

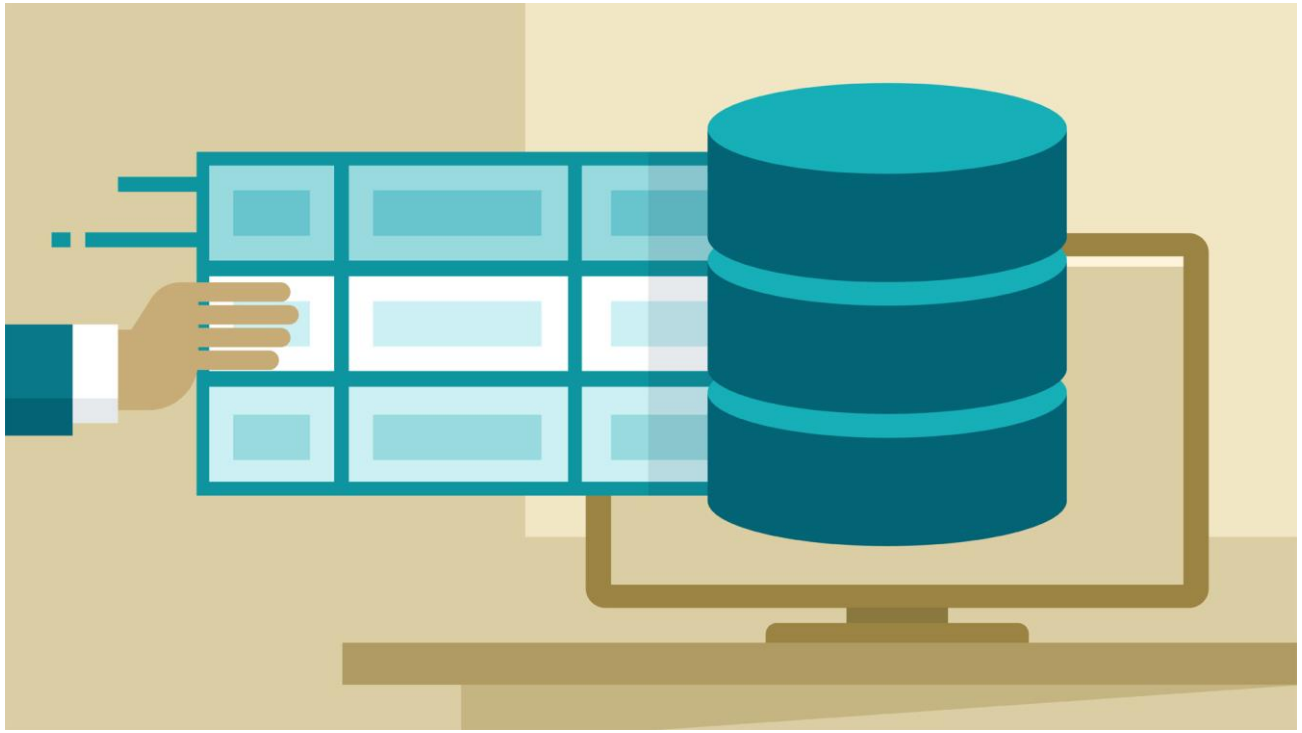
БАЗЫ ДАННЫХ

Лекция 5.

Транзакции в БД.
Свойства ACID.
Уровни изоляции

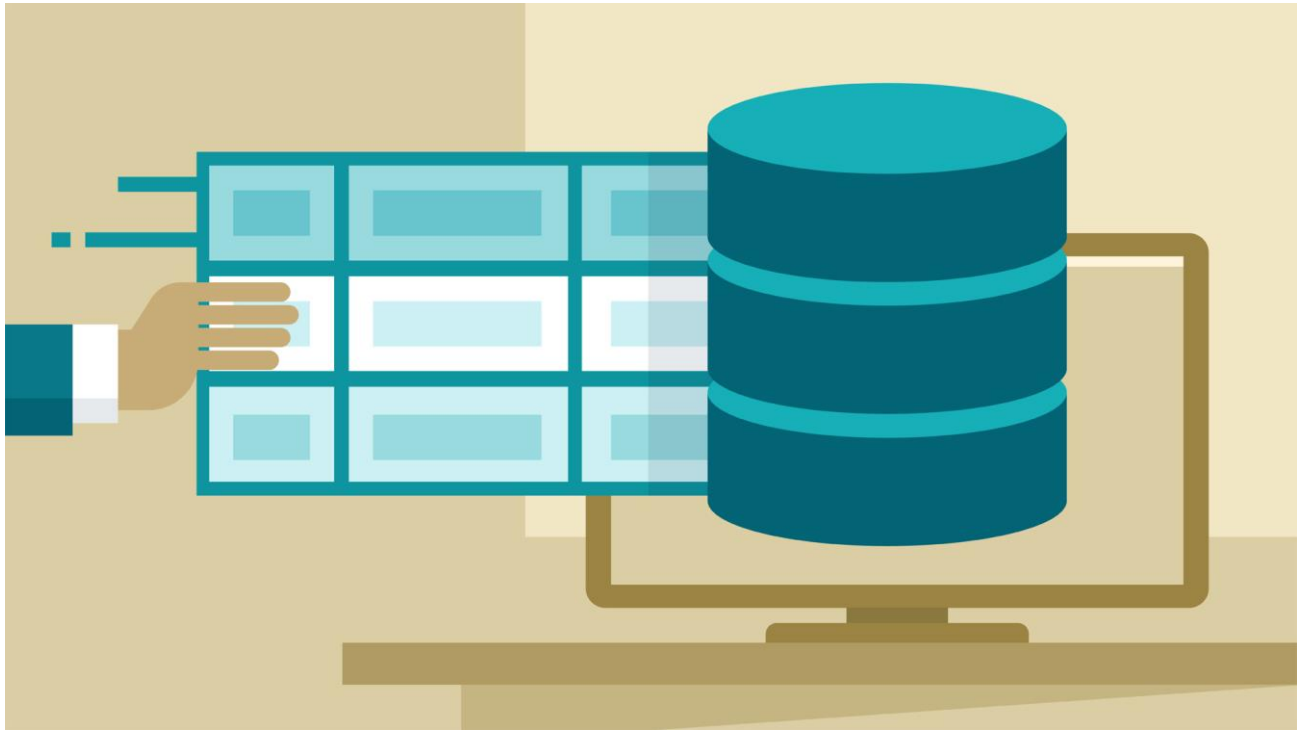
Определение транзакции

Транзакция - последовательность операций, производимых над БД и переводящих базу данных из одного непротиворечивого (согласованного) состояния в другое непротиворечивое (согласованное) состояние.



