Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут”

Кафедра АСОІУ

**ЗВІТ**

про виконання комп’ютерного практикуму № 1

з дисципліни

“ Програмні додатки з використанням баз даних”

по темі

«Аналітичні запити»

Варіант 6

|  |  |  |
| --- | --- | --- |
| Прийняв: |  | Виконав: |
| Cт.вик. Клименко О. М. |  | студент 3-го курсу  гр. ІП-51 ФІОТ  Зарічковий Олександр Анатолійович |

Київ – 2017

**ЗМІСТ:**

[1 Завдання 3](#_Toc495230724)

[2 Виконання Завдання 5](#_Toc495230725)

# Завдання

Завдання 1:

Кожен студент згідно предметної області за варіантом повинен визначити таблицю\_1, що містить оперативні дані, та таблицю\_2, що містить довідник, що використовується в таблиці\_1, скласти наступні види запитів, попередньо описавши словами, що ці запити виконують (запит має бути адекватним, а не формальне застосування наведених службових слів):

1) Використовуючи count() (або будь-яку іншу агрегатну функцію), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції.

2) Використовуючи rank() або dense\_rank(), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції.

3) Використовуючи sliding window (rows), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції.

4) Використовуючи sliding window (range) , partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції.

5) Самостійно розібратися, як застосовується функція lag(). Використовуючи lag(), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції.

6) Самостійно розібратися, як застосовується функція lead(). Використовуючи lead(), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції.

Завдання 2 (повторення «Нормалізації» - матеріалу попереднього семестру):

Згідно Вашої предметної області, вибрати таблицю:

1) Змінити таблицю так, щоб вона не була у першій нормальній формі, навести цю таблицю у наповненому вигляді. Потім описати, які зміни треба зробити, щоб вона відповідала першій нормальній формі, навести цю таблицю у наповненому вигляді.

2) Описати, які зміни треба зробити, щоб таблиця відповідала другій нормальній формі, навести цю таблицю у наповненому вигляді.

3) Описати, які зміни треба зробити, щоб таблиця відповідала третій нормальній формі, навести цю таблицю у наповненому вигляді.

4) Описати, які зміни треба зробити, щоб таблиця відповідала нормальній формі Бойса-Кода, навести цю таблицю у наповненому вигляді.

**Преметна область**: Навчання з охорони праці

# Виконання Завдання

1. Завдання по написанню запитів
   1. Використовуючи count() (або будь-яку іншу агрегатну функцію), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції:  
      *Призначення*: Для кожного предмета підрахувати його дохід. Дохід предмета = Кількість слухачів \* Ціна предмету.

*Текст запиту з використанням аналітичних функцій*:

SELECT DISTINCT

[T].[SubjectId],

SUM([S].[SubjectPrice]) OVER (PARTITION BY [T].[SubjectId]) AS [SubjectIncome]

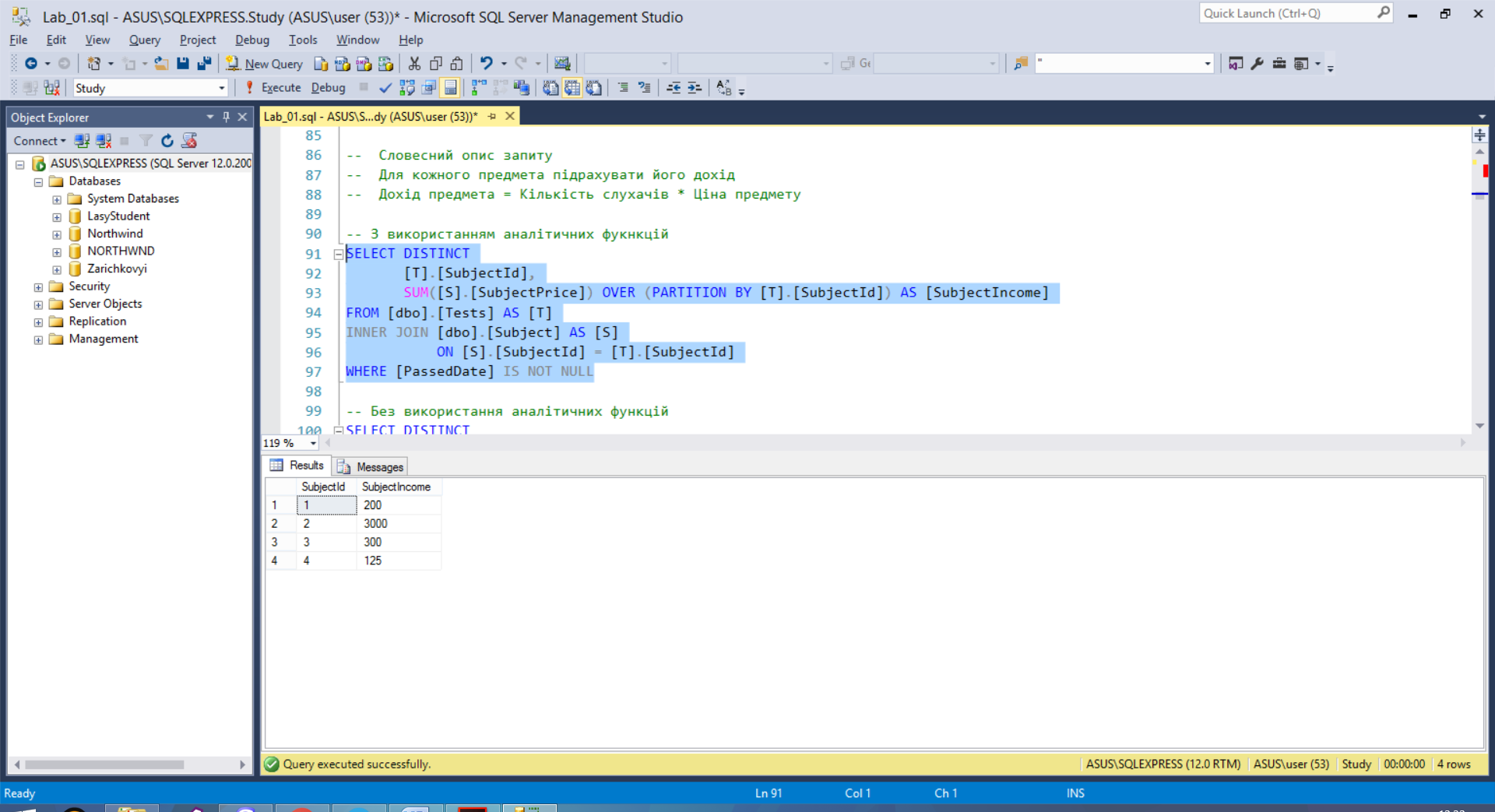
FROM [dbo].[Tests] AS [T]

INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [PassedDate] IS NOT NULL

*Результат виконання з використання аналітичних функцій*:



*Текст запиту без використанням аналітичних функцій*:

SELECT DISTINCT

[T].[SubjectId],

SUM([S].[SubjectPrice]) AS [SubjectIncome]

FROM [dbo].[Tests] AS [T]

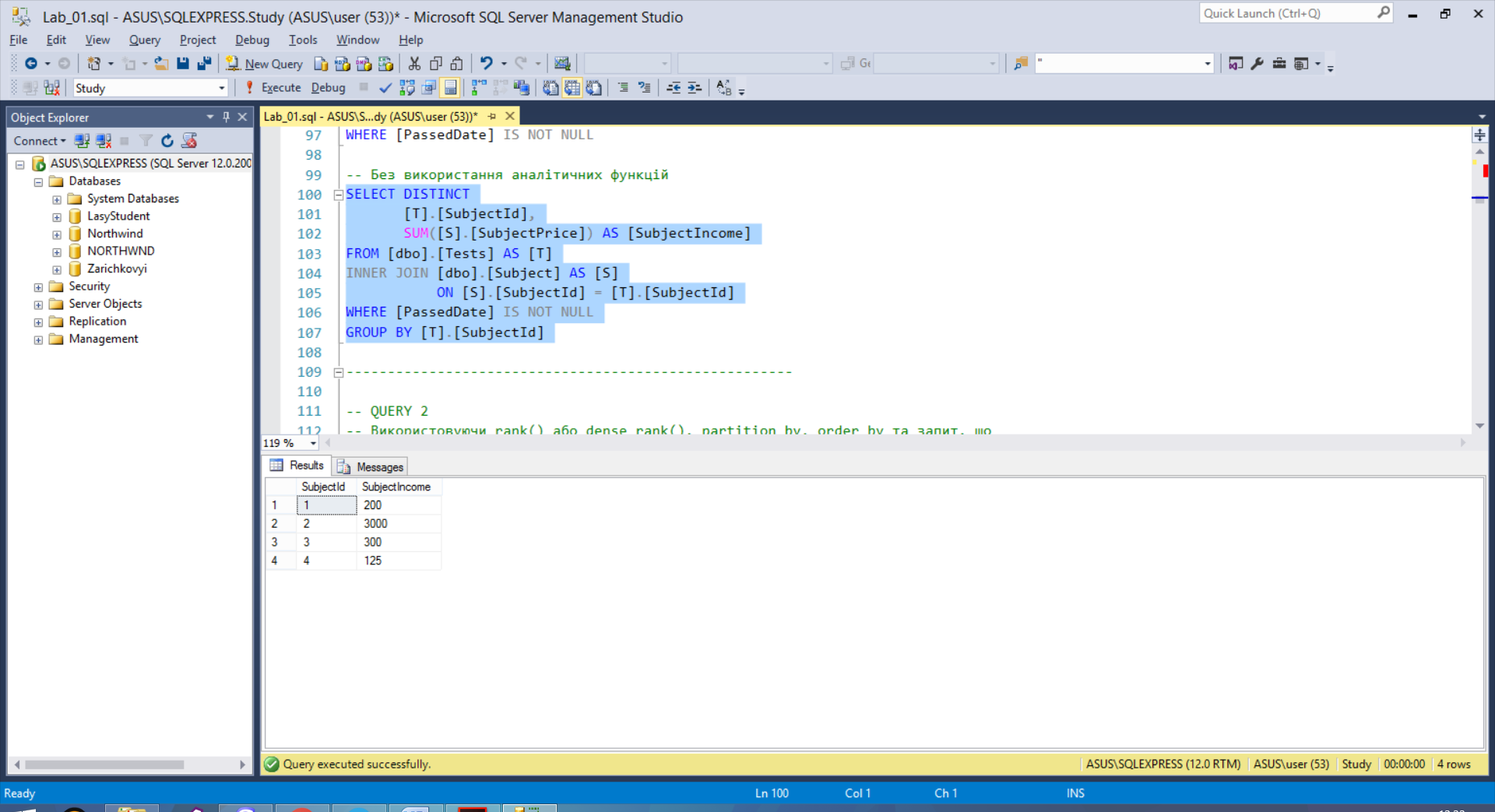
INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [PassedDate] IS NOT NULL

