

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА – Российский технологический университет»**

**РТУ МИРЭА**

Институт кибербезопасности и цифровых технологий

Кафедра КБ-2 «Информационно-аналитические системы кибербезопасности»

Отчет по практической работе № 5

по дисциплине «Безопасность систем баз данных»

«Среда управления SQL Server Management Studio. ЯзыкTransact-SQL. Компоненты SQL»

Выполнил студент 3 курса по специальности (направлению подготовки):

10.05.04 «Информационно-аналитические системы безопасности»

Зубарев Василий Сергеевич

|  |  |
| --- | --- |
|  | Проверил:  старший преподаватель Войтенков Д.В.  Отметка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Дата \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Подпись \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Москва 2025

**Цели работы:**

1. Процедурные расширения
2. Хранимые процедуры
3. Определяемые пользователем функции

В этой работе мы познакомимся с пакетами и подпрограммами.

**Пример 1. Исполнение блока инструкций посредством инструкции if**

use sample\_Zub

IF (SELECT COUNT(\*)

FROM works\_on

WHERE project\_no = 'p1'

GROUP BY project\_no) > 3

BEGIN

PRINT 'The number of employees in the project p1 is 4 or more';

END

ELSE

BEGIN

PRINT 'The following employees work for the project p1';

SELECT e.emp\_fname, e.emp\_lname

FROM employee e

JOIN works\_on w ON e.emp\_no = w.emp\_no

WHERE w.project\_no = 'p1';

END

В примере 1 демонстрируется использование блока инструкций внутри инструкции if. Следующее далее логическое выражение инструкции if

(SELЕCT COUNT(\*)

FROM works\_on

WHERЕ projectno = ’p1’

GROP BY projectno) > 3

возвращает значение true (истина) для базы данных sample. Поэтому будет выполняться инструкция print, входящая в часть инструкции if. Обратите внимание на то обстоятельство, что в этом примере используется подзапрос, чтобы возвратить число строк (посредством агрегатной функции count), удовлетворяющих условию предложения where (project\_no=’p1’).

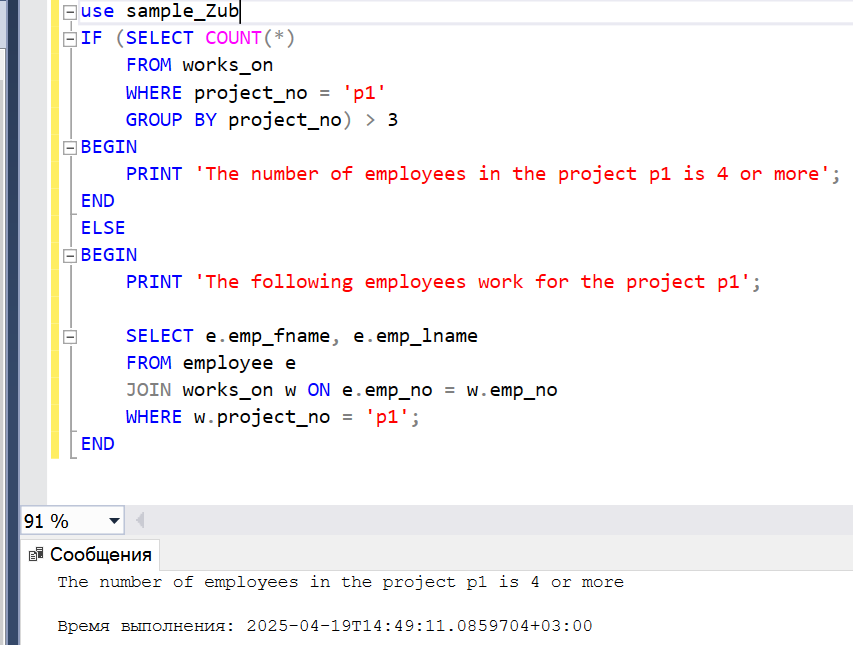


Рисунок 1 - Выполнение примера 1

**Пример 2. Использование инструкции WHILE**

use sample\_Zub

WHILE (SELECT SUM(budget) FROM project) < 500000

BEGIN

UPDATE project SET budget = budget \* 1.1;

IF (SELECT MAX(budget) FROM project) > 240000

BREAK;

ELSE

CONTINUE; -- Необязательная строка

END

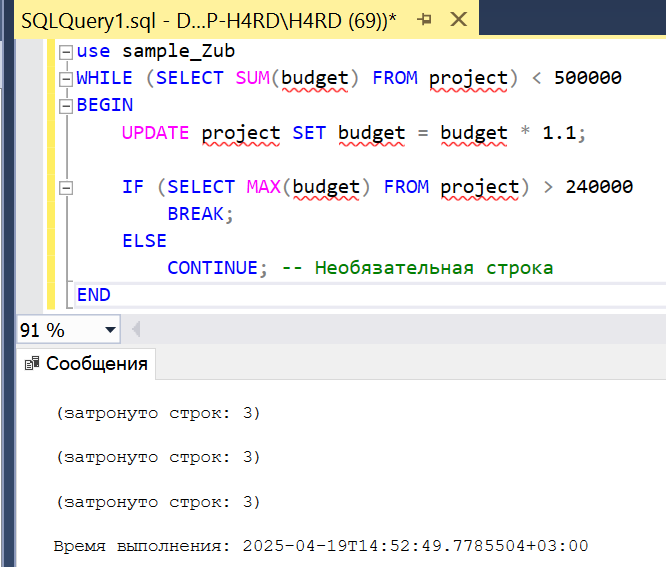


Рисунок 2 - Выполнение примера 2

**Пример 3. Присвоение значения локальным переменным**

use sample\_Zub

DECLARE @avg\_budget MONEY, @extra\_budget MONEY;

SET @extra\_budget = 15000;

SELECT @avg\_budget = AVG(budget)

FROM project;

IF (SELECT budget FROM project WHERE project\_no = 'p1') < @avg\_budget

BEGIN

UPDATE project

SET budget = budget + @extra\_budget

WHERE project\_no = 'p1';

