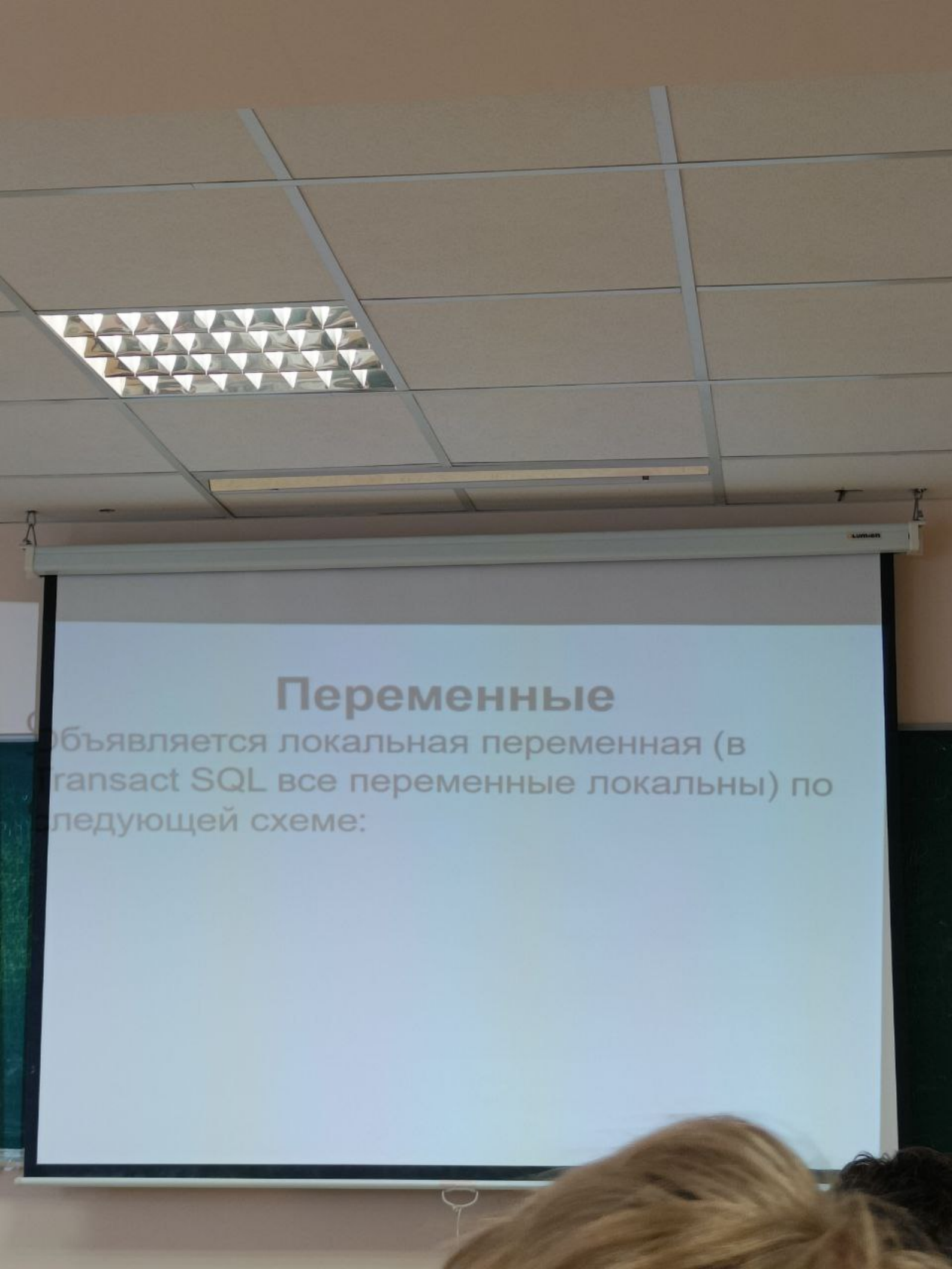
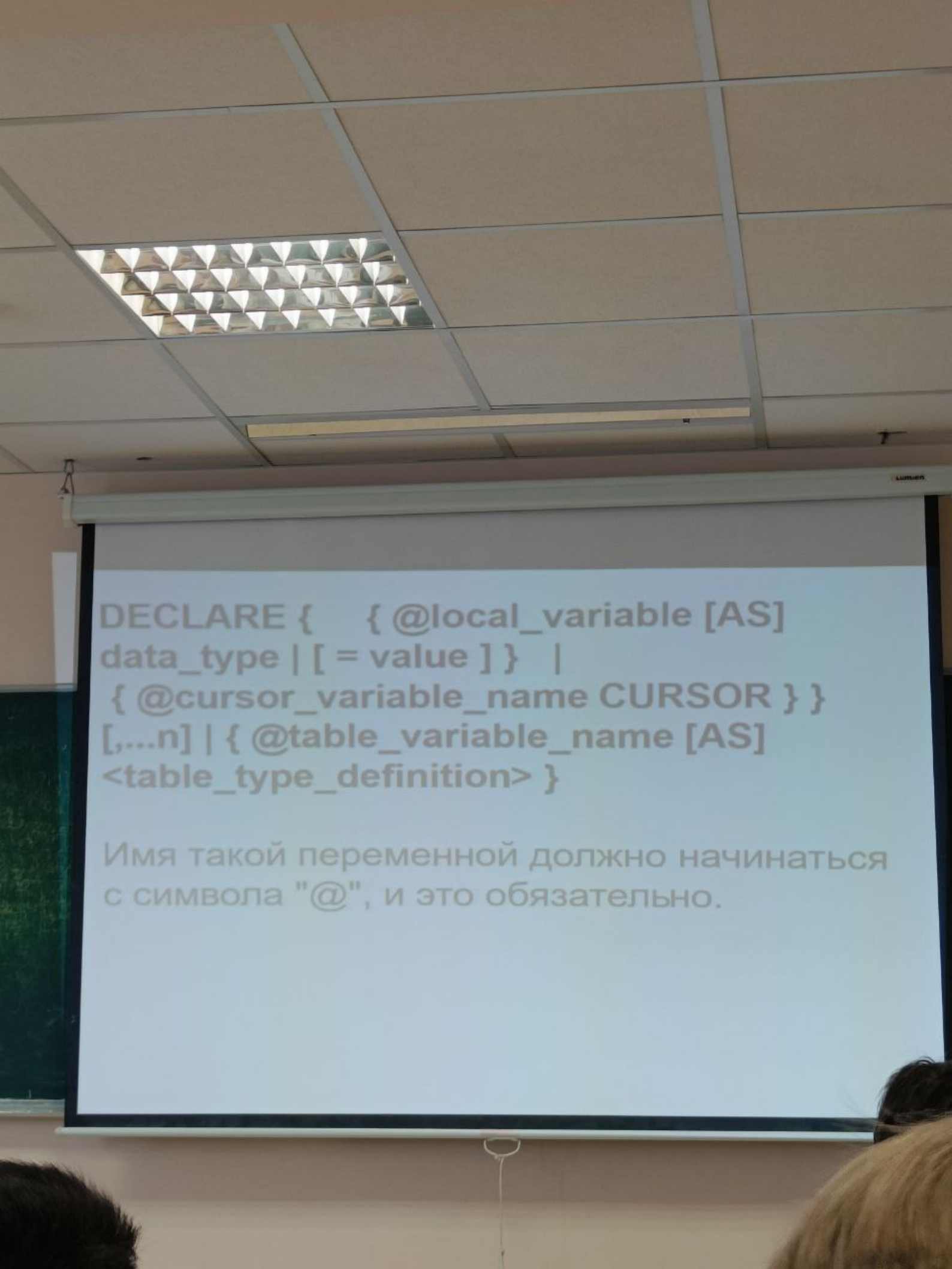
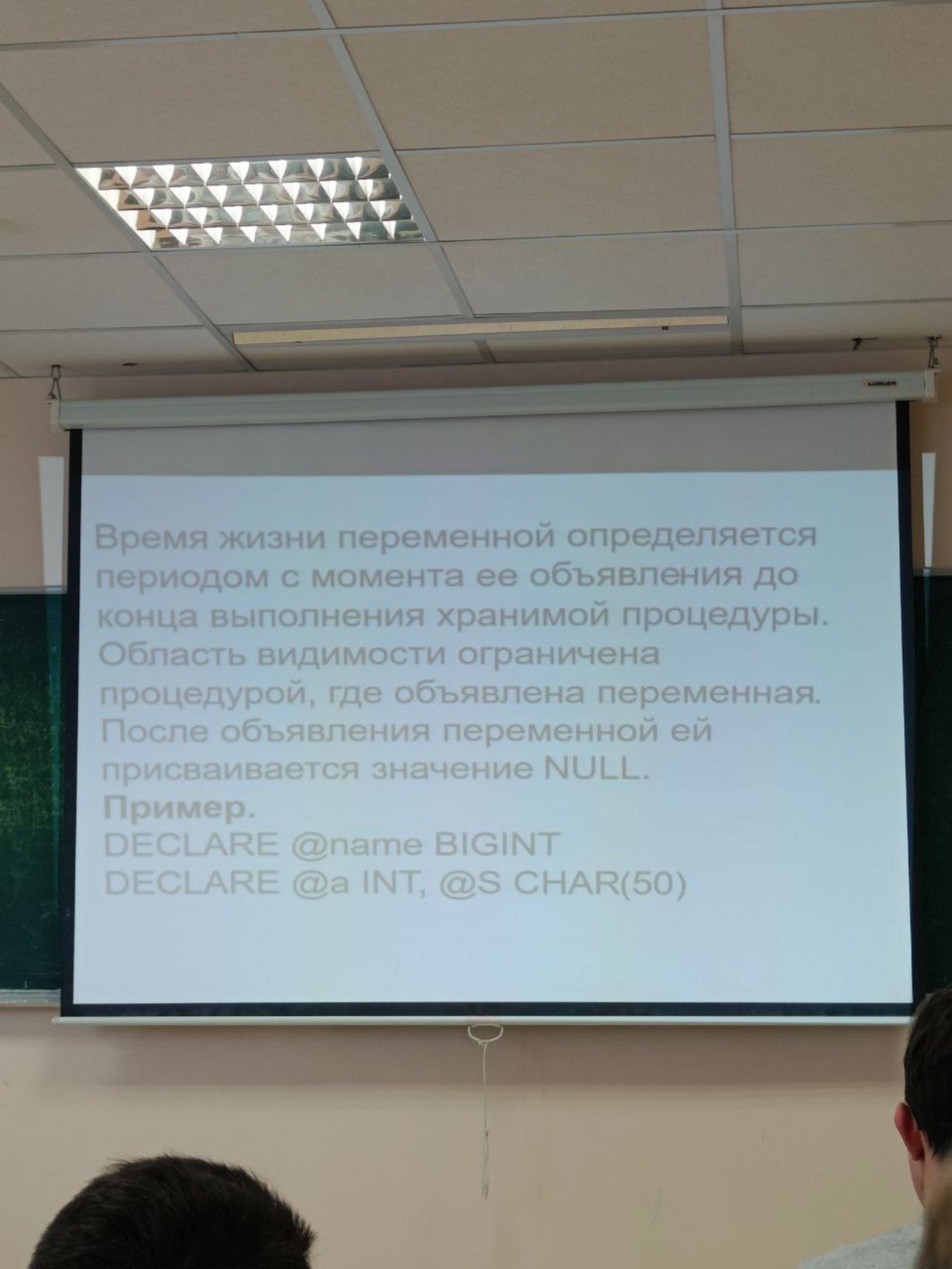
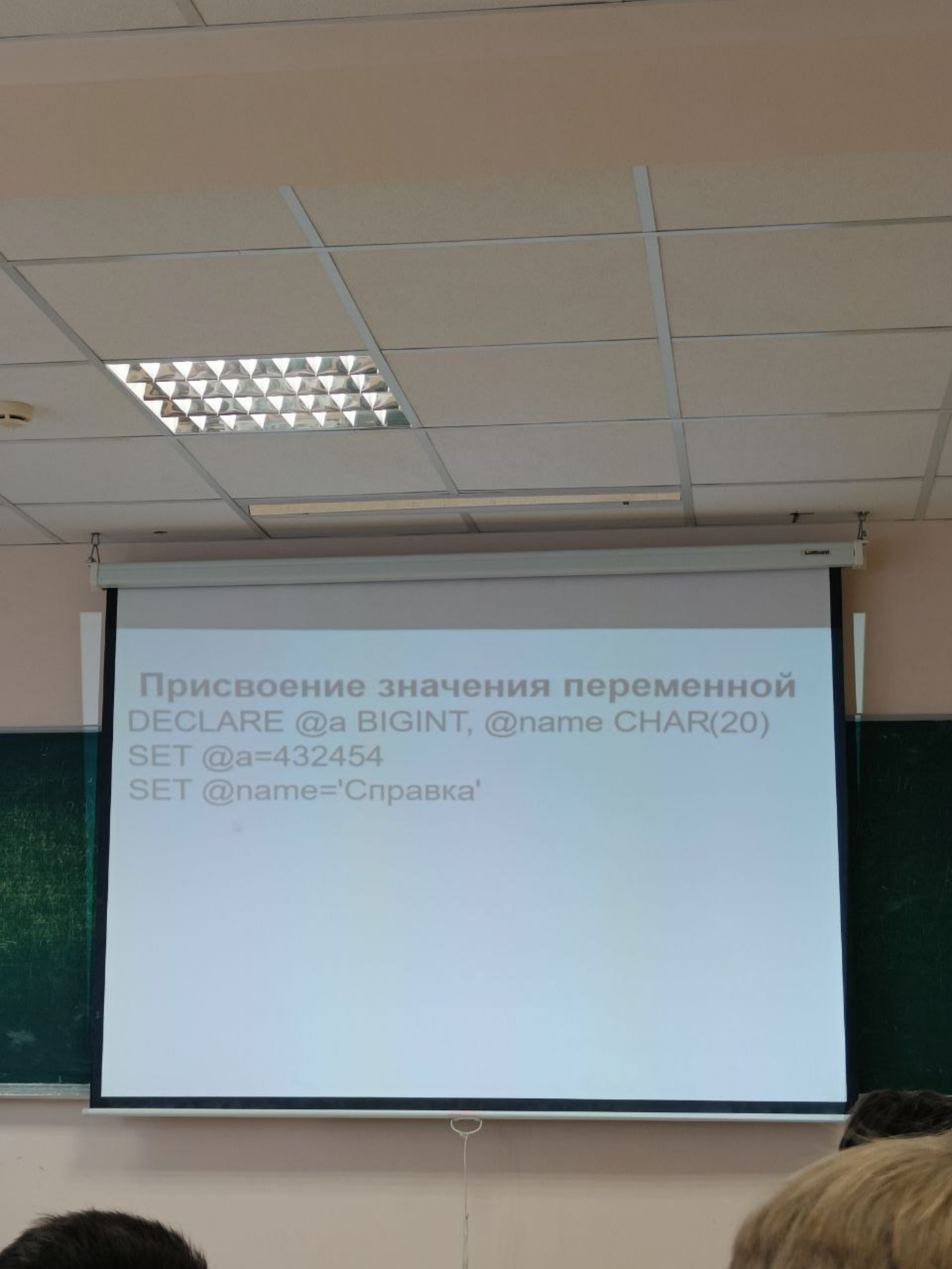
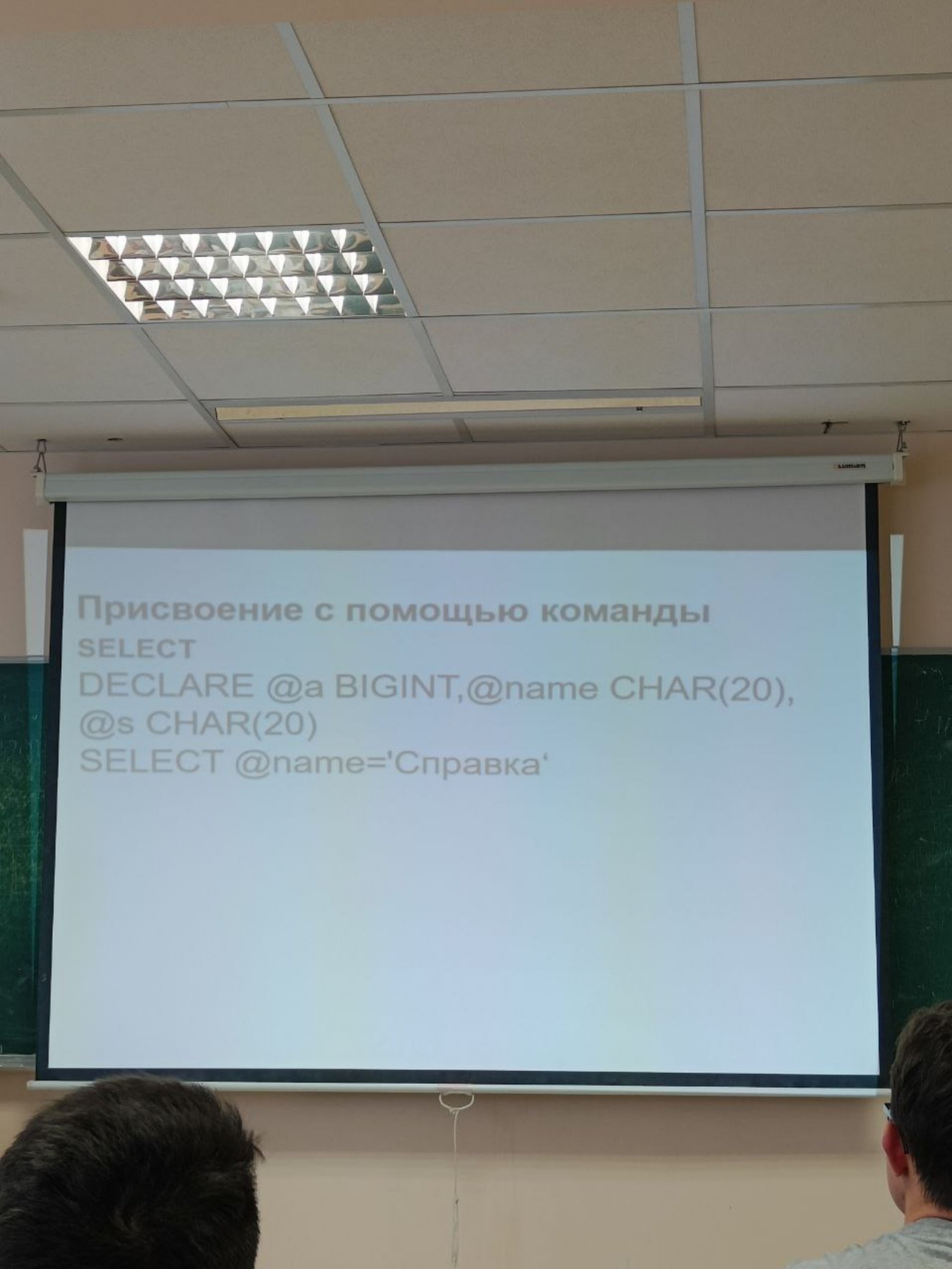
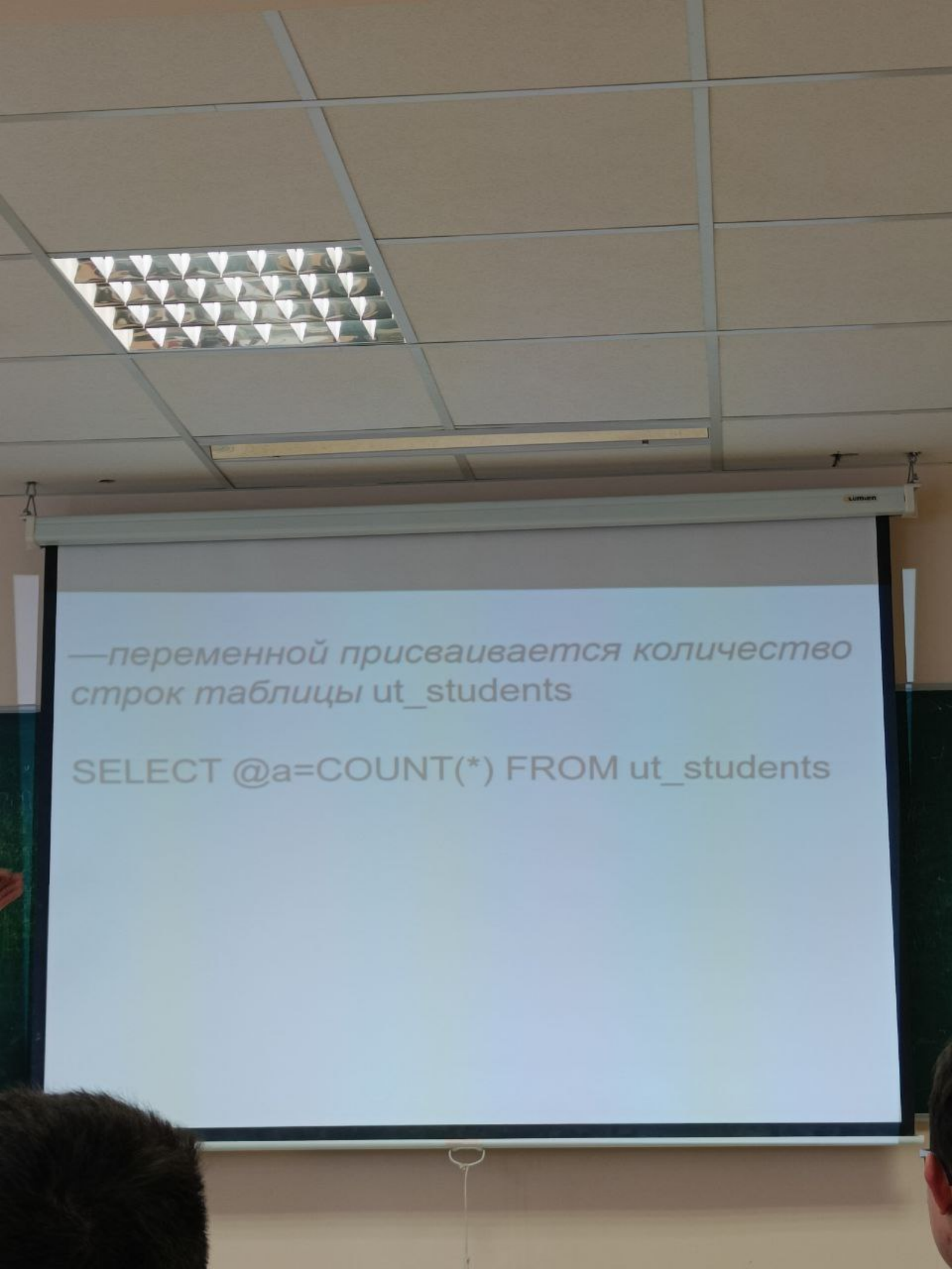

Переменные
бъявляется локальная переменная (в
ransact SQL все переменные локальны) по
ледующей схеме:

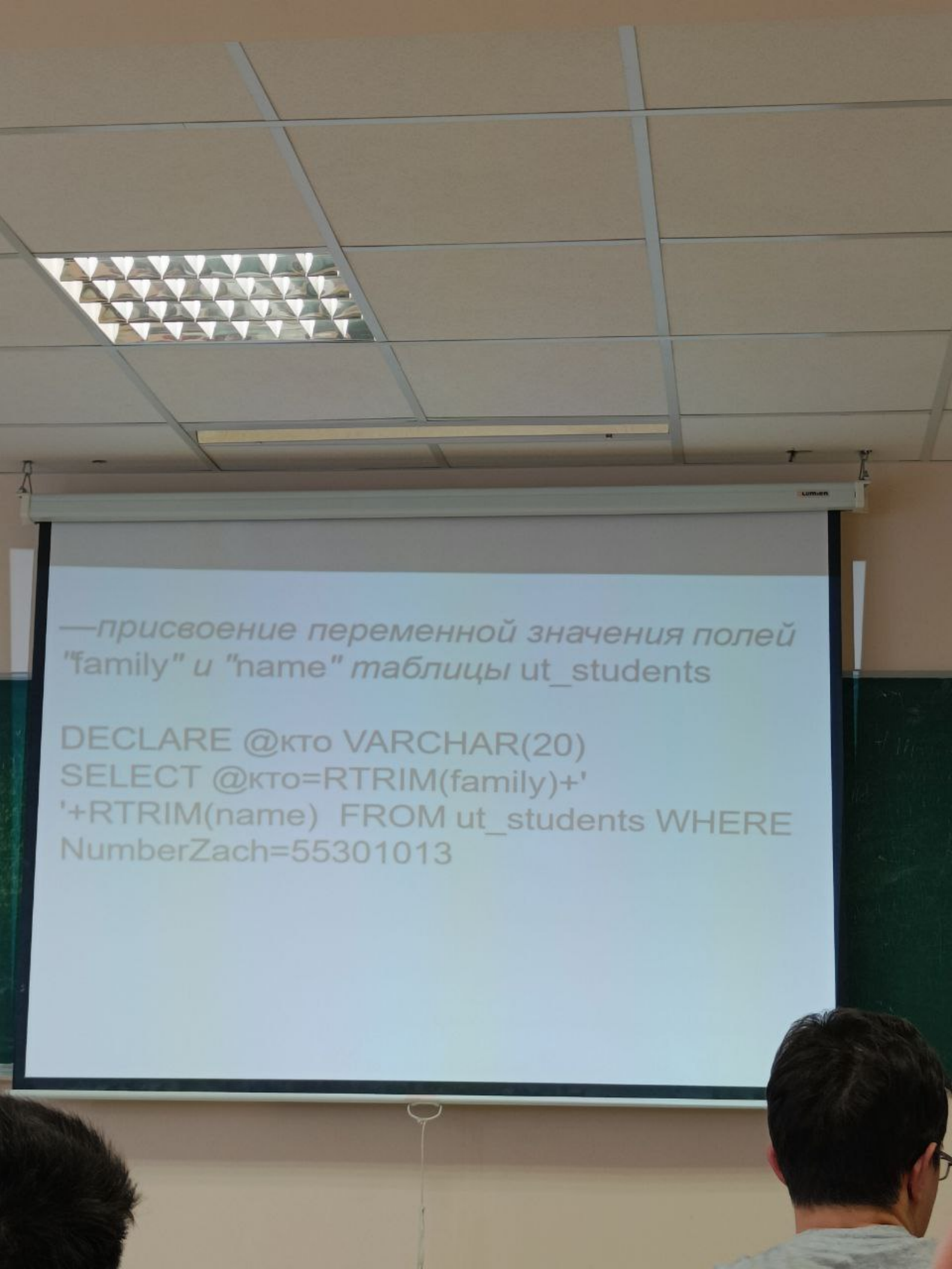

DECLARE { { @local_variable [AS]
data_type | [= value] } |
{ @cursor_variable_name CURSOR } }
[,...n] | { @table_variable_name [AS]
<table_type_definition> }
Имя такой переменной должно начинаться
с символа "@", и это обязательно.