GROUP BY [T].[SubjectId]

*Результат виконання без використання аналітичних функцій*:



* 1. Використовуючи rank() або dense\_rank(), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції:

*Призначення*: Використовуючи rank() або dense\_rank(), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції..

*Текст запиту з використанням аналітичних функцій*:

SELECT \*

FROM (

SELECT DISTINCT

[T].[ListenerId],

[S].[SubjectPrice],

RANK() OVER (PARTITION BY [T].[ListenerId]

ORDER BY [S].[SubjectPrice] DESC) AS [Top]

FROM [dbo].[Tests] AS [T]

INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [PassedDate] IS NOT NULL) AS [T]

WHERE [T].[Top] = 1

*Результат виконання з використання аналітичних функцій*:



*Текст запиту без використанням аналітичних функцій*:

SELECT DISTINCT

[T].[ListenerId],

MAX([S].[SubjectPrice]) AS [Top]

FROM [dbo].[Tests] AS [T]

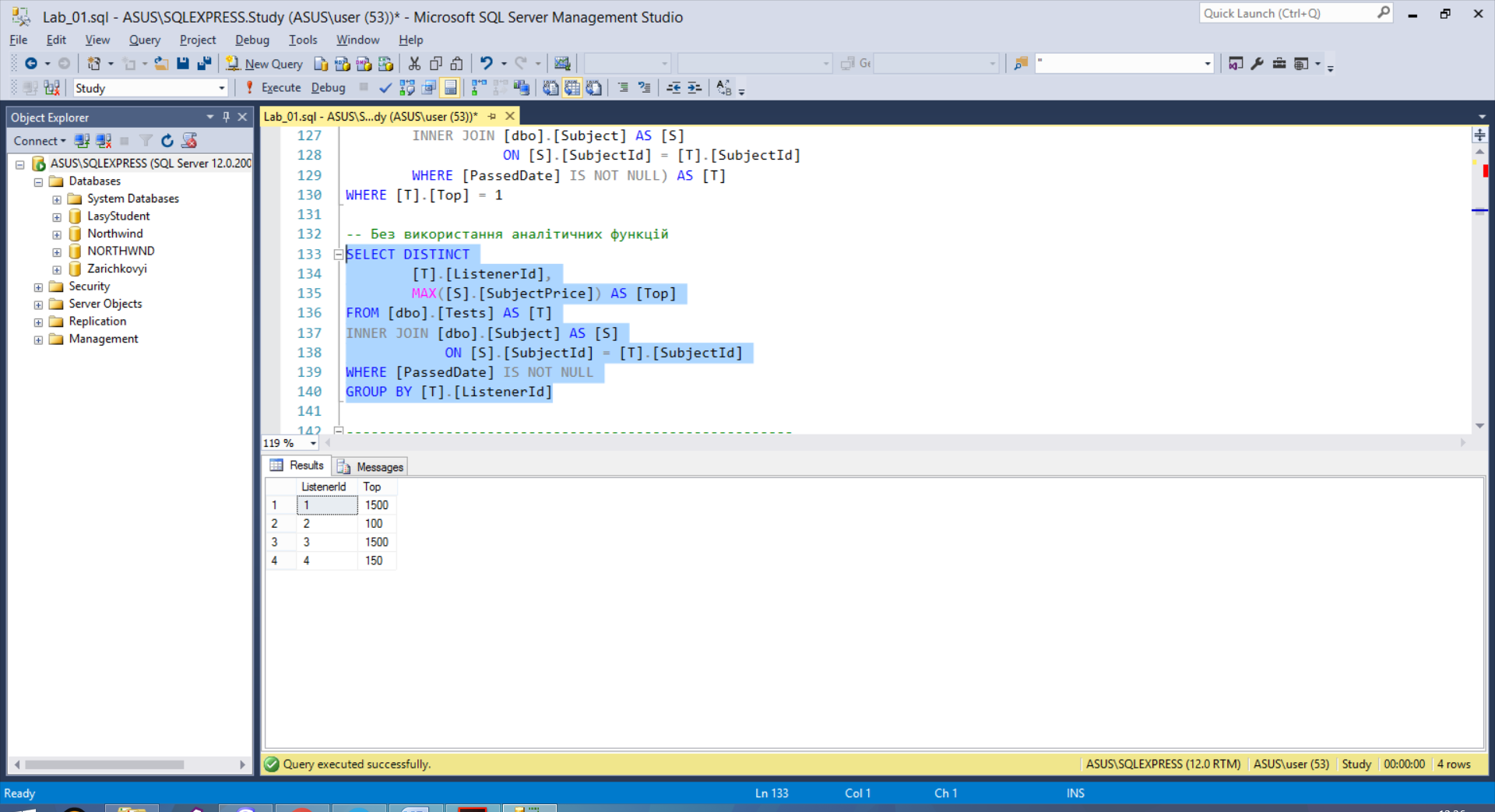
INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [PassedDate] IS NOT NULL

GROUP BY [T].[ListenerId]

*Результат виконання без використання аналітичних функцій*:



* 1. Використовуючи sliding window (rows), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції

*Призначення*: Для кожного слухача та кожного придбаного курсу підрахувати сердні розходи на курси (хронологічно середнє поточного та +/- купленого 1 курса).

*Текст запиту з використанням аналітичних функцій*:

;WITH [merged\_table] ([TestId], [SubjectId], [ListenerId], [SubjectPrice], [PassedDate]) AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[S].[SubjectPrice],

[T].[PassedDate]

FROM [dbo].[Tests] AS [T]

INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [T].[PassedDate] IS NOT NULL

)

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

AVG([T].[SubjectPrice]) OVER(PARTITION BY [T].[ListenerId] ORDER BY [T].[PassedDate]

ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING) AS [AVG\_PRICE],

[T].[PassedDate]

FROM [merged\_table] AS [T]

*Результат виконання з використання аналітичних функцій*:



*Текст запиту без використанням аналітичних функцій*:

;WITH [merged\_table] ([TestId], [SubjectId], [ListenerId], [SubjectPrice], [PassedDate]) AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[S].[SubjectPrice],

[T].[PassedDate]

FROM [dbo].[Tests] AS [T]

INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [T].[PassedDate] IS NOT NULL

),

[rows\_table] AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[T].[SubjectPrice],

[T].[PassedDate],

COUNT([T2].[SubjectId]) AS [ROW\_COUNT]

FROM [merged\_table] AS [T], [merged\_table] AS [T2]

WHERE [T].[PassedDate] >= [T2].[PassedDate]

AND [T].[ListenerId] = [T2].[ListenerId]

GROUP BY [T].[TestId], [T].[SubjectId],

[T].[ListenerId], [T].[SubjectPrice],

[T].[PassedDate]

)

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

AVG([T2].[SubjectPrice]) AS [AVG\_PRICE],

[T].[PassedDate]

FROM [rows\_table] AS [T],

[rows\_table] AS [T2]

