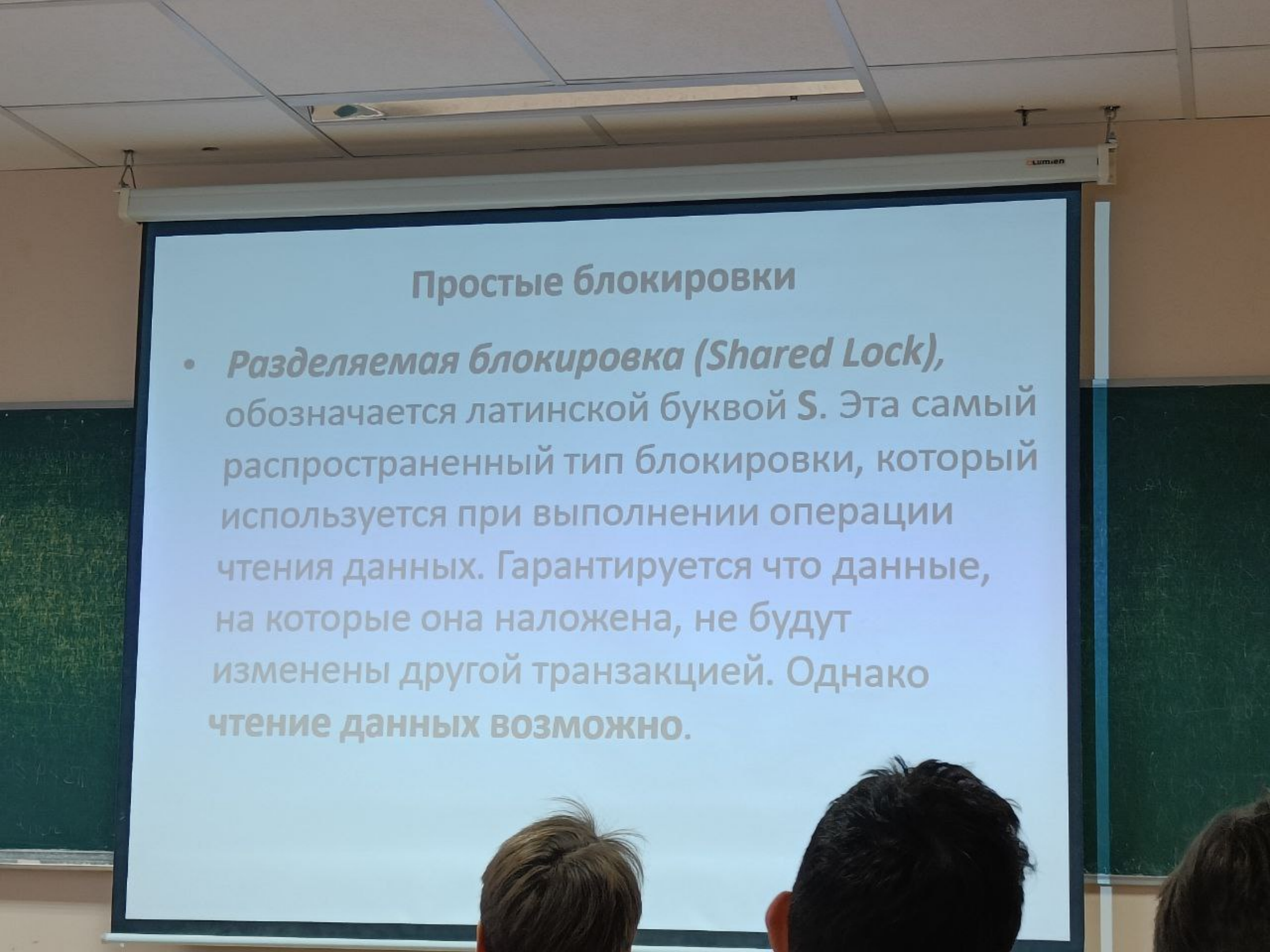
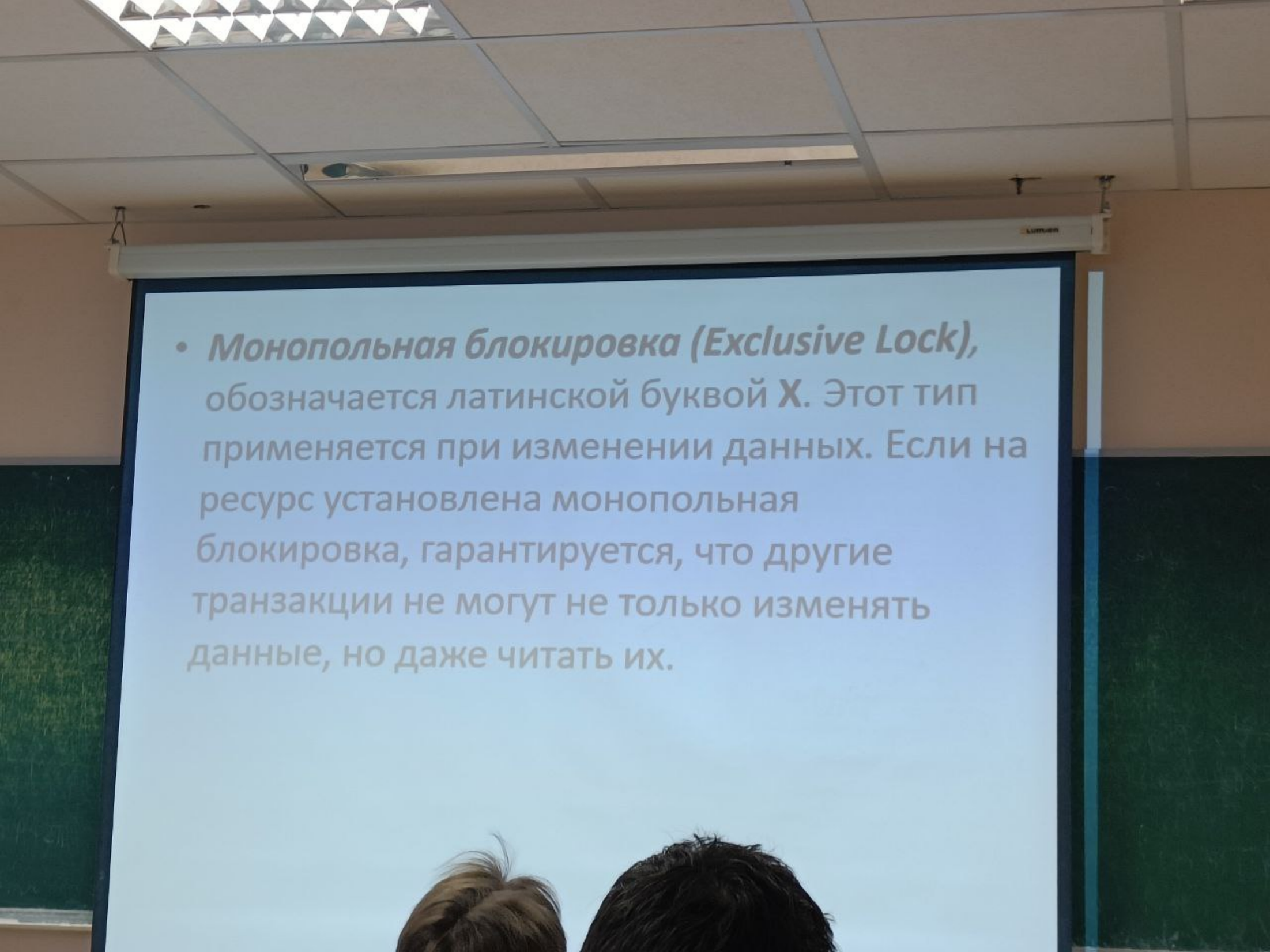
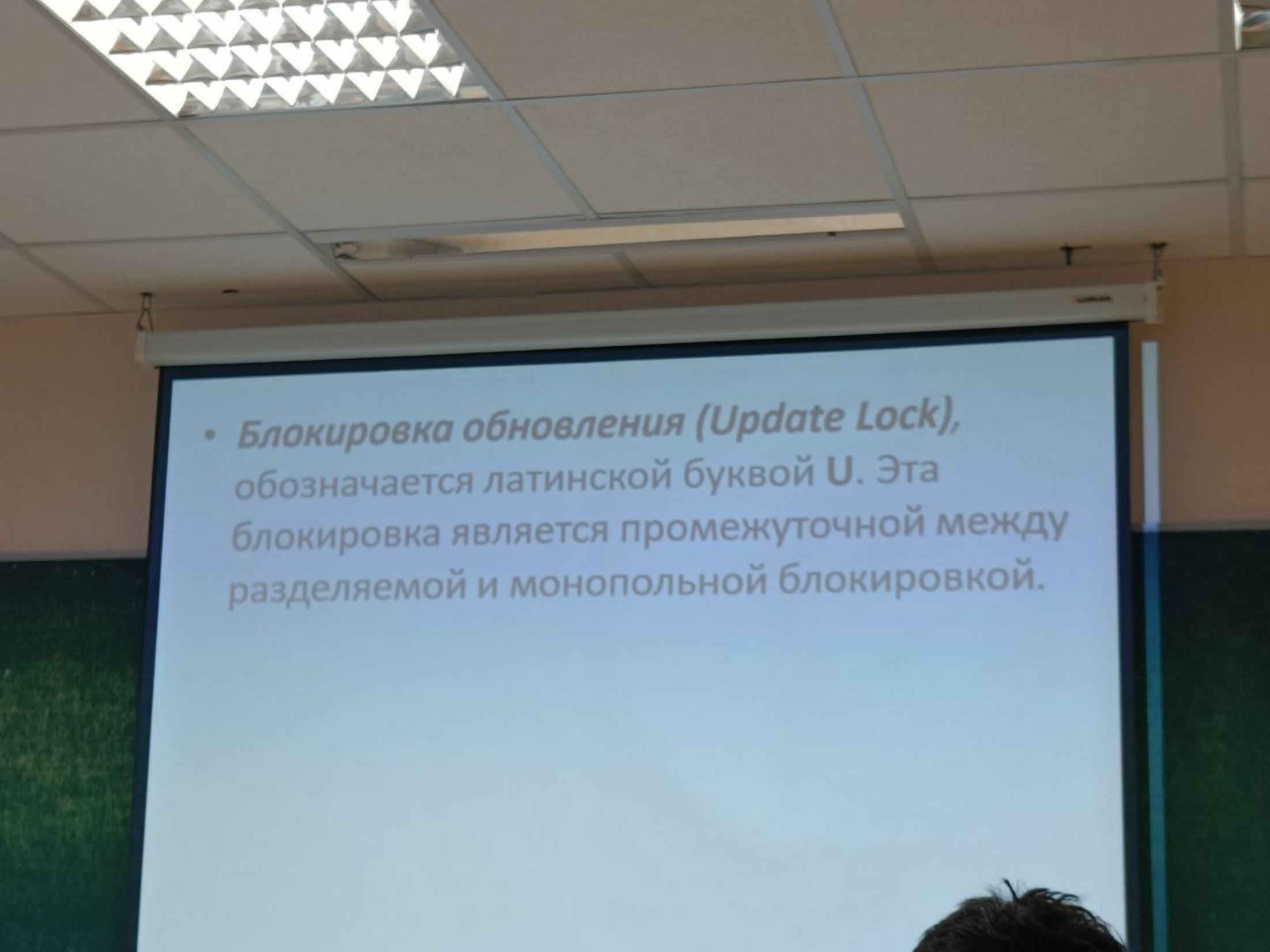
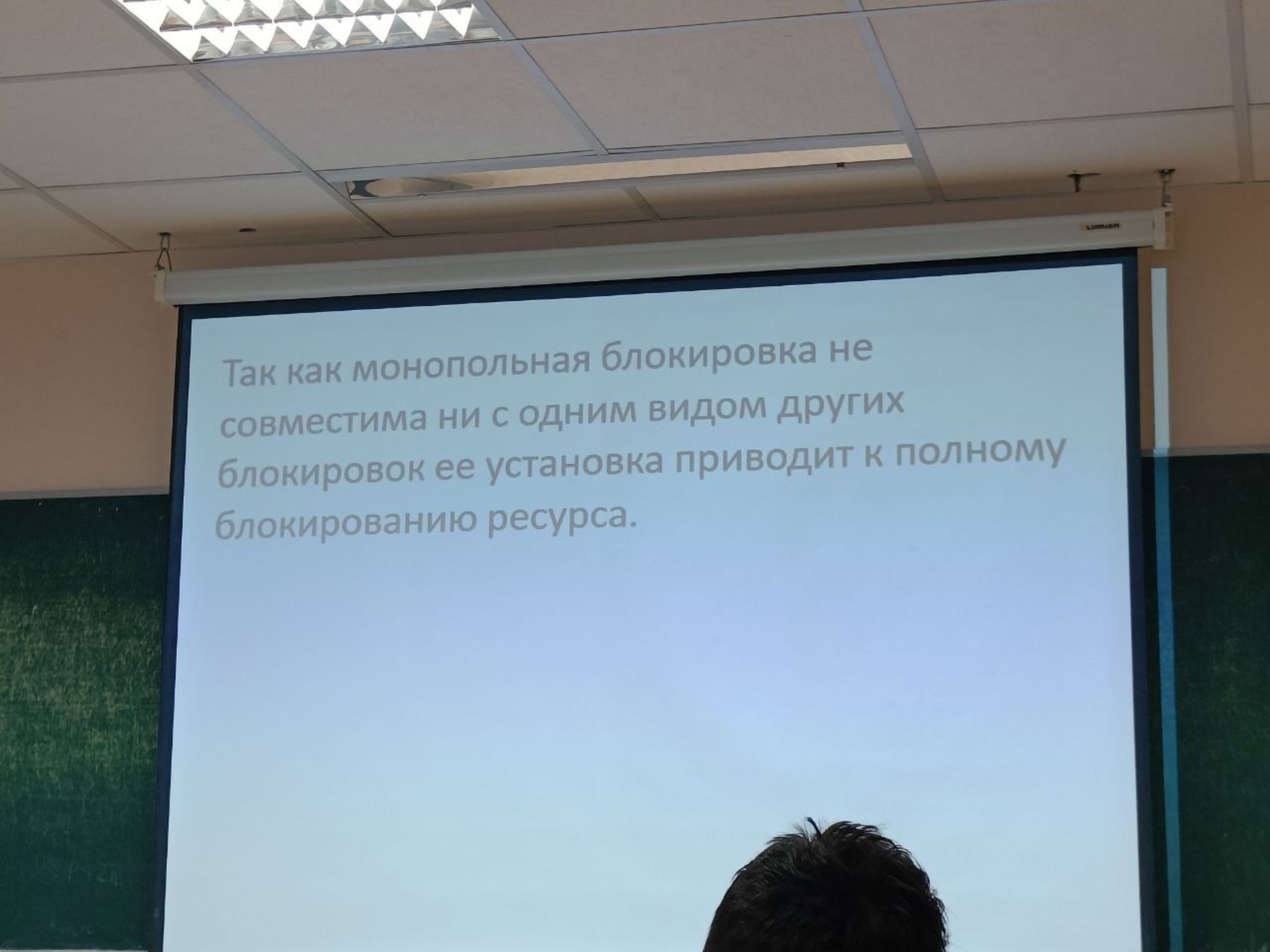

Простые блокировки
• Разделяемая блокировка (Shared Lock), обозначается латинской буквой S. Эта самый распространенный тип блокировки, который используется при выполнении операции чтения данных. Гарантируется что данные, на которые она наложена, не будут изменены другой транзакцией. Однако чтение данных возможно.

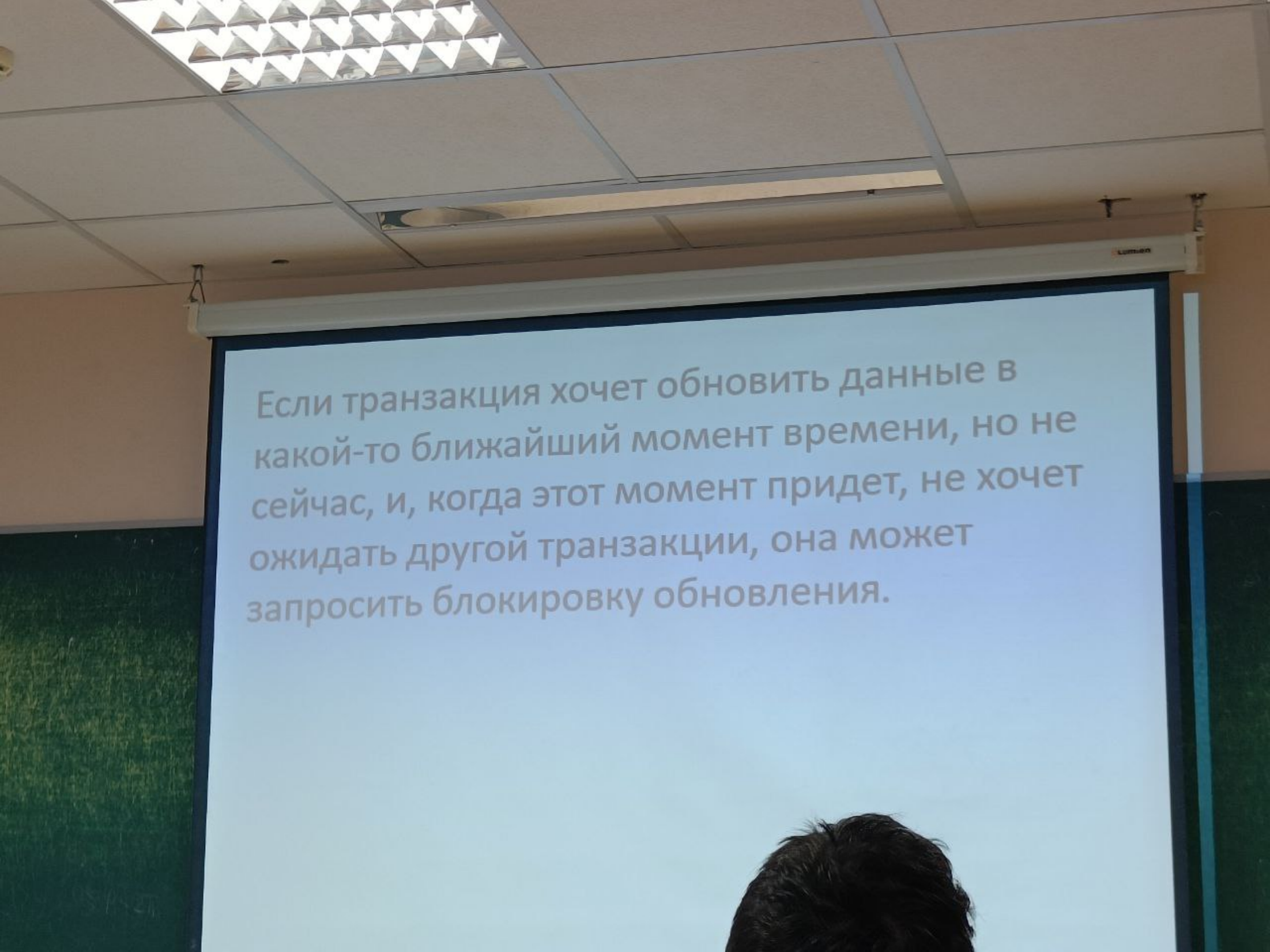
- ***Монопольная блокировка (Exclusive Lock),*** обозначается латинской буквой **X**. Этот тип применяется при изменении данных. Если на ресурс установлена монопольная блокировка, гарантируется, что другие транзакции не могут не только изменять данные, но даже читать их.

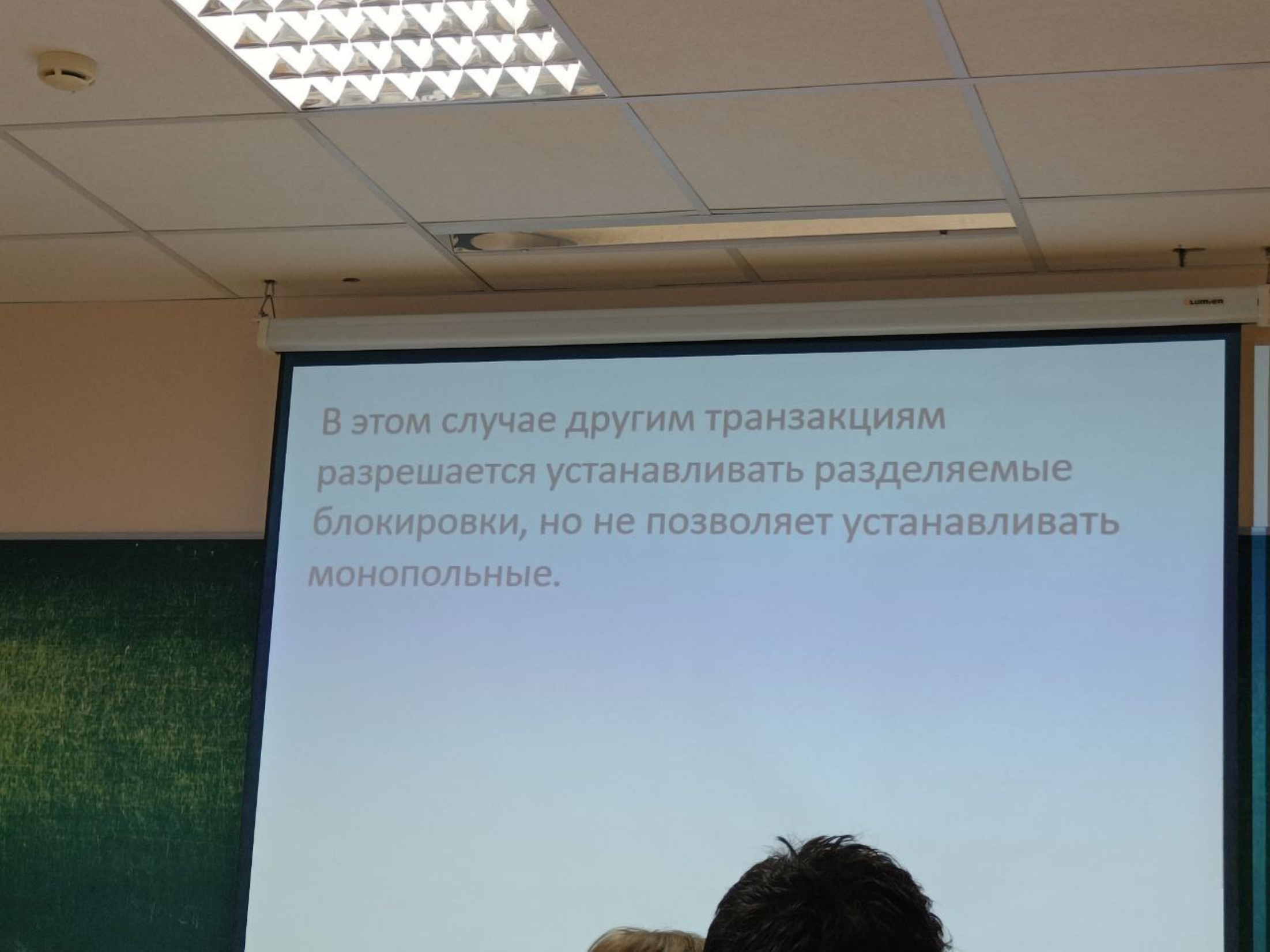



• Блокировка обновления (Update Lock), обозначается латинской буквой U. Эта блокировка является промежуточной между разделяемой и монопольной блокировкой.

Так как монопольная блокировка не совместима ни с одним видом других блокировок ее установка приводит к полному блокированию ресурса.



