**Транзакция –** Единица работы, состоящая из множества операций, которые переводят БД из одного согласованного состояния в другое.

**ACID –** свойство транзакций, направленные на обеспечение надежности и целостности данных в условиях параллельного доступа.

*Atomicity* (Атомарность) – Транзакция не может быть зафиксирована частично, то есть не может иметь промежуточных состояний, она либо выполнена полностью, либо не выполнена вовсе.

*Consistency* (Согласованность) - свойство, гарантирующее, что каждая успешная транзакция зафиксирует только допустимые состояния. То есть, это гарантия того, что при успешной транзакции будут выполнены все правила, ограничения, которые предъявляет система к конкретным данным, иначе транзакция не будет выполнена и данные в системе вернутся к прежнему состоянию.

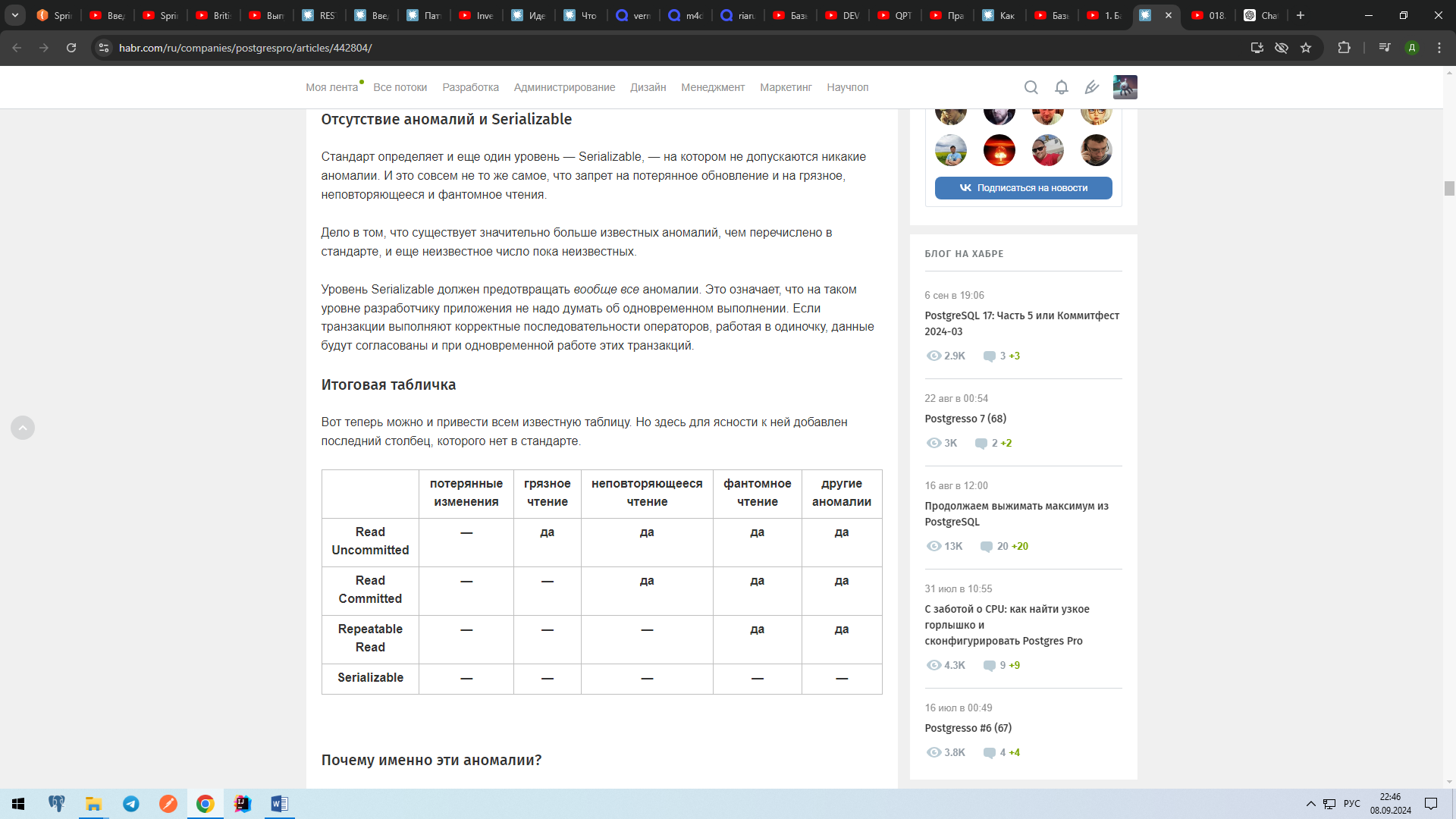
*Isolation* (Изолированность) – Параллельные транзакции (транзакции, работающие с одними и теми же данными) не должны влиять на результаты выполнения друг друга. Подразумевает уровни изоляции, которые регламентируют уровни изоляции транзакций, строгость которых уменьшает возможные побочные эффекты, но накладывают ограничения на производительность системы.