PRINT 'Budget for p1 increased by ' + CAST(@extra\_budget AS VARCHAR(10));

END

ELSE

BEGIN

PRINT 'Budget for p1 unchanged';

END

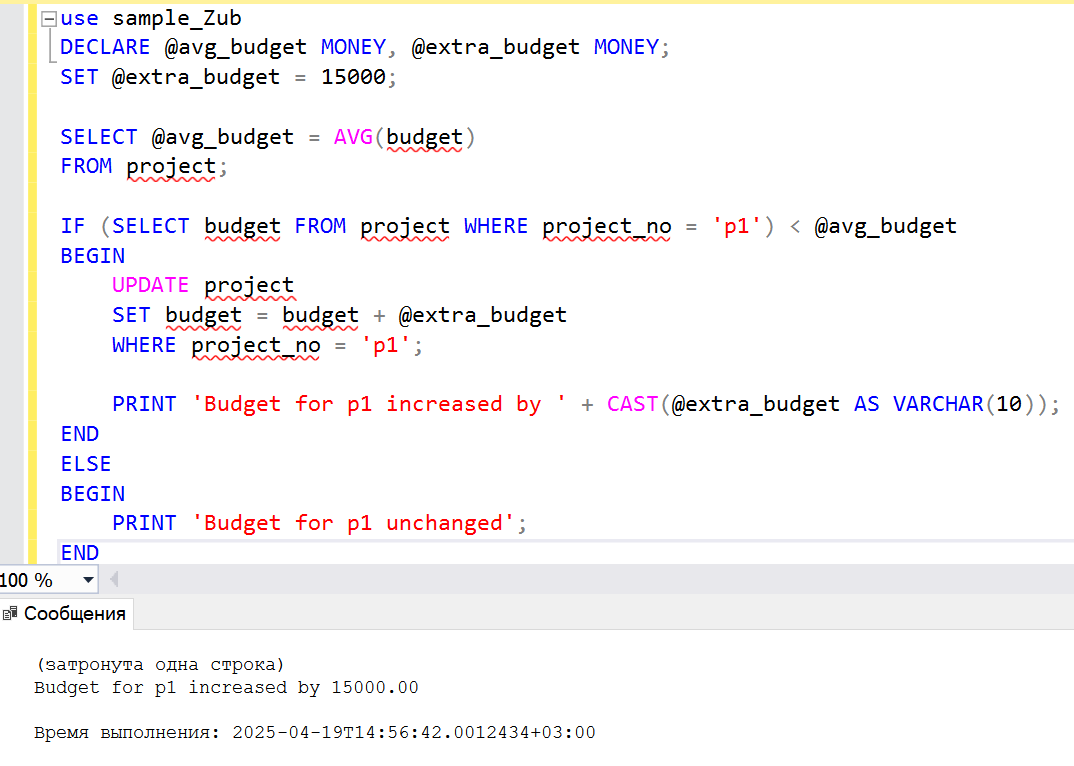


Рисунок 3 - Выполнение примера 3

Пример 4. **Добавление ограничений внешнего ключа.**

Используйте инструкцию ALTER TABLE employee для добавления ограничений внешнего ключа в базе данных sample.

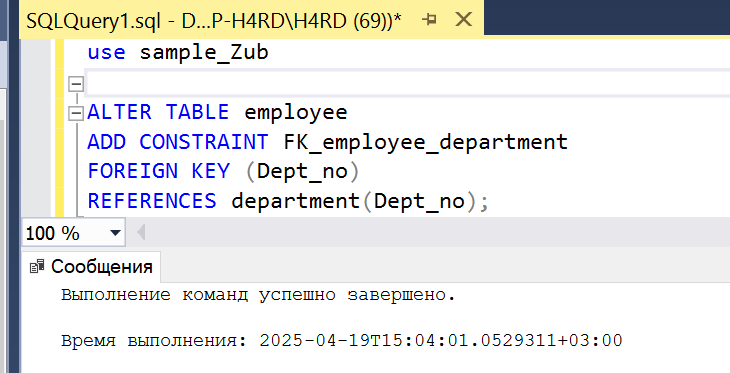


Рисунок 4 - Выполнение задания 4

**Пример 5 Применение инструкций TRY, CATCH и THROW для обработки исключений**

use sample\_Zub

BEGIN TRY

BEGIN TRANSACTION;

INSERT INTO employee VALUES (11111, 'Ann', 'Smith', 'd2');

INSERT INTO employee VALUES (22222, 'Matthew', 'Jones', 'd4');

INSERT INTO employee VALUES (33333, 'John', 'Barrimore', 'd2');

COMMIT TRANSACTION;

PRINT 'Transaction committed';

END TRY

BEGIN CATCH

IF @@TRANCOUNT > 0

ROLLBACK TRANSACTION;

PRINT 'Transaction rolled back';

THROW;

