

## Транзакции ACID

- Транзакция — это последовательность операций с базой данных, которая выполняется как единое целое. Транзакция либо завершается успешно (COMMIT), либо полностью отменяется (ROLLBACK), чтобы гарантировать целостность данных.

## Транзакции ACID

- Atomicity — Атомарность
- Consistency — Согласованность
- Isolation — Изолированность
- Durability — Надёжность

## Транзакции ACID

- Consistency (Согласованность) – После завершения транзакции база данных остается в согласованном состоянии, то есть не возникает нарушений ограничений целостности.

## Транзакции

### Ограничение целостности

- NOT NULL – запрет NULL значений
- UNIQUE – уникальность значений
- PRIMARY KEY – первичный ключ (NOT NULL + UNIQUE)
- FOREIGN KEY – внешний ключ (связь между таблицами)
- CHECK – проверка условий
- EXCLUSION CONSTRAINTS – расширенные ограничения

## Транзакции ACID

- **Isolation (Изоляция)** — Параллельные транзакции выполняются так, как если бы они были последовательными, предотвращая конфликты данных.

## Транзакции

### Изоляция транзакций

- **READ UNCOMMITTED** – Позволяет читать даже незавершённые изменения других транзакций (не используется в PostgreSQL).
- **READ COMMITTED** (по умолчанию) – Транзакция видит только зафиксированные (COMMIT) изменения других транзакций.
- **REPEATABLE READ** – Транзакция видит данные в том виде, в каком они были на момент её начала, даже если другие транзакции вносят изменения.
- **SERIALIZABLE** – Максимальный уровень изоляции. Гарантирует, что параллельные транзакции выполняются так, как если бы они шли последовательно.

## Транзакции

### Изоляция транзакций

Isolation Level	Dirty Read	Nonrepeatable Read	Phantom Read	Serialization Anomaly
Read uncommitted	Allowed, but not in PG	Possible	Possible	Possible
Read committed	Not possible	Possible	Possible	Possible
Repeatable read	Not possible	Not possible	Allowed, but not in PG	Possible
Serializable	Not possible	Not possible	Not possible	Not possible

