SQL - Funkcje okna (Window functions)

Lab 1-2

|  |  |
| --- | --- |
| Imię i Nazwisko: |  |

Celem ćwiczenia jest 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 **czerwone pola**.

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 lib 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.postgresqltutorial.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 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?

|  |
| --- |
|  |

# 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

.

|  |
| --- |
|  |

# Zadanie 3

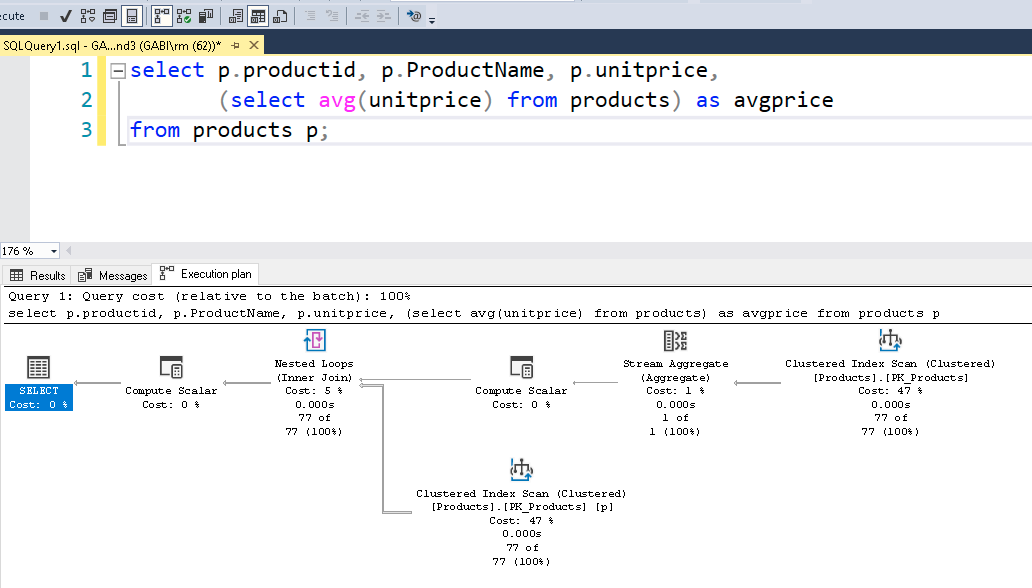
Baza: Northwind, tabela: products

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)

