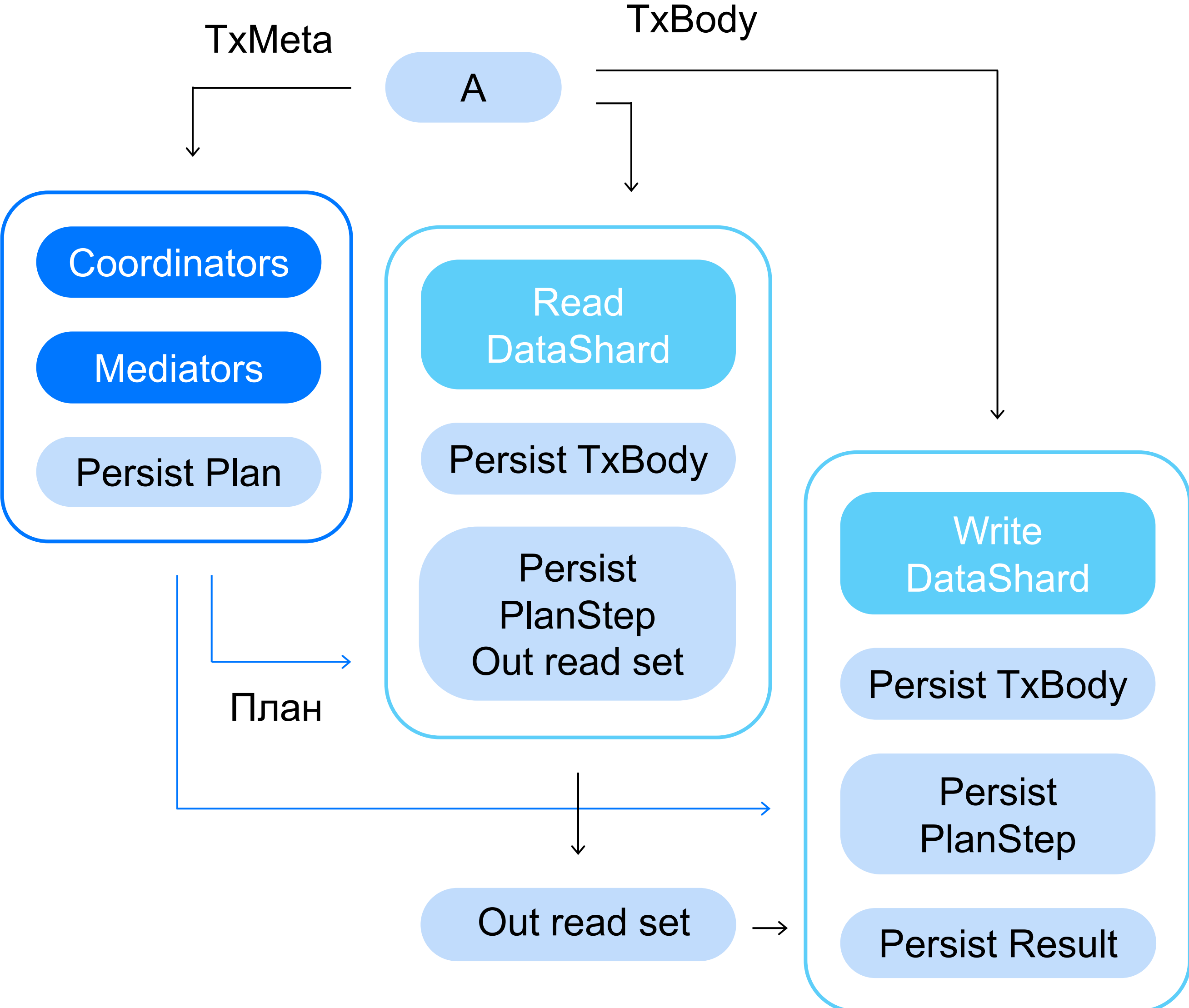


- > 1 RTT
- > 1 write

- > 0.5 RTT
- > 1 plan batch time
- > 1 write
- > 0.5 RTT

- > 0.5 RTT
- > 1 write

Транзакция RW



- > 1 RTT
- > 1 write

- > 0.5 RTT
- > 1 plan batch time
- > 1 write
- > 0.5 RTT

- > 0.5 RTT
- > 1 write

- > 0.5 RTT
- > 1 write
- > 0.5 RTT

- Итого**
- > 7.5 RTT + 1 plan batch time + 4 diskIO

