

## Funkcje użytkownika

(\*) – zadania, które bazują na poprzednim rozwiązaniu

Kwoty proszę zaokrągać do 2 miejsc po przecinku. Załóż, że uwzględniamy tylko rekordy, które mają wszystkie dostępne informacje – nie ma potrzeby stosowania połączeń zewnętrznych (chyba, że w zadaniu jest powiedziane inaczej).

### Zadanie 1.

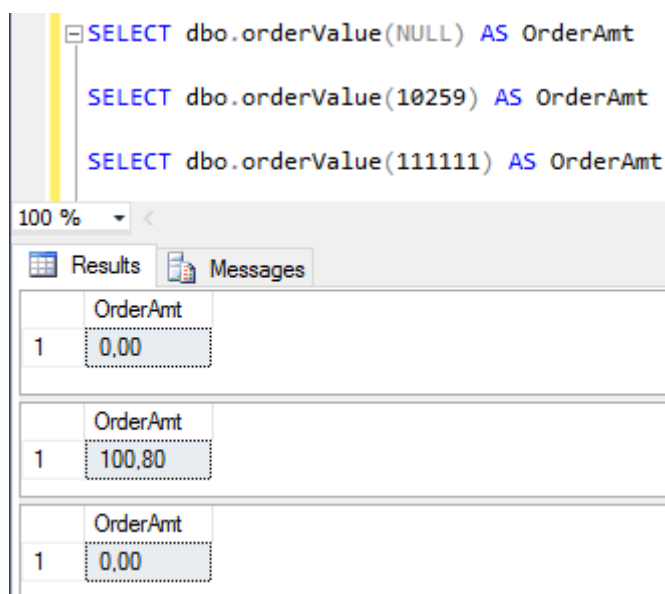
Zaprojektuj funkcję **orderValue**, która korzystając z tabeli **Order Details** dla danego identyfikatora zamówienia zwróci wartość danego zamówienia (bez uwzględnienia zniżki). W przypadku gdy podane zamówienie nie istnieje, to funkcja powinna zwrócić **0,00**.

Oczekiwany rezultat:

```
SELECT dbo.orderValue(NULL) AS OrderAmt
```

```
SELECT dbo.orderValue(10259) AS OrderAmt
```

```
SELECT dbo.orderValue(111111) AS OrderAmt
```



The screenshot shows a SQL query window with three queries executed. Below the queries, the 'Results' tab is active, displaying a table grid with three rows of results. Each row has two columns: an implicit row number and 'OrderAmt'.

	OrderAmt
1	0,00
1	100,80
1	0,00

### Zadanie 2.(\*)

Korzystając z tabeli **Orders** i funkcji stworzonej w poprzednim zapytaniu, zaprojektuj zapytanie, które zwróci identyfikator zamówienia, identyfikator klienta, datę zamówienia oraz wartość zamówienia. Wynik posortuj po identyfikatorze zamówienia (rosnąco).

Oczekiwany rezultat (częściowy; liczba wszystkich zwróconych rekordów: 830):

	OrderID	CustomerID	OrderDate	OrderAmt
1	10248	VINET	1996-07-04 00:00:00.000	440,00
2	10249	TOMSP	1996-07-05 00:00:00.000	1863,40
3	10250	HANAR	1996-07-08 00:00:00.000	1813,00
4	10251	VICTE	1996-07-08 00:00:00.000	670,80
5	10252	SUPRD	1996-07-09 00:00:00.000	3730,00
6	10253	HANAR	1996-07-10 00:00:00.000	1444,80
7	10254	CHOPS	1996-07-11 00:00:00.000	625,20
8	10255	RICSU	1996-07-12 00:00:00.000	2490,50
9	10256	WELLI	1996-07-15 00:00:00.000	517,80
10	10257	HILAA	1996-07-16 00:00:00.000	1119,90
11	10258	ERNSH	1996-07-17 00:00:00.000	2018,60
12	10259	CENTC	1996-07-18 00:00:00.000	100,80
13	10260	OTTIK	1996-07-19 00:00:00.000	1746,20
14	10261	QUEDE	1996-07-19 00:00:00.000	448,00
15	10262	RATTC	1996-07-22 00:00:00.000	624,80