END CATCH

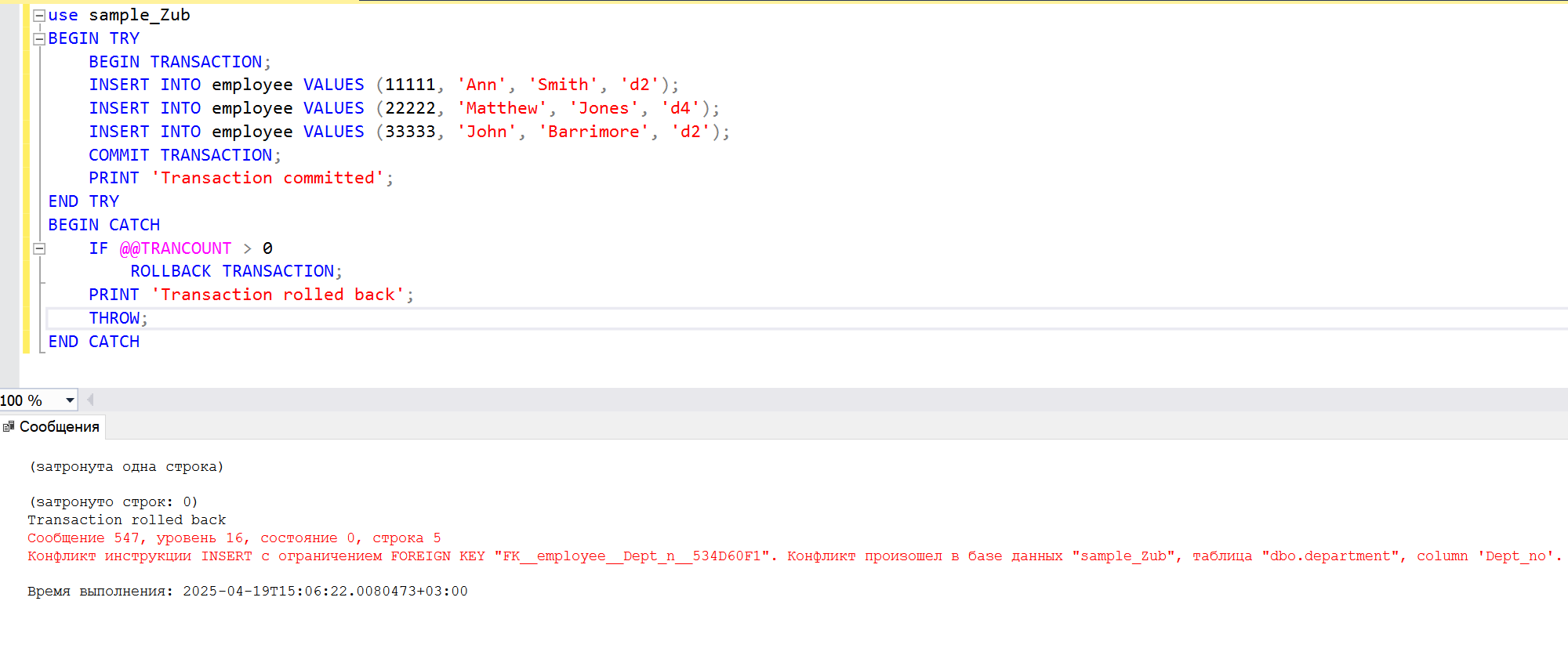


Рисунок 5 - Выполнение задания 5

**Пример 6. Создание хранимой процедуры для изменения данных таблицы**

use sample\_Zub

GO

CREATE PROCEDURE increase\_budget

@percent INT = 5

AS

BEGIN

SET NOCOUNT ON;

UPDATE project

SET budget = budget + (budget \* @percent / 100.0);

PRINT CONCAT('All project budgets increased by ', @percent, '%');

END;

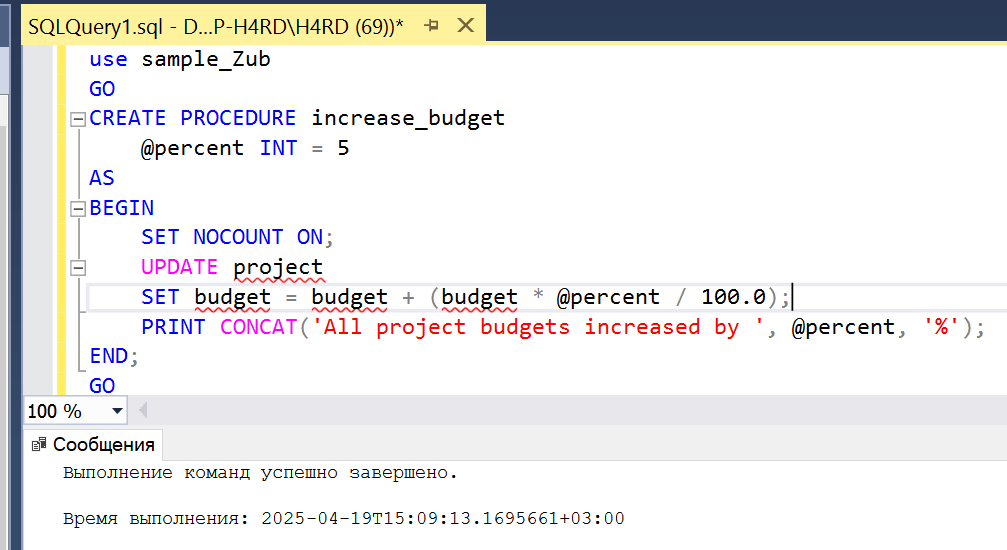
GO  


Рисунок 6 - Выполнение примера 6

**Пример 7. Применение инструкции EXECUTE** ;

USE sample\_Zub;

GO

EXEC increase\_budget 10;

Инструкция execute в примере 7 выполняет хранимую процедуру increase\_budget (см. Пример 6), которая увеличивает бюджете всех проектов на 10%.

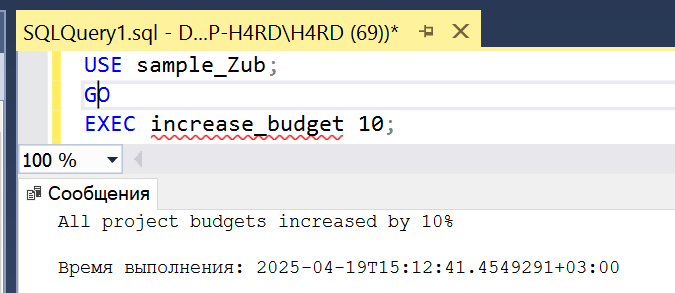


Рисунок 7 - Выполнение задания 7

**Пример 8. Создание хранимой процедуры для обработки таблиц EMPLOYEE и WORKS ON**

USE sample\_Zub;

GO

CREATE PROCEDURE modify\_empno

@old\_no INT,

@new\_no INT

AS

BEGIN

SET NOCOUNT ON;

BEGIN TRY

BEGIN TRANSACTION;

IF NOT EXISTS (SELECT 1 FROM employee WHERE emp\_no = @old\_no)

BEGIN

RAISERROR('Сотрудник с номером %d не найден', 16, 1, @old\_no);