Overview

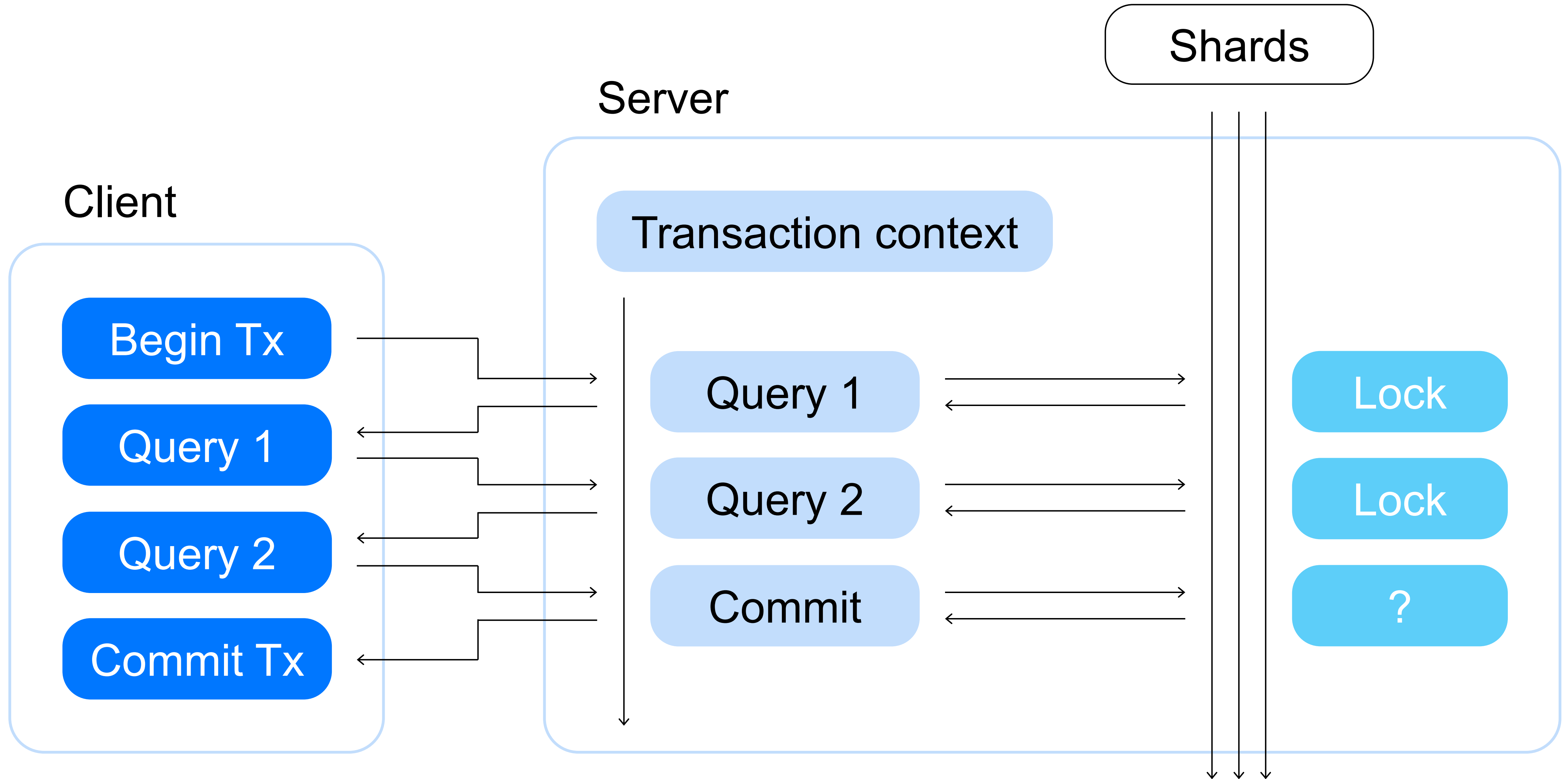


1. Read only, one shard — RO Immediate
— **1 RTT**
2. Write only, one shard — WO Immediate
— **2 RTT + 1 diskIO**
3. Read only / write only, multi shard — RO/WO
— **6 RTT + 1 plan batch time + 3 diskIO**
4. Read write, multi shard — RW
— **7.5 RTT + 1 plan batch time + 4 diskIO**

