

The background features two sets of abstract, concentric, wavy orange lines on a dark grey background. One set is in the upper right corner, and another is in the lower right corner. The lines are smooth and flowing, creating a sense of movement and depth.


Транзакции

и управление целостностью данных

Свойства транзакций (ACID)

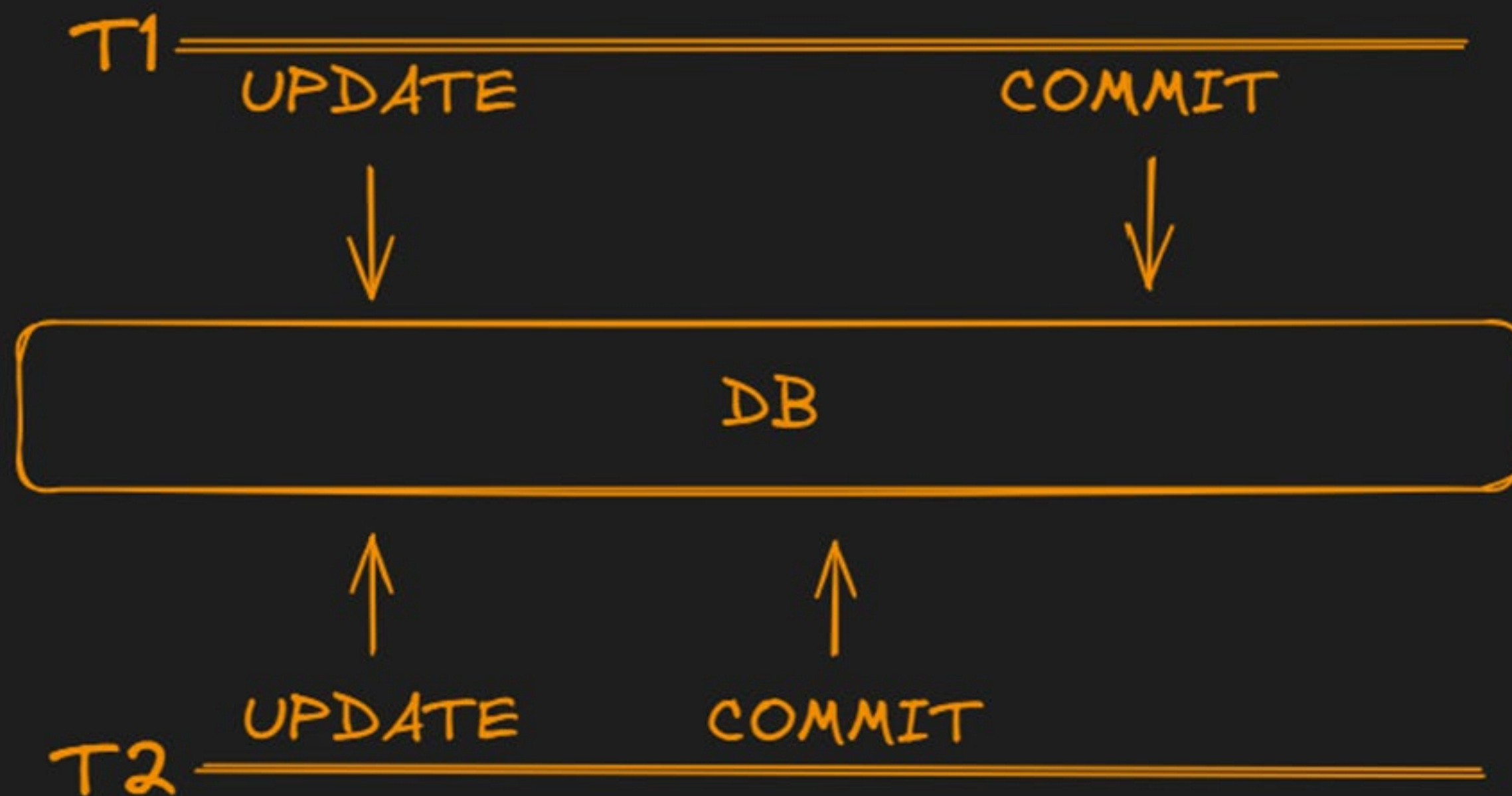
1. Atomicity (атомарность)
2. Consistency (согласованность)
3. Isolation (изолированность)
4. Durability (надёжность)



