

SQL - Funkcje okna (Window functions)

Lab 1

Imiona i nazwiska: Kacper Cienkosz, Miłosz Dubiel

Celem ćwiczenia jest przygotowanie środowiska pracy, wstępne zapoznanie się z działaniem funkcji okna (window functions) w SQL, analiza wydajności zapytań i porównanie z rozwiązaniami przy wykorzystaniu "tradycyjnych" konstrukcji SQL

Swoje odpowiedzi wpisuj w miejsca oznaczone jako:

Wyniki:

-- ...

Ważne/wymagane są komentarze.

Zamieść kod rozwiązania oraz zrzuty ekranu pokazujące wyniki, (dołącz kod rozwiązania w formie tekstowej/źródłowej)

Zwróć uwagę na formatowanie kodu

Oprogramowanie - co jest potrzebne?

Do wykonania ćwiczenia potrzebne jest następujące oprogramowanie:

- MS SQL Server - wersja 2019, 2022
- PostgreSQL - wersja 15/16
- SQLite
- Narzędzia do komunikacji z bazą danych
 - SSMS - Microsoft SQL Managment Studio
 - DtataGrip lub DBeaver
- Przykładowa baza Northwind
 - W wersji dla każdego z wymienionych serwerów

Oprogramowanie dostępne jest na przygotowanej maszynie wirtualnej

Dokumentacja/Literatura

- Kathi Kellenberger, Clayton Groom, Ed Pollack, Expert T-SQL Window Functions in SQL Server 2019, Apres 2019
- Itzik Ben-Gan, T-SQL Window Functions: For Data Analysis and Beyond, Microsoft 2020

- Kilka linków do materiałów które mogą być pomocne - <https://learn.microsoft.com/en-us/sql/t-sql/queries/select-over-clause-transact-sql?view=sql-server-ver16>
 - <https://www.sqlservertutorial.net/sql-server-window-functions/>
 - <https://www.sqlshack.com/use-window-functions-sql-server/>
 - <https://www.postgresql.org/docs/current/tutorial-window.html>
 - <https://www.postgresqtutorial.com/postgresql-window-function/>
 - <https://www.sqlite.org/windowfunctions.html>
 - <https://www.sqlitetutorial.net/sqlite-window-functions/>
- Ikonki używane w graficznej prezentacji planu zapytania w SSMS opisane są tutaj:
 - <https://docs.microsoft.com/en-us/sql/relational-databases/showplan-logical-and-physical-operators-reference>

Przygotowanie

Uruchom SSMS - Skonfiguruj połączenie z bazą Northwind na lokalnym serwerze MS SQL

Uruchom DataGrip (lub Dbeaver)

- Skonfiguruj połączenia z bazą Northwind
 - na lokalnym serwerze MS SQL
 - na lokalnym serwerze PostgreSQL
 - z lokalną bazą SQLite

Zadanie 1 - obserwacja

Wykonaj i porównaj wyniki następujących poleceń.

```
select avg(unitprice) avgprice
from products p;

select avg(unitprice) over () as avgprice
from products p;

select categoryid, avg(unitprice) avgprice
from products p
group by categoryid

select avg(unitprice) over (partition by categoryid) as avgprice
from products p;
```

Jaka jest są podobieństwa, jakie różnice pomiędzy grupowaniem danych a działaniem funkcji okna?

Wyniki:

```
northwind.public> select avg(unitprice) avgprice from products p
[2025-03-21 10:36:29] 1 row retrieved starting from 1 in 60 ms (execution:
10 ms, fetching: 50 ms)
```

avgprice	
1	28.83389609200614

```
northwind.public> select avg(unitprice) over () as avgprice from products
p
[2025-03-21 10:41:16] 77 rows retrieved starting from 1 in 20 ms
(execution: 3 ms, fetching: 17 ms)
```

avgprice	
1	28.83389609200614
2	28.83389609200614
3	28.83389609200614
4	28.83389609200614
5	28.83389609200614
6	28.83389609200614
7	28.83389609200614
8	28.83389609200614
9	28.83389609200614

```
northwind.public> select categoryid, avg(unitprice) avgprice
                    from products p
                    group by categoryid
[2025-03-21 10:42:36] 8 rows retrieved starting from 1 in 20 ms
(execution: 5 ms, fetching: 15 ms)
```

