WOJSKOWA AKADEMIA TECHNICZNA

im. Jarosława Dąbrowskiego

**Wydział Cybernetyki**



Przedmiot:

**Systemy Baz Danych**

**SPRAWOZDANIE**

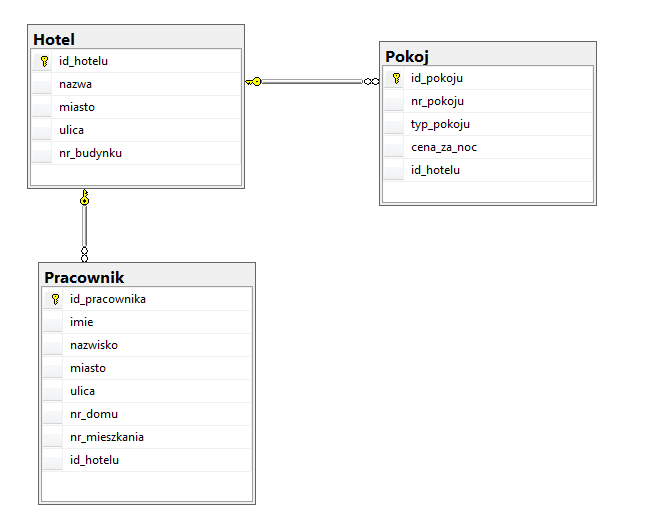
**Projekt rozproszonej bazy danych**

|  |  |
| --- | --- |
| Autor:  **sierż. pchor. inż. Michał Duda** | Prowadzący: |
| **sierż. pchor. inż. Mateusz Rzemieniewski** | **mgr inż. Maciej Szymczyk** |
| Grupa: **WCY18IH1S4** |  |

Link do repozytorium projektu:

<https://github.com/owndyye/RBD-Hotel>

1. Model RBD



Klasy:

|  |  |
| --- | --- |
| Tabela | opis |
| Pracownik | Przechowuje dane o pracownikach hotelu |
| Hotel | Przechowuje dane o hotelu |
| Pokoj | Przechowuje dane o pokojach hotelowych |

Związki:

|  |  |
| --- | --- |
| związek | opis |
| zatrudnia | Hotel zatrudnia pracowników |
| posiada | Hotel posiada pokój |

1. Zapytania generujące bazę danych MS SQL 2017

**Hotel**

CREATE TABLE dbo.Hotel(

id\_hotelu INTEGER identity(1,1) NOT NULL,

nazwa VARCHAR(20) NOT NULL,

miasto VARCHAR(20) NOT NULL,

ulica VARCHAR(20) NOT NULL,

nr\_budynku VARCHAR(5) NOT NULL,

CONSTRAINT PK\_HOTEL PRIMARY KEY CLUSTERED (id\_hotelu)

)

**Pokoj**

CREATE TABLE dbo.Pokoj(

id\_pokoju INTEGER identity(1,1) NOT NULL,

nr\_pokoju VARCHAR(3) NOT NULL,

typ\_pokoju VARCHAR(10) NOT NULL,

cena\_za\_noc Int NOT NULL,

id\_hotelu int FOREIGN KEY REFERENCES Hotel(id\_hotelu),

CONSTRAINT PK\_POKOJ PRIMARY KEY CLUSTERED (id\_pokoju)

)

**Pracownik**

CREATE TABLE dbo.Pracownik(

id\_pracownika INTEGER identity(1,1) NOT NULL,

imie VARCHAR(20) NOT NULL,

nazwisko VARCHAR(20) NOT NULL,

miasto VARCHAR(20) NOT NULL,

ulica VARCHAR(20) NOT NULL,

nr\_domu VARCHAR(5) NOT NULL,

nr\_mieszkania VARCHAR(5) NOT NULL,

id\_hotelu int FOREIGN KEY REFERENCES Hotel(id\_hotelu),

constraint PK\_PRACOWNIK primary key (id\_pracownika)

)

1. Skrypt do bazy MS SQL 2017

DECLARE @index INT

DECLARE @index1 INT

DECLARE @index2 INT

DECLARE @index3 INT

Declare @Nazwa varchar(30)

Declare @Nazwa\_hotelu varchar(30)

Declare @Ulica varchar(30)

Declare @Nr\_budynku int

Declare @Kod\_pocztowy int

Declare @Miasto varchar(30)

Declare @NIP1 int

Declare @ile\_gwiazdek1 int

Declare @Numer\_pokoju int

Declare @Typ\_pokoju varchar(30)

Declare @cena\_za\_noc int

Declare @id\_hotelu int

Declare @Imie varchar(30)

Declare @Nazwisko varchar(30)

Declare @PESEL int

Declare @Telefon int

Declare @Limit1DWN int

Declare @Limit1UP int

Set @Limit1DWN = 1

Set @Limit1UP = 40000

Declare @Lowerlimit2 int

Declare @Uplimit2 int

Set @Lowerlimit2 = 1

Set @Uplimit2 = 350

Declare @count int

Set @count = 1

While @count <= 100000

Begin

Set @index = CAST(RAND() \* 5 + 1 AS INT)

Set @index1 = CAST(RAND() \* 5 + 1 AS INT)

Set @index2 = CAST(RAND() \* 5 + 1 AS INT)

Set @index3 = CAST(RAND() \* 5 + 1 AS INT)

Select @Nazwa = CHOOSE(@index,'Adam','Stefania','Jacek','Gawel','Zbigniew')

Select @Nazwa\_hotelu = CHOOSE(@index1,'Abra','Elwis','Hilton','Mariot','Sheraton')

Select @Ulica = CHOOSE(@index2,'Abrowska','Kielecka','Wojskowa','Polowa','Wolna')

Select @Nr\_budynku = Round(((@Uplimit2 - @Lowerlimit2) \* Rand()) + @Lowerlimit2, 0)

Select @Miasto = CHOOSE(@index3,'Poznan','Kielce','Warszawa','Gdansk','Krakow')

Select @Numer\_pokoju = Round(((@Uplimit2 - @Lowerlimit2) \* Rand()) + @Lowerlimit2, 0)

Select @Typ\_pokoju = CHOOSE(@index2,'jedynka','dwojka','trojka','czworka','vip')

Select @cena\_za\_noc = Round(((@Uplimit2 - @Lowerlimit2) \* Rand()) + @Lowerlimit2, 0)

Select @id\_hotelu = Round(((@Limit1UP - @Limit1DWN) \* Rand()) + @Limit1DWN, 0)

Select @Imie = CHOOSE(@index3,'Adam','Stefania','Jacek','Gawel','Zbigniew')

Select @Nazwisko = CHOOSE(@index,'Wojtkowiak','Adamowski','Kowalski','Nowak','Mazurek')

Insert Into Hotel (nazwa,miasto,ulica,nr\_budynku) values (@Nazwa\_hotelu, @Miasto, @Ulica,@Nr\_budynku)

Insert Into Pracownik (imie,nazwisko,miasto,ulica,nr\_domu,nr\_mieszkania,id\_hotelu) values (@Imie, @Nazwisko, @Miasto,@Ulica, @Nr\_budynku,@Nr\_budynku,@id\_hotelu)

Insert Into Pokoj (nr\_pokoju,typ\_pokoju,cena\_za\_noc,id\_hotelu) values (@Numer\_pokoju, @Typ\_pokoju, @cena\_za\_noc, @id\_hotelu)