## Транзакции

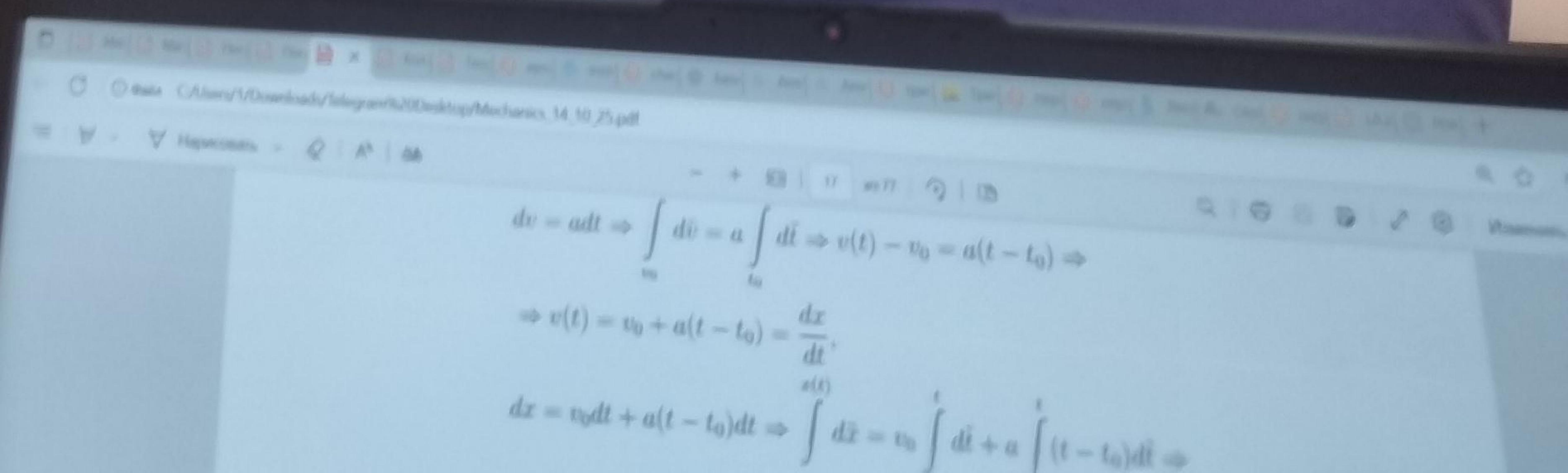
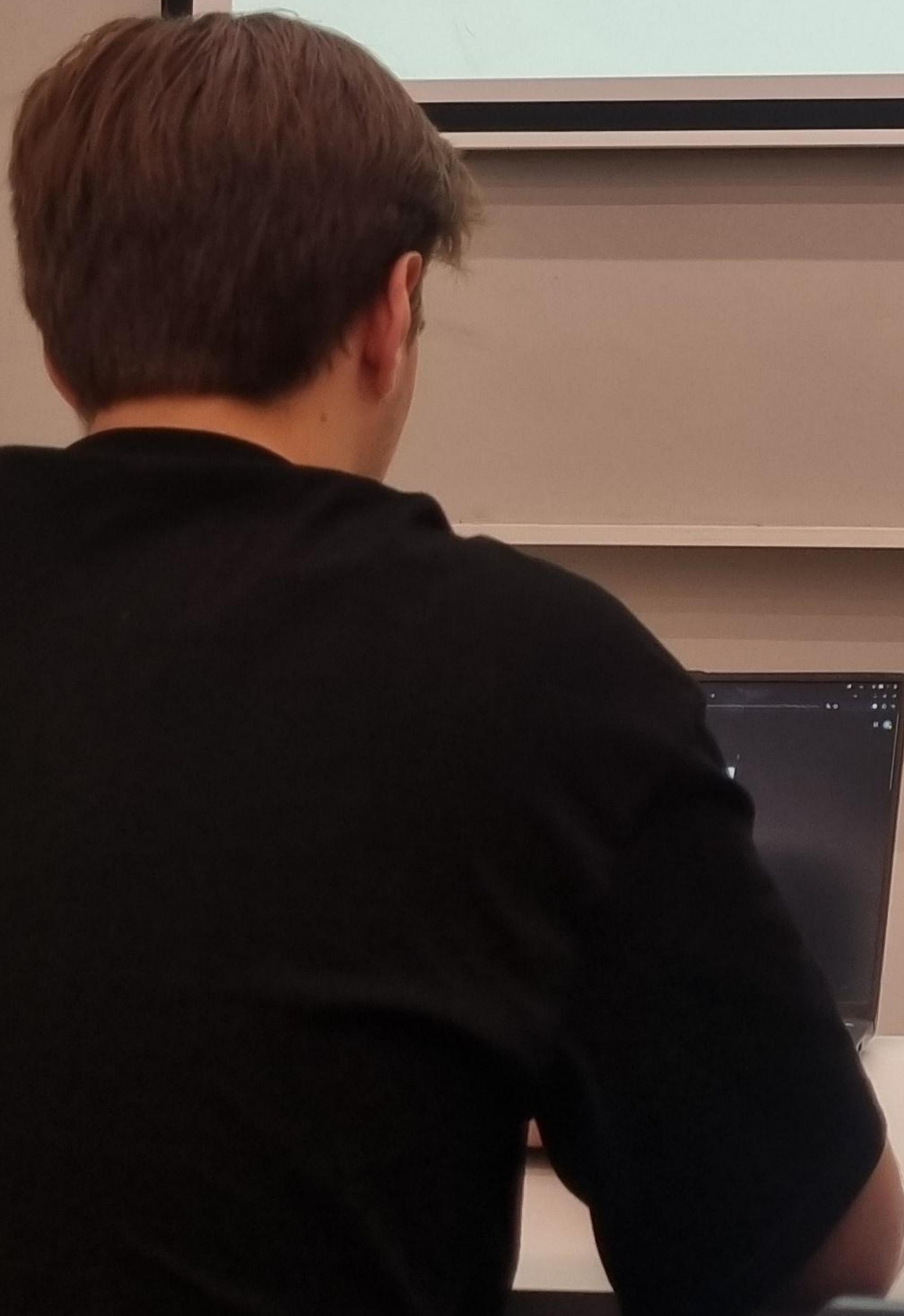
Никогда не увидим