RETURN;

END

IF EXISTS (SELECT 1 FROM employee WHERE emp\_no = @new\_no)

BEGIN

RAISERROR('Номер %d уже занят другим сотрудником', 16, 1, @new\_no);

RETURN;

END

UPDATE employee

SET emp\_no = @new\_no

WHERE emp\_no = @old\_no;

UPDATE works\_on

SET emp\_no = @new\_no

WHERE emp\_no = @old\_no;

COMMIT TRANSACTION;

PRINT CONCAT('Номер сотрудника успешно изменен с ', @old\_no, ' на ', @new\_no);

END TRY

BEGIN CATCH

IF @@TRANCOUNT > 0

ROLLBACK TRANSACTION;

PRINT 'Ошибка при изменении номера сотрудника: ' + ERROR\_MESSAGE();

THROW;

END CATCH

END;

GO

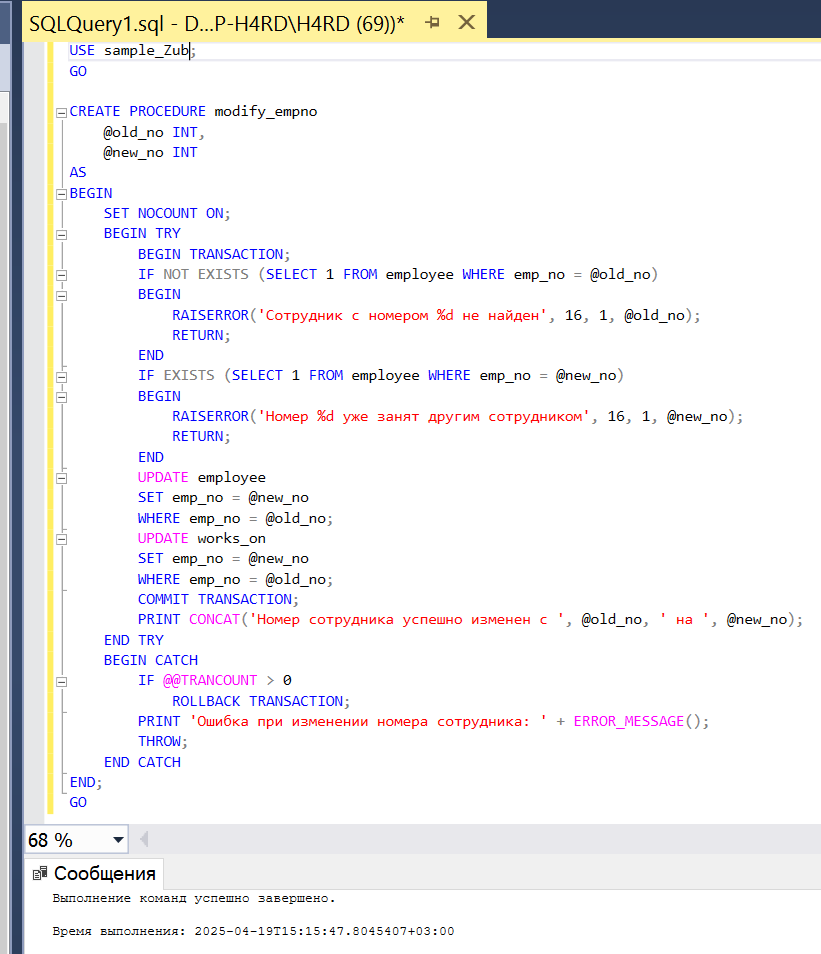


Рисунок 8 - Выполнение задания 8

**Пример 9. Использование в хранимой процедуре предложения OUTPUT**

USE sample\_Zub;

GO

CREATE PROCEDURE delete\_emp

@employee\_no INT,

@counter INT OUTPUT

AS

BEGIN

SET NOCOUNT ON;

SELECT @counter = COUNT(\*)

FROM works\_on

WHERE emp\_no = @employee\_no;

DELETE FROM works\_on

WHERE emp\_no = @employee\_no;

DELETE FROM employee

WHERE emp\_no = @employee\_no;

PRINT CONCAT('Сотрудник ', @employee\_no, ' удален. Удалено ', @counter, ' записей в works\_on.');

END;

GO

-- Пример вызова процедуры

DECLARE @quantity INT;

EXECUTE delete\_emp

@employee\_no = 28559,

@counter = @quantity OUTPUT;

PRINT 'Всего удалено записей: ' + CAST(@quantity AS VARCHAR(10));

