Dominik Krystkowiak

BD2-Projekt Sprawozdanie, Część 3

Temat: System rezerwacji – Fitness Klub

Część 3: Obsługa pojedynczej tabeli w bazie danych z wykorzystaniem procedur składowanych, funkcji użytkownika, kontrola integralności bazy danych z wykorzystaniem trigerów. Prezentacja danych zdefiniowanych za pomocą widoków.

1. Operacje wykonywane w kontekście bazy danych:

```
Base = declarative_base()
root=Tk()
try:
    engine = create_engine("mssql+pyodbc://dominik:dominik@DOMINIK-LAPTOP\SQLEXPRESS/System_rezerwacji?driver=SQL+Server+Native+Client+11.0")
    metadata = MetaData(bind=engine)
except:
    tkinter.messagebox.showinfo("Blad", "Nie udalo sie polaczyc")
class Klienci(Base):
    try:
        __table__ = Table('Klienci', metadata, autoload=True)
    except:
        tkinter.messagebox.showinfo("Blad", "Nie udalo sie pobrac tabeli")
```

2. Zawartość tabeli odświeżana jest poleceniem:

```
    def readfromdatabase(self):
        session = create_session(bind=engine)
        testlist = session.query(Klienci).all();
        session.close()
        return testlist
```

3. Operacje na tabeli:

3.1. Dodawanie obiektu:

```
cursor = engine.raw_connection().cursor()
cursor.execute("dodaj_klienta ?, ?, ?, ?, ?", [self.fnameEntry.get(), self.lnameEntry.get(), self.emailEntry.get(),
                                               self.datestartEntry.get(), self.dateendEntry.get()])
cursor.commit()
self.root.destroy()
Odswiez()
```

Podany kod wywołuje następującą procedurę

```
CREATE PROCEDURE dodaj_klienta
@Imie varchar(25),
@Nazwisko varchar(25),
@Adres_email varchar(255),
@Karnet_data_rozpoczecia date,
@Karnet_data_zakonczenia date
AS
BEGIN
if LEN(@Imie) < 2
BEGIN
raiserror('Imie musi mieć co najmniej 2 znaki', 16, 1)
END
if LEN(@Nazwisko) < 2
BEGIN
raiserror('Nazwisko musi mieć co najmniej 2 znaki', 16, 1)
END

INSERT INTO Klienci Values (@Imie, @Nazwisko, @Adres_email, @Karnet_data_rozpoczecia,
@Karnet_data_zakonczenia)
return 0
END
```

3.2. Modyfikowanie obiektu:

```
cursor = engine.raw_connection().cursor()
cursor.execute("modyfikuj_klienta ?, ?, ?, ?, ?, ?",
            [self.index_Entry.get() ,self.fnameEntry.get(), self.lnameEntry.get(), self.emailEntry.get(),
                self.datestartEntry.get(), self.dateendEntry.get()])
cursor.commit()
self.root.destroy()
Odswiez()
```

Podany kod wywołuje następującą procedurę

```
CREATE PROCEDURE modyfikuj_klienta
@ID_Klienta int,
@Imie varchar(25),
@Nazwisko varchar(25),
@Adres_email varchar(255),
@Karnet_data_rozpoczecia date,
@Karnet_data_zakonczenia date
AS
BEGIN
if LEN(@Imie) < 2
BEGIN
raiserror('Imie musi mieć co najmniej 2 znaki', 16, 1)
END
if LEN(@Nazwisko) < 2
BEGIN
raiserror('Nazwisko musi mieć co najmniej 2 znaki', 16, 1)
END

UPDATE Klienci SET Imie=@Imie, Nazwisko=@Nazwisko, Adres_email=@Adres_email,
                    Karnet_data_rozpoczecia=@Karnet_data_rozpoczecia,
                    Karnet_data_zakonczenia=@Karnet_data_zakonczenia
                    WHERE ID_Klienta=@ID_Klienta
IF @@ROWCOUNT=0
BEGIN
raiserror('Aktualizacja danych nieudana',16,1)
END
return 0
END
```

3.3. Usunięcie obiektu:

```
cursor = engine.raw_connection().cursor()
cursor.execute("usun_klienta ?", [self.index_Entry.get()])
cursor.commit()
self.root.destroy()
Odswiez()
```