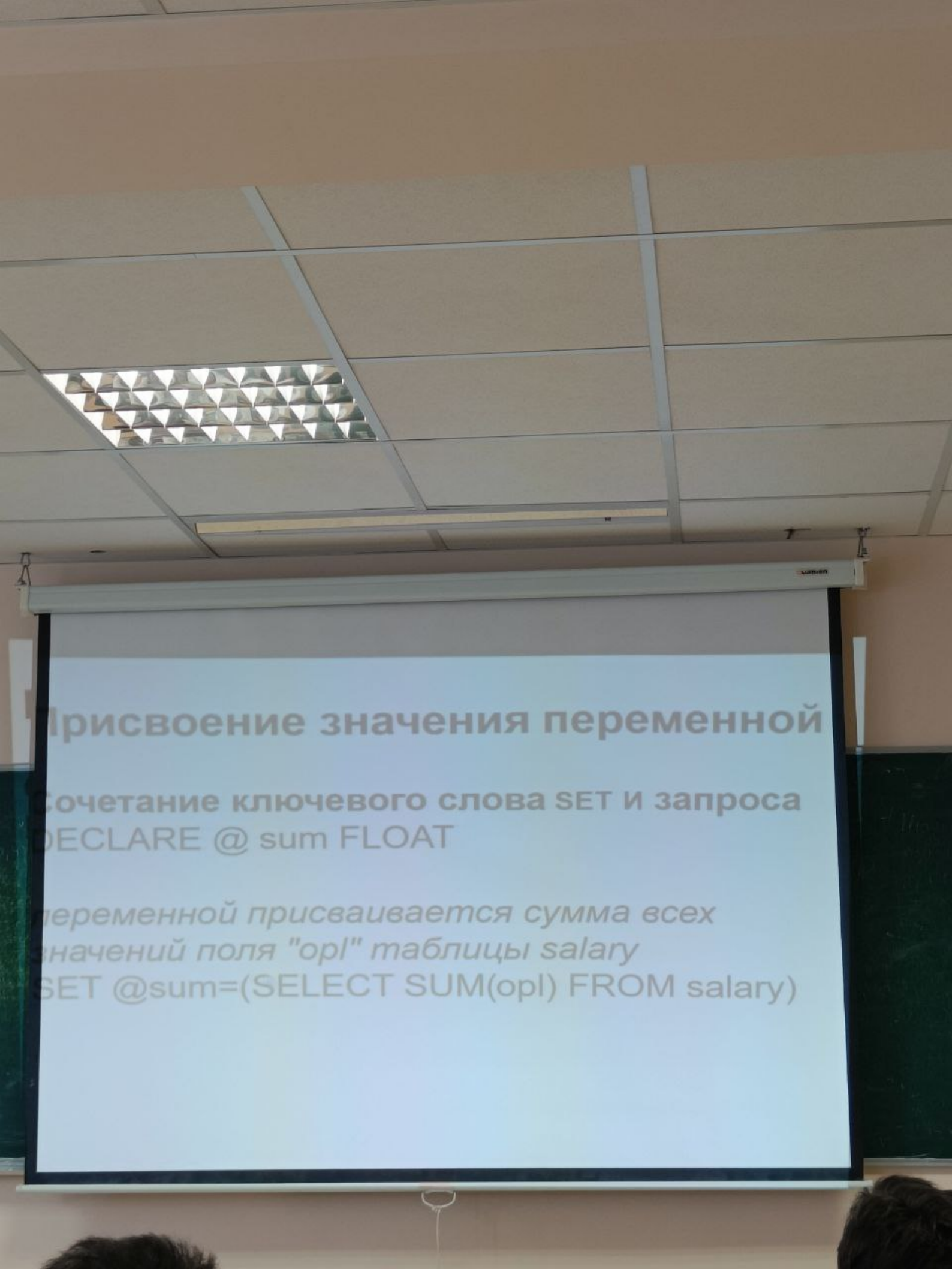

Время жизни переменной определяется периодом с момента ее объявления до конца выполнения хранимой процедуры.
Область видимости ограничена процедурой, где объявлена переменная.
После объявления переменной ей присваивается значение NULL.
Пример.
DECLARE @name BIGINT
DECLARE @a INT, @S CHAR(50)

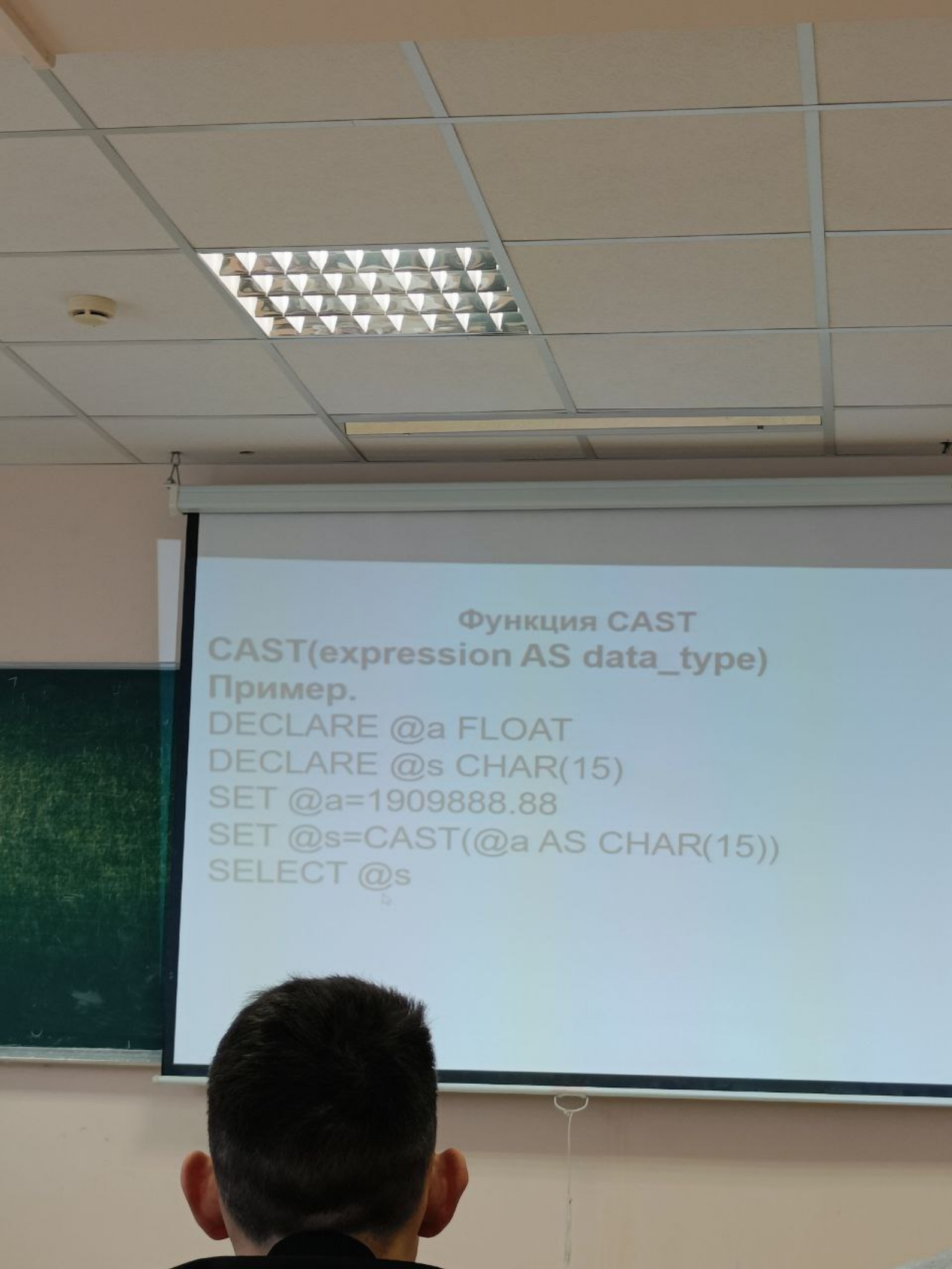

Присвоение значения переменной
DECLARE @a BIGINT, @name CHAR(20)
SET @a=432454
SET @name='Справка'

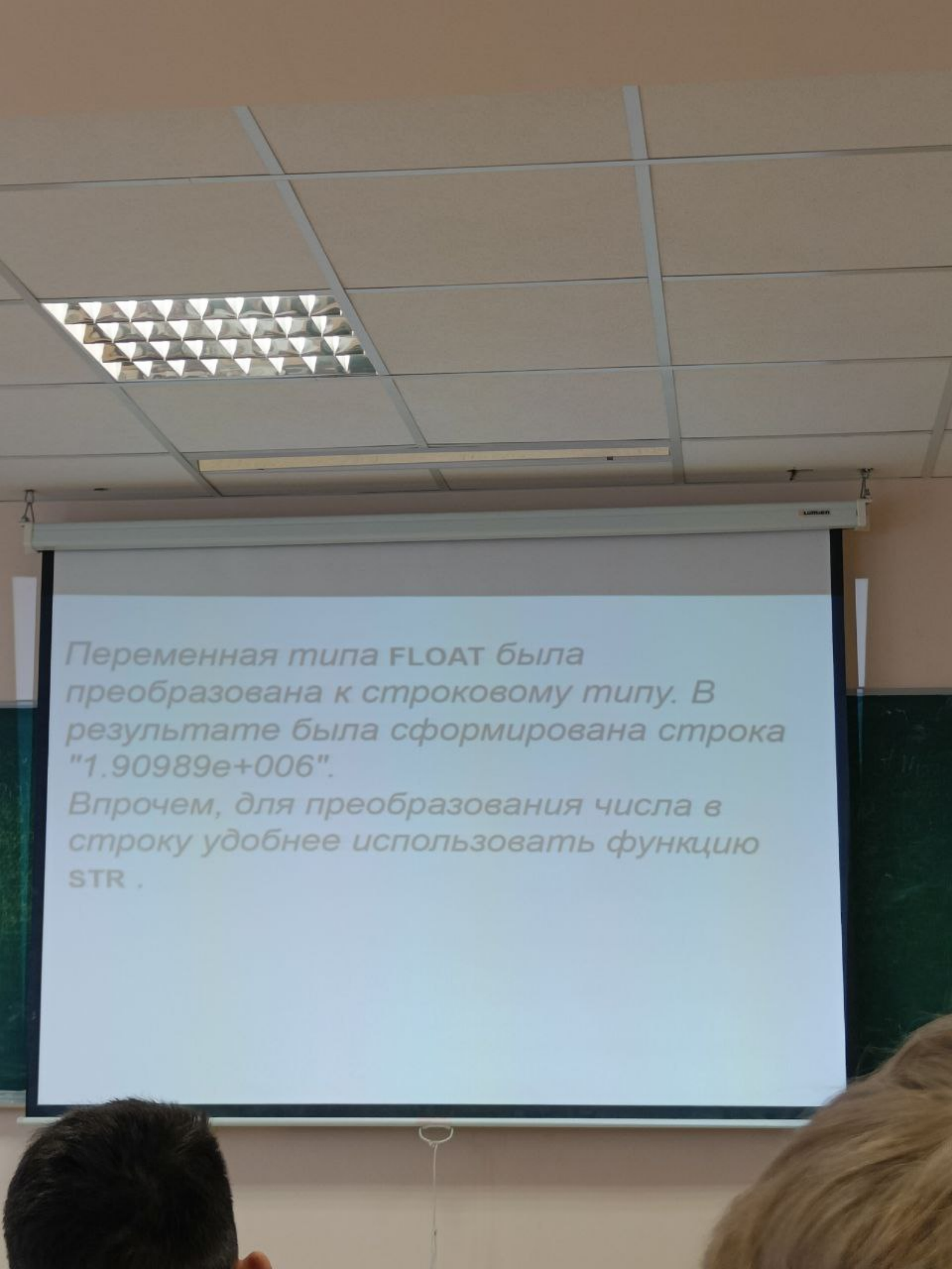

Присвоение с помощью команды
SELECT
DECLARE @a BIGINT,@name CHAR(20),
@s CHAR(20)
SELECT @name='Справка'

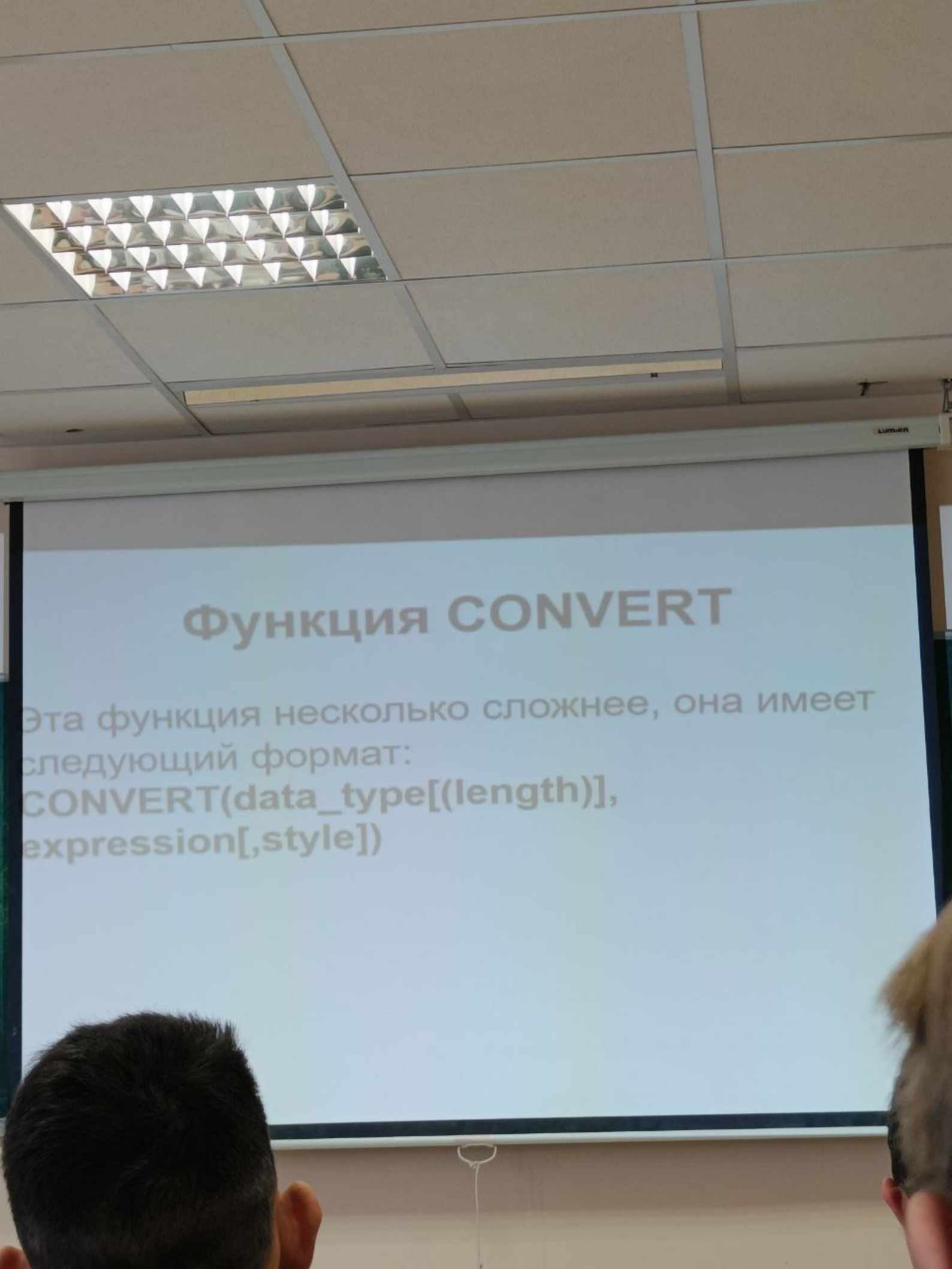

—переменной присваивается количество строк таблицы ut_students
SELECT @a=COUNT(*) FROM ut_students

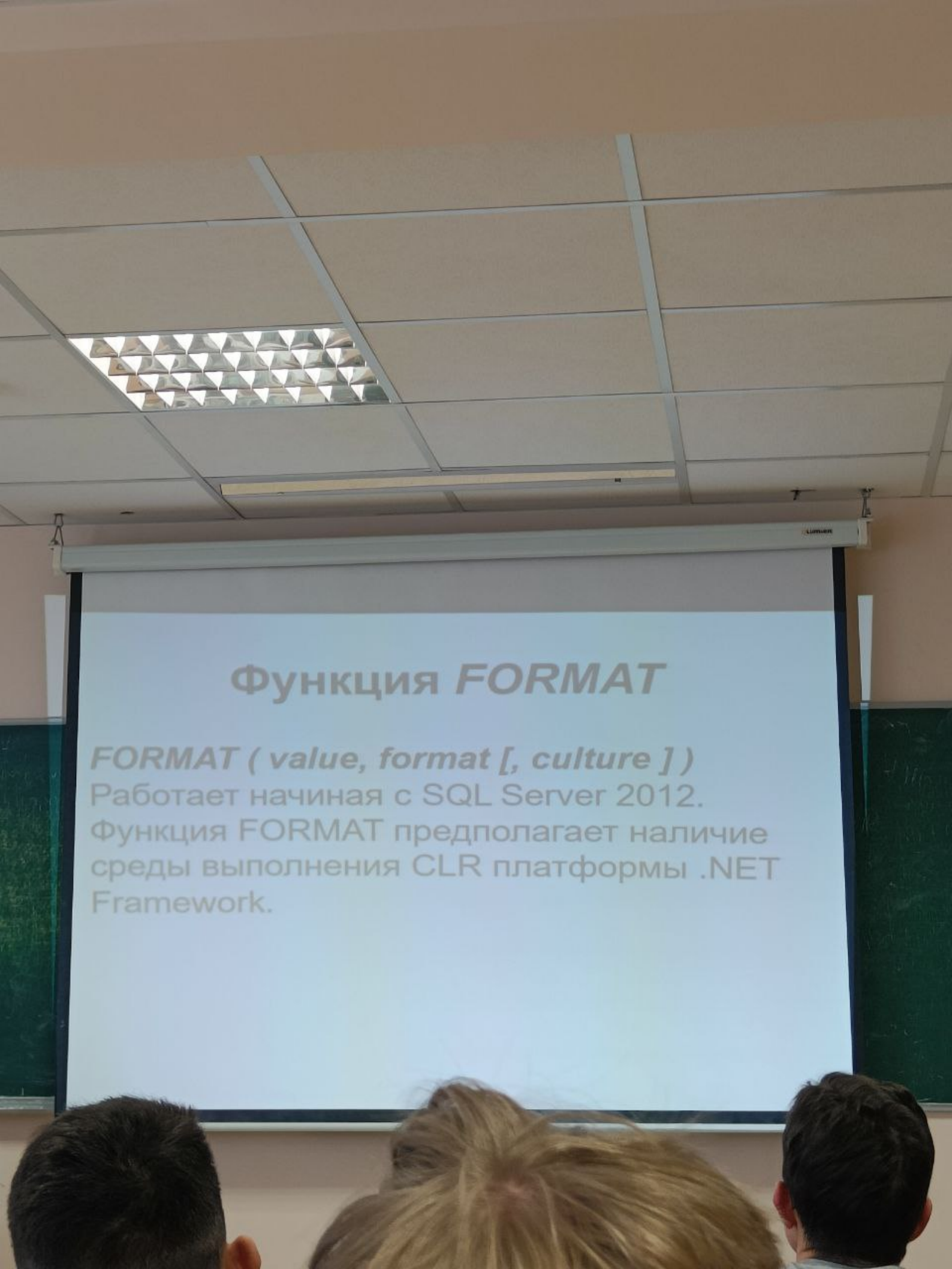

—присвоение переменной значения полей
"family" и "name" таблицы ut_students
DECLARE @кто VARCHAR(20)
SELECT @кто=RTRIM(family)+'
'+RTRIM(name) FROM ut_students WHERE
NumberZach=55301013