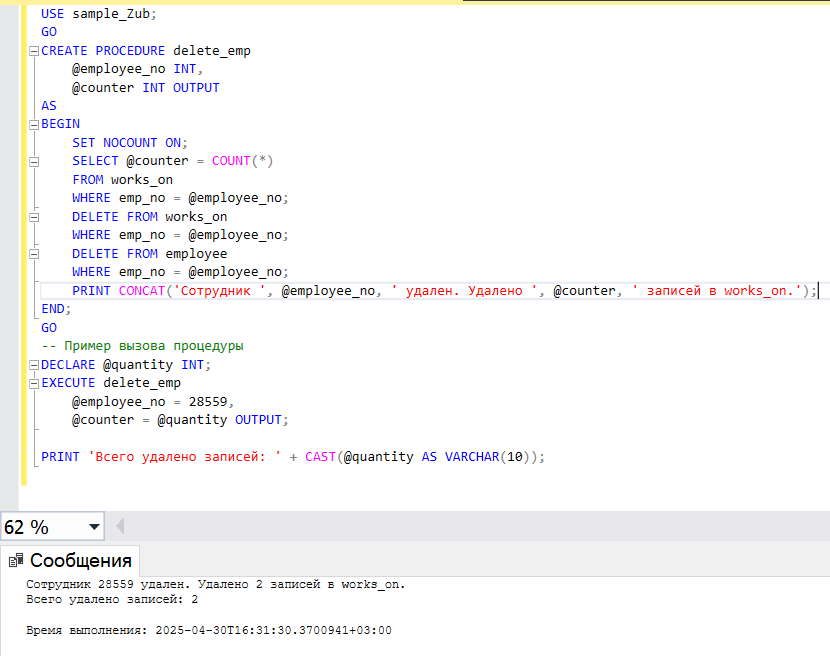


Рисунок 9 - выполнение задания 9

**Пример 10. Применение инструкции EXECUTE с предложением WITH RESULT SETS**

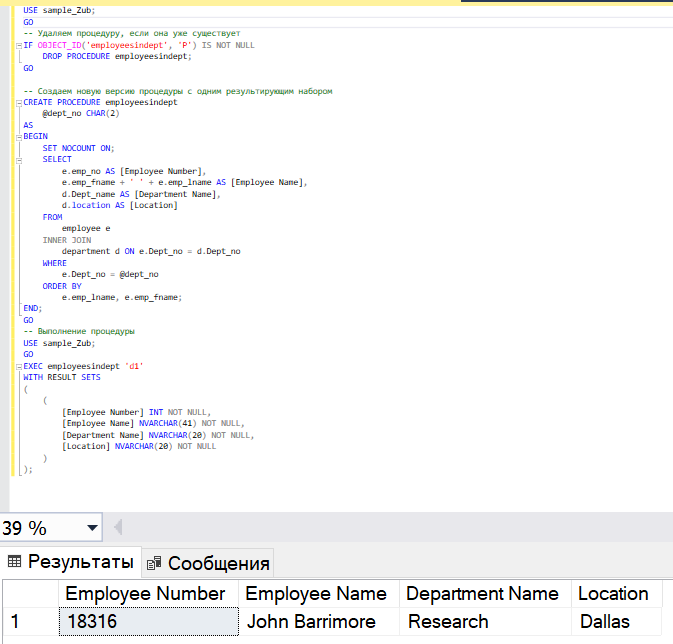


Рисунок 10 - выполнение задания 10

USE sample\_Zub;

GO

-- Удаляем процедуру, если она уже существует

IF OBJECT\_ID('employeesindept', 'P') IS NOT NULL

DROP PROCEDURE employeesindept;

GO

-- Создаем новую версию процедуры с одним результирующим набором

CREATE PROCEDURE employeesindept

@dept\_no CHAR(2)

AS

BEGIN

SET NOCOUNT ON;

SELECT

e.emp\_no AS [Employee Number],

e.emp\_fname + ' ' + e.emp\_lname AS [Employee Name],

d.Dept\_name AS [Department Name],

d.location AS [Location]

FROM

employee e

INNER JOIN

department d ON e.Dept\_no = d.Dept\_no

WHERE

e.Dept\_no = @dept\_no

ORDER BY

e.emp\_lname, e.emp\_fname;

END;

GO

-- Выполнение процедуры

USE sample\_Zub;

GO

EXEC employeesindept 'd1'

WITH RESULT SETS

(

(

[Employee Number] INT NOT NULL,

[Employee Name] NVARCHAR(41) NOT NULL,

[Department Name] NVARCHAR(20) NOT NULL,

[Location] NVARCHAR(20) NOT NULL

)

);

**Пример 11. Создание определяемой пользователем функции**

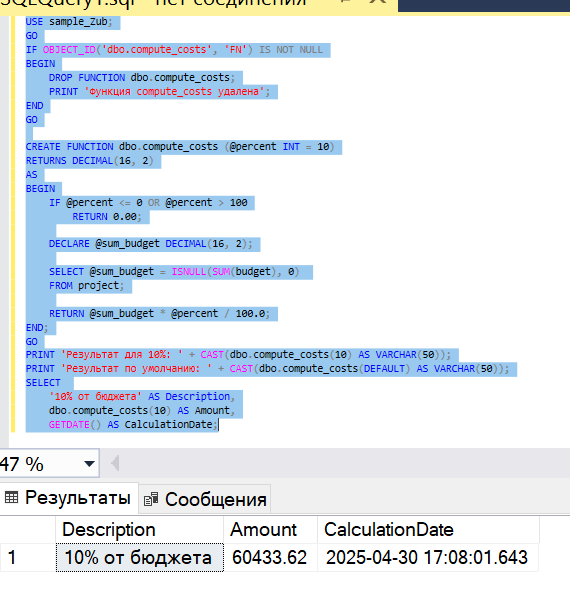
* Эта функция вычисляет возникающие дополнительные обшие затраты,
* при увеличении бюджетов проектов
* USE sample\_Zub;
* GO
* IF OBJECT\_ID('dbo.compute\_costs', 'FN') IS NOT NULL
* BEGIN
* DROP FUNCTION dbo.compute\_costs;
* PRINT 'Функция compute\_costs удалена';
* END
* GO
* CREATE FUNCTION dbo.compute\_costs (@percent INT = 10)
* RETURNS DECIMAL(16, 2)
* AS
* BEGIN
* IF @percent <= 0 OR @percent > 100
* RETURN 0.00;
* DECLARE @sum\_budget DECIMAL(16, 2);
* SELECT @sum\_budget = ISNULL(SUM(budget), 0)
* FROM project;
* RETURN @sum\_budget \* @percent / 100.0;
* END;
* GO
* PRINT 'Результат для 10%: ' + CAST(dbo.compute\_costs(10) AS VARCHAR(50));
* PRINT 'Результат по умолчанию: ' + CAST(dbo.compute\_costs(DEFAULT) AS VARCHAR(50));
* SELECT
* '10% от бюджета' AS Description,
* dbo.compute\_costs(10) AS Amount,
* GETDATE() AS CalculationDate;
* 

Рисунок 11 - выполнение задания 11

* **Пример 12. Вызов ОПФ в инструкции select**

USE sample\_Zub;

GO

SELECT

p.project\_no AS [Project Number],

p.project\_name AS [Project Name],

p.budget AS [Current Budget],

dbo.compute\_costs(25) AS [25% of Total Budget]