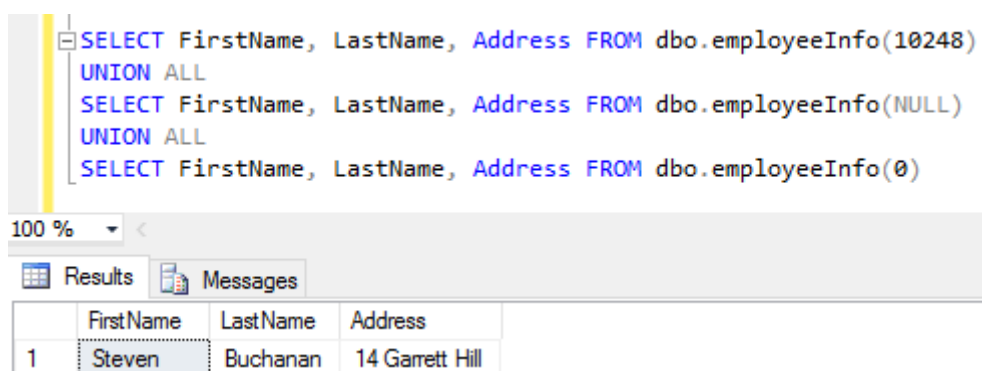
### Zadanie 3.

Korzystając z tabel **Orders** oraz **Employees** zaprojektuj funkcję **employeeInfo**, która dla podanego **OrderID** (parametr) zwróci imię, nazwisko oraz adres pracownika realizującego dane zamówienie. Jeżeli podany numer zamówienia nie istnieje, funkcja nie powinna zwracać żadnego rekordu.

Po rozwiązaniu zapisz funkcję, tak aby można było ją wykorzystać później.

Przykładowy rezultat:

```
SELECT FirstName, LastName, Address FROM dbo.employeeInfo(10248)
UNION ALL
SELECT FirstName, LastName, Address FROM dbo.employeeInfo(NULL)
UNION ALL
SELECT FirstName, LastName, Address FROM dbo.employeeInfo(0)
```



The screenshot shows a SQL query window with the following text:

```
SELECT FirstName, LastName, Address FROM dbo.employeeInfo(10248)
UNION ALL
SELECT FirstName, LastName, Address FROM dbo.employeeInfo(NULL)
UNION ALL
SELECT FirstName, LastName, Address FROM dbo.employeeInfo(0)
```

Below the query window, the 'Results' tab is active, displaying a table with the following data:

	FirstName	LastName	Address
1	Steven	Buchanan	14 Garrett Hill

### Zadanie 4.

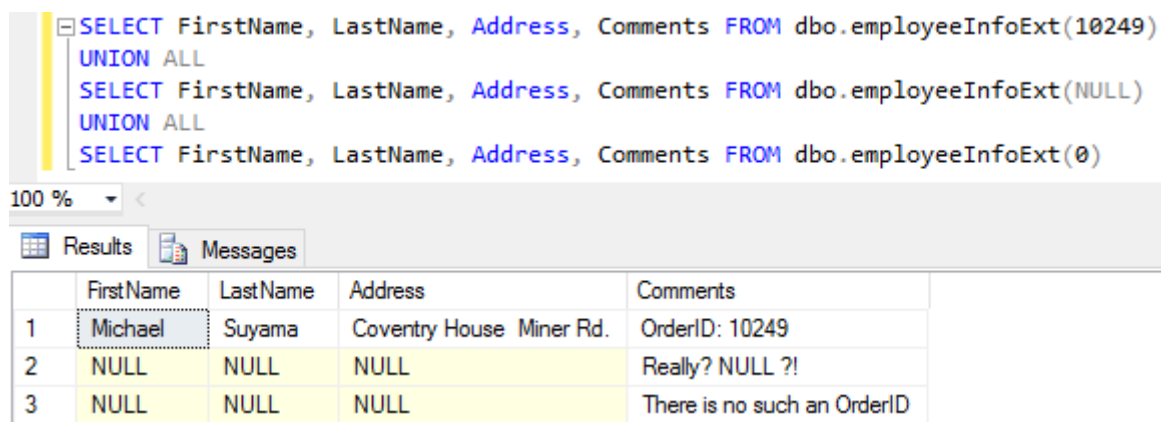
Korzystając z tabel **Orders** oraz **Employees** zaprojektuj funkcję **employeeInfoExt**, która dla podanego **OrderID** (parametr) zwróci imię, nazwisko, adres pracownika oraz dodatkową kolumnę z uwagami (**Comments**). Jeżeli identyfikator zamówienia istnieje w bazie danych to funkcja powinna zwrócić dane

pracownika. Jeżeli podany identyfikator nie istnieje w bazie danych lub ma wartość NULL to funkcja powinna zwrócić wartość NULL w kolumnach dotyczących danych pracownika. Kolumna **Comments** powinna przyjąć następujące wartości:

- Dla istniejącego OrderID: „*OrderID: @OrderID*”
- Dla OrderID – NULL: „*Really? NULL ?!*”
- Dla nieistniejącego OrderID w bazie danych: „*There is no such an OrderID*”

Przykładowy rezultat:

```
SELECT FirstName, LastName, Address, Comments FROM dbo.employeeInfoExt(10249)
UNION ALL
SELECT FirstName, LastName, Address, Comments FROM dbo.employeeInfoExt(NULL)
UNION ALL
SELECT FirstName, LastName, Address, Comments FROM dbo.employeeInfoExt(0)
```



	FirstName	LastName	Address	Comments
1	Michael	Suyama	Coventry House Miner Rd.	OrderID: 10249
2	NULL	NULL	NULL	Really? NULL ?!
3	NULL	NULL	NULL	There is no such an OrderID

#### Zadanie 5.

Zaimplementuj funkcję **stringsConcat**, która przyjmie 3 parametry:

- **String1** typu NVARCHAR(100)
- **String2** typu NVARCHAR(100)
- **Separator** typu NVARCHAR(10)

Rezultatem funkcji będzie konkatencja (złączenie) dwóch pierwszych parametrów z uwzględnieniem separatora pomiędzy nimi (parametr 3.).

Oczekiwany rezultat:

```
SELECT dbo.stringsConcat('Ala ma', 'kota', ' ') AS Result
UNION ALL
SELECT dbo.stringsConcat('Ten znak: ', ' to jest myślnik', '-') AS Result
UNION ALL
SELECT dbo.stringsConcat('Więcej: ', ' myślników', '---') AS Result
UNION ALL
SELECT dbo.stringsConcat('NULL', 'NULL', ' != ') AS Result
UNION ALL
SELECT dbo.stringsConcat('Prawdziwy NULL', NULL, ': ') AS Result
```

```

SELECT dbo.stringsConcat('Ala ma', 'kota', ' ') AS Result
UNION ALL
SELECT dbo.stringsConcat('Ten znak: ', ' to jest myślnik', '-') AS Result
UNION ALL
SELECT dbo.stringsConcat('Więcej: ', ' myślników', '---') AS Result
UNION ALL
SELECT dbo.stringsConcat('NULL', 'NULL', ' != ') AS Result
UNION ALL
SELECT dbo.stringsConcat('Prawdziwy NULL', NULL, ': ') AS Result

```

100 %

Results Messages

	Result
1	Ala ma kota
2	Ten znak: - to jest myślnik
3	Więcej: --- myślników
4	NULL != NULL
5	Prawdziwy NULL:

Jeżeli Twoja funkcja zamiast poprawnego rezultatu zwraca taki jak na poniższym zrzucie ekranu, to zmień sposób łączenia zmiennych - użyj funkcji CONCAT:

```

SELECT dbo.stringsConcat('Ala ma', 'kota', ' ') AS Result
UNION ALL
SELECT dbo.stringsConcat('Ten znak: ', ' to jest myślnik', '-') AS Result
UNION ALL
SELECT dbo.stringsConcat('Więcej: ', ' myślników', '---') AS Result
UNION ALL
SELECT dbo.stringsConcat('NULL', 'NULL', ' != ') AS Result
UNION ALL
SELECT dbo.stringsConcat('Prawdziwy NULL', NULL, ': ') AS Result

```

100 %

Results Messages

	Result
1	Ala ma kota
2	Ten znak: - to jest myślnik
3	Więcej: --- myślników
4	NULL != NULL
5	NULL

#### Zadanie 6. (\*)

Zaktualizuj utworzoną funkcję, tak, aby w przypadku, gdy dwa pierwsze parametry (łańcuchy znaków do złączenia) mają wartość NULL, na wyjściu powinien zostać zwrócony wynik: *Serio? NULLe?*

Oczekiwany rezultat:

```

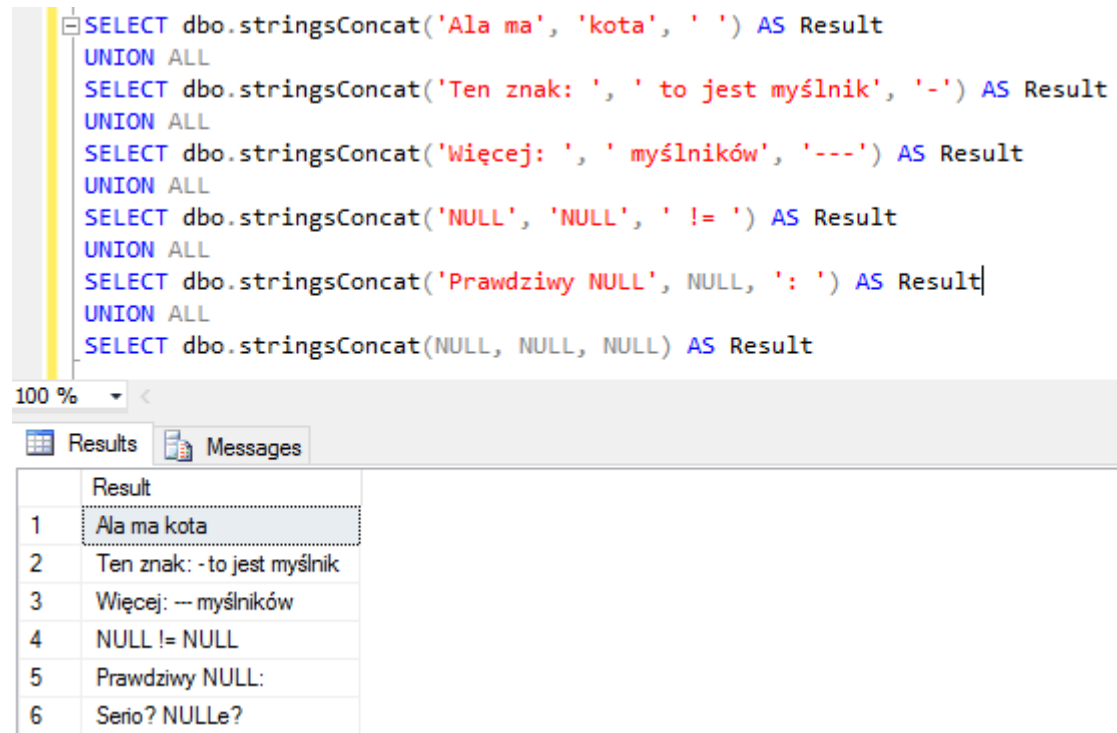
SELECT dbo.stringsConcat('Ala ma', 'kota', ' ') AS Result
UNION ALL

