**Projekt bazy danych dla hotelu**

Przedmiot: **Bazy danych**

Prowadzący: **dr inż. Arkadiusz Mirakowski**

Gdańsk, 2022

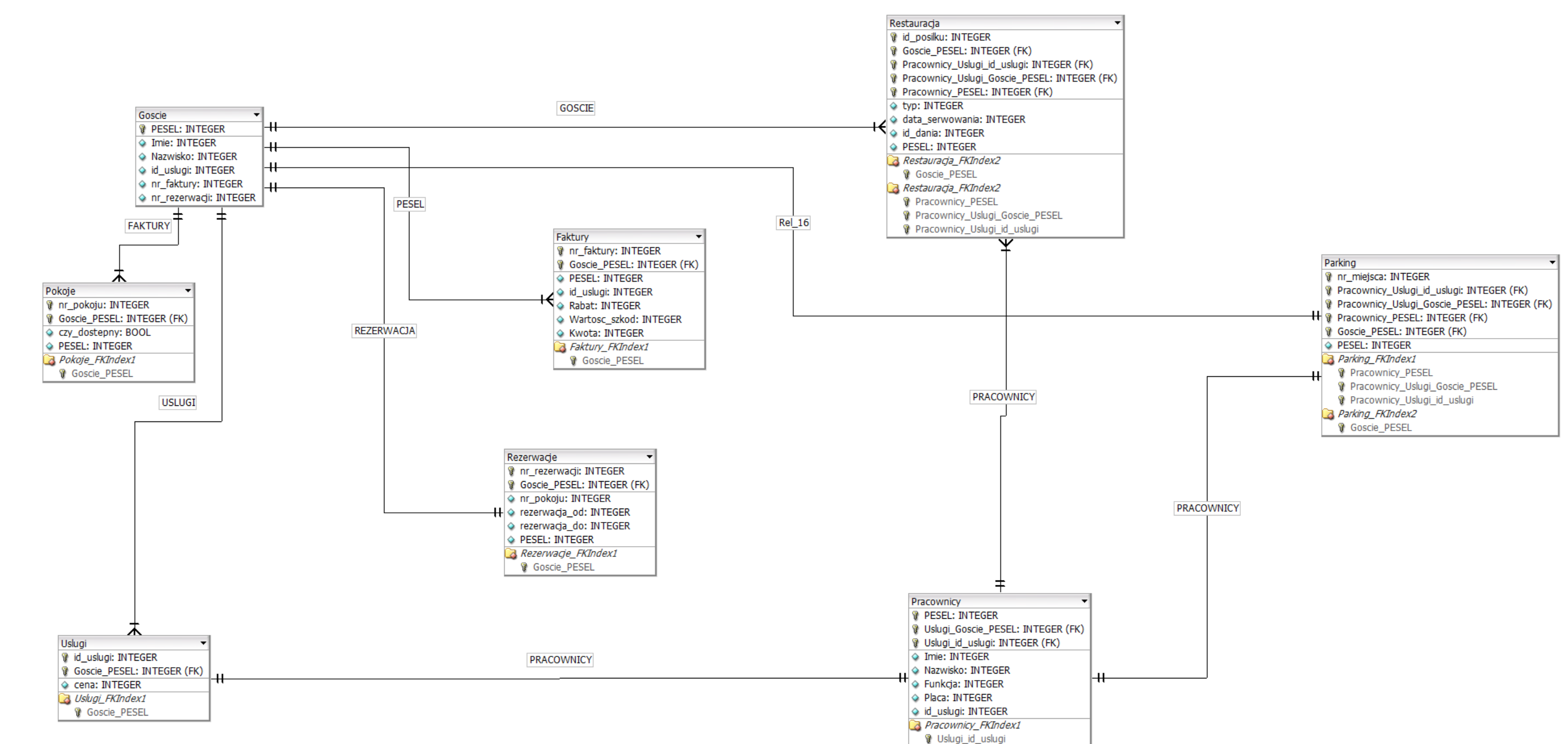
1. **Opis bazy danych**

**W tym punkcie umieszczamy w formie opisu jakie jest przeznaczenie bazy danych, co baza danych umożliwia, jakich danych dostarcza itd.**

Baza danych przedstawia informacje na temat hotelu. Podstawowym rekordem łączącym encje jest pesel gościa bądź pracownika. Stworzone tabele sięgają zasięgiem wiele usług hotelowych. Baza mogłaby stanowić szkielet serwera bądź datebase’a na recepcji.