FROM

project p

WHERE

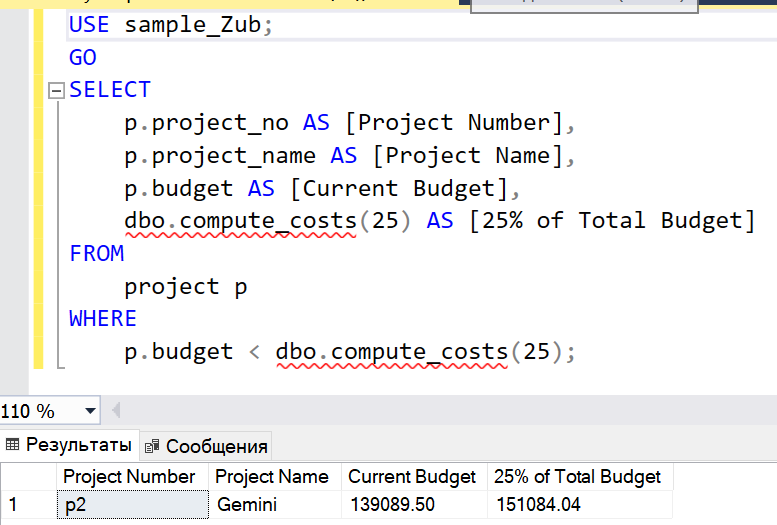
* p.budget < dbo.compute\_costs(25);
* 

Рисунок 12 – выполнение задания 12

**Пример 13. Создание возвращающей табличное значение функции**

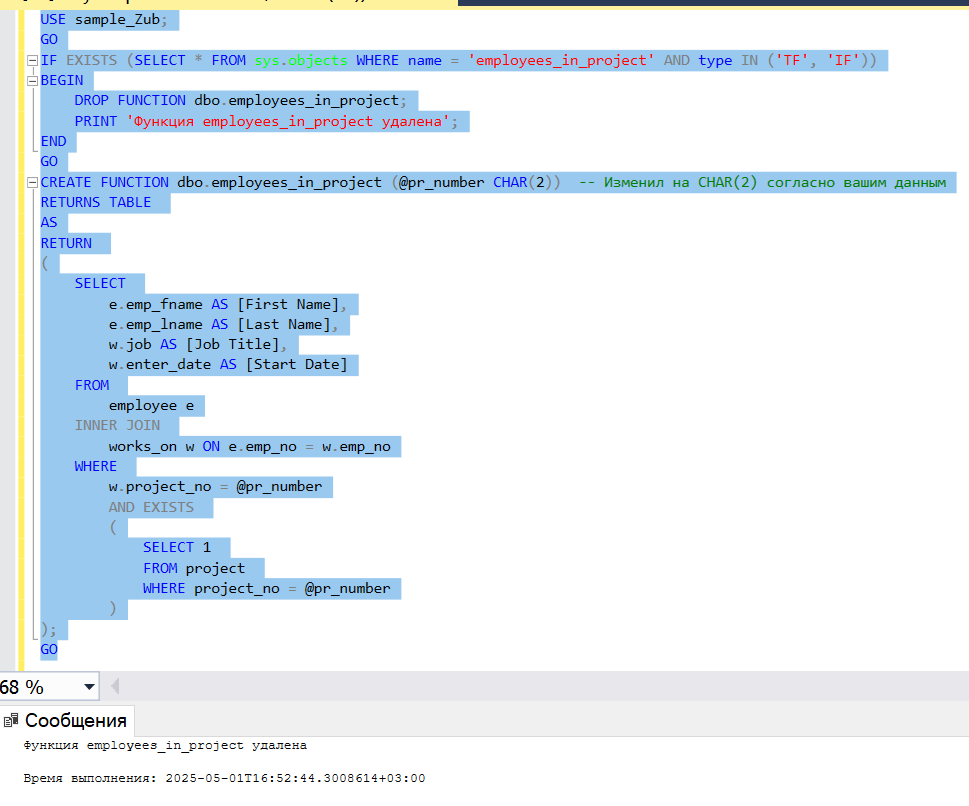


Рисунок 13 - выполнение задания 13

USE sample\_Zub;

GO

IF EXISTS (SELECT \* FROM sys.objects WHERE name = 'employees\_in\_project' AND type IN ('TF', 'IF'))

BEGIN

DROP FUNCTION dbo.employees\_in\_project;

PRINT 'Функция employees\_in\_project удалена';

END

GO

CREATE FUNCTION dbo.employees\_in\_project (@pr\_number CHAR(2)) -- Изменил на CHAR(2) согласно вашим данным

RETURNS TABLE

AS

RETURN

(

SELECT

e.emp\_fname AS [First Name],

e.emp\_lname AS [Last Name],

w.job AS [Job Title],

w.enter\_date AS [Start Date]

FROM

employee e

INNER JOIN

works\_on w ON e.emp\_no = w.emp\_no

WHERE

w.project\_no = @pr\_number

AND EXISTS

(

SELECT 1

FROM project

WHERE project\_no = @pr\_number

)

);

GO

**Пример 14. Использование возвращающей табличное значение функции**

SELECT \* FROM dbo.employees\_in\_project('p3');

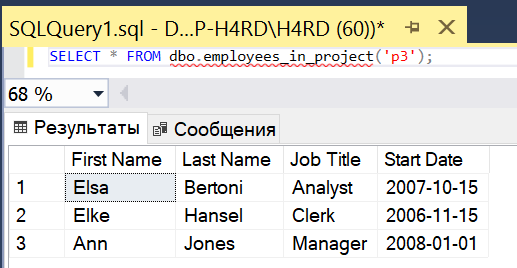


Рисунок 14 - выполнение задания 14

**Пример 15. Создание возвращающей табличное значение функции**

USE sample\_Zub;

GO

IF OBJECT\_ID('dbo.fn\_getjob', 'TF') IS NOT NULL

DROP FUNCTION dbo.fn\_getjob;

GO

CREATE FUNCTION dbo.fn\_getjob (@empid INT)

RETURNS TABLE

AS

RETURN

(

SELECT

job

FROM

works\_on

WHERE

emp\_no = @empid

AND job IS NOT NULL

AND project\_no = 'p1'