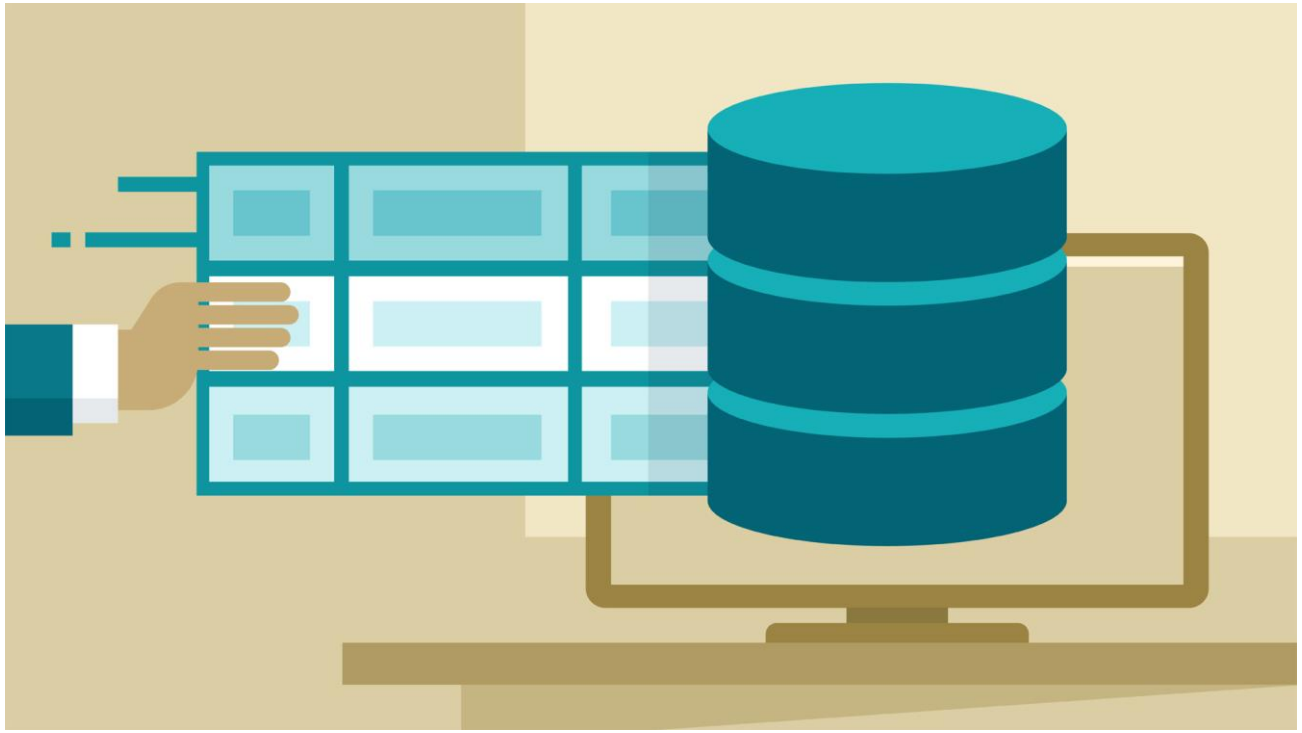
Определение транзакции

Транзакция - последовательность операций,
производимых над БД и переводящих базу данных
из одного непротиворечивого (согласованного) состояния
в другое непротиворечивое (согласованное) состояние.



Определение транзакции

Транзакция - последовательность операций, производимых над БД и переводящих базу данных из одного непротиворечивого (согласованного) состояния в другое непротиворечивое (согласованное) состояние.



Определение транзакции

Транзакция

- *Для пользователя*: некоторое неделимое действие над БД, осмысленное с точки зрения пользователя.

- *Для системы*: логическая единица работы системы



Определение транзакции

UPDATE account SET balance = balance – 1000
WHERE owner = ‘Alice’

UPDATE account SET balance = balance + 1000
WHERE owner = ‘Bob’



Определение транзакции

UPDATE account SET balance = balance – 1000
WHERE owner = ‘Alice’

UPDATE account SET balance = balance + 1000
WHERE owner = ‘Bob’



Если произошла ошибка в любом из действий,
то вся операция должна быть отменена

↓

2 обновления (update) составляют единую транзакцию

Виды транзакций

Модели транзакций классифицируются на основании различных свойств:

- структура транзакции;
- параллельность внутри транзакции;
- продолжительность.

Типы транзакций:

- плоские (классические);
- цепочечные (распределенные);
- вложенные.



Операторы транзакций

TCL – Transaction Control Language –
команды управления транзакциями



BEGIN – начало транзакции



COMMIT – успешное завершение транзакции



ROLLBACK – откат транзакции

По умолчанию в СУБД установлен **AUTOCOMMIT**
(без явного оператора BEGIN)

Операторы транзакций



COMMIT

Производится если:

- все операторы выполнены успешно
- в процессе выполнения транзакции не произошло никаких сбоев программного или аппаратного обеспечения

Фиксация транзакции — это действие, обеспечивающее запись на диск изменений в базе данных, которые были сделаны в процессе выполнения транзакции.

Операторы транзакций

BEGIN;

UPDATE account SET balance = balance – 1000
WHERE owner = ‘Alice’;

UPDATE account SET balance = balance + 1000
WHERE owner = ‘Bob’;

COMMIT;

Произведено 2 обновления

Операторы транзакций



ROLLBACK

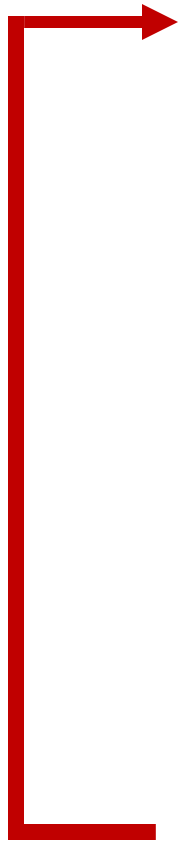
Производится если:

- в процессе выполнения транзакции случилось нечто такое, что делает невозможным ее нормальное завершение

База данных должна быть возвращена в исходное состояние.

Откат транзакции — это действие, обеспечивающее аннулирование всех изменений данных, которые были сделаны операторами SQL в теле текущей незавершенной транзакции.