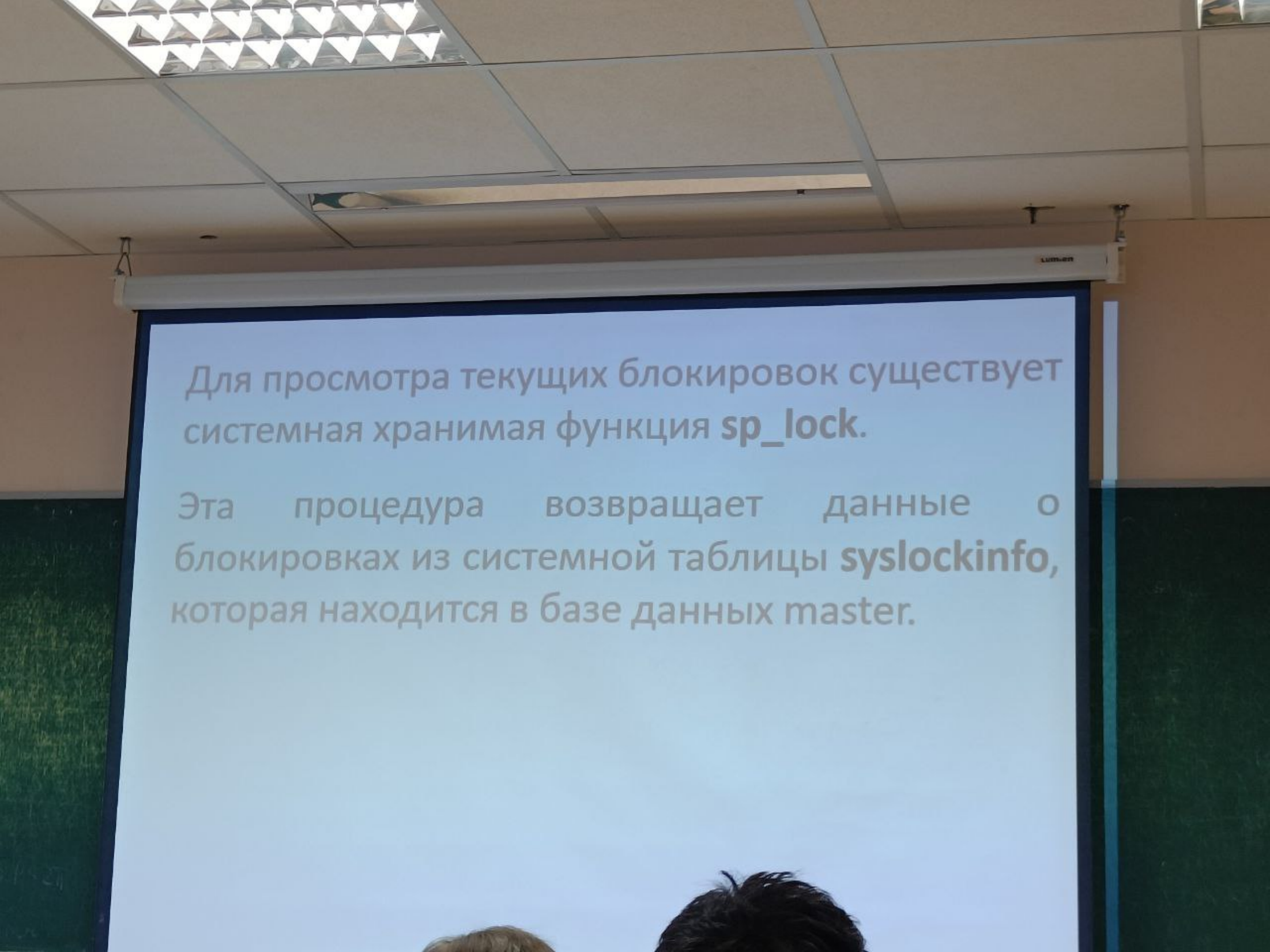

Если транзакция хочет обновить данные в какой-то ближайший момент времени, но не сейчас, и, когда этот момент придет, не хочет ожидать другой транзакции, она может запросить блокировку обновления.


В этом случае другим транзакциям разрешается устанавливать разделяемые блокировки, но не позволяет устанавливать монопольные.

Если данная транзакция установила на ресурс блокировку обновления, никакая другая транзакция не сможет получить на этот же ресурс монопольную блокировку или блокировку обновления до тех пор, пока установившая блокировку транзакция не будет завершена.

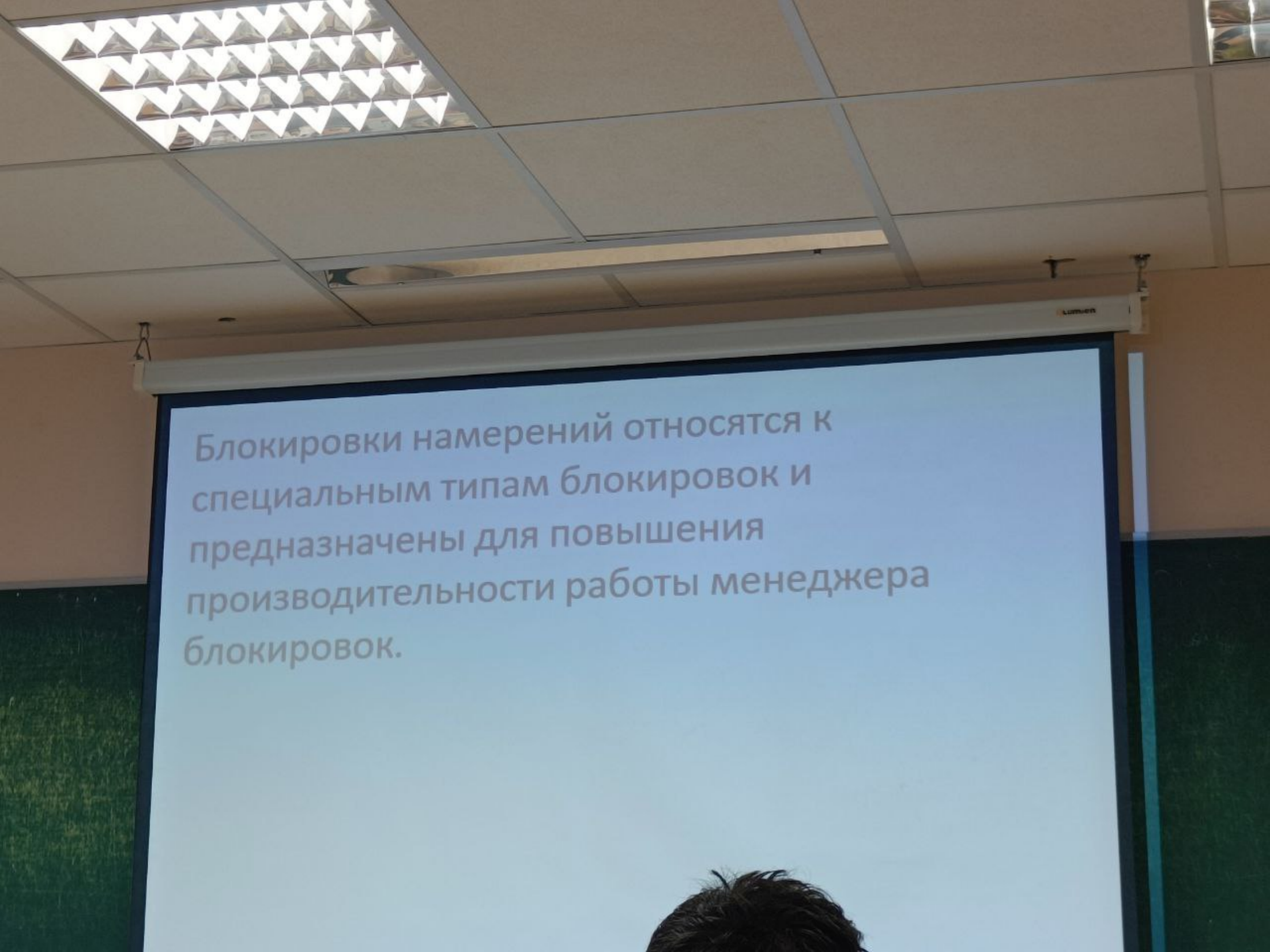
Для просмотра текущих блокировок существует системная хранимая функция **sp_lock**.

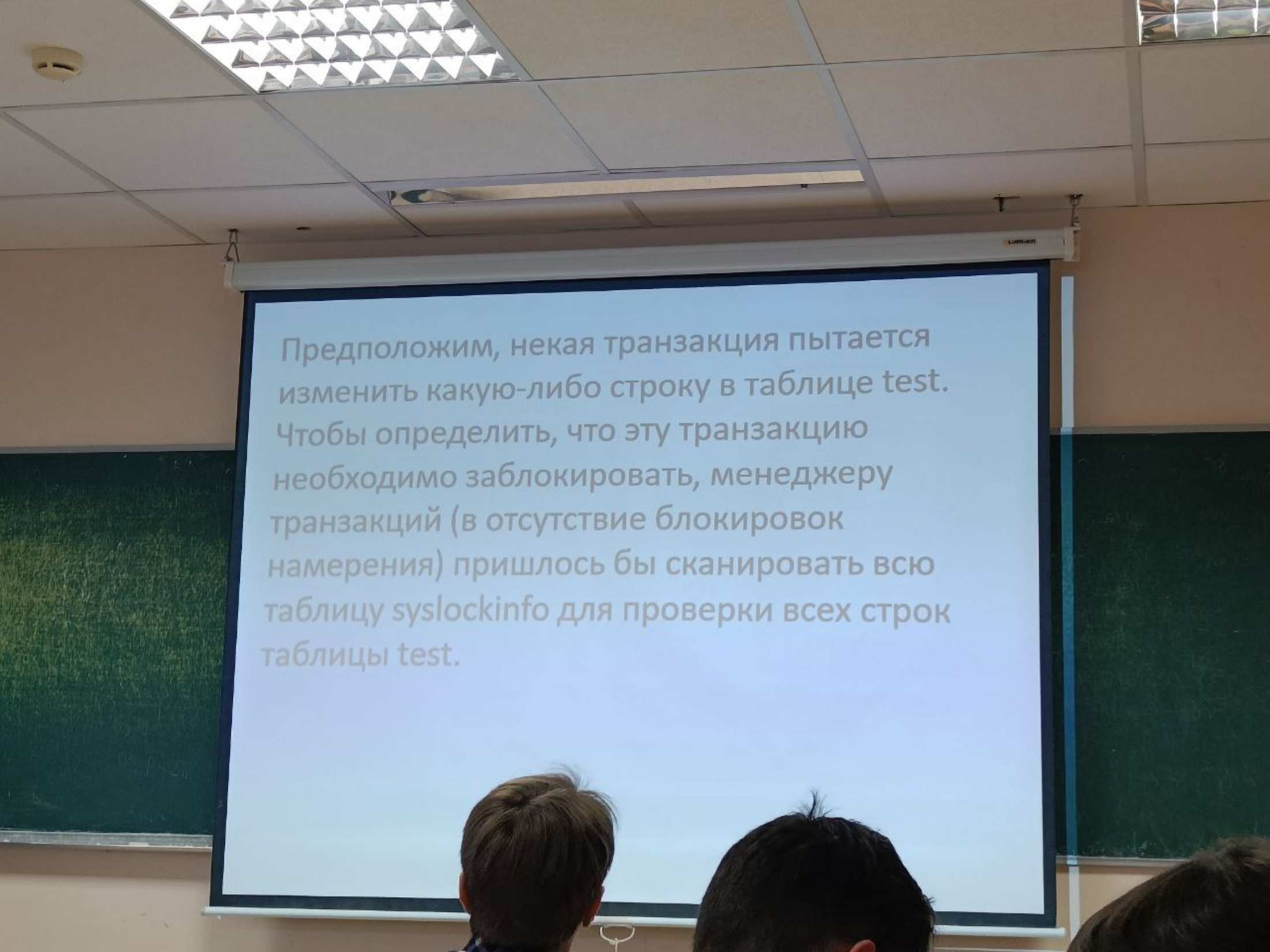
Эта процедура возвращает данные о блокировках из системной таблицы **syslockinfo**, которая находится в базе данных master.

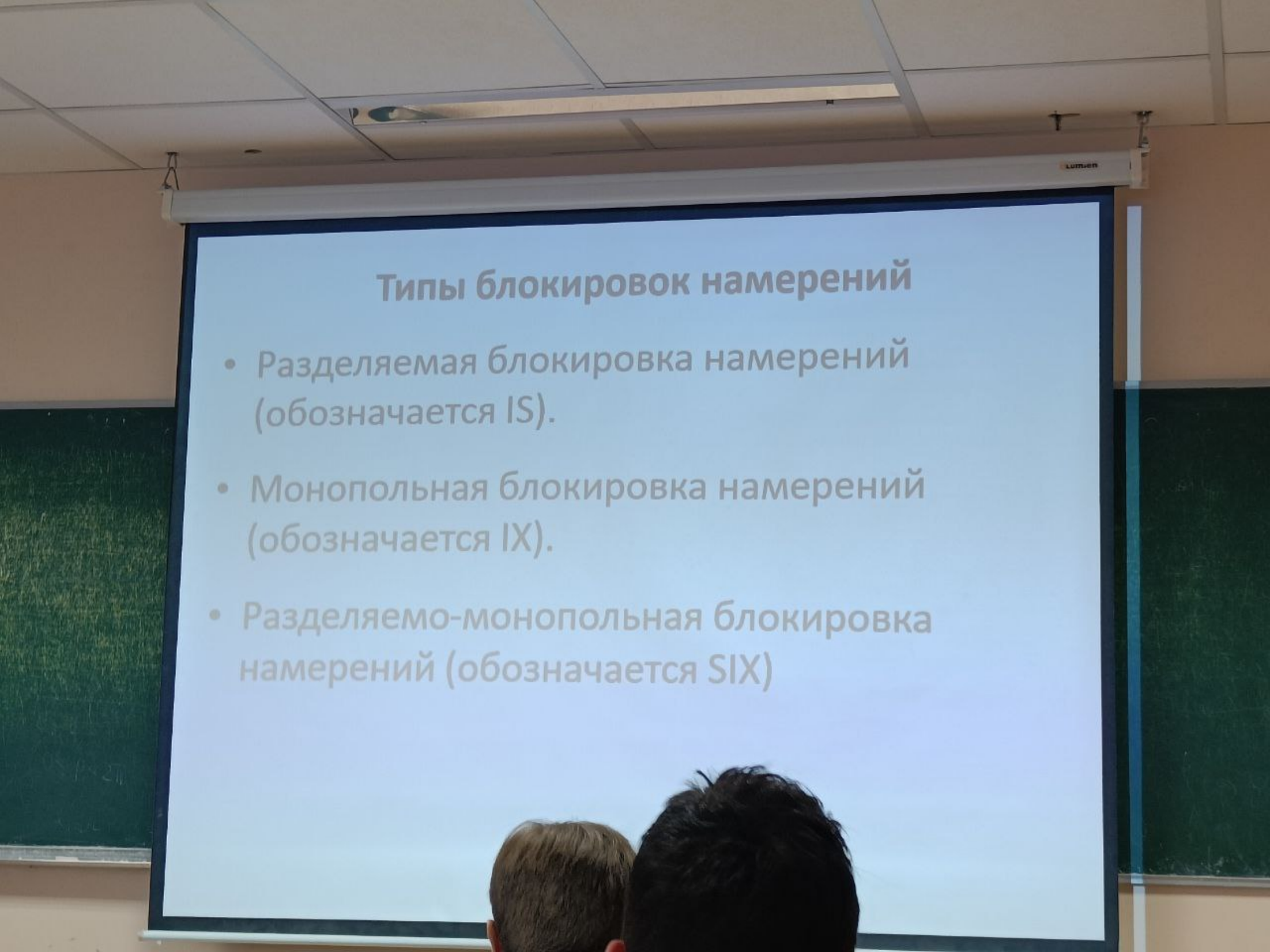


Блокировки намерений (intent locks)

Блокировки намерений всегда устанавливаются на таблицу или страницу, но никогда – на строку.

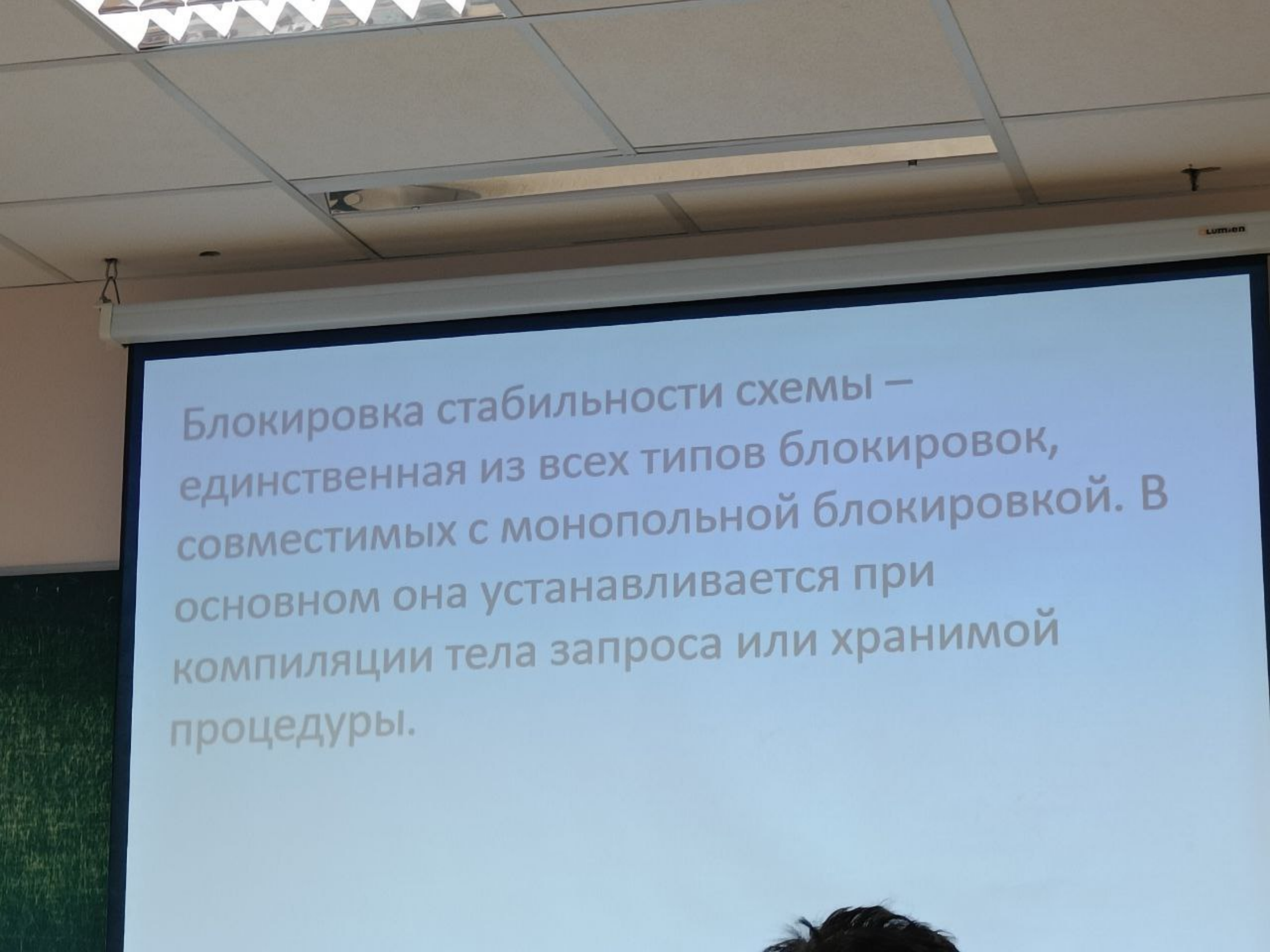

Блокировки намерений относятся к специальным типам блокировок и предназначены для повышения производительности работы менеджера блокировок.


Предположим, некая транзакция пытается изменить какую-либо строку в таблице test. Чтобы определить, что эту транзакцию необходимо заблокировать, менеджеру транзакций (в отсутствие блокировок намерения) пришлось бы сканировать всю таблицу syslockinfo для проверки всех строк таблицы test.


Типы блокировок намерений
• Разделяемая блокировка намерений (обозначается IS).
• Монопольная блокировка намерений (обозначается IX).
• Разделяемо-монопольная блокировка намерений (обозначается SIX)

Блокировки схемы данных

- Блокировка стабильности схемы (Schema Stability Lock), обозначается Sch-S. Данный тип блокировки предназначен для гарантии неизменности метаданных, но не самих данных.


Блокировка стабильности схемы –
единственная из всех типов блокировок,
совместимых с монопольной блокировкой. В
основном она устанавливается при
компиляции тела запроса или хранимой
процедуры.

На это время запрещается вносить изменения в схему данных, однако разрешается устанавливать любой тип блокировок на сами данные, с которыми будет работать компилируемый запрос.

- Блокировка изменения схемы (Schema Modification Lock), обозначается Sch-M. Данный тип блокировки не совместим ни с каким другим типом, ни с самим собой. Устанавливается после внесения изменений в схему данных и снимается после завершения транзакции.

