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
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

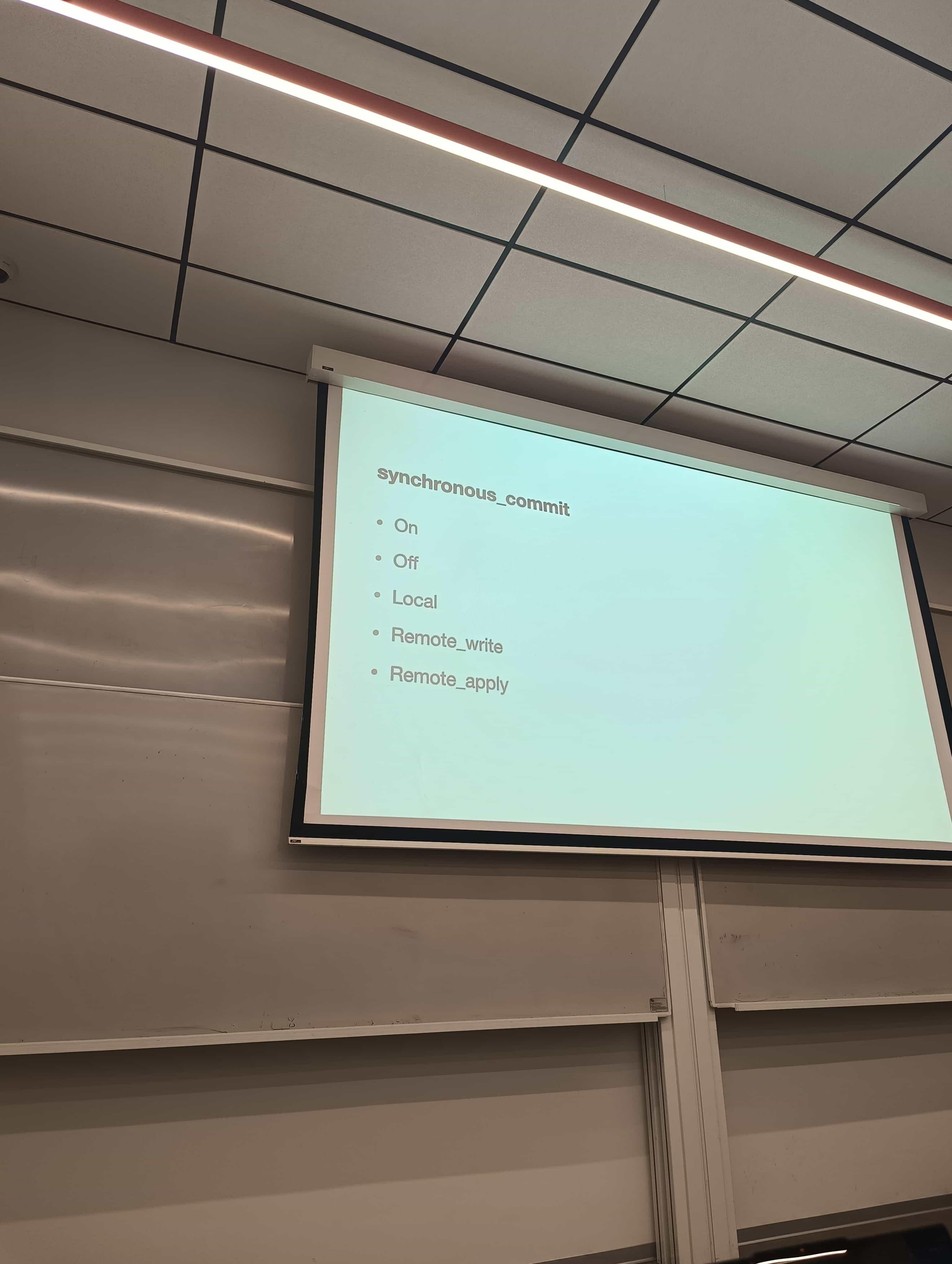
Почему именно такие уровни в стандарте?

- 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 и 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.

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