Print @count

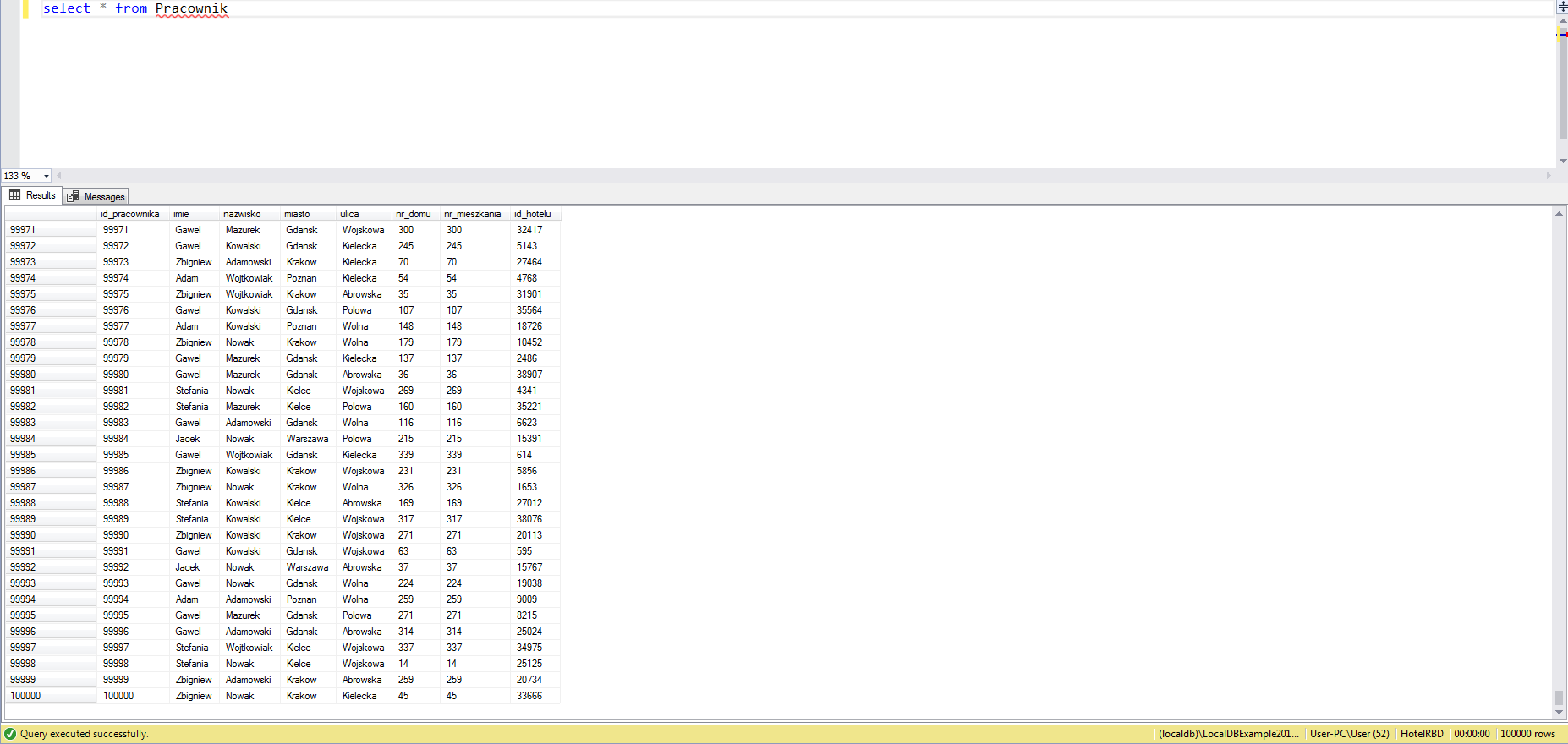
Print @Nazwa

Set @count = @count + 1

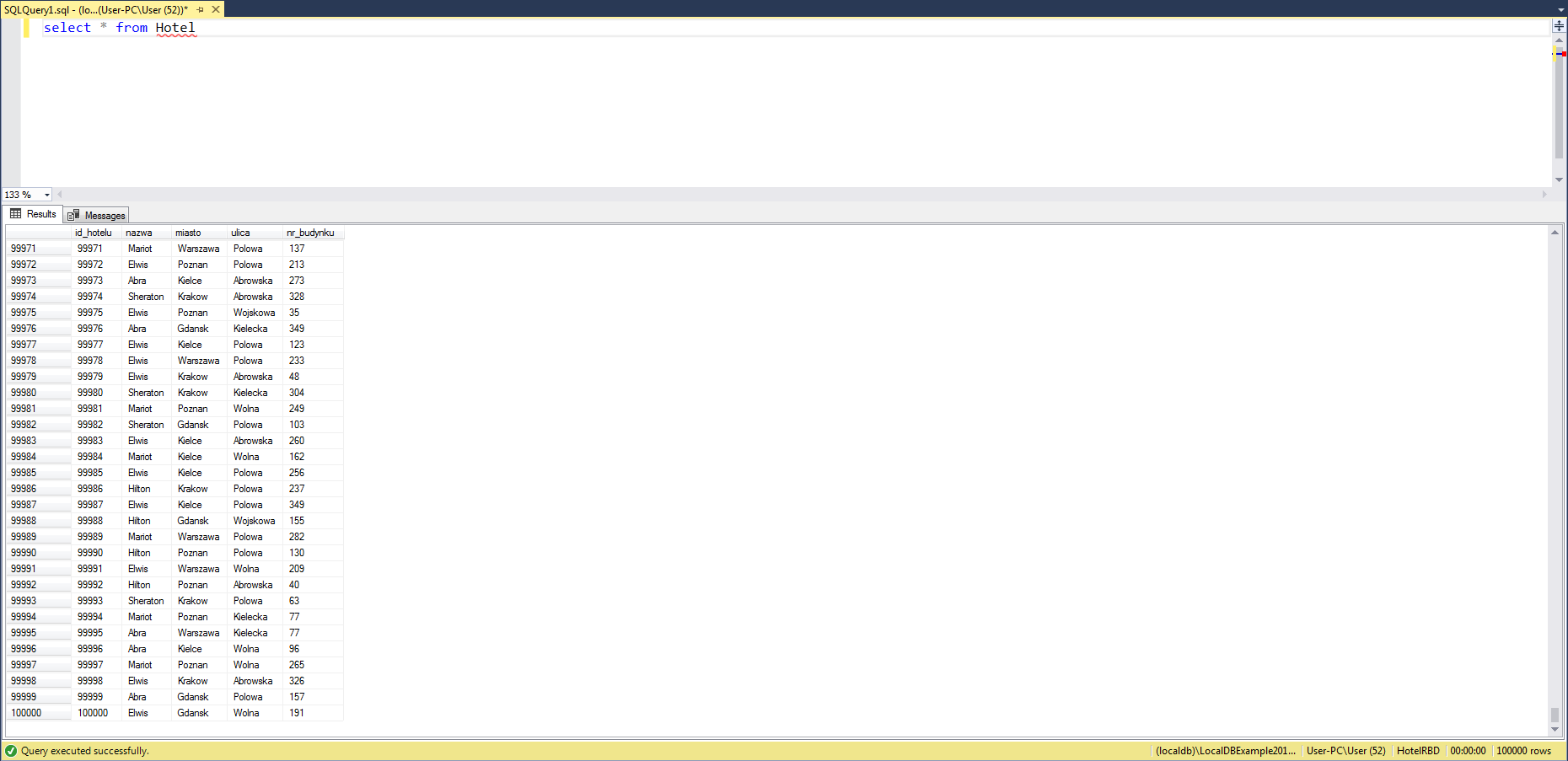
End

**Wynik dla 100k rekordów:**

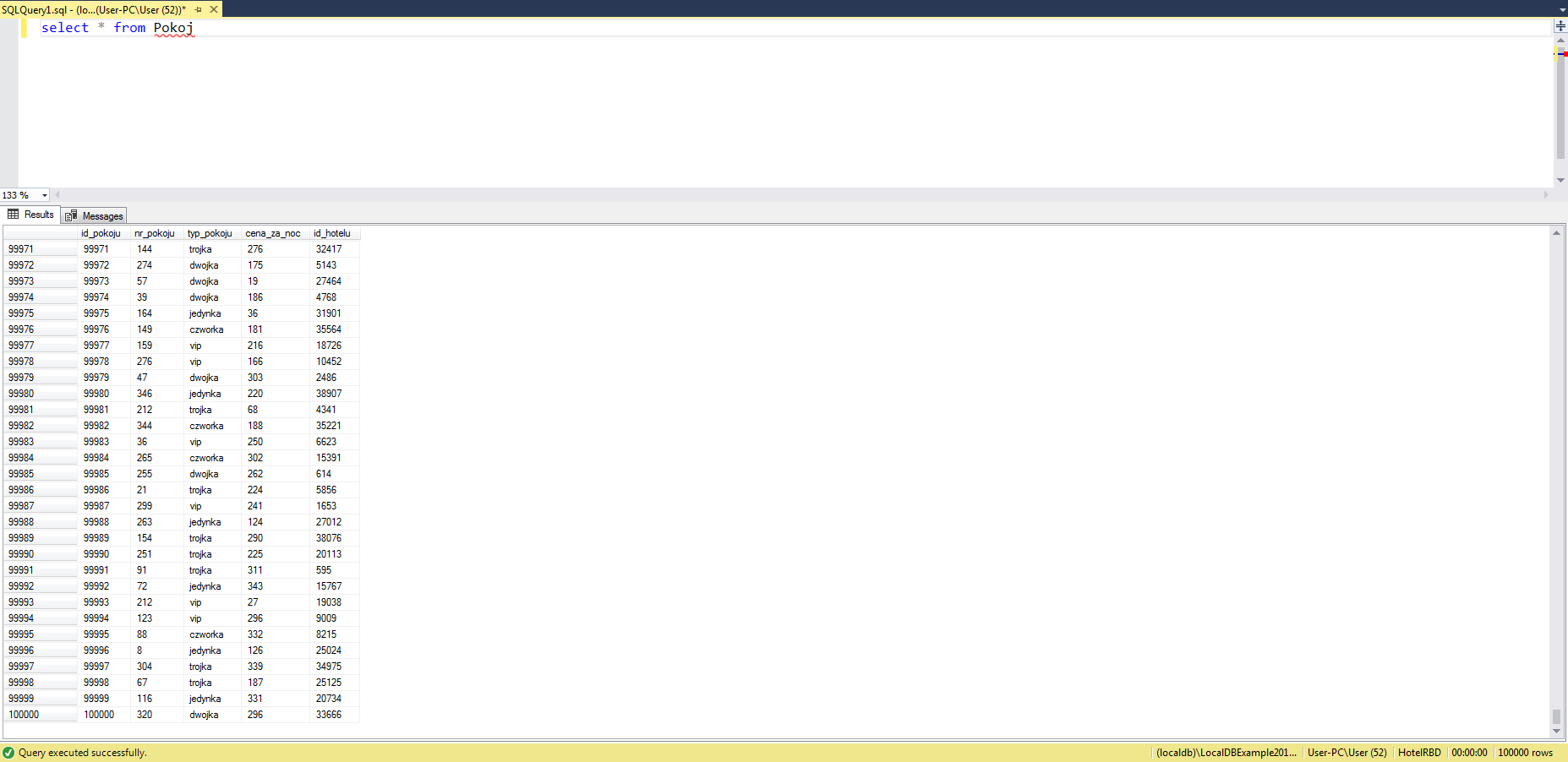
Pracownik



Hotel

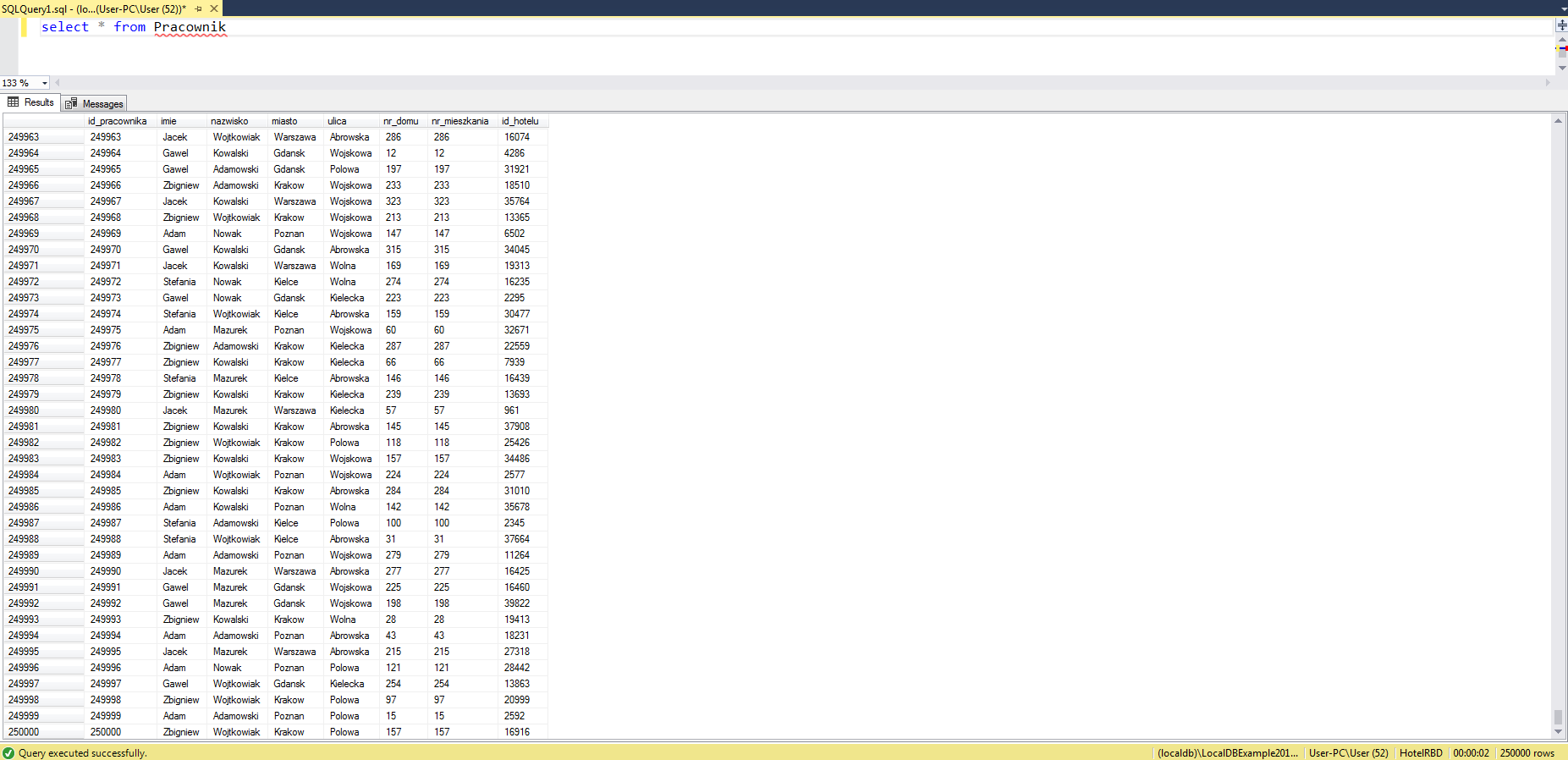


Pokoj

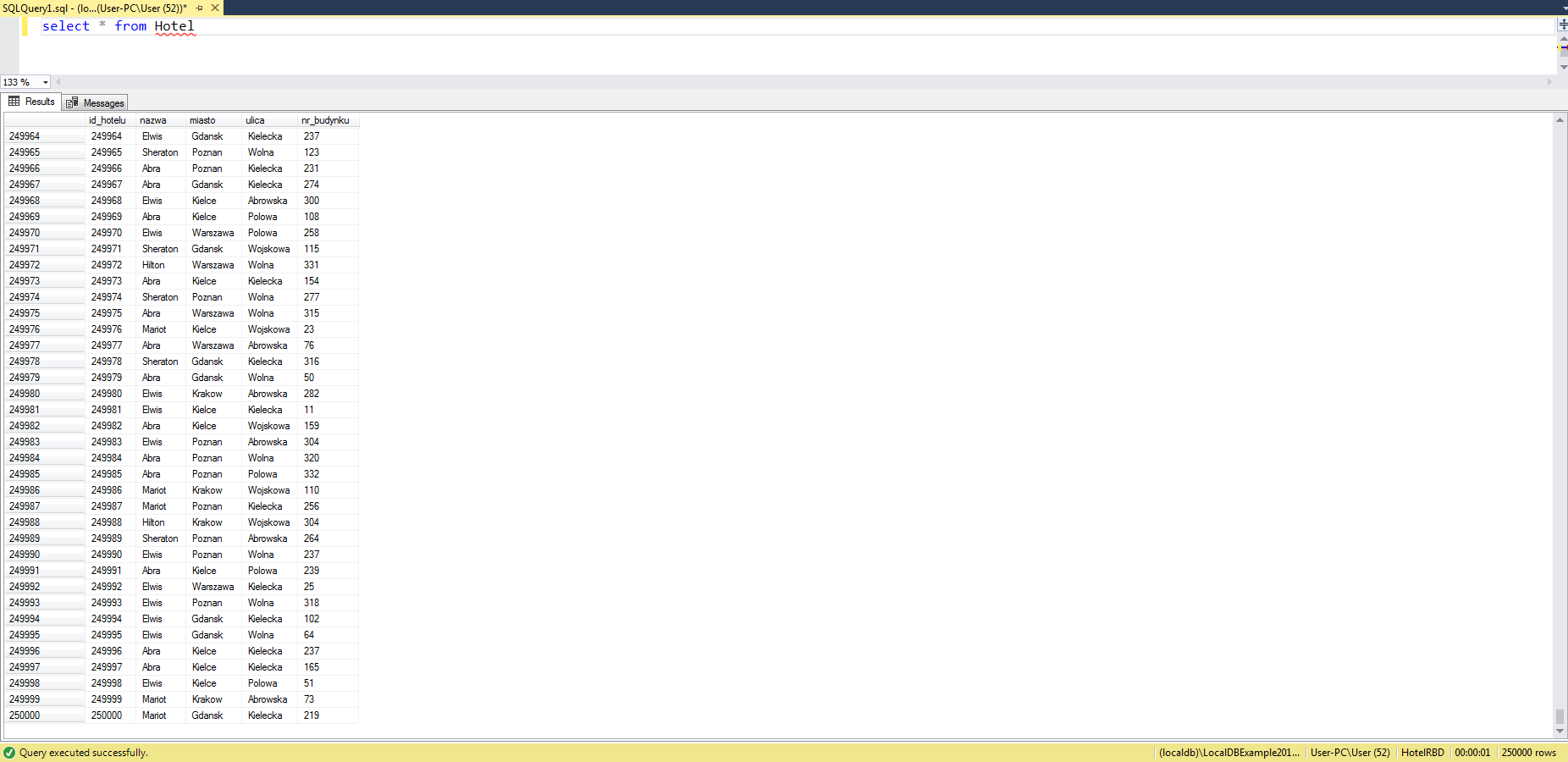


**Wynik dla 250k rekordów:**

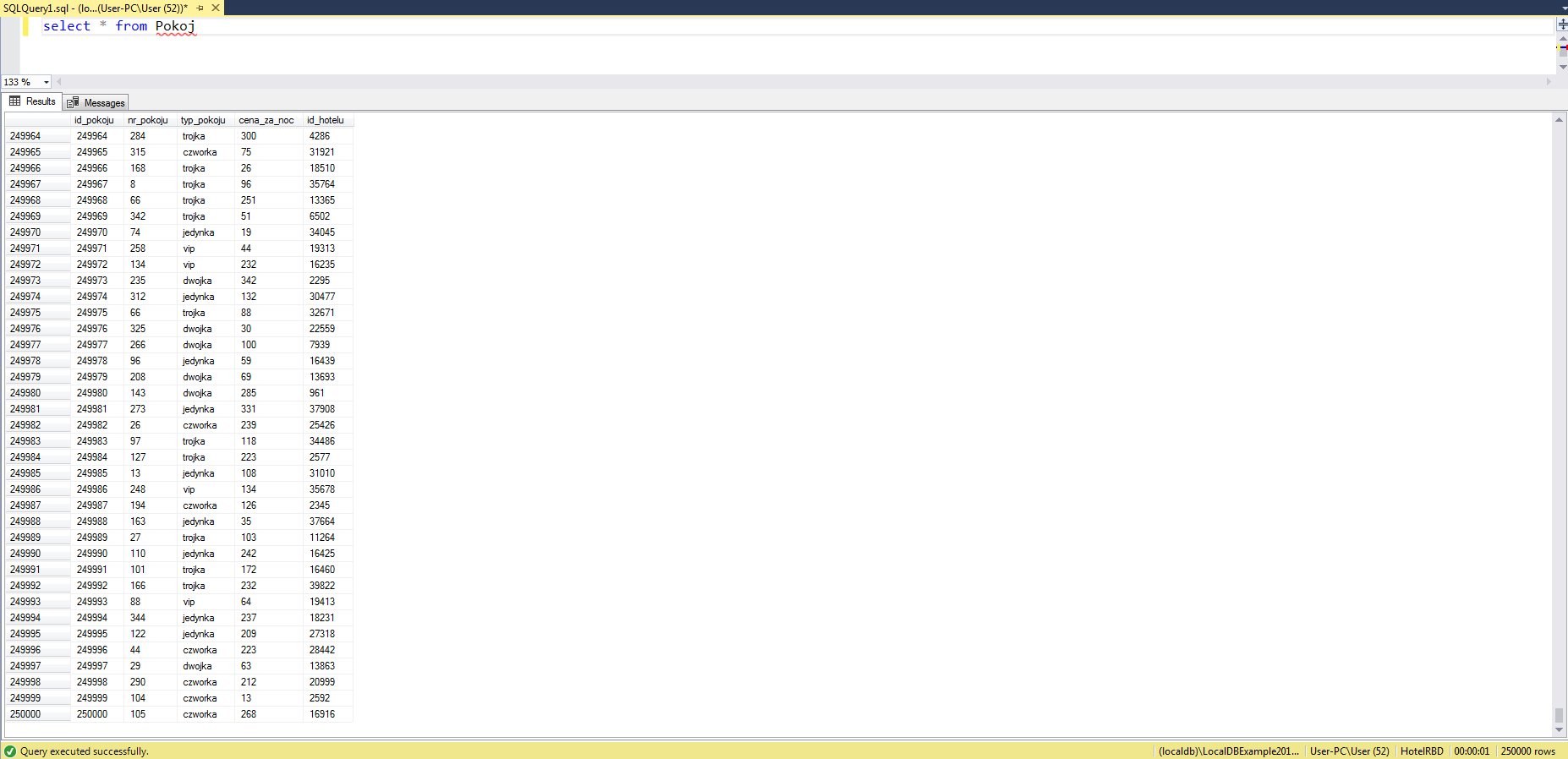