- Блокировки могут преобразовываться друг в друга по следующей схеме

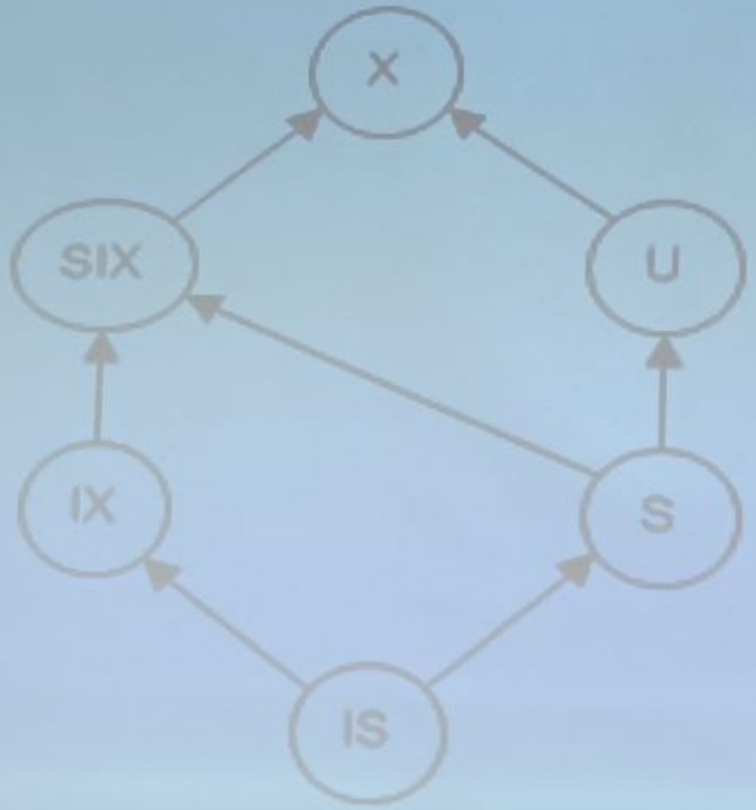
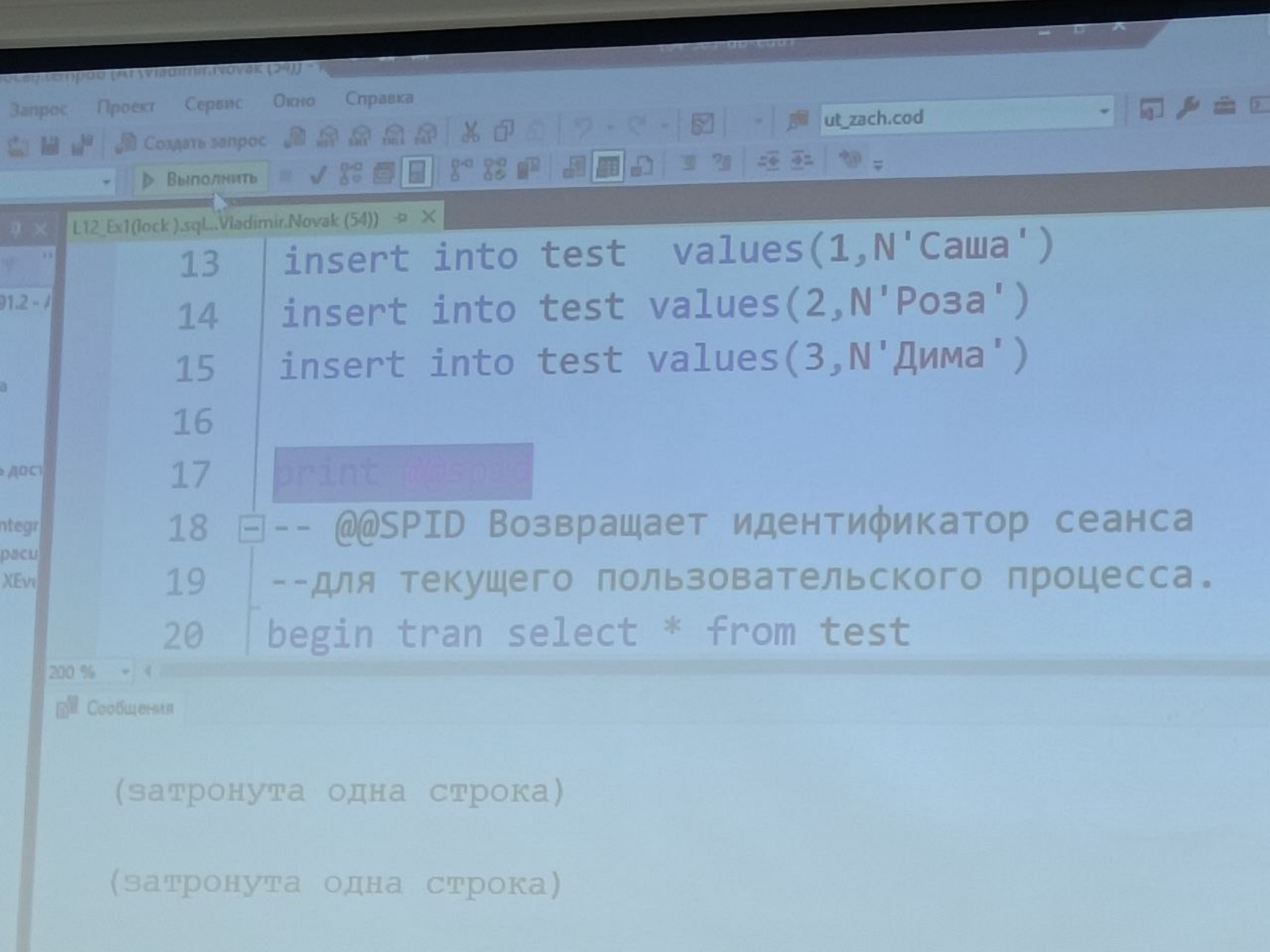
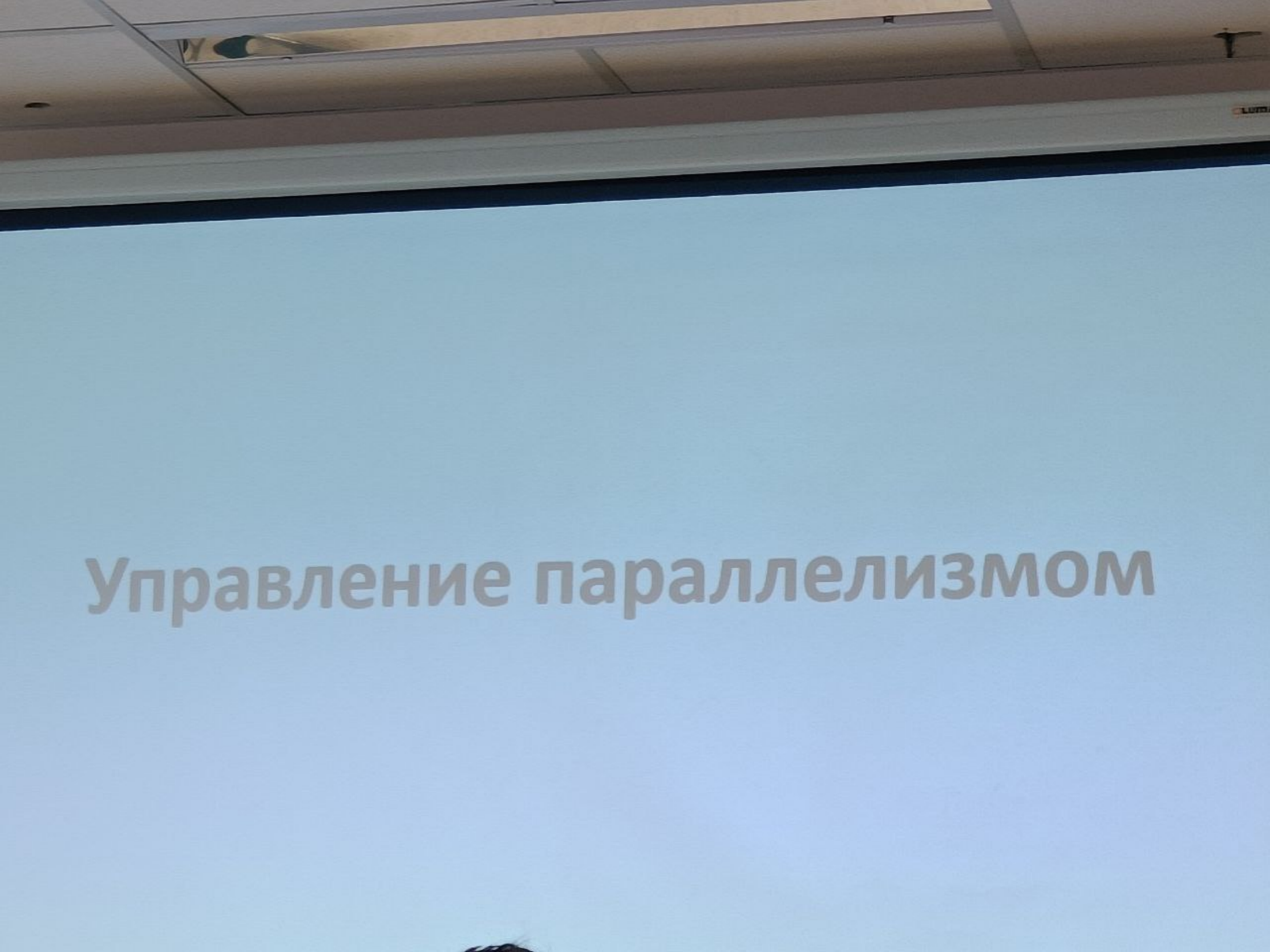
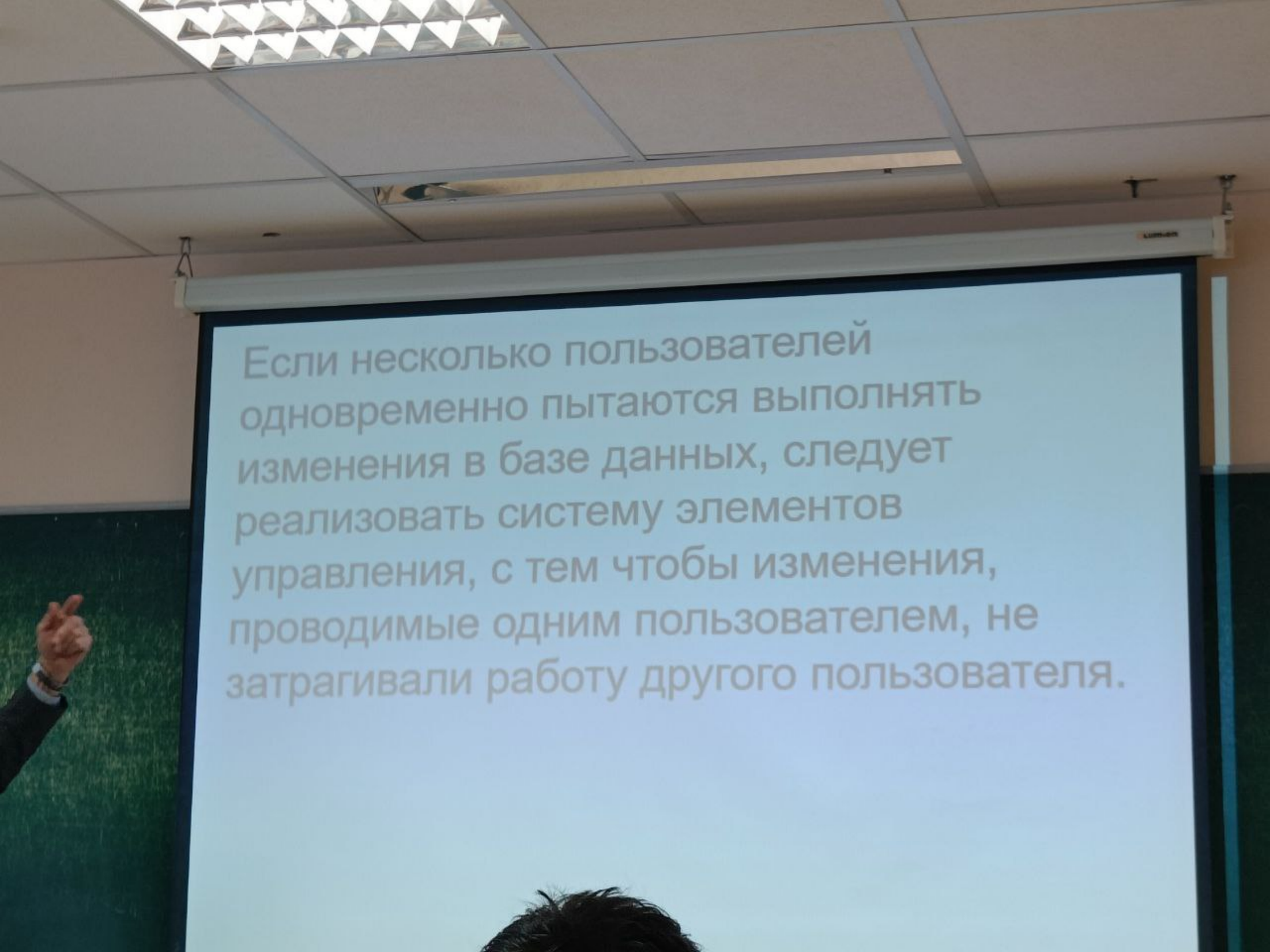


Table Hints (Табличные подсказки)

Поскольку оптимизатор запросов SQL Server обычно выбирает лучший план выполнения запроса, использовать подсказки рекомендуется только опытным разработчикам и администраторам баз данных в самом крайнем случае.


Запрос Проект Сервис Окно Справка
Создать запрос
ut_zach.cod
Выполнить
L12_Ex1(lock).sql...Vladimir.Novak (54))
13 insert into test values(1,N'Саша')
14 insert into test values(2,N'Роза')
15 insert into test values(3,N'Дима')
16
17 print @@SPID
18 -- @@SPID Возвращает идентификатор сеанса
19 --для текущего пользовательского процесса.
20 begin tran select * from test
200 %
Сообщения
(затронута одна строка)
(затронута одна строка)


Управление параллелизмом


Если несколько пользователей
одновременно пытаются выполнять
изменения в базе данных, следует
реализовать систему элементов
управления, с тем чтобы изменения,
проводимые одним пользователем, не
затрагивали работу другого пользователя.

Такая система называется управлением параллелизмом.
Теория управления параллелизмом предлагает два способа осуществления управления параллелизмом.

Пессимистическое управление параллелизмом

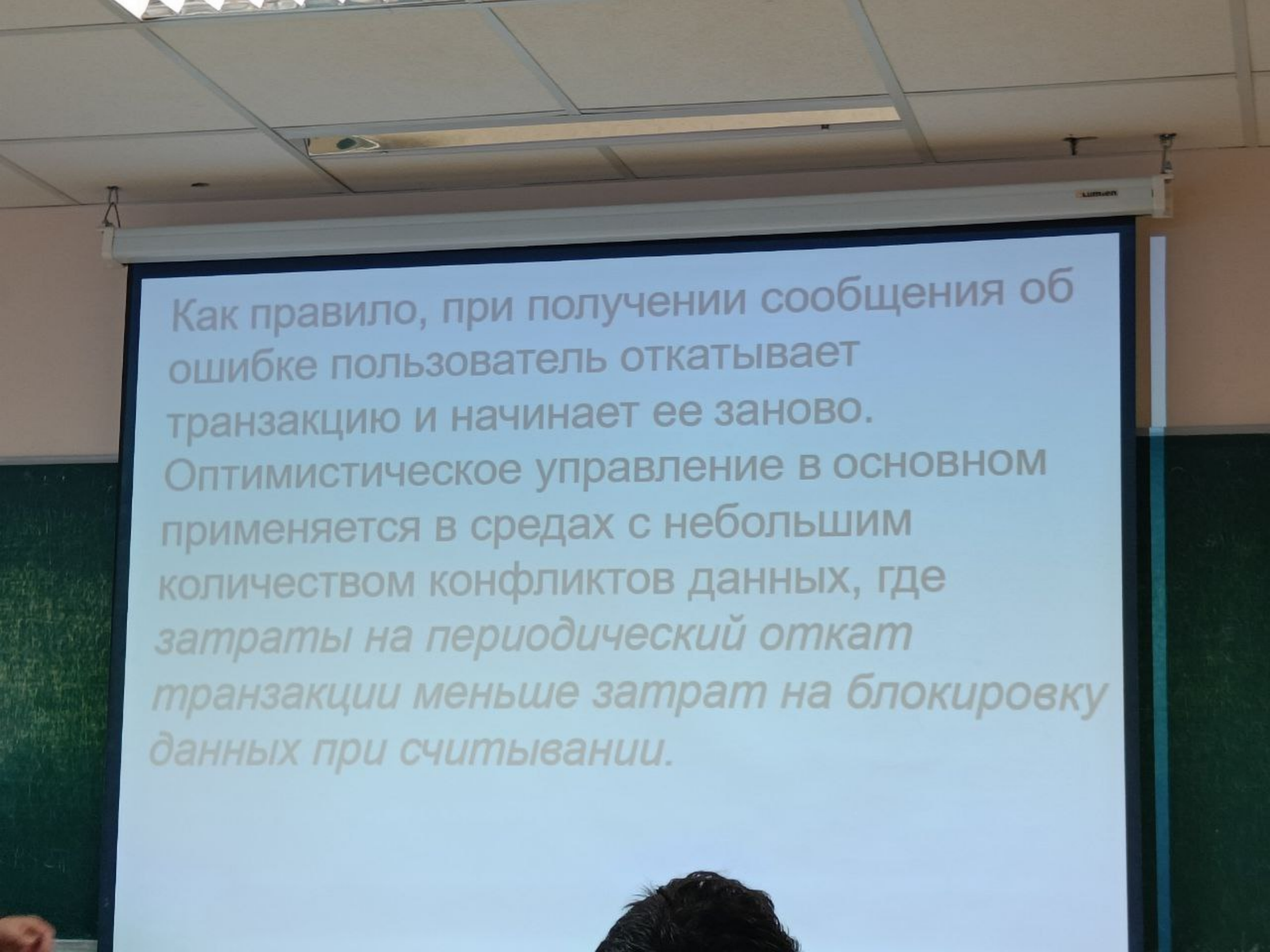
Система блокировок не допускает, чтобы изменение данных одними пользователями влияло на других пользователей. После того как действие пользователя приводит к блокировке, до тех пор пока инициатор ее не снимет, другие пользователи не могут выполнять действия, которые могут вызвать конфликт с блокировкой.

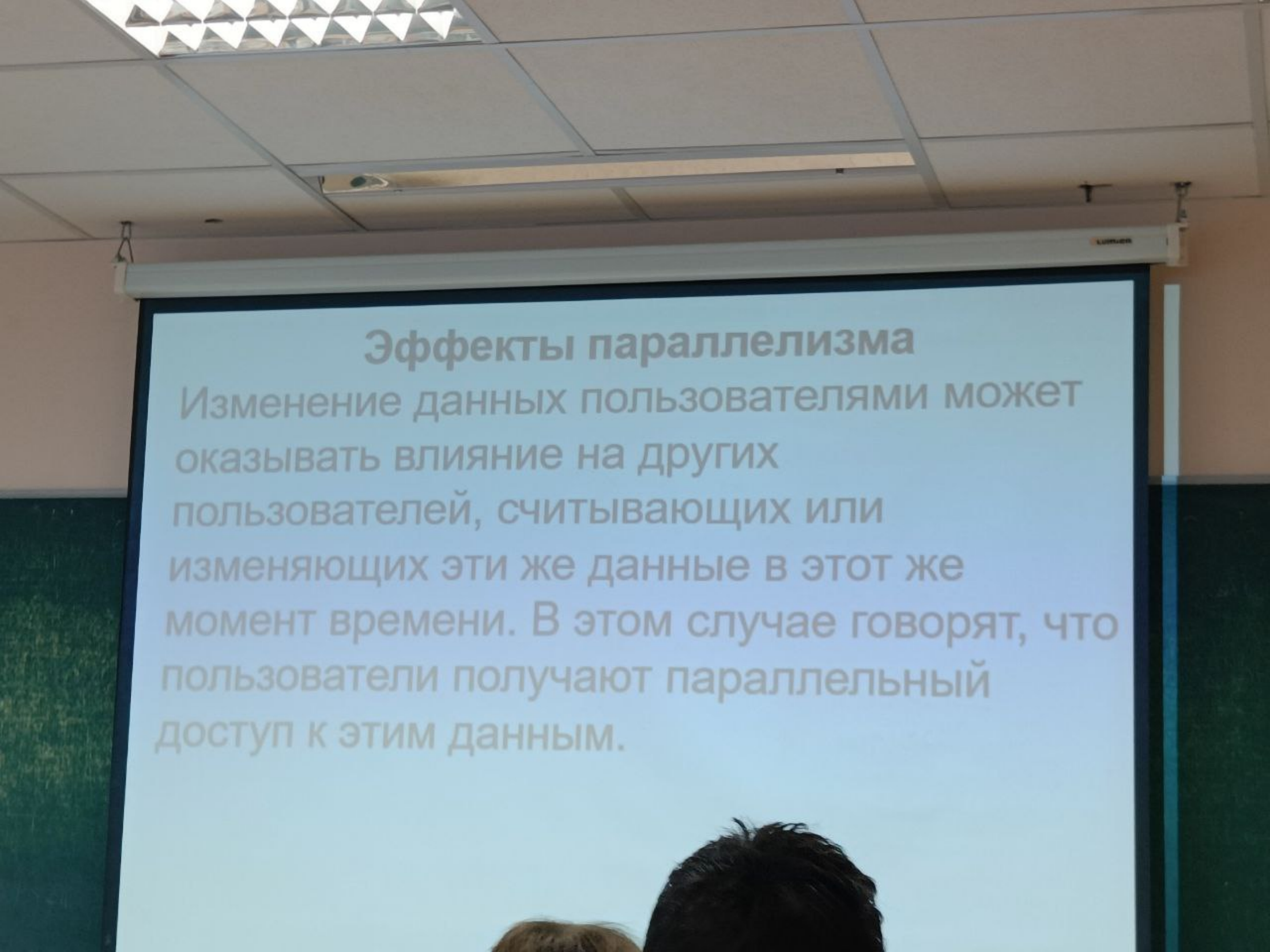
Это управление в основном применяется в средах с большим количеством конфликтов данных, где *затраты на защиту данных с помощью блокировок меньше затрат на откат транзакций в случае конфликтов параллелизма.*

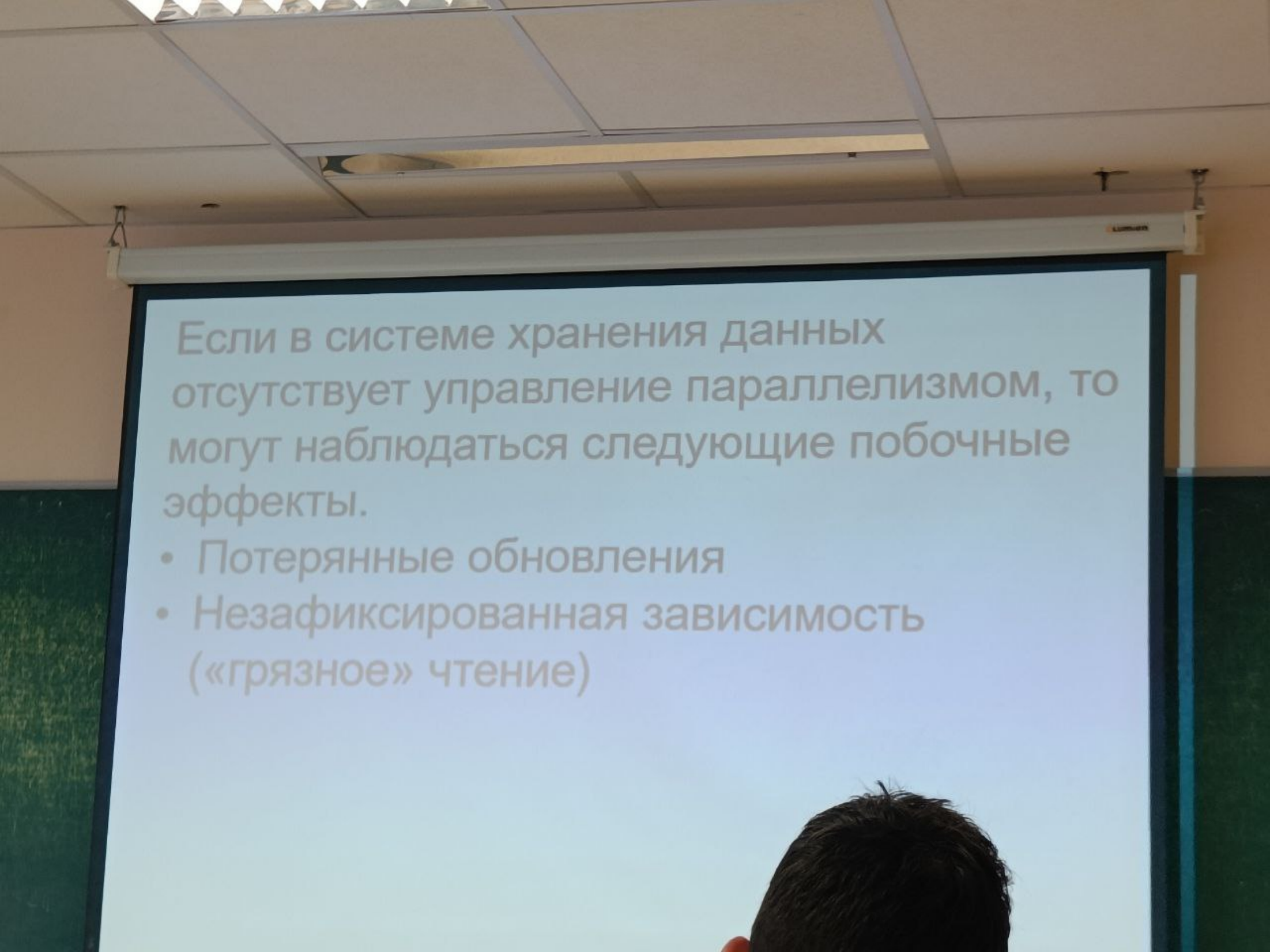
Оптимистическое управление параллелизмом

Пользователи не блокируют данные на период чтения. Когда пользователь обновляет данные, система проверяет, вносил ли другой пользователь в них изменение после считывания. Если другой пользователь изменял данные, возникает ошибка.

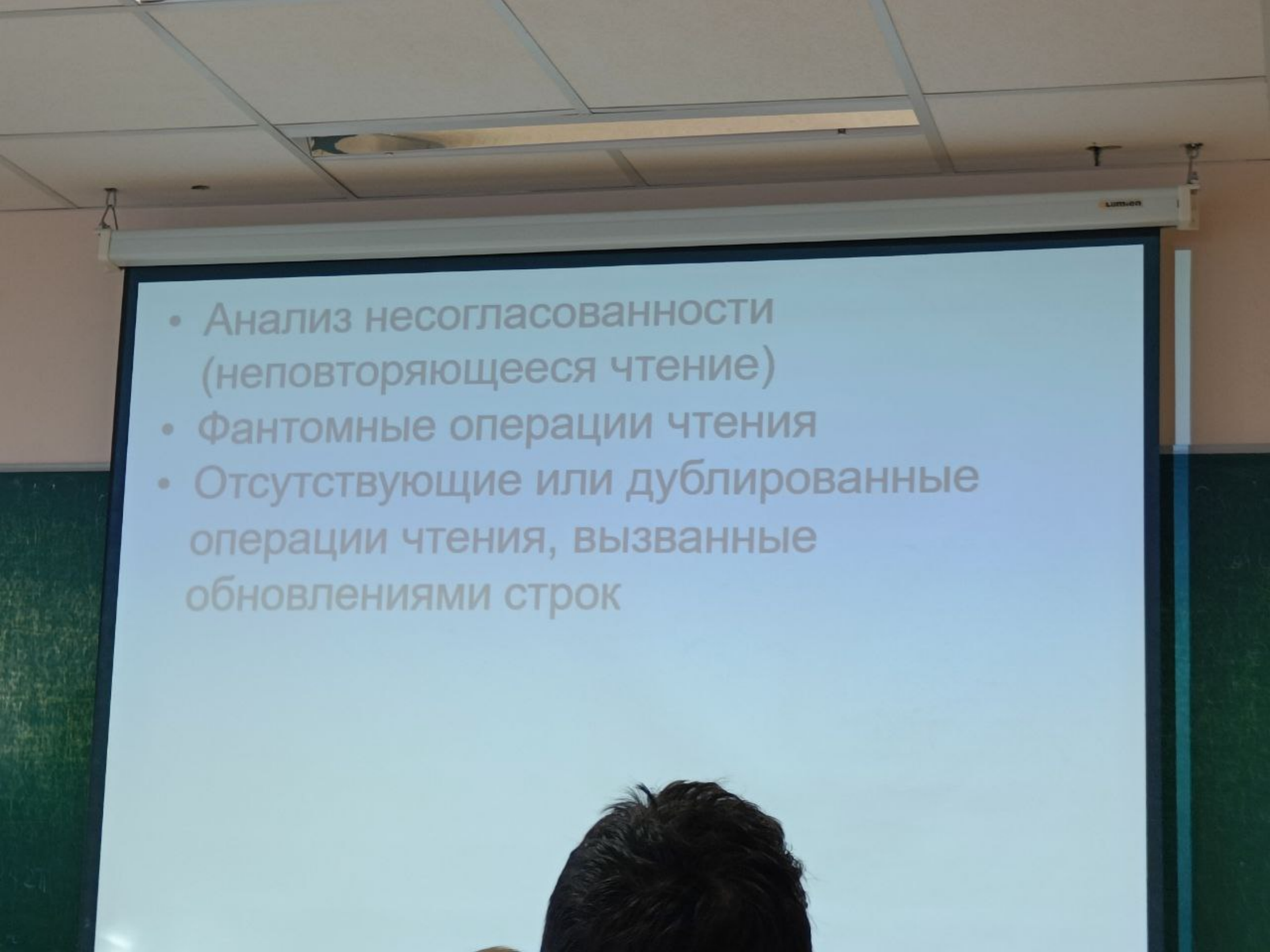
Как правило, при получении сообщения об ошибке пользователь откатывает транзакцию и начинает ее заново.
Оптимистическое управление в основном применяется в средах с небольшим количеством конфликтов данных, где *затраты на периодический откат транзакции меньше затрат на блокировку данных при считывании.*

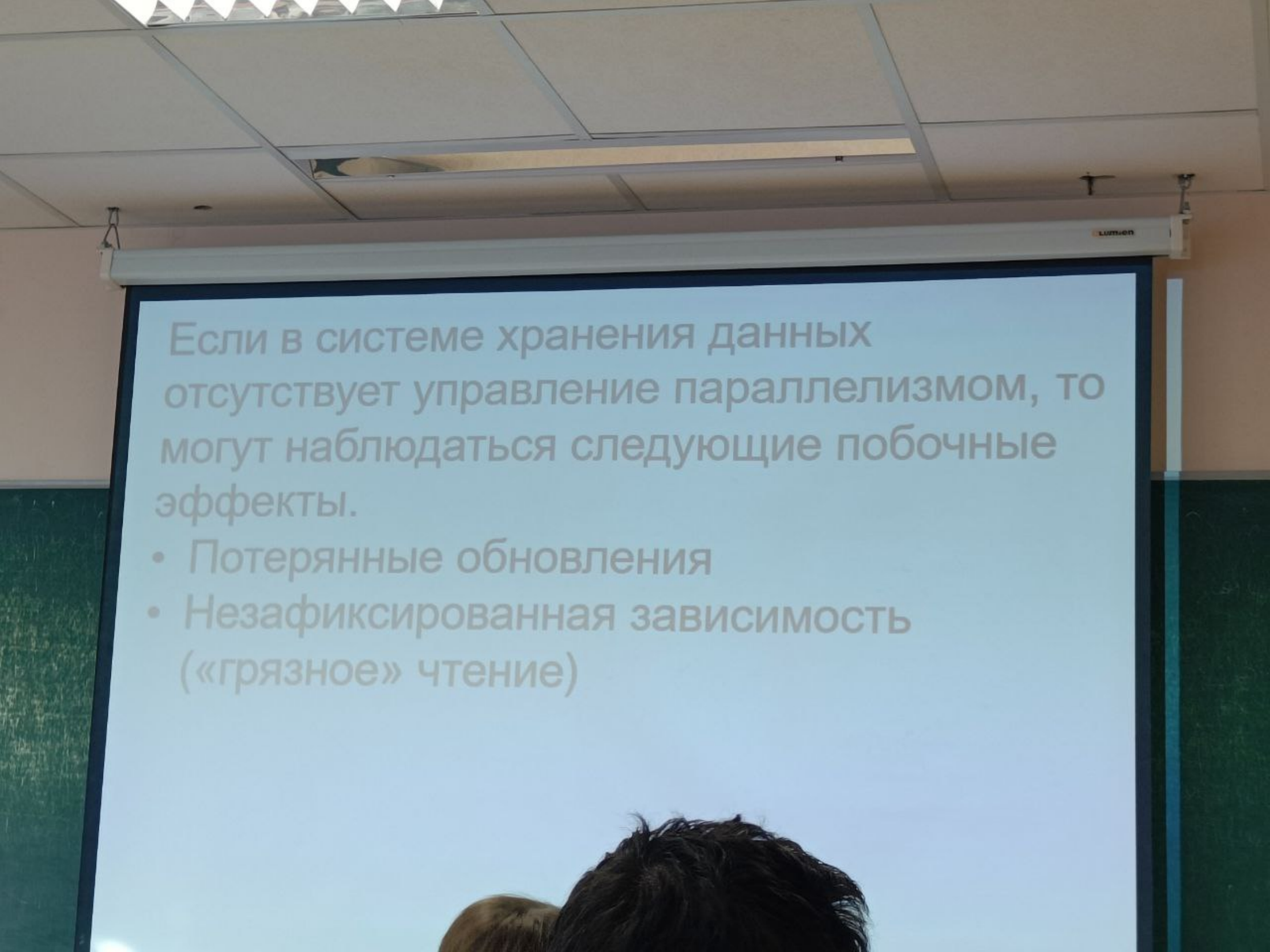



Эффекты параллелизма
Изменение данных пользователями может оказывать влияние на других пользователей, считывающих или изменяющих эти же данные в этот же момент времени. В этом случае говорят, что пользователи получают параллельный доступ к этим данным.