);

GO

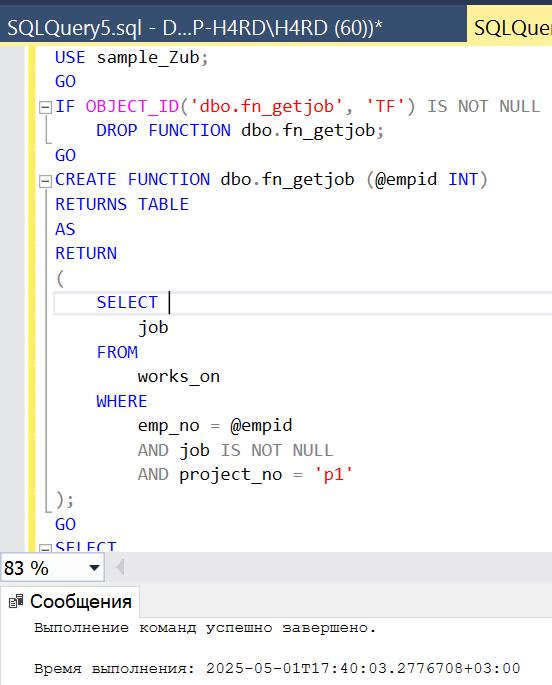


Рисунок 15 - выполнение задания 15

**Пример 16. "Соединение" двух таблиц посредством предложения APPLY**

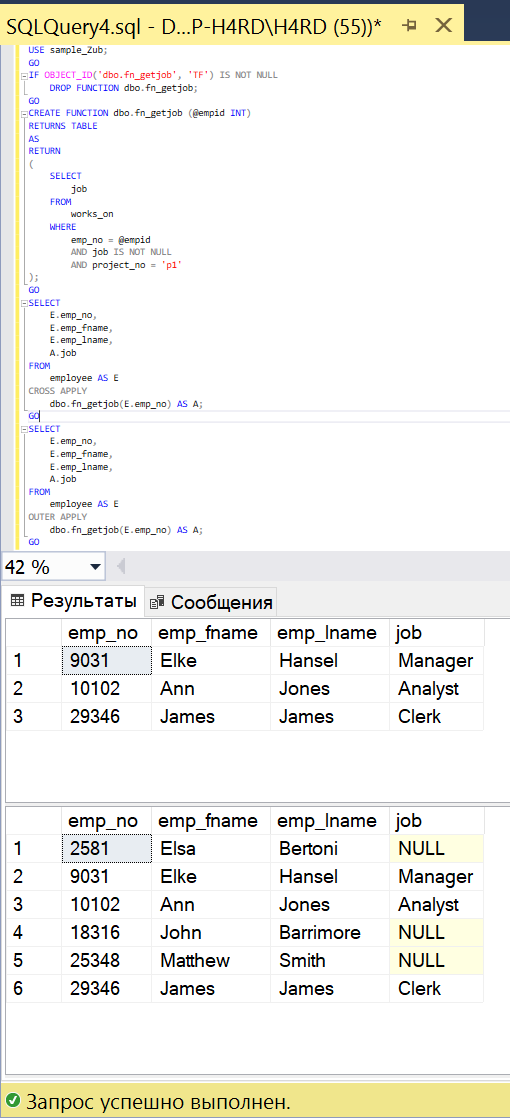


Рисунок 16 - выполнение задания 16

USE sample\_Zub;

GO

IF OBJECT\_ID('dbo.fn\_getjob', 'TF') IS NOT NULL

DROP FUNCTION dbo.fn\_getjob;

GO

CREATE FUNCTION dbo.fn\_getjob (@empid INT)

RETURNS TABLE

AS

RETURN

(

SELECT

job

FROM

works\_on

WHERE

emp\_no = @empid

AND job IS NOT NULL

AND project\_no = 'p1'

);

GO

SELECT

E.emp\_no,

E.emp\_fname,

E.emp\_lname,

A.job

FROM

employee AS E

CROSS APPLY

dbo.fn\_getjob(E.emp\_no) AS A;

GO

SELECT

E.emp\_no,

E.emp\_fname,

E.emp\_lname,

A.job

FROM

employee AS E

OUTER APPLY

dbo.fn\_getjob(E.emp\_no) AS A;

GO

**Пример 24. Применение возвращающего табличное значение параметра**

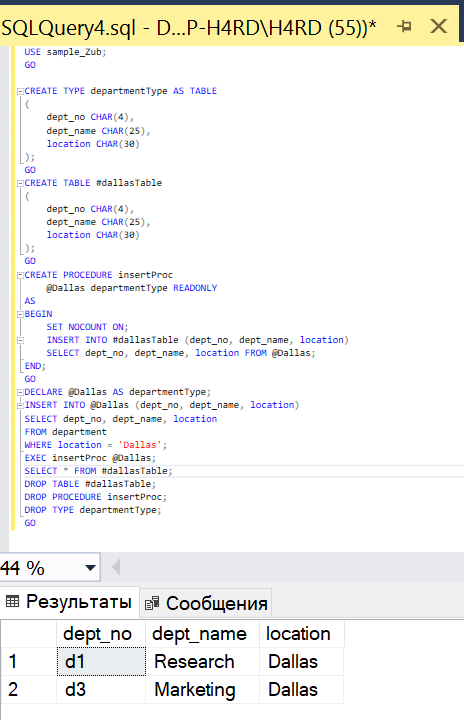


Рисунок 17 - выполнение упражнения 24

USE sample\_Zub;

GO

CREATE TYPE departmentType AS TABLE

(

dept\_no CHAR(4),

dept\_name CHAR(25),

location CHAR(30)

);

GO

CREATE TABLE #dallasTable

(

dept\_no CHAR(4),

dept\_name CHAR(25),

location CHAR(30)

);

GO

CREATE PROCEDURE insertProc

@Dallas departmentType READONLY

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO #dallasTable (dept\_no, dept\_name, location)

SELECT dept\_no, dept\_name, location FROM @Dallas;

END;

GO

DECLARE @Dallas AS departmentType;

INSERT INTO @Dallas (dept\_no, dept\_name, location)

SELECT dept\_no, dept\_name, location

FROM department

WHERE location = 'Dallas';

EXEC insertProc @Dallas;

SELECT \* FROM #dallasTable;

DROP TABLE #dallasTable;

DROP PROCEDURE insertProc;

DROP TYPE departmentType;

GO

**Упражнения**

**Упражнение 1**

Создайте пакет для вставки 3000 строк в таблицу employee. Значения столбца emp\_no должны быть однозначными в диапазоне от 1 до 3000. Всем ячейкам столбцов emp\_lname, emp\_fname и dept\_no присваиваются значения "Jane", "Smith" и "d1" соответственно.