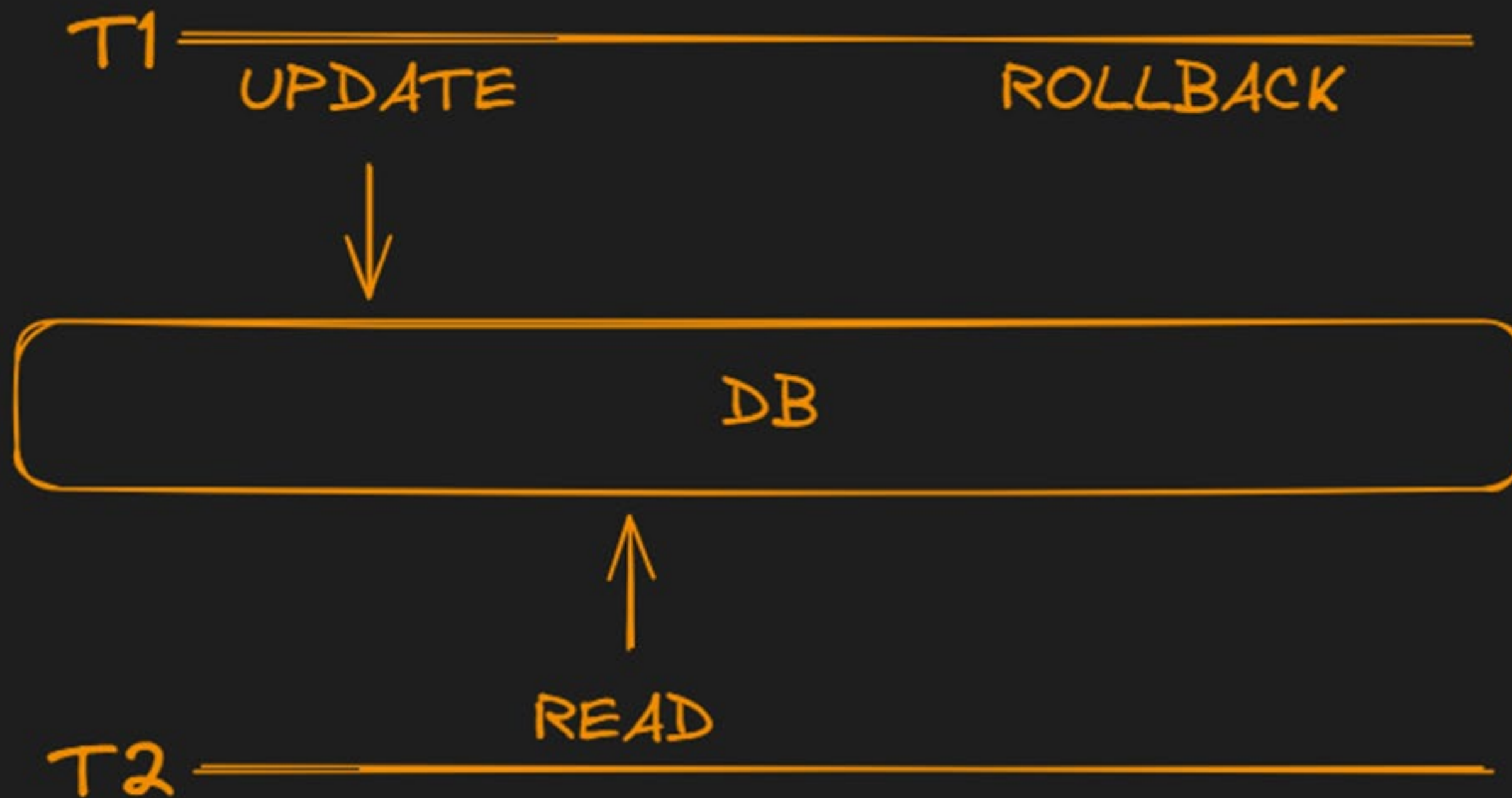


Феномены параллельного выполнения транзакций

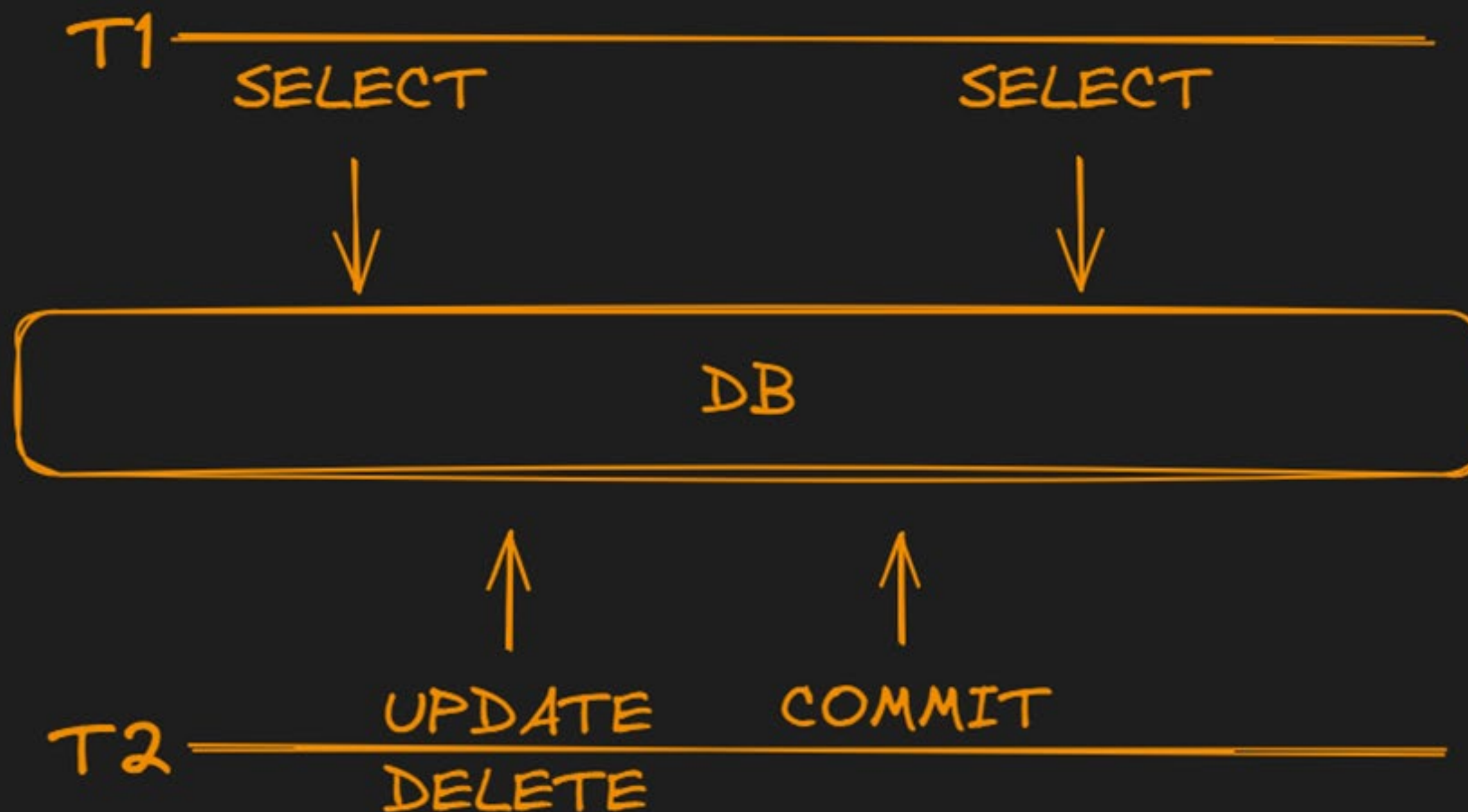
1. Потерянное обновление



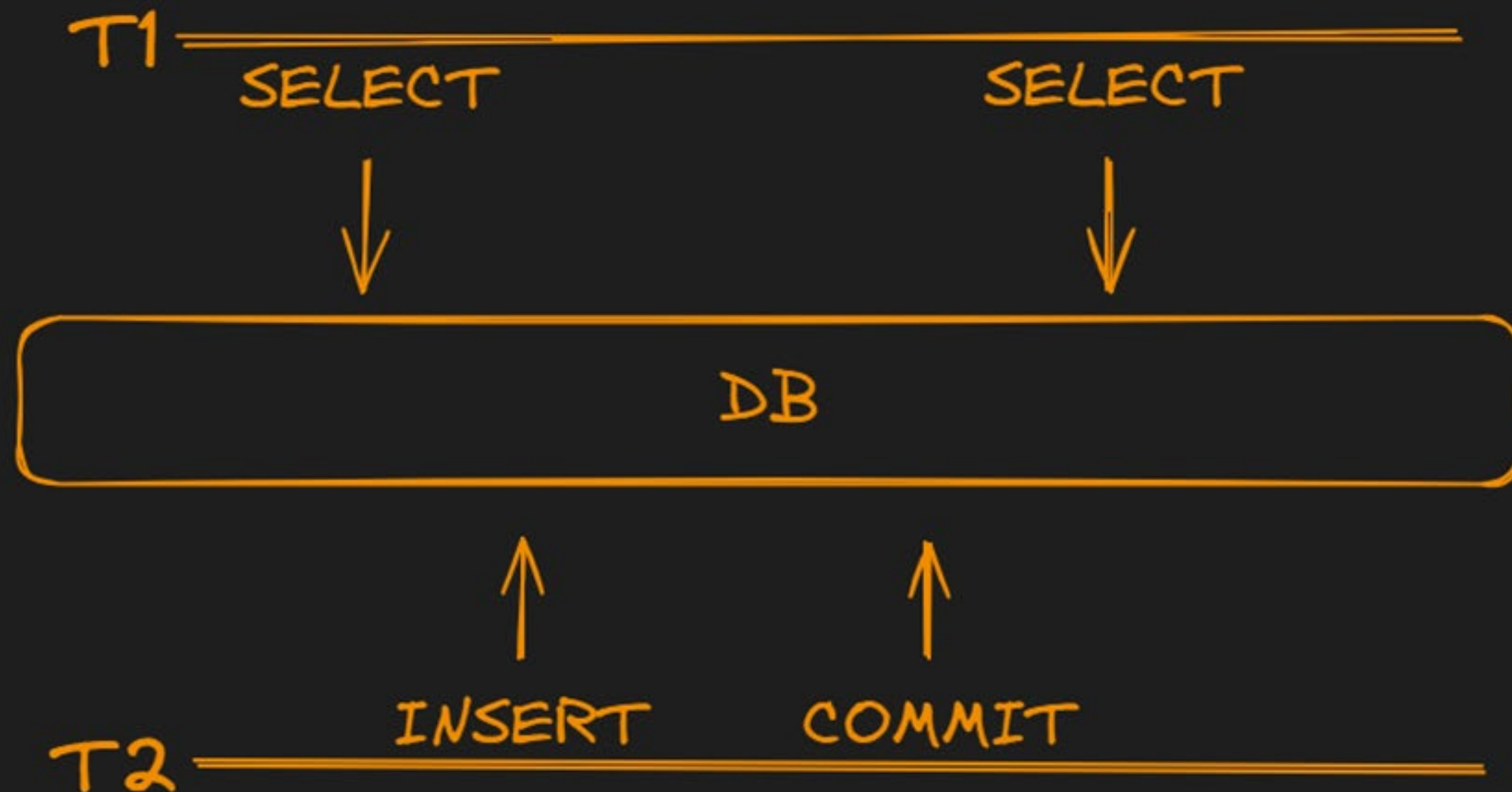
2. Грязное чтение



3. Неповторяющееся чтение



4. Фантомное чтение



5. Аномалия сериализации



Уровни изоляции

Уровень изоляции	Dirty Read	Non-repeatable Read	Phantom Read
READ UNCOMMITTED	Нет (в PostgreSQL — Нет)	Да	Да
READ COMMITTED	Нет	Да	Да
REPEATABLE READ	Нет	Нет	Нет (в PostgreSQL — устраняется MVCC)
SERIALIZABLE	Нет	Нет	Нет