Если в системе хранения данных отсутствует управление параллелизмом, то могут наблюдаться следующие побочные эффекты.
• Потерянные обновления
• Незафиксированная зависимость («грязное» чтение)

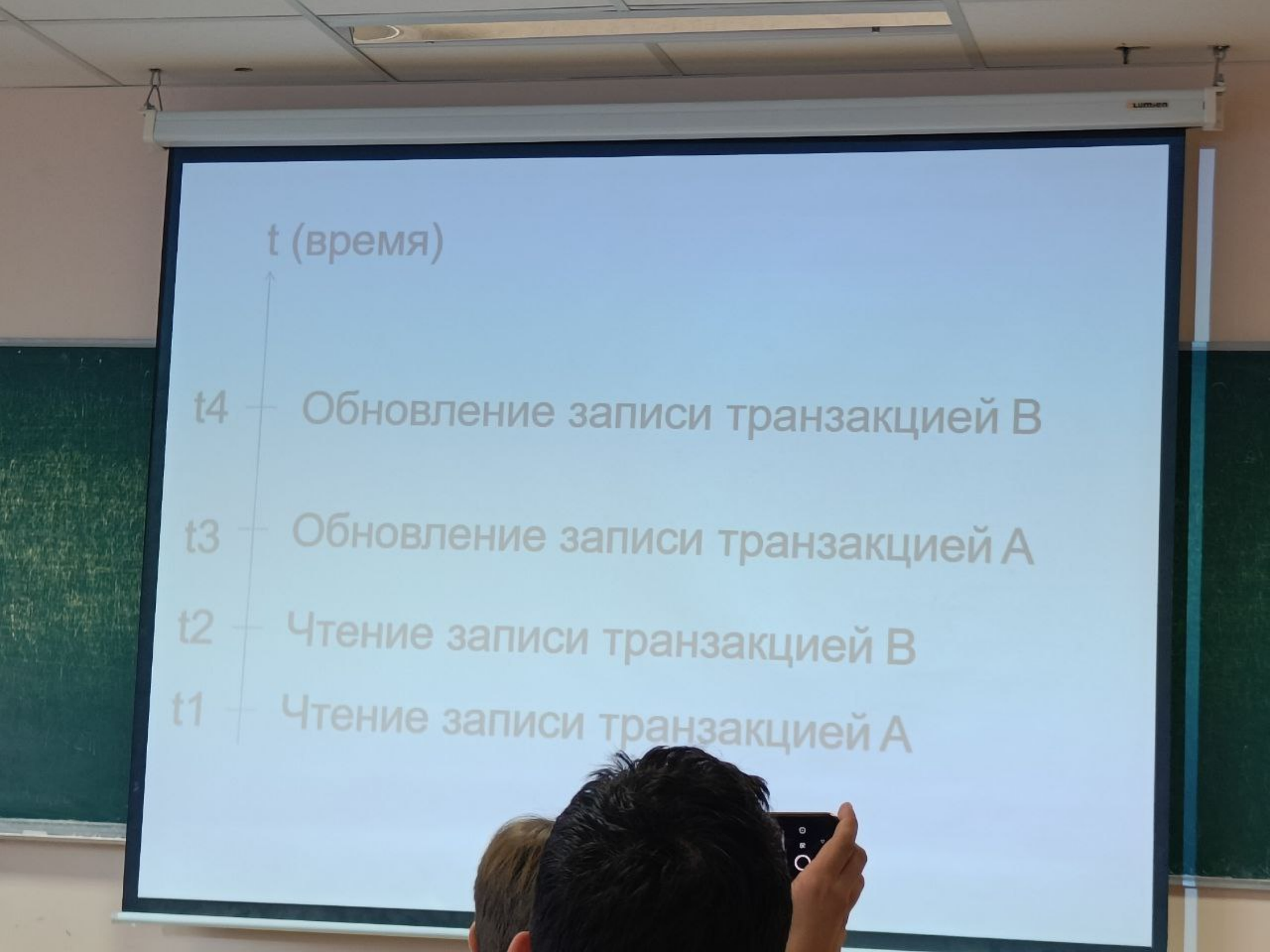
- Анализ несогласованности (неповторяющееся чтение)
- Фантомные операции чтения
- Отсутствующие или дублированные операции чтения, вызванные обновлениями строк




Если в системе хранения данных отсутствует управление параллелизмом, то могут наблюдаться следующие побочные эффекты.
• Потерянные обновления
• Незафиксированная зависимость («грязное» чтение)

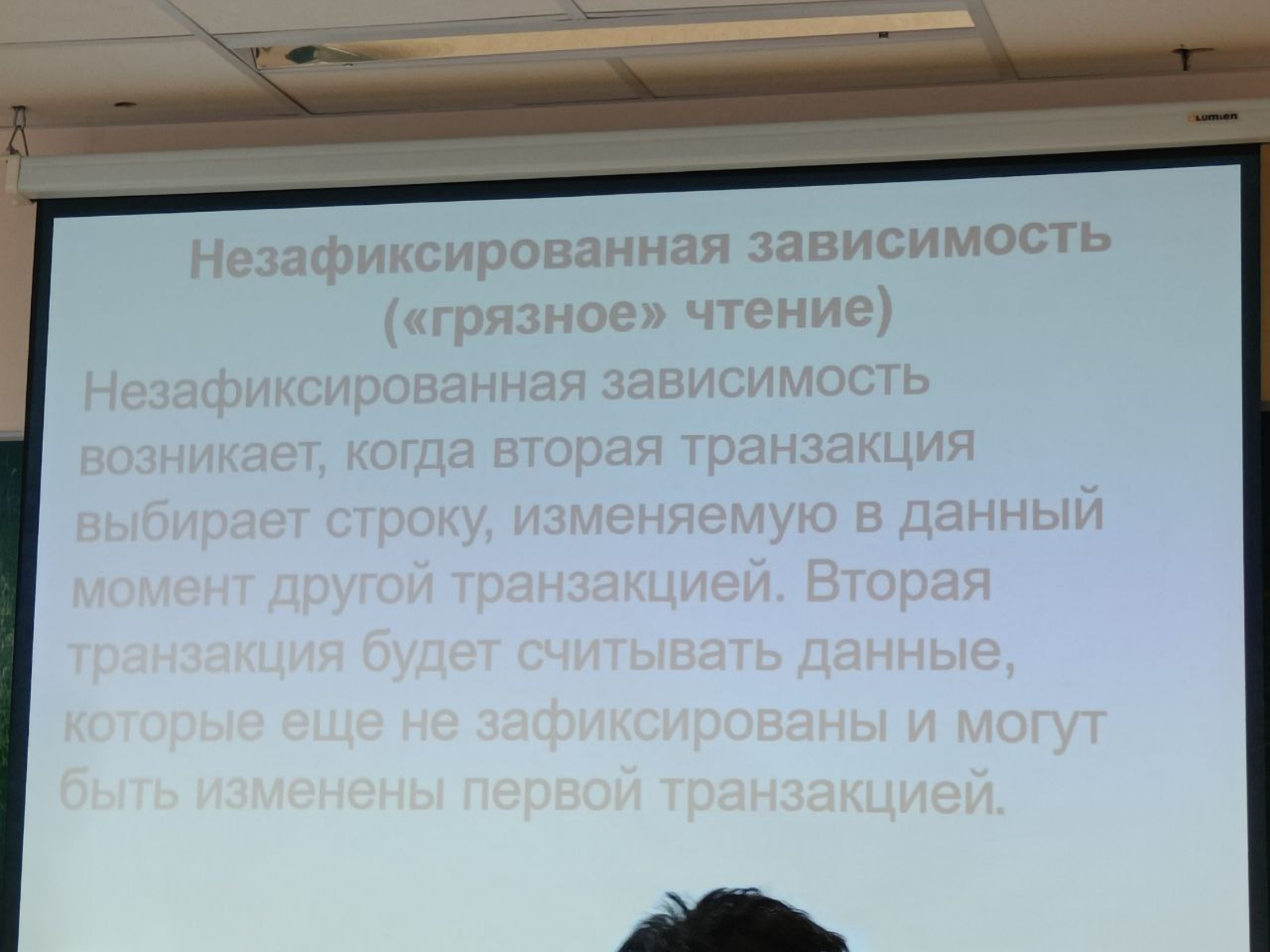
Потерянные обновления

Потерянные обновления возникают, когда две или более транзакций выбирают одну и ту же строку и изменяют ее на основании ее исходного значения. Каждая транзакция не знает о других транзакциях. Последнее обновление изменяет данные других транзакций, что приводит к потере данных.


t (время)
t4
Обновление записи транзакцией B
t3
Обновление записи транзакцией A
t2
Чтение записи транзакцией B
t1
Чтение записи транзакцией A

Незафиксированная зависимость («грязное» чтение)

Незафиксированная зависимость возникает, когда вторая транзакция выбирает строку, изменяемую в данный момент другой транзакцией. Вторая транзакция будет считывать данные, которые еще не зафиксированы и могут быть изменены первой транзакцией.

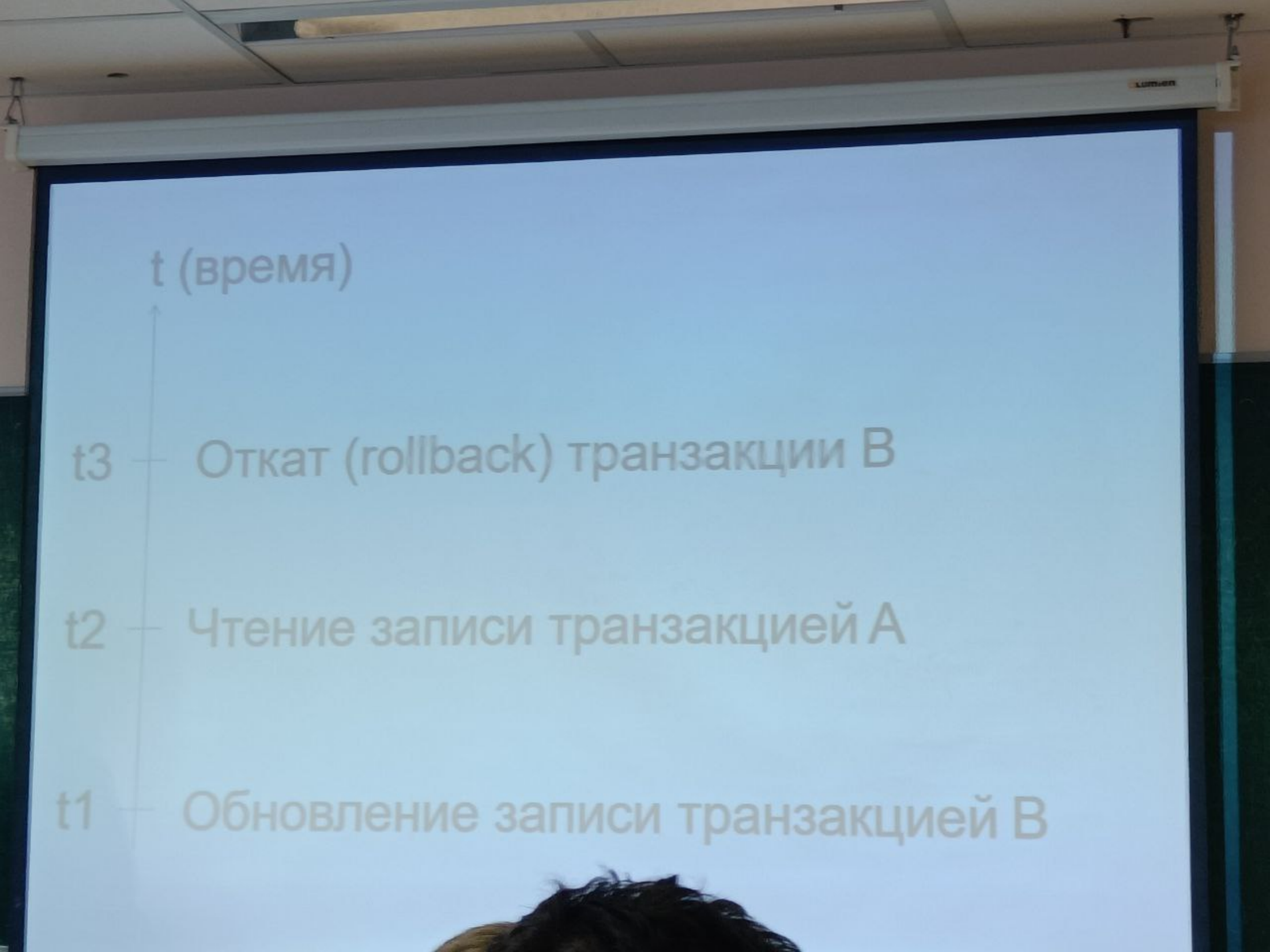


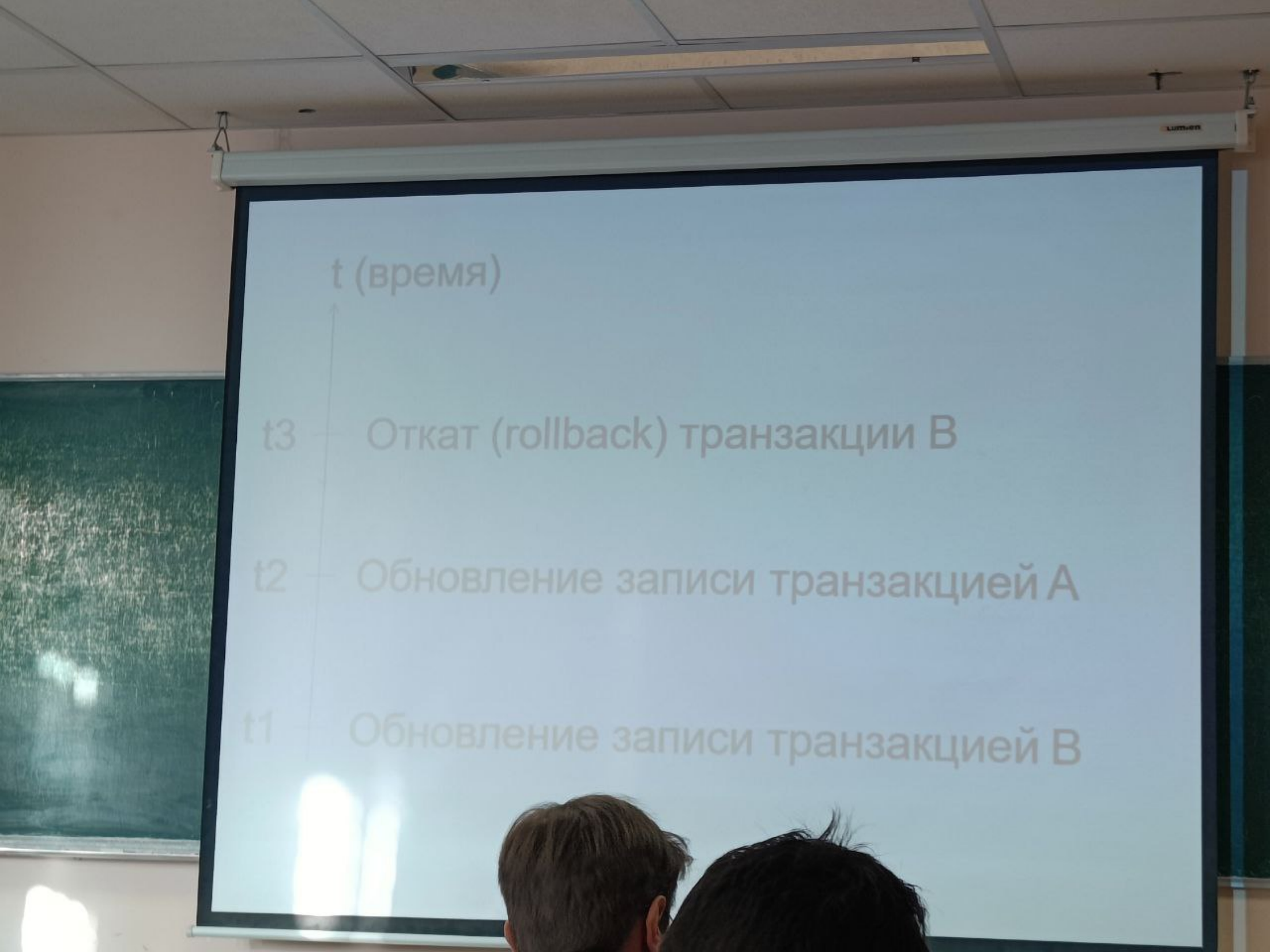
t (время)