	categoryid	avgprice
1	8	20.682499885559082
2	7	32.369999694824216
3	1	37.979166666666664
4	5	20.25
5	4	28.729999923706053
6	2	22.854166825612385
7	6	54.00666666030884
8	3	25.1600000674908

```
northwind.public> select avg(unitprice) over (partition by categoryid) as
avgprice
                        from products p
[2025-03-21 10:44:36] 77 rows retrieved starting from 1 in 23 ms
(execution: 4 ms, fetching: 19 ms)
```

	avgprice
1	37.979166666666664
2	37.979166666666664
3	37.979166666666664
4	37.979166666666664
5	37.979166666666664
6	37.979166666666664
7	37.979166666666664
8	37.979166666666664
9	37.979166666666664

Główna różnica, którą zauważyliśmy, polega na tym, że funkcje okna nie zmieniają ilości wierszy zwracanych w wynikach zapytania. Jedynie dodana jest kolumna z odpowiednimi wartościami przypisanymi do odpowiedniego wiersza. Grupowanie danych zmienia wyniki, ponieważ łączy wiersze na podstawie określonej kolumny, przez co liczba wyników może się zmniejszyć.

Zadanie 2 - obserwacja

Wykonaj i porównaj wyniki następujących poleceń.

```
--1)

select p.productid, p.ProductName, p.unitprice,
       (select avg(unitprice) from products) as avgprice
from products p
where productid < 10

--2)
select p.productid, p.ProductName, p.unitprice,
       avg(unitprice) over () as avgprice
from products p
where productid < 10
```

Jaka jest różnica? Czego dotyczy warunek w każdym z przypadków? Napisz polecenie równoważne

- 1. z wykorzystaniem funkcji okna. Napisz polecenie równoważne
- 2. z wykorzystaniem podzapytania

Wyniki:

1. Wyniki z wykorzystaniem grupowania

```
northwind.public> select p.productid, p.ProductName, p.unitprice,
                        (select avg(unitprice) from products) as avgprice
                        from products p
                        where productid < 10
[2025-03-21 10:49:46] 9 rows retrieved starting from 1 in 40 ms
(execution: 8 ms, fetching: 32 ms)
```

	productid	productname	unitprice	avgprice
1	1	Chai	18	28.83389609200614
2	2	Chang	19	28.83389609200614
3	3	Aniseed Syrup	10	28.83389609200614
4	4	Chef Anton's Cajun Seasoning	22	28.83389609200614
5	5	Chef Anton's Gumbo Mix	21.35	28.83389609200614
6	6	Grandma's Boysenberry Spread	25	28.83389609200614
7	7	Uncle Bob's Organic Dried Pears	30	28.83389609200614
8	8	Northwoods Cranberry Sauce	40	28.83389609200614
9	9	Mishi Kobe Niku	97	28.83389609200614

Polecenie równoważne z wykorzystaniem funkcji okna

```
select
  p.productid,
  p.ProductName,
  p.unitprice,
```

```
avg(p.unitprice) over () as avgprice
from products p
order by productid
limit 9;
```

	productid	productname	unitprice	avgprice
1	1	Chai	18	28.83389609200614
2	2	Chang	19	28.83389609200614
3	3	Aniseed Syrup	10	28.83389609200614
4	4	Chef Anton's Cajun Seasoning	22	28.83389609200614
5	5	Chef Anton's Gumbo Mix	21.35	28.83389609200614
6	6	Grandma's Boysenberry Spread	25	28.83389609200614
7	7	Uncle Bob's Organic Dried Pears	30	28.83389609200614
8	8	Northwoods Cranberry Sauce	40	28.83389609200614
9	9	Mishi Kobe Niku	97	28.83389609200614