1. **Diagram związków encji – DBDesigner (min. 8 encji)**

W tym punkcie umieszczamy diagram związków encji wykonany w DBDesigner4, zgodnie z poniższym wzorem:



Rys.1. Diagram związków encji – DBDesigner4

1. **Omówienie tabel (min. 8 tabel)**
   1. **Goście** – tabela zawiera dane osobowe gości, wraz z usługami jakie zamawiają, numerem wynajmowanego pokoju, faktury oraz rezerwacji
   2. **Pracownicy** – tabela zawiera dane osobowe pracowników, jaką funkcje pełnią i jakie dostają wynagrodzenie
   3. **Pokoje** – tabela zawiera numer danego pokoju oraz wartość bool czy jest on aktualnie dostępny
   4. **Restauracja** – hotel serwuje w swojej restauracji codziennie trzy posiłki. Tabela zawiera unikalny identyfikator każdego posiłku, informacje jaki to jest posiłek (śniadanie, obiad kolacja) oraz informacje o nim.
   5. **Faktury** – tabela zawiera dane o fakturach wypisywanych przez hotel dla gości
   6. **Rezerwacje** – tabela zawiera dane o rezerwacjach w hotelu.
   7. **Usługi** – tabela zawiera dane o usługach prowadzonych w hotelu, takich jak fryzjer, spa, salon gier. Rekordy to identyfikator usługi oraz cena
   8. **Parking –** tabela zawiera informacje o parkingu
2. **Definicje tabel**

W tym punkcie umieszczamy definicje wszystkich encji wykorzystanych w projekcie, zgodnie z poniższym wzorem:

|  |  |  |
| --- | --- | --- |
| nazwa atrybutu | typ atrybutu | inne cechy atrybutu |

Goście:

|  |  |  |
| --- | --- | --- |
| PESEL | varchar(30) | PK |
| Imie | varchar(30) | - |
| Nazwisko | varchar(30) | - |
| Id\_uslugi | varchar(30) | FK |
| Nr\_faktury | varchar(30) | FK |
| Nr\_rezerwacji | varchar(30) | FK |

Pracownicy:

|  |  |  |
| --- | --- | --- |
| PESEL | Varchar(30) | PK |
| Imie | Varchar(30) | - |
| Nazwisko | Varchar(30) | - |
| Funkcja | Varchar(30) | - |
| Placa | Int | - |
| Id\_uslugi | Varchar(30) | FK |

Pokoje:

|  |  |  |
| --- | --- | --- |
| Nr\_pokoju | Int | PK |
| Czy\_dostepny | Bool | - |
| PESEL | Varchar(30) | FK |

Restauracja:

|  |  |  |
| --- | --- | --- |
| Id\_posilku | Int | PK |
| Typ | Varchar(30) | - |
| Data\_serwowania | Date | - |
| Id\_dania | Int | FK |
| PESEL | Varchar(30) | FK |

Faktury:

|  |  |  |
| --- | --- | --- |
| Nr\_faktury | Varchar(30) | PK |
| PESEL | Varchar(30) | FK |
| Rabat | Int | - |
| Wartosc\_szkod | Int |  |
| Kwota | Int | - |

Rezerwacje:

|  |  |  |
| --- | --- | --- |
| Nr\_rezerwacji | Varchar(30) | PK |
| Nr\_pokoju | Int | FK |
| Rezerwacja\_od | Date | - |
| Rezerwacja\_do | Date | - |
| PESEL | Varchar(30) | FK |

Usługi:

|  |  |  |
| --- | --- | --- |
| Id\_uslugi | Varchar(30) | PK |
| PESEL | Varchar(30) | FK |
| Cena | Int | - |

Parking:

|  |  |  |
| --- | --- | --- |
| Nr\_miejsca | Varchar(3) | PK |
| PESEL | Varchar(30) | VK |