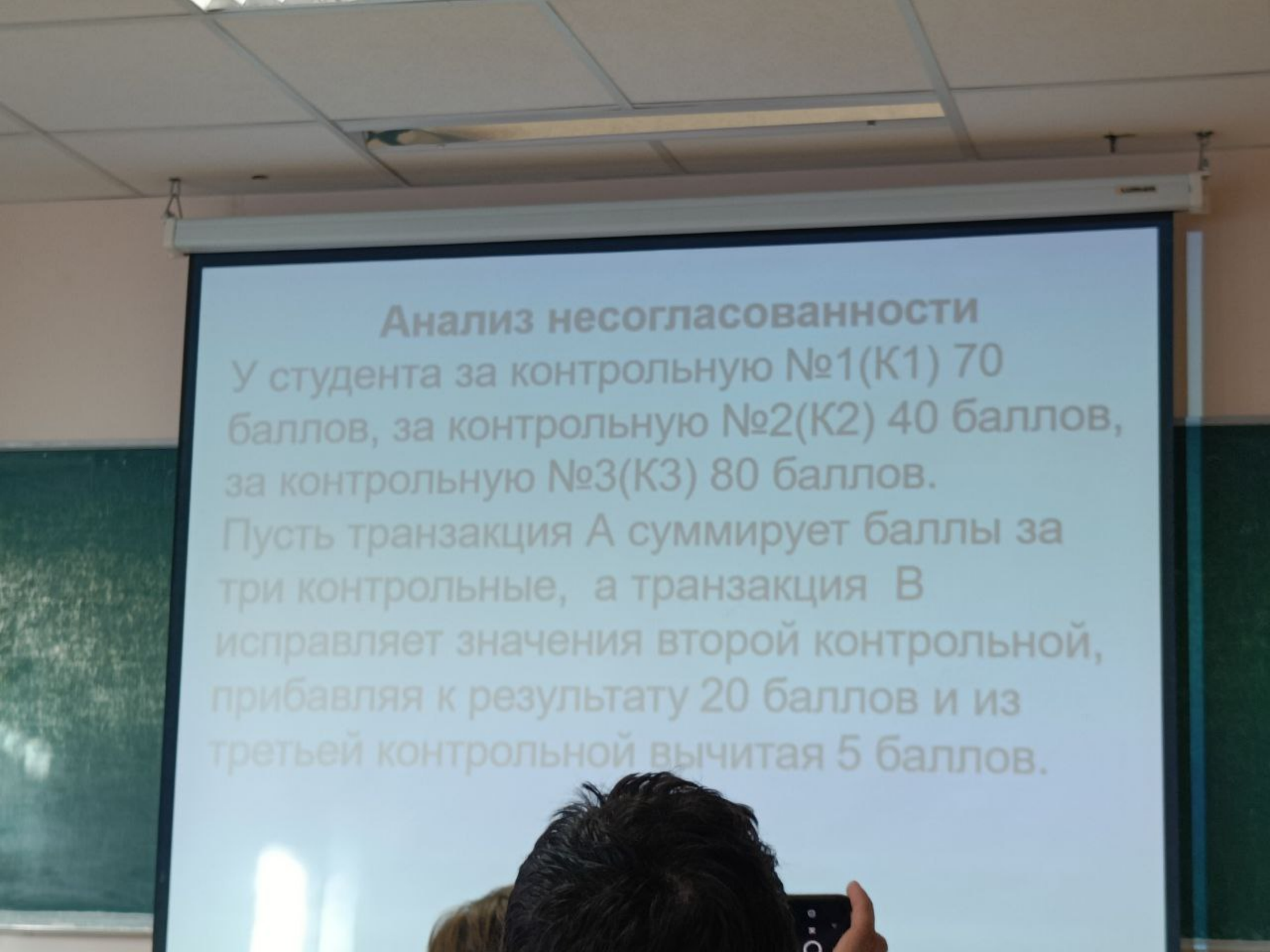
t3 — Откат (rollback) транзакции B

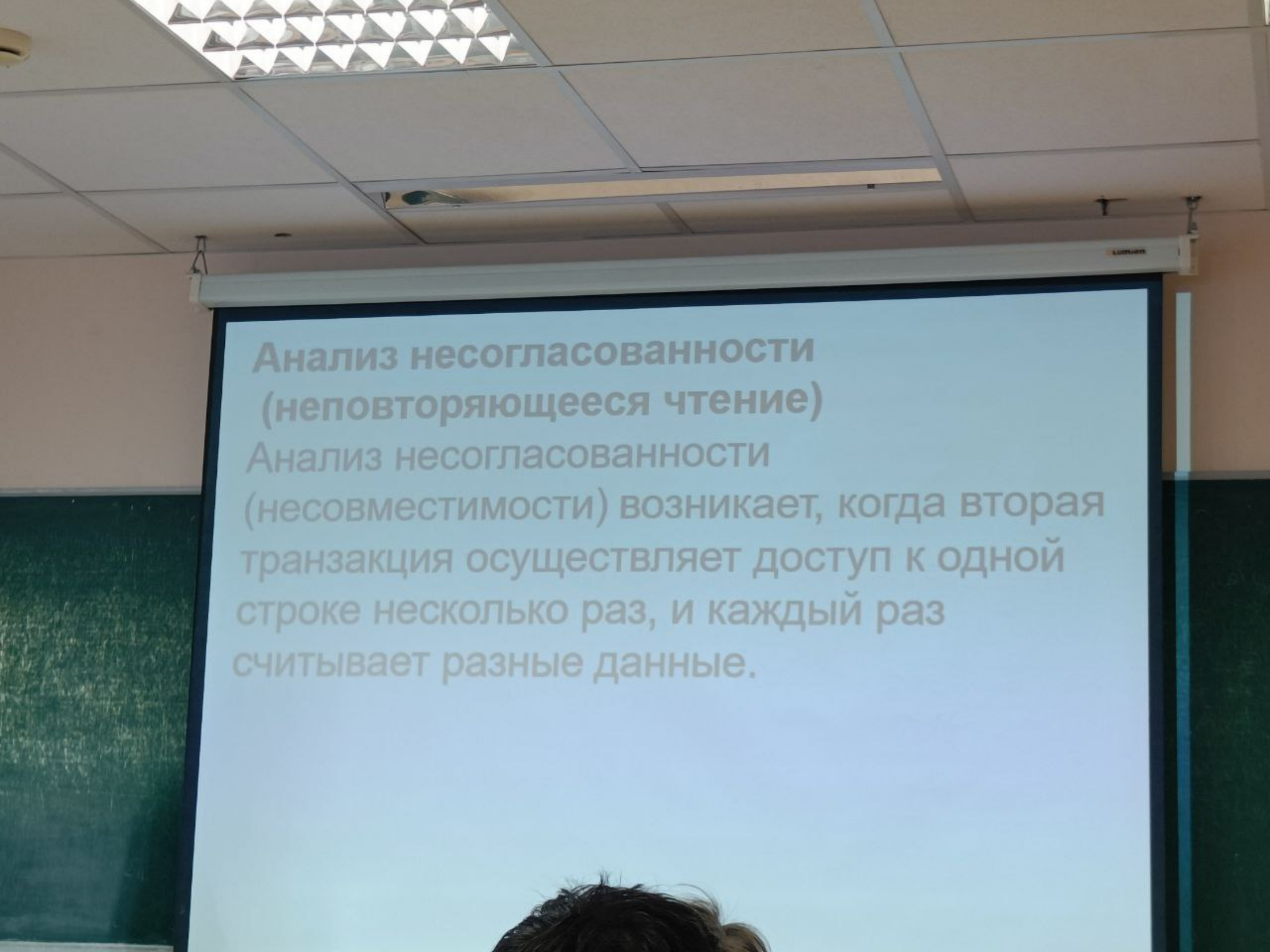
t2 — Чтение записи транзакцией A

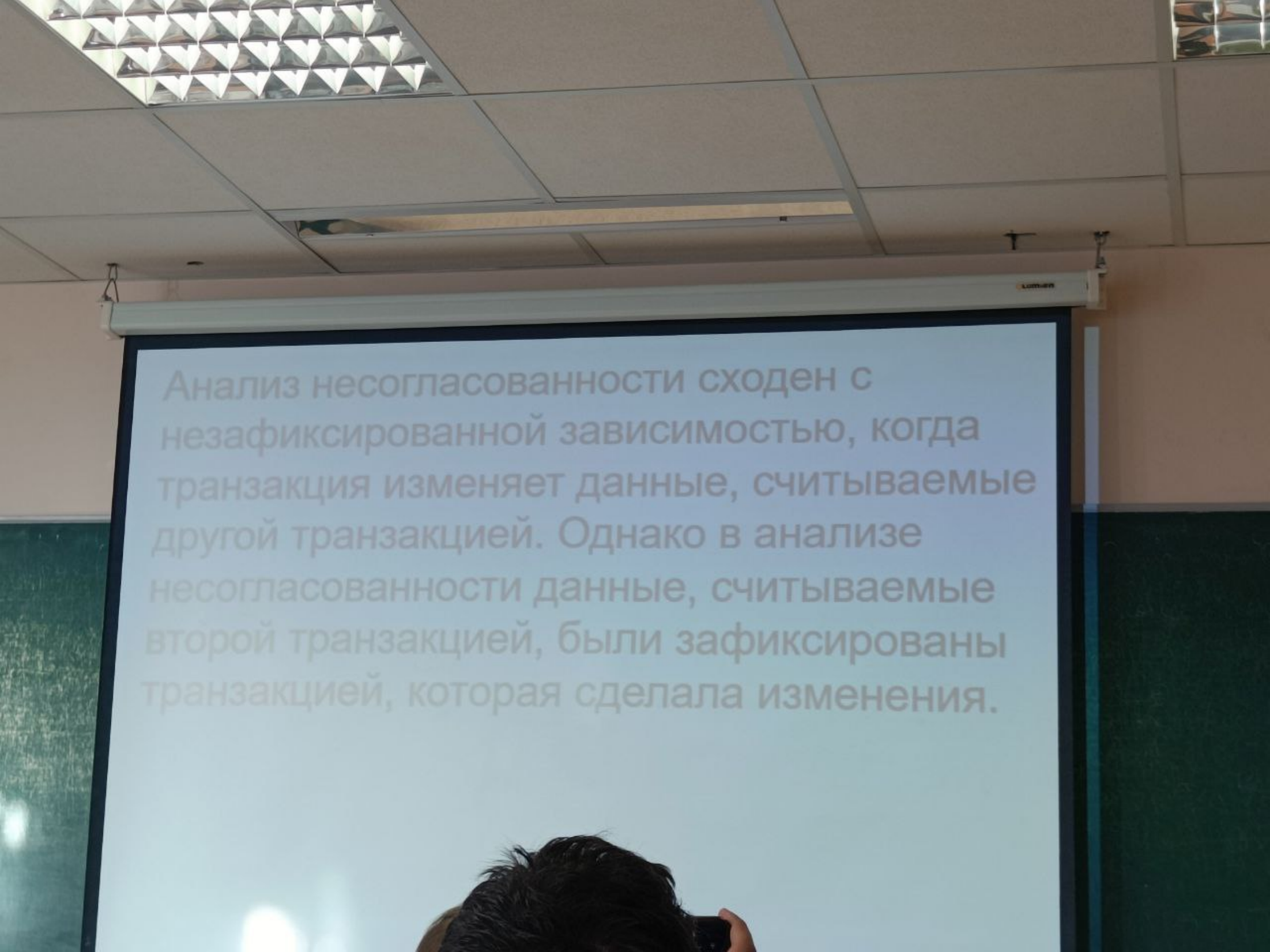
t1 — Обновление записи транзакцией B

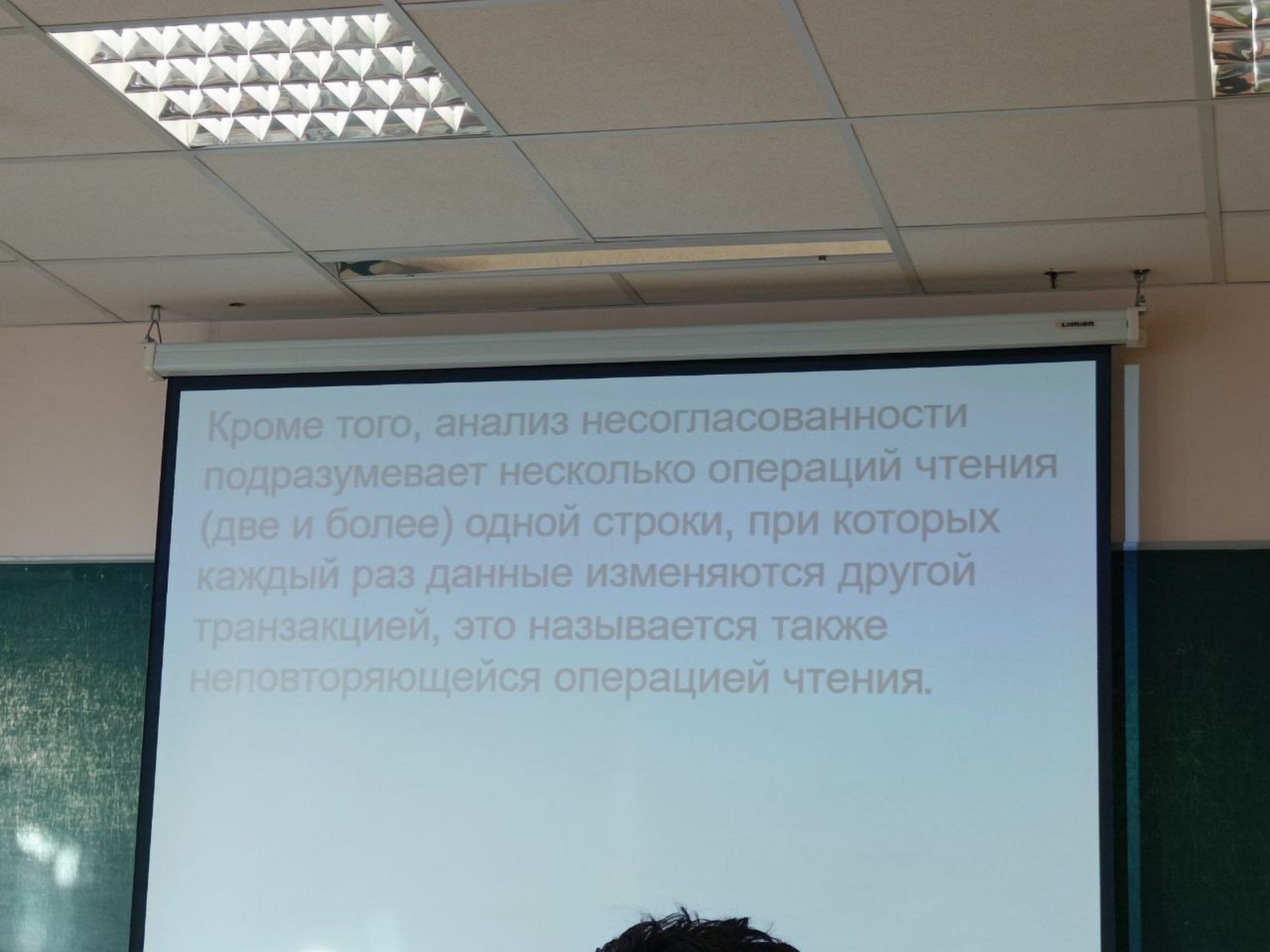


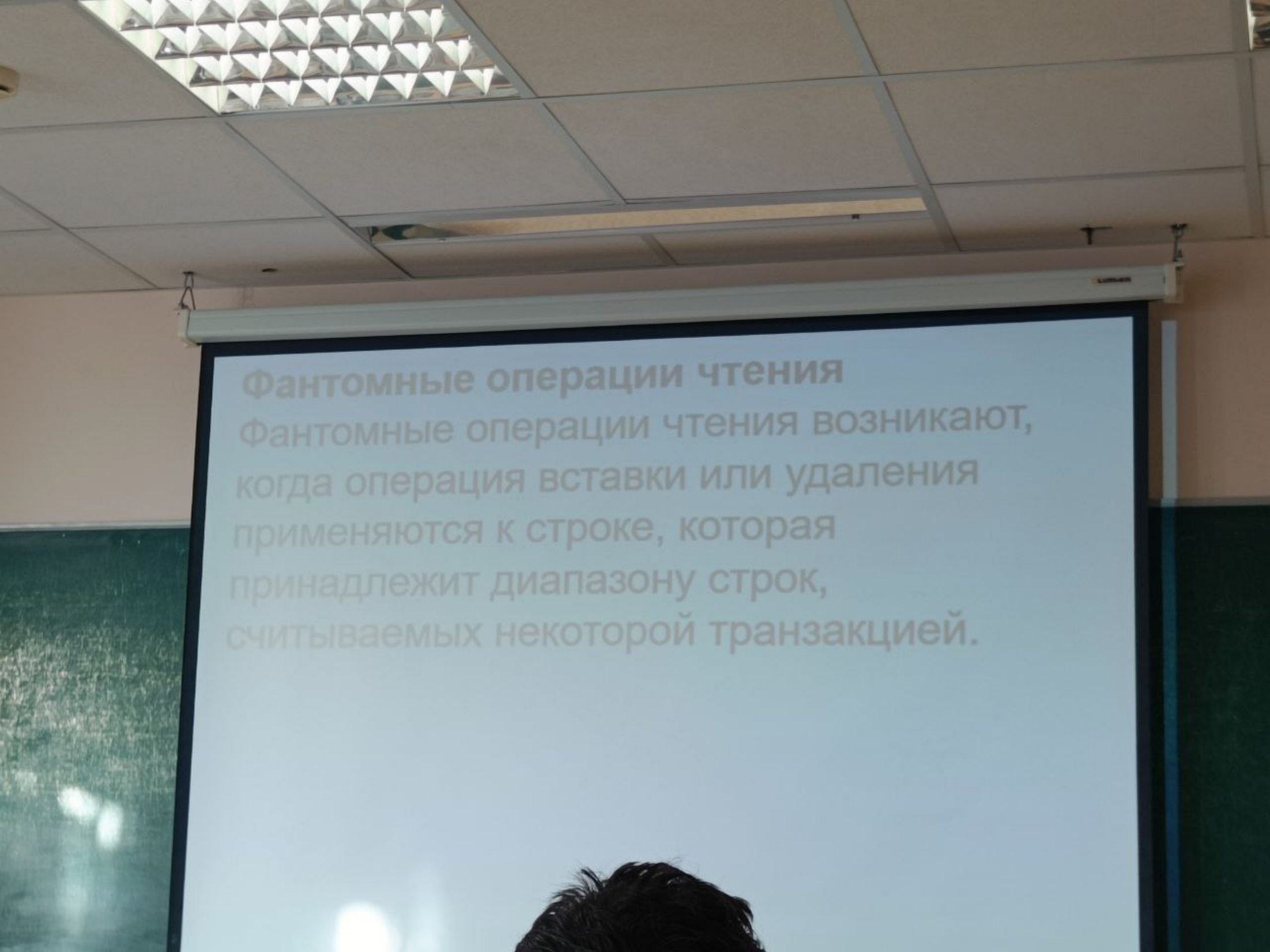

t (время)
t3
Откат (rollback) транзакции B
t2
Обновление записи транзакцией A
t1
Обновление записи транзакцией B


Анализ несогласованности
У студента за контрольную №1(К1) 70 баллов, за контрольную №2(К2) 40 баллов, за контрольную №3(К3) 80 баллов.
Пусть транзакция А суммирует баллы за три контрольные, а транзакция В исправляет значения второй контрольной, прибавляя к результату 20 баллов и из третьей контрольной вычитая 5 баллов.

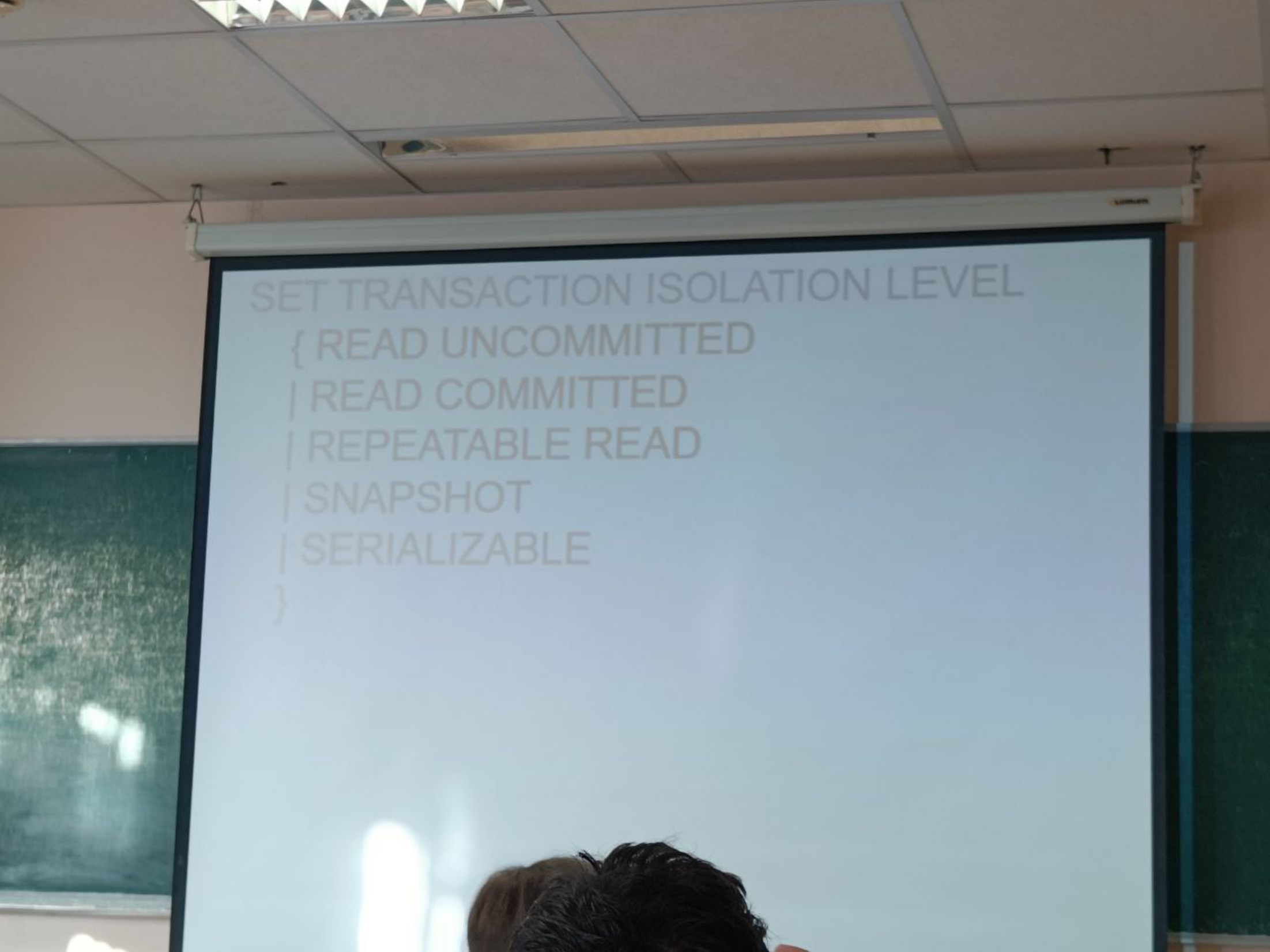

Анализ несогласованности (неповторяющееся чтение)
Анализ несогласованности (несовместимости) возникает, когда вторая транзакция осуществляет доступ к одной строке несколько раз, и каждый раз считывает разные данные.


Анализ несогласованности сходен с
незафиксированной зависимостью, когда
транзакция изменяет данные, считываемые
другой транзакцией. Однако в анализе
несогласованности данные, считываемые
второй транзакцией, были зафиксированы
транзакцией, которая сделала изменения.


Кроме того, анализ несогласованности
подразумевает несколько операций чтения
(две и более) одной строки, при которых
каждый раз данные изменяются другой
транзакцией, это называется также
неповторяющейся операцией чтения.

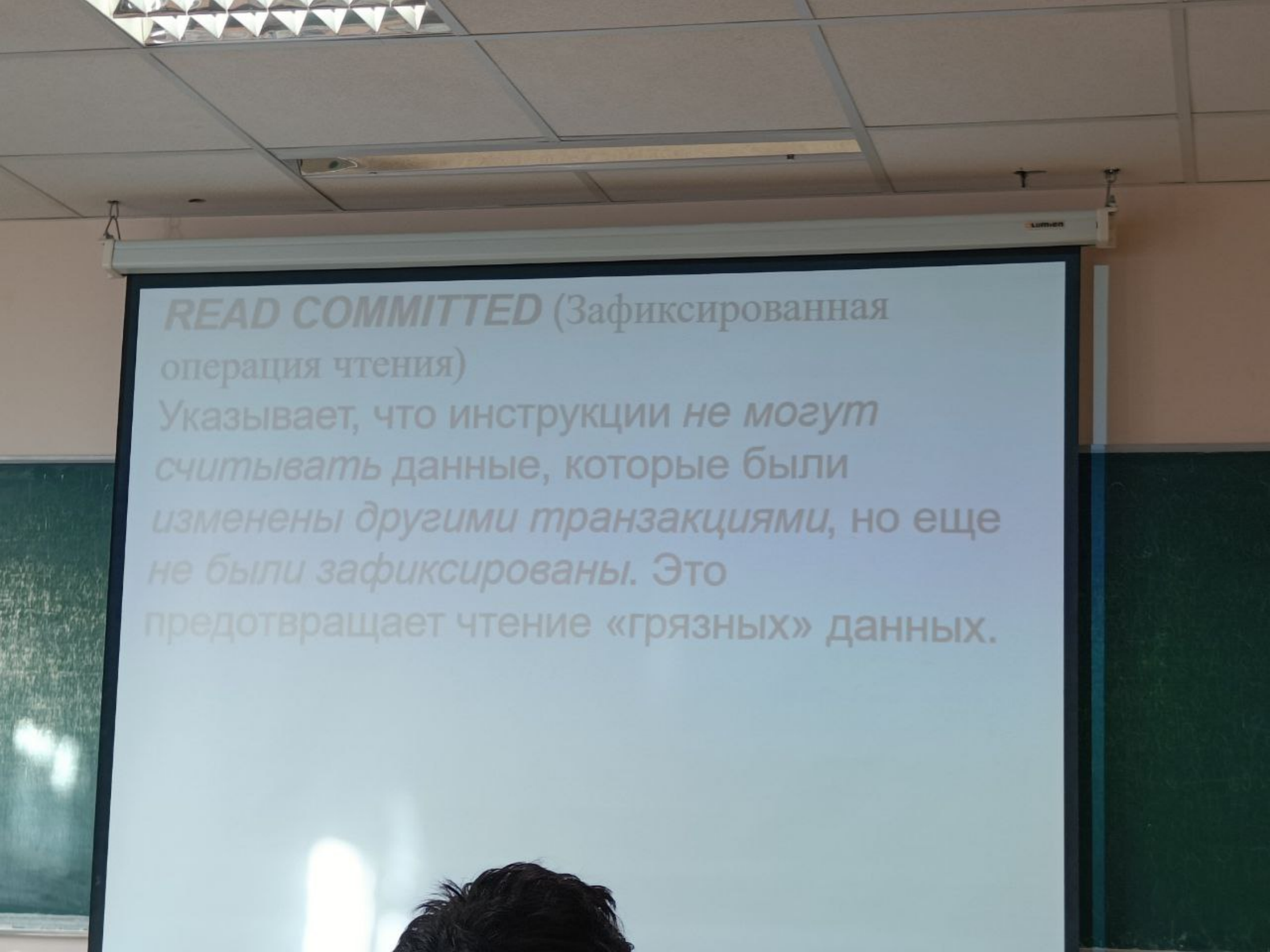

Фантомные операции чтения
Фантомные операции чтения возникают,
когда операция вставки или удаления
применяются к строке, которая
принадлежит диапазону строк,
считываемых некоторой транзакцией.

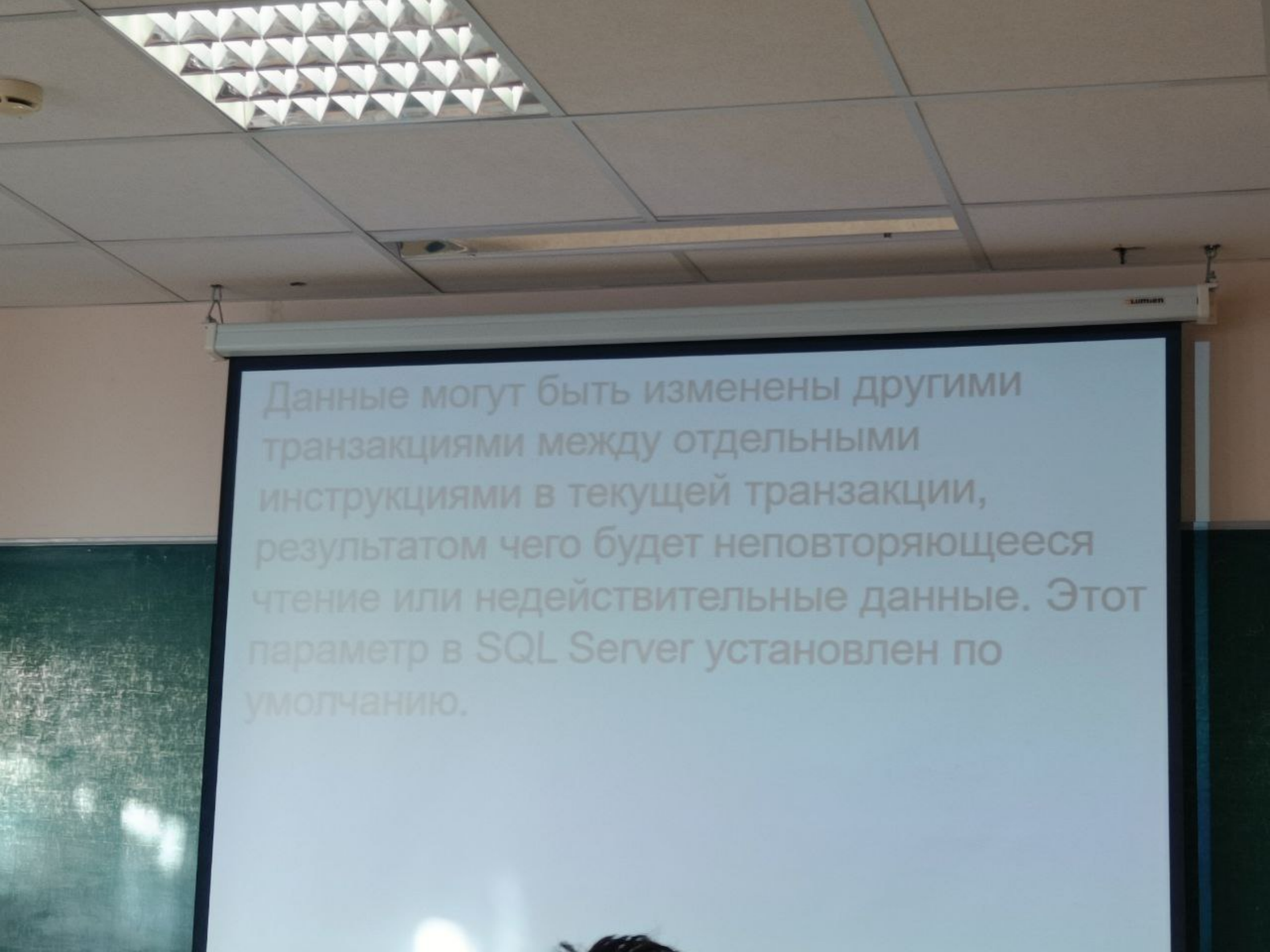
Первая транзакция считывает набор строк, содержащий строку, которая уже не будет существовать при второй и последующих операциях чтения в результате ее удаления другой транзакцией. Аналогично вторая и последующие транзакции считают строку, которая не существовала при первом чтении и была вставлена другой транзакцией.

