|  |  |  |  |
| --- | --- | --- | --- |
| Informatyka, studia dzienne | | semestr V | |
| **Zaawansowane systemy baz danych** | | **2019/2020** | |
| Prowadzący: dr inż. Krzysztof Myszkorowski | | wtorek, 10:30 | |
| **Bartłomiej Jencz…...** | **216783 IOAD gr 1** | |

Instrukcja 5

1. Napisz procedurę, która:

1. dodaje nowego właściciela,
2. wywoływana jest z 4 parametrami: imię, nazwisko, adres, telefon
3. numer właściciela będzie generowany automatycznie (należy sprawdzić numery w tabeli, wybrać brakujący i dopisać przedrostek)
4. Napisz wywołanie procedury

CREATE PROCEDURE NowyWlasciciel

(

@imie VARCHAR(20),

@nazwisko VARCHAR(20),

@adres VARCHAR(50),

@telefon VARCHAR(20)

)

AS

BEGIN

DECLARE @numer INT

SET @numer = 01

WHILE(CONCAT('CO', CONVERT(VARCHAR(2), @numer)) IN (SELECT wlascicielnr FROM wlasciciele))

BEGIN

set @numer = @numer +1;

END

INSERT INTO wlasciciele VALUES (CONCAT('CO', CONVERT(VARCHAR(2), @numer)), @imie, @nazwisko, @adres, @telefon)

END

GO

NowyWlasciciel 'Jan', 'Kowalski', '95-050 Konstantynow, Lodzka 6', '0-99-999 9999'

2. Napisz funkcję, która dla każdego biura poda jego przychody z wynajmu.

CREATE FUNCTION PrzychodyWynajem()

RETURNS @UDZIAL TABLE (biuroNr VARCHAR(4), udzial FLOAT)

AS

BEGIN

INSERT INTO @udzial

SELECT biuroNr, SUM(czynsz) AS udzial FROM nieruchomosci GROUP BY biuroNr

RETURN

END

GO

3. Napisz wyzwalacz, który podczas wstawiania nowego rekordu do tabeli wynajęcia, w przypadku, gdy podany czynsz jest wyższy od maksymalnego podanego przez klienta:

1. wprowadzi maksymalny czynsz dla tego klienta
2. wypisze stosowny komunikat

CREATE TRIGGER CzynszOverflow ON wynajecia

FOR INSERT

AS

BEGIN

DECLARE @max\_czynsz SMALLINT

SET @max\_czynsz = (SELECT k.max\_czynsz FROM klienci AS k, inserted AS i WHERE k.klientnr = i.klientnr)

IF ((SELECT czynsz FROM inserted) > @max\_czynsz)

BEGIN

PRINT 'Poprawiam czynsz'

UPDATE wynajecia SET czynsz = @max\_czynsz WHERE umowanr = (SELECT umowanr FROM inserted)

END

END

GO

4. Napisz wyzwalacz, który przy wprowadzeniu nowego klienta, dodaje dla niego rejestrację z wybranym numerem pracownika.

CREATE TRIGGER RejestracjaNowegoKlienta ON klienci

FOR INSERT

AS

BEGIN

DECLARE @numer VARCHAR(4)

SET @numer = (SELECT TOP 1 personelnr FROM personel ORDER BY NEWID())

INSERT INTO rejestracje

SELECT klientnr, biuronr, @numer, GETDATE() FROM inserted, personel WHERE personelNr = @numer

END

GO

5. Napisz funkcję obliczającą prowizję dla każdego pracownika przyjmując następujące

założenia:

1. prowizja liczna jest w zakresie podanym przez użytkownika (data\_od data\_do)
2. za wizytę pracownik otrzymuje 2% pensji, za wynajęcie 10%

CREATE FUNCTION Prowizja (@data\_od DATETIME, @data\_do DATETIME)

RETURNS @Tabela TABLE(pracownik VARCHAR(4), prowizja FLOAT)

AS

BEGIN

INSERT INTO @Tabela

SELECT personel.personelnr, COUNT(umowanr) \* 0.10 \* pensja + COUNT(data\_wizyty) \* 0.02 \* pensja

FROM personel, wynajecia, nieruchomosci, wizyty

WHERE nieruchomosci.personelNr = personel.personelNr AND

wizyty.nieruchomoscnr = nieruchomosci.nieruchomoscnr AND

data\_wizyty <= @data\_do AND data\_wizyty >= @data\_od AND

wynajecia.nieruchomoscNr = nieruchomosci.nieruchomoscnr AND

wynajecia.od\_kiedy <= @data\_do AND wynajecia.od\_kiedy >= @data\_od

GROUP BY personel.personelNr, personel.pensja

RETURN

END

GO

6. Napisz procedurę, która wypisuje wszystkie niezapłacone nieruchomości w postaci: Brak wpłaty od .... (nazwisko najemcy) za nieruchomość nr .... za okres .... miesięcy. Uporządkuj je od najstarszej niezapłaconej.

CREATE PROCEDURE NiezaplaconeRachunki

AS

BEGIN

SELECT 'Brak wplaty od ' + kli.nazwisko + ' za nieruchomosc nr ' + wynaj.nieruchomoscNr + ' za okres ' + CONVERT(VARCHAR(2), DATEDIFF(MONTH, wynaj.od\_kiedy, wynaj.do\_kiedy) + 1) + ' mies.'

FROM klienci AS kli, (SELECT \* FROM wynajecia where wynajecia.zaplacona = 0) AS wynaj

WHERE wynaj.klientnr = kli.klientnr

ORDER BY wynaj.od\_kiedy ASC

END

GO

EXEC NiezaplaconeRachunki