2. Wyniki z wykorzystaniem funkcji okna

```
northwind.public> select p.productid, p.ProductName, p.unitprice,
                        avg(unitprice) over () as avgprice
                        from products p
                        where productid < 10
[2025-03-21 10:59:34] 9 rows retrieved starting from 1 in 16 ms
(execution: 3 ms, fetching: 13 ms)
```

	productid	productname	unitprice	avgprice
1	1	Chai	18	31.372222264607746
2	2	Chang	19	31.372222264607746
3	3	Aniseed Syrup	10	31.372222264607746
4	4	Chef Anton's Cajun Seasoning	22	31.372222264607746
5	5	Chef Anton's Gumbo Mix	21.35	31.372222264607746
6	6	Grandma's Boysenberry Spread	25	31.372222264607746
7	7	Uncle Bob's Organic Dried Pears	30	31.372222264607746
8	8	Northwoods Cranberry Sauce	40	31.372222264607746
9	9	Mishi Kobe Niku	97	31.372222264607746

Polecenie równoważne z wykorzystaniem podzapytania

```
select p.productid, p.ProductName, p.unitprice,
       (select avg(unitprice) from products pp where pp.productid < 10) as
avgprice
from products p
where productid < 10;
```

	productid	productname	unitprice	avgprice
1	1	Chai	18	31.372222264607746
2	2	Chang	19	31.372222264607746
3	3	Aniseed Syrup	10	31.372222264607746
4	4	Chef Anton's Cajun Seasoning	22	31.372222264607746
5	5	Chef Anton's Gumbo Mix	21.35	31.372222264607746
6	6	Grandma's Boysenberry Spread	25	31.372222264607746
7	7	Uncle Bob's Organic Dried Pears	30	31.372222264607746
8	8	Northwoods Cranberry Sauce	40	31.372222264607746
9	9	Mishi Kobe Niku	97	31.372222264607746

Różnica między wykorzystaniem podzapytania i funkcji okna polega na tym, że funkcja okna stosuje się do danych już zawężonych przez warunek WHERE, a podzapytanie stosuje się do wszystkich danych wybranych w podzapytaniu, a następnie wybiera odpowiednie dane na podstawie warunku. Stąd, aby zrobić do pierwszego polecenia odpowiadające mu z wykorzystaniem funkcji okna, należy zastosować funkcję okna do wszystkich danych następnie posortować po productid i wziąć 9 pierwszych elementów. Aby zrobić drugie zapytanie za pomocą podzapytania, należy zawęzić dane warunkiem where w podzapytaniu.