Pracownik



Hotel

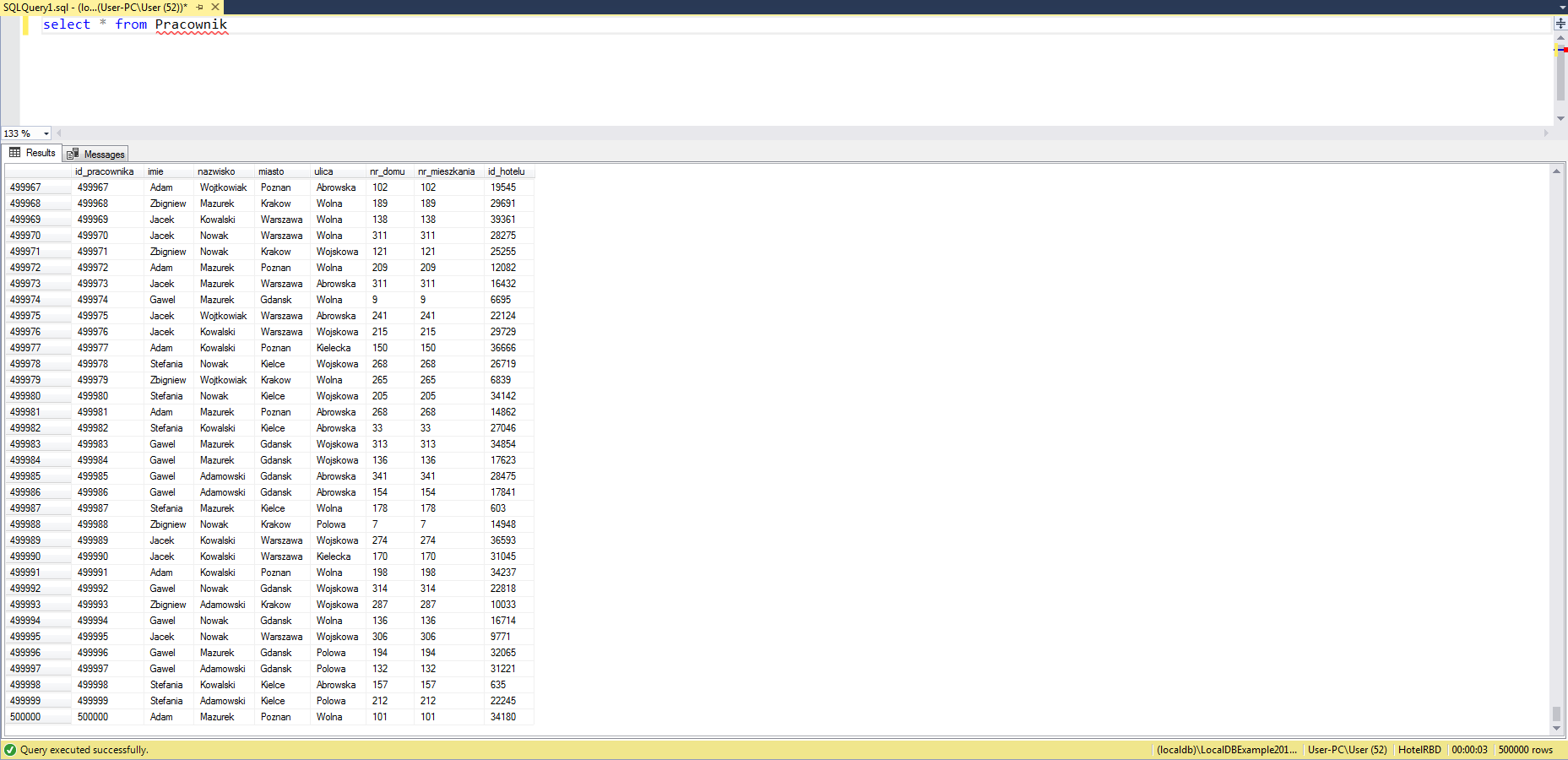


Pokoj

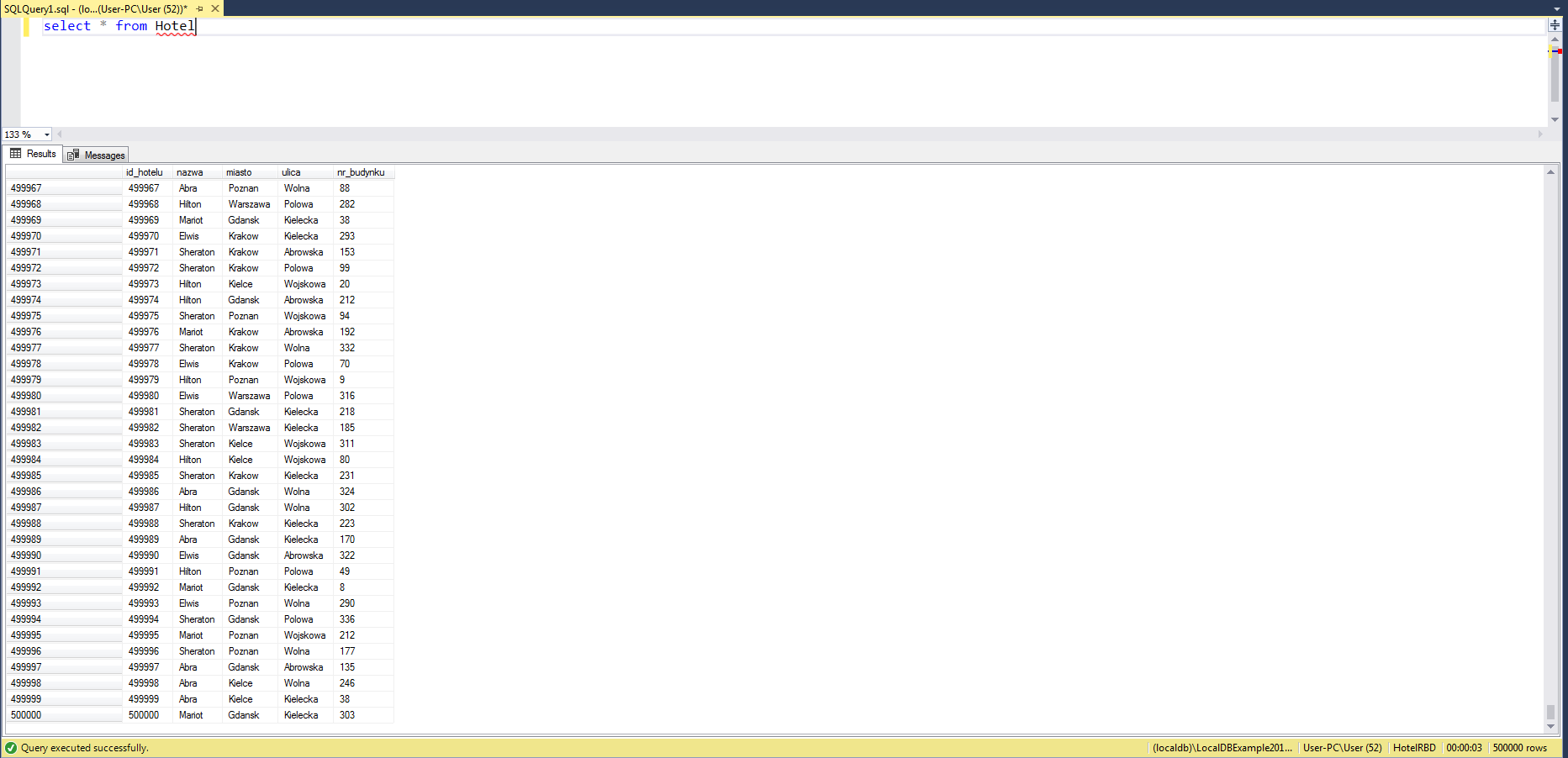


**Wynik dla 500k rekordów:**

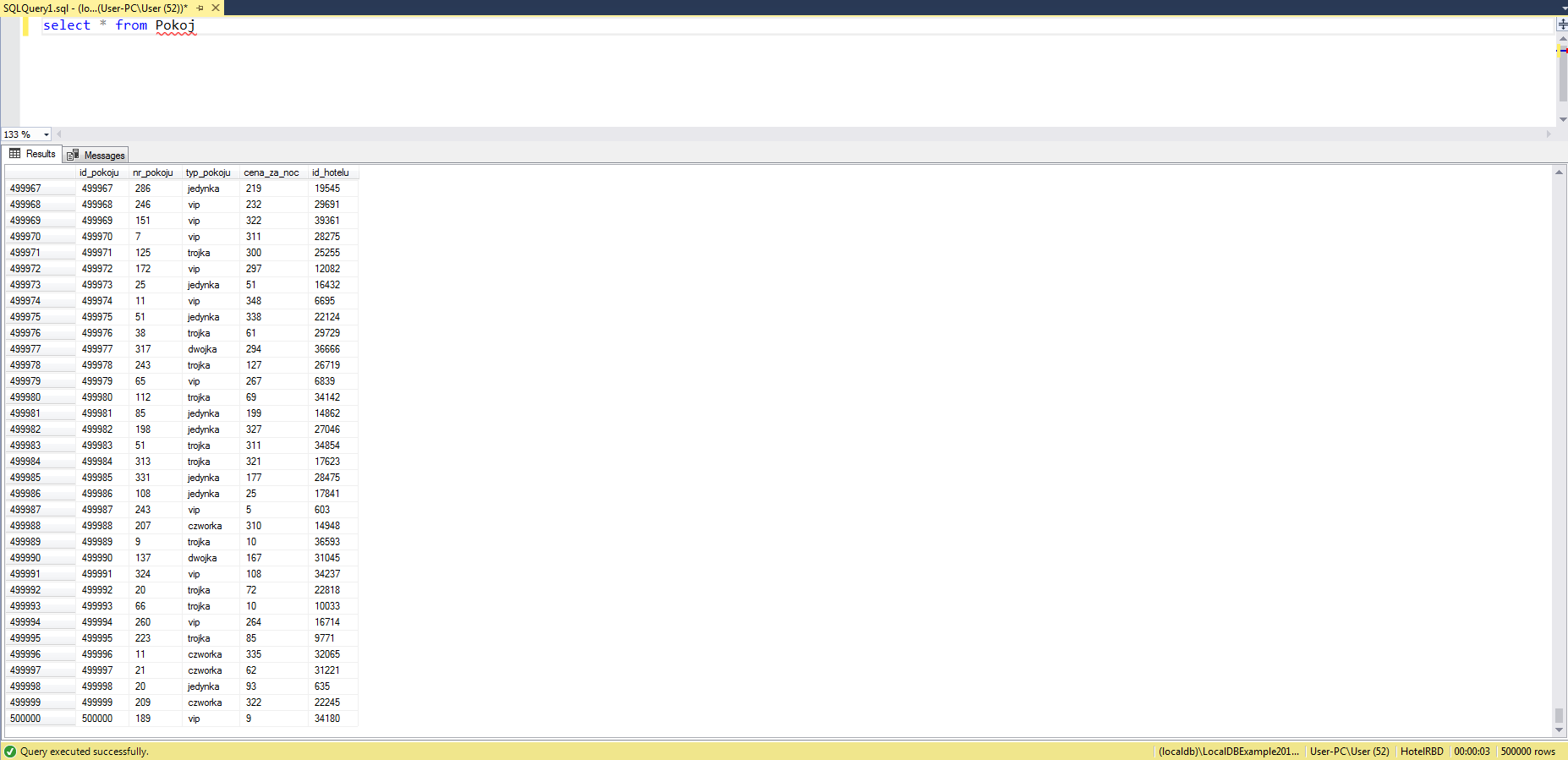
Pracownik



Hotel



Pokoj



1. Zapytanie do bazy danych MS SQL 2017

select

imie as 'Imie',

nazwisko as 'Nazwisko',

Pracownik.miasto+ ' ' + Pracownik.ulica + ' ' + cast(Pracownik.nr\_mieszkania as varchar(2)) as 'Adres pracownika' ,