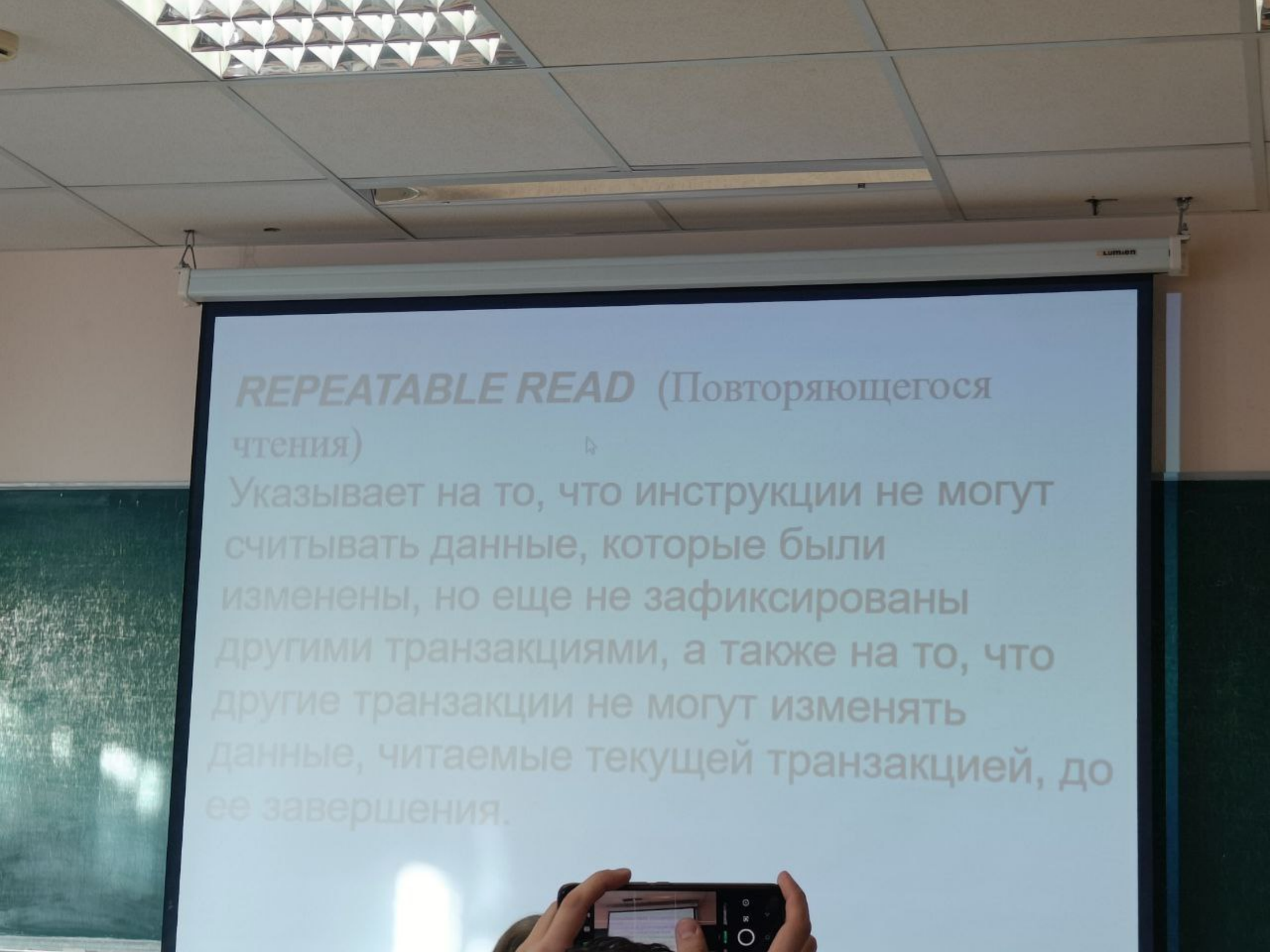

SET TRANSACTION ISOLATION LEVEL
{ READ UNCOMMITTED
| READ COMMITTED
| REPEATABLE READ
| SNAPSHOT
| SERIALIZABLE
}

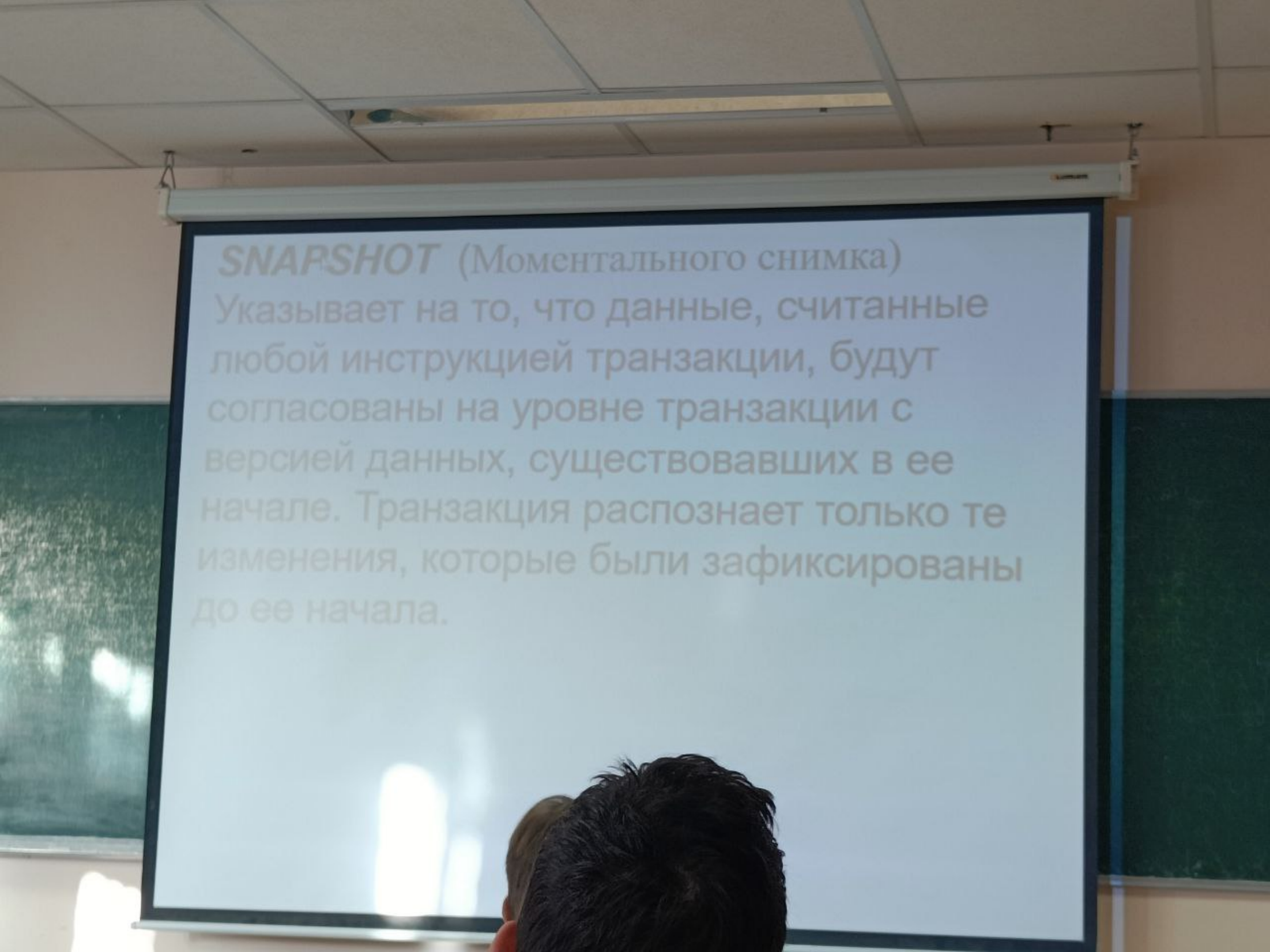
READ UNCOMMITTED (Незафиксированная операция чтения)

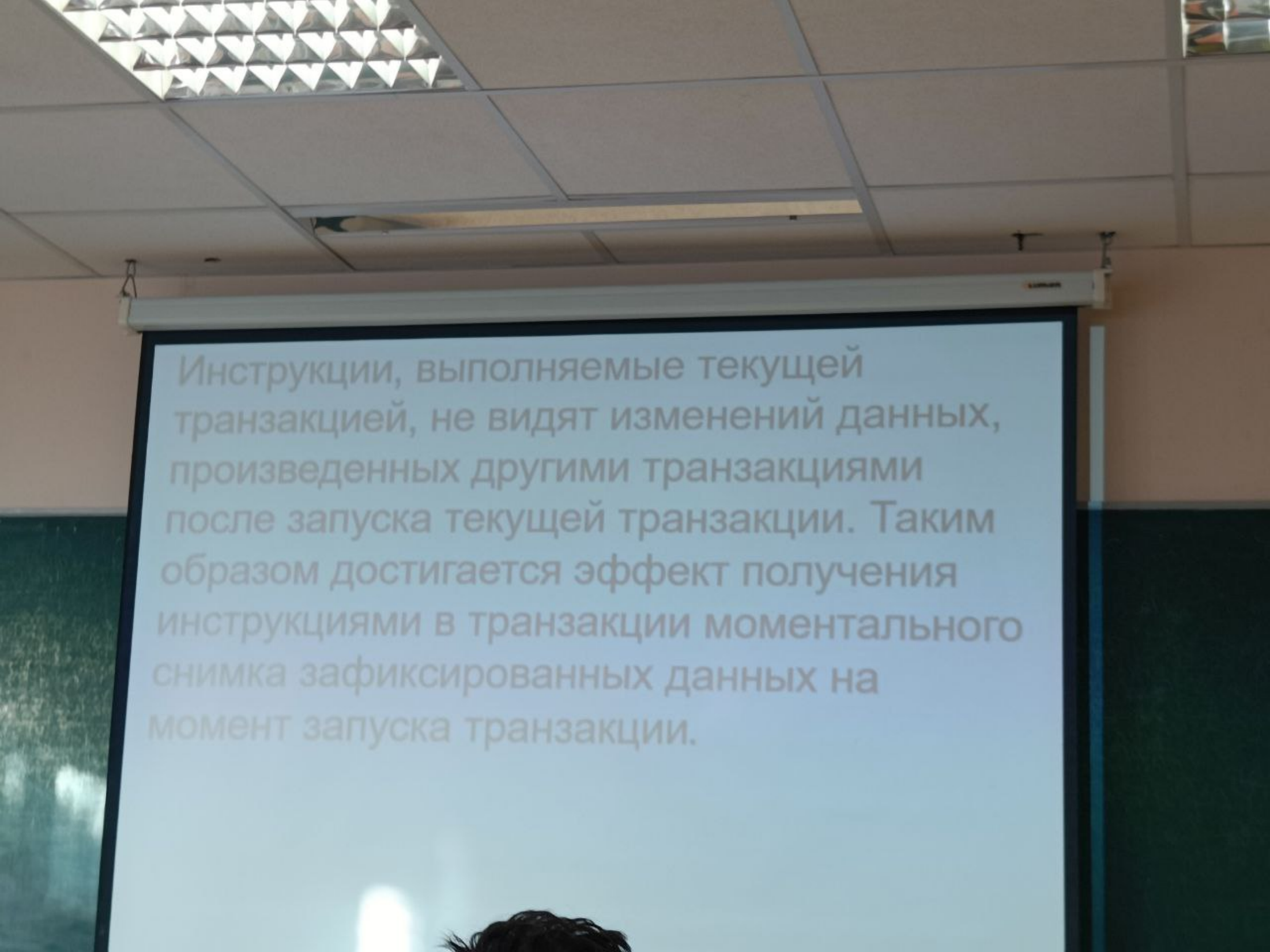
Указывает, что инструкции могут считывать строки, которые были изменены другими транзакциями, но еще не были зафиксированы. Установка этого параметра позволяет считывать незафиксированные изменения, которые называются чтением «грязных» данных. Это наименьшее ограничение уровней изоляции.

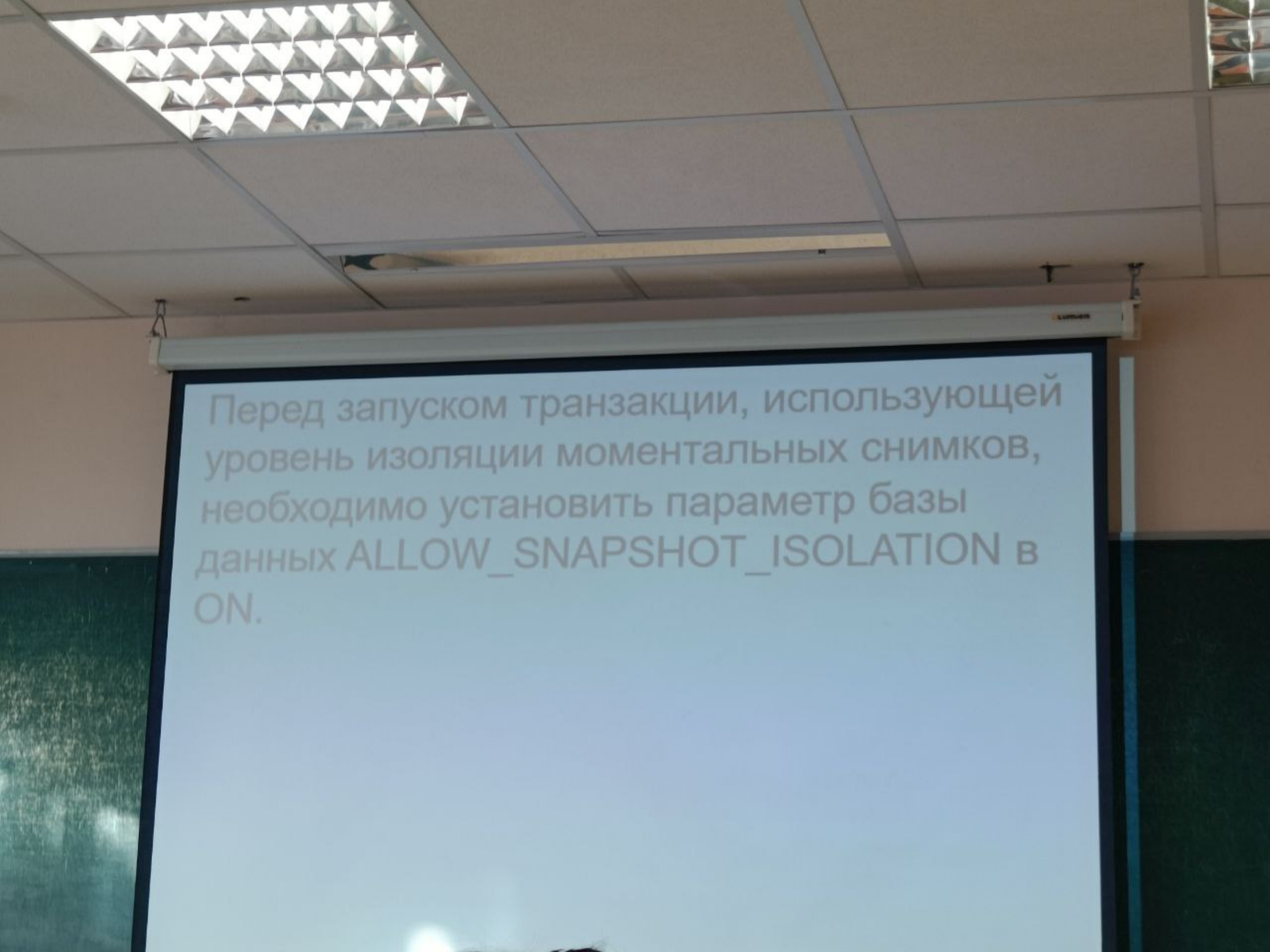

READ COMMITTED (Зафиксированная операция чтения)
Указывает, что инструкции не могут считывать данные, которые были изменены другими транзакциями, но еще не были зафиксированы. Это предотвращает чтение «грязных» данных.

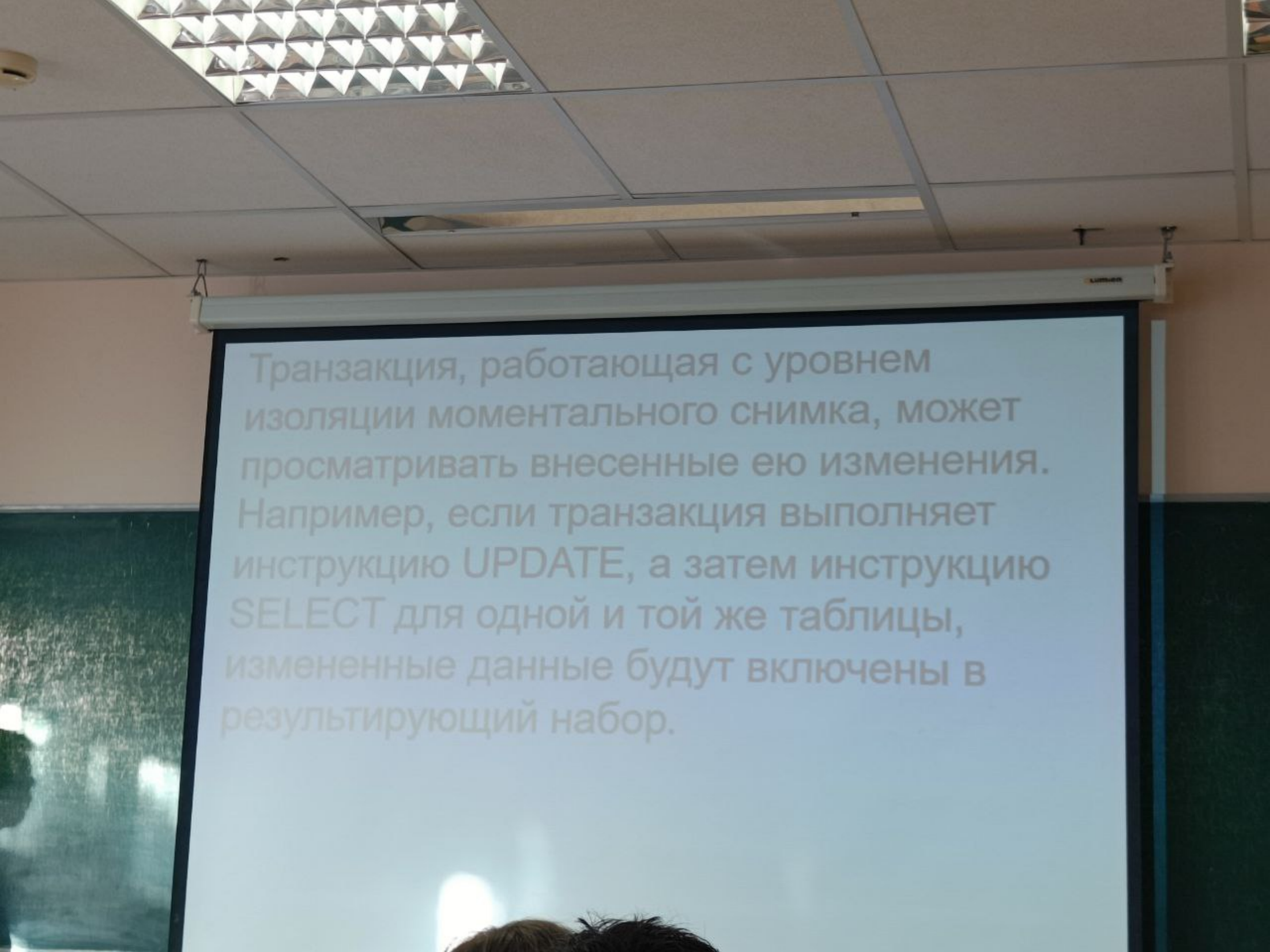

Данные могут быть изменены другими транзакциями между отдельными инструкциями в текущей транзакции, результатом чего будет неповторяющееся чтение или недействительные данные. Этот параметр в SQL Server установлен по умолчанию.

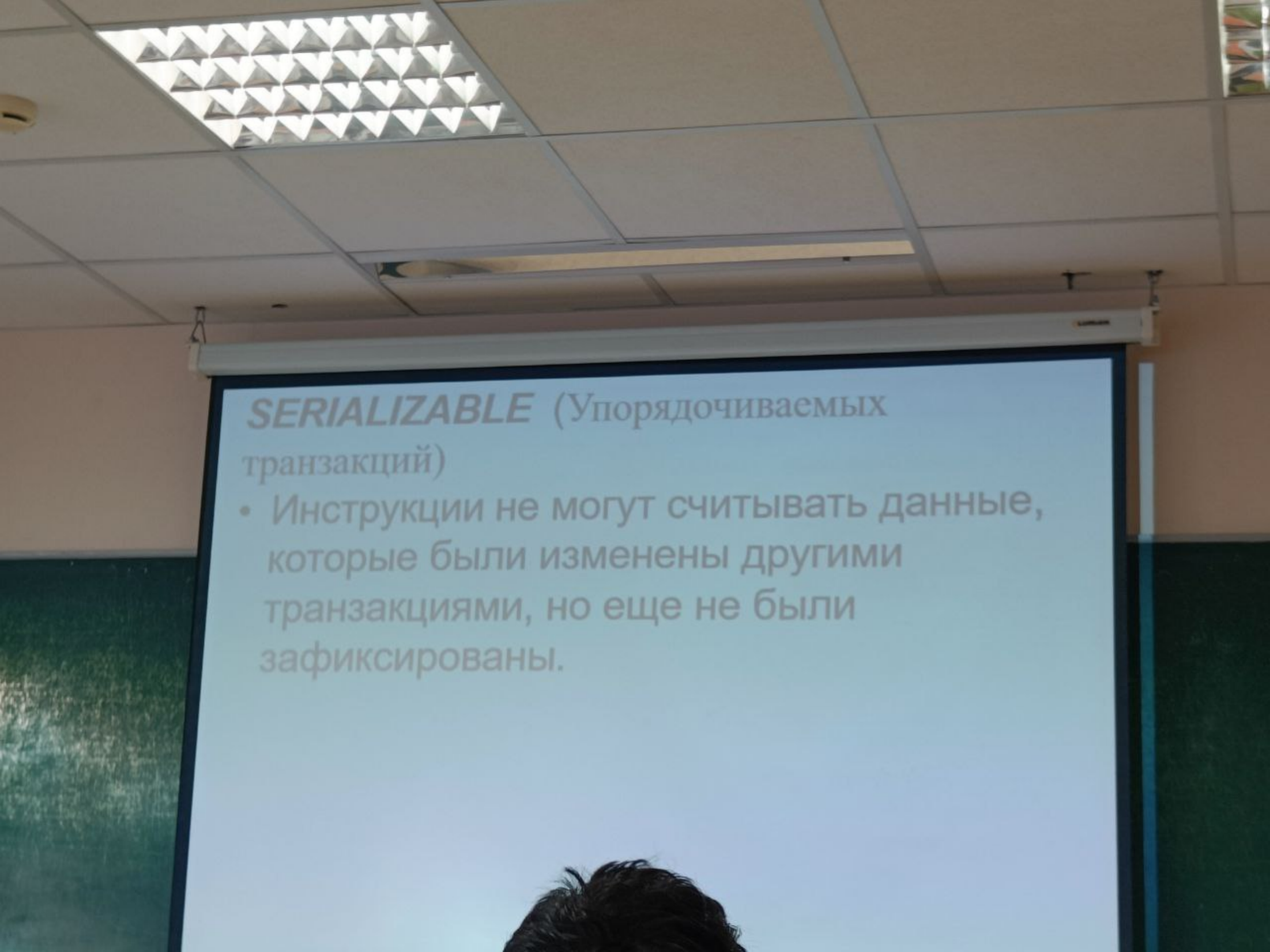

REPEATABLE READ (Повторяющегося чтения)
Указывает на то, что инструкции не могут считывать данные, которые были изменены, но еще не зафиксированы другими транзакциями, а также на то, что другие транзакции не могут изменять данные, читаемые текущей транзакцией, до ее завершения.