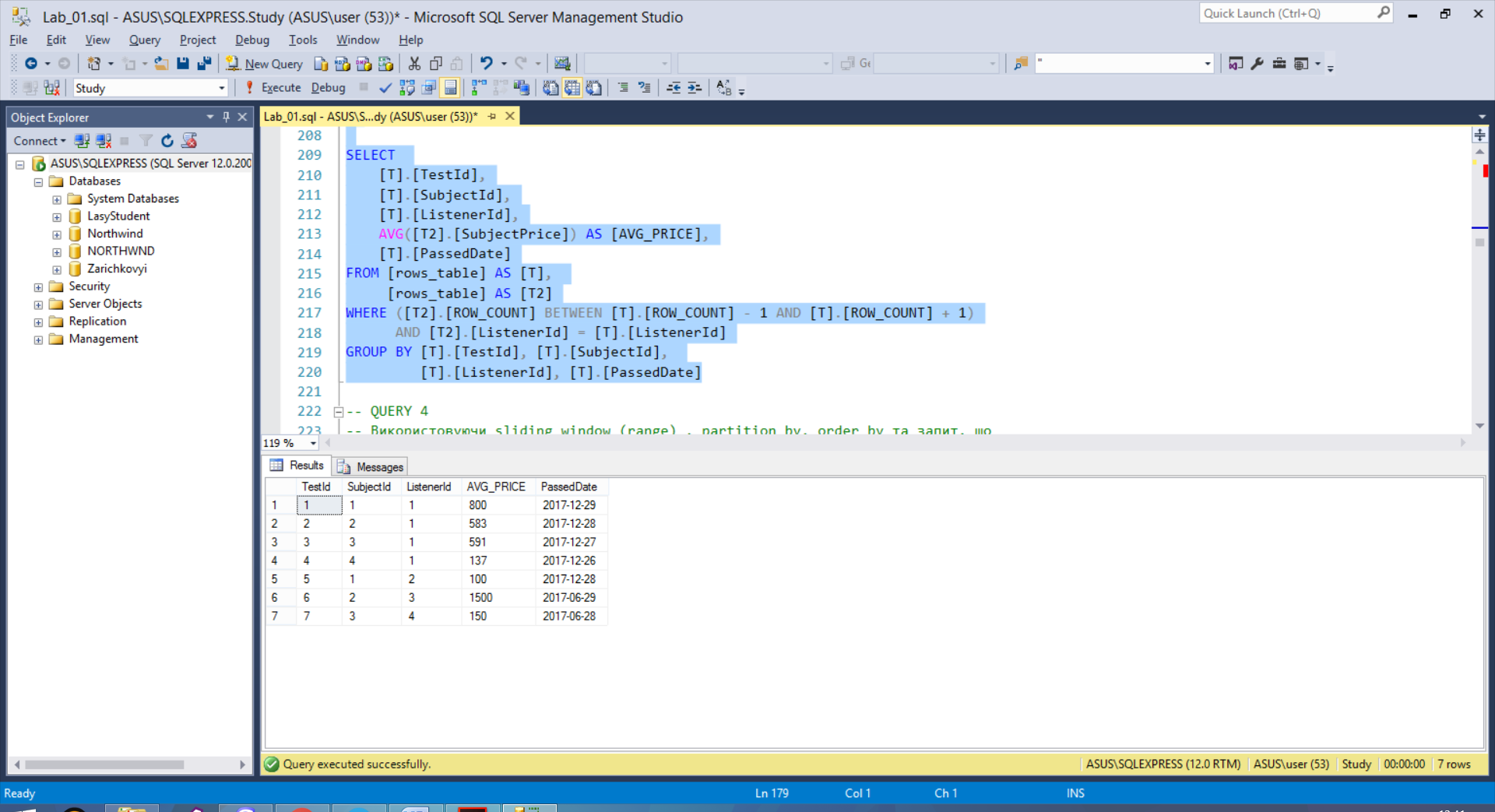
WHERE ([T2].[ROW\_COUNT] BETWEEN [T].[ROW\_COUNT] - 1 AND [T].[ROW\_COUNT] + 1)

AND [T2].[ListenerId] = [T].[ListenerId]

GROUP BY [T].[TestId], [T].[SubjectId],

[T].[ListenerId], [T].[PassedDate]

*Результат виконання без використання аналітичних функцій*:



* 1. Використовуючи sliding window (range) , partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції.:

*Призначення*: Для кожного слухача та кожного придбаного курсу підрахувати сердні розходи на курси (хронологічно середнє курсів придбаних починаючи з поточного курсу).

*Текст запиту з використанням аналітичних функцій*:

;WITH [merged\_table] ([TestId], [SubjectId], [ListenerId], [SubjectPrice], [PassedDate]) AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[S].[SubjectPrice],

[T].[PassedDate]

FROM [dbo].[Tests] AS [T]

INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [T].[PassedDate] IS NOT NULL

)

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

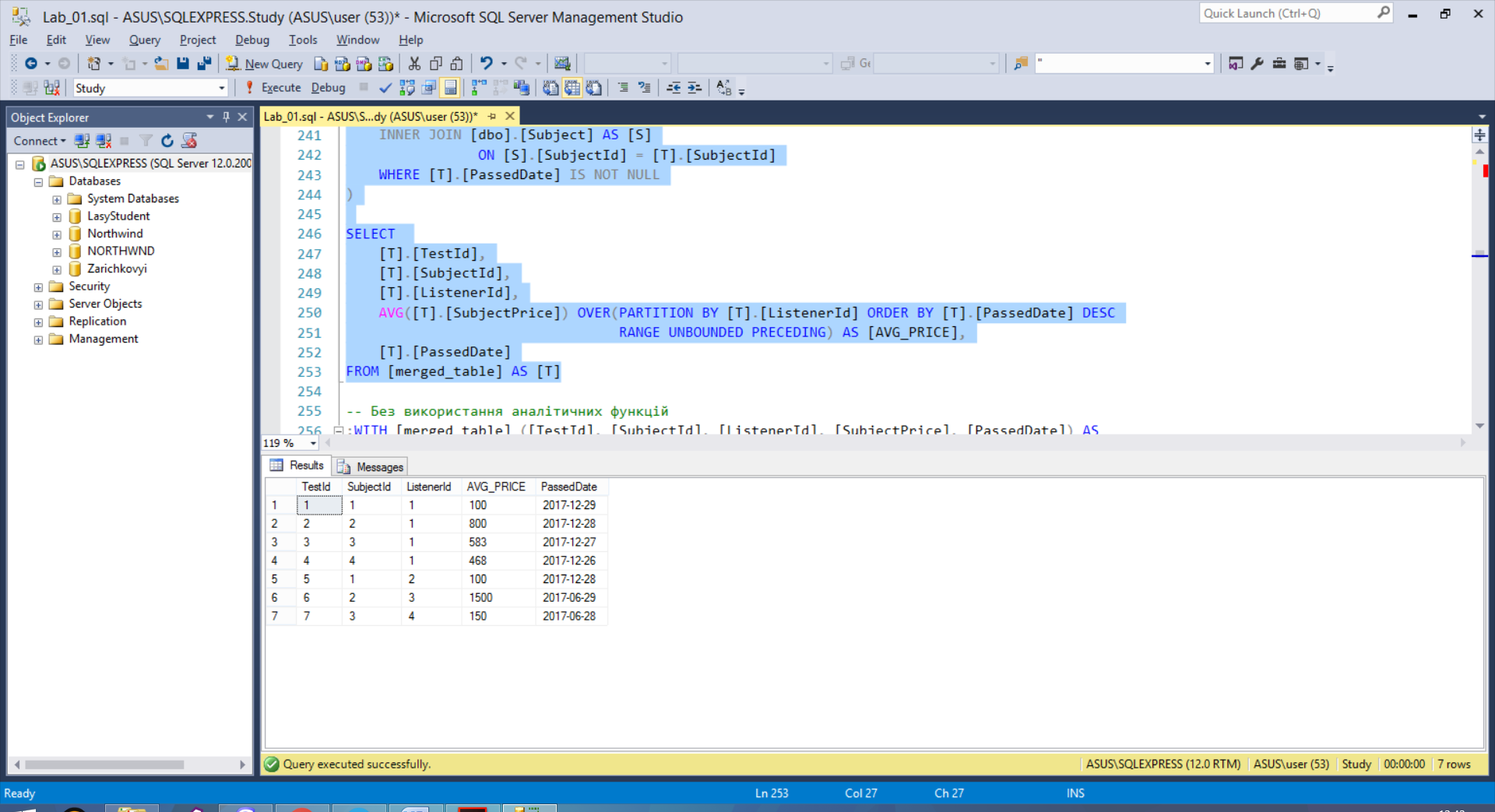
AVG([T].[SubjectPrice]) OVER(PARTITION BY [T].[ListenerId] ORDER BY [T].[PassedDate] DESC

RANGE UNBOUNDED PRECEDING) AS [AVG\_PRICE],

[T].[PassedDate]

FROM [merged\_table] AS [T]

*Результат виконання з використання аналітичних функцій*:



*Текст запиту без використанням аналітичних функцій*:

;WITH [merged\_table] ([TestId], [SubjectId], [ListenerId], [SubjectPrice], [PassedDate]) AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[S].[SubjectPrice],

[T].[PassedDate]

FROM [dbo].[Tests] AS [T]

INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [T].[PassedDate] IS NOT NULL

)

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

AVG([T2].[SubjectPrice]) AS [AVG\_PRICE],

[T].[PassedDate]

FROM [merged\_table] AS [T],

[merged\_table] AS [T2]