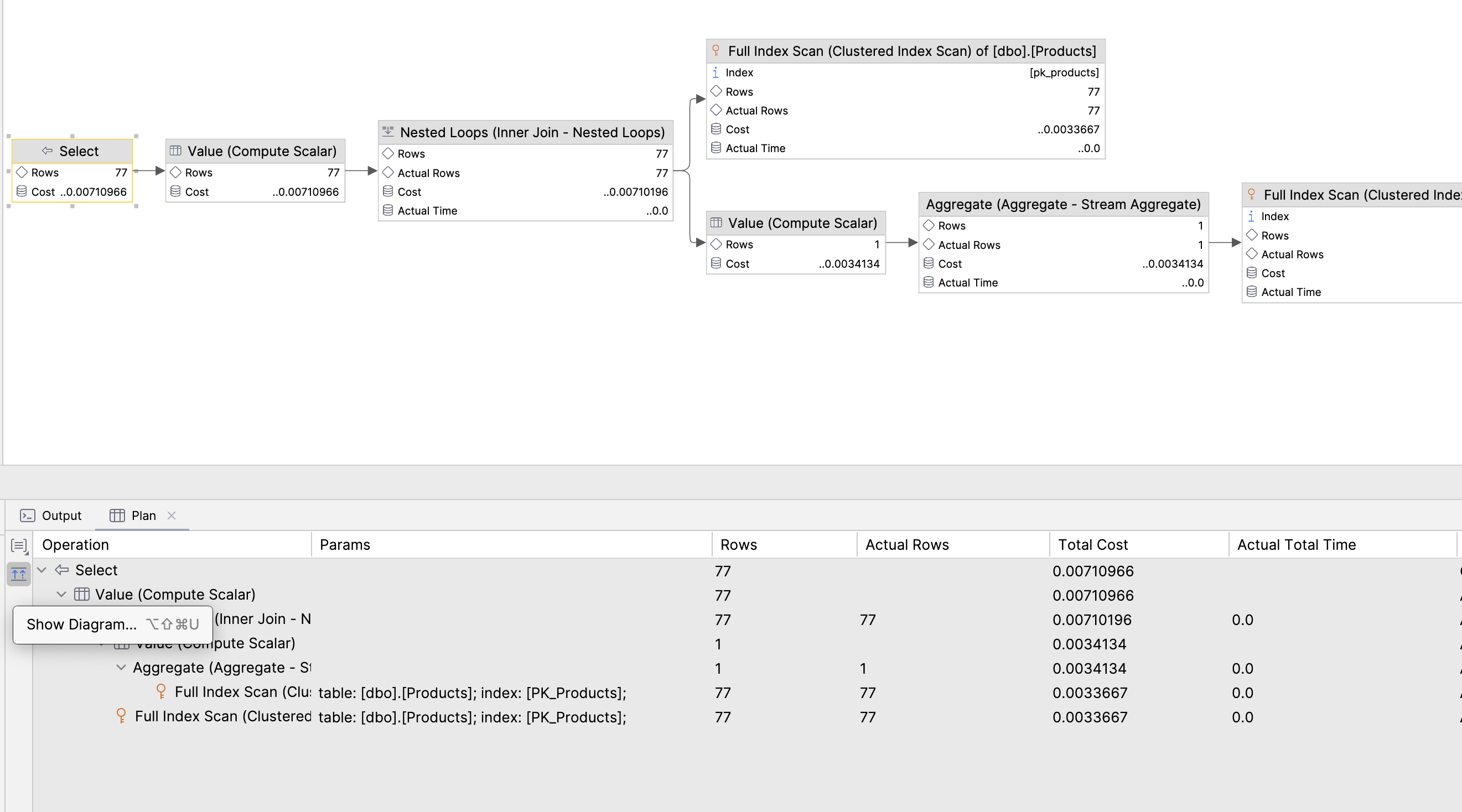
W SSMS włącz dwie opcje: Include Actual Execution Plan oraz Include Live Query Statistics



W DataGrip użyj opcji Explain Plan/Explain Analyze

Obraz zawierający tekst

Opis wygenerowany automatycznie



|  |
| --- |
|  |

# 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)

|  |
| --- |
|  |

# Zadanie 5 - przygotowanie

Baza: Northwind

Tabela products zawiera tylko 77 wiersz. Warto zaobserwować działanie na większym zbiorze danych.

Wygeneruj tabelę zawierającą kilka milionów (kilkaset tys.) wierszy

Stwórz tabelę o następującej strukturze:

Skrypt dla SQL Srerver

|  |
| --- |
| create table product\_history(  id int identity(1,1) not null,  productid int,  productname varchar(40) not null,  supplierid int null,  categoryid int null,  quantityperunit varchar(20) null,  unitprice decimal(10,2) null,  quantity int,  value decimal(10,2),  date date,  constraint pk\_product\_history primary key clustered  (id asc ) ) |

Wygeneruj przykładowe dane:

Dla 30000 iteracji, tabela będzie zawierała nieco ponad 2mln wierszy (dostostu ograniczenie do możliwości swojego komputera)

Skrypt dla SQL Srerver

|  |
| --- |
| declare @i int set @i = 1 while @i <= 30000 begin  insert product\_history  select productid, ProductName, SupplierID, CategoryID,  QuantityPerUnit,*round*(*RAND*()\*unitprice + 10,2),  *cast*(*RAND*() \* productid + 10 as int), 0,  *dateadd*(day, @i, '1940-01-01')  from products  set @i = @i + 1; end;  update product\_history set value = unitprice \* quantity where 1=1; |

Skrypt dla Postgresql

|  |
| --- |
| create table product\_history(  id int generated always as identity not null  constraint pkproduct\_history  primary key,  productid int,  productname varchar(40) not null,  supplierid int null,  categoryid int null,  quantityperunit varchar(20) null,  unitprice decimal(10,2) null,  quantity int,  value decimal(10,2),  date date ); |