Операторы транзакций



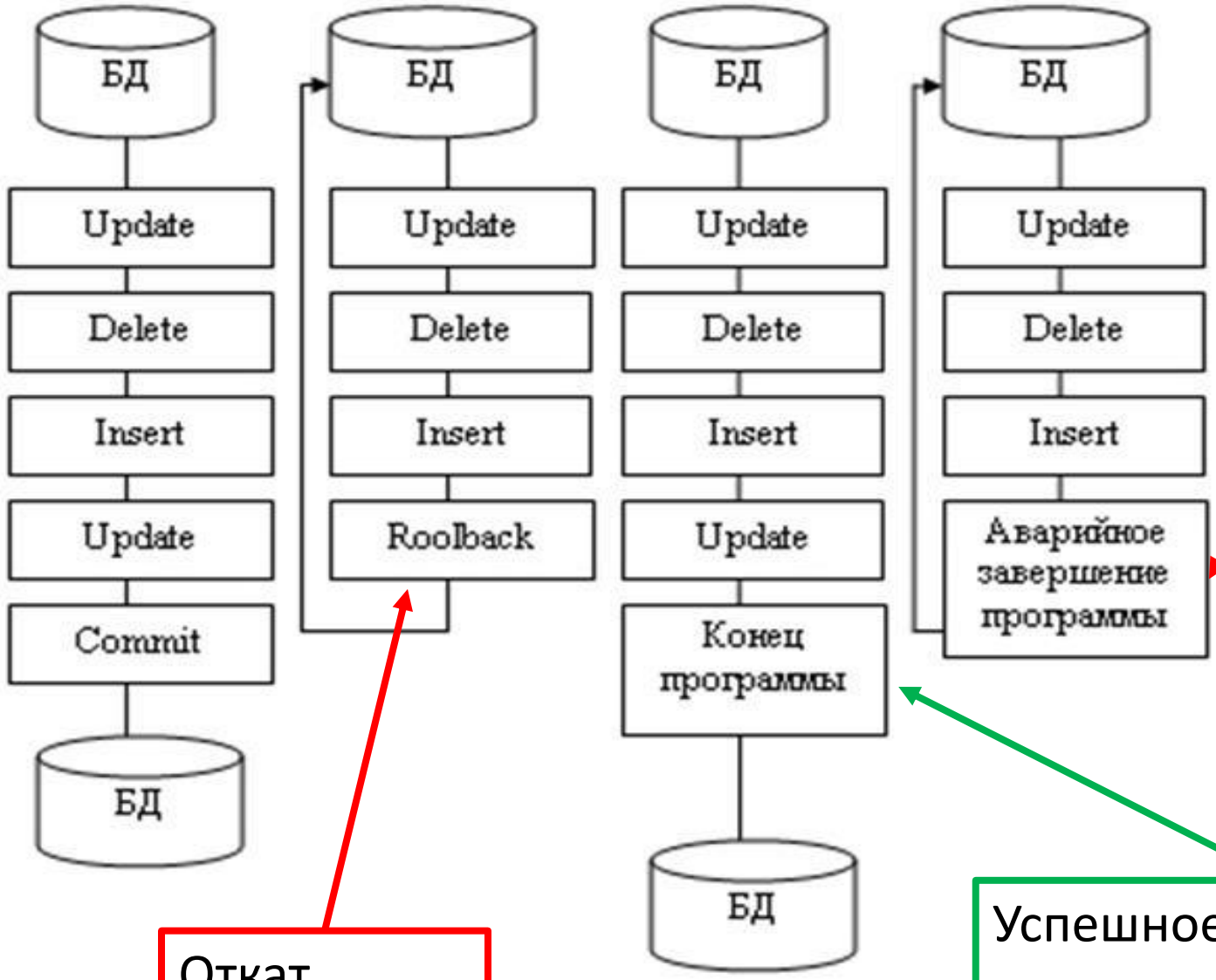
BEGIN;

UPDATE account SET balance = balance – 1000
WHERE owner = ‘Alice’;

UPDATE account SET balance = balance + 1000
WHERE owner = ‘Bob’;

ROLLBACK;

Не произведено обновлений (отменены)



Успешное завершение транзакции

Откат транзакции

Аварийное завершение программы (откат транзакции)

Успешное завершение программы (коммит транзакции)

Точки сохранения в транзакции

- **SAVEPOINT имя** – создать точку сохранения
- **ROLLBACK TO имя** – откатиться до точки сохранения
- **RELEASE имя** – удалить точку сохранения

Точки сохранения позволяют устанавливать *маркеры*
внутри транзакции таким образом,
чтобы имелась возможность отмены только части работы,
проделанной в транзакции.

Точки сохранения в транзакции

BEGIN;
UPDATE account SET balance = balance – 1000
WHERE owner = ‘Alice’;



SAVEPOINT sp;

UPDATE account SET balance = balance + 1000
WHERE owner = ‘Bob’;

ROLLBACK TO sp;
COMMIT;

Произведено только первое обновление

Операторы транзакций

DML-операторы

DDL-операторы

Операторы транзакций

DML-операторы

SELECT
UPDATE
INSERT
DELETE

Транзакционные
(выполняются внутри блока транзакции, можно зафиксировать или откатить)

DDL-операторы

Операторы транзакций

DML-операторы

SELECT
UPDATE
INSERT
DELETE

Транзакционные
(выполняются внутри блока транзакции, можно зафиксировать или откатить)

DDL-операторы

CREATE
DROP
TRUNCATE
ALTER

во многих СУБД (MySQL, Oracle…):

Нетранзакционные
(выполняются в отдельной транзакции, которая автоматически фиксируется, невозможно откатить)

Операторы транзакций

DML-операторы

SELECT
UPDATE
INSERT
DELETE

Транзакционные
(выполняются внутри блока транзакции, можно зафиксировать или откатить)

DDL-операторы

CREATE
DROP
TRUNCATE
ALTER

В PostgreSQL:

Тоже транзакционные
За некоторыми исключениями (команды VACUUM, CREATE DATABASE и др.), можно откатить любую команду.

Свойства транзакций

4 свойства классической (плоской, традиционной) транзакции

Atomicity

Consistency

Isolation

Durability



Атомарность

Согласованность

Изолированность

Долговечность

(устойчивость, надежность)

Свойства транзакций

Атомарность (Atomicity)

Транзакция должна быть выполнена в целом или не выполнена вовсе.

Транзакция «неделима»

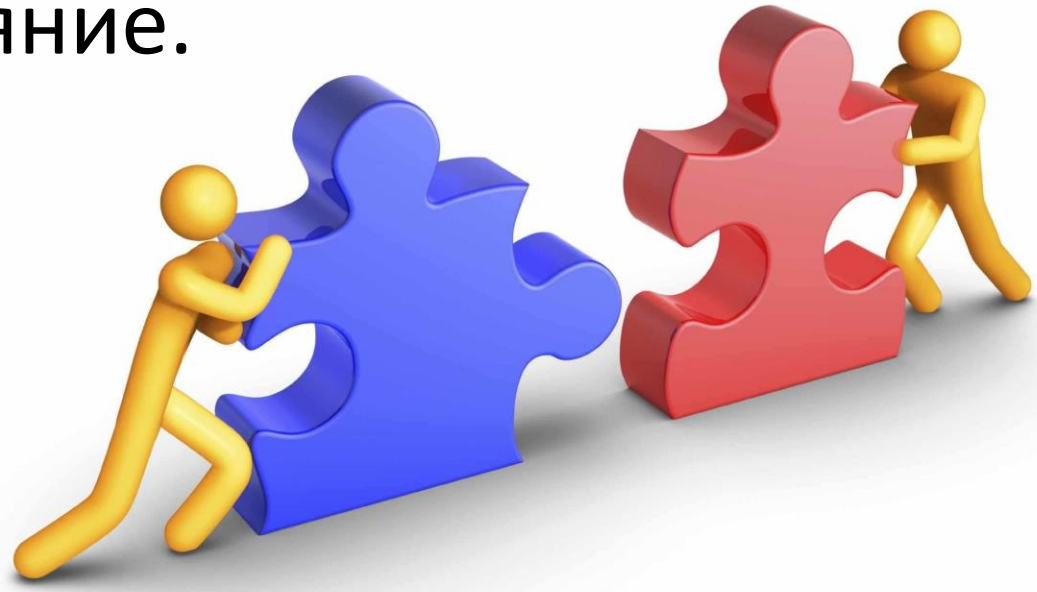


Свойства транзакций

Согласованность (Consistency) //целостность

По мере выполнения транзакции
данные переходят
из одного согласованного состояния
в другое согласованное состояние.