Zadanie 3

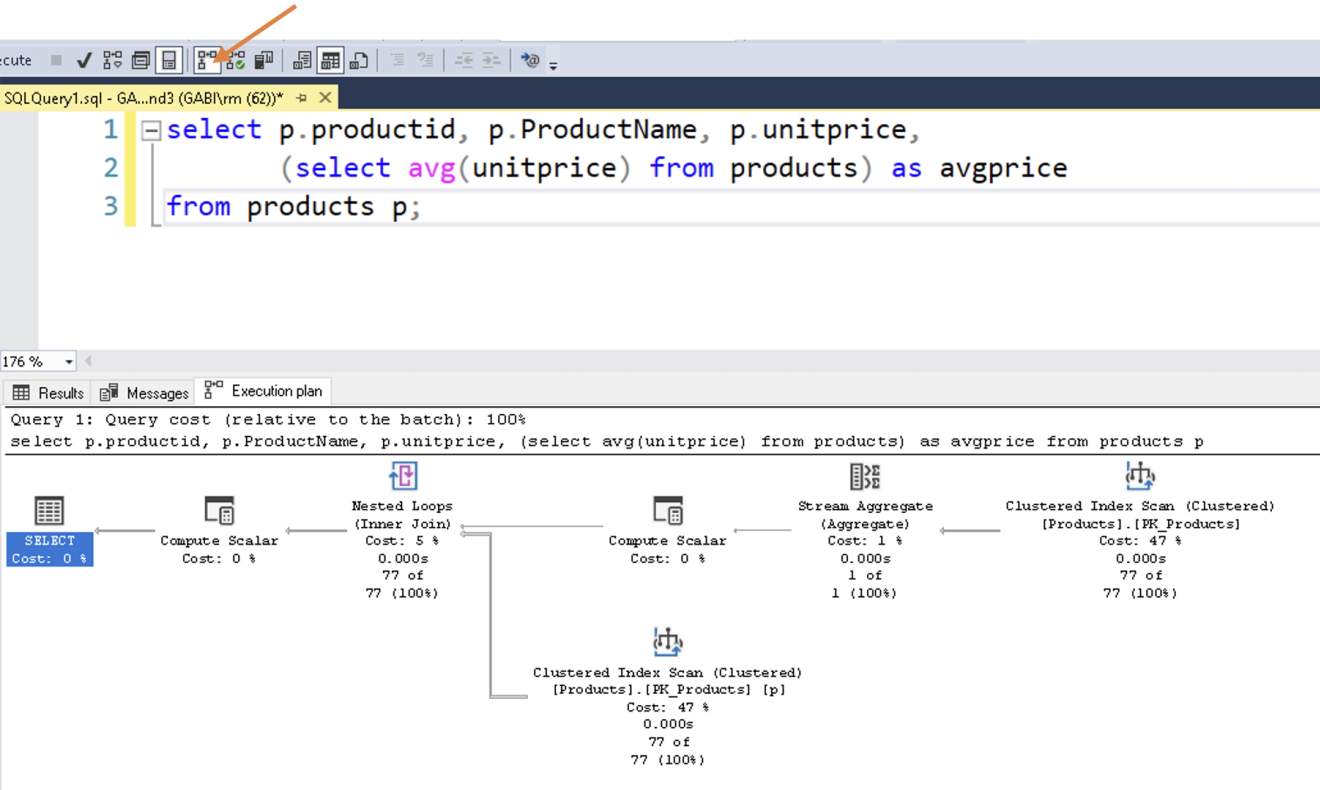
Baza: Northwind, tabela: products

Napisz polecenie, które zwraca: id produktu, nazwę produktu, cenę produktu, średnią cenę wszystkich produktów.

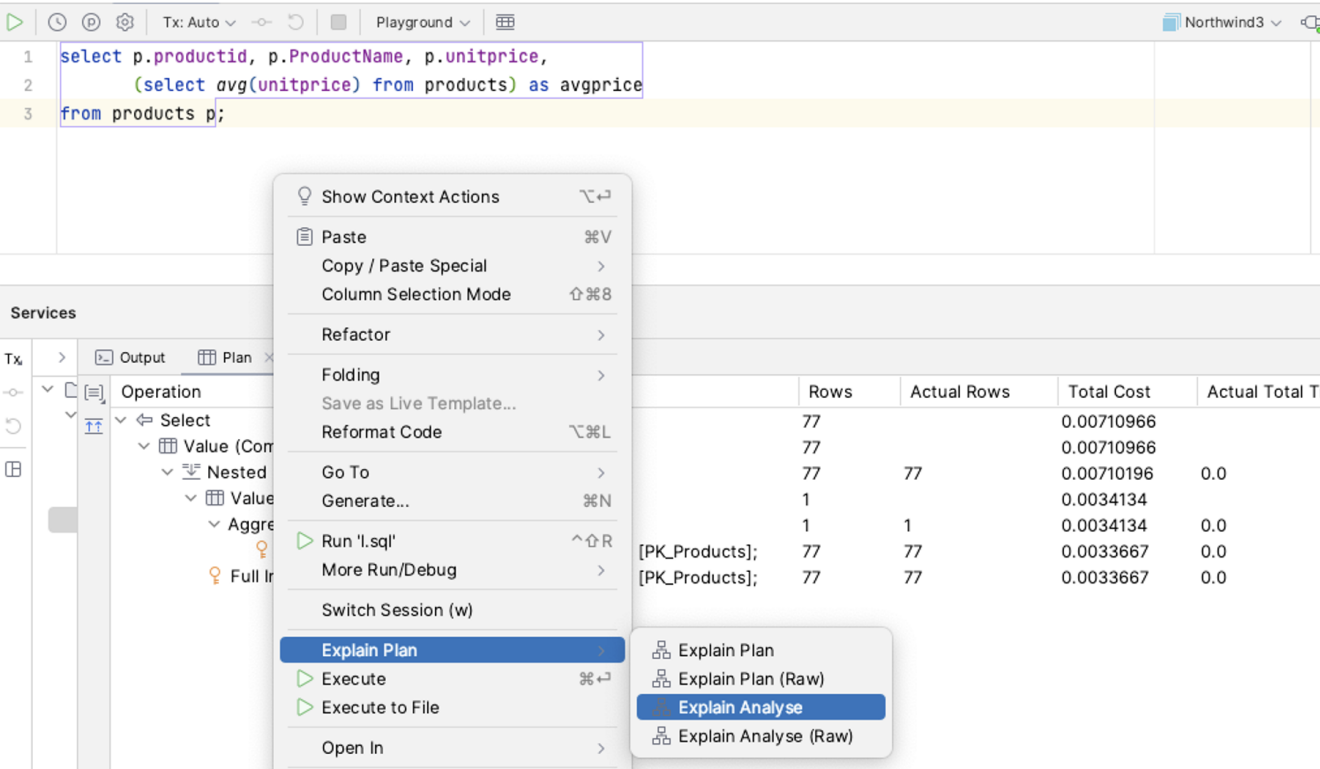
Napisz polecenie z wykorzystaniem z wykorzystaniem podzapytania, join'a oraz funkcji okna. Porównaj czasy oraz plany wykonania zapytań.