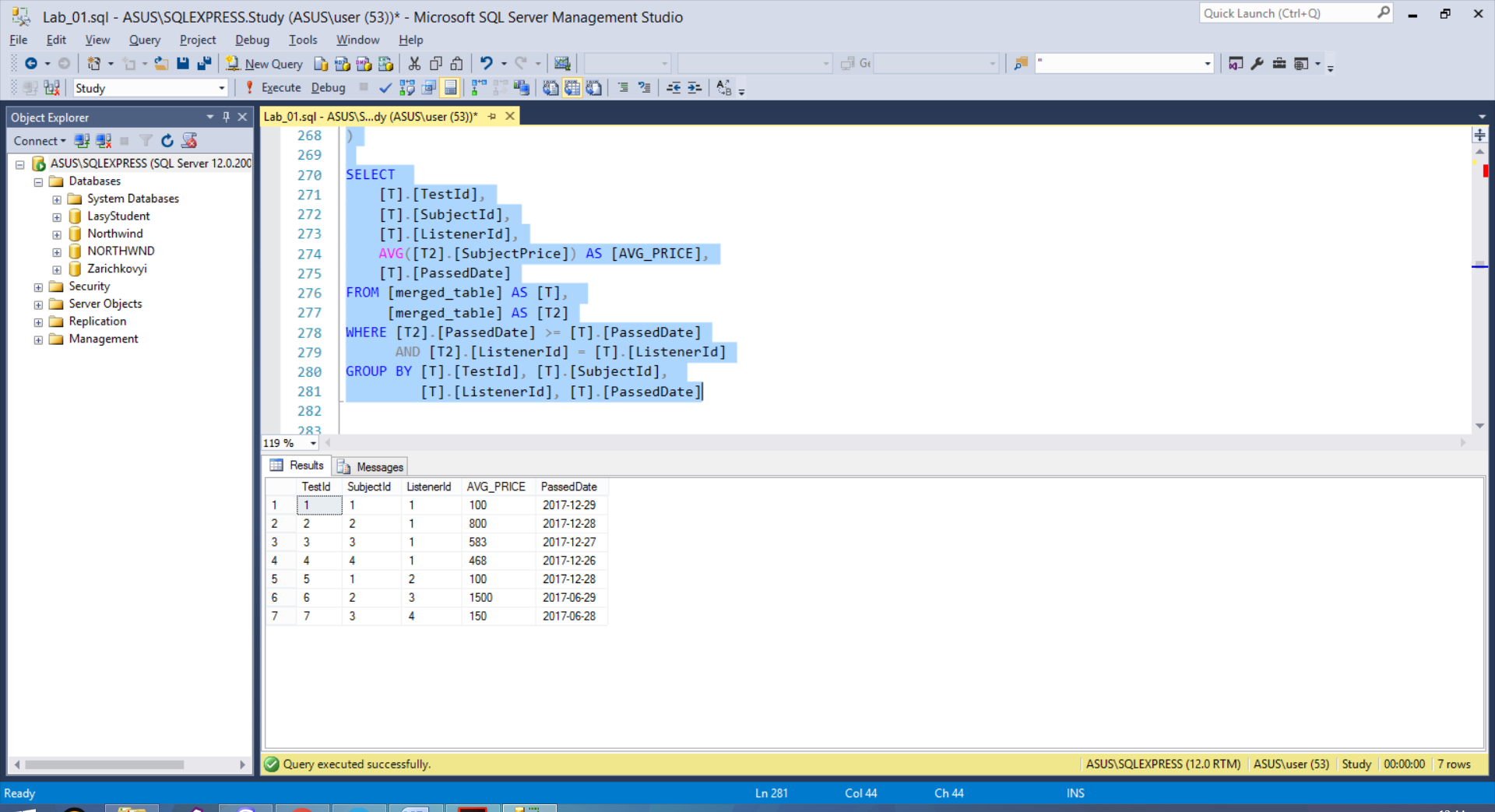
WHERE [T2].[PassedDate] >= [T].[PassedDate]

AND [T2].[ListenerId] = [T].[ListenerId]

GROUP BY [T].[TestId], [T].[SubjectId],

[T].[ListenerId], [T].[PassedDate]

*Результат виконання без використання аналітичних функцій*:



* 1. Самостійно розібратися, як застосовується функція lag(). Використовуючи lag(), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції:

*Призначення*: Для кожного слухача та кожного придбаного курсу вивести вартість поточного та хронологічно попереднього пройденого курса.

*Текст запиту з використанням аналітичних функцій*:

;WITH [merged\_table] ([TestId], [SubjectId], [ListenerId], [SubjectPrice], [PassedDate]) AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[S].[SubjectPrice],

[T].[PassedDate]

FROM [dbo].[Tests] AS [T]

INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [T].[PassedDate] IS NOT NULL

)

SELECT \*

FROM (

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[T].[SubjectPrice],

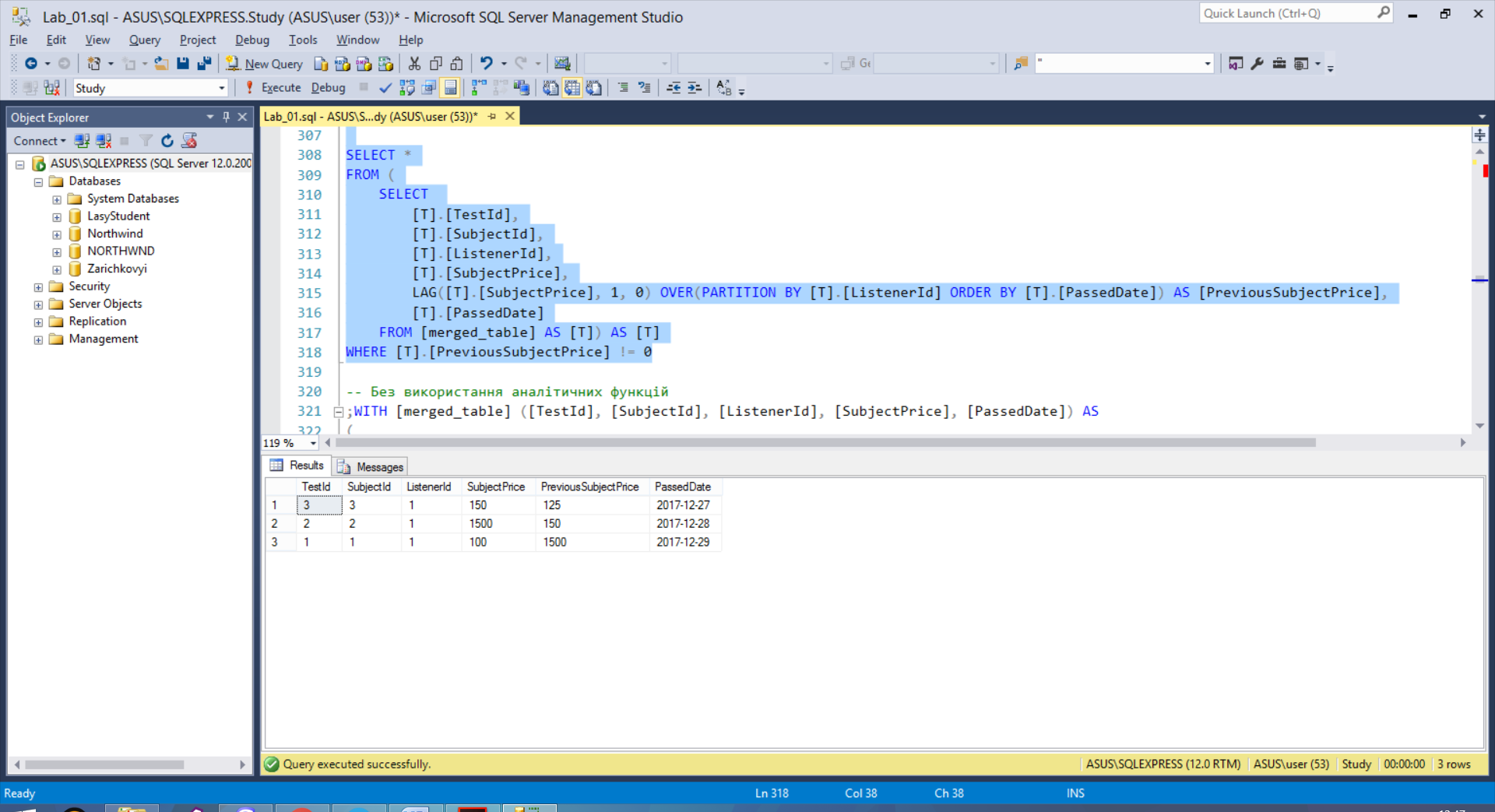
LAG([T].[SubjectPrice], 1, 0) OVER(PARTITION BY [T].[ListenerId] ORDER BY [T].[PassedDate]) AS [PreviousSubjectPrice],

[T].[PassedDate]

FROM [merged\_table] AS [T]) AS [T]

WHERE [T].[PreviousSubjectPrice] != 0

*Результат виконання з використання аналітичних функцій*:



*Текст запиту без використанням аналітичних функцій*:

;WITH [merged\_table] ([TestId], [SubjectId], [ListenerId], [SubjectPrice], [PassedDate]) AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[S].[SubjectPrice],

[T].[PassedDate]

FROM [dbo].[Tests] AS [T]

INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [T].[PassedDate] IS NOT NULL

),

[rows\_table] AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[T].[SubjectPrice],

[T].[PassedDate],

COUNT([T2].[SubjectId]) AS [ROW\_COUNT]

FROM [merged\_table] AS [T], [merged\_table] AS [T2]

WHERE [T].[PassedDate] >= [T2].[PassedDate]

AND [T].[ListenerId] = [T2].[ListenerId]

GROUP BY [T].[TestId], [T].[SubjectId],

[T].[ListenerId], [T].[SubjectPrice],

[T].[PassedDate]

)

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[T].[SubjectPrice],

[T2].[SubjectPrice] AS [PreviousSubjectPrice],

[T].[PassedDate]