h.id\_hotelu,

h.nazwa as 'Nazwa hotelu',

count(id\_pokoju) as 'Liczba rekordow id\_pokoj'

from Pracownik

left join Hotel h on h.id\_hotelu=Pracownik.id\_hotelu

left join Pokoj p on p.id\_hotelu=h.id\_hotelu

where

h.id\_hotelu>5 and

p.id\_pokoju>5 and

p.cena\_za\_noc>50

group by

Pracownik.imie,

h.nazwa,

Pracownik.nazwisko,

h.id\_hotelu,

h.nazwa,

Pracownik.miasto,

Pracownik.ulica,

Pracownik.nr\_mieszkania,

p.id\_pokoju,

p.cena\_za\_noc

having h.nazwa='Abra' or h.nazwa='Hilton' or h.nazwa='Mariot' or h.nazwa='Elwis'

order by

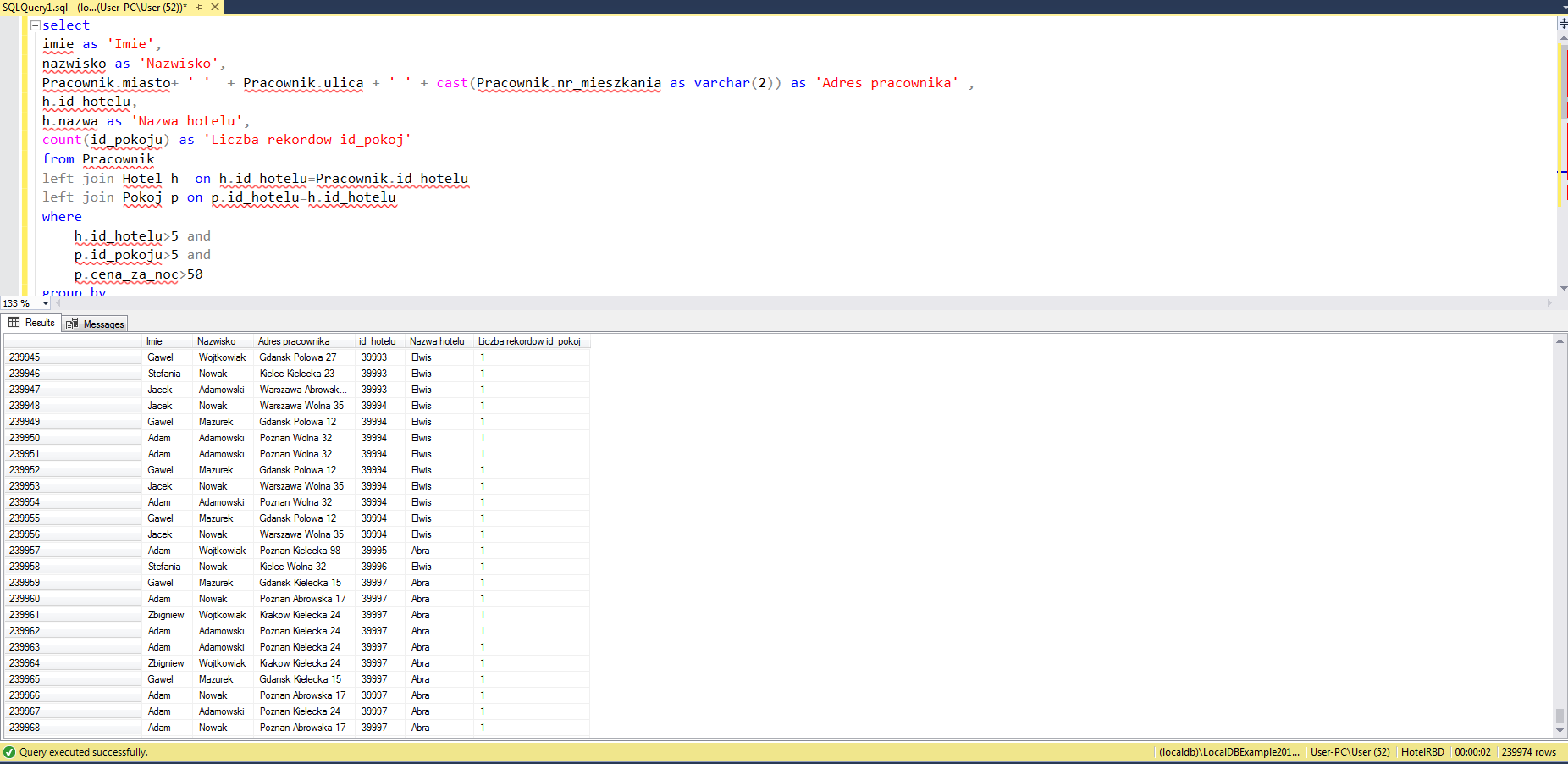
h.id\_hotelu ASC,

p.id\_pokoju ASC,

p.cena\_za\_noc DESC

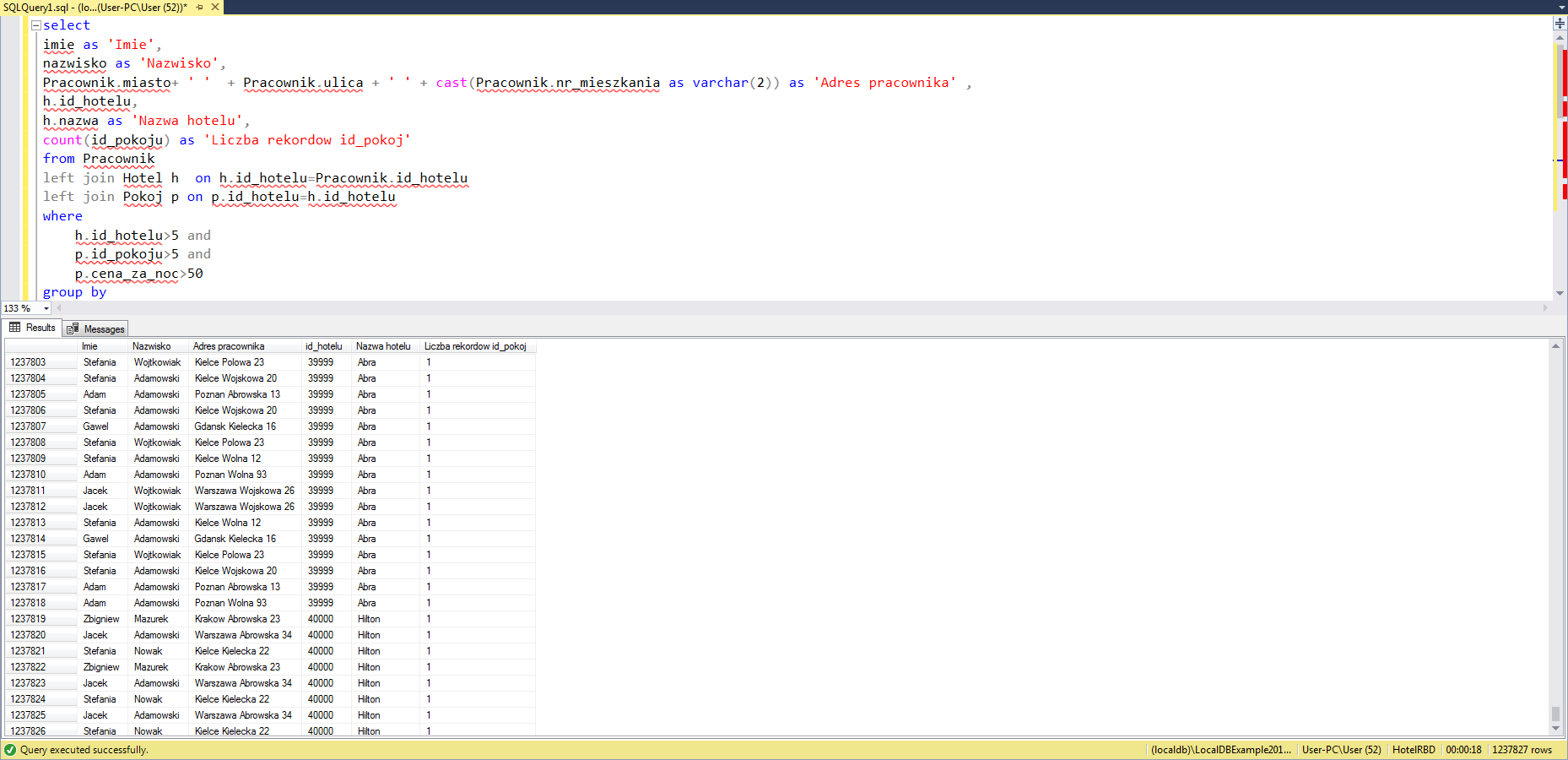
Wynik zapytania bazy danych:

* Dla 100k rekordów



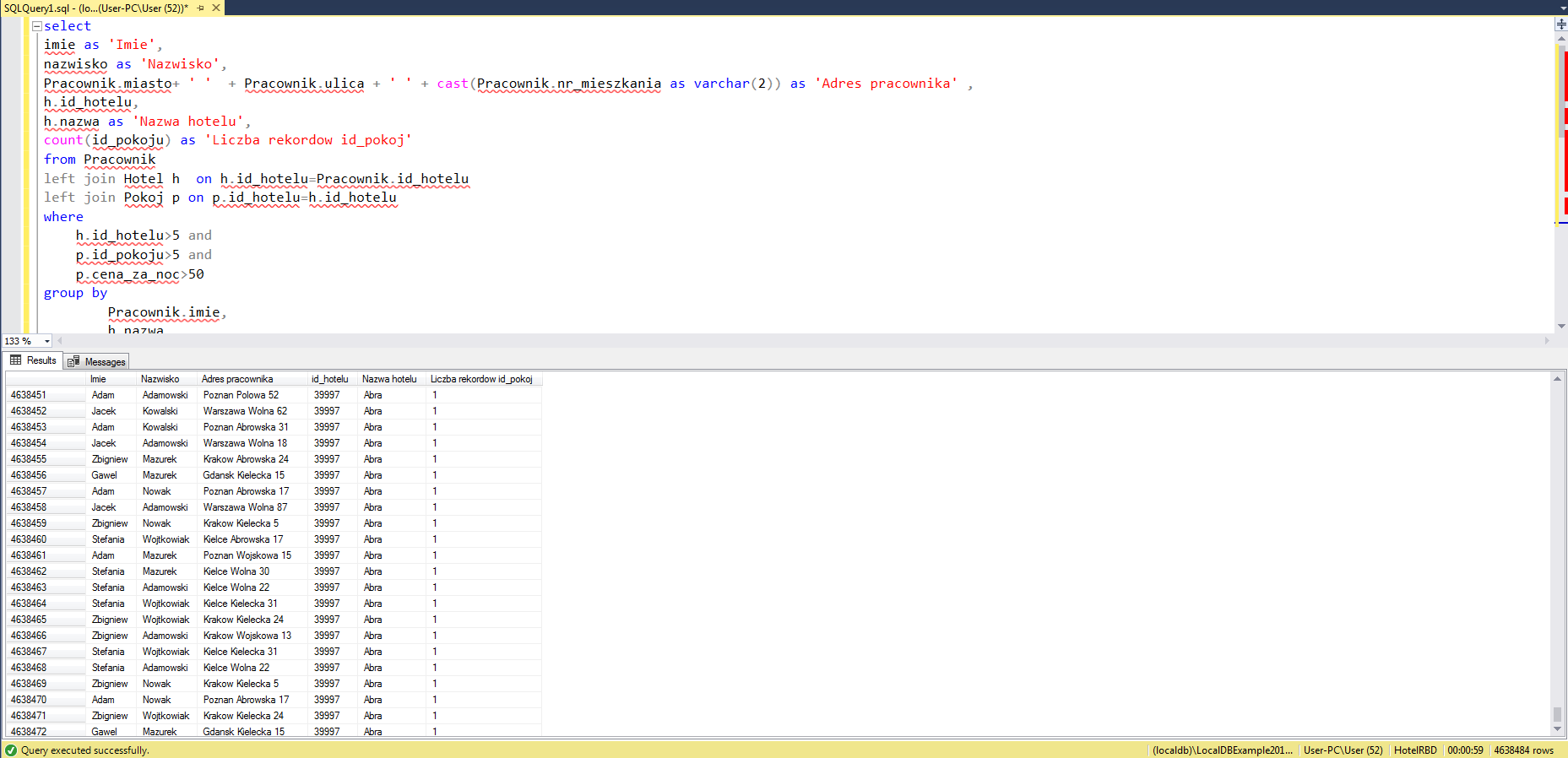
Czas wykonania: 0,02s

* Dla 250k rekordów



Czas wykonania: 0,18s

* Dla 500k rekordów

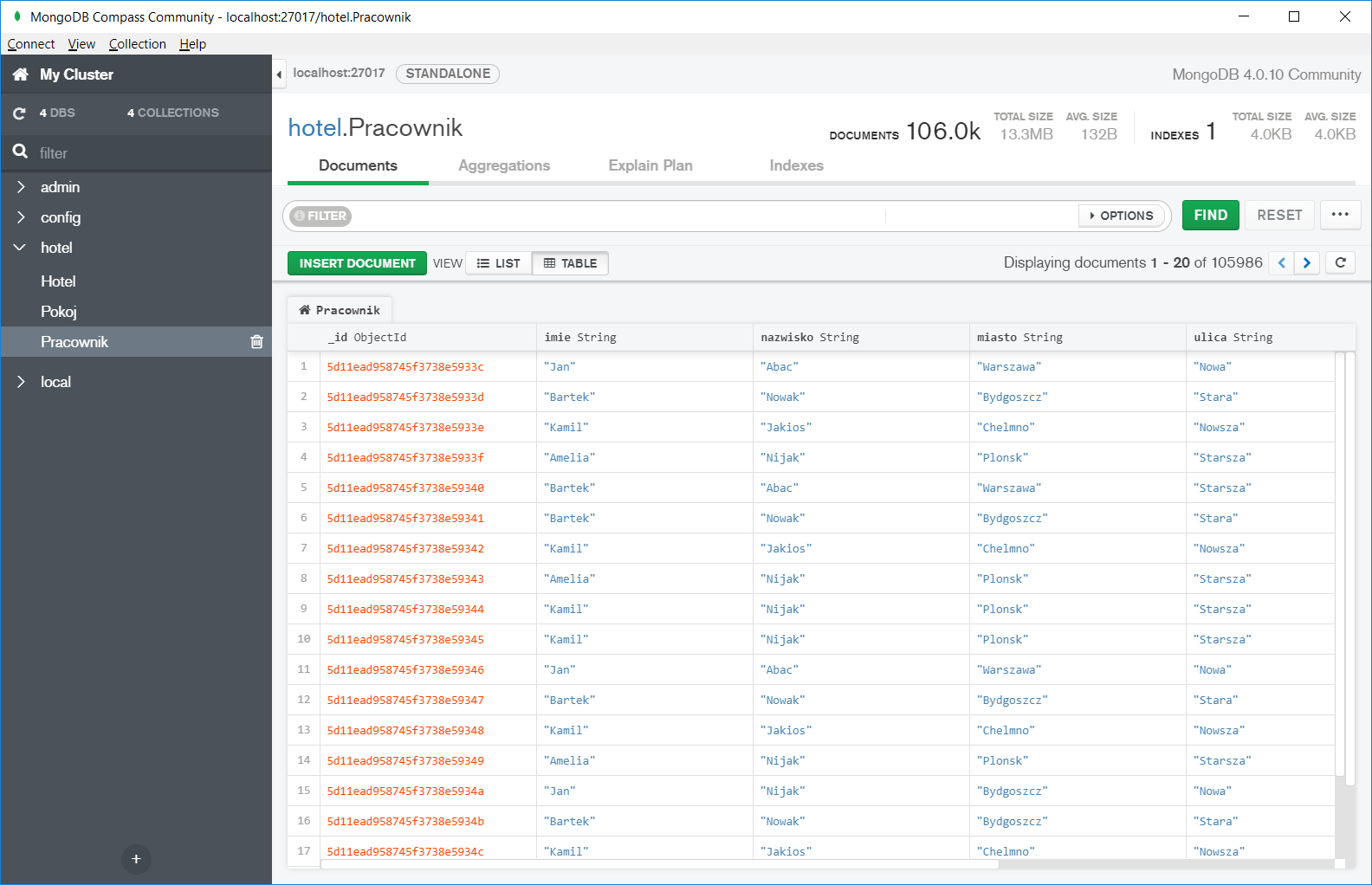


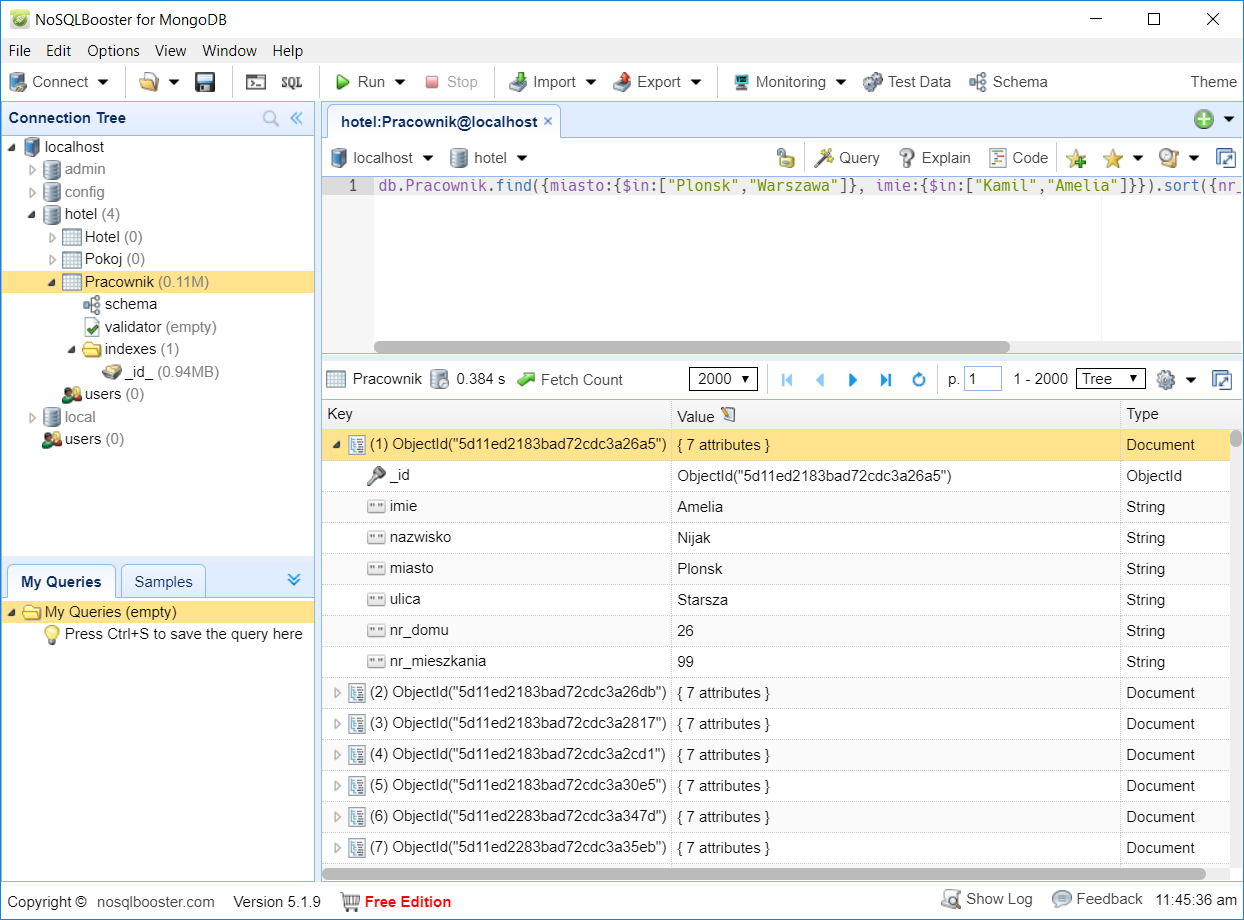