05

YQL-транзакции

Общий вид

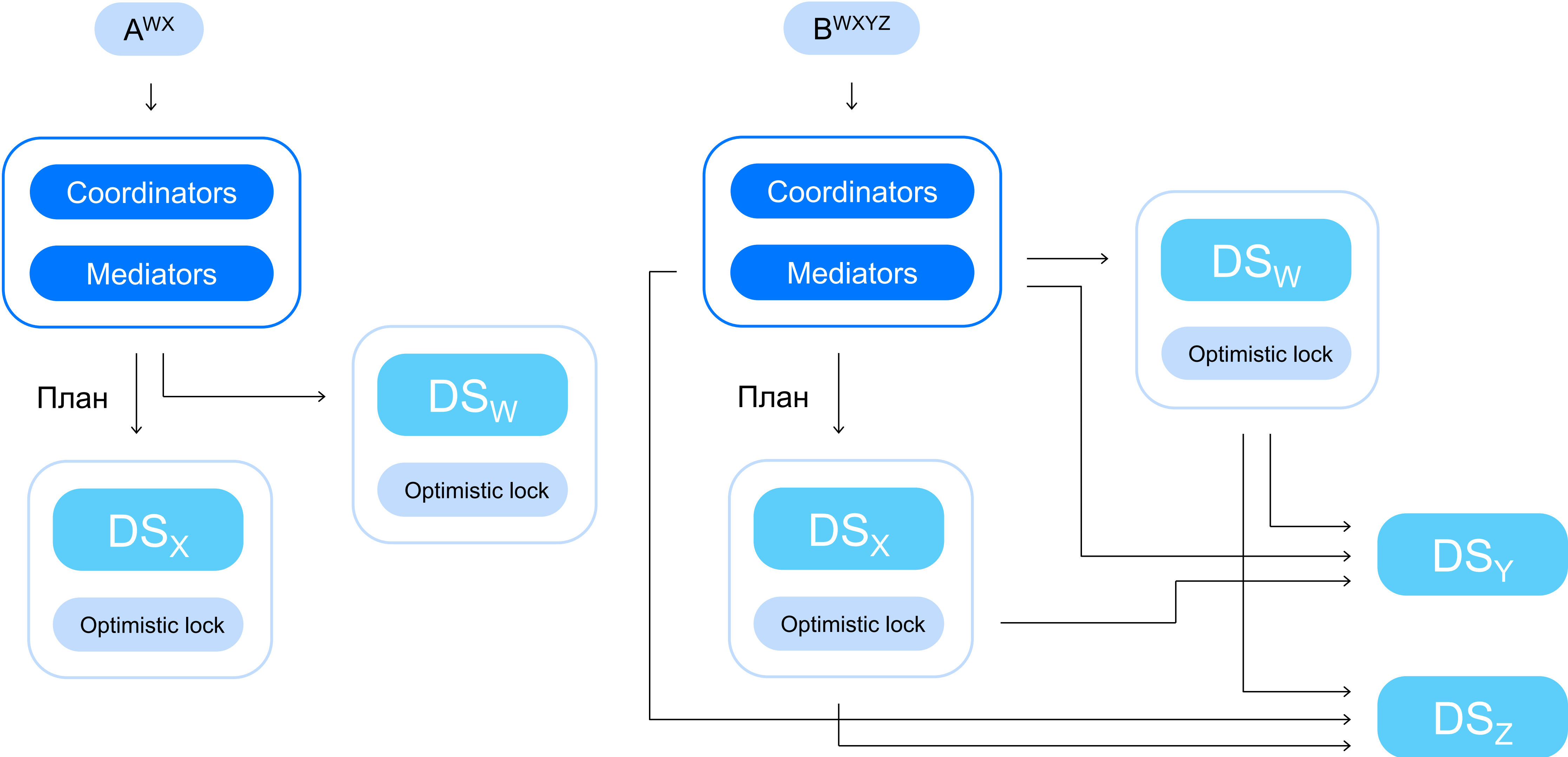


YQL-транзакции

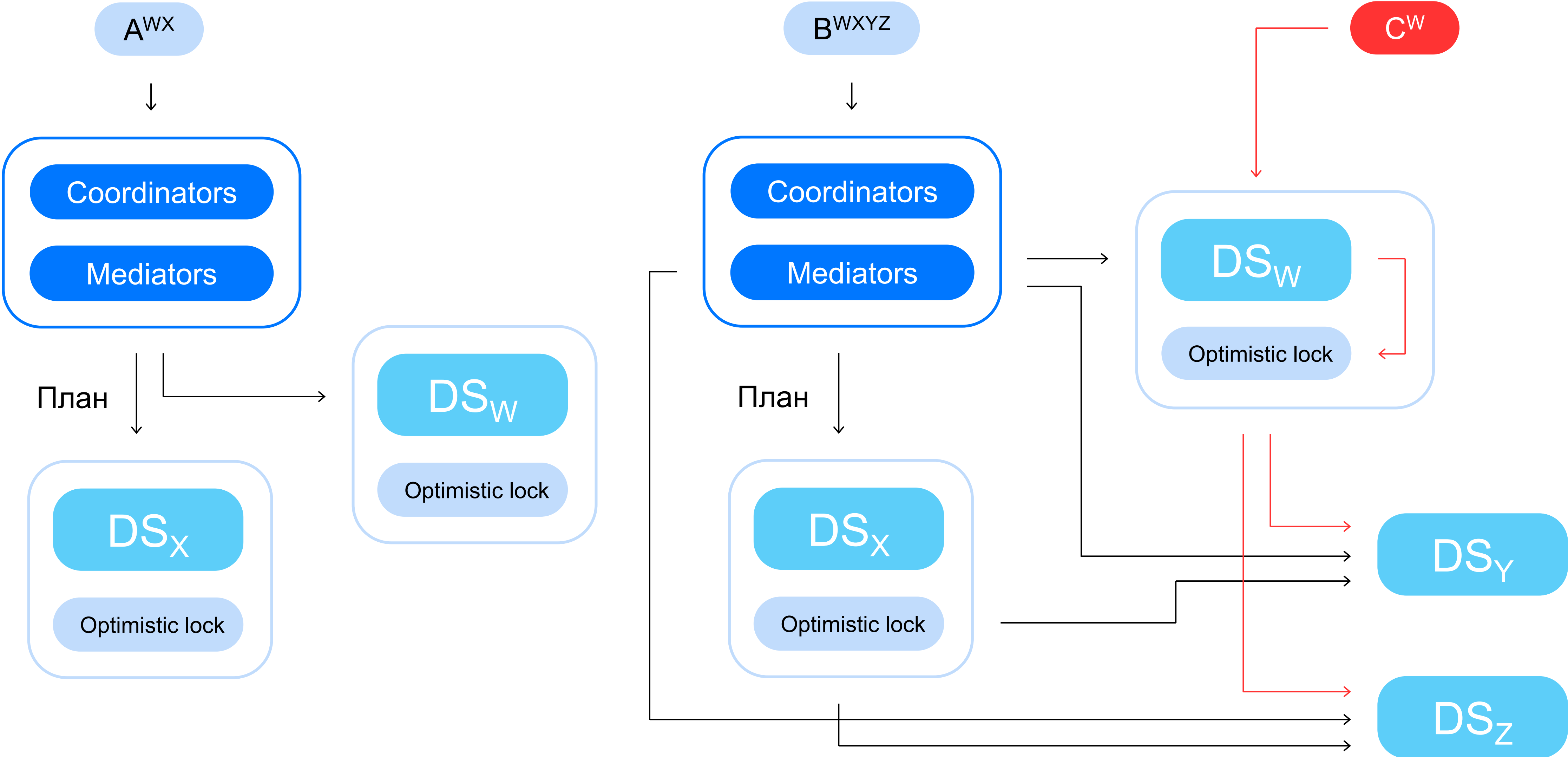


- › Несколько отдельных атомарных запросов в рамках одной YQL-транзакции
- › Набор read / write set'ов ключей каждого запроса определен
- › Каждый запрос чтения захватывает оптимистичную блокировку
- › Изменения применяются в последнем запросе на коммите транзакции
- › Взятые блокировки проверяются в последнем запросе на коммите транзакции