Рисунок 18 - Выполнение упражнения 1

Текст ошибки

Ошибка при вставке данных: Нарушено "PK\_\_employee\_\_263E648F18BF2DAE" ограничения PRIMARY KEY. Не удается вставить повторяющийся ключ в объект "dbo.employee". Повторяющееся значение ключа: (2581).

Текст скрипта

USE sample\_Zub;

GO

BEGIN TRANSACTION;

BEGIN TRY

-- Вставка 3000 строк

WITH Numbers AS (

SELECT TOP 3000

ROW\_NUMBER() OVER (ORDER BY (SELECT NULL)) AS n

FROM master.dbo.spt\_values v1

CROSS JOIN master.dbo.spt\_values v2

)

INSERT INTO employee (emp\_no, emp\_fname, emp\_lname, dept\_no)

SELECT

n AS emp\_no,

'Jane' AS emp\_fname,

'Smith' AS emp\_lname,

'd1' AS dept\_no

FROM Numbers;

COMMIT TRANSACTION;

PRINT 'Успешно добавлено 3000 строк в таблицу employee';

END TRY

BEGIN CATCH

ROLLBACK TRANSACTION;

PRINT 'Ошибка при вставке данных: ' + ERROR\_MESSAGE();

END CATCH;

GO

-- Проверка количества строк

SELECT COUNT(\*) AS TotalEmployees FROM employee;

GO

**Упражнение 2**

Измените пакет из упражнения 1 таким образом, чтобы генерировать случайные значения для столбца emp\_no, используя функцию rand. (Подсказка: для получения случайных значений используйте системные функции datepart и GETDATE.)

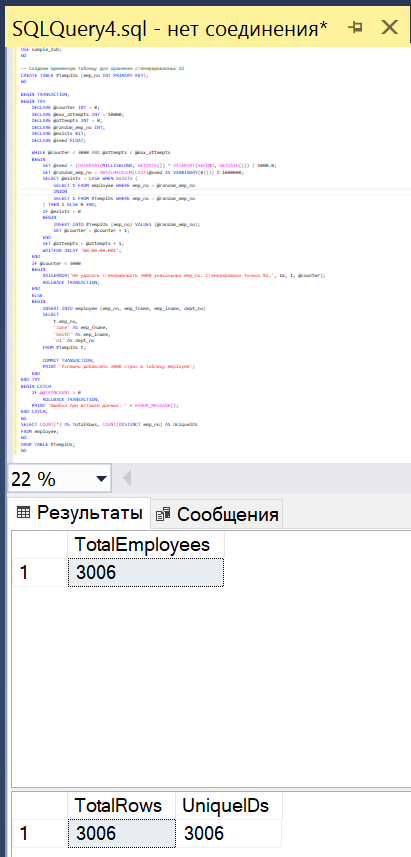


Рисунок 19 - Выполнение упражнения 2

Код запроса  
USE sample\_Zub;

GO

-- Создаем временную таблицу для хранения сгенерированных ID

CREATE TABLE #TempIDs (emp\_no INT PRIMARY KEY);

GO

BEGIN TRANSACTION;

BEGIN TRY

DECLARE @counter INT = 0;

DECLARE @max\_attempts INT = 10000;

DECLARE @attempts INT = 0;

DECLARE @random\_emp\_no INT;

DECLARE @exists BIT;

DECLARE @seed FLOAT;

WHILE @counter < 3000 AND @attempts < @max\_attempts

BEGIN

SET @seed = (DATEPART(MILLISECOND, GETDATE()) \* DATEPART(SECOND, GETDATE())) / 1000.0;

SET @random\_emp\_no = ABS(CHECKSUM(CAST(@seed AS VARBINARY(8)))) % 1000000;

SELECT @exists = CASE WHEN EXISTS (

SELECT 1 FROM employee WHERE emp\_no = @random\_emp\_no

UNION

SELECT 1 FROM #TempIDs WHERE emp\_no = @random\_emp\_no

) THEN 1 ELSE 0 END;

IF @exists = 0

BEGIN

INSERT INTO #TempIDs (emp\_no) VALUES (@random\_emp\_no);

SET @counter = @counter + 1;

END

SET @attempts = @attempts + 1;

WAITFOR DELAY '00:00:00.001';

END

IF @counter < 3000

BEGIN

RAISERROR('Не удалось сгенерировать 3000 уникальных emp\_no. Сгенерировано только %d.', 16, 1, @counter);

ROLLBACK TRANSACTION;

END

ELSE

BEGIN

INSERT INTO employee (emp\_no, emp\_fname, emp\_lname, dept\_no)

SELECT

t.emp\_no,

'Jane' AS emp\_fname,

'Smith' AS emp\_lname,

'd1' AS dept\_no

FROM #TempIDs t;

COMMIT TRANSACTION;

PRINT 'Успешно добавлено 3000 строк в таблицу employee';

END

END TRY

BEGIN CATCH

IF @@TRANCOUNT > 0

ROLLBACK TRANSACTION;

PRINT 'Ошибка при вставке данных: ' + ERROR\_MESSAGE();

END CATCH;

GO

SELECT COUNT(\*) AS TotalRows, COUNT(DISTINCT emp\_no) AS UniqueIDs

FROM employee;

GO

DROP TABLE #TempIDs;

GO