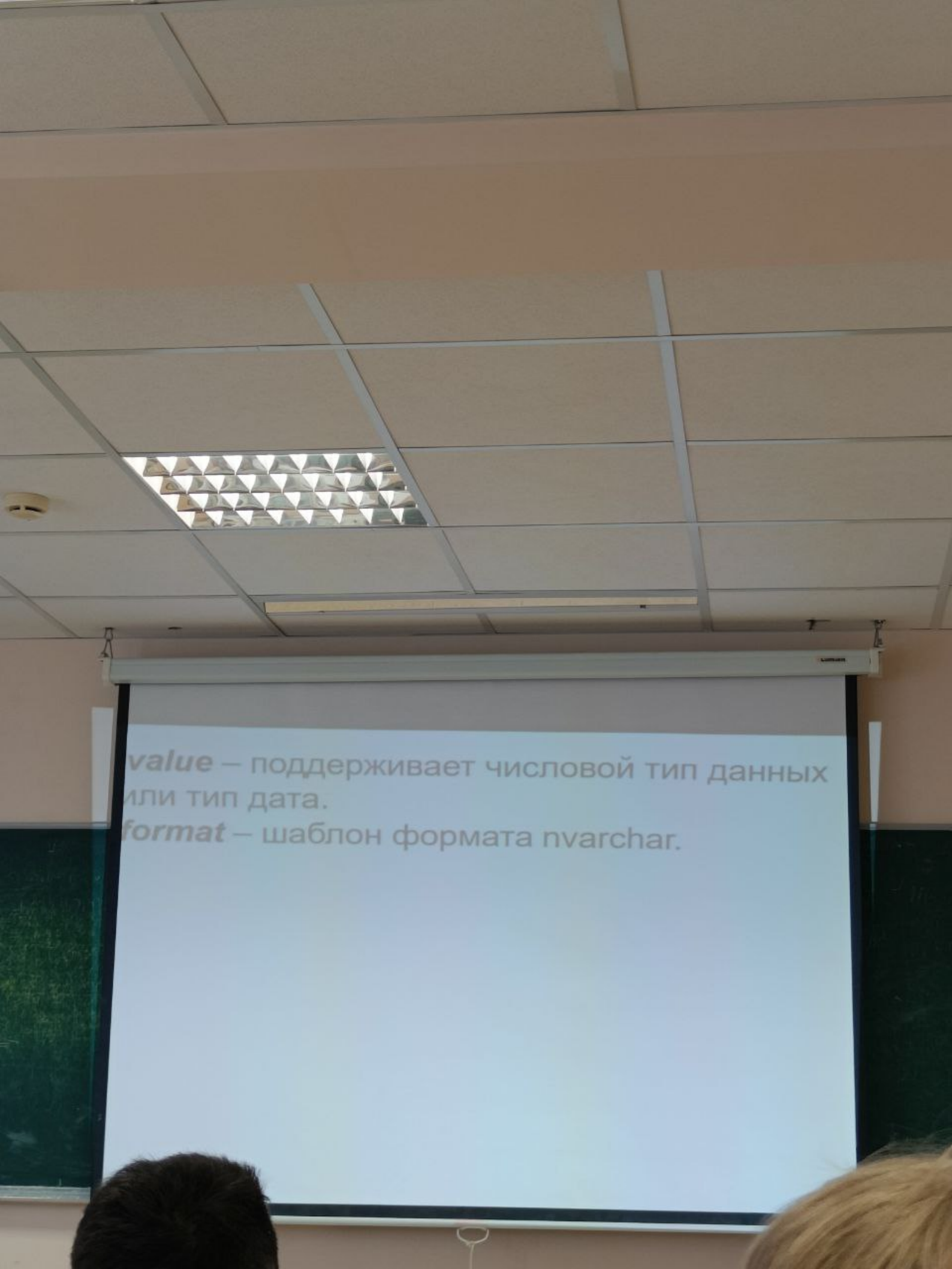

Присвоение значения переменной
Сочетание ключевого слова SET и запроса
DECLARE @ sum FLOAT
переменной присваивается сумма всех
значений поля "opl" таблицы salary
SET @sum=(SELECT SUM(opl) FROM salary)

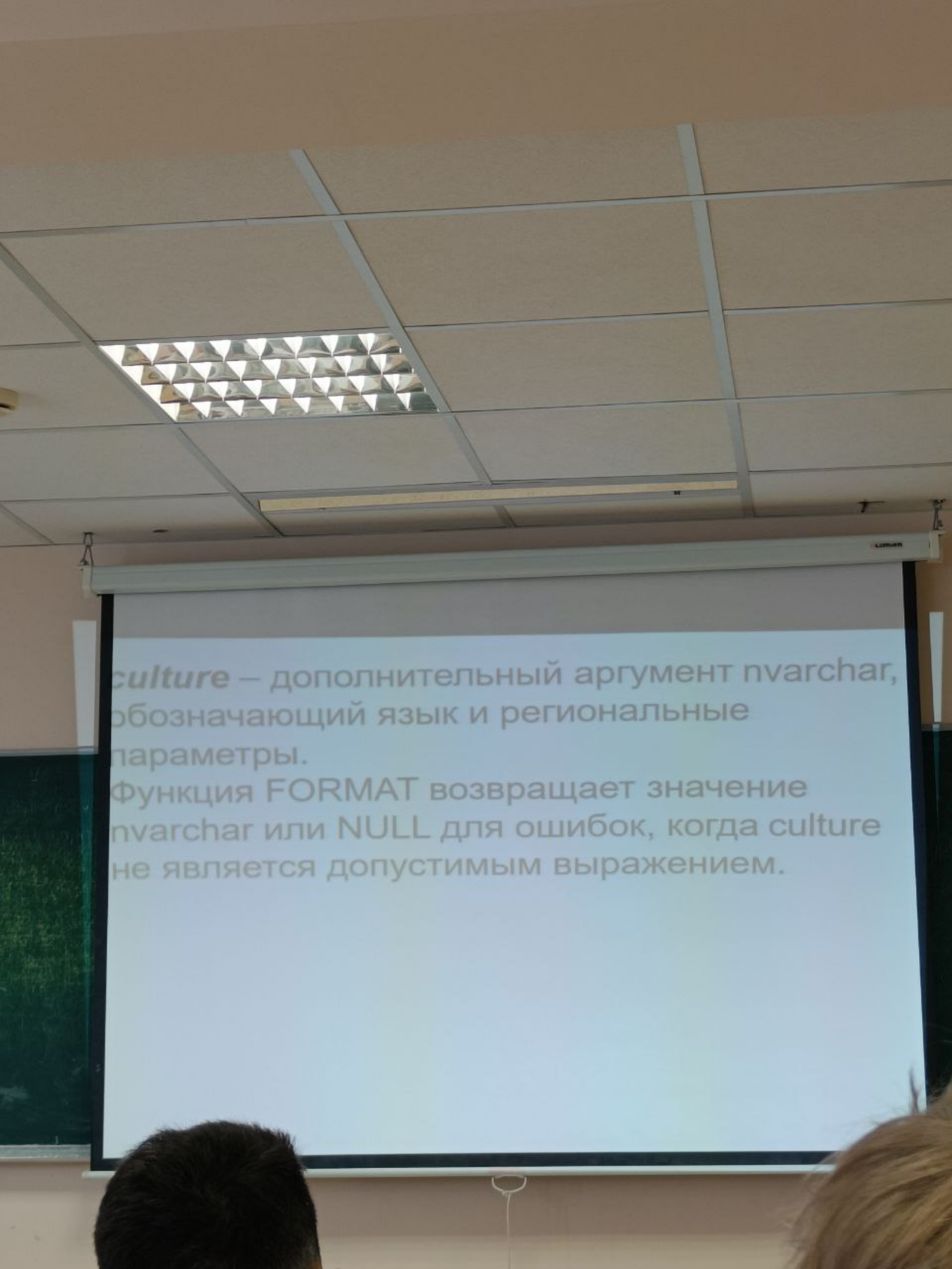

Функция CAST
CAST(expression AS data_type)
Пример.
DECLARE @a FLOAT
DECLARE @s CHAR(15)
SET @a=1909888.88
SET @s=CAST(@a AS CHAR(15))
SELECT @s

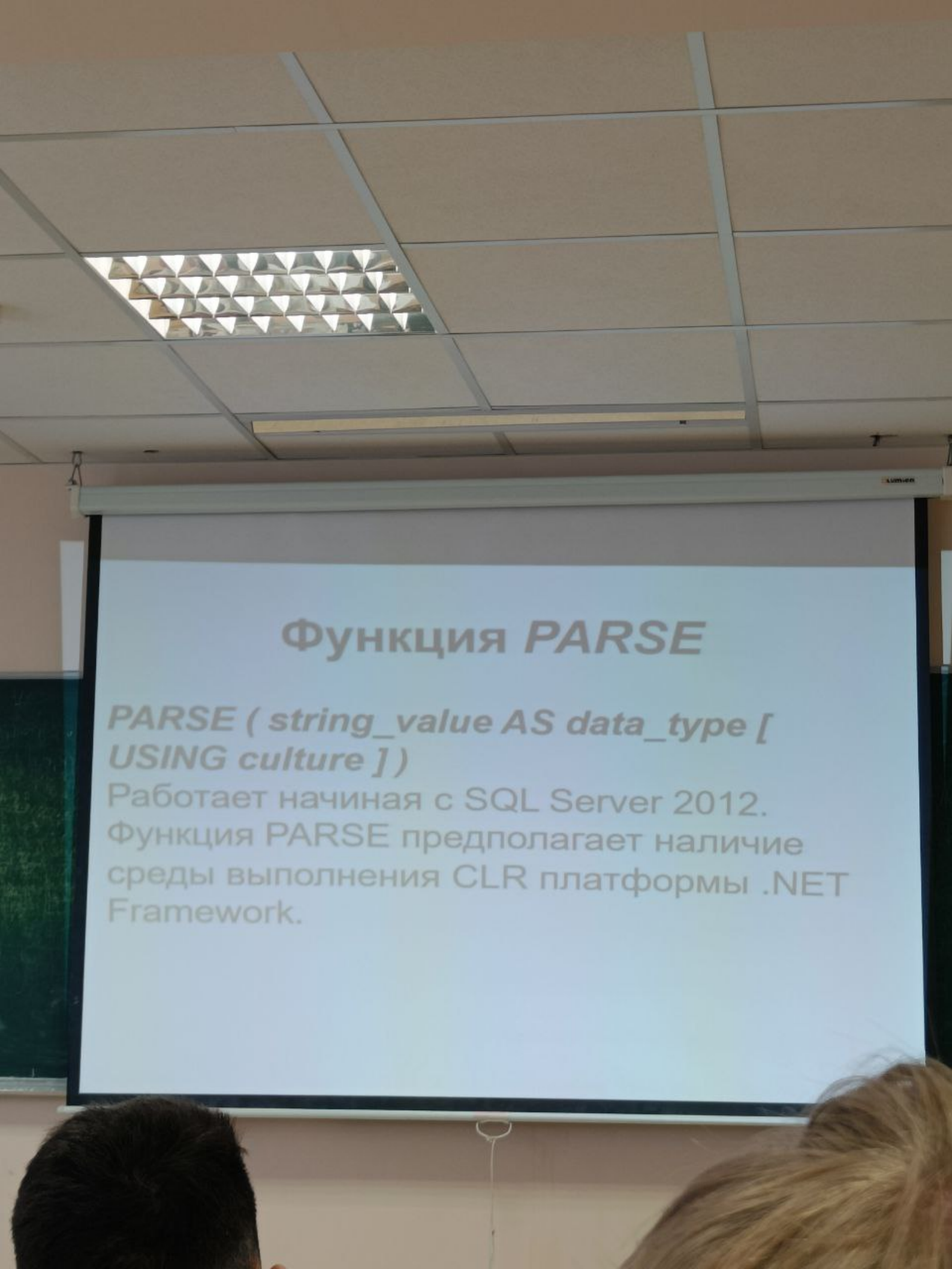

Переменная типа FLOAT была преобразована к строковому типу. В результате была сформирована строка "1.90989e+006".
Впрочем, для преобразования числа в строку удобнее использовать функцию STR .

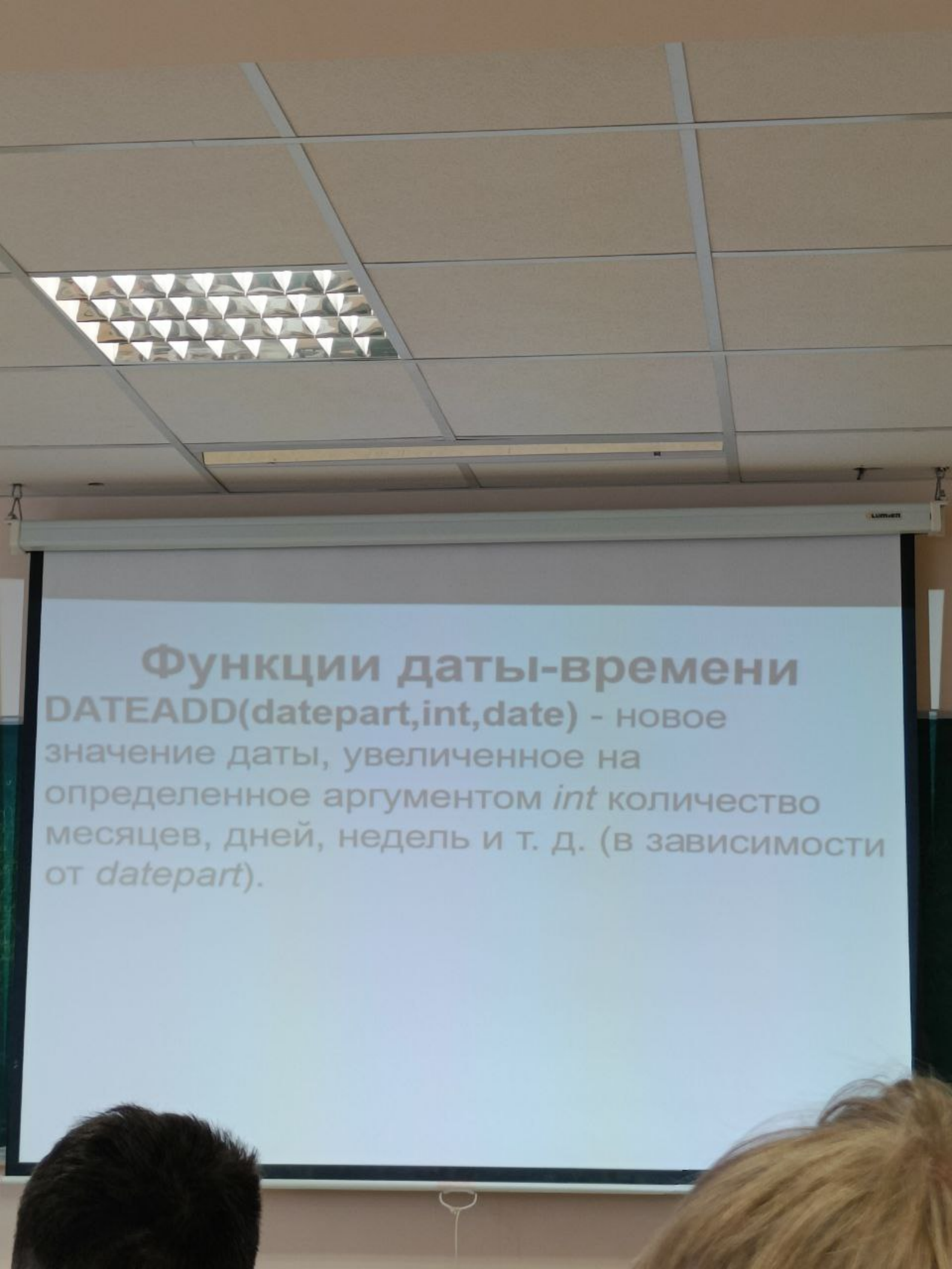

Функция CONVERT
Эта функция несколько сложнее, она имеет следующий формат:
CONVERT(data_type[(length)], expression[,style])

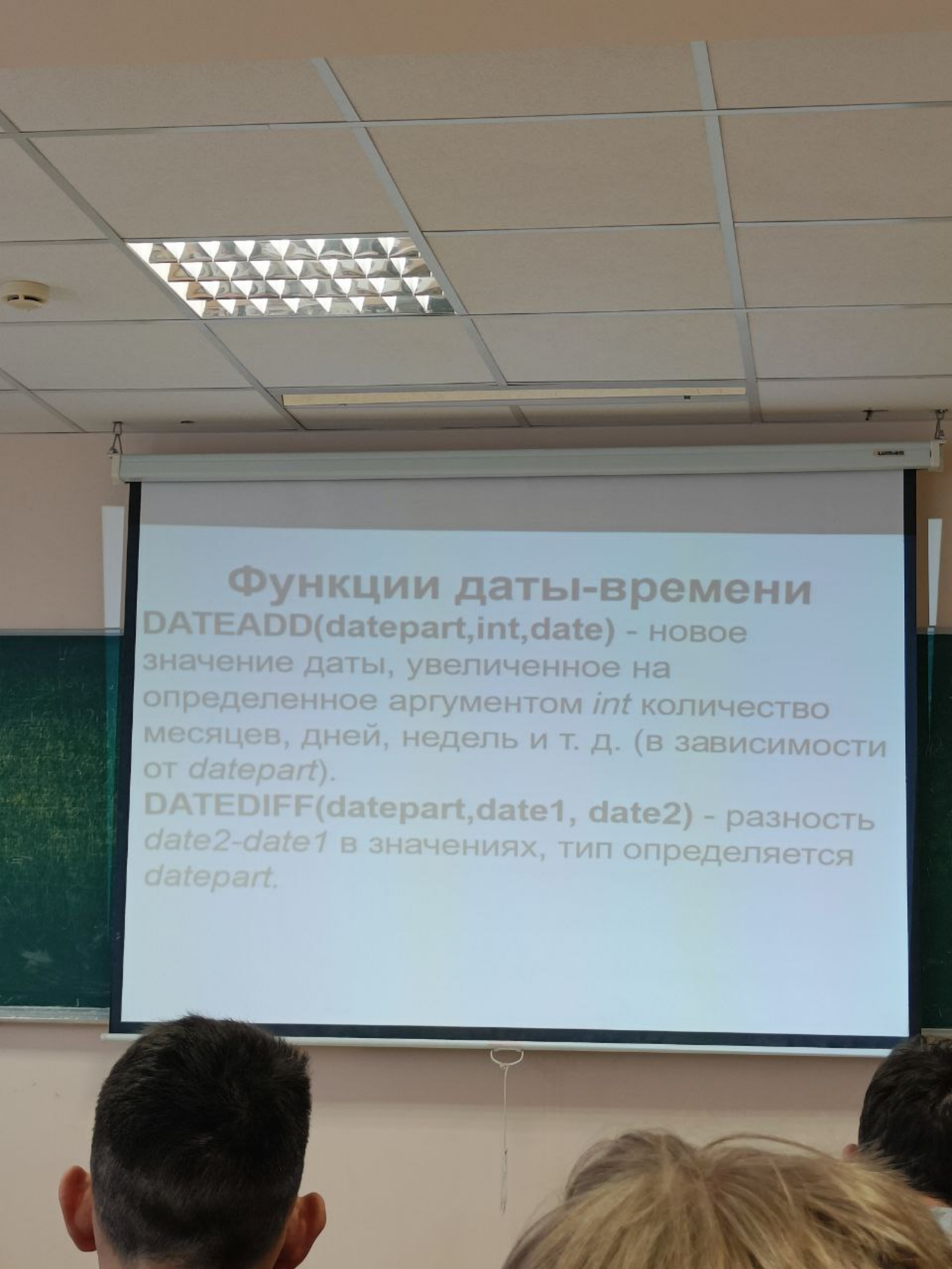

Функция FORMAT
FORMAT (value, format [, culture])
Работает начиная с SQL Server 2012.
Функция FORMAT предполагает наличие среды выполнения CLR платформы .NET Framework.

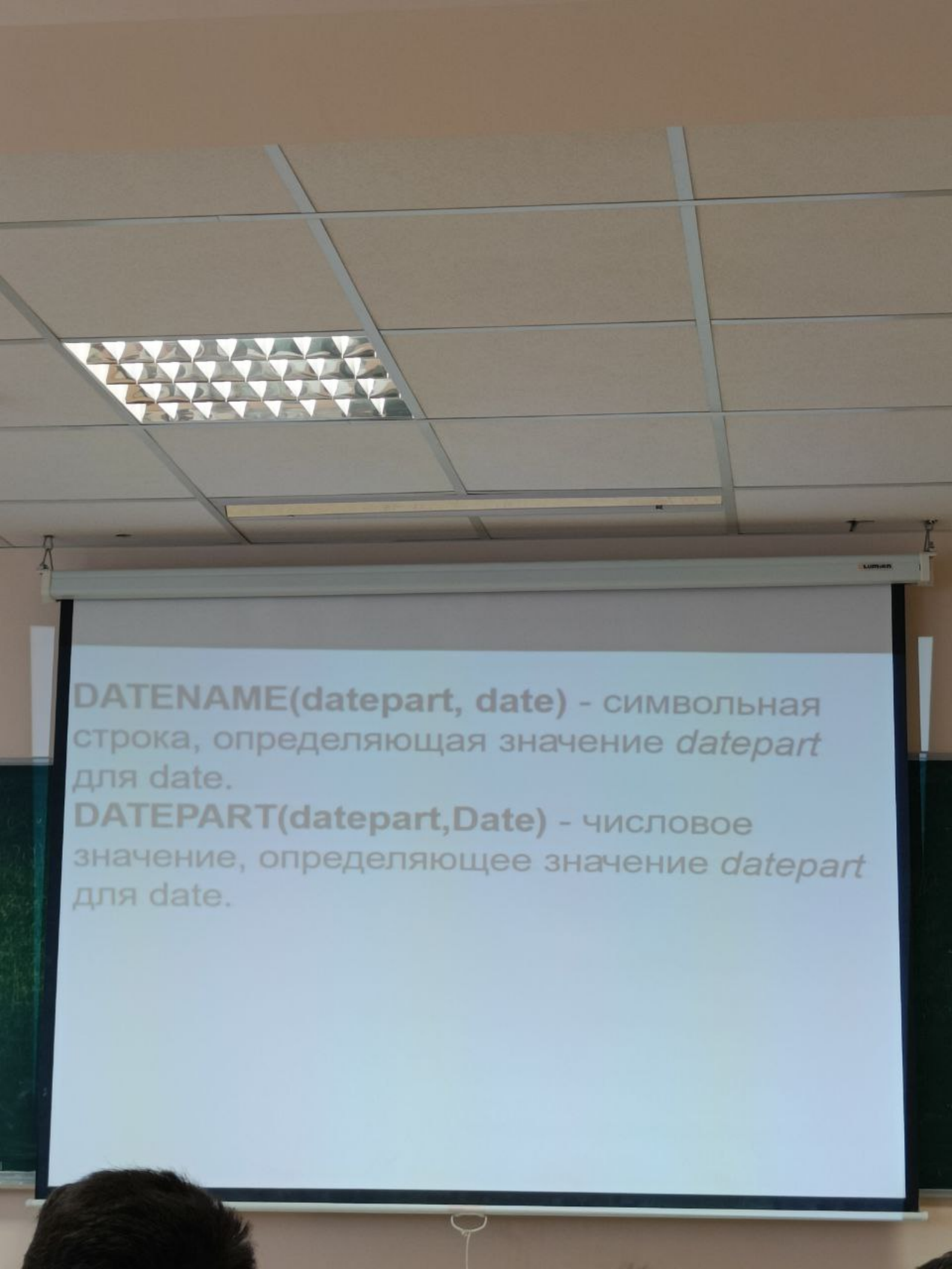

value – поддерживает числовой тип данных
или тип дата.
format – шаблон формата nvarchar.

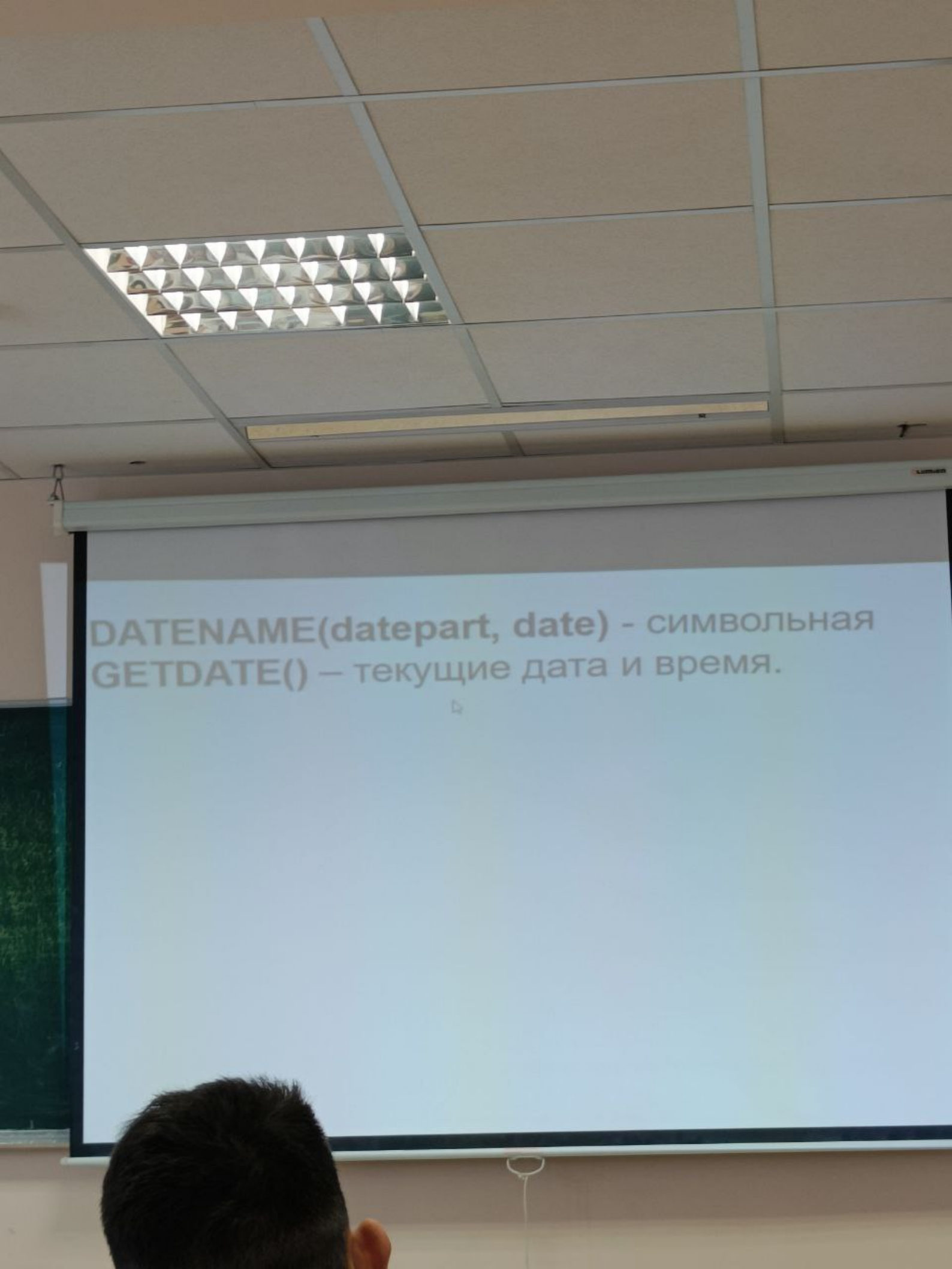

culture – дополнительный аргумент nvarchar,
обозначающий язык и региональные
параметры.
Функция FORMAT возвращает значение
nvarchar или NULL для ошибок, когда culture
не является допустимым выражением.