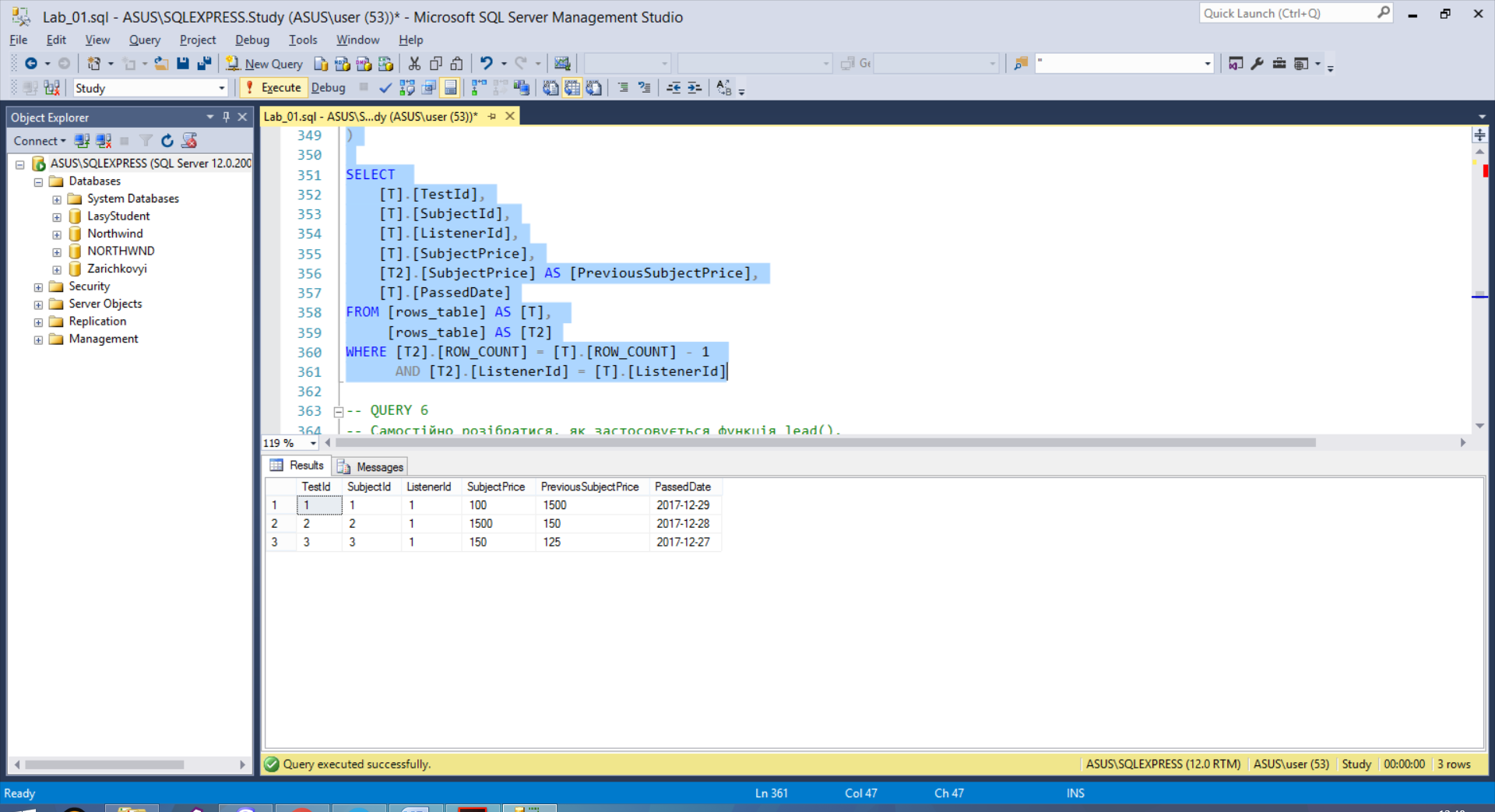
FROM [rows\_table] AS [T],

[rows\_table] AS [T2]

WHERE [T2].[ROW\_COUNT] = [T].[ROW\_COUNT] - 1

AND [T2].[ListenerId] = [T].[ListenerId]

*Результат виконання без використання аналітичних функцій*:



* 1. Самостійно розібратися, як застосовується функція lead(). Використовуючи lead(), partition by, order by та запит, що дасть такий самий результат, але не застосовуючи аналітичні функції:

*Призначення*: Для кожного слухача та кожного придбаного курсу вивести вартість поточного та хронологічно наступного пройденого курса.

*Текст запиту з використанням аналітичних функцій*:

;WITH [merged\_table] ([TestId], [SubjectId], [ListenerId], [SubjectPrice], [PassedDate]) AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[S].[SubjectPrice],

[T].[PassedDate]

FROM [dbo].[Tests] AS [T]

INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [T].[PassedDate] IS NOT NULL

)

SELECT \*

FROM (

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[T].[SubjectPrice],

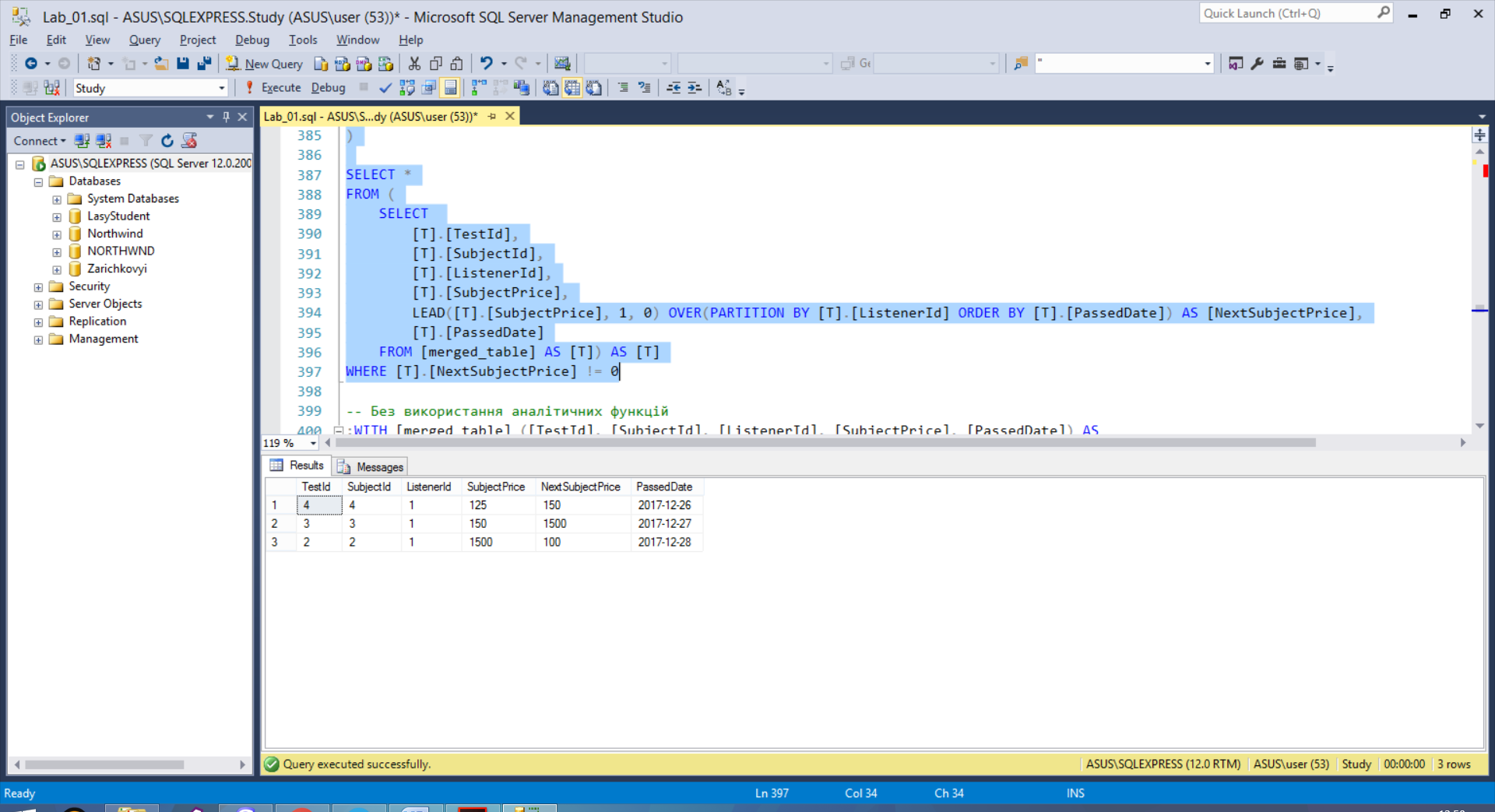
LEAD([T].[SubjectPrice], 1, 0) OVER(PARTITION BY [T].[ListenerId] ORDER BY [T].[PassedDate]) AS [NextSubjectPrice],

[T].[PassedDate]

FROM [merged\_table] AS [T]) AS [T]

WHERE [T].[NextSubjectPrice] != 0

*Результат виконання з використання аналітичних функцій*:



*Текст запиту без використанням аналітичних функцій*:

;WITH [merged\_table] ([TestId], [SubjectId], [ListenerId], [SubjectPrice], [PassedDate]) AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[S].[SubjectPrice],

[T].[PassedDate]

FROM [dbo].[Tests] AS [T]

INNER JOIN [dbo].[Subject] AS [S]

ON [S].[SubjectId] = [T].[SubjectId]