YQL-транзакции



YQL-транзакции



05

Доступные уровни изоляции для транзакций

Уровень изоляции



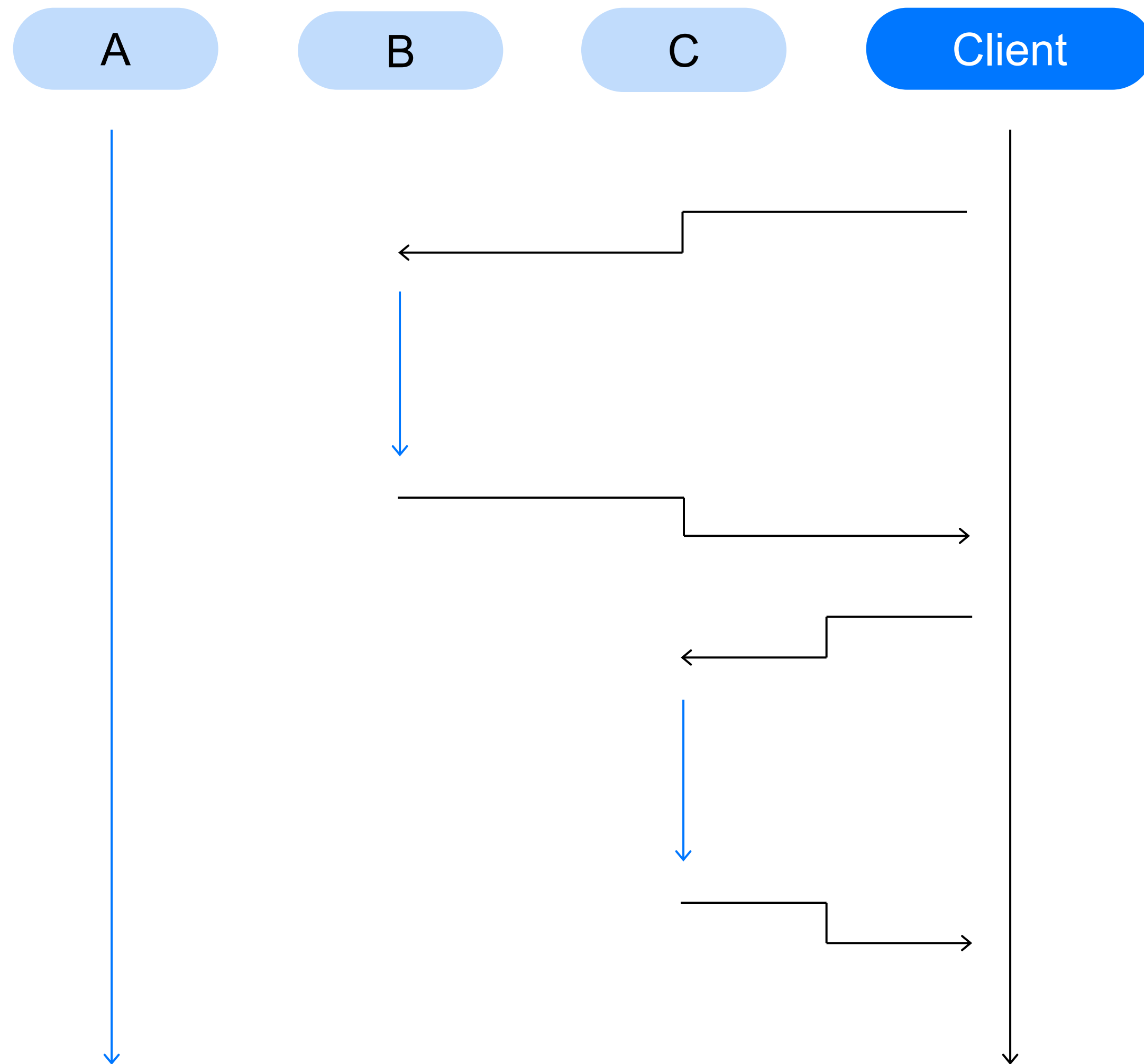
Serializable — default isolation level

- › Координируемые и immediate-транзакции

Strict serializable — maximum isolation level

- › Все транзакции координируемы

Strict serializable

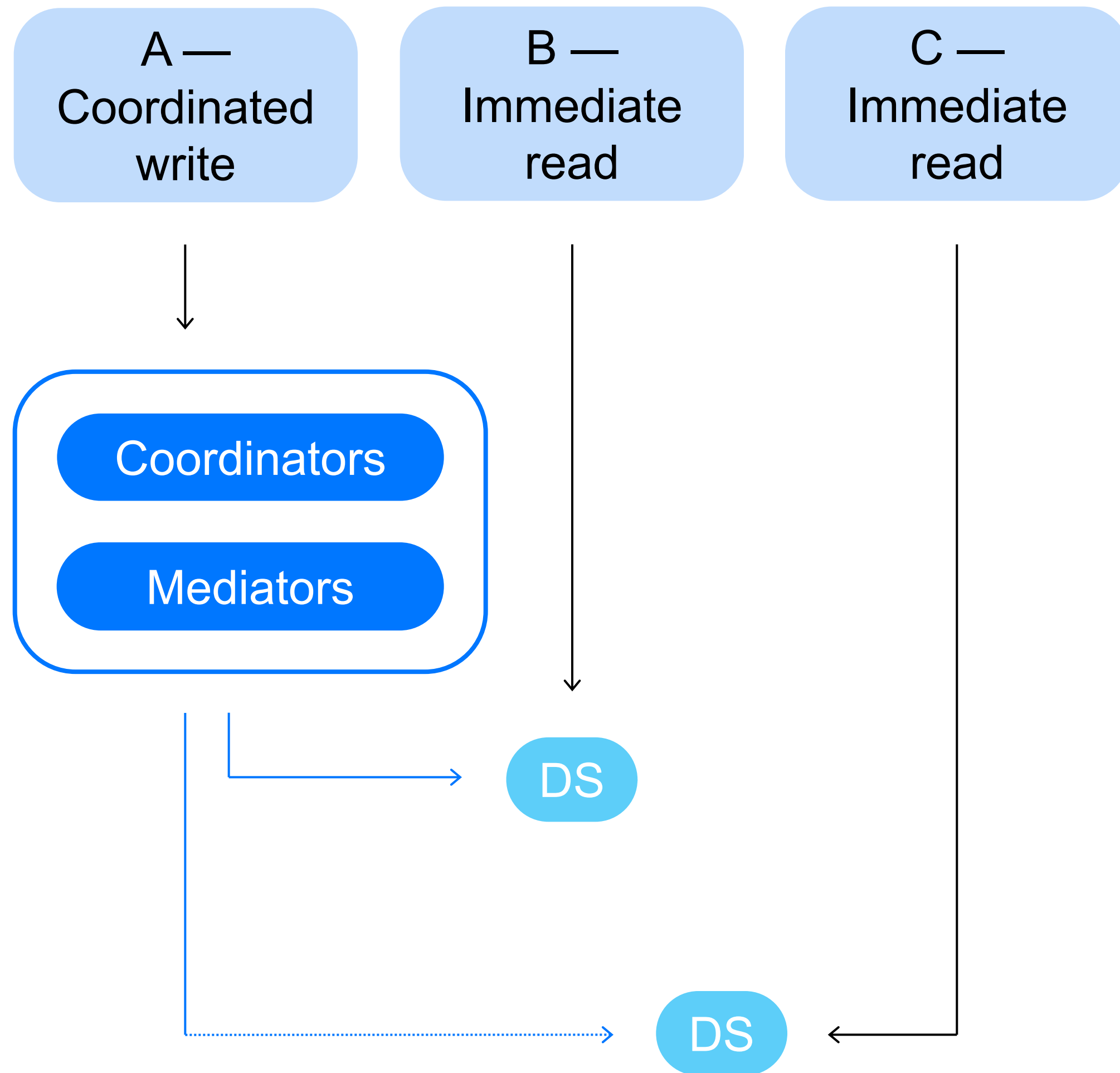


- › Транзакция A в процессе выполнения