Функция PARSE
PARSE (string_value AS data_type [USING culture])
Работает начиная с SQL Server 2012.
Функция PARSE предполагает наличие среды выполнения CLR платформы .NET Framework.


Функции даты-времени
DATEADD(datepart,int,date) - новое значение даты, увеличенное на определенное аргументом int количество месяцев, дней, недель и т. д. (в зависимости от datepart).


Функции даты-времени
DATEADD(datepart,int,date) - новое значение даты, увеличенное на определенное аргументом int количество месяцев, дней, недель и т. д. (в зависимости от datepart).
DATEDIFF(datepart,date1, date2) - разность date2-date1 в значениях, тип определяется datepart.


DATENAME(datepart, date) - символьная строка, определяющая значение *datepart* для date.
DATEPART(datepart,Date) - числовое значение, определяющее значение *datepart* для date.


DATENAME(datepart, date) - символьная
GETDATE() – текущие дата и время.

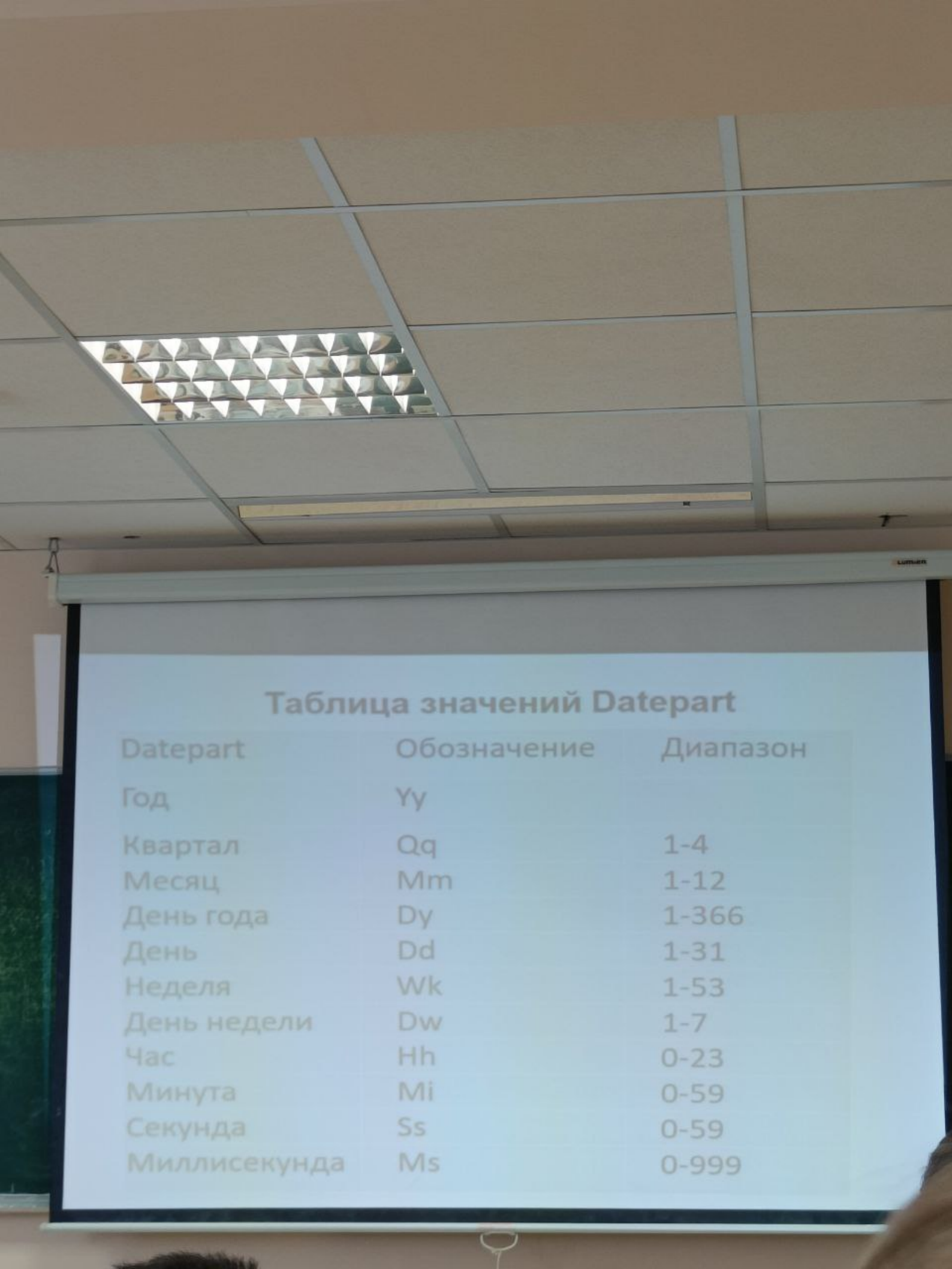
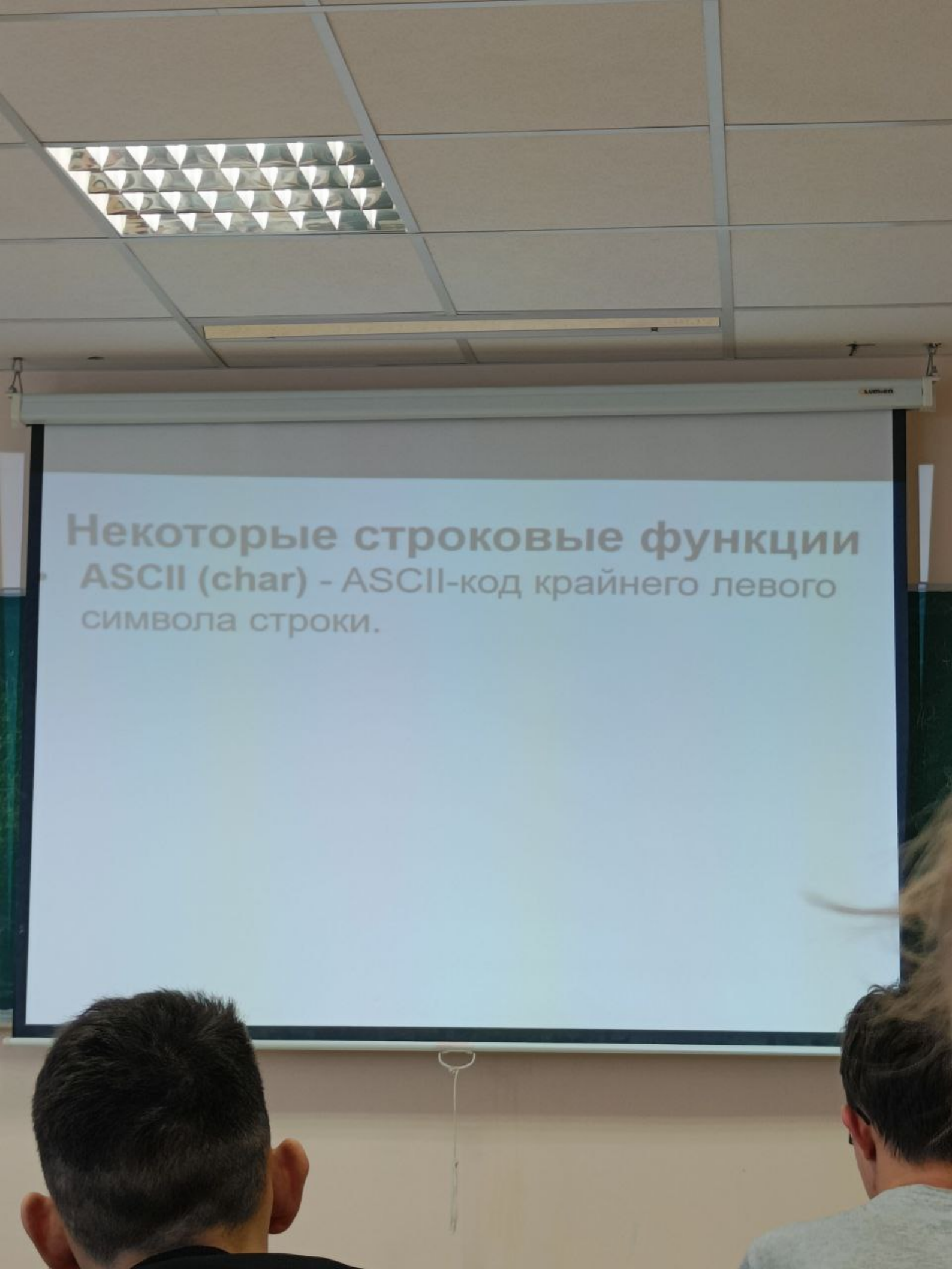
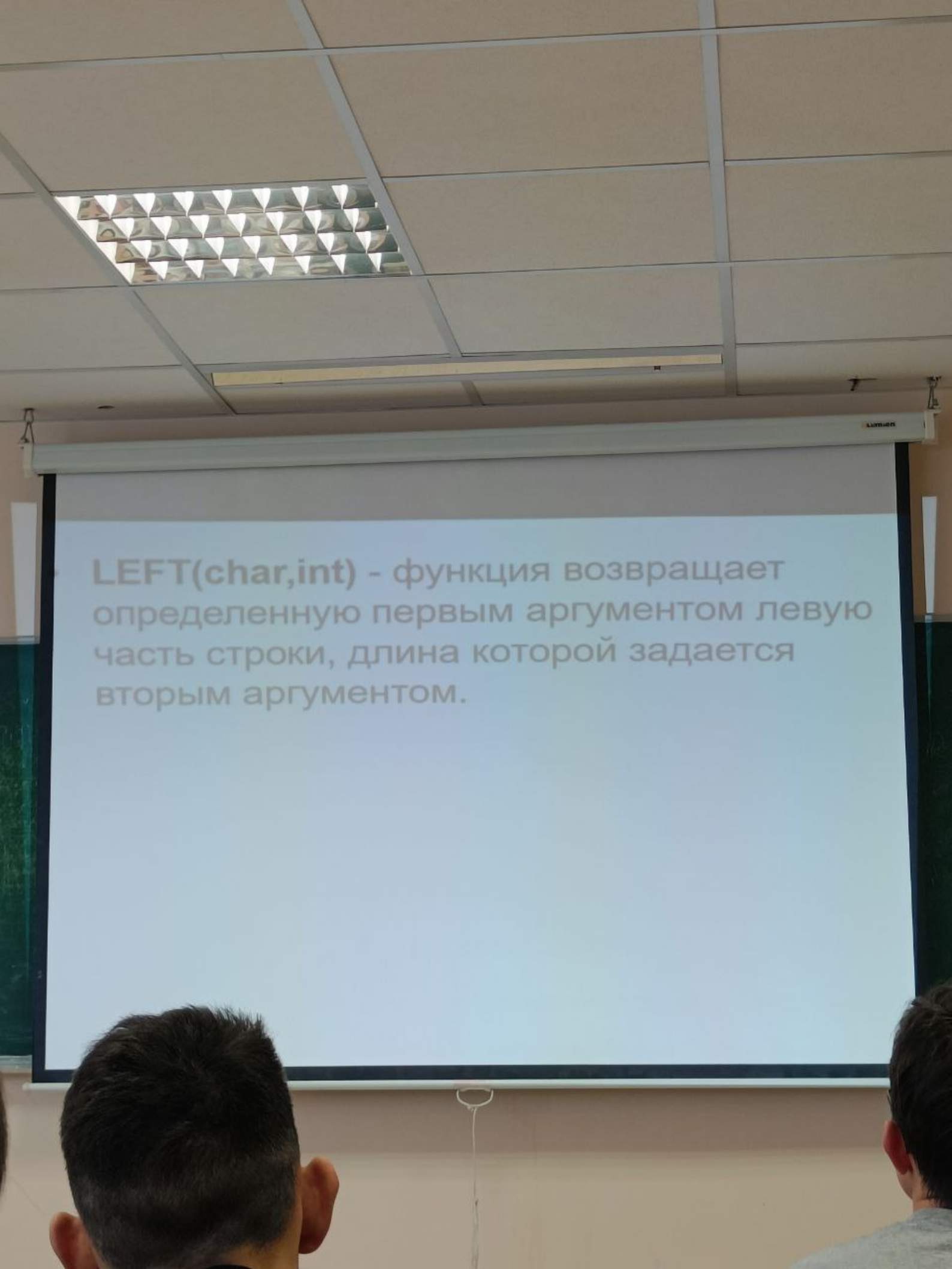
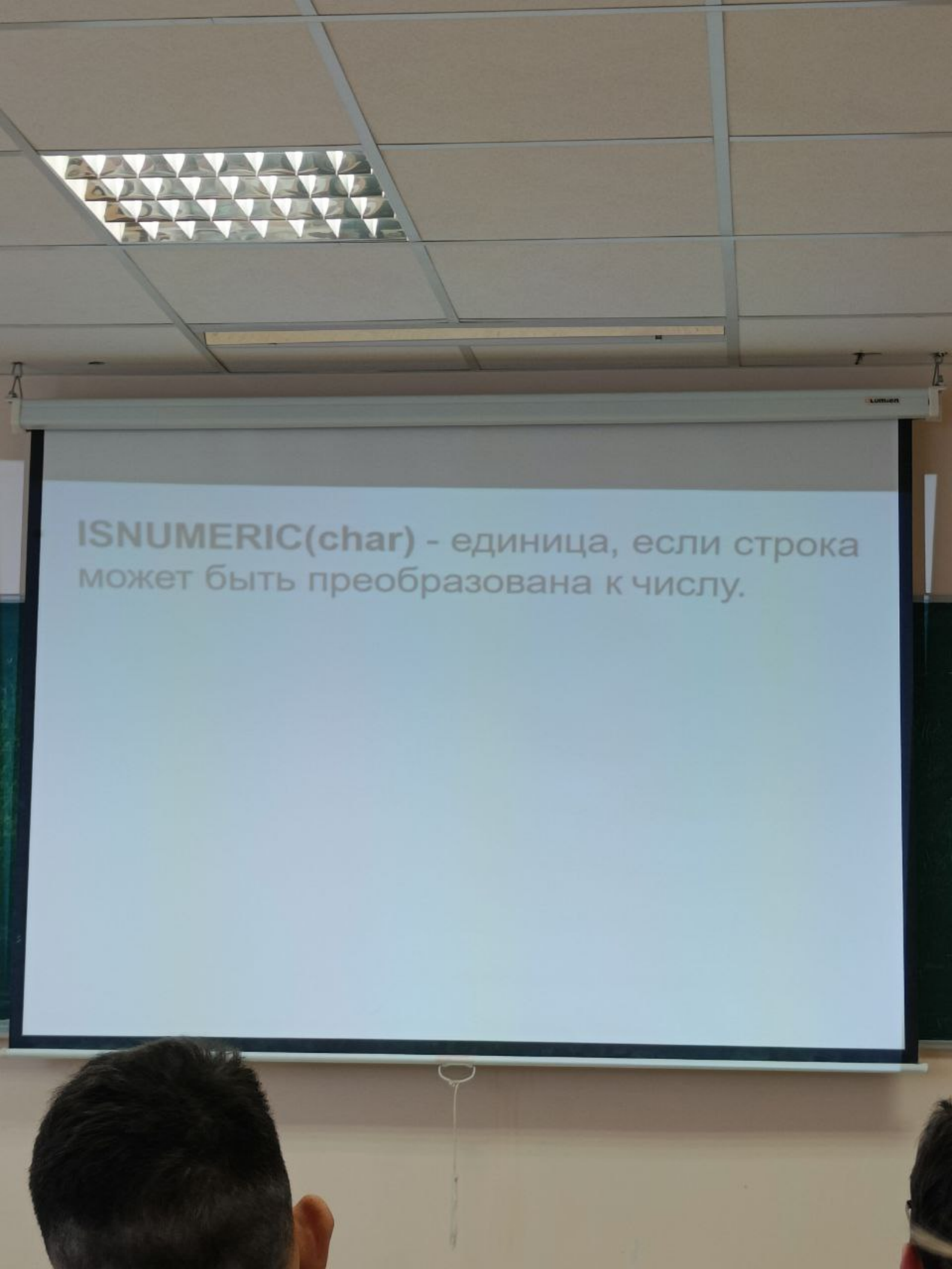
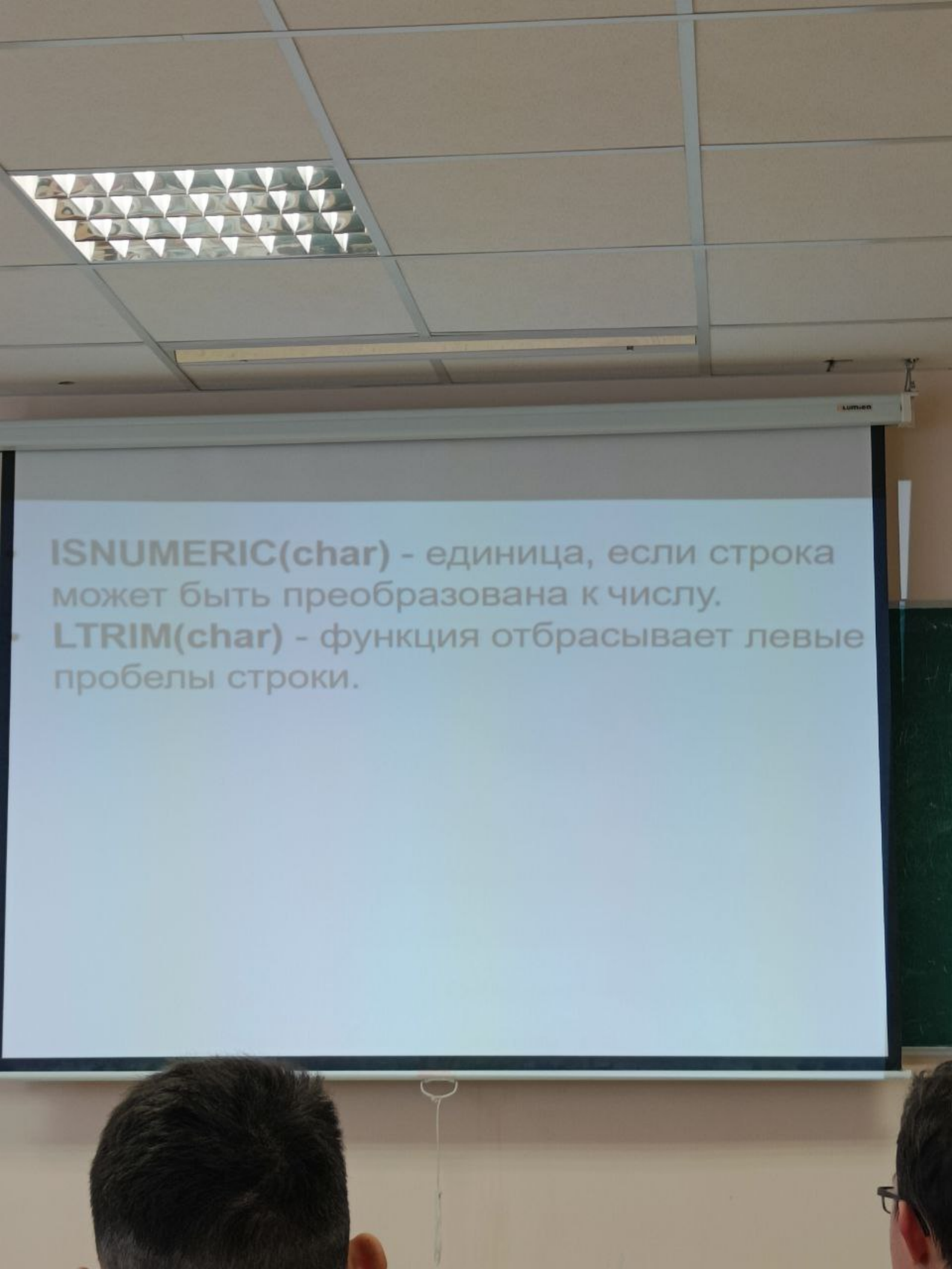


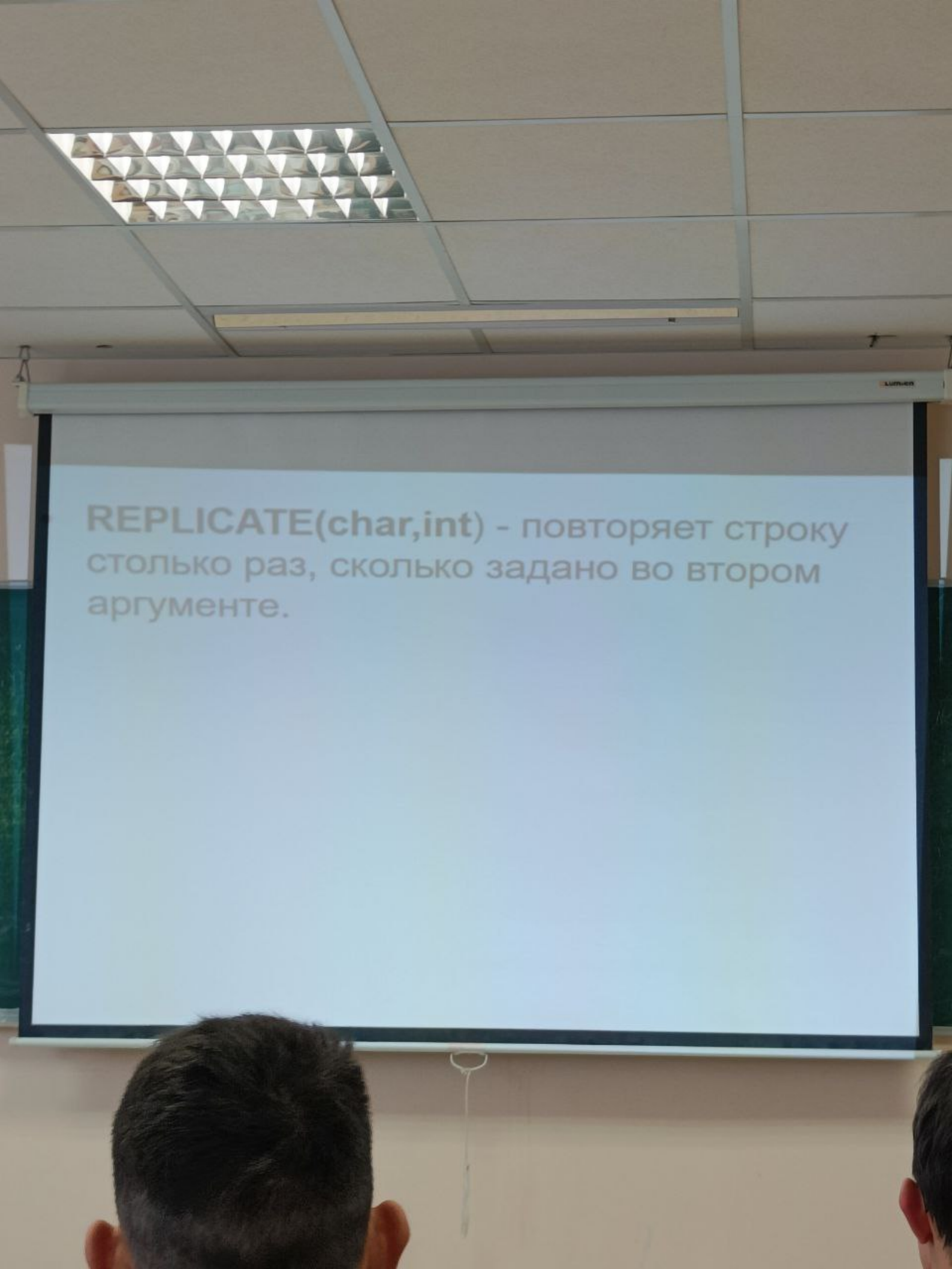
Таблица значений Datepart
Datepart Обозначение Диапазон
Год Yy
Квартал Qq 1-4
Месяц Mm 1-12
День года Dy 1-366
День Dd 1-31
Неделя Wk 1-53
День недели Dw 1-7
Час Hh 0-23
Минута Mi 0-59
Секунда Ss 0-59
Миллисекунда Ms 0-999