1. **Zawartość tabel**

W tym punkcie umieszczamy wszystkie tabele wraz z rekordami, zgodnie z poniższym wzorem:

**Nazwa\_tabeli\_1**

|  |  |  |
| --- | --- | --- |
| kolumna 1 | kolumna 2 | kolumna 3 |
| wartość 1 | wartość 2 | wartość 3 |
| wartość 4 | wartość 5 | wartość 6 |

**Goście:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PESEL** | **Imię** | **Nazwisko** | **Id\_uslugi** | **Nr\_faktury** | **Nr\_rezerwacji** |
| 0228900062 | Jan | Kowalski | 00 | 1234 | 889 |
| 9928444754 | Maciej | Jankowski | 00 | 1123 | 888 |
| 0029476843 | Bartek | Studniarek | 01 | 1111 | 887 |

**Pracownicy:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **PESEL** | **Imię** | **Nazwisko** | **Funkcja** | **Placa** | **Id\_uslugi** |
| 01223990341 | Karol | Malinowski | Masażysta | 3200 | 04 |
| 01923459012 | Jan | Maciejewski | Fryzjer | 3200 | 03 |
| 03994022222 | Karolina | Malec | Kosmetyczka | 3100 | 05 |

**Pokoje:**

|  |  |  |
| --- | --- | --- |
| **Nr\_pokoju** | **PESEL** | **Czy\_dostępny** |
| 101 | 0228900062 | NIE |
| 102 | 9928444754 | NIE |
| 103 | 0029476843 | NIE |

**Restauracja:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id\_posilku** | **Typ** | **Data\_serwowania** | **Id\_dania** | **PESEL** |
| 302 | Śniadanie | 01-02-2021 | 11 | 02289000622 |
| 303 | Śniadanie | 01-02-2021 | 11 | 9928444754 |
| 304 | Śniadanie | 01-02-2021 | 11 | 0029476843 |
| 305 | Lunch pracowniczy | 01-02-2021 | 73 | 01223990341 |

**Faktury:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Nr\_faktury | PESEL | Id\_uslugi | Rabat | Wartosc\_szkod | Kwota |
| 1234 | 0228900062 | 00 | 0 | 0 | 600 |
| 1123 | 9928444754 | 00 | 0 | 300 | 900 |
| 1111 | 0029476843 | 01 | 100 | 0 | 260 |

**Rezerwacje:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nr\_rezerwacji** | **Nr\_pokoju** | **Rezerwacja\_od** | **Rezerwacja\_do** | **PESEL** |
| 889 | 101 | 12-31-2020 | 01-02-2021 | 0228900062 |
| 888 | 102 | 12-30-2020 | 01-03-2021 | 9928444754 |
| 887 | 103 | 12-23-2020 | 01-02-2021 | 0029476843 |

**Usługi:**

|  |  |  |
| --- | --- | --- |
| **Id\_uslugi** | **Cena** | **PESEL** |
| 01 | 60 | 0029476843 |
| 03 | 120 | - |
| 04 | 160 | - |

**Parking:**

|  |  |
| --- | --- |
| **Nr\_miejsca** | **PESEL** |
| 001 | 0228900062 |
| 002 | 9928444754 |
| 003 | 0029476843 |
| 004 | 0122399034 |

1. **Relacje bazodanowe**

W tym punkcie umieszczamy kody źródłowe T-SQL związane z utworzeniem wszystkich relacji w bazie danych na podstawie diagramu encji z p. 2 według poniższego wzoru:

**Relacja 1 – 1:1** – nazwa\_tabeli\_1 – nazwa\_tabeli\_2

* 1. kod T-SQL
  2. kod T-SQL

Kod T-SQL

Courier New, 8

* 1. kod T-SQL
  2. …

**Relacja 2 – 1:n** – nazwa\_tabeli\_3 – nazwa\_tabeli\_4

1. kod T-SQL
2. kod T-SQL

Kod T-SQL

Courier New, 8

1. kod T-SQL
2. …

**Relacja 1 – 1:n** – Goście – Pokoje

Alter table Pokoje

add CONSTRAINT fk\_pok\_pesel foreign key (PESEL)

references Goscie (PESEL)