- › Транзакции B и C выполняются последовательно

Strict serializable

- › Порядок между B и C определен однозначно
- › Если B видит изменения A, то C также должно их видеть

Strict serializable



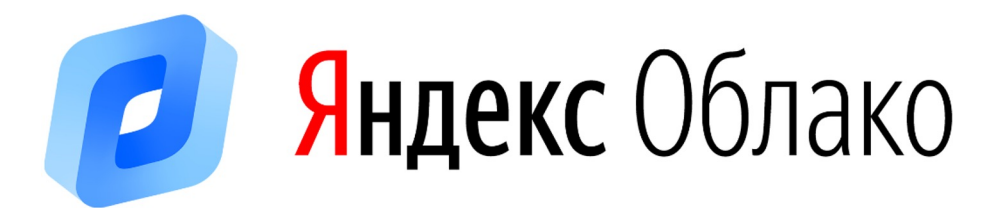
Serializable

- › Порядок между B и C существует, но не определен заранее
- › Возможно, что B видит изменения A, в то время, как C не видит изменений A
- › Возможный порядок выполнения: C A B

Yandex Database (YDB)



- › Надёжное хранение данных с избыточностью и автоматической репликацией
- › Отказоустойчивость, автоматическое восстановление от сбоев
- › Распределённые ACID-транзакции с serializable-уровнем изоляции транзакций
- › Высокая пропускная способность при малом времени отклика
- › Горизонтальная масштабируемость до тысяч нод



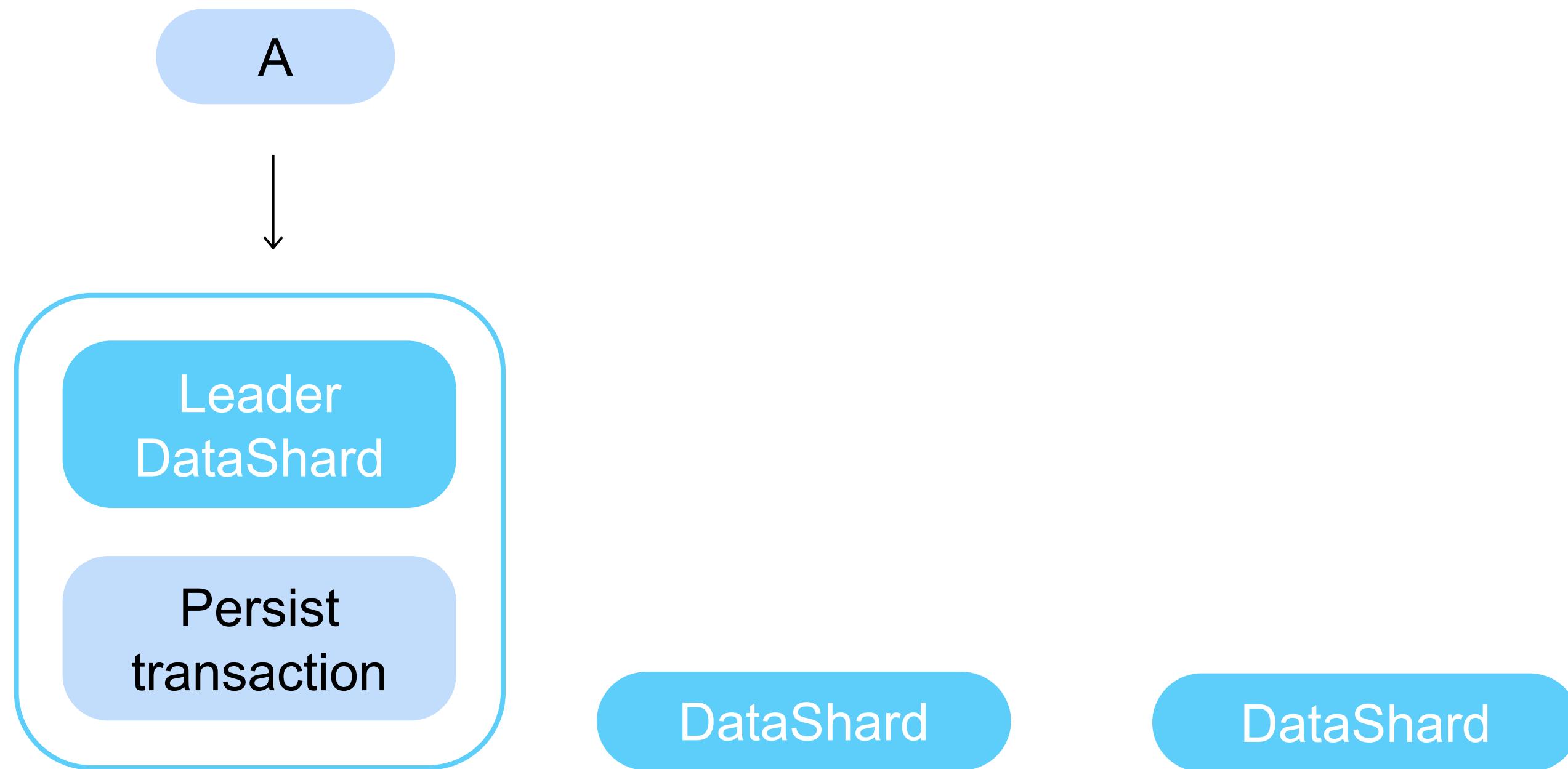
Спасибо!