SNAPSHOT (Моментального снимка)
Указывает на то, что данные, считанные любой инструкцией транзакции, будут согласованы на уровне транзакции с версией данных, существовавших в ее начале. Транзакция распознает только те изменения, которые были зафиксированы до ее начала.

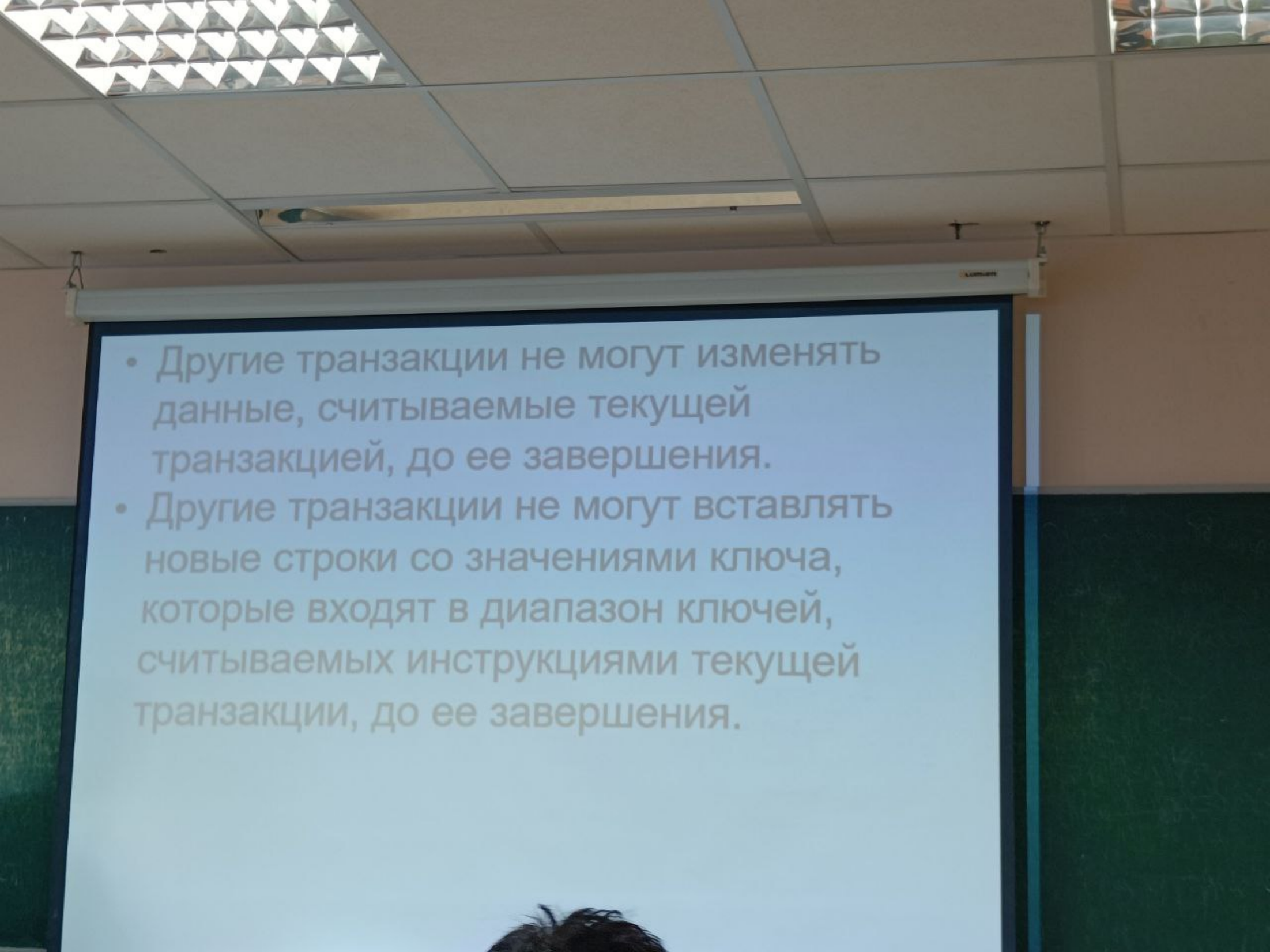

Инструкции, выполняемые текущей транзакцией, не видят изменений данных, произведенных другими транзакциями после запуска текущей транзакции. Таким образом достигается эффект получения инструкциями в транзакции моментального снимка зафиксированных данных на момент запуска транзакции.


Перед запуском транзакции, использующей уровень изоляции моментальных снимков, необходимо установить параметр базы данных ALLOW_SNAPSHOT_ISOLATION в ON.

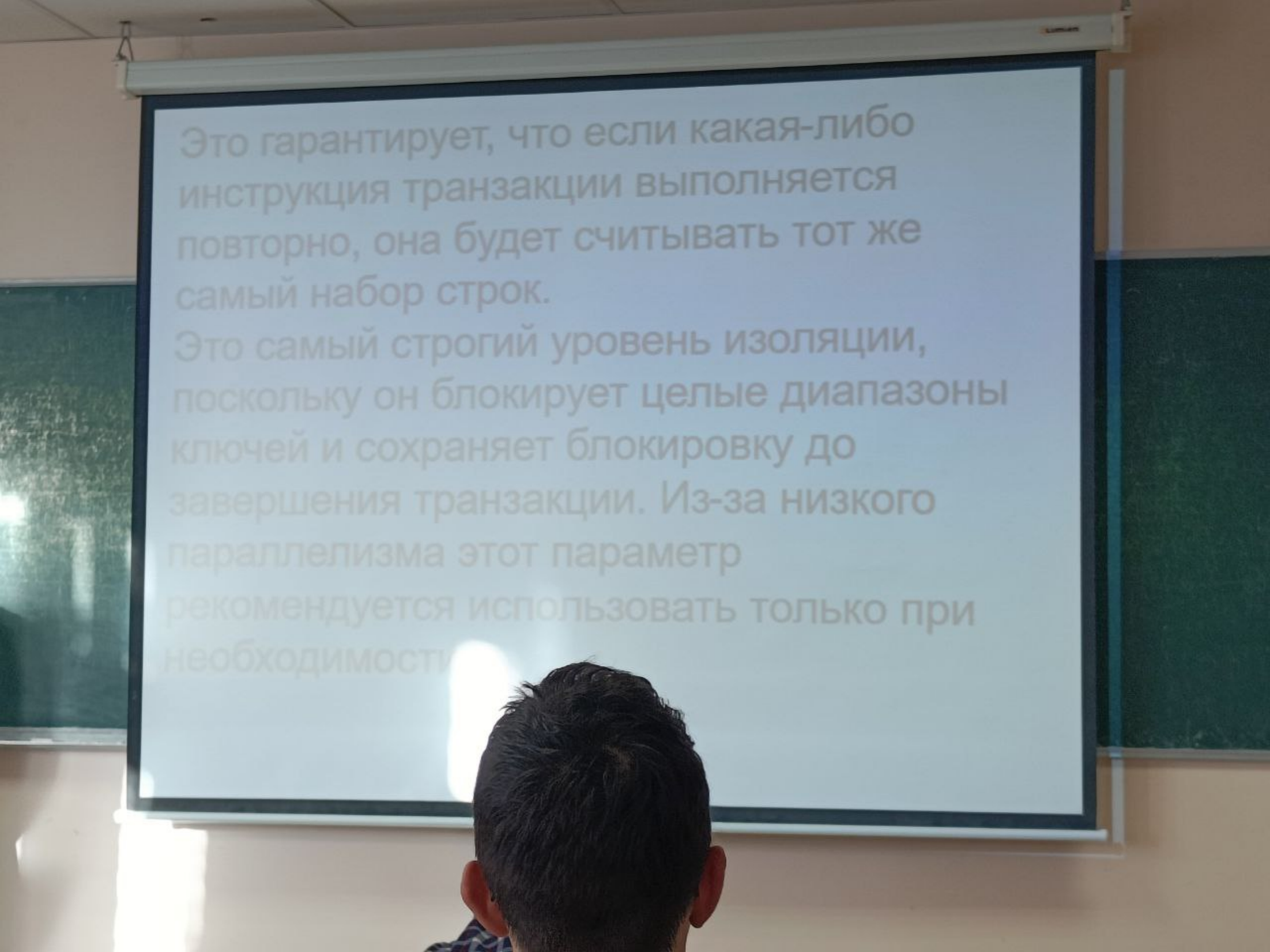

Транзакция, работающая с уровнем изоляции моментального снимка, может просматривать внесенные ею изменения. Например, если транзакция выполняет инструкцию UPDATE, а затем инструкцию SELECT для одной и той же таблицы, измененные данные будут включены в результирующий набор.

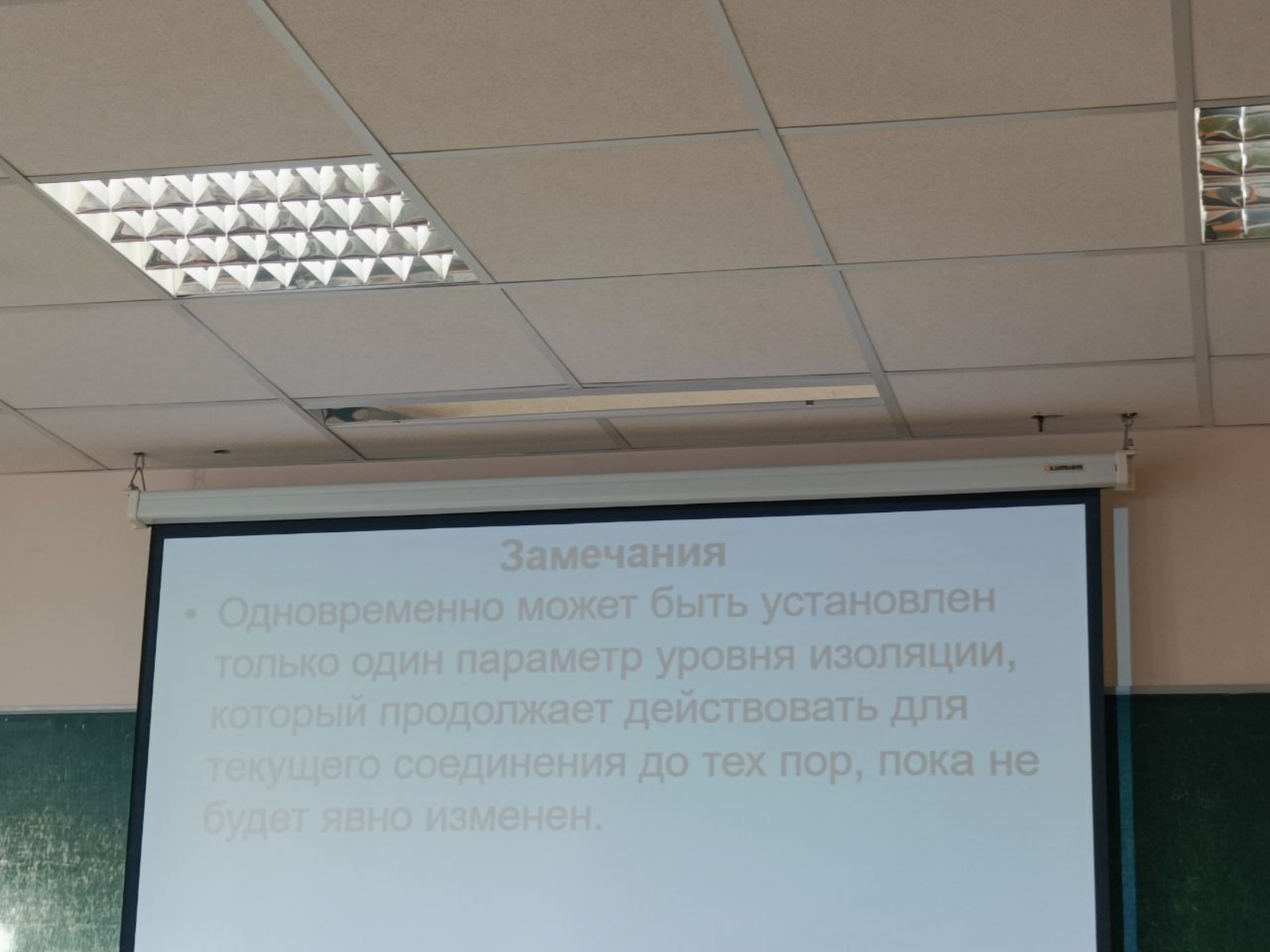

SERIALIZABLE (Упорядочиваемых транзакций)
• Инструкции не могут считывать данные, которые были изменены другими транзакциями, но еще не были зафиксированы.

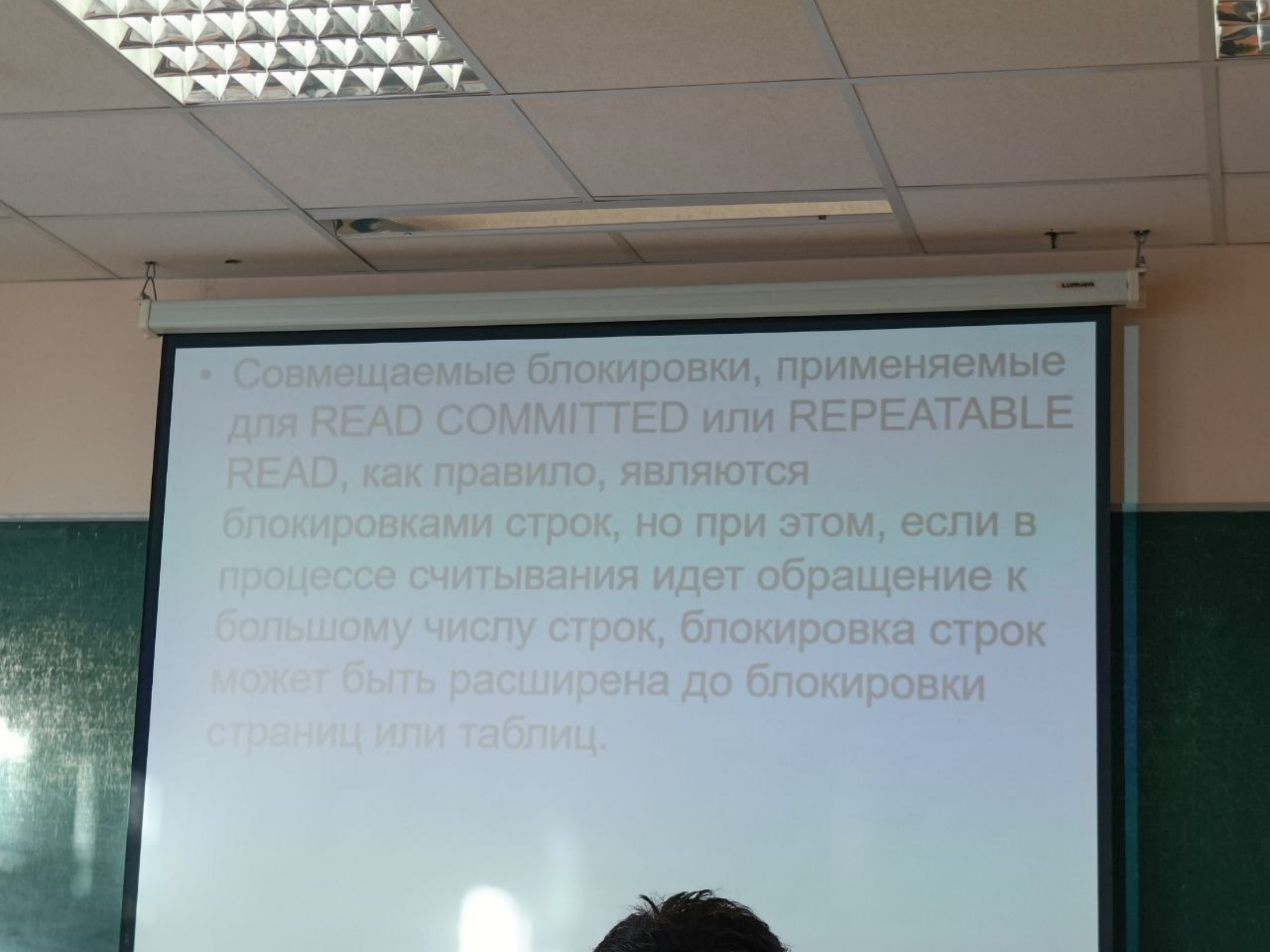
- Другие транзакции не могут изменять данные, считываемые текущей транзакцией, до ее завершения.
- Другие транзакции не могут вставлять новые строки со значениями ключа, которые входят в диапазон ключей, считываемых инструкциями текущей транзакции, до ее завершения.

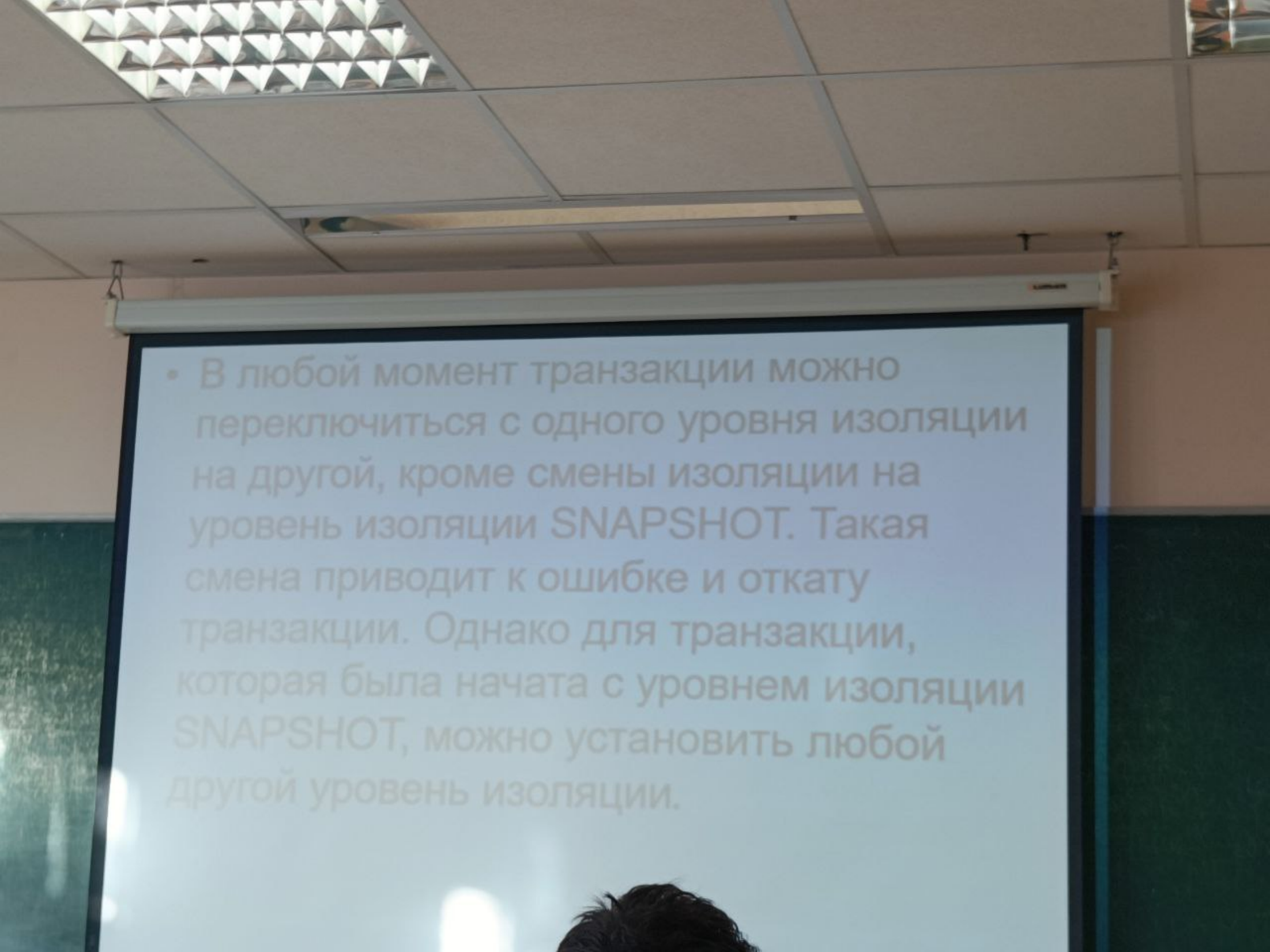


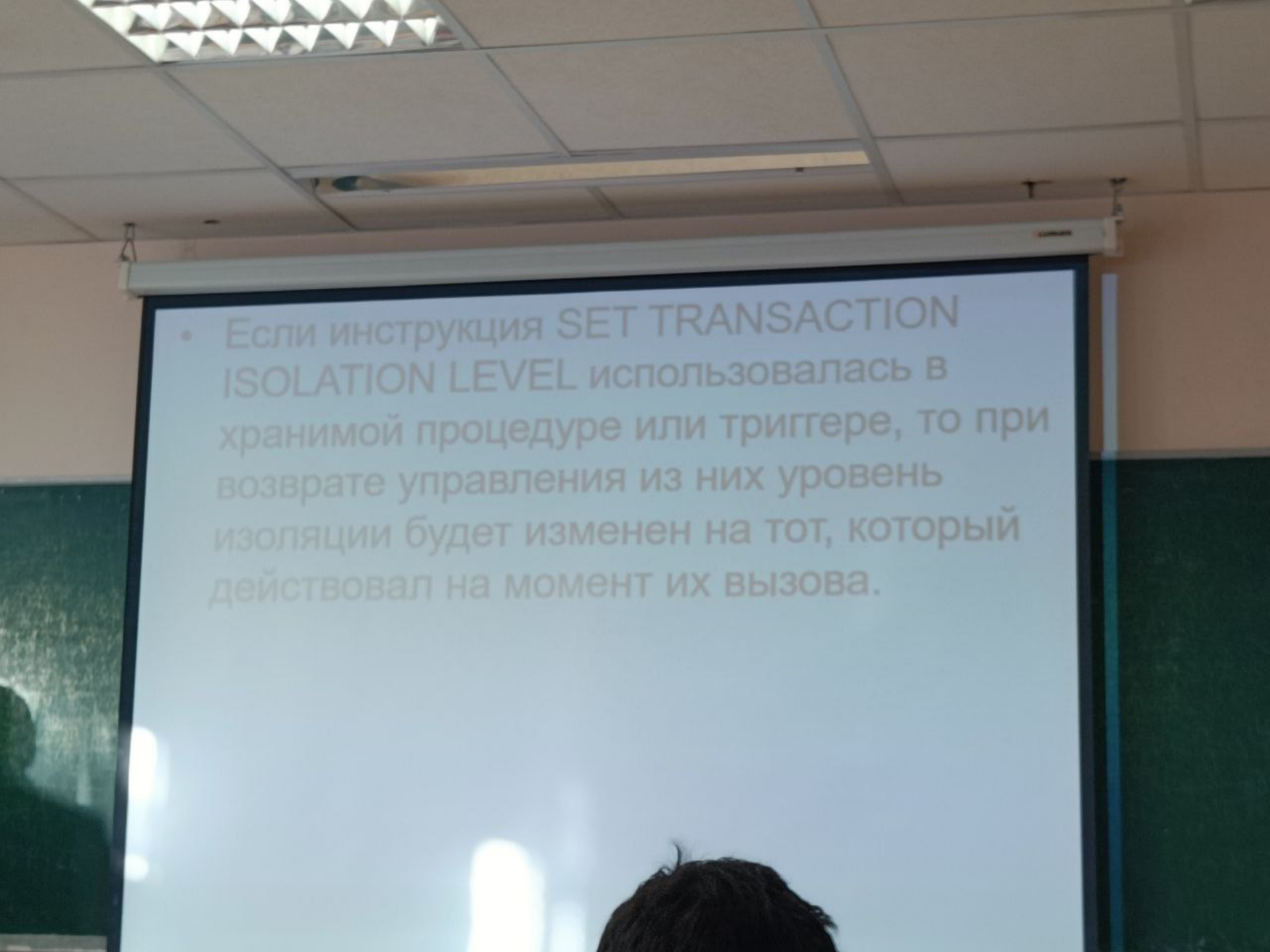
Блокировка диапазона устанавливается в диапазоне значений ключа, соответствующих условиям поиска любой инструкции, выполненной во время транзакции. Обновление и вставка строк, удовлетворяющих инструкциям текущей транзакции, блокируется для других транзакций.