Транзакция не разрушает взаимной согласованности (целостности) данных



Согласованная БД

	id_stud	name
1	1	Alice
2	2	Ivan
3	3	Denis
4	4	Eva
5	5	Petr

Таблица student

	id_sub	name	teacher
1	1	TYAP	Osipova U.V.
2	2	ZI	Dyachkova I.S.
3	3	PMU	Nechta I.V.
4	4	BD	Dyachkova M.S.
5	5	MMO	Dementieva K.I.

Таблица subject

	id_d	id_stud	id_sub	deadline
1	1	1	1	2025-09-30
2	2	1	2	2025-09-20
3	3	1	3	2025-09-25
4	4	2	1	2025-10-01
5	5	3	3	2025-09-30
6	6	3	4	2025-12-31
7	7	4	2	2025-10-31

Таблица debt

Согласованная БД

	id_stud	name
1	1	Alice
2	2	Ivan
3	3	Denis
4	4	Eva
5	5	Petr

Таблица student

	id_sub	name	teacher
1	1	TYAP	Osipova U.V.
2	2	ZI	Dyachkova I.S.
3	3	PMU	Nechta I.V.
4	4	BD	Dyachkova M.S.
5	5	MMO	Dementieva K.I.

Таблица subject

	id_d	id_stud	id_sub	deadline
1	1	1	1	2025-09-30
2	2	1	2	2025-09-20
3	3	1	3	2025-09-25
4	4	2	1	2025-10-01
5	5	3	3	2025-09-30
6	6	3	4	2025-12-31
7	7	4	2	2025-10-31

Таблица debt

INSERT INTO student VALUES (5, ‘Dima’);

?

Согласованная БД

	id_stud	name
1	1	Alice
2	2	Ivan
3	3	Denis
4	4	Eva
5	5	Petr

Таблица student

	id_sub	name	teacher
1	1	TYAP	Osipova U.V.
2	2	ZI	Dyachkova I.S.
3	3	PMU	Nechta I.V.
4	4	BD	Dyachkova M.S.
5	5	MMO	Dementieva K.I.

Таблица subject

	id_d	id_stud	id_sub	deadline
1	1	1	1	2025-09-30
2	2	1	2	2025-09-20
3	3	1	3	2025-09-25
4	4	2	1	2025-10-01
5	5	3	3	2025-09-30
6	6	3	4	2025-12-31
7	7	4	2	2025-10-31

Таблица debt

INSERT INTO debt
VALUES
(8, 5, 6, ‘2025-10-31’);

?

Свойства транзакций

Долговечность (Durability)

//устойчивость, надежность

Если транзакция завершена успешно, то внесенные ею изменения не могут быть потеряны
(даже в случае последующих ошибок)

Зафиксированные (commit) данные «вечны»



Журнал предзаписи WAL

Журнал транзакций - это файл или набор файлов, в которых PostgreSQL записывает все изменения, внесенные в базу данных.

Журнал предзаписи



Механизм W*rite-Ahead Logging*

WAL –

стандартный метод обеспечения целостности данных.

Журнал предзаписи WAL

Журнал предзаписи *Write-Ahead Logging*

Основная идея:

изменения в файлах с данными (где находятся таблицы и индексы)
должны записываться только после того,
как эти изменения были занесены в WAL,
т. е. после того как записи WAL, описывающие данные изменения,
будут сохранены на постоянное устройство хранения.

Сначала запись в журнал, затем изменения в файлах БД

Журнал предзаписи WAL

Основное назначение файлов журнала:

1. **Восстановление после сбоя**: В случае сбоя системы PostgreSQL может использовать файлы журнала для восстановления базы данных до последнего состоящего момента.
2. **Обеспечение ACID-свойств**: Журнал транзакций гарантирует, что все изменения, внесенные в базу данных, выполняются атомарно, согласованно, изолированно и долговечно.
3. **Репликация**: журнал транзакций используется для репликации данных на другие серверы, что позволяет создать отказоустойчивую систему.
4. **Точечное восстановление**: журнал позволяет восстановить базу данных до определенного момента времени, что может быть полезно для восстановления удаленных данных или отката изменений

Журнал предзаписи WAL

Преимущества WAL:

1. **Повышенная производительность**: значительное уменьшение количества запросов записи на диск.

Для гарантии, что транзакция подтверждена, в записи на диск нуждается только файл WAL, а не каждый файл данных, изменённый в результате транзакции.

Файл WAL записывается последовательно и таким образом затраты на синхронизацию WAL намного меньше, чем затраты на запись страниц с данными.

Это особенно справедливо для серверов, которые обрабатывают много маленьких транзакций, изменяющих разные части хранилища данных.

Журнал предзаписи WAL

Преимущества WAL:

2. Улучшенная отказоустойчивость: журнал обеспечивает возможность быстрого и полного восстановления базы данных после сбоя.

WAL также делает возможным поддержку резервного копирования «на ходу» и восстановления на определённый момент времени.

С помощью архивирования данных WAL поддерживается возврат к любому моменту времени, который доступен в данных WAL:
мы просто устанавливаем предыдущую физическую резервную копию базы данных и воспроизводим WAL до нужного момента времени.

Журнал предзаписи WAL

Преимущества WAL:

3. Поддержка репликации.

Журнал транзакций используется для репликации данных на другие серверы.

Журнал предзаписи WAL

Журнал предзаписи *Write-Ahead Logging*

Сначала запись в журнал, затем изменения в файлах БД

Если следовать этой процедуре:

- Нет необходимости записывать страницы данных на диск после подтверждения каждой транзакции
- Если случится сбой, то будет возможность восстановить БД с помощью журнала
- Любые изменения, которые не были применены к страницам с данными, могут быть воссозданы из записей WAL. (Восстановление с воспроизведением или REDO.)