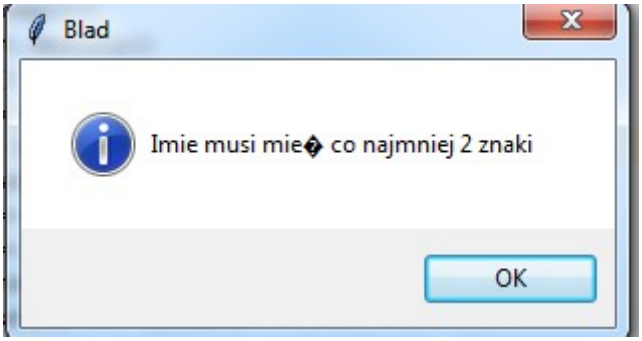
Podany kod wywołuje następującą procedurę

```
CREATE PROCEDURE usun_klienta
@ID_Klienta int
AS
BEGIN
DELETE FROM Klienci WHERE ID_Klienta=@ID_Klienta
IF @@ROWCOUNT=0
BEGIN
raiserror('Usuniecie danych nie powiodlo sie',16,1)
END
return 0
END
```

4. Obsługa wyjątków:
W programie przechwytywane są zarówno wyjątki z procedur i triggerów oraz warunków spójności. Łatwiej jest obsłużyć wyjątki generowane z procedur i triggerów.

```
def click_procedure(self):
    try:
        cursor = engine.raw_connection().cursor()
        cursor.execute("modyfikuj_klienta ?, ?, ?, ?, ?, ?",
                    [self.index_Entry.get() ,self.fnameEntry.get(), self.lnameEntry.get(), self.emailEntry.get(),
                       self.datestartEntry.get(), self.dateendEntry.get()])
        cursor.commit()
        self.root.destroy()
        Odswiez()
    except pyodbc.Error as ex:
        sqlstate = ex.args[0]
        if sqlstate == '42000':
            mes = ex.args[1].split(']')
            mes = mes[4].split('(')
            tkinter.messagebox.showinfo("Blad", mes[0])
        elif sqlstate == '23000':
            tkinter.messagebox.showinfo("Blad", "Naruszono warunki spojnosci. \nAdres email musi byc w formie aaa@aaa.aaa, "
                                        "data w formacie YYYY.MM.DD, data zakonczenia pozniejsza od daty rozpoczecia")
```

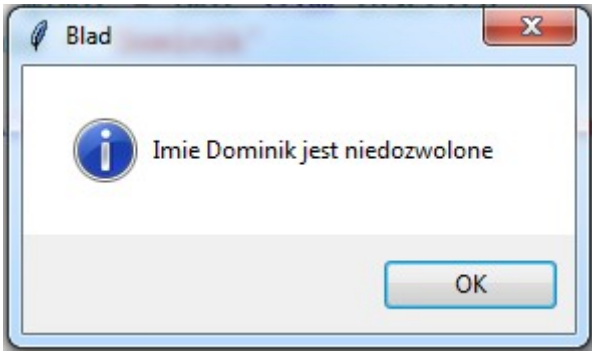
Sposób wyświetlania komunikatów



5. Przykłady triggerów

```
CREATE TRIGGER trig_im ON Klienci
AFTER INSERT, UPDATE
AS
DECLARE @Imie varchar(25)
SELECT @Imie = Imie from inserted
IF @Imie = 'Dominik'
BEGIN
    RAISERROR('Imie %s jest niedozwolone',16,10,@Imie)
        ROLLBACK
END
```

Obsługa:



Wysyłanie e-mail po dodaniu nowego klienta za pomocą triggera

```
CREATE TRIGGER wyslijEmail
ON Klienci
AFTER INSERT
AS
declare @klient varchar(50)
set @klient=(select Imie+', '+Nazwisko from inserted)
declare @tresc varchar(200)
set @tresc='Mamy nowego klienta ' + @klient;
EXEC msdb.dbo.sp_send_dbmail
@profile_name = 'Profil',
@recipients = 'dominik.krystkowiak@student.put.poznan.pl',
@body = @tresc,
@subject = @klient;
GO
```

Adam, Malysz 13 grudnia, 2017 9:39

Od: Gmail Public

Do: dominik krystkowiak

Mamy nowego klienta Adam, Malysz

6. Przykłady widoku

```
create view Lista_Imion as
select Imie,count(*) as Ilosc
from Klienci
group by Imie
```

7. Podać przykłady zmiany logiki biznesowej poprzez zmianę procedur, trigerów, ... a nie kodu aplikacji.

   - Ograniczenia związane z przyjmowanymi danymi realizowane są przez bazę danych za pomocą warunków spójności lub triggerów.

   - Sprawdzane parametry przez procedurę pomagają programiście, ponieważ skupia się on wyłącznie na obsłudze wyjątków zwracanych przez bazę danych.

   - Za pomocą triggerów w łatwy sposób można skonfigurować system powiadomień emailowych informujący o np. nowych klientach czy kończącym się okresie użytkowania.