- Потерянное обновления - такая аномалия возникает, когда две транзакции читают одну и ту же строку таблицы, затем одна транзакция обновляет эту строку, а после этого вторая транзакция тоже обновляет ту же строку, не учитывая изменений, сделанных первой транзакцией.
- Грязное чтение - такая аномалия возникает, когда транзакция читает еще не зафиксированные изменения, сделанные другой транзакцией.

## Транзакции

### Изоляция транзакций

- READ UNCOMMITTED — Позволяет читать даже незавершённые изменения других транзакций (не используется в PostgreSQL).
- READ COMMITTED (по умолчанию) — Транзакция видит только зафиксированные (COMMIT) изменения других транзакций.
- REPEATABLE READ — Транзакция видит данные в том виде, в каком они были на момент её начала, даже если другие транзакции вносят изменения.
- SERIALIZABLE — Максимальный уровень изоляции. Гарантирует, что параллельные транзакции выполняются так, как если бы они шли последовательно.



## Транзакции

### Фантомное чтение

- Фантомное чтение возникает, когда транзакция два раза читает набор строк по одному и тому же условию, и в промежутке между чтениями вторая транзакция добавляет строки, удовлетворяющие этому условию (и фиксирует изменения). Тогда первая транзакция получит разные наборы строк.



## Транзакции Serializable

- Уровень Serializable должен предотвращать вообще все аномалии. Это означает, что на таком уровне разработчику приложения не надо думать об одновременном выполнении. Если транзакции выполняют корректные последовательности операторов, работая в одиночку, данные будут согласованы и при одновременной работе этих транзакций.