Свойства транзакций

Изолированность (Isolation)

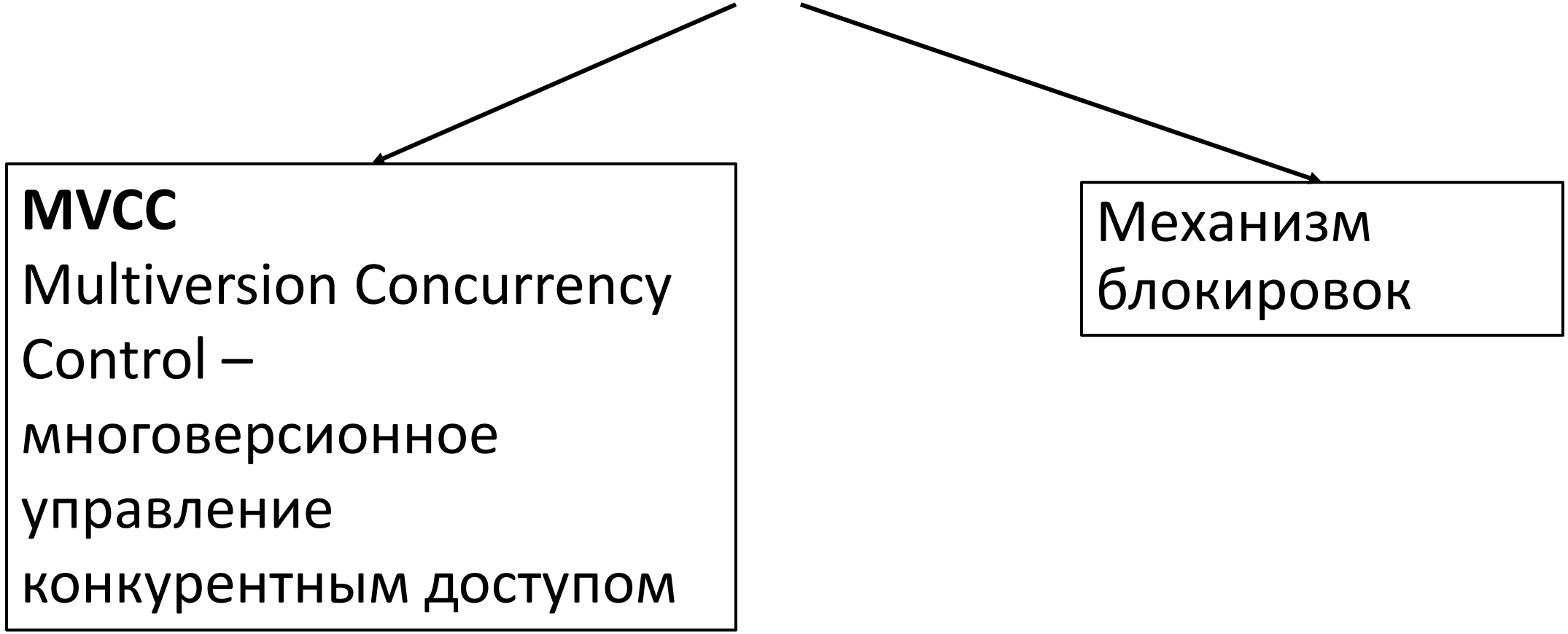
Конкурирующие за доступ к БД (параллельно выполняющиеся) транзакции не влияют друг на друга. Результат их выполнения должен быть таким же, как если бы они выполнялись последовательно, по очереди.

Транзакции «не мешают» друг другу



Изоляция данных в БД

2 парадигмы подхода к обеспечению изоляции данных



MVCC

Multiversion Concurrency Control –
многоверсионное управление конкурентным доступом

- «писатели» не блокируют «читателей» и наоборот
- У каждой транзакции есть «номер» *xid (32-bit)*
- Хранятся **версии строк** с указанием транзакций, в которых они действительны

Используется в PostgreSQL

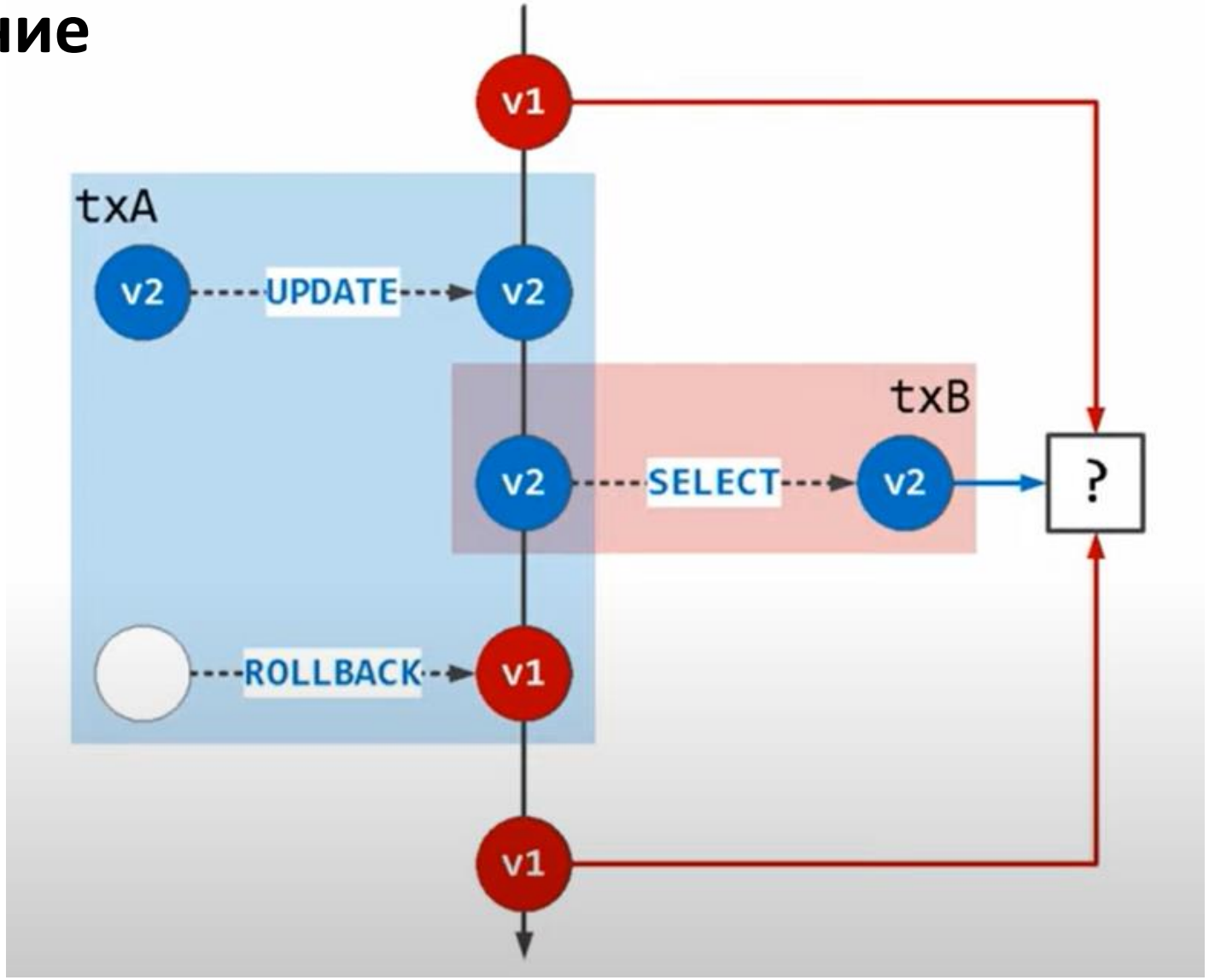
Уровни изоляции в БД

Уровень изоляции	*«Грязное» чтение (dirty read)*	*Неповторяемое чтение (nonrepeatable read)*	*Фантомное чтение (phantom read)*	*Аномалия сериализации (serialization anomaly)*
READ UNCOMMITTED (Чтение Незафиксированных данных)	Допускается, но не в PG	Возможно	Возможно	Возможно
READ COMMITTED (Чтение зафиксированных данных)	Невозможно	Возможно	Возможно	Возможно
REPEATABLE READ (Повторяемое чтение)	Невозможно	Невозможно	Допускается, но не в PG	Возможно
SERIALIZABLE (Сериализуемость)	Невозможно	Невозможно	Невозможно	Невозможно