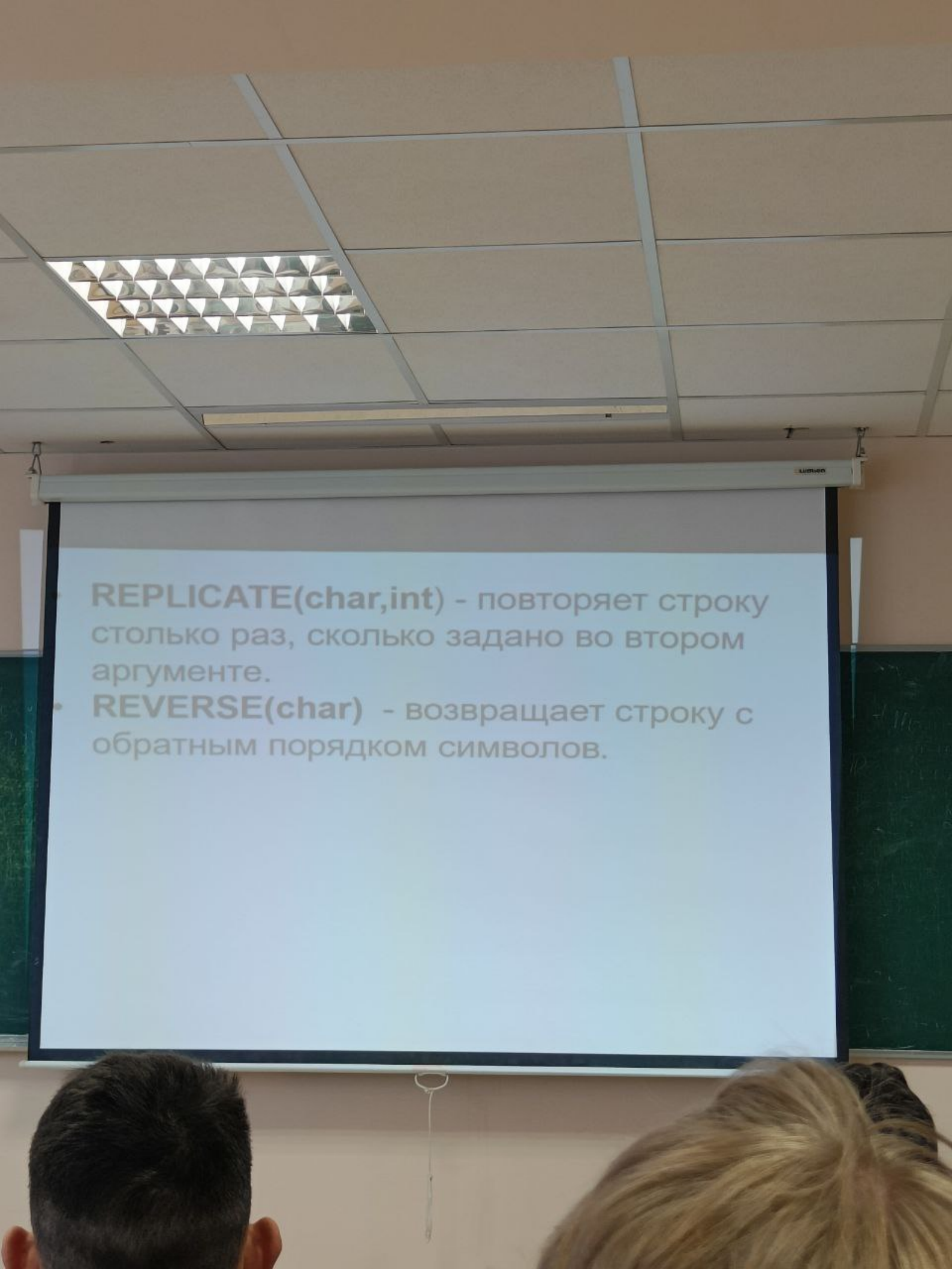

Некоторые строковые функции
ASCII (char) - ASCII-код крайнего левого символа строки.

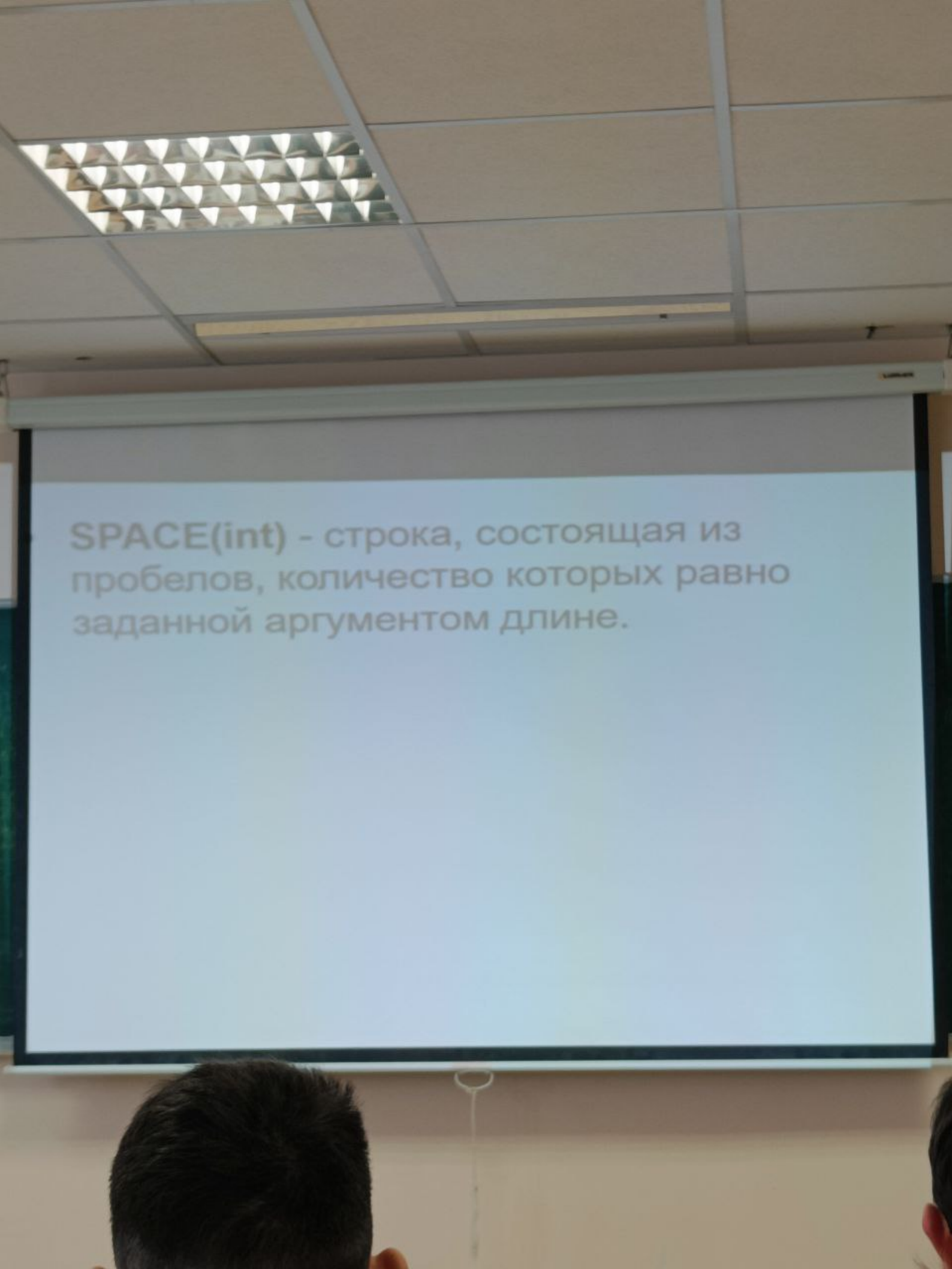

LEFT(char,int) - функция возвращает определенную первым аргументом левую часть строки, длина которой задается вторым аргументом.

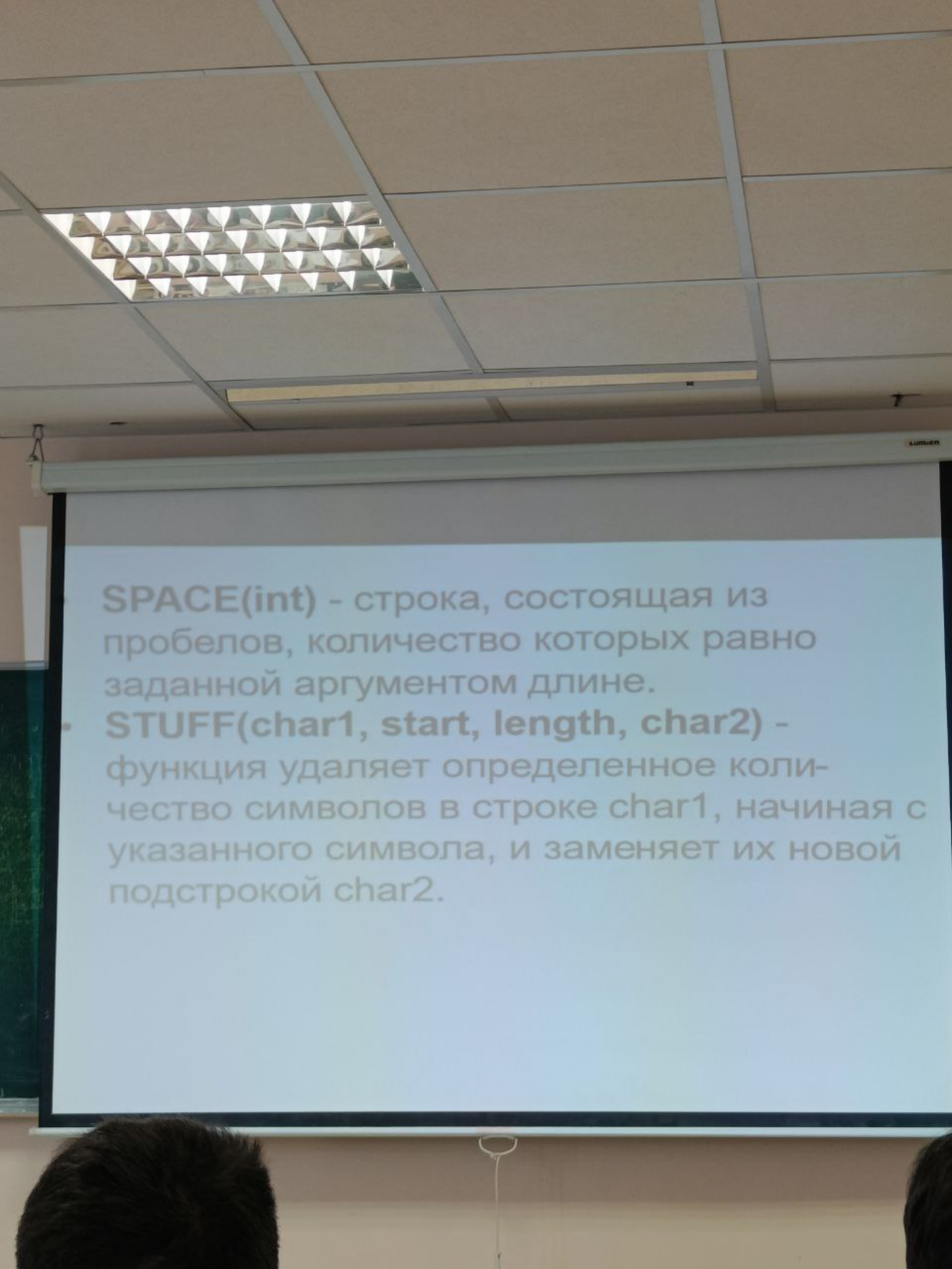

ISNUMERIC(char) - единица, если строка может быть преобразована к числу.

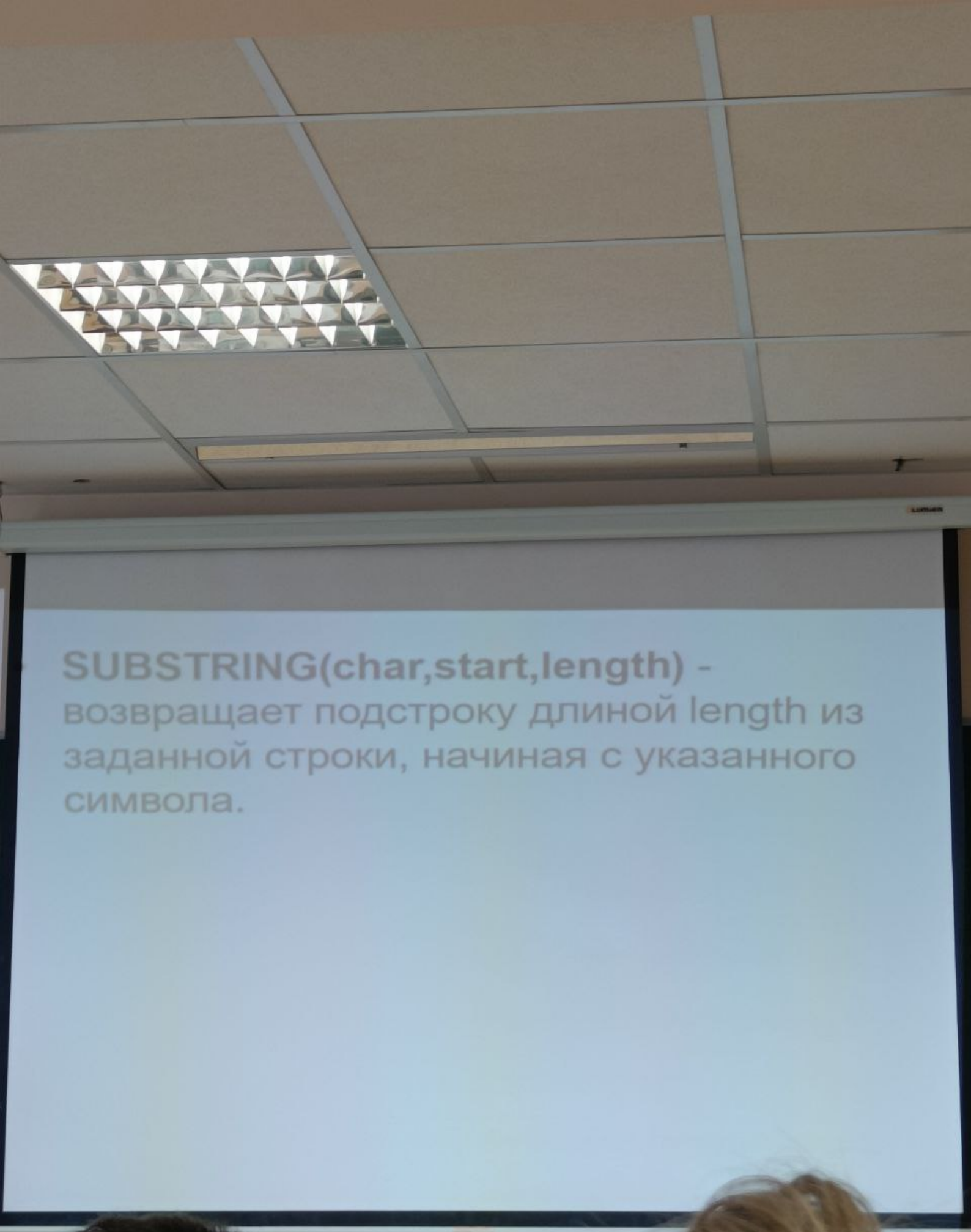

ISNUMERIC(char) - единица, если строка может быть преобразована к числу.
LTRIM(char) - функция отбрасывает левые пробелы строки.

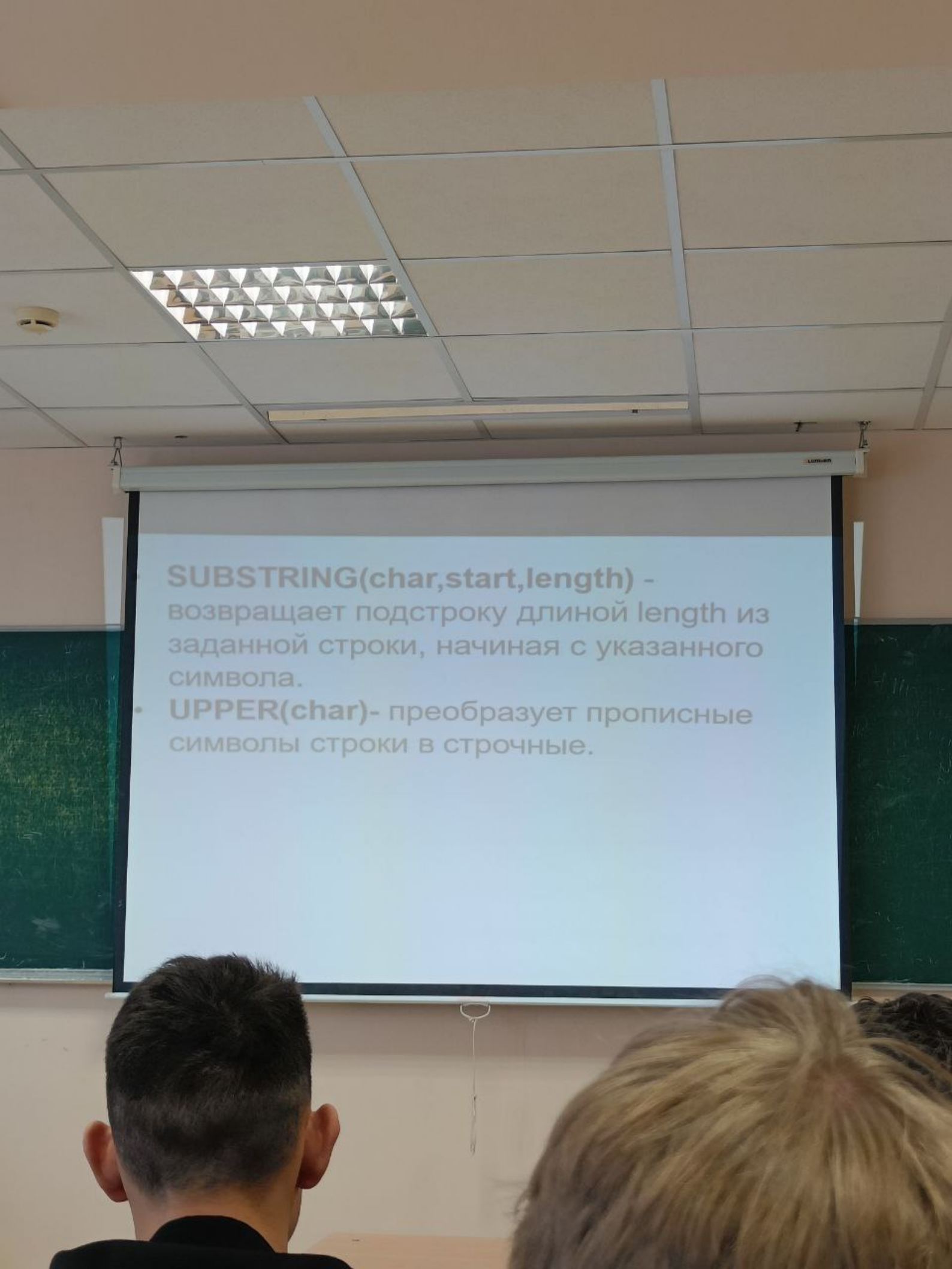

REPLICATE(char,int) - повторяет строку столько раз, сколько задано во втором аргументе.

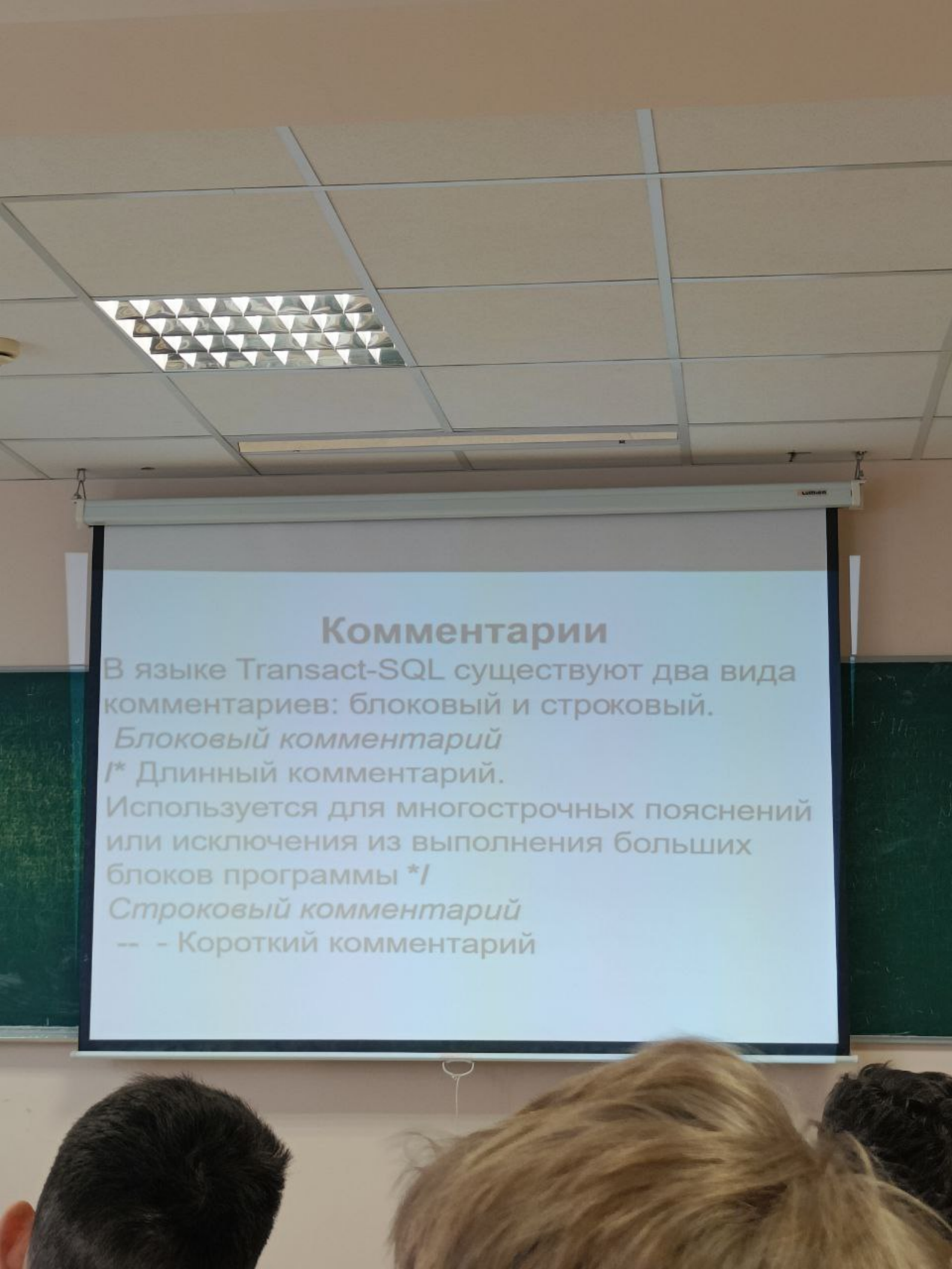

REPLICATE(char,int) - повторяет строку столько раз, сколько задано во втором аргументе.
REVERSE(char) - возвращает строку с обратным порядком символов.