Wygeneruj przykładowe dane:

Skrypt dla Postgresql

|  |
| --- |
| do $$ begin  for cnt in 1..30000 loop  insert into product\_history(productid, productname, supplierid,  categoryid, quantityperunit,  unitprice, quantity, value, date)  select productid, productname, supplierid, categoryid,  quantityperunit,  *round*((*random*()\*unitprice + 10)::numeric,2),  *cast*(*random*() \* productid + 10 as int), 0,  *cast*('1940-01-01' as date) + cnt  from products;  end loop; end; $$;  update product\_history set value = unitprice \* quantity where 1=1; |

Wykonaj polecenia: select count(\*) from product\_history, potwierdzające wykonanie zadania

|  |
| --- |
|  |

# Zadanie 6

Baza: Northwind, tabela product\_history

To samo co w zadaniu 3, ale dla większego zbioru danych

Napisz polecenie, które zwraca: id pozycji, 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)

|  |
| --- |
|  |

## Zadanie 7

Baza: Northwind, tabela product\_history

Lekka modyfikacja poprzedniego zadania

Napisz polecenie, które zwraca: id pozycji, id produktu, nazwę produktu, cenę produktu oraz

* średnią cenę produktów w kategorii do której należy dany produkt.
* łączną wartość sprzedaży produktów danej kategorii (suma dla pola value)
* średnią cenę danego produktu w roku którego dotyczy dana pozycja
* łączną wartość sprzedaży produktów danej kategorii (suma dla pola value)

Napisz polecenie z wykorzystaniem podzapytania, join'a oraz funkcji okna. Porównaj zapytania. W przypadku funkcji okna spróbuj użyć klauzuli WINDOW.

Porównaj czasy oraz plany wykonania zapytań.

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

|  |
| --- |
|  |

## Zadanie 8 - obserwacja

Funkcje rankingu, row\_number(), rank(), dense\_rank()

Wykonaj polecenie, zaobserwuj wynik. Porównaj funkcje row\_number(), rank(), dense\_rank()

|  |
| --- |
| select productid, productname, unitprice, categoryid,  *row\_number*() over(partition by categoryid order by unitprice desc)  as rowno,  *rank*() over(partition by categoryid order by unitprice desc)  as rankprice,  *dense\_rank*() over(partition by categoryid order by unitprice desc)  as denserankprice from products; |

|  |
| --- |
|  |

Zadanie

Spróbuj uzyskać ten sam wynik bez użycia funkcji okna

|  |
| --- |
|  |

## Zadanie 9

Baza: Northwind, tabela product\_history

Dla każdego produktu, podaj 4 najwyższe ceny tego produktu w danym roku. Zbiór wynikowy powinien zawierać:

rok

id produktu

nazwę produktu

cenę

datę (datę uzyskania przez produkt takiej ceny)

pozycję w rankingu

Uporządkuj wynik wg roku, nr produktu, pozycji w rankingu

|  |
| --- |
|  |

Spróbuj uzyskać ten sam wynik bez użycia funkcji okna, porównaj wyniki, czasy i plany zapytań. Przetestuj działanie w różnych SZBD (MS SQL Server, PostgreSql, SQLite)

|  |
| --- |
|  |

## Zadanie 10 - obserwacja

Funkcje lag(), lead()

Wykonaj polecenia, zaobserwuj wynik. Jak działają funkcje lag(), lead()

|  |
| --- |
| select productid, productname, categoryid, date, unitprice,  *lag*(unitprice) over (partition by productid order by date)  as previousprodprice,  *lead*(unitprice) over (partition by productid order by date)  as nextprodprice from product\_history where productid = 1 and *year*(date) = 2022 order by date;  with t as (select productid, productname, categoryid, date, unitprice,  *lag*(unitprice) over (partition by productid  order by date) as previousprodprice,  *lead*(unitprice) over (partition by productid  order by date) as nextprodprice  from product\_history  ) select *\** from t where productid = 1 and *year*(date) = 2022 order by date; |