Семён Чечеринда

Ведущий разработчик

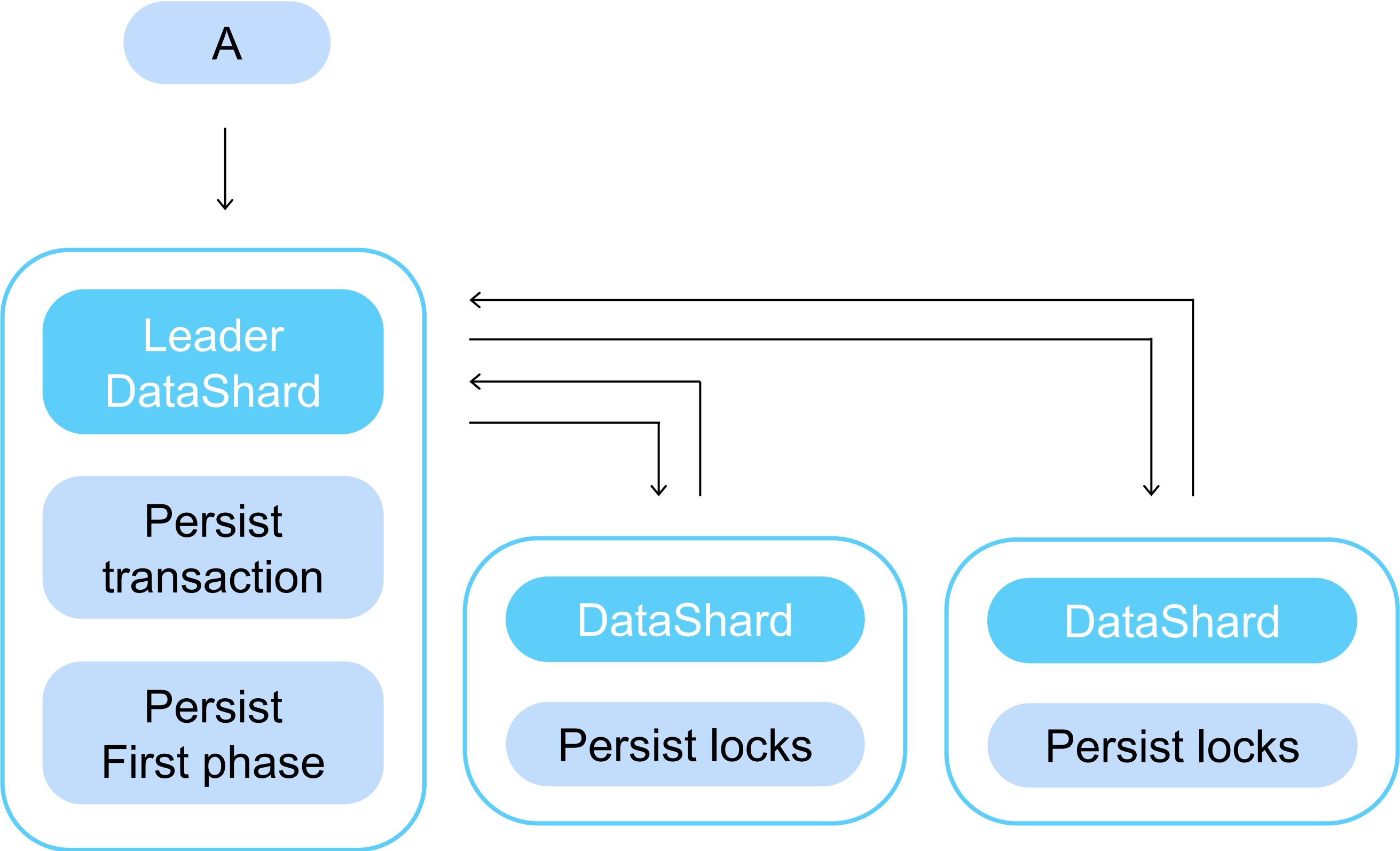
 svc@yandex-team.ru

What if two-phase commit



- > 0.5 RTT
- > 1 write

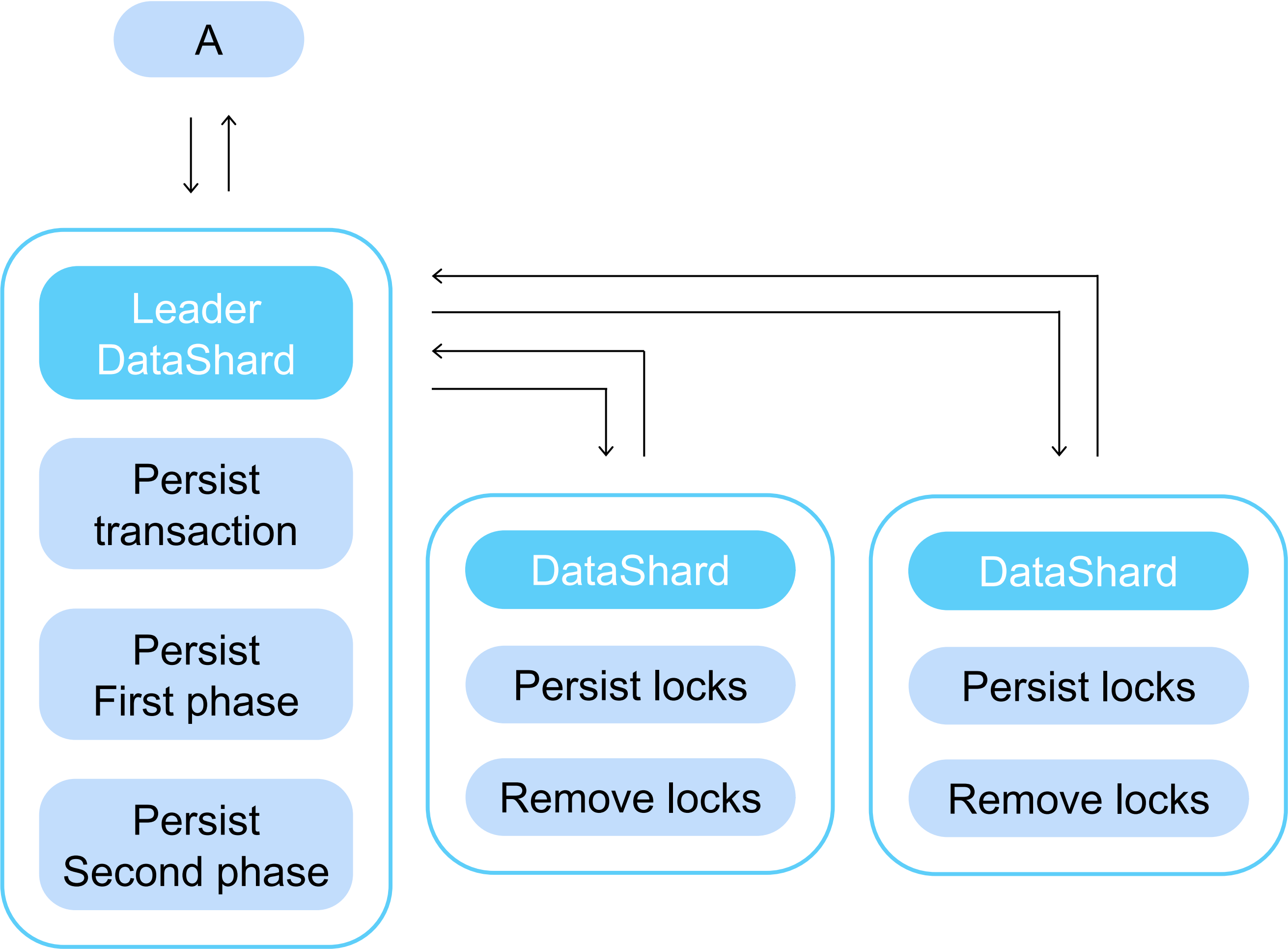
What if two-phase commit



- > 0.5 RTT
- > 1 write

- > 1 RTT
- > 1 write
- > 1 write

What if two-phase commit



- > 0.5 RTT
- > 1 write

- > 1 RTT
- > 1 write
- > 1 write

- > 1 RTT
- > 1 write
- > 1 write
- > 0.5 RTT

Итого
> 8 RTT + 5 diskIO