Durability (Надежность) – Если клиент получил подтверждение от системы подтверждение завершения транзакции (транзакция зафиксирована), то эти изменения не будут отменены из-за сбоя (не будут потеряны).

***Уровни изоляции транзакций:***

Стандарт определяет четыре уровня изоляции, эти уровни определяют аномалии, которые допускаются или не допускаются на этом уровне изоляции. Стандарт предполагает, что изоляции должна быть построена на блокировках строк, но на практике современные реляционные СУБД (включая PostgreSQL) используют протоколы изоляции на основе снимков – каждая транзакция работает с согласованным снимком данных в определенный момент времени.

Стандарт определяет следующее соответствие уровней изоляции и аномалий:



**Уровни изоляции в PostgreSQL:**

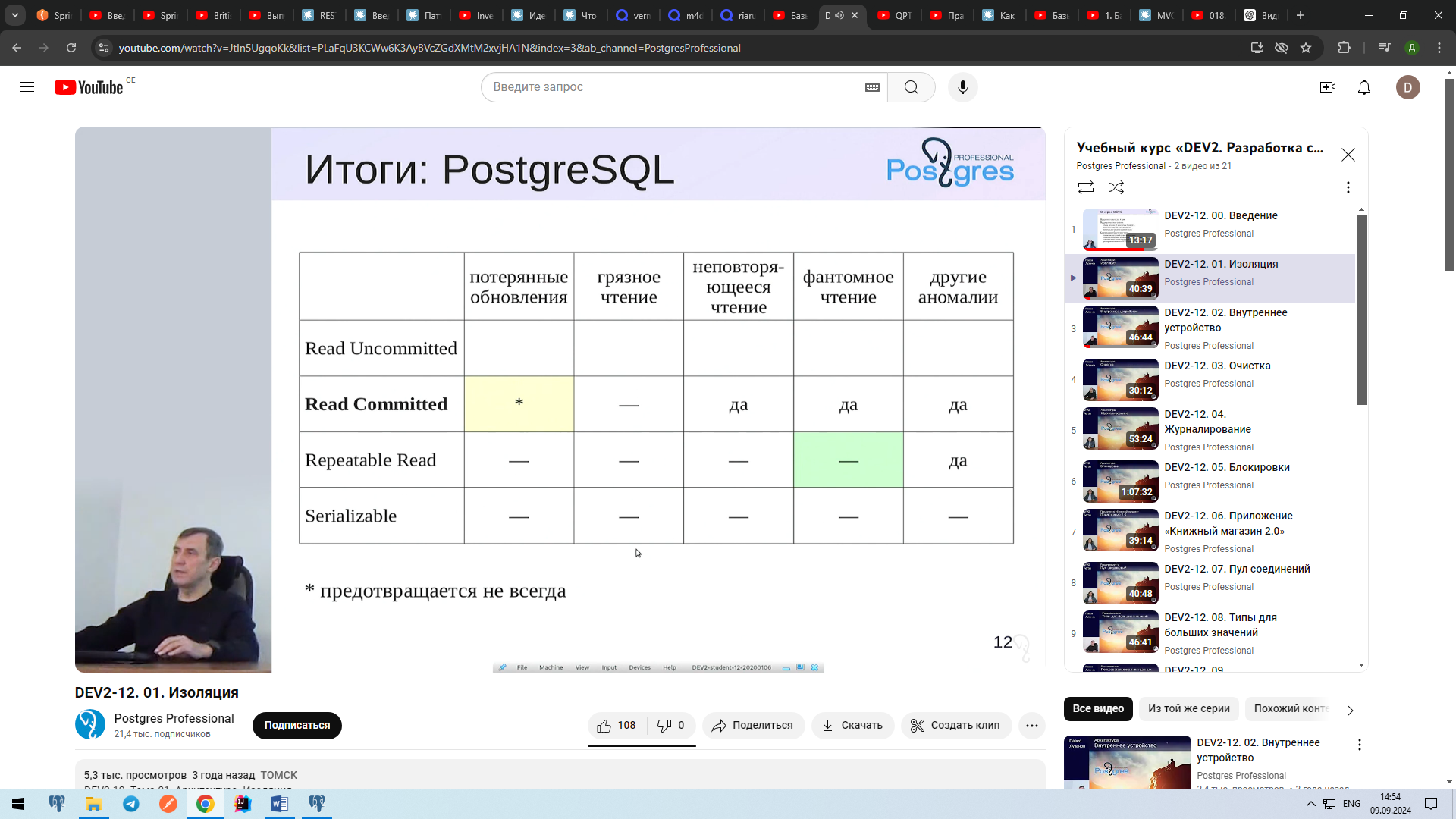
Использование протокола снимков в качестве альтернативы блокировкам делают уровни изоляции более строгими, чем того требует стандарт.