|  |
| --- |
|  |

Zadanie

Spróbuj uzyskać ten sam wynik bez użycia funkcji okna, porównaj wyniki, czasy i plany zapytań. Przetestuj działanie w różnych SZBD (MS SQL Server, PostgreSql, SQLite)

|  |
| --- |
|  |

## Zadanie 11

Baza: Northwind, tabele customers, orders, order details

Napisz polecenie które wyświetla inf. o zamówieniach

Zbiór wynikowy powinien zawierać:

nazwę klienta, nr zamówienia,

datę zamówienia,

wartość zamówienia (wraz z opłatą za przesyłkę),

nr poprzedniego zamówienia danego klienta,

datę poprzedniego zamówienia danego klienta,

wartość poprzedniego zamówienia danego klienta.

|  |
| --- |
|  |

## Zadanie 12 - obserwacja

Funkcje first\_value(), last\_value()

Wykonaj polecenia, zaobserwuj wynik. Jak działają funkcje first\_value(), last\_value(). Skomentuj uzyskane wyniki. Czy funkcja first\_value pokazuje w tym przypadku najdroższy produkt w danej kategorii, czy funkcja last\_value() pokazuje najtańszy produkt? Co jest przyczyną takiego działania funkcji last\_value. Co trzeba zmienić żeby funkcja last\_value pokazywała najtańszy produkt w danej kategorii

|  |
| --- |
| select productid, productname, unitprice, categoryid,  *first\_value*(productname) over (partition by categoryid  order by unitprice desc) first,  *last\_value*(productname) over (partition by categoryid  order by unitprice desc) last from products order by categoryid, unitprice desc; |

|  |
| --- |
|  |

Zadanie

Spróbuj uzyskać ten sam wynik bez użycia funkcji okna, porównaj wyniki, czasy i plany zapytań. Przetestuj działanie w różnych SZBD (MS SQL Server, PostgreSql, SQLite)

|  |
| --- |
|  |

## Zadanie 13

Baza: Northwind, tabele orders, order details

Napisz polecenie które wyświetla inf. o zamówieniach

Zbiór wynikowy powinien zawierać:

Id klienta,

nr zamówienia,

datę zamówienia,

wartość zamówienia (wraz z opłatą za przesyłkę),

dane zamówienia klienta o najniższej wartości w danym miesiącu

nr zamówienia o najniższej wartości w danym miesiącu

datę tego zamówienia

wartość tego zamówienia

dane zamówienia klienta o najwyższej wartości w danym miesiącu

nr zamówienia o najniższej wartości w danym miesiącu

datę tego zamówienia

wartość tego zamówienia

|  |
| --- |
|  |

## Zadanie 14

Baza: Northwind, tabela product\_history

Napisz polecenie które pokaże wartość sprzedaży każdego produktu narastająco od początku każdego miesiąca. Użyj funkcji okna

Zbiór wynikowy powinien zawierać:

id pozycji

id produktu

datę

wartość sprzedaży produktu w danym dniu

wartość sprzedaży produktu narastające od początku miesiąca

|  |
| --- |
|  |

Spróbuj wykonać zadanie bez użycia funkcji okna. Spróbuj uzyskać ten sam wynik bez użycia funkcji okna, porównaj wyniki, czasy i plany zapytań. Przetestuj działanie w różnych SZBD (MS SQL Server, PostgreSql, SQLite)

|  |
| --- |
|  |

## Zadanie 15

Wykonaj kilka "własnych" przykładowych analiz. Czy są jeszcze jakieś ciekawe/przydatne funkcje okna (z których nie korzystałeś w ćwiczeniu)? Spróbuj ich użyć w zaprezentowanych przykładach.

|  |
| --- |
|  |

Punktacja

|  |  |
| --- | --- |
| zadanie | pkt |
| 1 | 0,5 |
| 2 | 0,5 |
| 3 | 1 |
| 4 | 1 |
| 5 | 0,5 |
| 6 | 2 |
| 7 | 2 |
| 8 | 0,5 |
| 9 | 2 |
| 10 | 1 |
| 11 | 2 |
| 12 | 1 |
| 13 | 2 |
| 14 | 2 |
| 15 | 2 |
| razem | 20 |