```

```

SELECT dbo.stringsConcat('Ten znak: ', ' to jest myślnik', '-') AS Result
UNION ALL
SELECT dbo.stringsConcat('Więcej: ', ' myślników', '---') AS Result
UNION ALL
SELECT dbo.stringsConcat('NULL', 'NULL', ' != ') AS Result
UNION ALL
SELECT dbo.stringsConcat('Prawdziwy NULL', NULL, ': ') AS Result
UNION ALL
SELECT dbo.stringsConcat(NULL, NULL, NULL) AS Result

```



	Result
1	Ala ma kota
2	Ten znak: - to jest myślnik
3	Więcej: --- myślników
4	NULL != NULL
5	Prawdziwy NULL:
6	Serio? NULLe?

### Zadanie 7.(\*)

Korzystając z tabeli **Employees** oraz poprzednio utworzonej funkcji do łączenia łańcuchów znaków, napisz zapytanie, które zwróci w jednej kolumnie konkatenację 3 pól: **FirstName**, **LastName**, **Title** w postaci:

**FirstName** (spacja) **LastName** (,spacja) **Title**.

Konkatenacja powinna być wykonana jedynie przy użyciu funkcji (bez dodatkowych złączeń na poziomie SQL).

Oczekiwany rezultat:

	FullName
1	Nancy Davolio, Sales Representative
2	Andrew Fuller, Vice President, Sales
3	Janet Leverling, Sales Representative
4	Margaret Peacock, Sales Representative
5	Steven Buchanan, Sales Manager
6	Michael Suyama, Sales Representative
7	Robert King, Sales Representative
8	Laura Callahan, Inside Sales Coordinator
9	Anne Dodsworth, Sales Representative

### Zadanie 8.

Zaimplementuj funkcję **fibonacci**, które dla podanego N, zwróci N-ty wyraz ciągu Fibonacciego opisanego wzorem (źródło: [https://pl.wikipedia.org/wiki/Ci%C4%85g\\_Fibonacciego](https://pl.wikipedia.org/wiki/Ci%C4%85g_Fibonacciego)):

$$F_n := \begin{cases} 0 & \text{dla } n = 0; \\ 1 & \text{dla } n = 1; \\ F_{n-1} + F_{n-2} & \text{dla } n > 1. \end{cases}$$

$F_0$	$F_1$	$F_2$	$F_3$	$F_4$	$F_5$	$F_6$	$F_7$	$F_8$	$F_9$	$F_{10}$	$F_{11}$	$F_{12}$	$F_{13}$	$F_{14}$	$F_{15}$	$F_{16}$	$F_{17}$	$F_{18}$	$F_{19}$
0	1	1	2	3	5	8	13	21	34	55	89	144	233	377	610	987	1597	2584	4181

Oczekiwany rezultat:

```
SELECT 0 AS N, dbo.fibonacci(0) AS FibValue
UNION ALL
SELECT 1, dbo.fibonacci(1) AS FibValue
UNION ALL
SELECT 2, dbo.fibonacci(2) AS FibValue
UNION ALL
SELECT 10, dbo.fibonacci(10) AS FibValue
UNION ALL
SELECT 18, dbo.fibonacci(18) AS FibValue
```

```

SELECT 0 AS N, dbo.fibonacci(0) AS FibValue
UNION ALL
SELECT 1, dbo.fibonacci(1) AS FibValue
UNION ALL
SELECT 2, dbo.fibonacci(2) AS FibValue
UNION ALL
SELECT 10, dbo.fibonacci(10) AS FibValue
UNION ALL
SELECT 18, dbo.fibonacci(18) AS FibValue

```

100 %

Results Messages

	N	FibValue
1	0	0
2	1	1
3	2	1
4	10	55
5	18	2584