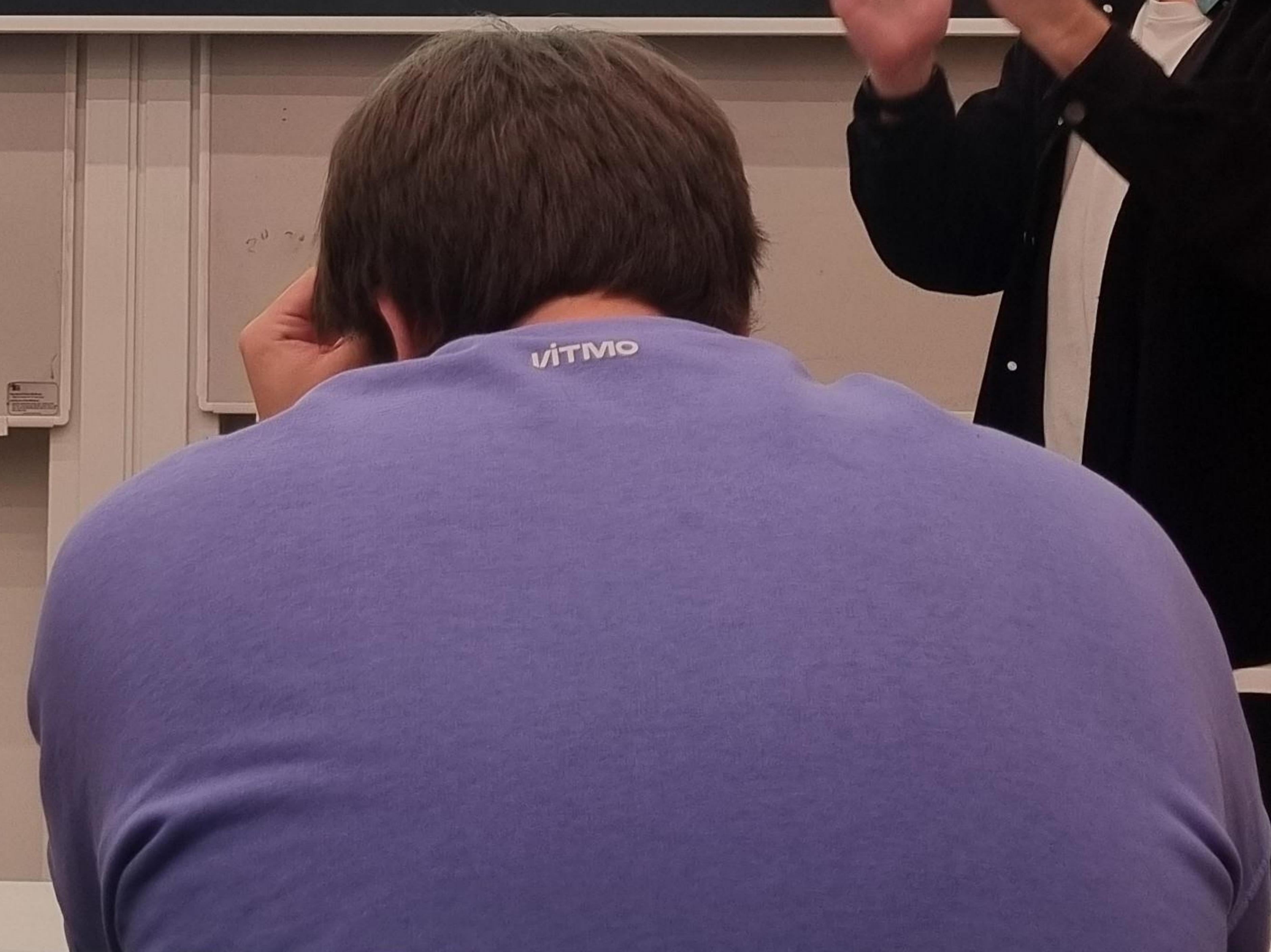
# Транзакции

## Почему именно такие уровни в стандарте?

- Разница между уровнями изоляции стандарта объясняется как раз количеством необходимых блокировок.
- Если транзакция блокирует изменяемые строки от изменения, но не от чтения, получаем уровень Read Uncommitted:
- Если транзакция блокирует изменяемые строки и от чтения, и от изменения, получаем уровень Read Committed
- Если транзакция блокирует читаемые, и изменяемые строки и от чтения, и от изменения, получаем уровень Repeatable Read.
- Но с Serializable проблема: невозможно заблокировать строку, которой еще нет. Из-за этого остается возможность фантомного чтения. Поэтому для реализации уровня Serializable обычных блокировок не хватает — нужно блокировать не строки, а условия (предикаты). Такие блокировки и были названы предикатными.

## Write-Ahead Logging

- стратегия, при которой все изменения данных сначала записываются в журнал, и только потом применяются к основным файлам базы данных.



A laptop screen displays a PDF document containing mathematical derivations for motion equations. The equations show the derivation of velocity and position from initial conditions and acceleration.

$$dv = adt \Rightarrow \int_{v_0}^{v(t)} dv = a \int_{t_0}^t dt \Rightarrow v(t) - v_0 = a(t - t_0) \Rightarrow$$
$$\Rightarrow v(t) = v_0 + a(t - t_0) = \frac{dx}{dt}$$
$$dx = v_0 dt + a(t - t_0) dt \Rightarrow \int_{x_0}^{x(t)} dx = v_0 \int_{t_0}^t dt + a \int_{t_0}^t (t - t_0) dt \Rightarrow$$
$$\Rightarrow x(t) = x_0 = v_0(t - t_0) + \frac{a(t - t_0)^2}{2},$$
$$x(t) = x_0 + v_0(t - t_0) + \frac{a(t - t_0)^2}{2} \quad (2.1)$$

Лабс

$$dv = adt \Rightarrow \int dv = a \int dt + const_1 \Rightarrow v = a \cdot t + C_1, \quad C_1 \equiv const_1 \quad (2.2)$$

## Структура WAL

- Header (24 байта)
- Page ID (8 байт)
- Old Data (Размер страницы)
- New Data (Размер страницы)
- Checksum (8 байт)