WHERE [T].[PassedDate] IS NOT NULL

),

[rows\_table] AS

(

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[T].[SubjectPrice],

[T].[PassedDate],

COUNT([T2].[SubjectId]) AS [ROW\_COUNT]

FROM [merged\_table] AS [T], [merged\_table] AS [T2]

WHERE [T].[PassedDate] >= [T2].[PassedDate]

AND [T].[ListenerId] = [T2].[ListenerId]

GROUP BY [T].[TestId], [T].[SubjectId],

[T].[ListenerId], [T].[SubjectPrice],

[T].[PassedDate]

)

SELECT

[T].[TestId],

[T].[SubjectId],

[T].[ListenerId],

[T].[SubjectPrice],

[T2].[SubjectPrice] AS [NextSubjectPrice],

[T].[PassedDate]

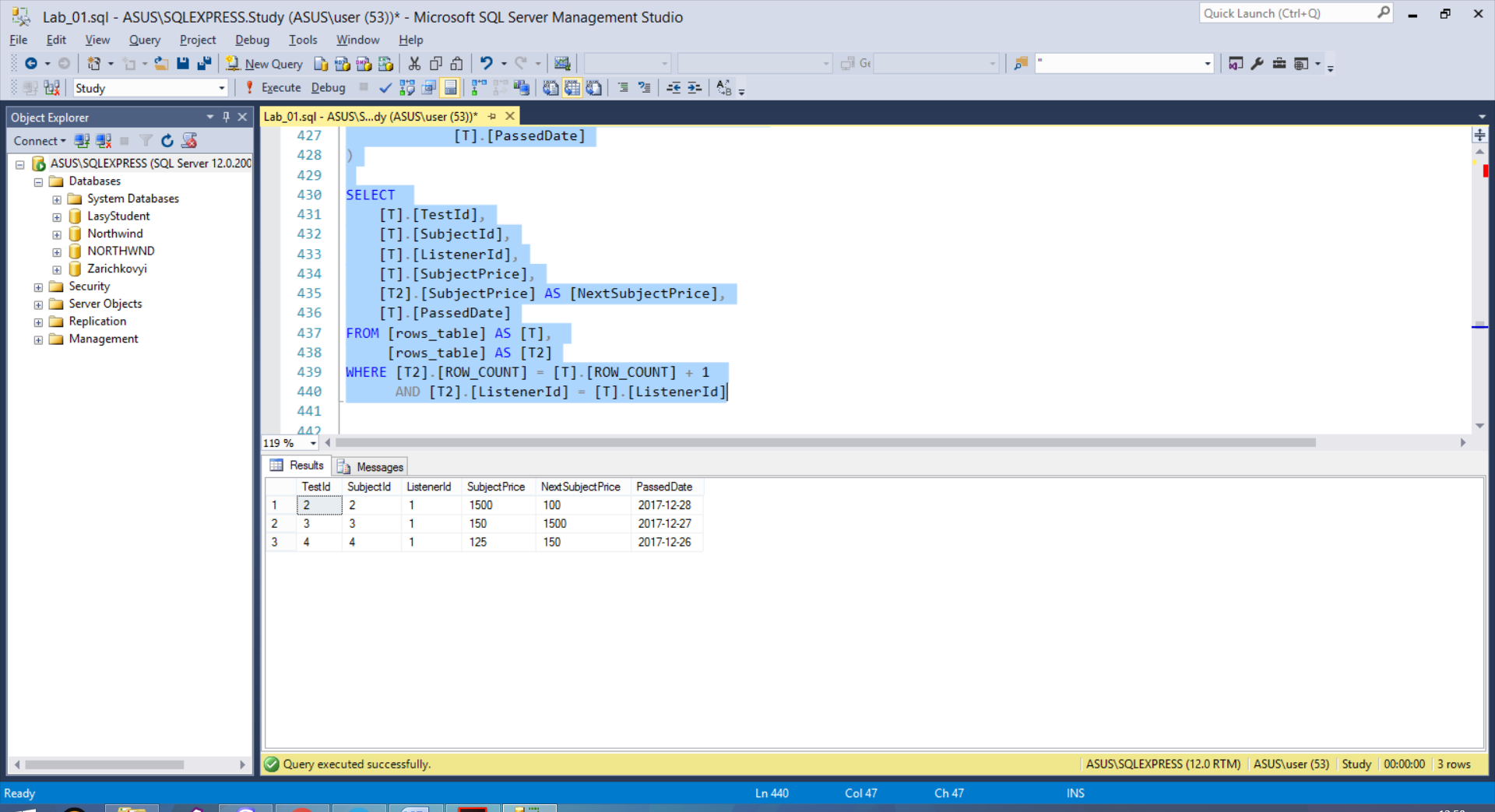
FROM [rows\_table] AS [T],

[rows\_table] AS [T2]

WHERE [T2].[ROW\_COUNT] = [T].[ROW\_COUNT] + 1

AND [T2].[ListenerId] = [T].[ListenerId]

*Результат виконання без використання аналітичних функцій*:



1. Завдання по нормалізації БД:
   1. Змінити таблицю так, щоб вона не була у першій нормальній формі, навести цю таблицю у наповненому вигляді. Потім описати, які зміни треба зробити, щоб вона відповідала першій нормальній формі, навести цю таблицю у наповненому вигляді.

CREATE TABLE [Tests] (

[SubjectName] NVARCHAR(255) NOT NULL,

[ListenerName] NVARCHAR(255) NOT NULL,

[SubjectPrice] INT NOT NULL,

[SubjectDuration] INT NOT NULL,

[TeacherName] NVARCHAR(255) NOT NULL,

[TeacherPhoneNumber] NVARCHAR(255) NOT NULL,

[PassedDate] DATE NULL,

CONSTRAINT PK\_Test PRIMARY KEY CLUSTERED ([SubjectName], [ListenerName]))

INSERT INTO [dbo].[Tests] (

[SubjectName],

[ListenerName],

[SubjectPrice],

[SubjectDuration],

[TeacherName],

[TeacherPhoneNumber],

[PassedDate])

VALUES ('Math', 'Alexander Zarichkovyi', 100, 4, 'Bodnarchyk Semen', '+380123456789,+380987654321', '2017-12-31'),

('Physics', 'Alexander Zarichkovyi', 80, 8, 'QWERTY Semen', '+380123456789', '2017-12-29'),

('C++', 'Alexander Zarichkovyi', 800, 3, 'Mykha Semen', '+380123456789', '2017-12-20'),

('OOP', 'Alexander Zarichkovyi', 1000, 3, 'Kyd Jasper', '+380123456789', '2017-12-01'),

('Math', 'Alexander Onbysh', 100, 4, 'Bodnarchyk Semen', '+380123456789,+380987654321', '2017-06-21'),

('C++', 'Anna Khuda', 800, 3, 'Mykha Irina', '+380123456789', '2017-12-20'),

('OOP', 'Nastya Starchnko', 1000, 3, 'Kyd Jasper', '+380123456789', '2017-12-01')

SELECT \* FROM [dbo].[Tests]

GO

-- 1 NF

-- Зміни: Добавити поле для другого номеру викладача.

IF EXISTS(SELECT \*

FROM [dbo].[Tests])

DROP TABLE [dbo].[Tests]

GO

CREATE TABLE [Tests] (

[SubjectName] NVARCHAR(255) NOT NULL,

[ListenerName] NVARCHAR(255) NOT NULL,

[SubjectPrice] INT NOT NULL,

[SubjectDuration] INT NOT NULL,

[TeacherName] NVARCHAR(255) NOT NULL,

[TeacherPhoneNumber\_01] NVARCHAR(255) NOT NULL,

[TeacherPhoneNumber\_02] NVARCHAR(255) NULL,

[PassedDate] DATE NULL,

CONSTRAINT PK\_Test PRIMARY KEY CLUSTERED ([SubjectName], [ListenerName]))

INSERT INTO [dbo].[Tests] (

[SubjectName],

[ListenerName],

[SubjectPrice],

[SubjectDuration],

[TeacherName],

[TeacherPhoneNumber\_01],

[TeacherPhoneNumber\_02],

[PassedDate])

VALUES ('Math', 'Alexander Zarichkovyi', 100, 4, 'Bodnarchyk Semen', '+380123456789', '+380987654321', '2017-12-31'),

('Physics', 'Alexander Zarichkovyi', 80, 8, 'QWERTY Semen', '+380123456789', NULL, '2017-12-29'),

('C++', 'Alexander Zarichkovyi', 800, 3, 'Mykha Semen', '+380123456789', NULL, '2017-12-20'),

('OOP', 'Alexander Zarichkovyi', 1000, 3, 'Kyd Jasper', '+380123456789', NULL, '2017-12-01'),

('Math', 'Alexander Onbysh', 100, 4, 'Bodnarchyk Semen', '+380123456789', '+380987654321', '2017-06-21'),