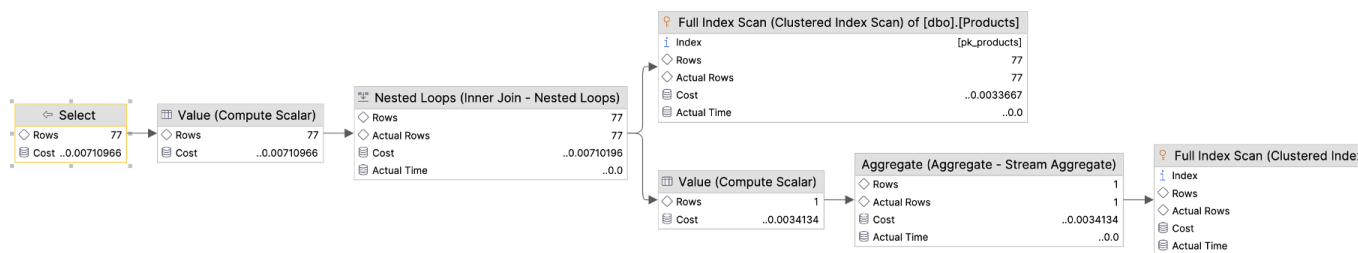
Przetestuj działanie w różnych SZBD (MS SQL Server, PostgreSQL, SQLite)

W SSMS włącz dwie opcje: Include Actual Execution Plan oraz Include Live Query Statistics



W DataGrip użyj opcji Explain Plan/Explain Analyze





Operation	Params	Rows	Actual Rows	Total Cost	Actual Total Time
Select		77		0.00710966	
Value (Compute Scalar)		77		0.00710966	
Inner Join - Nested Loop		77	77	0.00710196	0.0
Compute Scalar		1		0.0034134	
Aggregate (Aggregate - Scalar)		1	1	0.0034134	0.0
Full Index Scan (Clustered table: [dbo].[Products]; index: [PK_Products];)		77	77	0.0033667	0.0
Full Index Scan (Clustered table: [dbo].[Products]; index: [PK_Products];)		77	77	0.0033667	0.0

Wyniki:

— — — — —

MSSQL

The screenshot displays the DataGrip IDE interface. At the top, the menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, and Git. The main window is divided into several panes:

- Left Pane (Database Explorer):** Shows a tree view of the database structure. The 'northwind' database is selected, showing tables like 'public', 'sequences', and 'Database Objects'.
- Top Center Pane (SQL Editor):** Contains a SQL query:


```
select p.productid, p.ProductName, p.unitprice, (select avg(unitprice) from products) as avgprice
from products p;
```
- Bottom Center Pane (Execution Plan):** Visualizes the query execution plan. It shows a 'Full Index Scan (Clustered Index Scan) of [dbo].[products]' feeding into an 'Aggregate (Aggregate - Stream Aggregate)' operation, which then feeds into a 'Value (Compute Scalar)' operation. The plan also shows a 'Nested Loops (Inner Join - Nested Loops)' operation.
- Bottom Right Pane (Table):** Displays a table with execution statistics. The table has columns: Operation, Params, Rows, Actual Rows, Total Cost, Actual Total Time, and Raw Desc.