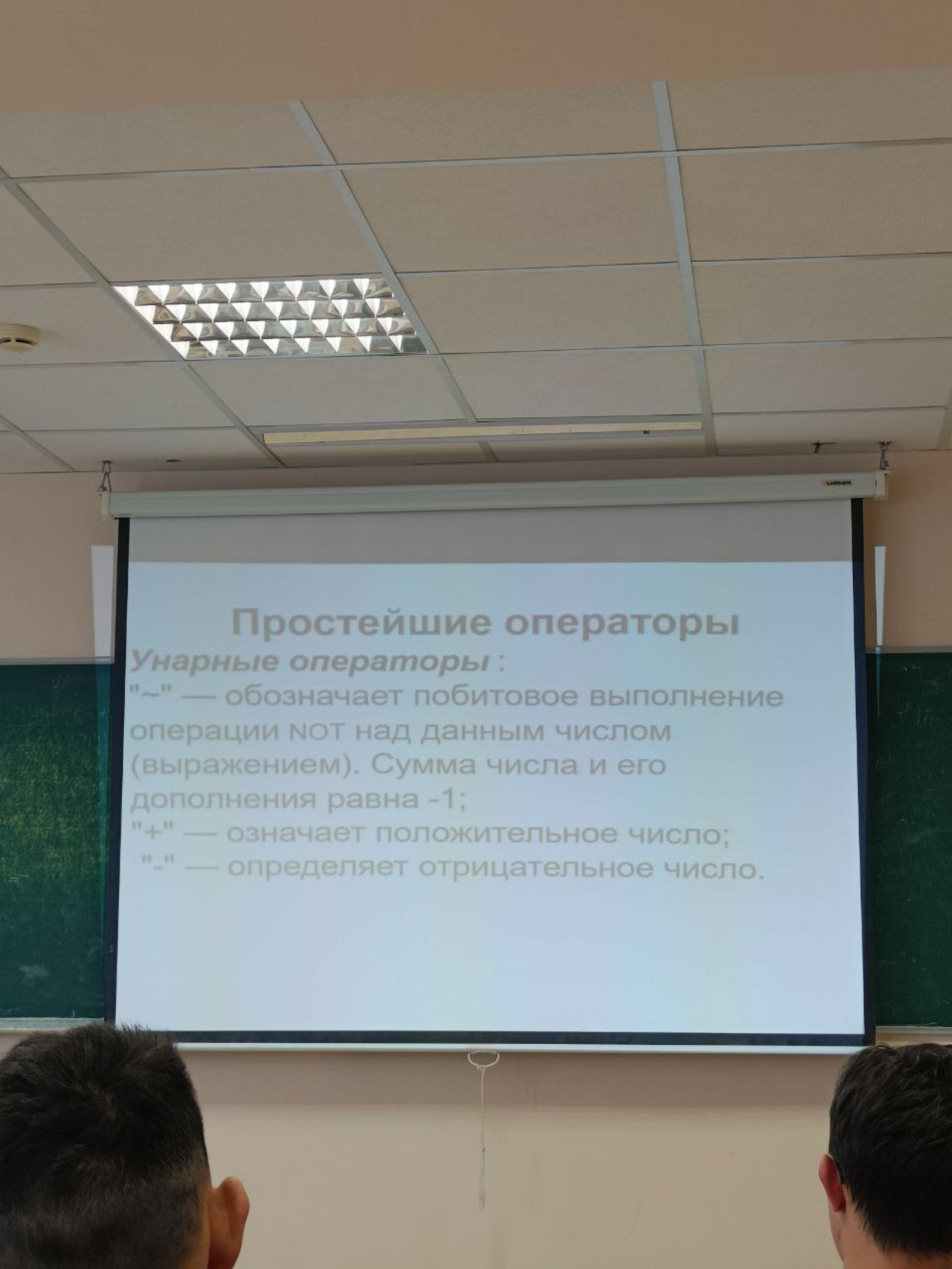

SPACE(int) - строка, состоящая из пробелов, количество которых равно заданной аргументом длине.

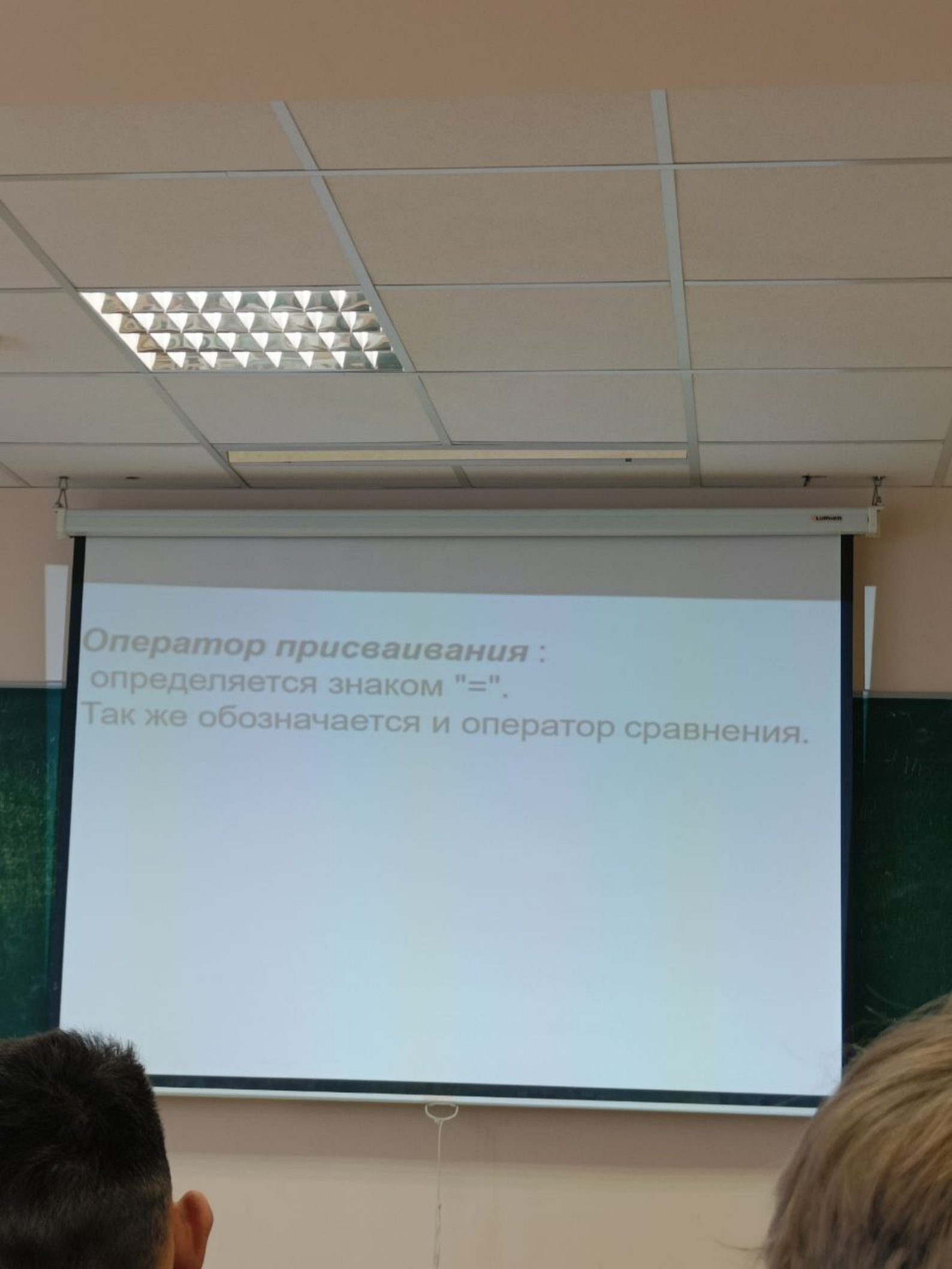

SPACE(int) - строка, состоящая из пробелов, количество которых равно заданной аргументом длине.
STUFF(char1, start, length, char2) - функция удаляет определенное количество символов в строке char1, начиная с указанного символа, и заменяет их новой подстрокой char2.

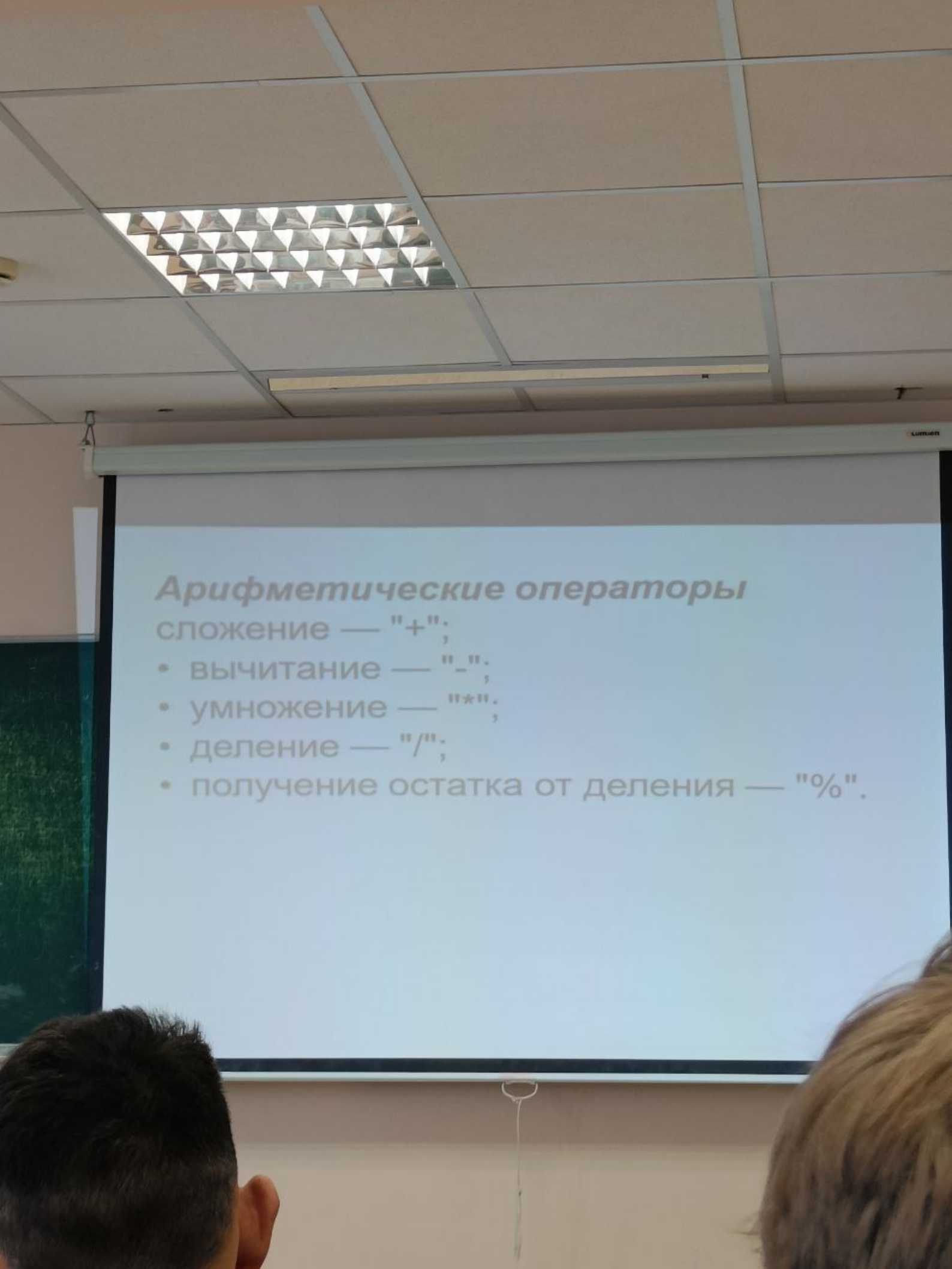

SUBSTRING(char,start,length) -
возвращает подстроку длиной length из заданной строки, начиная с указанного символа.

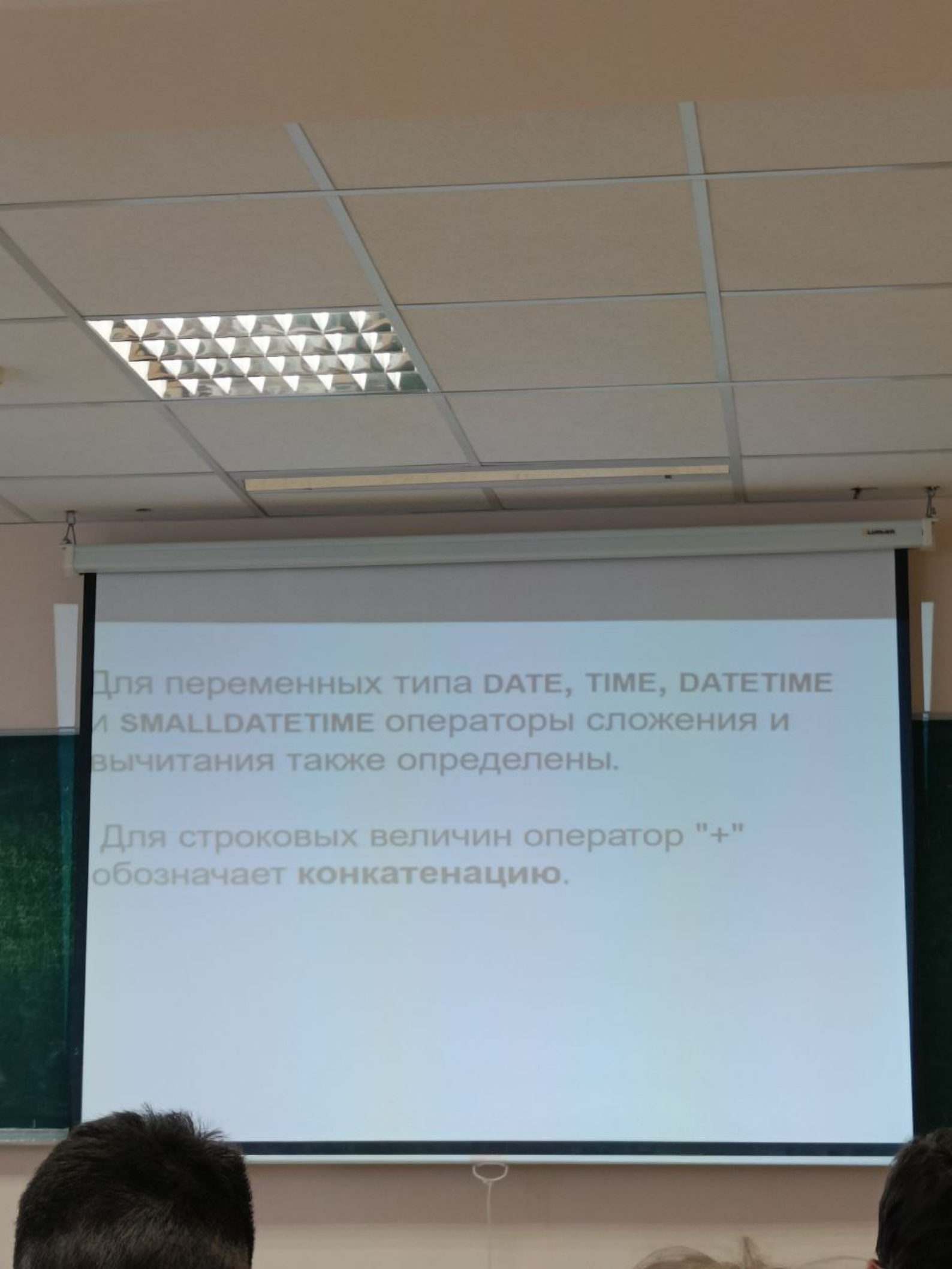

SUBSTRING(char,start,length) - возвращает подстроку длиной length из заданной строки, начиная с указанного символа.
UPPER(char)- преобразует прописные символы строки в строчные.

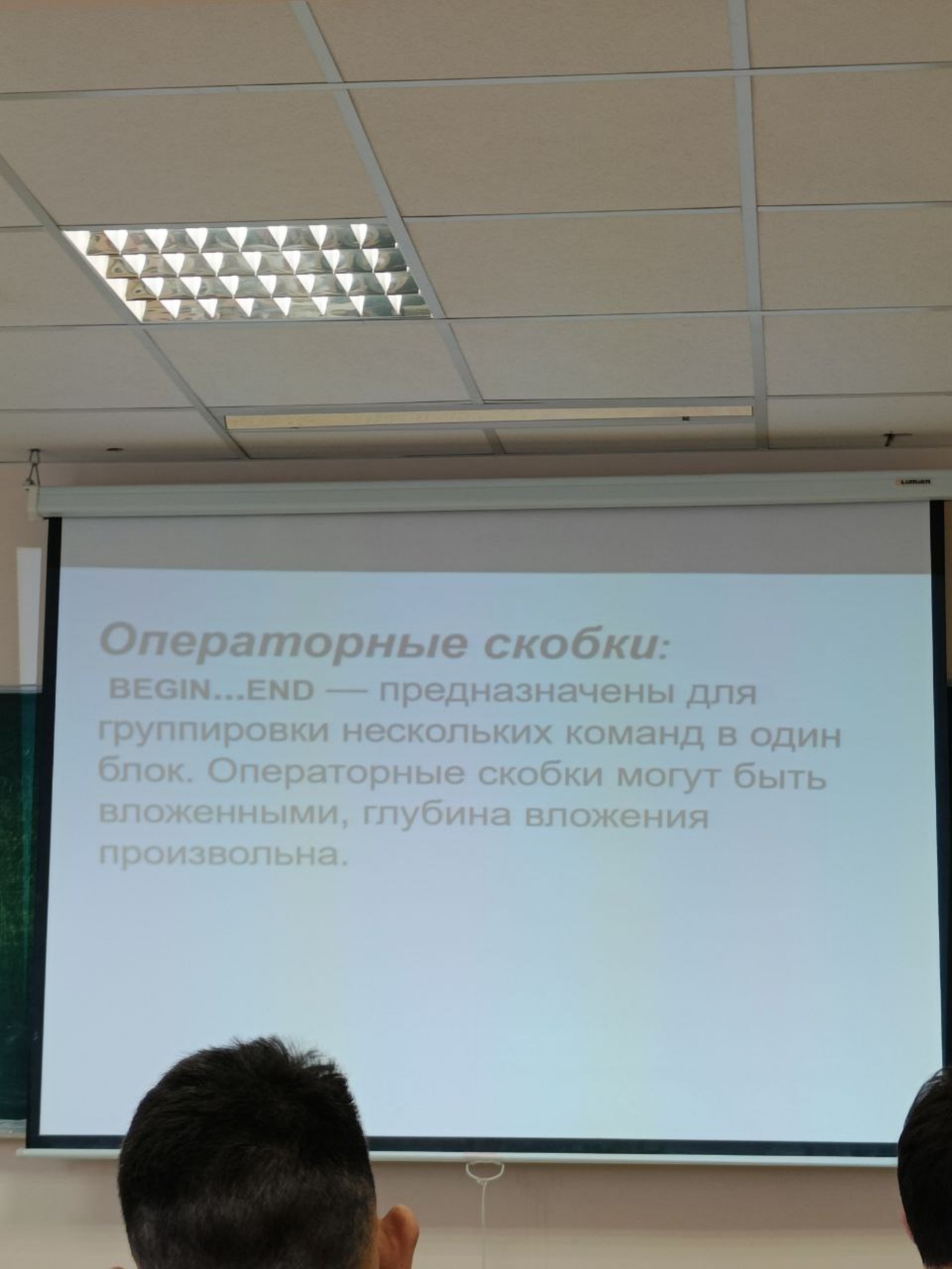

Комментарии
В языке Transact-SQL существуют два вида комментариев: блоковый и строковый.
Блоковый комментарий
/* Длинный комментарий.
Используется для многострочных пояснений или исключения из выполнения больших блоков программы */
Строковый комментарий
-- - Короткий комментарий

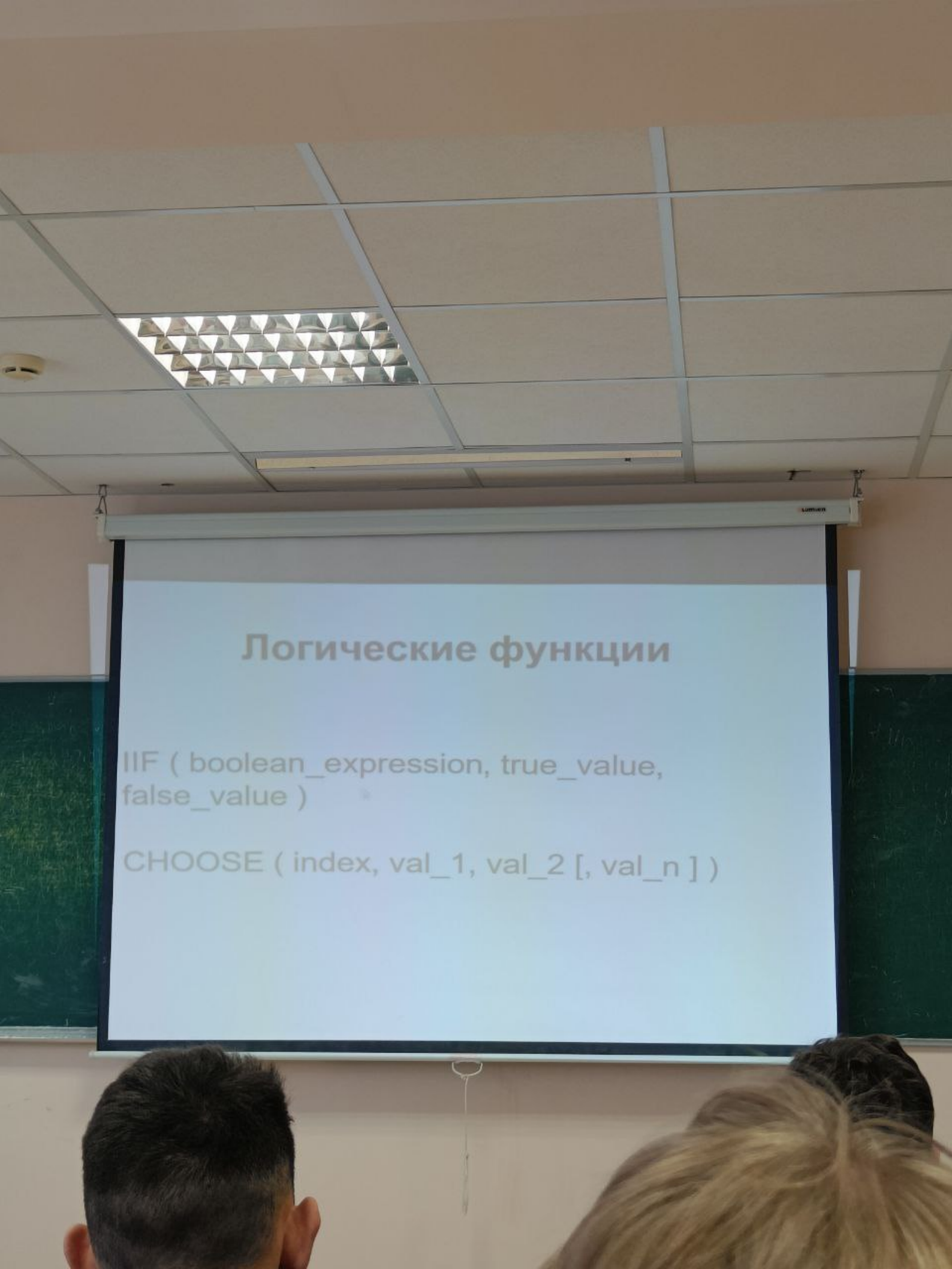

Простейшие операторы
Унарные операторы :
"~" — обозначает побитовое выполнение операции NOT над данным числом (выражением). Сумма числа и его дополнения равна -1;
"+" — означает положительное число;
"-" — определяет отрицательное число.