PostgreSQL = MVCC, поэтому Repeatable Read \approx Serializable



MVСС
(Многоверсионность)

Основные принципы работы MVCC

1. Многоверсионность данных

- Модификации создают новые версии строк (row versions)
- Предыдущие версии сохраняются для обеспечения изоляции

2. Транзакционный снимок (snapshot)

- Каждая транзакция работает с состоянием БД на момент своего старта
- Чтения видят только данные, зафиксированные до начала транзакции

3. Неблокирующая модель доступа

- Чтения не требуют блокировок (no read locks)
- Параллельные транзакции не конфликтуют на уровне чтения

Как PostgreSQL хранит версии строк?

Каждая строка содержит скрытые системные поля:

Поле	Описание
<code>xmin</code>	ID транзакции, создавшей эту версию строки
<code>xmax</code>	ID транзакции, удалившей или заменившей строку

Принцип видимости строк:

`xmin > ID текущей транзакции` → строка не видна

`xmax` указывает на активную транзакцию → строка заблокирована/удаляется

`xmax` указывает на завершённую транзакцию → строка невидима

Условная аналогия



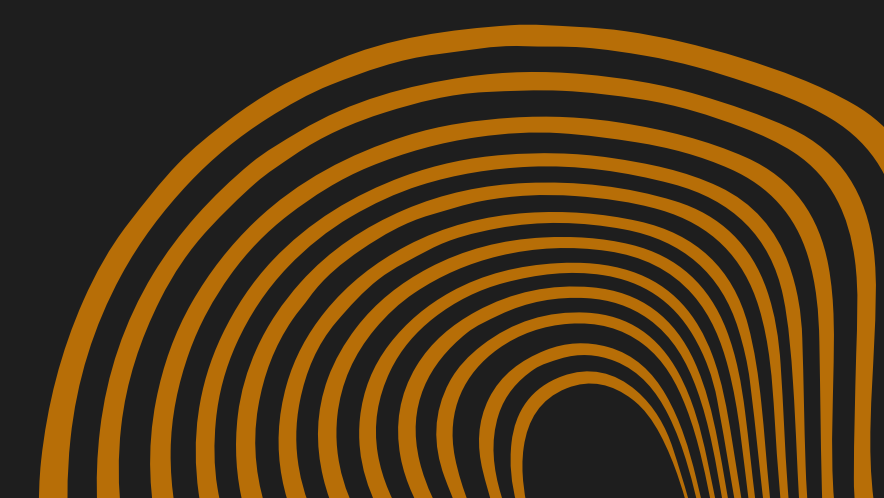


Конфликты и блокировки в MVCC

MVCC(ного версионность) в PostgreSQL обеспечивает:

- Параллельное выполнение запросов без конфликтов при чтении (SELECT).
- Каждая транзакция работает со своим снимком данных (snapshot).

Но при изменениях данных (INSERT, UPDATE, DELETE) блокировки необходимы:

- Блокировка на уровне строки – если одна транзакция изменяет строку, другая ждёт её завершения.
 - Блокировки на уровне таблицы – автоматически применяются для операций вроде TRUNCATE, ALTER TABLE.
 - Явные блокировки – можно задать командой LOCK.
- 

Режимы блокировок и взаимоблокировки

Режимы блокировок и взаимоблокировки

Уровни строгости блокировок:

- Слабые (SELECT) – не мешают другим транзакциям.
- Строгие (UPDATE, DELETE) – ограничивают параллельные изменения.
- Эксклюзивные (TRUNCATE, DROP) – полностью исключают доступ.

МСС+блокировки = баланс между параллелизмом и целостностью данных.

