Maciej Domagalski

Bazy Danych 2 – projekt

Dokumentacja

Temat 1. Obsługa danych przestrzennych dwuwymiarowych.

I. Opis problemu.

Celem projektu było stworzenie UDT reprezentujących punkt oraz obszar oraz metod do nich: obliczanie odległości między punktami, pola obszaru i sprawdzenia, czy punkt należy do obszaru. Przygotowane typy należało porównać z systemowymi.

II. Instrukcja pierwszego uruchomienia.

A) Konfiguracja bazy danych.

Pierwszym krokiem jest stworzenie bazy danych (*database.sql*). Jeśli nie zostało to wcześniej zrobione, należy umożliwić używanie CLR w bazie (*configure.sql*)

B) Budowanie projektu.

Po otwarciu projektu należy sprawdzić, czy jest podłączony do odpowiedniej bazy danych. Następnie należy go zbudować i zdeployować. Teraz baza danych jest gotowa do przyjmowania zapytań z SQL.

III. Opis typów danych.

Projekt składa się z dwóch głównych części - typu *Point* reprezentujący punkt na płaszczyźnie oraz typu *Quadrilateral,* reprezentującego czworokątny obszar oparty na czterech wierzchołkach.

III.I. Typ *Point*

Złożony z dwóch zmiennych typu *double: m_x* i *m_y*. Przechowują odpowiednie współrzędne punktu. Typ zawiera metody:

a. *Point*(double *x*, double *y*): konstruktor tworzący punkt o danych współrzędnych. Wykorzystywany w innych metodach.
b. *Parse*(SqlString *s*): funkcja wywoływana przy tworzeniu punktu z poziomu SQL. Wyciąga z ciągu znaków w formie *x y* poszczególne współrzędne i tworzy z nich punkt.
c. *ToString*(): funkcja wyświetlająca punkt w postaci (*x y*).
d. *get*X(), *getY*(): funkcje zwracające pojedynczą współrzędną punktu.
e. *distance*(*point P*): funkcja licząca odległość pomiędzy dwoma punktami.

III.II. Typ *Quadrilateral*

Złożony z czterech zmiennych typu *Point: A, B, C* i *D.* Przechowują wierzchołki czworokąta. Typ zawiera metody:

a. *Parse*(SqlString *s*): funkcja wywoływana przy tworzeniu obszaru z poziomu SQL. Wyciąga z ciągu znaków w formie *xA yA*|*xB yB*|*xC yC*|*xD yD* poszczególne współrzędne wierzchołków i tworzy z nich punkty. Z tych punktów tworzony jest czworokąt.
b. *getA*(), *getB*(), *getC*(), *getD*(): funkcje drukujące poszczególne punkty w postaci (*x y*).
c. *ToString*(): funkcja wyświetlająca czworokąt w postaci *A*: (*xA yA*), *B*: (*xB yB*), *C*: (*xC yC*), *D*: (*xD yD*).
d. *Area*(): funkcja licząca pole obszaru.
e. *IsInside*(Point *P*): funkcja sprawdzająca, czy punkt *P* znajduje się w obszarze.
f. *GetAngle*(double *Ax*, double *Ay*, double *Bx*, double *By*, double *Cx*, double *Cy*): funkcja pomocnicza w *IsInside*. Służy do liczenia kąta między prostymi wyznaczonymi przez trzy punkty.
g. *DotProduct*, *CrossProductLength*: funkcje pomocnicze w *GetAngle*, służące do liczenia iloczynu skalarnego oraz długości wektora z iloczynu wektorowego.

IV. Opis funkcjonalności.

Przedstawione metody są gotowe do wykorzystania w kodzie SQL. Przykładowe zastosowania przedstawiłem w plikach *point.sql* i *quadrilateral.sql*.

IV.I. Typ *Point*

Aby stworzyć punkt, najpierw należy zadeklarować typ obiektu (*DECLARE @punkt [dbo].[Point]*), a następnie stworzyć punkt, używając instrukcji *SET @punkt = 'x y'*. Z powodu ograniczeń SQL jako separatora dziesiętnego należy używać przecinka, a nie kropki. Na punkcie można wywołać metody:

a. **Wyświetlanie punktu:** *SELECT @punkt.ToString() AS nazwaKolumny*
b. **Wyświetlanie jednej współrzędnej** (tutaj x, współrzędna y analogicznie): *SELECT @punkt.getX() AS nazwaKolumny*
c. **Obliczanie odległości** (zakładając, że stworzony został punkt *@inny*): *SELECT @punkt.distance(@inny) AS 'nazwaKolumny'*

IV.II. Typ *Quadrilateral*

Aby stworzyć obszar czworokątny, najpierw należy zadeklarować typ obiektu (*DECLARE @obszar [dbo].[Quadrilateral]*), a następnie stworzyć obszar, używając instrukcji *SET @obszar = 'xA yA|xB yB|xC yC|xD yD'*. Z powodu ograniczeń SQL jako separatora dziesiętnego należy używać przecinka, a nie kropki. Na czworokącie można wywołać metody:

a. **Wyświetlanie obszaru:** *SELECT @obszar.ToString() AS nazwaKolumny*
b. **Wyświetlanie jednego punktu** (tutaj A, pozostałe analogicznie): *SELECT @obszar.getA() AS nazwaKolumny*
c. **Obliczanie pola czworokąta:** *SELECT @obszar.Area() AS nazwaKolumny*
d. **Sprawdzanie, czy punkt należy do obszaru** (zakładając, że stworzony został punkt *@punkt*): *SELECT @obszar.IsInside(@punkt) AS 'nazwaKolumny'*

VII. Wykorzystane linki.

1. Główne źródło wiedzy dotyczące UDT
2. Algorytm do obliczania pola wielokąta
3. Algorytm do sprawdzania, czy punkt należy do obszaru