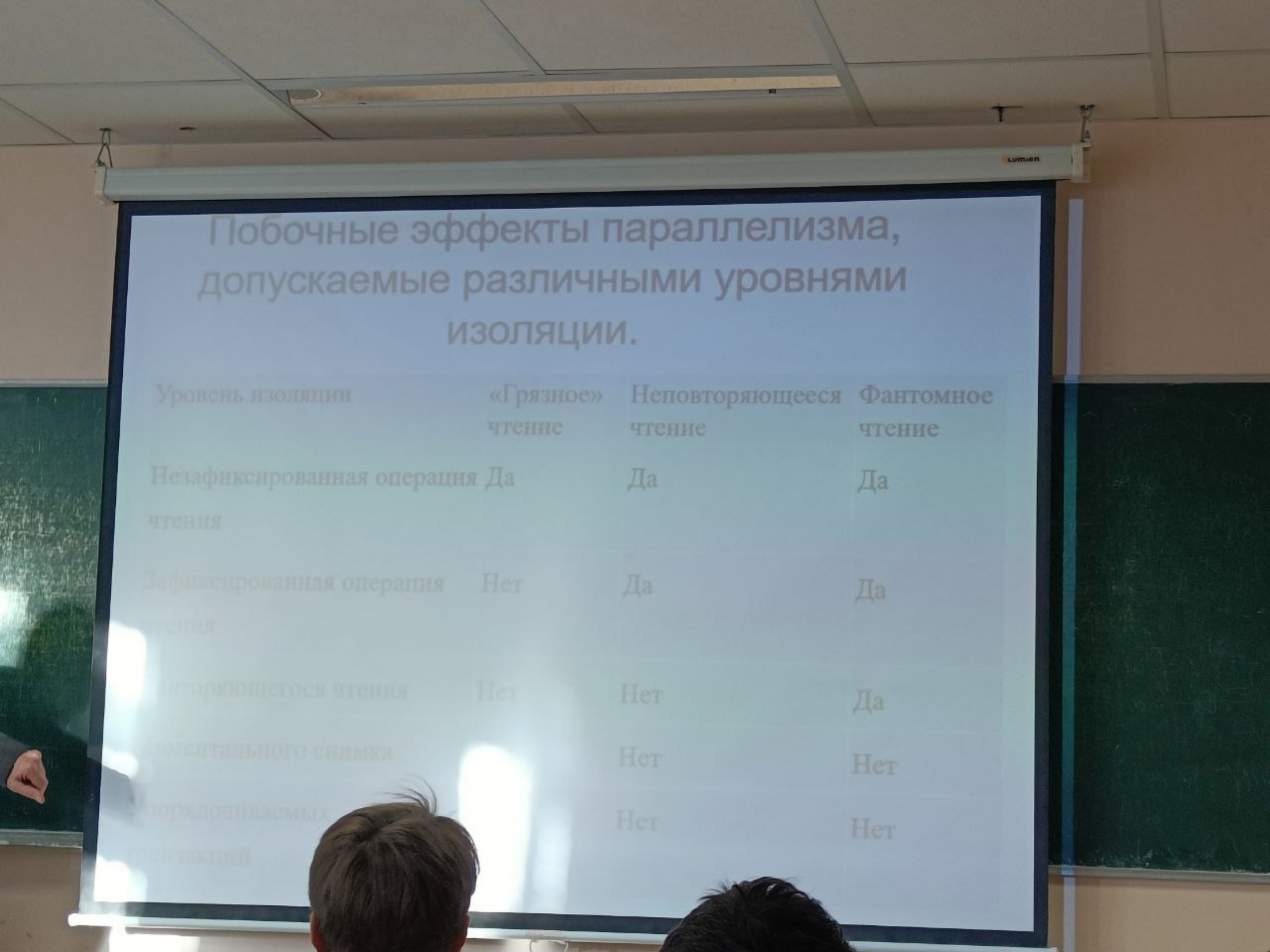

Это гарантирует, что если какая-либо инструкция транзакции выполняется повторно, она будет считывать тот же самый набор строк.
Это самый строгий уровень изоляции, поскольку он блокирует целые диапазоны ключей и сохраняет блокировку до завершения транзакции. Из-за низкого параллелизма этот параметр рекомендуется использовать только при необходимости

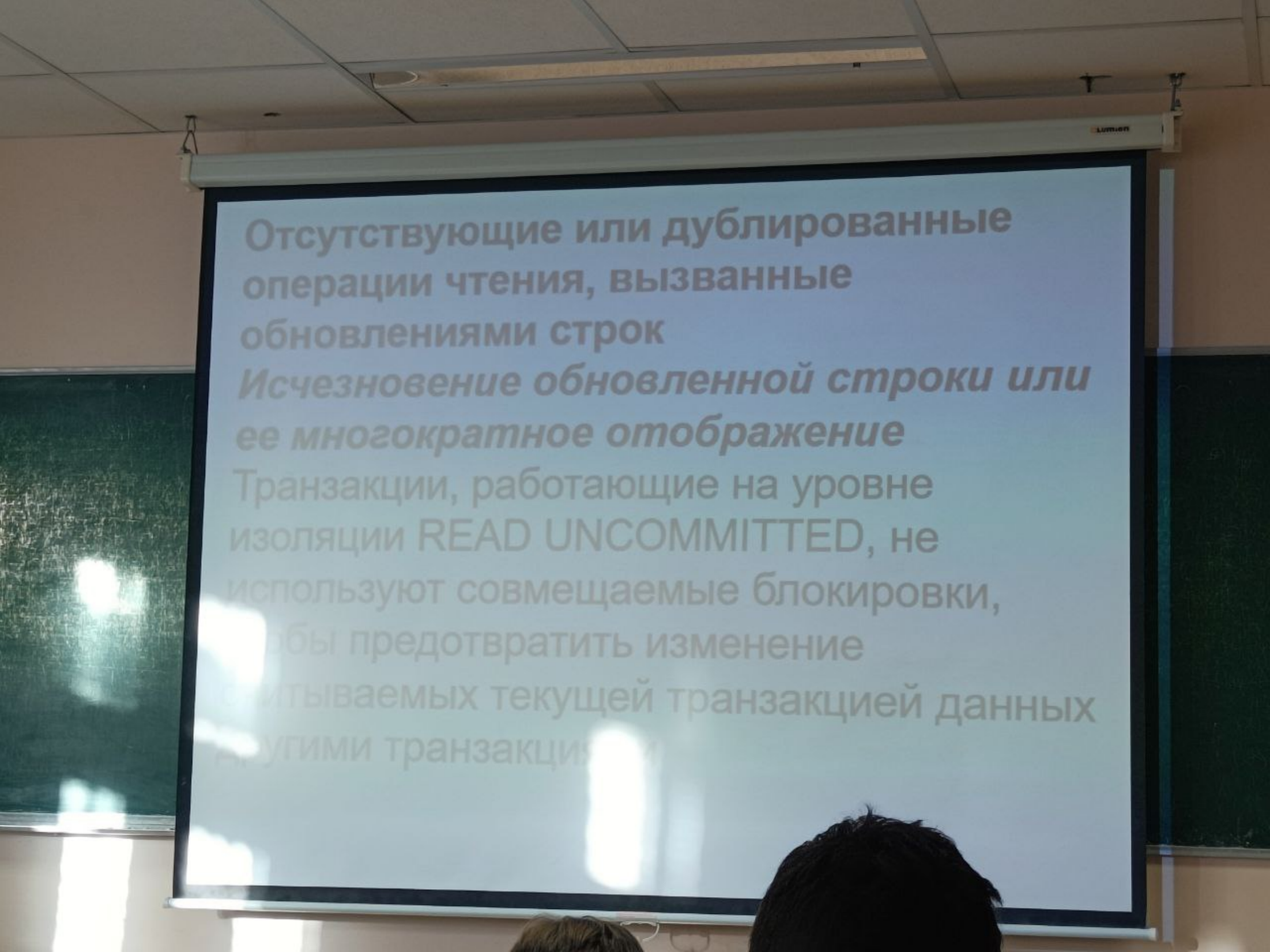

Замечания
• Одновременно может быть установлен только один параметр уровня изоляции, который продолжает действовать для текущего соединения до тех пор, пока не будет явно изменен.

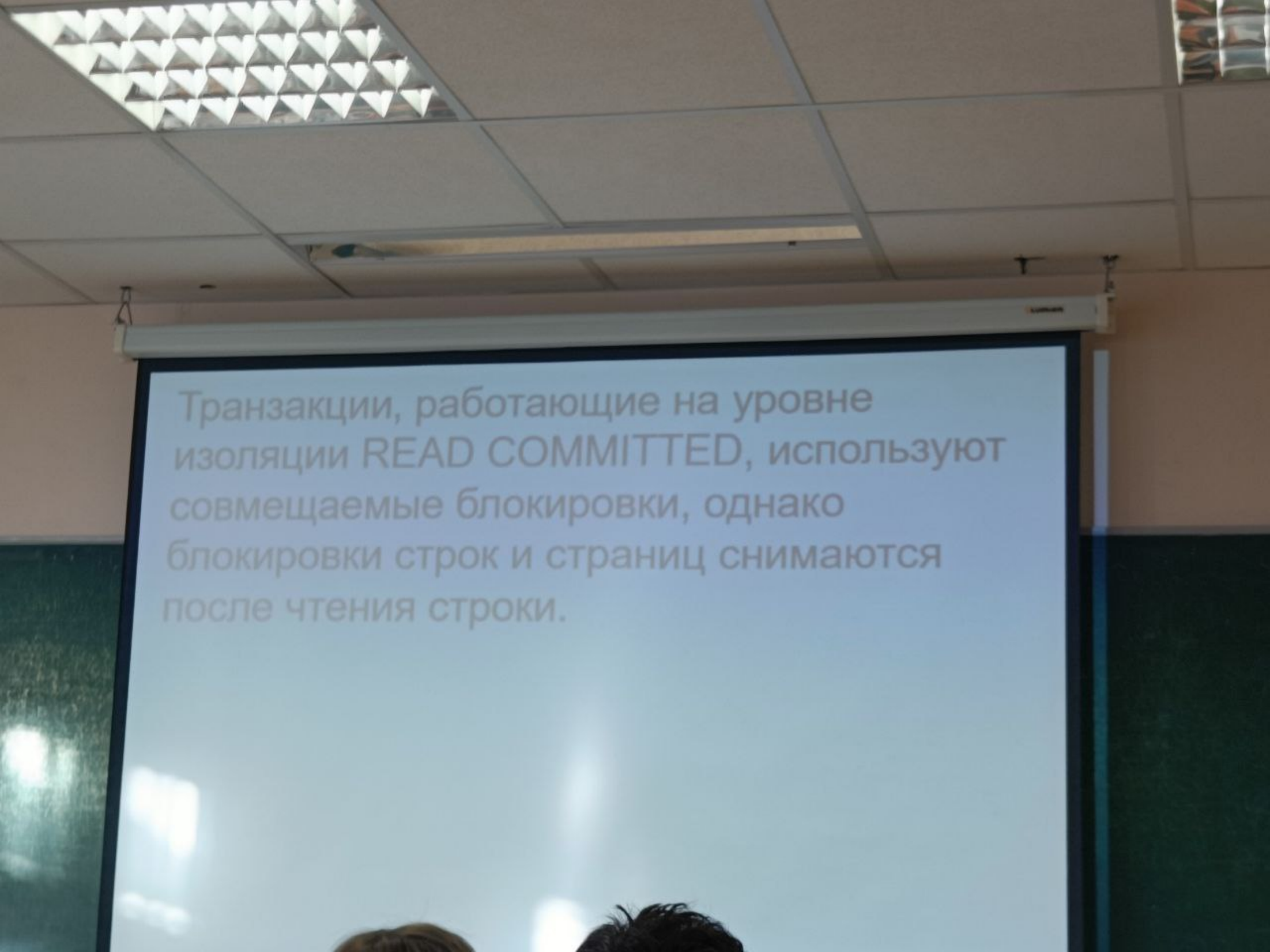

• Совмещаемые блокировки, применяемые для READ COMMITTED или REPEATABLE READ, как правило, являются блокировками строк, но при этом, если в процессе считывания идет обращение к большому числу строк, блокировка строк может быть расширена до блокировки страниц или таблиц.

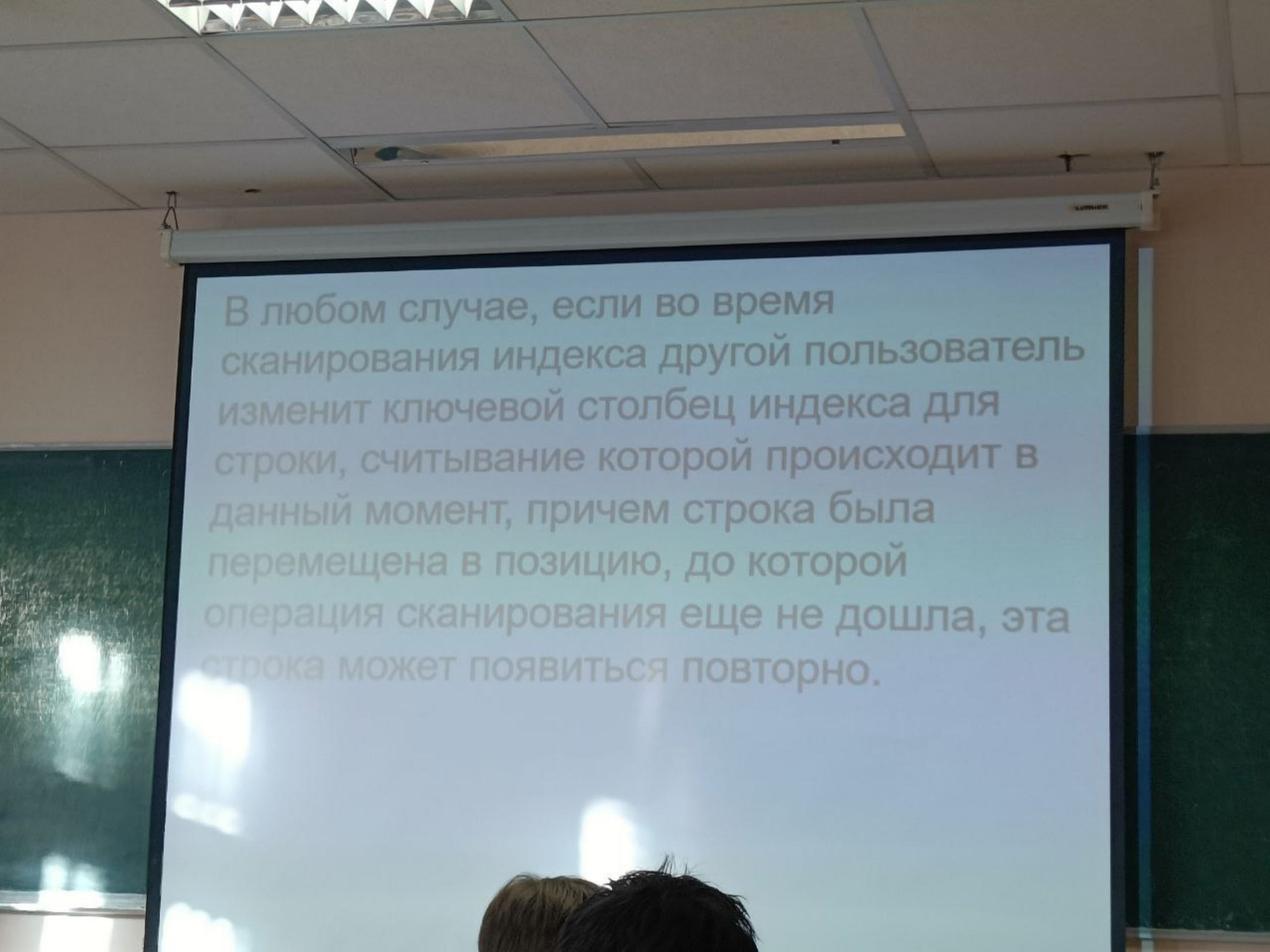

• В любой момент транзакции можно переключиться с одного уровня изоляции на другой, кроме смены изоляции на уровень изоляции SNAPSHOT. Такая смена приводит к ошибке и откату транзакции. Однако для транзакции, которая была начата с уровнем изоляции SNAPSHOT, можно установить любой другой уровень изоляции.

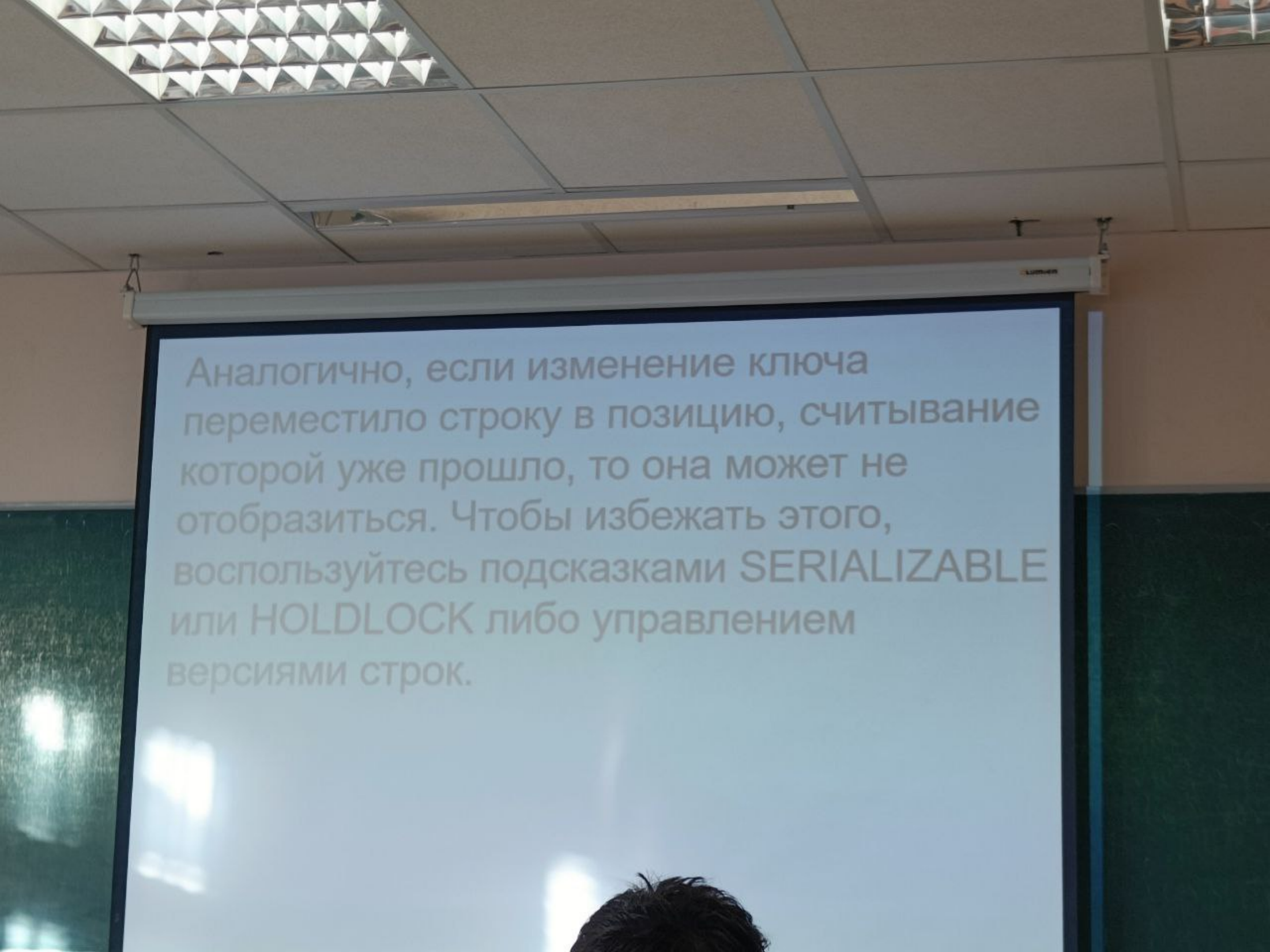

• Если инструкция SET TRANSACTION ISOLATION LEVEL использовалась в хранимой процедуре или триггере, то при возврате управления из них уровень изоляции будет изменен на тот, который действовал на момент их вызова.

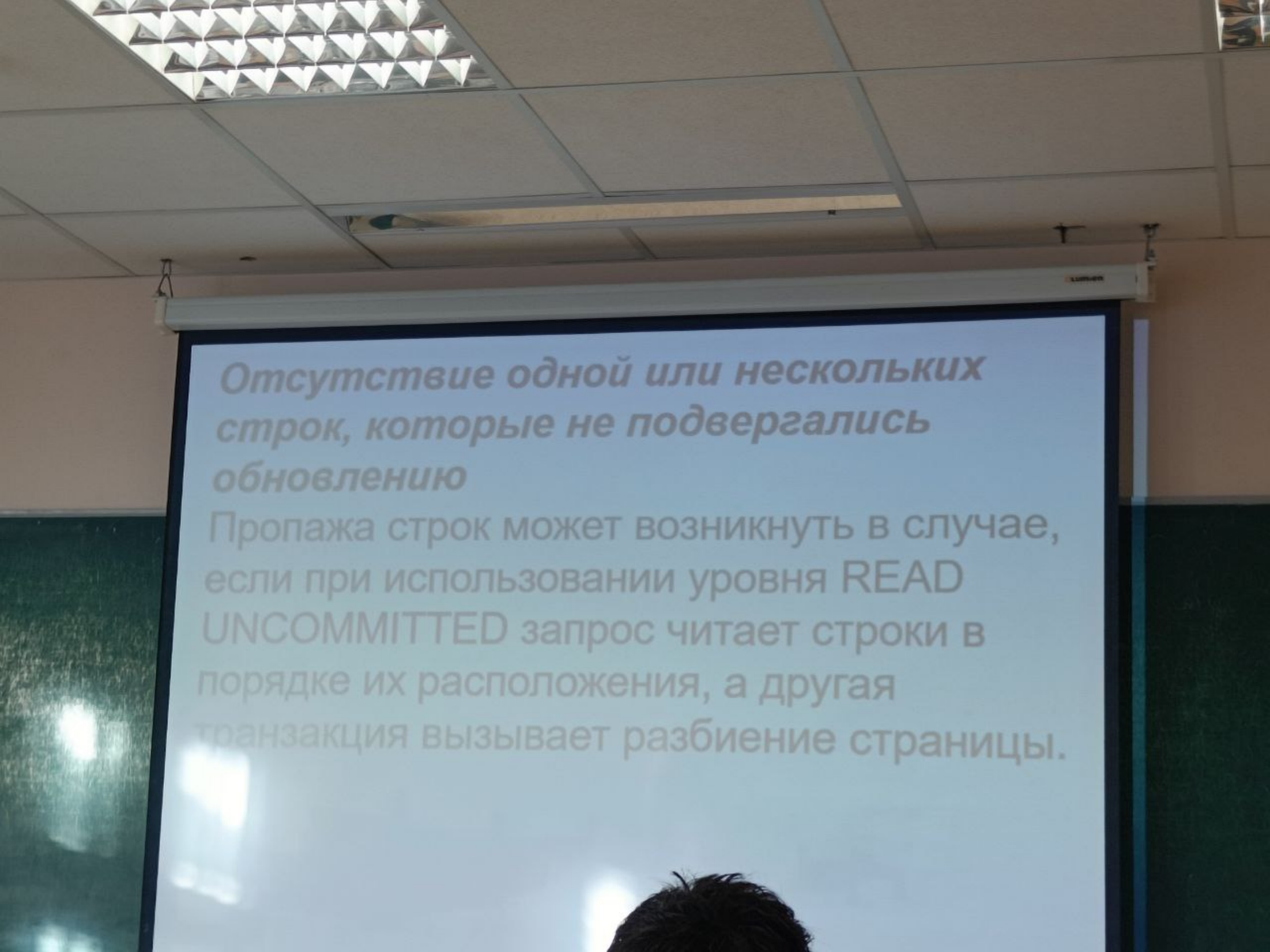

Побочные эффекты параллелизма, допускаемые различными уровнями изоляции.
Уровень изоляции
«Грязное» чтение
Неповторяющееся чтение
Фантомное чтение
Незафиксированная операция чтения
Да
Да
Да
Зафиксированная операция чтения
Нет
Да
Да
Нет
Нет
Да
Нет
Нет
Нет
Нет


Отсутствующие или дублированные операции чтения, вызванные обновлениями строк
Исчезновение обновленной строки или ее многократное отображение
Транзакции, работающие на уровне изоляции READ UNCOMMITTED, не используют совмещаемые блокировки, чтобы предотвратить изменение считываемых текущей транзакцией данных другими транзакциями.


Транзакции, работающие на уровне изоляции READ COMMITTED, используют совмещаемые блокировки, однако блокировки строк и страниц снимаются после чтения строки.


В любом случае, если во время сканирования индекса другой пользователь изменит ключевой столбец индекса для строки, считывание которой происходит в данный момент, причем строка была перемещена в позицию, до которой операция сканирования еще не дошла, эта строка может появиться повторно.


Аналогично, если изменение ключа переместило строку в позицию, считывание которой уже прошло, то она может не отобразиться. Чтобы избежать этого, воспользуйтесь подсказками SERIALIZABLE или HOLDLOCK либо управлением версиями строк.


Отсутствие одной или нескольких строк, которые не подвергались обновлению
Пропажа строк может возникнуть в случае, если при использовании уровня READ UNCOMMITTED запрос читает строки в порядке их расположения, а другая транзакция вызывает разбиение страницы.

Этого не может произойти при использовании уровня изоляции READ COMMITTED, поскольку во время разбиения страницы включается блокировка таблицы. Также этого не может произойти, если таблица не имеет кластеризованного индекса, поскольку в таком случае обновления не вызывают разбиения страниц.