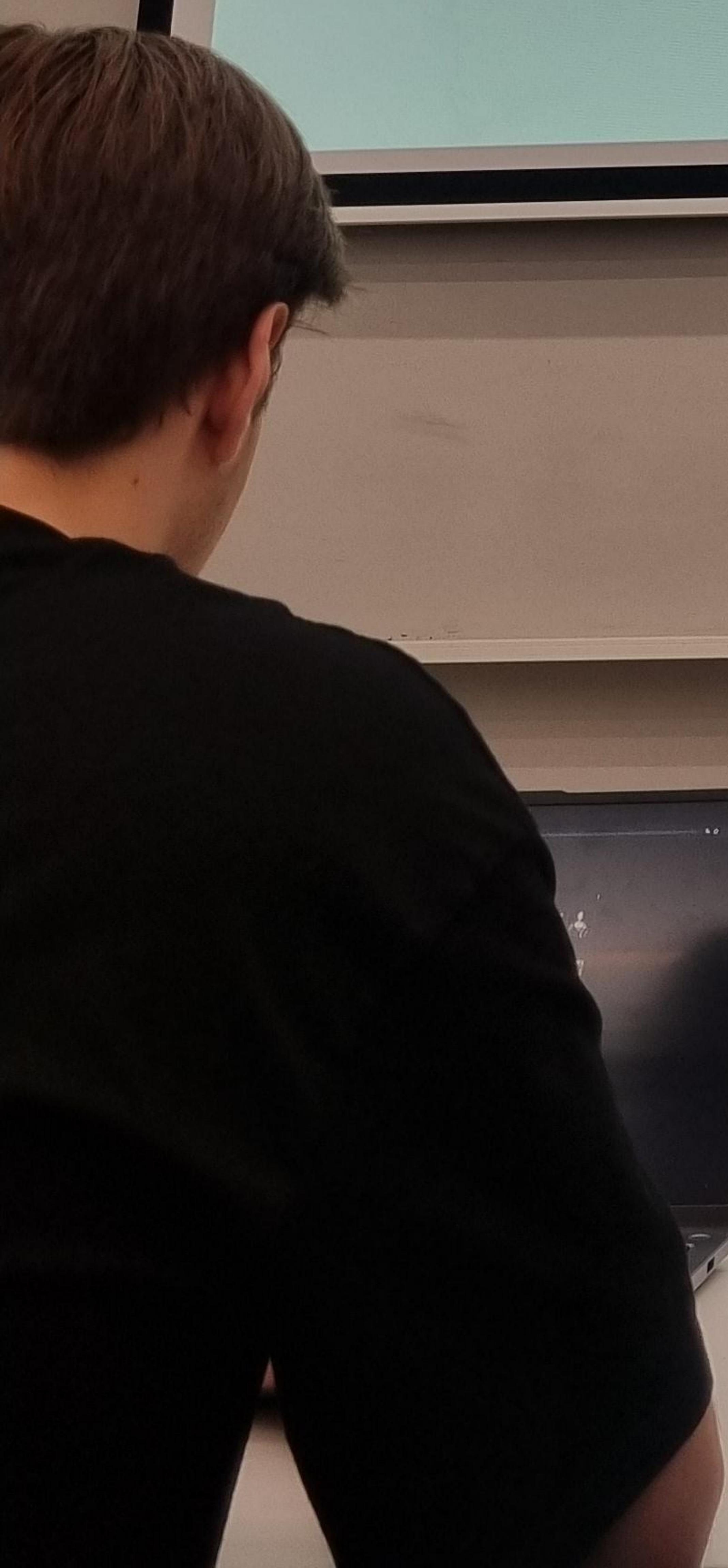
- `fsync = on`
- `synchronous_commit = on`
- `wal_buffers = 64MB`

## *synchronous\_commit*

- On
- Off
- Local
- Remote\_write
- Remote\_apply

## **Checkpoint**

- это процесс, при котором все измененные данные из буферного кэша сбрасываются на диск, создавая согласованное состояние базы данных.



xmin	xmax	
60	null	Vasya
100	106	Ivan
102	104	Mark
106	null	Ivonyshka

(x[i])

- 98 Select
- 100 Insert
- 101 Select
- 102 Insert
- 103 Select
- 104 Delete
- 105 Select
- 106 UPDATE
- 107 SELECT