**Relacja 2 – 1:n** – Goście – Faktury

Alter table Faktury

add CONSTRAINT fk\_fakt\_pesel foreign key (PESEL)

references Goscie (PESEL)

**Relacja 3 – 1:1** – Goście – Rezerwacje

Alter table Rezerwacje

add CONSTRAINT fk\_rezerwacje\_pesel foreign key (PESEL)

references Goscie (PESEL)

**Relacja 4 – 1:n** – Goście – Restauracja

Alter table Restauracja

add CONSTRAINT fk\_restauracja\_pesel foreign key (PESEL)

references Goscie (PESEL)

**Relacja 5 – 1:1** – Goście – Parking

Alter table Parking

add CONSTRAINT fk\_parking\_pesel foreign key (PESEL)

references Pracownicy (PESEL)

**Relacja 6 – 1:n** – Goście – Usługi

Alter table Uslugi

add CONSTRAINT fk\_uslugi\_pesel foreign key (PESEL)

references Goscie (PESEL)

**Relacja 7 – 1:n** – Pracownicy – Parking

Alter table Parking

add CONSTRAINT fk\_parking\_pesel foreign key (PESEL)

references Pracownicy (PESEL)

**Relacja 8 – 1:n** – Pracownicy - Restauracja

Alter table Restauracja

add CONSTRAINT fk\_restauracja\_pesel foreign key (PESEL)

references Pracownicy (PESEL)

1. **Omówienie i prezentacja widoków**

**goscieWPokojach**- jest to widok, który zestawia numer pokoju wraz z informacjami jego gościa i informacje, czy jest on gościem specjalnym (posiada rabat)

DROP VIEW IF EXISTS goscieWPokojach

GO

CREATE VIEW goscieWPokojach as

Select nr\_pokoju,Nazwisko,Imie,Goscie.PESEL,

case

when rabat>0 then 'posiada rabat'

else 'brak rabatu'

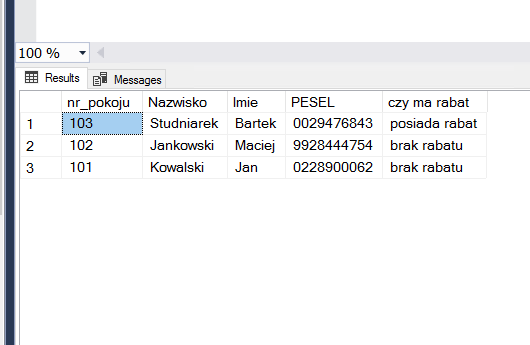
end as 'czy ma rabat'

from Pokoje,Goscie,Faktury

where Pokoje.PESEL=Goscie.PESEL and Pokoje.PESEL=Faktury.PESEL

GO

select\*from goscieWPokojach

****

Rys.2.Wynik działania goscieWPokojach

**parkingGoscie –** widok przedstawia informacje na temat gości,którzy zajmują dane miejsce na parkingu

DROP VIEW IF EXISTS parkingGoscie

GO

CREATE VIEW parkingGoscie as

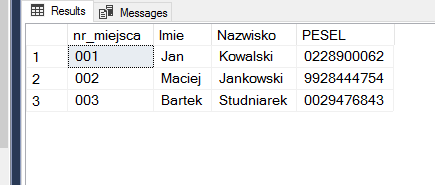
Select nr\_miejsca, Imie,Nazwisko,Parking.PESEL

from Goscie,Parking

where Parking.PESEL=Goscie.PESEL

GO

select\*from parkingGoscie



Rys.3.Wynik działania parkingGoscie

**zainteresowanieUsluga –** widok pokazuje kto korzystal a kto nie korzystal z usług hotelowych

DROP VIEW IF EXISTS zainteresowanieUsluga

GO

CREATE VIEW zainteresowanieUsluga as

Select Goscie.PESEL,Goscie.Imie,Goscie.Nazwisko,

case

when Faktury.id\_uslugi='00' then 'Nie korzystal'

else 'Korzystal'