Феномен чтения – отличия при параллельном и изолированном выполнении транзакций

Феномены чтения в БД

Грязное чтение
dirty read



Феномены чтения в БД

Грязное чтение
dirty read

Транзакция 1

```
INSERT INTO t(id,data) VALUES (1, 1);
BEGIN;
UPDATE t SET data = data + 1;
…
…
ROLLBACK;
```

Транзакция 2

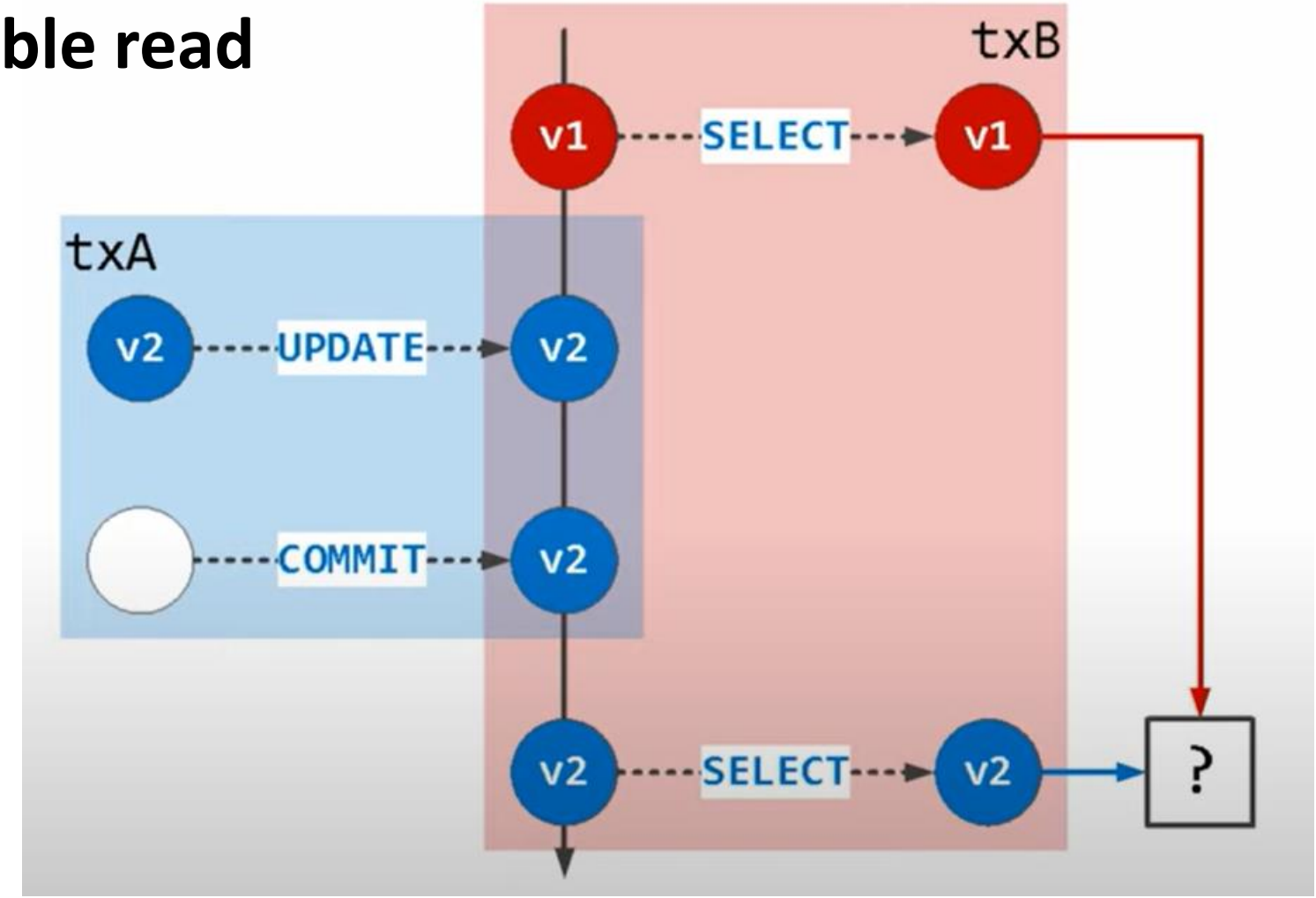
```
SELECT data from t WHERE id=1;
```


Реальная запись: (1, 1) Прочитано: (1, 2)

Запись прочитана, а в базе такой «нет и не было»!