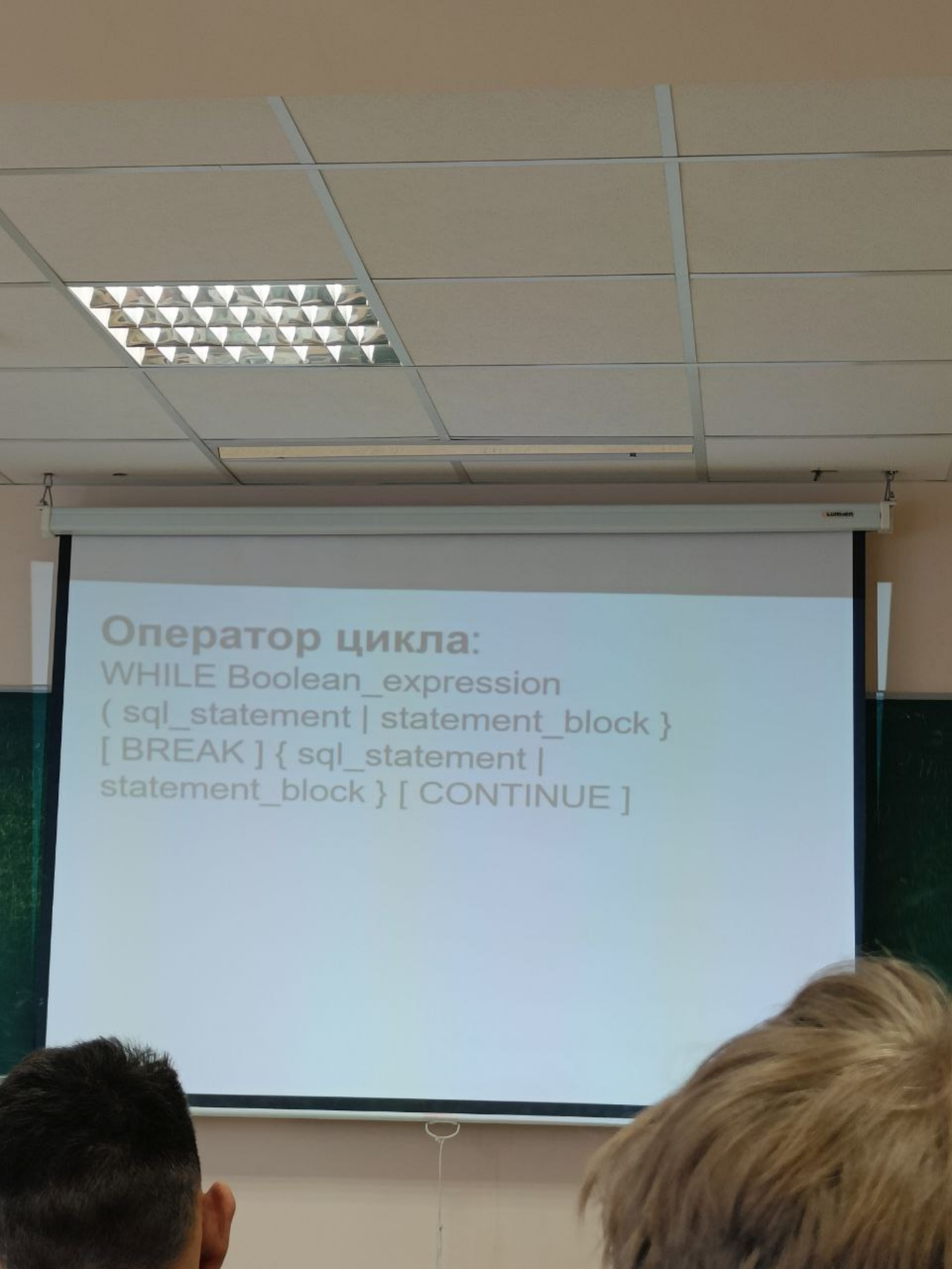

Оператор присваивания :
определяется знаком "=".
Так же обозначается и оператор сравнения.

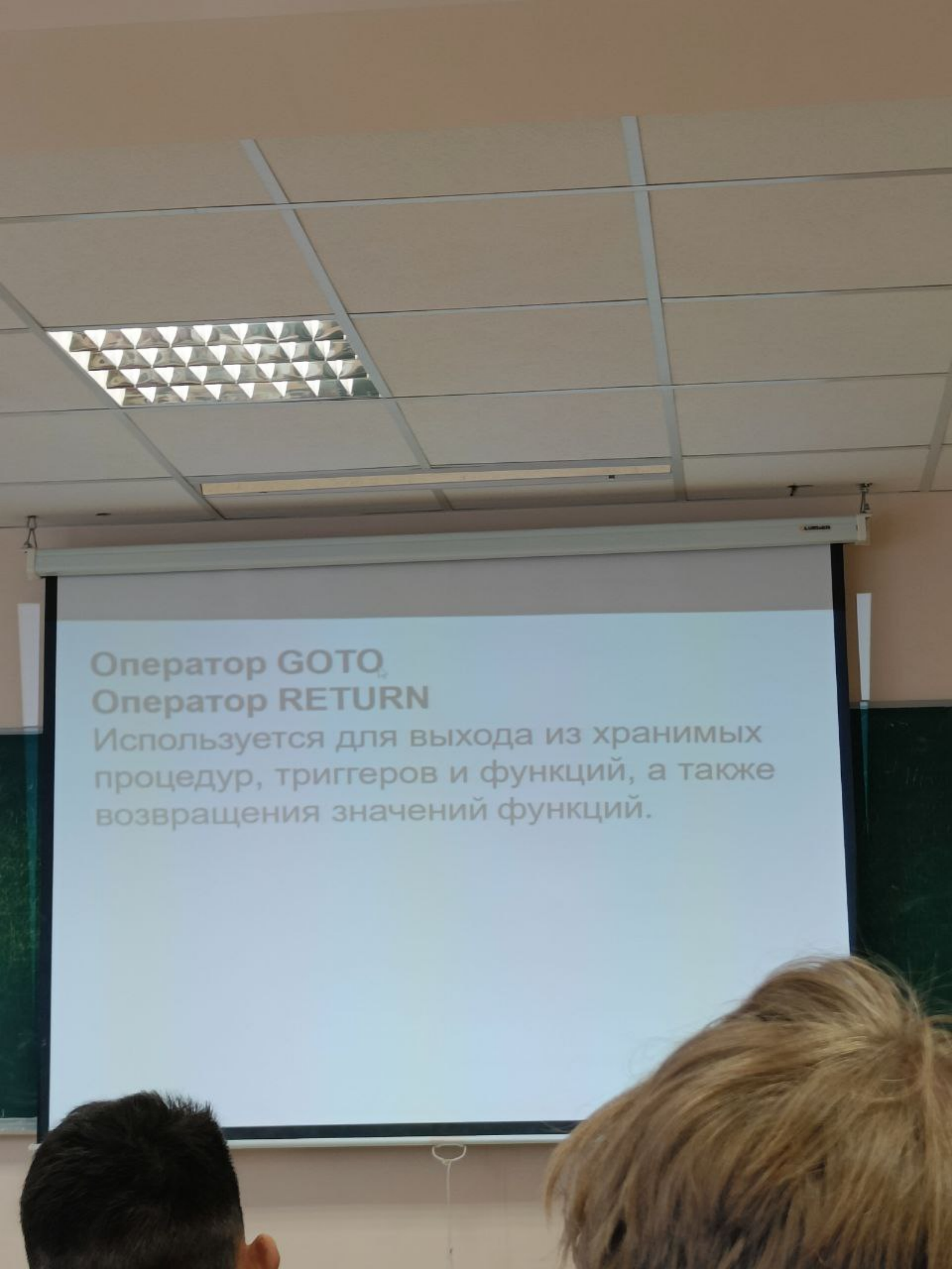

Арифметические операторы
сложение — "+";
• вычитание — "-";
• умножение — "*";
• деление — "/";
• получение остатка от деления — "%".

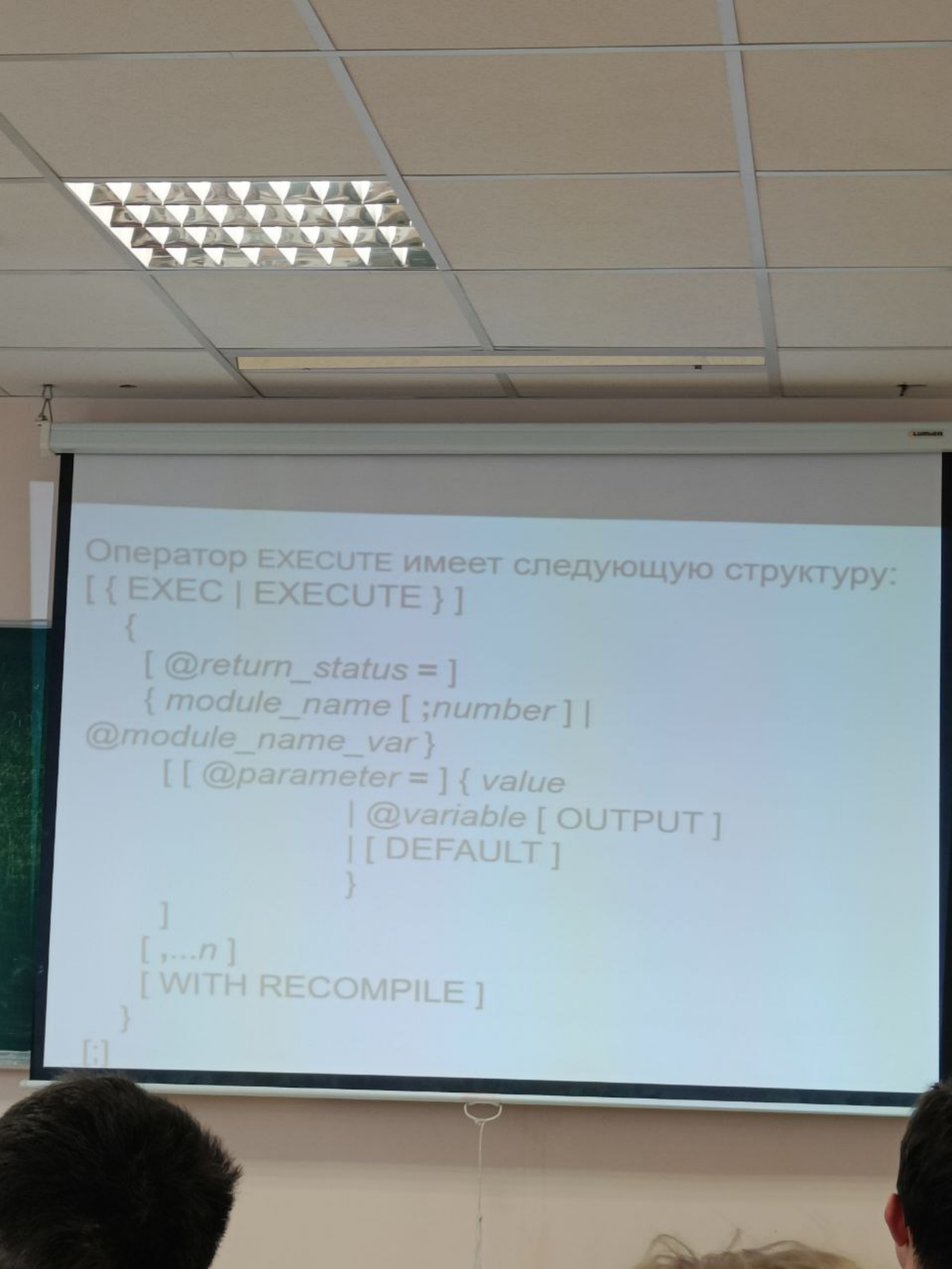

Для переменных типа DATE, TIME, DATETIME
и SMALLDATETIME операторы сложения и
вычитания также определены.
Для строковых величин оператор "+"
обозначает конкатенацию.

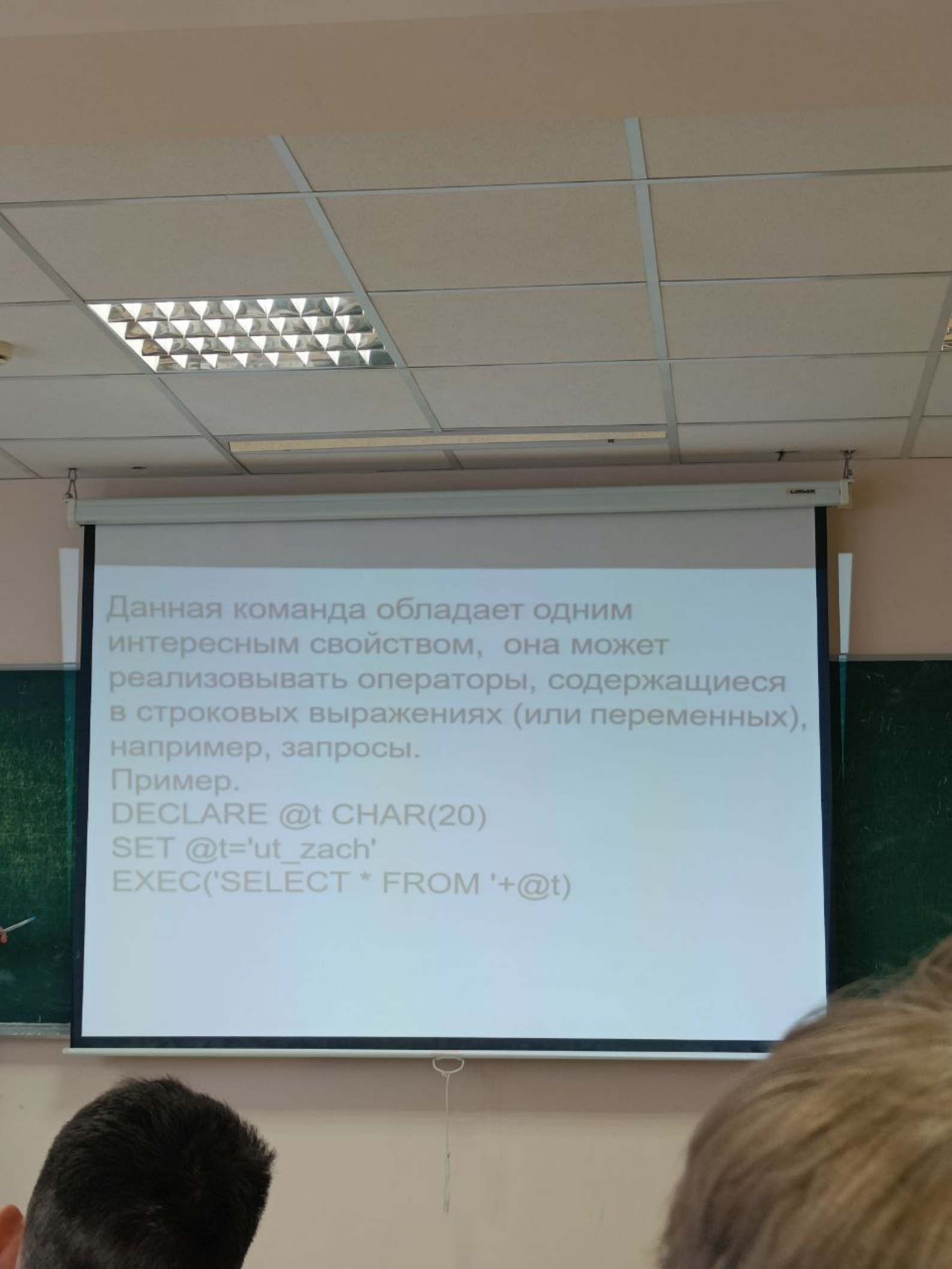

Операторные скобки:
BEGIN...END — предназначены для группировки нескольких команд в один блок. Операторные скобки могут быть вложенными, глубина вложения произвольна.

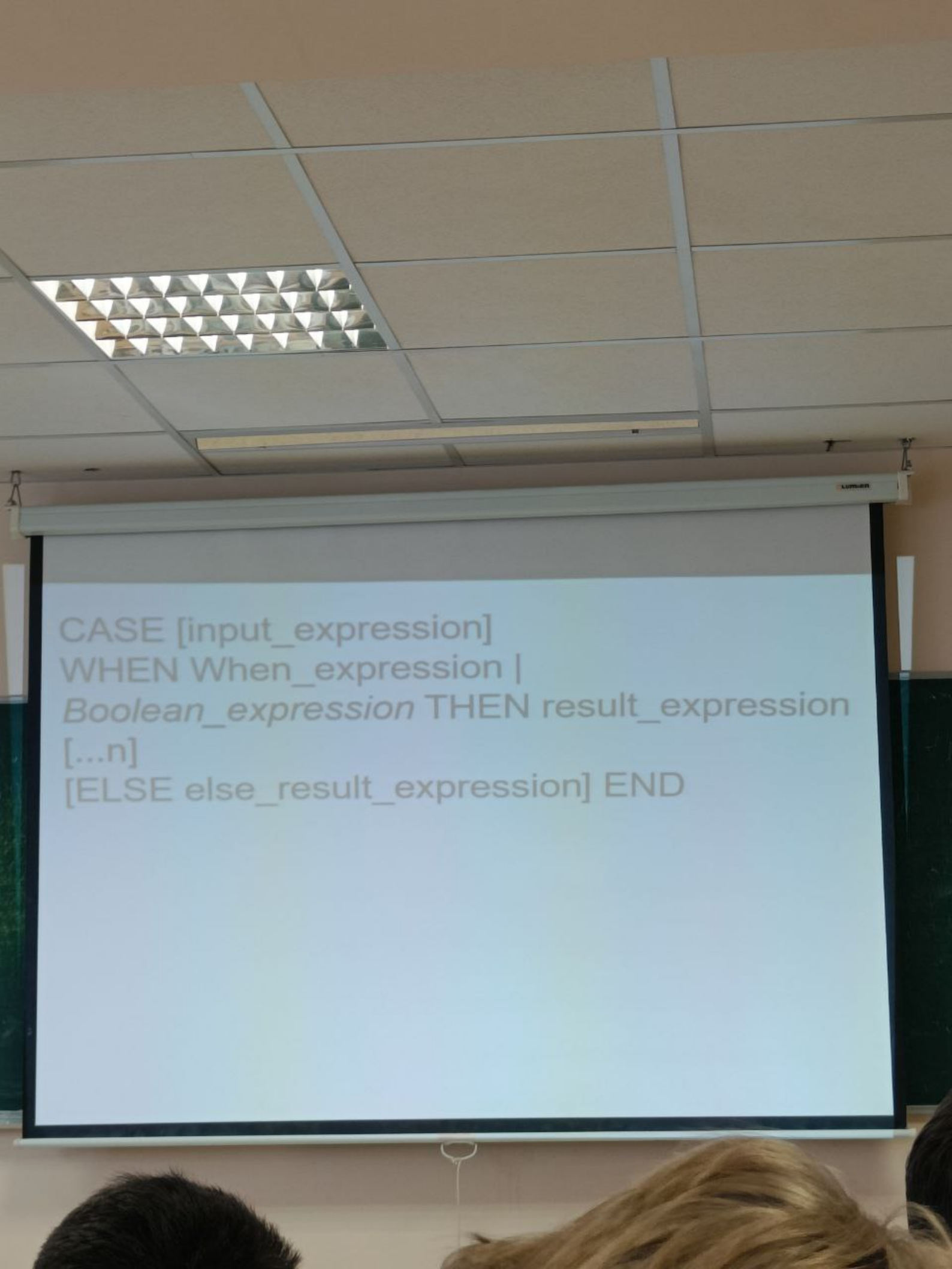

Логические функции
IIF (boolean_expression, true_value, false_value)
CHOOSE (index, val_1, val_2 [, val_n])

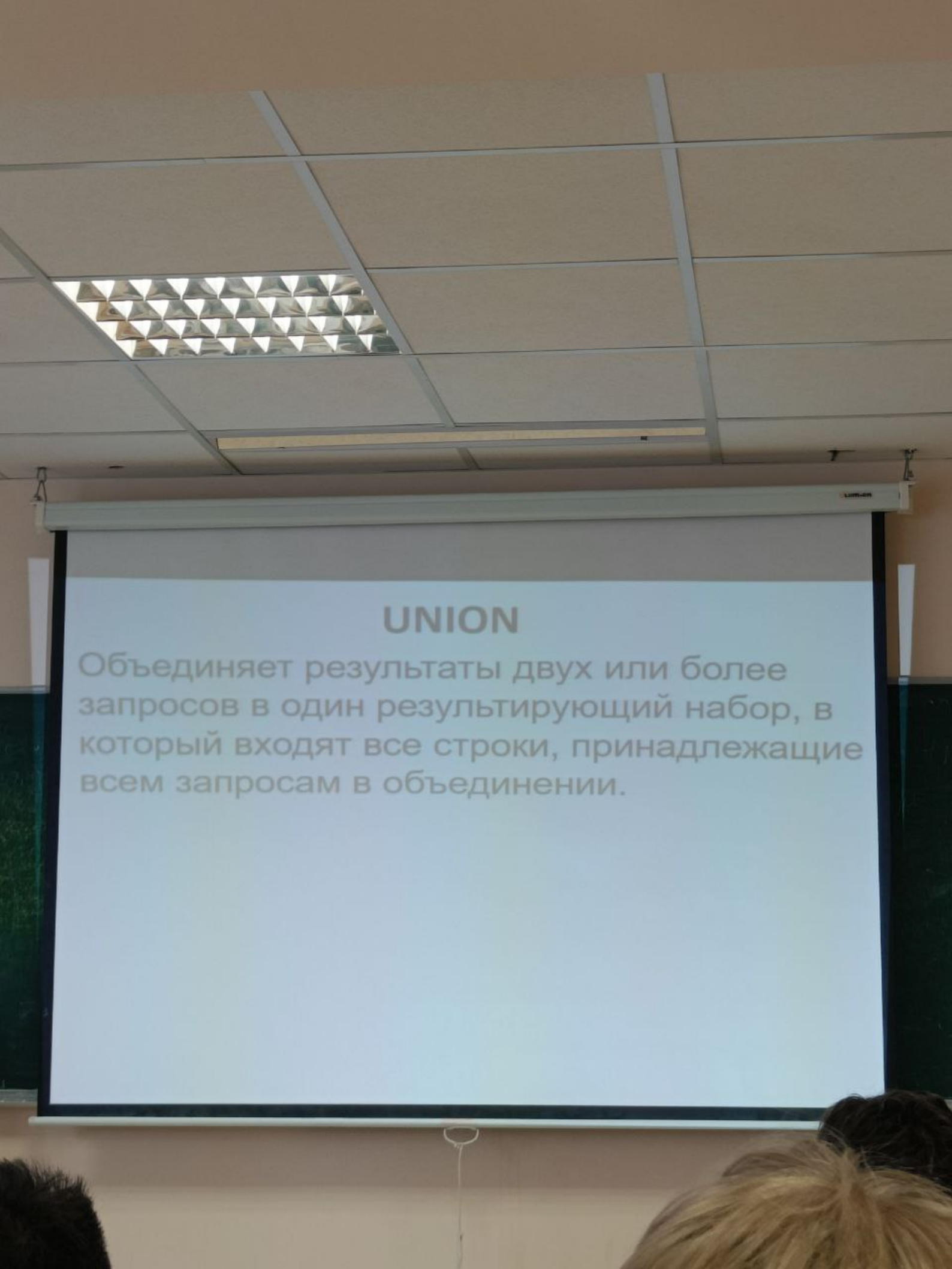

Оператор цикла:
WHILE Boolean_expression
(sql_statement | statement_block }
[BREAK] { sql_statement |
statement_block } [CONTINUE]

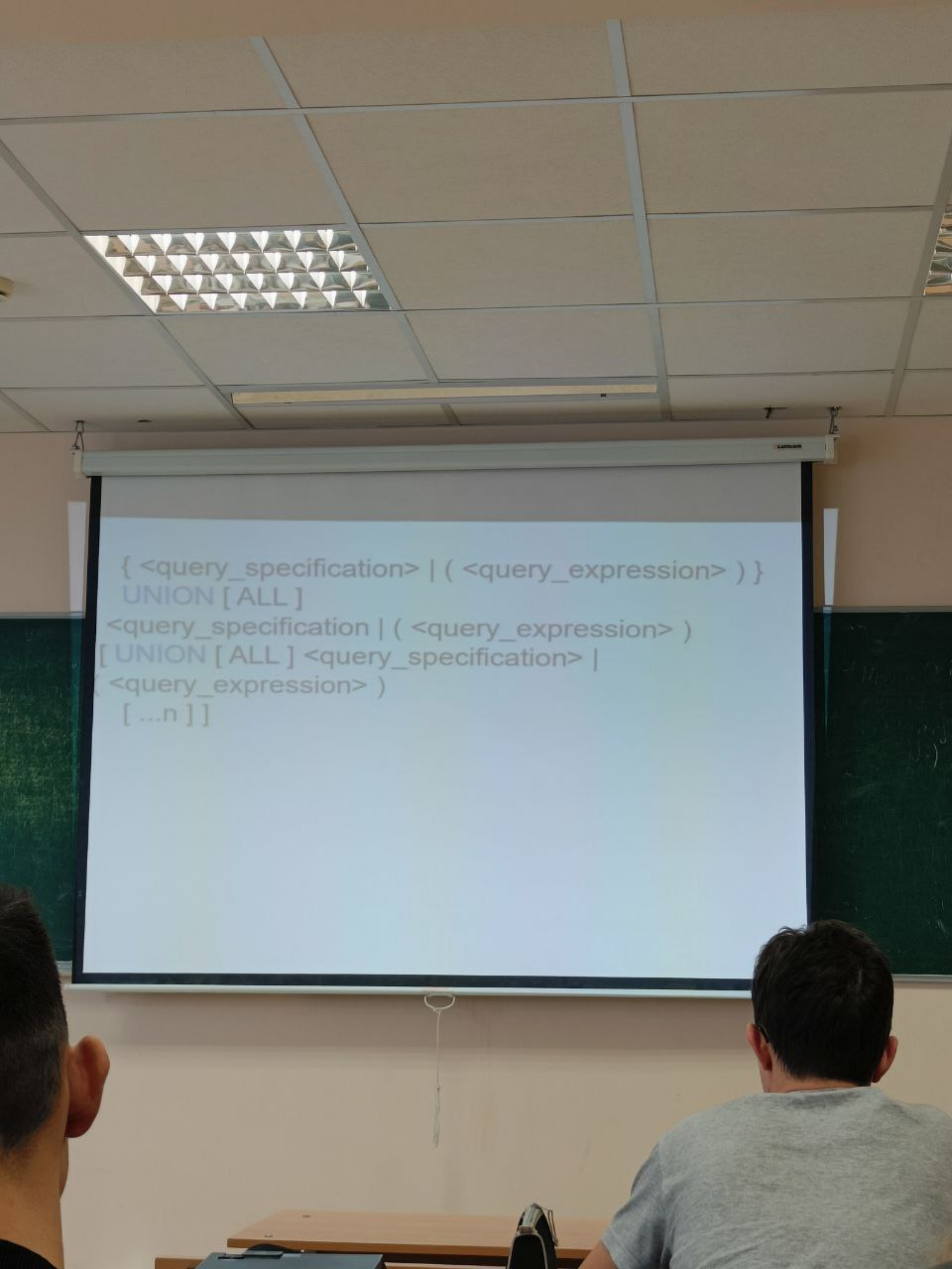

Оператор GOTO
Оператор RETURN
Используется для выхода из хранимых процедур, триггеров и функций, а также возвращения значений функций.