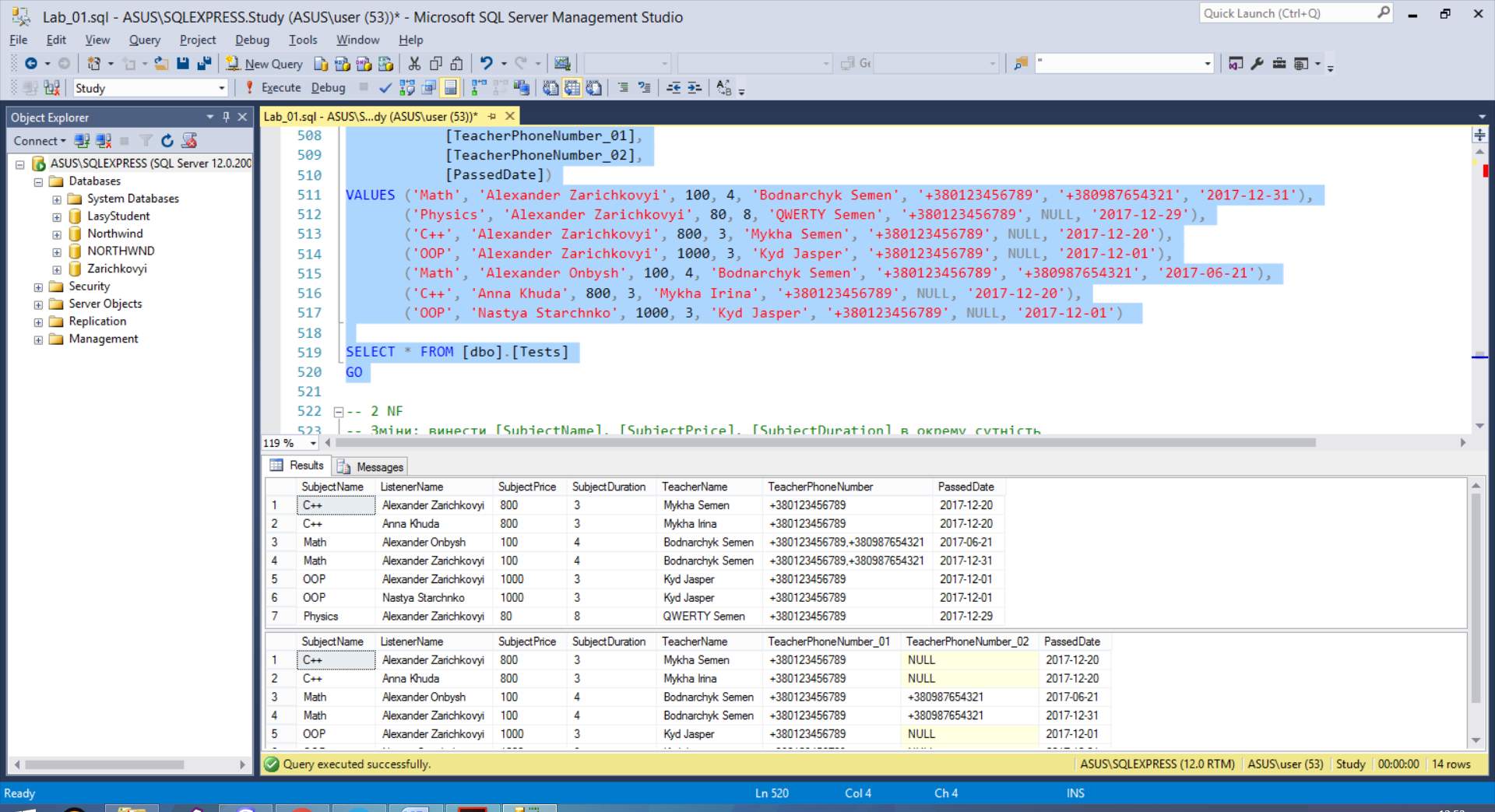
('C++', 'Anna Khuda', 800, 3, 'Mykha Irina', '+380123456789', NULL, '2017-12-20'),

('OOP', 'Nastya Starchnko', 1000, 3, 'Kyd Jasper', '+380123456789', NULL, '2017-12-01')

SELECT \* FROM [dbo].[Tests]

GO

*Результат виконання:*



* 1. Описати, які зміни треба зробити, щоб таблиця відповідала другій нормальній формі, навести цю таблицю у наповненому вигляді*.*

*Текст запиту*:

CREATE PROCEDURE NOT\_PASSED\_STUDENTS

-- 2 NF

-- Зміни: винести [SubjectName], [SubjectPrice], [SubjectDuration] в окрему сутність

IF EXISTS(SELECT \*

FROM [dbo].[Tests])

DROP TABLE [dbo].[Tests]

GO

CREATE TABLE [SubjectDict] (

[ID] INT IDENTITY(1,1) NOT NULL,

[SubjectName] NVARCHAR(255) NOT NULL,

[SubjectPrice] INT NOT NULL,

[SubjectDuration] INT NOT NULL,

CONSTRAINT PK\_Test PRIMARY KEY CLUSTERED ([ID]))

CREATE TABLE [Tests] (

[ListenerName] NVARCHAR(255) NOT NULL,

[SubjectDescript] INT NOT NULL FOREIGN KEY REFERENCES [SubjectDict]([ID]),

[TeacherName] NVARCHAR(255) NOT NULL,

[TeacherPhoneNumber\_01] NVARCHAR(255) NOT NULL,

[TeacherPhoneNumber\_02] NVARCHAR(255) NULL,

[PassedDate] DATE NULL,

CONSTRAINT PK\_Test2 PRIMARY KEY CLUSTERED ([SubjectDescript], [ListenerName]))

INSERT INTO [dbo].[SubjectDict] (

[SubjectName],

[SubjectPrice],

[SubjectDuration])

VALUES ('Math', 100, 4),

('Physics', 80, 8),

('C++', 800, 3),

('OOP', 1000, 3)

INSERT INTO [dbo].[Tests] (

[SubjectDescript],

[ListenerName],

[TeacherName],

[TeacherPhoneNumber\_01],

[TeacherPhoneNumber\_02],

[PassedDate])

VALUES (1, 'Alexander Zarichkovyi', 'Bodnarchyk Semen', '+380123456789', '+380987654321', '2017-12-31'),

(2, 'Alexander Zarichkovyi', 'QWERTY Semen', '+380123456789', NULL, '2017-12-29'),

(3, 'Alexander Zarichkovyi', 'Mykha Semen', '+380123456789', NULL, '2017-12-20'),

(4, 'Alexander Zarichkovyi', 'Kyd Jasper', '+380123456789', NULL, '2017-12-01'),

(1, 'Alexander Onbysh', 'Bodnarchyk Semen', '+380123456789', '+380987654321', '2017-06-21'),

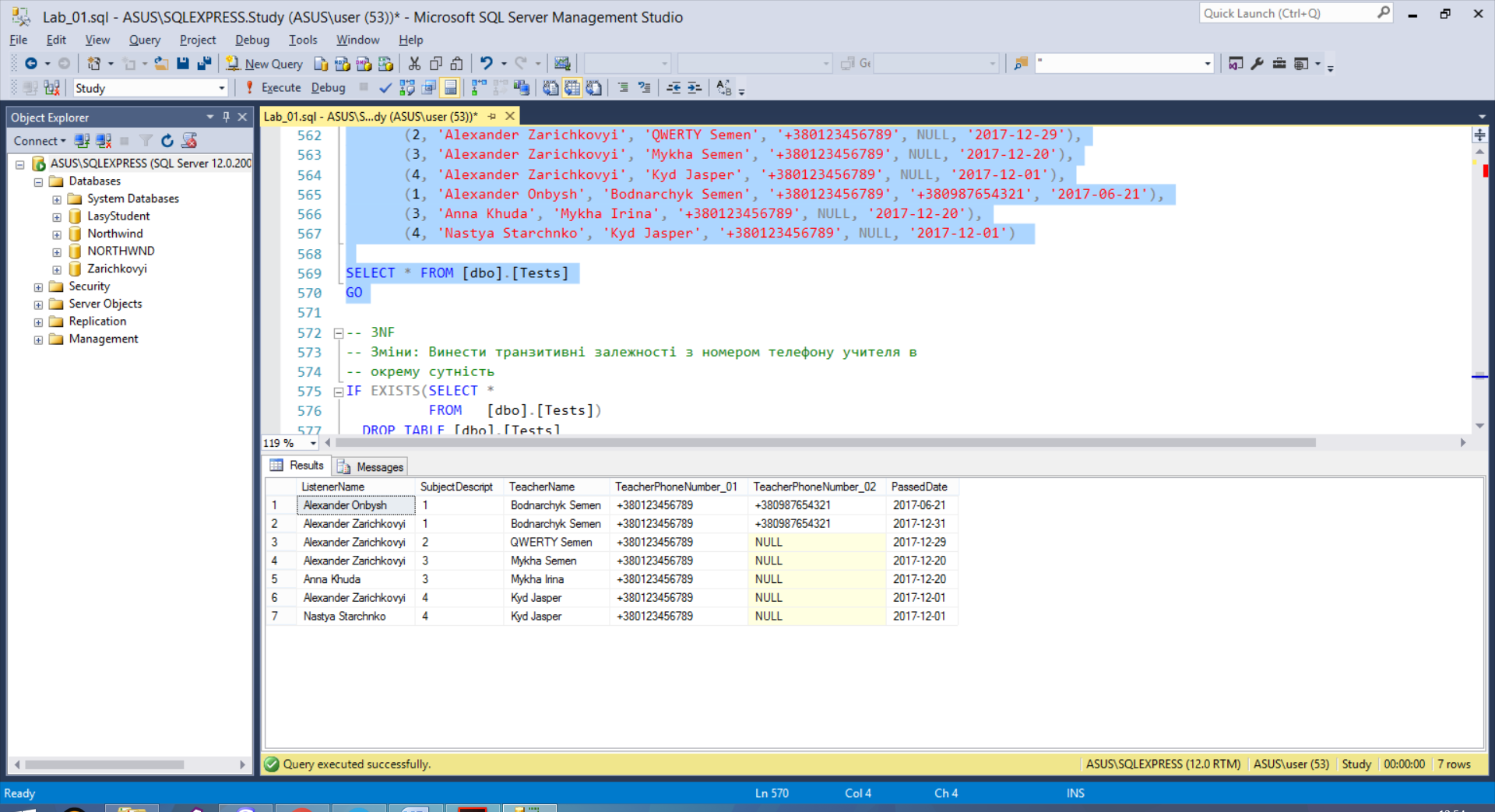
(3, 'Anna Khuda', 'Mykha Irina', '+380123456789', NULL, '2017-12-20'),

(4, 'Nastya Starchnko', 'Kyd Jasper', '+380123456789', NULL, '2017-12-01')

SELECT \* FROM [dbo].[Tests]

GO

*Результат виконання:*



* 1. Описати, які зміни треба зробити, щоб таблиця відповідала третій нормальній формі, навести цю таблицю у наповненому вигляді.