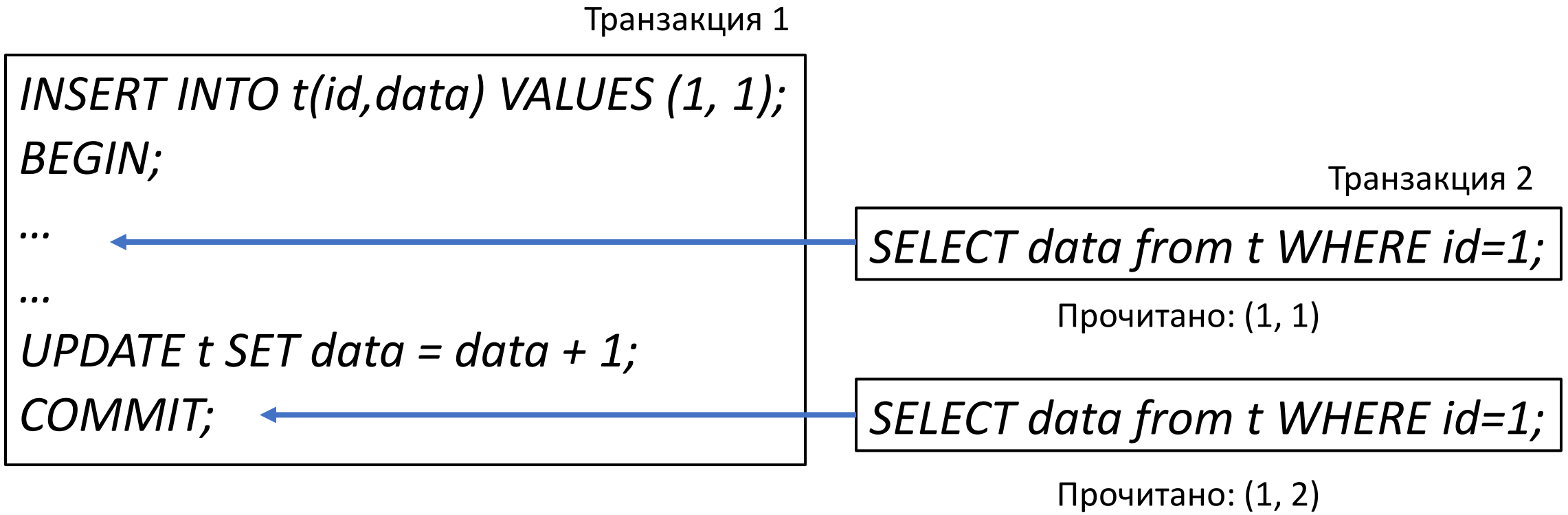
Феномены чтения в БД

Неповторяющееся чтение
nonrepeatable read



Феномены чтения в БД

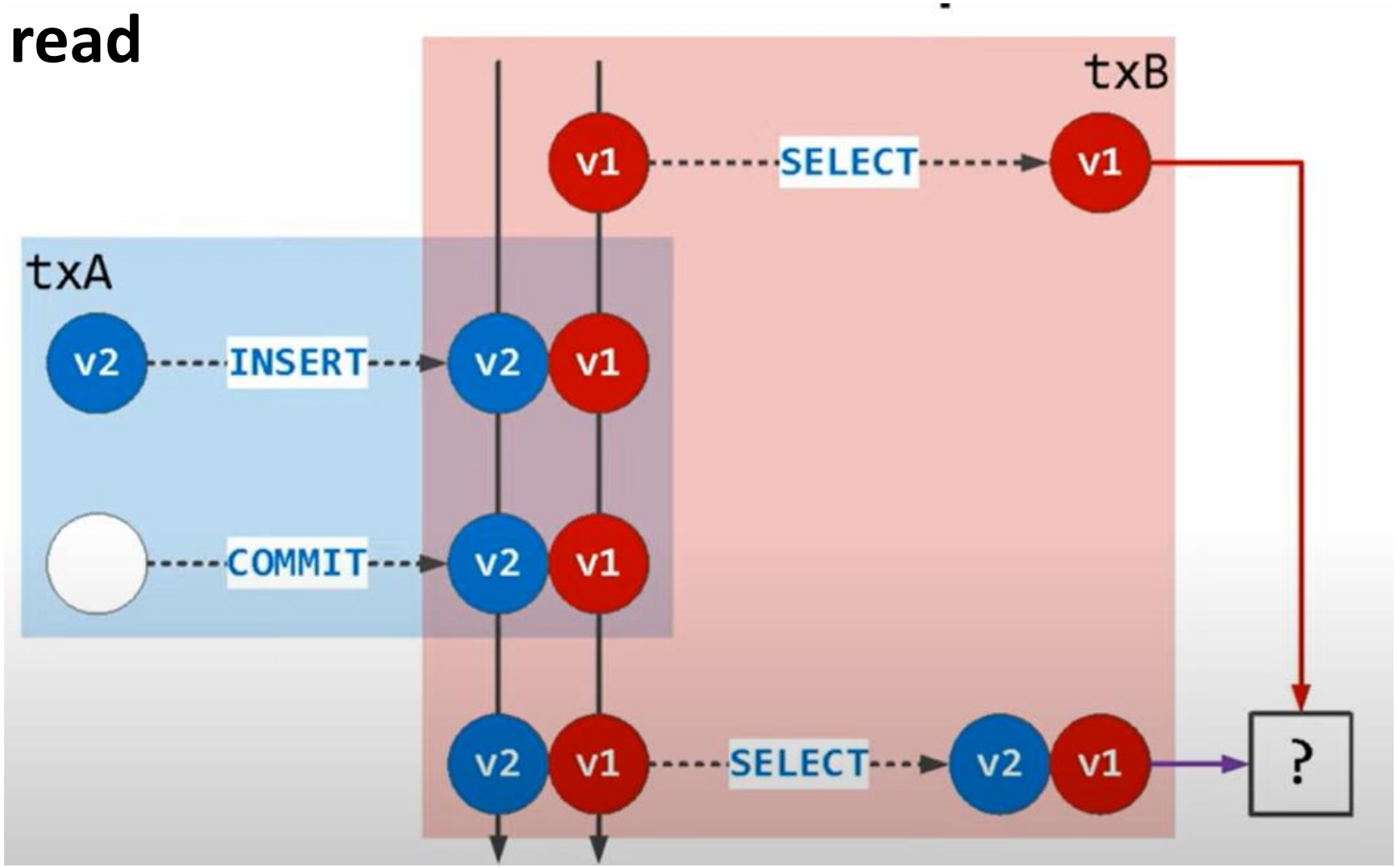
Неповторяющееся чтение
nonrepeatable read



«Перечитали» ту же запись, а она другая!

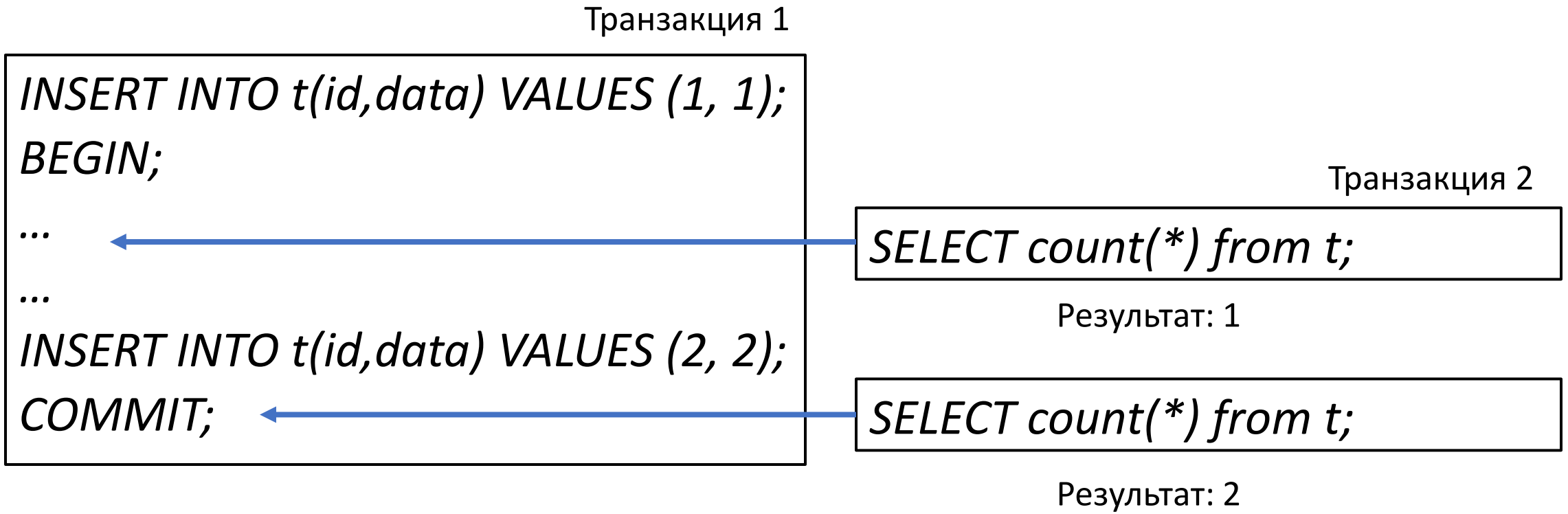
Феномены чтения в БД

Фантомное чтение
phantom read



Феномены чтения в БД

Фантомное чтение
phantom read



«Перечитали» тот же запрос, а набор записей другой!