Czas wykonania: 0,59s

1. Zapytanie do bazy danych MongoDB

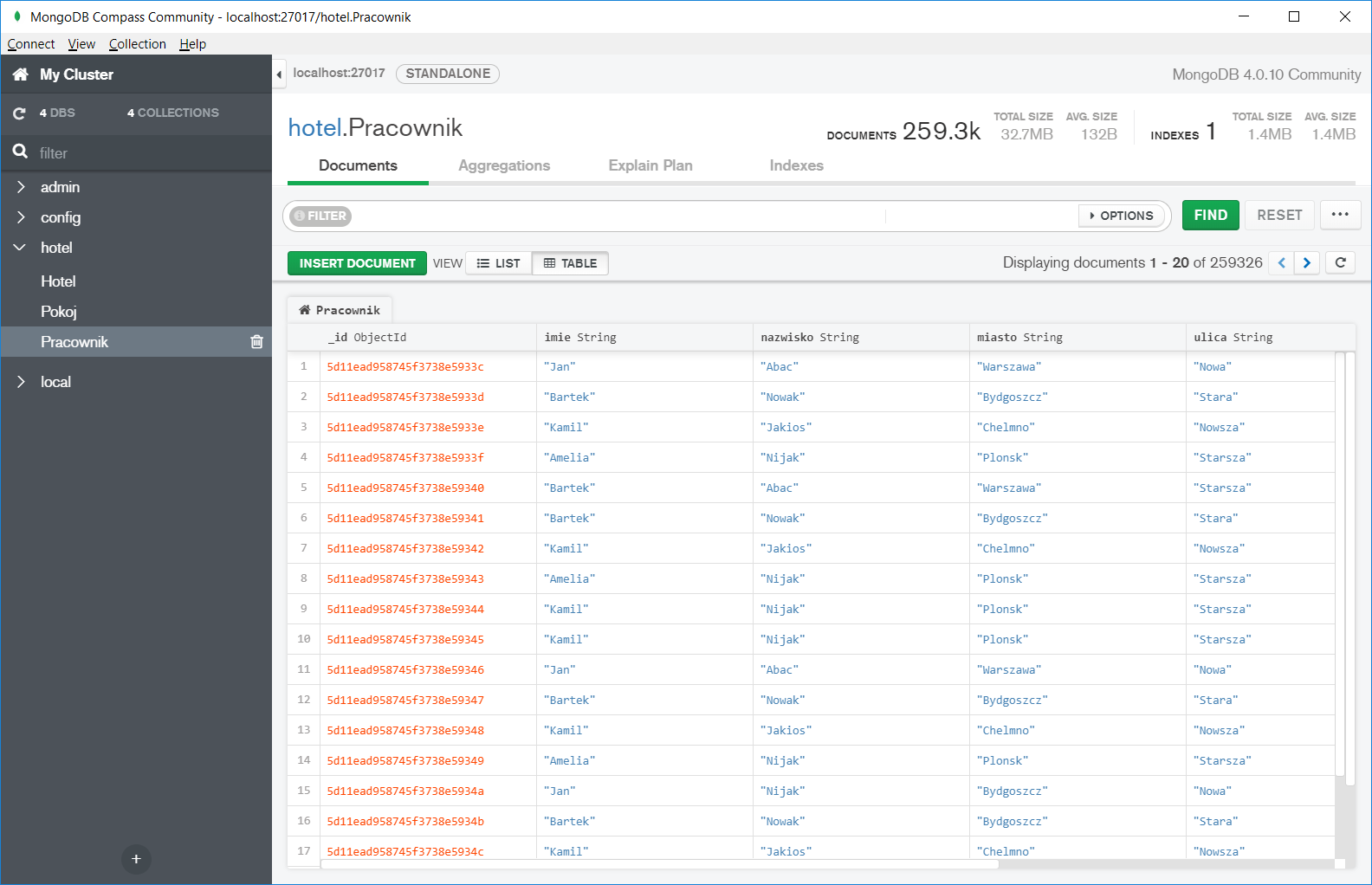
**db.Pracownik.find({miasto:{$in:["Plonsk","Warszawa"]}, imie:{$in:["Kamil","Amelia"]}}).sort({nr\_domu:-1}).sort({nr\_mieszkania:-1})**