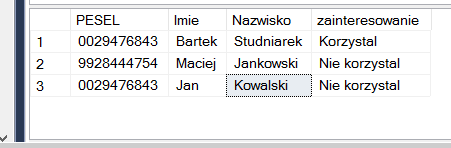
end as 'zainteresowanie'

from Goscie,Uslugi,Faktury

where Faktury.id\_uslugi=Goscie.id\_uslugi and Goscie.PESEL=Faktury.PESEL and Goscie.id\_uslugi=Uslugi.id\_uslugi

GO

select\*from zainteresowanieUsluga



Rys.4.Wynik działania zainsteresowanieUsluga

1. **Omówienie i prezentacja funkcji**

W tym punkcie umieszczamy informacje dotyczące 3 opracowanych funkcji. Funkcjonalność funkcji musi być utrzymana na poziomie uniwersyteckim. Dokumentację dotyczącą funkcji tworzymy zgodnie z poniższym wzorem:

**Nazwa\_funkcji\_1** – jest to funkcja, która …

KOD T-SQL …

**Iloscposilkow**- jest to funkcja która przyjmuje dana datę oraz typ posiłku (śniadanie, obiad, kolacja, lunch pracowniczy) i zwraca ilość wystąpień danego posiłku w ten dzień

drop function if exists dbo.iloscposilkow

go

create function dbo.iloscposilkow(@data\_serwowania date,@typ varchar(30))

returns int

begin

declare @ile int

set @ile=(select count(\*) from Restauracja

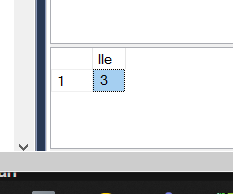
where data\_serwowania=@data\_serwowania and @typ=typ)

return @ile

end

go

select dbo.iloscposilkow('01-02-2021','Śniadanie') as 'Ile'

****

Rys.”5”. Wynik działania funkcji iloscposilkow

**Cena -**  jest to funkcja, która przyjmuje kwote i zlicza ile osob zapłaci więcej niż podana kwota

drop function if exists dbo.cena

go

create function dbo.cena(@kwota int)

returns int

begin

declare @ile int

set @ile=(select count (\*) from Faktury

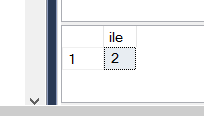
where kwota>@kwota)

return @ile

end

go

select dbo.cena('300') as 'ile'



Rys.”6”. Wynik działania funkcji cena

**doKiedy**- funkcja dostaje nr rezerwacji i zwraca datę w której ta rezerwacja się kończy

drop function if exists dbo.doKiedy

go

create function dbo.doKiedy(@nr\_rezerwacji int)

returns date

begin

declare @data date

set @data=( select rezerwacja\_do from Rezerwacje

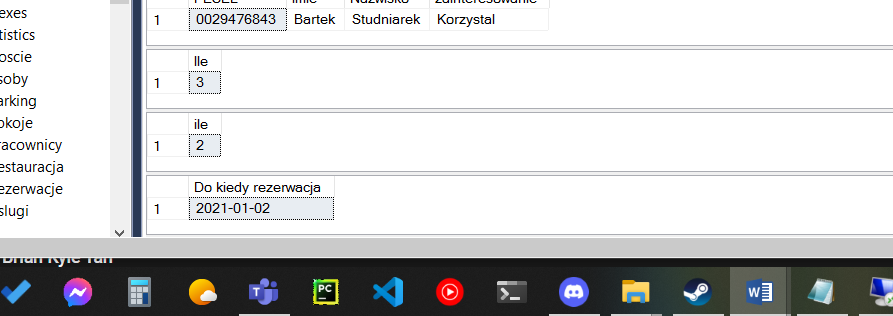
where @nr\_rezerwacji=nr\_rezerwacji)

return @data

end

go

select dbo.doKiedy('889') as 'Do kiedy rezerwacja'

****

Rys.”7”. Wynik działania funkcji doKiedy

**Punktacja (max 10 pkt):**

1. **0,5 pkt**
2. **2,5 pkt**
3. **0,5 pkt**
4. **0,5 pkt**
5. **0,5 pkt**
6. **1 pkt**
7. **1,5 pkt**
8. **3 pkt**

**Termin oddania – 14 blok (przedostatni)**

**Sposób oddania: droga mailowa, format PDF.**