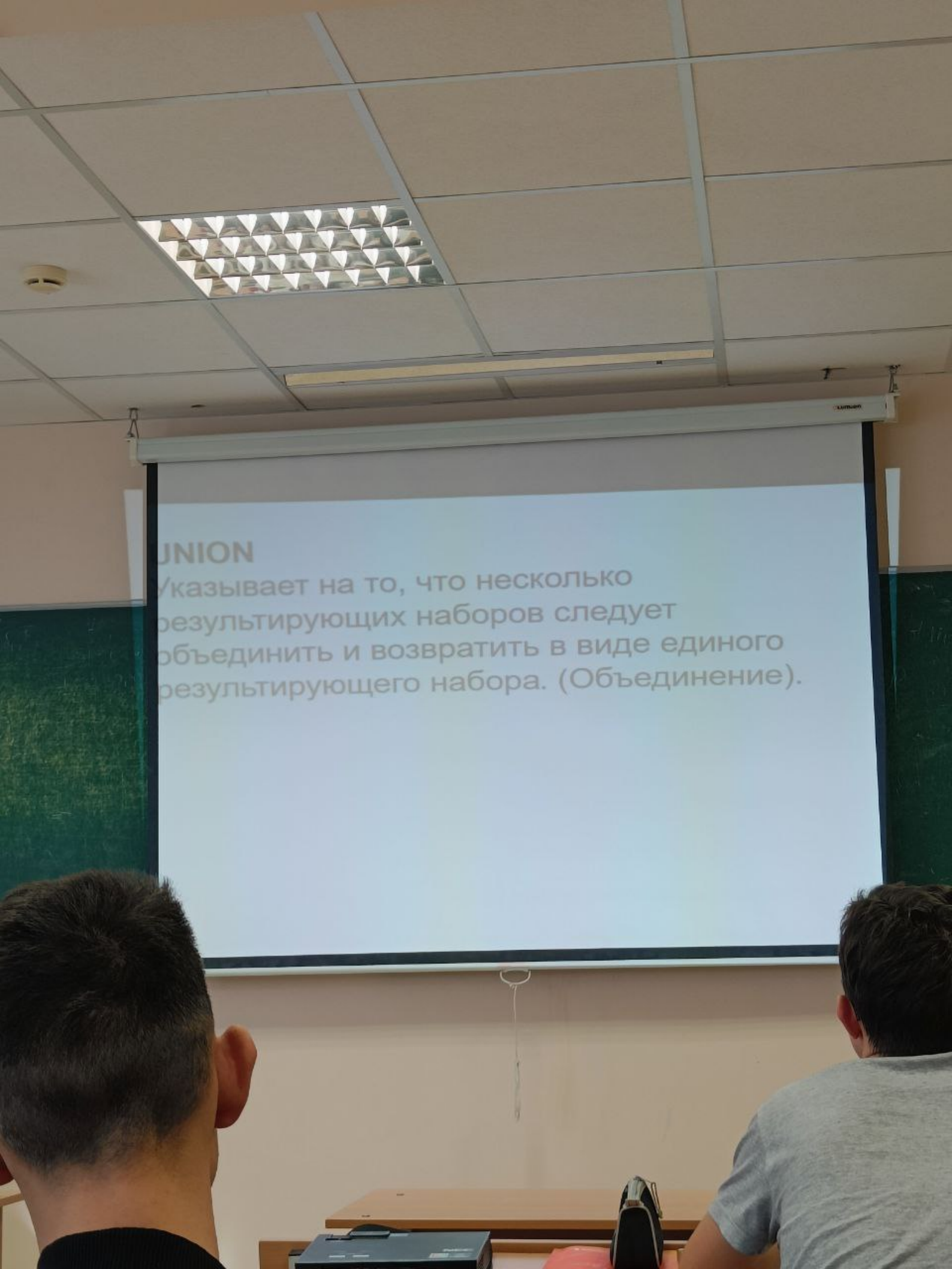

Оператор EXECUTE имеет следующую структуру:
[{ EXEC | EXECUTE }]
{
[@return_status =]
{ module_name [;number] |
@module_name_var }
[[@parameter =] { value
| @variable [OUTPUT]
| [DEFAULT]
}
]
[,...n]
[WITH RECOMPILE]
}
[;]

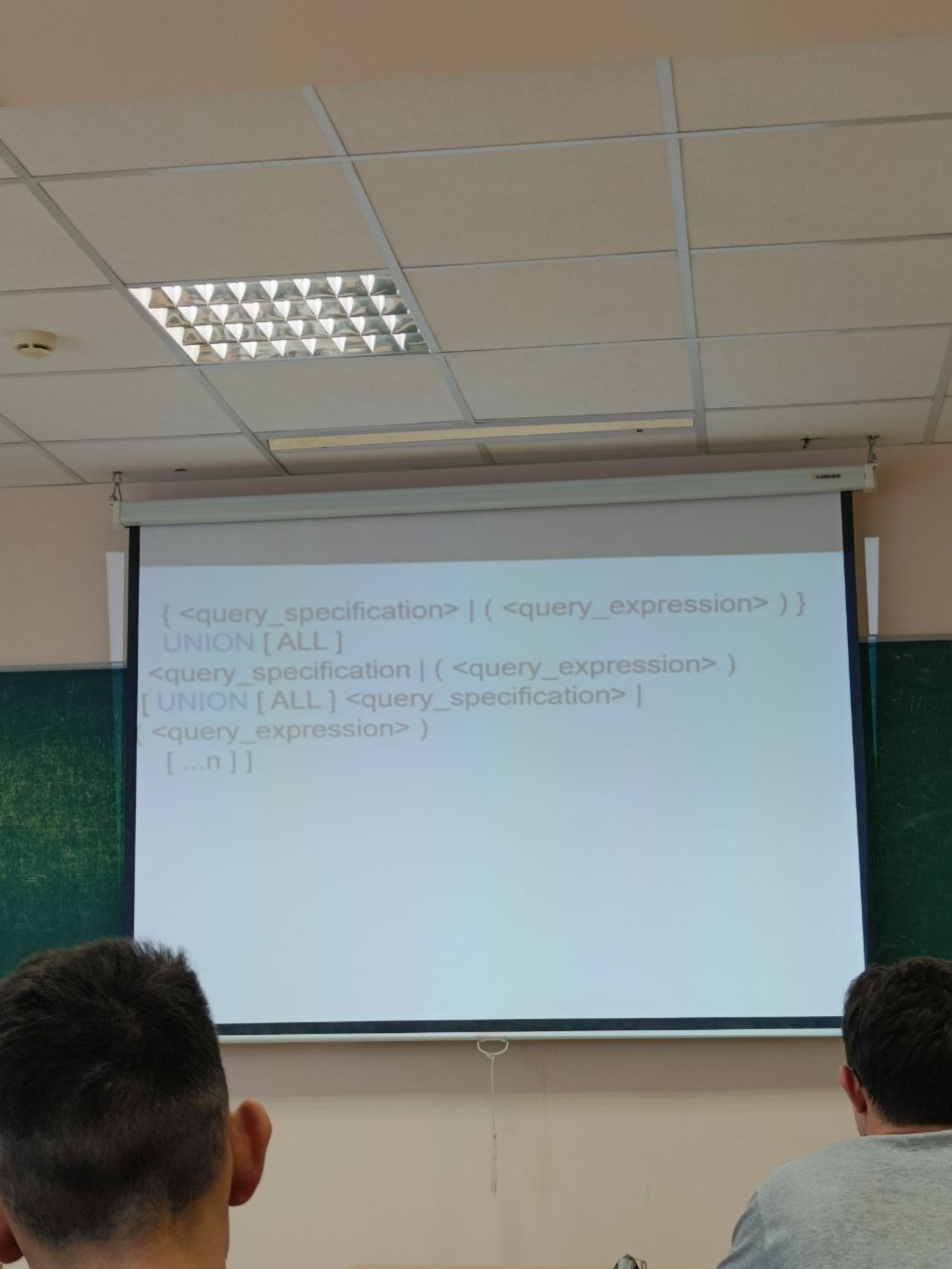

Данная команда обладает одним интересным свойством, она может реализовывать операторы, содержащиеся в строковых выражениях (или переменных), например, запросы.
Пример.
DECLARE @t CHAR(20)
SET @t='ut_zach'
EXEC('SELECT * FROM '+@t)

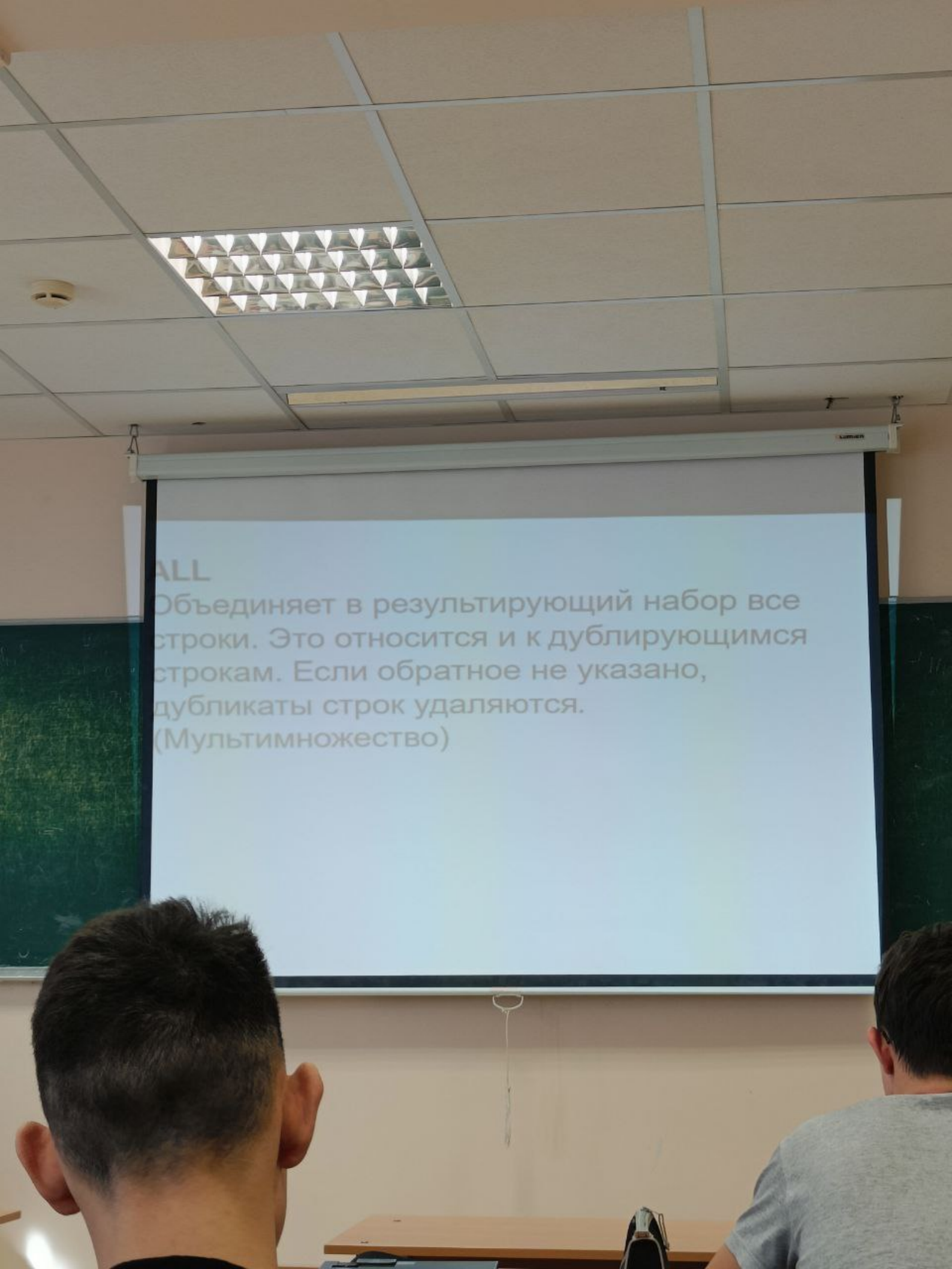

CASE [input_expression]
WHEN When_expression |
Boolean_expression THEN result_expression
[...n]
[ELSE else_result_expression] END

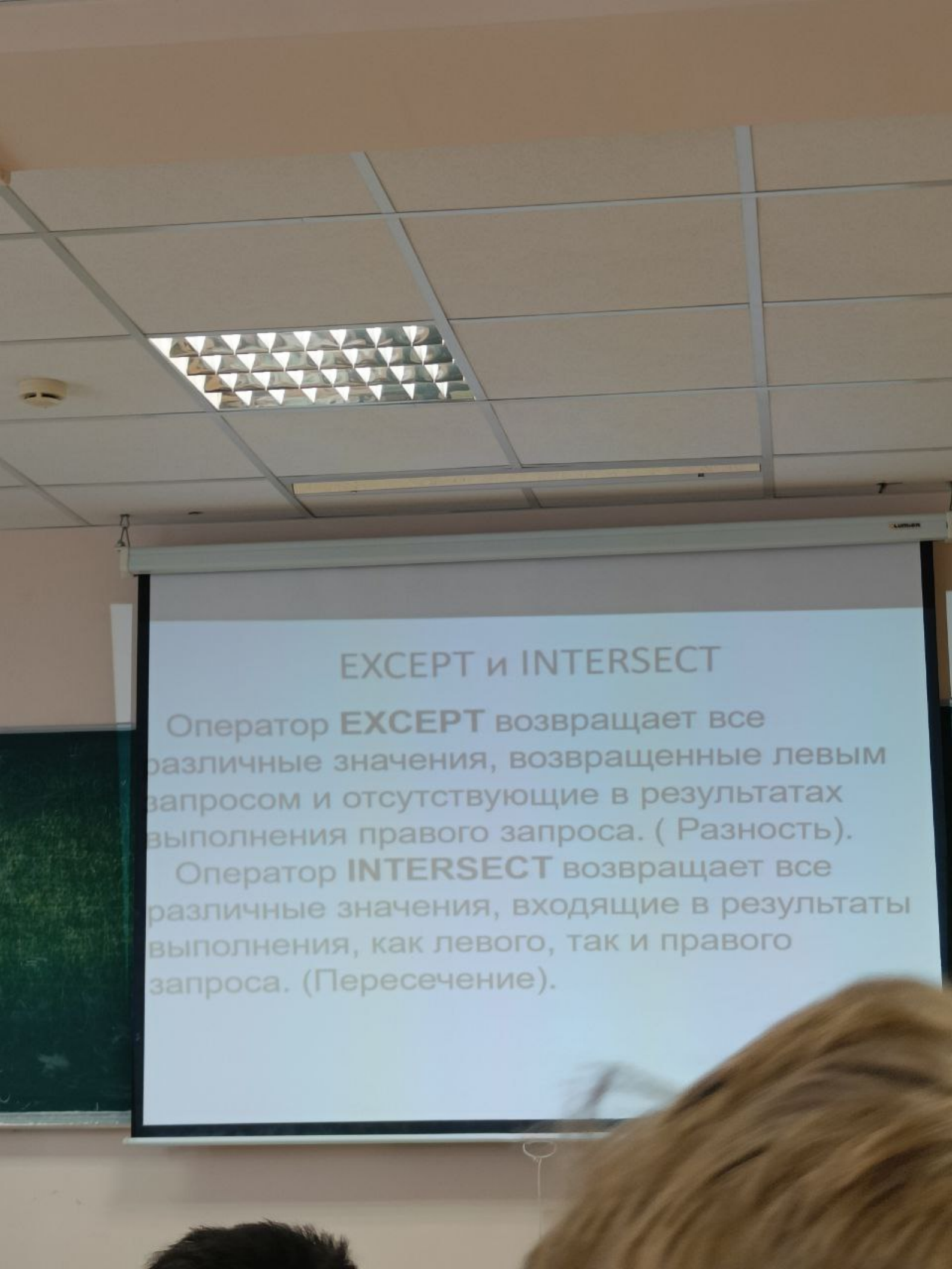

UNION
Объединяет результаты двух или более запросов в один результирующий набор, в который входят все строки, принадлежащие всем запросам в объединении.

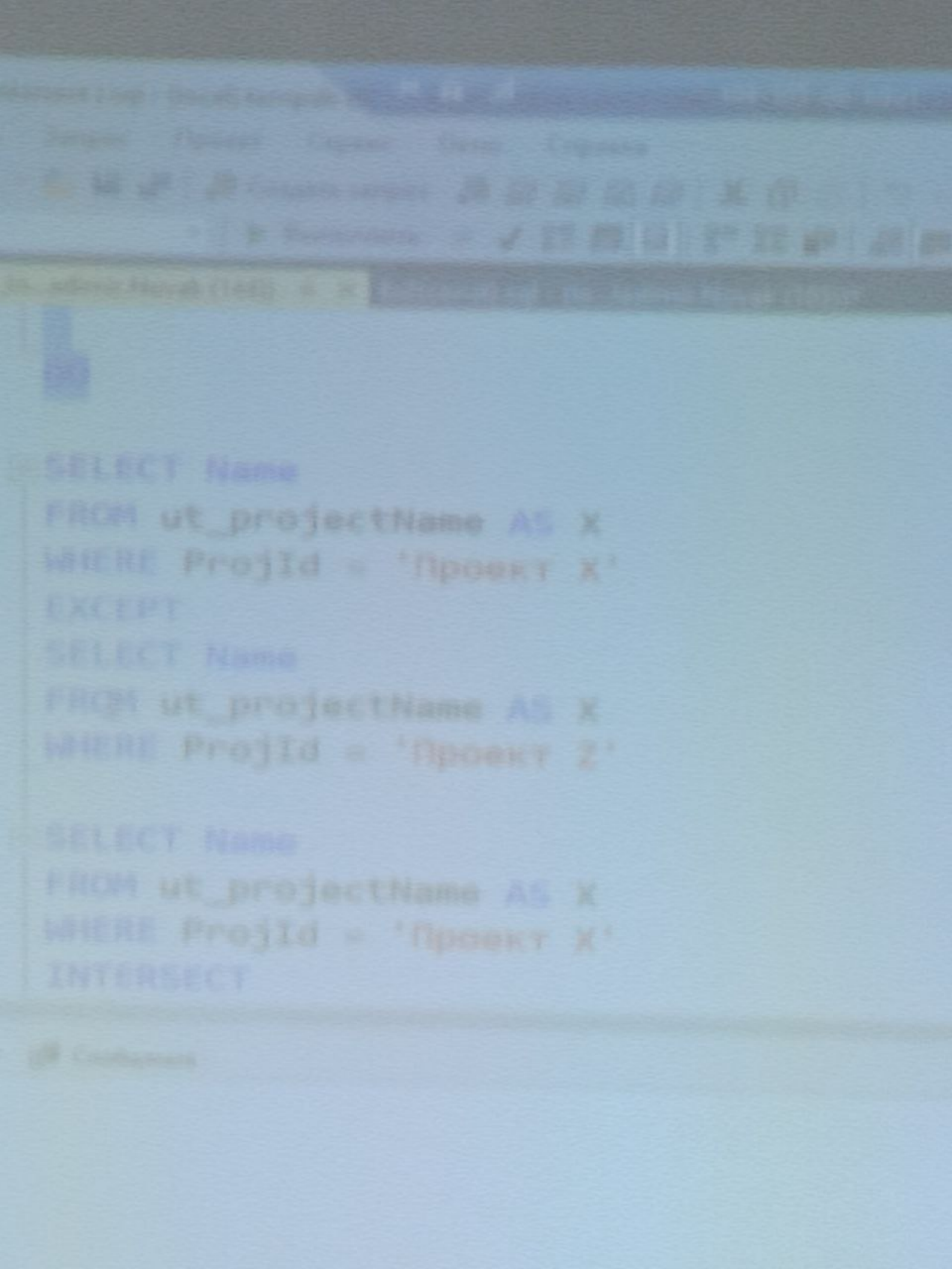

{ <query_specification> | (<query_expression>) }
UNION [ALL]
<query_specification | (<query_expression>)
[UNION [ALL] <query_specification> |
<query_expression>)
[...n]]


UNION
Указывает на то, что несколько
результирующих наборов следует
объединить и возвратить в виде единого
результирующего набора. (Объединение).


{ <query_specification> | (<query_expression>) }
UNION [ALL]
<query_specification | (<query_expression>)
[UNION [ALL] <query_specification> |
(<query_expression>)
[...n]]


ALL
Объединяет в результирующий набор все
строки. Это относится и к дублирующимся
строкам. Если обратное не указано,
дубликаты строк удаляются.
(Мультимножество)


EXCEPT и INTERSECT
Оператор EXCEPT возвращает все различные значения, возвращенные левым запросом и отсутствующие в результатах выполнения правого запроса. (Разность).
Оператор INTERSECT возвращает все различные значения, входящие в результаты выполнения, как левого, так и правого запроса. (Пересечение).



```
SELECT Name
FROM ut_projectName AS x
WHERE ProjId = 'Проект X'
EXCEPT
SELECT Name
FROM ut_projectName AS x
WHERE ProjId = 'Проект 2'

SELECT Name
FROM ut_projectName AS x
WHERE ProjId = 'Проект X'
INTERSECT
```