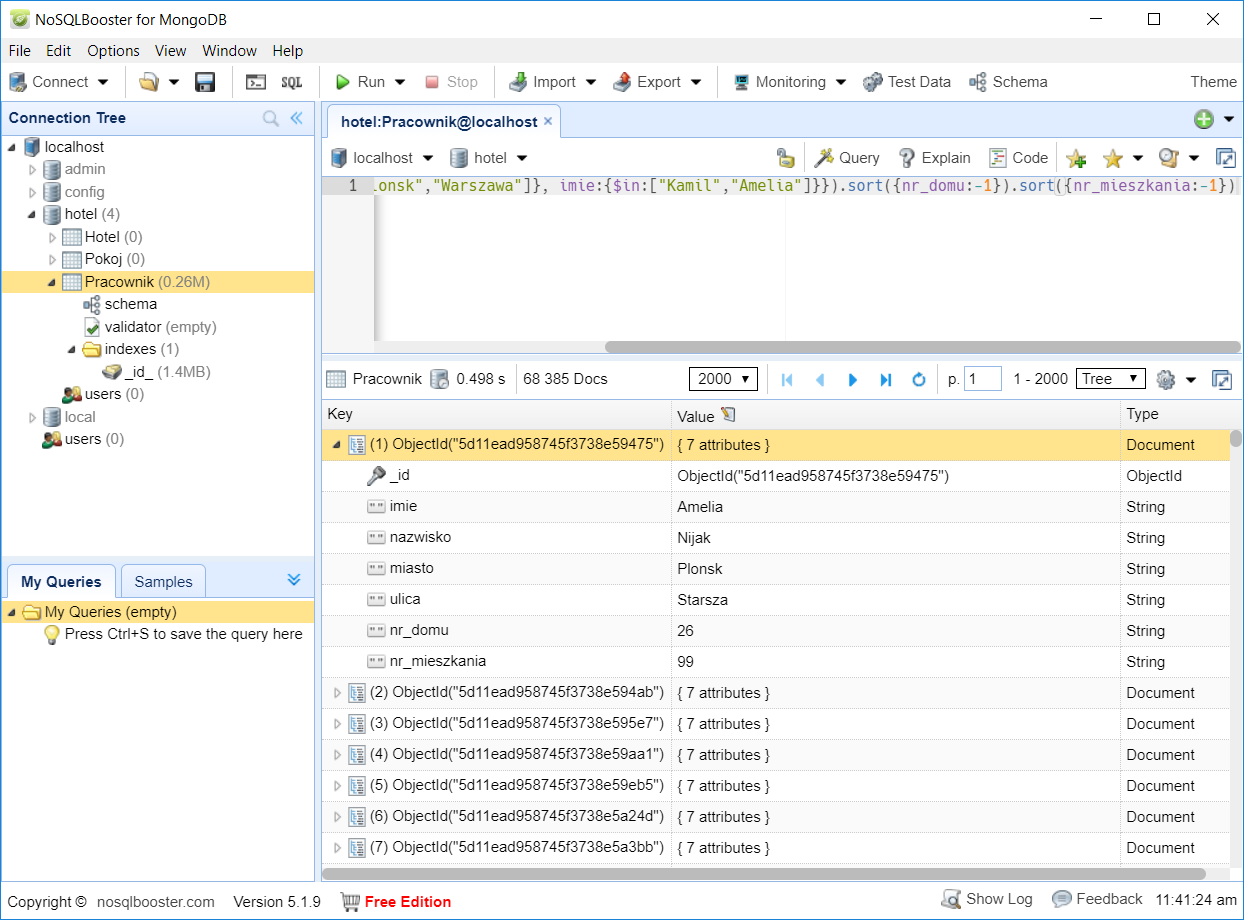
1. Wyniki wykonania zapytań w MongoDB



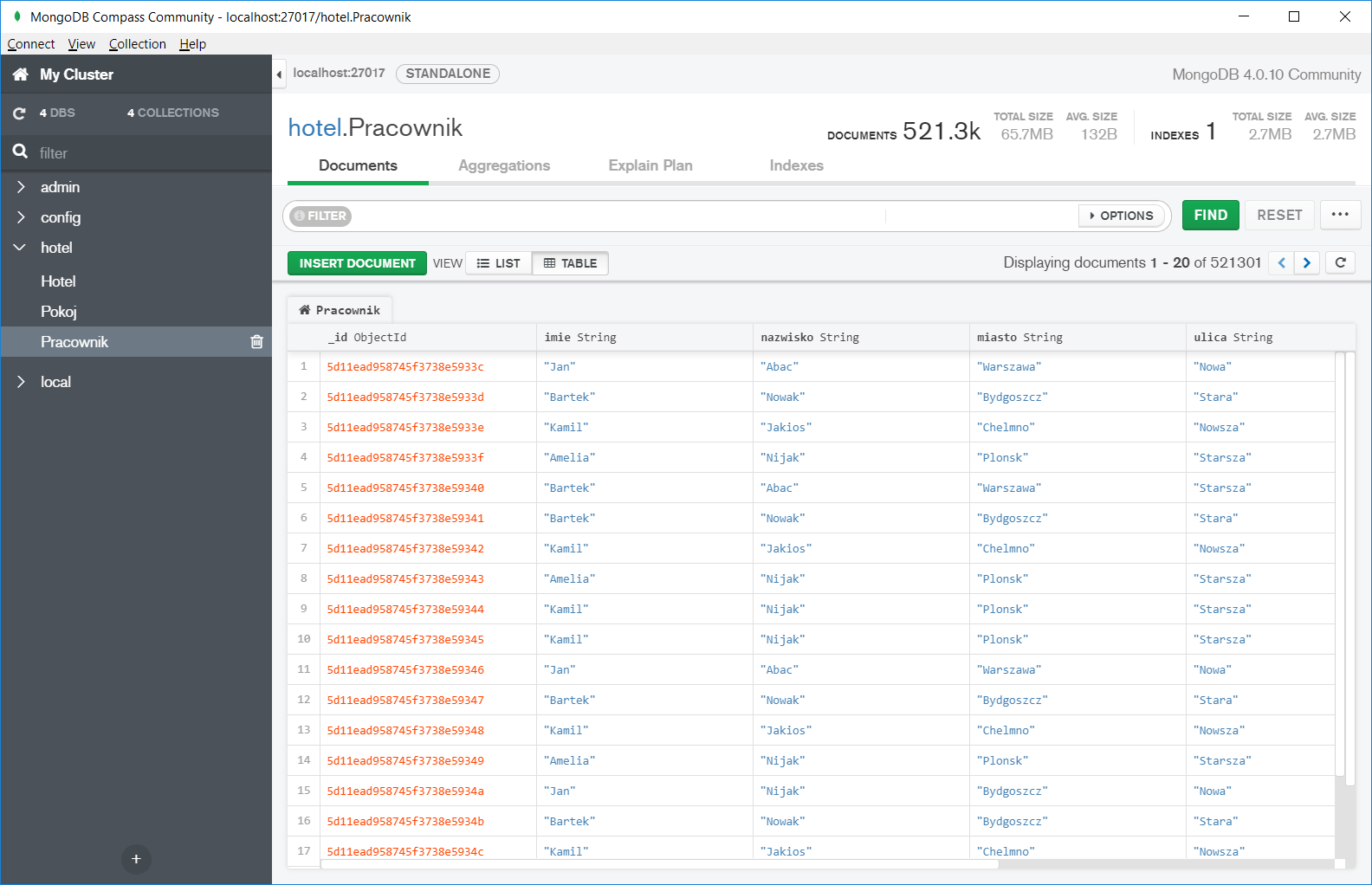


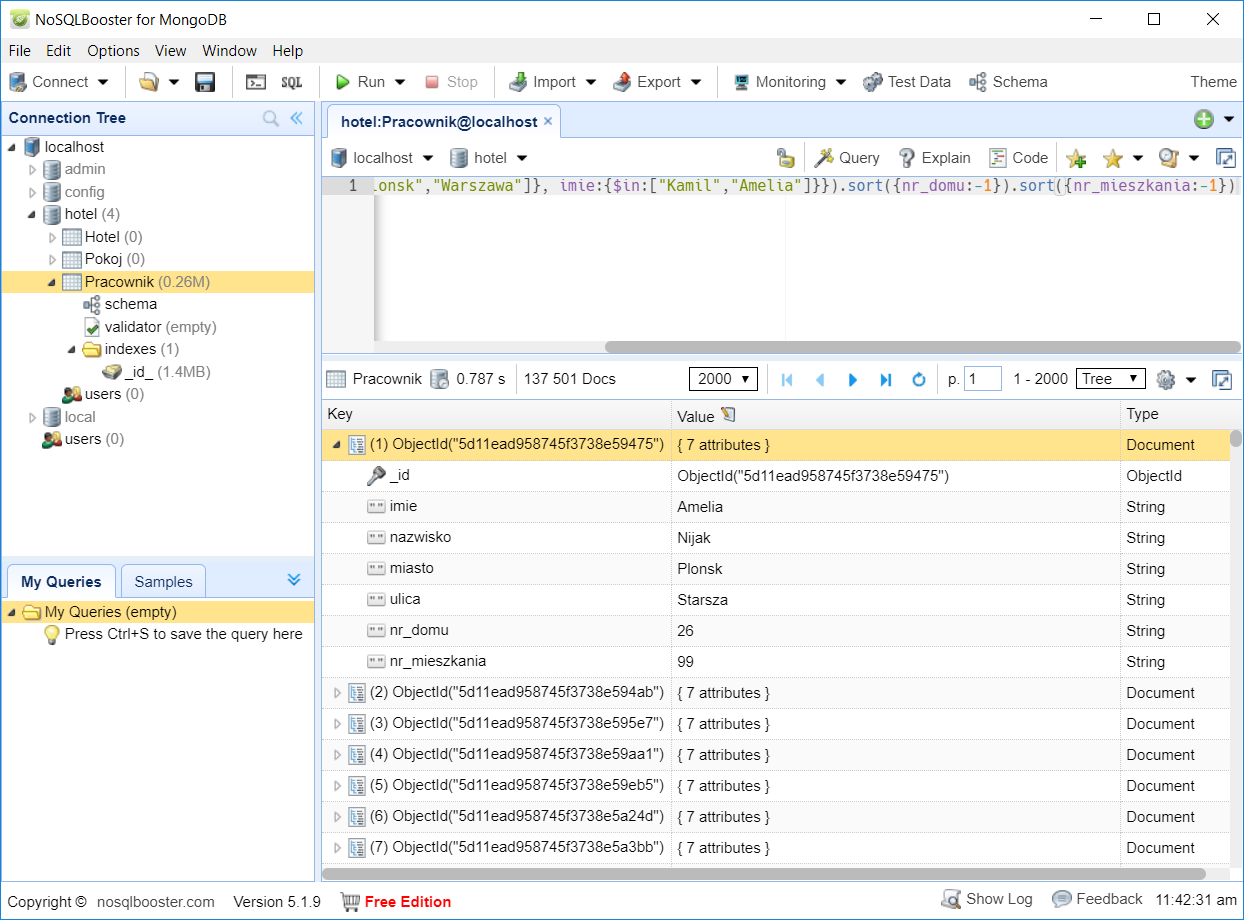
Wyniki dla 100 tys. Rekordów





Wyniki dla 250 tys. Rekordów





Wyniki dla 500 tys. Rekordów

1. Porównanie wyników bazy MS SQL 2017 oraz MongoDB

Na podstawie przeprowadzonych badań można stwierdzić, że w przypadku bazy noSQL mamy bardzo niewielki przyrost czasu wykonania zapytania mimo znacznego przyrostu danych. Dla bazy relacyjnej czas wykonania zapytania rośnie gwałtownie wraz ze wzrostem danych w bazie.

Podsumowanie:

Napotkane problemy:

-trudność ze zbudowaniem złożonego zapytania w MongoDB,

-nieintuicyjne środowisko MongoDB,

-długi czas oczekiwanie podczas generowania rekordów w obu bazach.

-mimo utworzonego pliku .csv o zadeklarowanej ilości danych, zgodnej z poleceniem, ilość wygenerowanych danych różniła się od oczekiwanej

-zapytania w MongoDB udało nam się zrobić bez łączenia tabel, działając tylko na jednej

Ocena środowiska:

Środowisko MongoDB umożliwiło zbudowanie bazy danych NoSQL. Baza danych jest zorientowana na dokumenty, co jest zupełnie innym podejściem niż przy bazach relacyjnych. MongoDB nie wspiera również two-phase-commit (bez wsparcia dla triggerów i kluczy obcych). Wykorzystanie podejścia noSQL takiego jak w MongoDB będzie bardzo dobrym pomysłem, zwłaszcza jeśli będziemy przechowywać znaczną ilość danych. W porównaniu z bazą relacyjną ma też takie zalety jak brak sztywnego modelu danych, co pozwala wprowadzać zmiany. Dodatkowo istnienie Środowiska MongoDB Compass znacznie ułatwia pracę. Program jest dość intuicyjny i na jego zrozumienie potrzeba mniej czasu niż w przypadku projektowania relacyjnej bazy danych.