*Текст запиту*:

-- 3NF

-- Зміни: Винести транзитивні залежності з номером телефону учителя в

-- окрему сутність

IF EXISTS(SELECT \*

FROM [dbo].[Tests])

DROP TABLE [dbo].[Tests]

GO

CREATE TABLE [TeachersPhones] (

[ID] INT IDENTITY(1,1) NOT NULL,

[TeacherName] NVARCHAR(255) NOT NULL,

[TeacherPhoneNumber] NVARCHAR(255) NOT NULL,

CONSTRAINT PK\_TeachersPhones PRIMARY KEY CLUSTERED ([ID]))

CREATE TABLE [Tests] (

[ListenerName] NVARCHAR(255) NOT NULL,

[SubjectDescript] INT NOT NULL FOREIGN KEY REFERENCES [SubjectDict]([ID]),

[TeacherName] NVARCHAR(255) NOT NULL,

[PassedDate] DATE NULL,

CONSTRAINT PK\_Test3 PRIMARY KEY CLUSTERED ([SubjectDescript], [ListenerName]))

INSERT INTO [dbo].[TeachersPhones] (

[TeacherName],

[TeacherPhoneNumber])

VALUES ('Bodnarchyk Semen', '+380123456789'),

('Bodnarchyk Semen', '+380987654321'),

('QWERTY Semen', '+380123456789'),

('Mykha Semen', '+380123456789'),

('Kyd Jasper', '+380123456789'),

('Mykha Irina', '+380123456789')

INSERT INTO [dbo].[Tests] (

[SubjectDescript],

[ListenerName],

[TeacherName],

[PassedDate])

VALUES (1, 'Alexander Zarichkovyi', 'Bodnarchyk Semen', '2017-12-31'),

(2, 'Alexander Zarichkovyi', 'QWERTY Semen', '2017-12-29'),

(3, 'Alexander Zarichkovyi', 'Mykha Semen', '2017-12-20'),

(4, 'Alexander Zarichkovyi', 'Kyd Jasper', '2017-12-01'),

(1, 'Alexander Onbysh', 'Bodnarchyk Semen', '2017-06-21'),

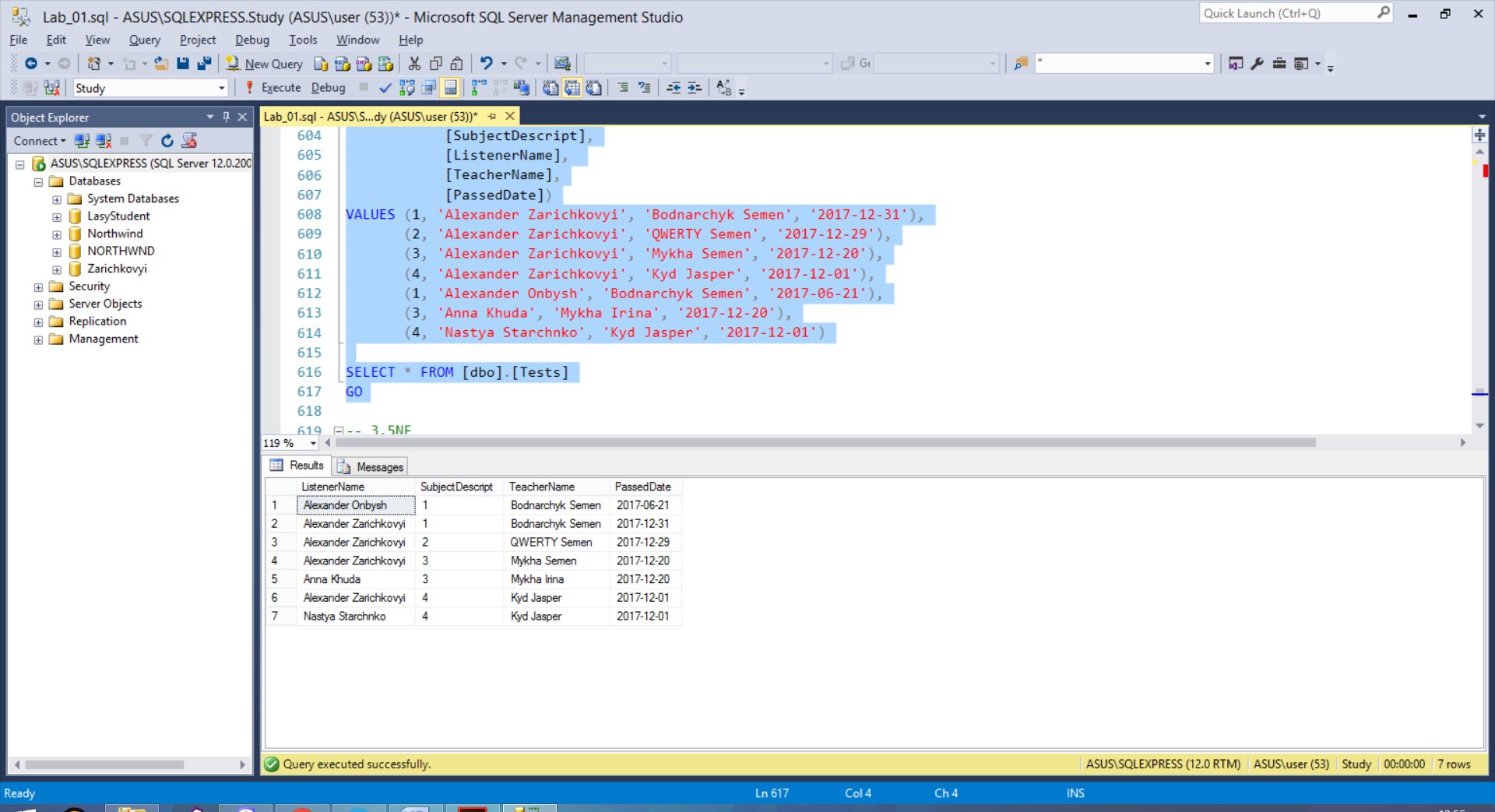
(3, 'Anna Khuda', 'Mykha Irina', '2017-12-20'),

(4, 'Nastya Starchnko', 'Kyd Jasper', '2017-12-01')

SELECT \* FROM [dbo].[Tests]

GO

*Результат виконання:*



* 1. Описати, які зміни треба зробити, щоб таблиця відповідала нормальній формі Бойса-Кода, навести цю таблицю у наповненому вигляді*.*

*Текст запиту*:

-- 3.5NF

-- Зміни: Винести виклада в іншу сутність (залежність по даті та предмету)

IF EXISTS(SELECT \*

FROM [dbo].[Tests])

DROP TABLE [dbo].[Tests]

GO

CREATE TABLE [Teachers](

[ID] INT IDENTITY(1,1) NOT NULL,

[SubjectDescript] INT NOT NULL FOREIGN KEY REFERENCES [SubjectDict]([ID]),

[TeacherName] NVARCHAR(255) NOT NULL

)

CREATE TABLE [Tests] (

[ListenerName] NVARCHAR(255) NOT NULL,

[SubjectDescript] INT NOT NULL FOREIGN KEY REFERENCES [SubjectDict]([ID]),

[PassedDate] DATE NULL,

CONSTRAINT PK\_Test4 PRIMARY KEY CLUSTERED ([SubjectDescript], [ListenerName]))

INSERT INTO [dbo].[Teachers] (

[TeacherName],

[SubjectDescript])

VALUES ('Bodnarchyk Semen', 1),

('QWERTY Semen', 2),

('Mykha Semen', 3),

('Kyd Jasper', 4),

('Mykha Irina', 3)

INSERT INTO [dbo].[Tests] (

[SubjectDescript],

[ListenerName],

[PassedDate])

VALUES (1, 'Alexander Zarichkovyi', '2017-12-31'),

(2, 'Alexander Zarichkovyi', '2017-12-29'),

(3, 'Alexander Zarichkovyi', '2017-12-20'),

(4, 'Alexander Zarichkovyi', '2017-12-01'),

(1, 'Alexander Onbysh', '2017-06-21'),

(3, 'Anna Khuda', '2017-12-20'),

(4, 'Nastya Starchnko', '2017-12-01')

SELECT \* FROM [dbo].[Tests]

GO

*Результат виконання:*