Феномены чтения в БД

- **READ UNCOMMITTED – отсутствует в PostgreSQL**
- READ COMMITTED ↑ лучше производительность
- REPEATABLE READ
- SERIALIZABLE ↓ «корректнее» данные

Уровни изоляции в БД

Уровень изоляции	*«Грязное» чтение (dirty read)*	*Неповторяемое чтение (nonrepeatable read)*	*Фантомное чтение (phantom read)*	*Аномалия сериализации (serialization anomaly)*
READ UNCOMMITTED (Чтение Незафиксированных данных)	Допускается, но не в PG	Возможно	Возможно	Возможно
READ COMMITTED (Чтение зафиксированных данных)	Невозможно	Возможно	Возможно	Возможно
REPEATABLE READ (Повторяемое чтение)	Невозможно	Невозможно	Допускается, но не в PG	Возможно
SERIALIZABLE (Сериализуемость)	Невозможно	Невозможно	Невозможно	Невозможно

Феномен чтения – отличия при параллельном и изолированном выполнении транзакций

Уровни изоляции в БД

Уровень изоляции	*«Грязное» чтение (dirty read)*	*Неповторяемое чтение (nonrepeatable read)*	*Фантомное чтение (phantom read)*	*Аномалия сериализации (serialization anomaly)*
READ UNCOMMITTED (Чтение Незафиксированных данных)	Допускается, но не в PG	Возможно	Возможно	Возможно
READ COMMITTED (Чтение зафиксированных данных)	Невозможно	Возможно	Возможно	Возможно
REPEATABLE READ (Повторяемое чтение)	Невозможно	Невозможно	Допускается, но не в PG	Возможно
SERIALIZABLE (Сериализуемость)	Невозможно	Невозможно	Невозможно	Невозможно

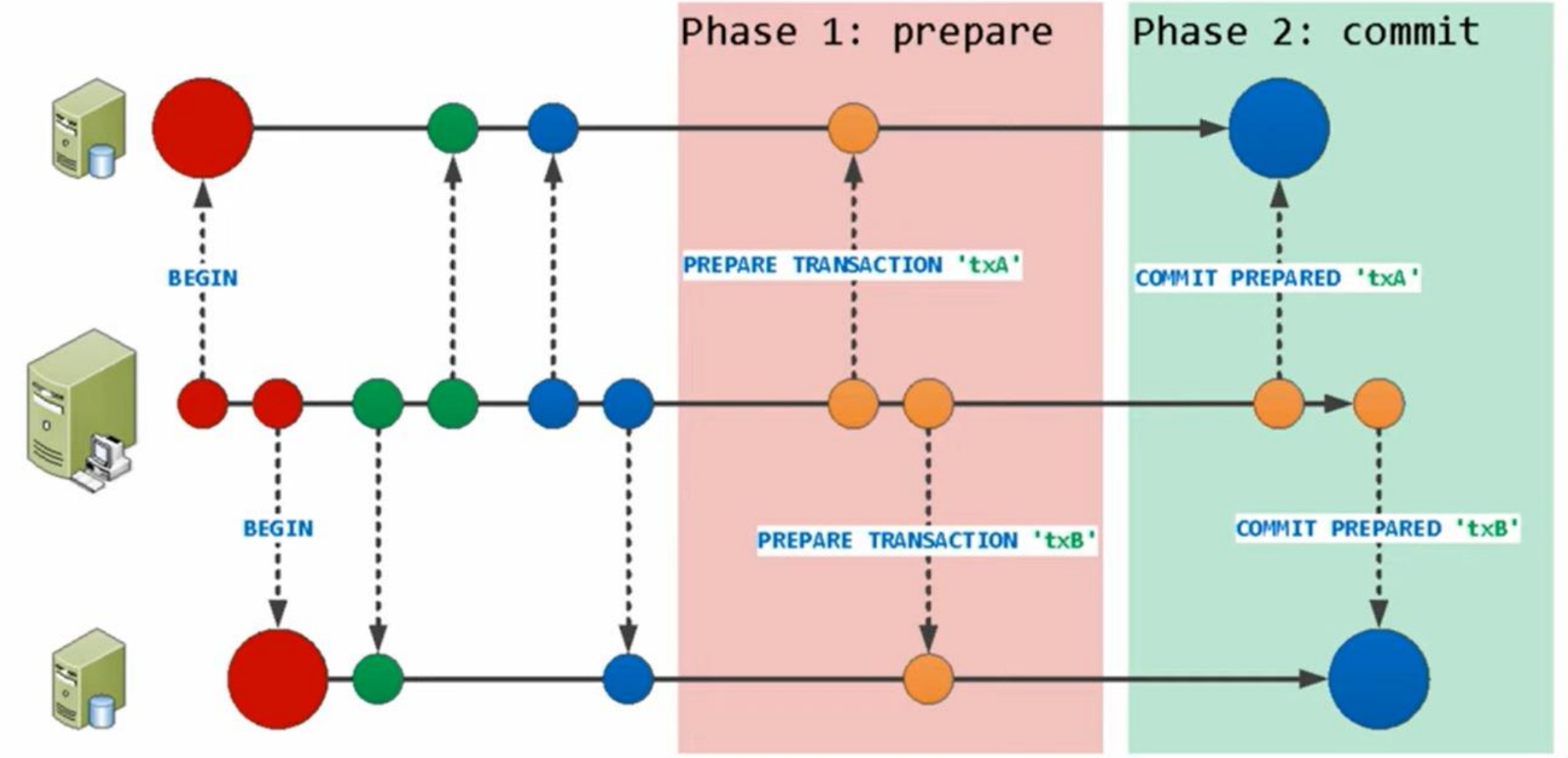
Феномен чтения – отличия при параллельном и изолированном выполнении транзакций

Уровни изоляции в различных БД

ISOLATION LEVEL \ DATABASE	MySQL	PostgreSQL	Microsoft SQL Server	ORACLE
Read Uncommitted	Supported	Not Supported (Work as Read Committed)	Supported	Not Supported
Read Committed	Supported	Default	Default	Default
Repeatable Read	Default	Supported	Supported	Not Supported
Snapshot	Not Supported	Not Supported	Supported	Not Supported
Serializable	Supported	Supported	Supported	Supported

Двухфазные транзакции (блокировки, коммиты)

2PC



Двухфазные транзакции (блокировки, коммиты)

2PC

PREPARE TRANSACTION id – подготовить транзакцию для 2PC

COMMIT PREPARED id – зафиксировать 2PC-транзакцию

ROLLBACK PREPARED id – отменить 2PC-транзакцию