**Read uncommitted *–*** не реализован в PostgreSQL, т.к. протокол на основе снимков не допускает аномалию грязного чтения, поэтому номинально он существует, но полностью соответствует уровню read committed.

**Read committed** – Транзакции видят только подтвержденные изменения. Незавершенная транзакция может видеть закомиченные изменения параллельных транзакции в течение своего выполнения, если использует те же данные. По умолчанию в PostgreSQL.

**Repeatable read**– Транзакции не видят изменений в параллельных транзакциях в течение выполнения собственной.

**Serializable** – Максимальный уровень изоляции. Предусматривает отсутствие аномалий.



|  |  |
| --- | --- |
| **Show transaction\_isolation** | Отображает текущий уровень изоляции путем обращения к переменной |
| **Show transaction isolation level** | Также отображает текущий уровень изоляции |
| **Set local transaction isolation level <уровень изоляции>**  **Set session transaction isolation level <уровень изоляции>** | Установка уровня изоляции для текущей транзакции  Установка уровня изоляции для текущей сессии |
| **Begin transaction isolation level** **<уровень изоляции>** | Альтернативный способ задать локальный уровень изоляции во время начала транзакции |

**Аномалии:**

*Потерянное обновление* – Работает как race condition (при вычитывании 2мя транзакциями значения, будет видимо то изменение, которое будет наложено последним). Возможно в ситуации с использованием переменной в качестве временного хранения значения ([пример](https://youtu.be/JtIn5UgqoKk?list=PLaFqU3KCWw6K3AyBVcZGdXMtM2xvjHA1N&t=1340)).

*Грязное чтение* – Возможность транзакции видеть и опираться на данные, которые не были подтверждены в параллельной транзакции. Не допускается ни на одном уровне изоляции из-за протокола изоляции на основе снимков.

*Неповторяющееся чтение* – ситуация, когда транзакция при повторном чтении данных видит изменения, зафиксированные параллельными транзакциями.

*Фантомное чтение* – ситуация, когда повторное чтение в рамках одной транзакции возвращает отличный от первого чтения набор строк, по причине редактирования их количества в параллельной транзакции.

**Нестандартные аномалии:**

*Несогласованное чтение* – ситуация, когда совокупные данные в рамках одной транзакции имеют результат отличный от той же совокупности данных в рамках параллельной транзакции.

*\**[*Подробнее в материалах*](https://habr.com/ru/companies/postgrespro/articles/442804/)*, обратить внимание на*:

Несогласованное чтение при использовании нескольких SQL операторов  
Несогласованное чтение при одновременном обновлении (разрешается путем перечитывания строки на уровне read committed, а на уровне repeatable read отлетит с ошибкой, т.к. нет возможности перечитывать строки).  
Несогласованное чтение с VOLATILE

В материалах также приведены разрешения ситуаций при использовании различных уровней изоляции:

Аномалия несогласованного чтения наряду с неповторяющимся чтением возможны только на уровне read committed, из-за возможности чтения транзакцией измененных данных в течение своей транзакции. Некоторые аномалии возможно избежать путем использования одного SQL оператора.

Аномалии, присутствующие на repeatable read:

*Несогласованная запись*, Аномалия только читающей транзакции (также [см. материалы](https://habr.com/ru/companies/postgrespro/articles/442804/)).