The table of execution statistics is as follows:

Operation	Params	Rows	Actual Rows	Total Cost	Actual Total Time	Raw Desc
Select		77		0.00859114		CardinalityEstim...
Value (Compute Scala)		77		0.00859114		AvgRowSize = 7...
Nested Loops (Inne)		77	77	0.00858344	1.0	AvgRowSize = 7...
Value (Compute)		1		0.00415414		AvgRowSize = 1...
Aggregate (Aggr)		1	1	0.00415414	1.0	AvgRowSize = 1...
Full Index Scan (table: [dbo].[products]; index: [pk_products];)		77	77	0.00410744	1.0	AvgRowSize = 1...
Full Index Scan (table: [dbo].[products]; index: [pk_products];)		77	77	0.00410744	0.0	AvgRowSize = 6...

The bottom status bar shows the current file path: Database Consoles > mssql-northwind > console_1 [mssql-northwind].

PostgreSQL

The screenshot shows the DataGrip interface with a PostgreSQL query executed in the console. The query is: `select p.productid, p.ProductName, p.unitprice, (select avg(unitprice) from products) as avgprice from products p;`

The execution plan visualizes the query execution flow:

- Select** (Operator)
- Full Scan (Seq Scan) of products** (Operator): Rows: 77, Actual Rows: 77, Cost: 1.97..3.75, Actual Time: 0.45..0.459.
- Aggregate** (Operator): Rows: 1, Actual Rows: 1, Cost: 1.96..1.97, Actual Time: 0.019..0.019.
- Full Scan (Seq Scan) of products** (Operator): Rows: 77, Actual Rows: 77, Cost: 0.0..1.77, Actual Time: 0.004..0.008.

The Services pane shows the execution details:

Operation	Params	Rows	Actual Rows	Total Cost	Actual Total Time	Startup Cost	Actual Startup Time	Raw Desc
Select								Planning Time ...
Full Scan (Seq: table: products;		77	77	3.75	0.459	1.97	0.45	Parallel Aware ...
Aggregate		1	1	1.97	0.019	1.96	0.019	Strategy = Plai...
Full Scan table: products;		77	77	1.77	0.008	0.0	0.004	Parent Relation...

SQLite

The screenshot shows the DataGrip interface with a SQLite query executed in the console. The query is: `select p.productid, p.ProductName, p.unitprice, (select avg(unitprice) from products) as avgprice from products p;`

The execution plan visualizes the query execution flow:

- Select** (Operator)
- Subquery** (Operator)
- Full Scan of products** (Operator)
- Full Scan of p** (Operator)

The Services pane shows the execution details:

ProductID	ProductName	UnitPrice	avgprice
1	Chai	18	28.8663636363637
2	Chang	19	28.8663636363637
3	Aniseed Syrup	10	28.8663636363637
4	Chef Anton's Cajun Seasoning	22	28.8663636363637
5	Chef Anton's Gumbo Mix	21.35	28.8663636363637
6	Grandma's Boysenberry Spread	25	28.8663636363637
7	Uncle Bob's Organic Dried Pears	30	28.8663636363637

Zadanie 4

Baza: Northwind, tabela products

Napisz polecenie, które zwraca: id produktu, nazwę produktu, cenę produktu, średnią cenę produktów w kategorii, do której należy dany produkt. Wyświetl tylko pozycje (produkty) których cena jest większa niż średnia cena.

Napisz polecenie z wykorzystaniem podzapytania, join'a oraz funkcji okna. Porównaj zapytania. Porównaj czasy oraz plany wykonania zapytań.

Przetestuj działanie w różnych SZBD (MS SQL Server, PostgreSQL, SQLite)

Wyniki:

--

...

zadanie	pkt
1	1
2	1
3	1
4	1
razem	4