Dokumentacja Techniczna Do Programu Importującego Dane z Pliku Do Bazy Ms Sql

Program importujący dane z pliku zbkrod.txt do bazy danych ms sql

Sporządził Karol Brzozowski

1. Opis programu

Celem programu, jest wczytanie danych z pliku zbkrod.txt do bazy danych mssql, założenie odpowiedniej tabeli w bazie danych w oparciu o strukturę pliku, oraz sprawdzenie poprawności danych, program działa w kilku etapach. Program korzysta z dwóch plików wejściowych, pliku z danymi „zbkrod.txt" oraz pliku z strukturą tabeli „dozbk.txt".

Program ma za zadanie przyspieszyć i usprawnić konwencjonalną ocenę wartości hodowlanej bydła mlecznego, przez wyeliminowanie operacji na plikach txt na rzecz operacji na bazie danych.

Pliki wejściowe :

zbkrod.txt – plik z danymi
dozbk.txt – plik z strukturą danych
Przykład pliku dozbk.txt

v	*14*	*1*	*14*	*nr_krowy*
v	*1*	*15*	*15*	*import*
v	*2*	*16*	*17*	*ET*
v	*2*	*18*	*19*	*kod_rasy_gł*
v	*87*	*20*	*106*	*rasa*
i	*8*	*107*	*114*	*data_urodz*
v	*2*	*115*	*116*	*księga*
v	*14*	*117*	*130*	*nr_ojca_krowy*
v	*14*	*131*	*144*	*nr_matki_krowy*
v	*2*	*145*	*146*	*kod_rasy_gł_matki_krowy*
v	*87*	*147*	*233*	*rasa_matki_krowy*
v	*14*	*234*	*247*	*nr_ojca_ojca*
v	*14*	*248*	*261*	*nr_matki_ojca*
v	*14*	*262*	*275*	*nr_ojca_matki*
v	*14*	*276*	*289*	*nr_matki_matki*
i	*2*	*290*	*291*	*liczba_wybr_lakt*
i	*4*	*292*	*295*	*liczba_wybr_prob_dla_krowy*
v	*1*	*296*	*296*	*szybkosc_doju*
v	*1*	*297*	*297*	*temperament*
i	*8*	*298*	*305*	*data_ubycia_spod_oceny*
v	*2*	*306*	*307*	*powod_ubycia*
i	*1*	*0*	*0*	*flaga_poprawnosci_nr*
v	*14*	*0*	*0*	*nr_poprawny*
i	*1*	*0*	*0*	*data_dodania*

Kolumna nr. 1 typ danych v - varchar, i – integer,
Kolumna nr. 2 ilość znaków do pobrania.
Kolumna nr. 3 informuje od której kolumny pobiera dane.
Kolumna nr. 4 informuje do której kolumny pobiera dane.
Kolumna nr. 5 nazwa używana w tabeli w bazie ms sql.

2. Mechanizm działania programu

-Pierwszym etapem programu zaraz po uruchomieniu jest zapytanie użytkownika o datę dodania danych, IP serwera bazodanowego, nazwę bazy, login do bazy, hasła, nazwę tabeli, po każdym zapytaniu program zapyta użytkownika o to czy dokonać zmiany t/n, jeśli damy t(tak) musimy wpisać odpowiednią wartość, jeśli n(nie) program przejdzie do nst. Pytania lub jeśli jest to ostanie pytanie to do nst. Etapu. Za prawidłowość tego etapu odpowiada funkcja AddInfo(server,datavase,udi,pwd,name,data);

-Jeśli plik istnieje zostanie wczytany do tablicy strukturalnej typu krowy.txt której elementami typu integer są liczba_znakow, od_znaku, do_znaku. Elementami typu string nazwa, rodzaj, dane. Zgodnie z strukturą pliku dozbk. tablica przechowuję strukturę wiersza pliku zbkrod.txt

- Zawartość tablicy strukturalnych typu krowy tab1 (pozycja nazwa i rodzaj) zostanie wczytana do zmiennej typu string o nazwie tabela, która na samym początku przechowuję polecenie w języku sql `string tabela = "CREATE TABLE "+name+"(nr_krowy VARCHAR(14) PRIMARY KEY,"`

```
tabela += tab1[k].nazwa + " " + tab1[k].rodzaj + ",";
```

- Następnie program tworzy obiekt typu DataTable, jest to tablica której struktura jest taka sama jak tablicy strukturalnej tab1 typu krowy.

- Następnie pojawi się na ekranie pytanie, „`"Jesli chcesz utworzc nowa tabela w bazie danych nacisnij t, jesli nie n",` jeśli damy „t” zostanie utworzona nowa tabela w bazie danych, jeśli wpiszemy „n” dane zostaną wczytane do istniejącej tabeli.

- Na ekranie pojawi się czas rozpoczęcia działania programu.

- Program wczytuję wiersze linia po linii, i zapisuje do tab1;

- Następnie funkcja SprNr() sprawdza poprawność numeru krowy i za pomocą funkcji AddRowToDataTable() zapisuję dane do zmiennej typu DataTable myTable;

-Po przekroczeniu 200 tyś rekordów w zmiennej myTable, dale za pomocą funkcji BulkCopyy zapisywane są w bazie danych. Ze względu na oszczędność pamięć na komputerze stacjonarnym dane przekazywane są partiami(co 200tys rekordów).

3. Szczegułowe omówienie konstrukcji programu

Wszystkie wytworzone i użyte funkcję i klasy w programie znajdują się w pliku z funkcją główną.

- Funkcja AddInfo() Funkcja pobiera dane z klawiatury, po uruchomieniu funkcji program poprosi użytkownika o dane typu Ip serwera, nazwy bazy…

- Funkcja BulkCopyToData() Funkcja odpowiada za przeniesienie tablicy typu DataTable do bazy danych Mssql, funkcja przy przesyłaniu danych stosuję mechanizm transakcji. Jako parametr funkcji stosuję się przekazaną przez referencje zmienną typu DataTable, zmienną typu SqlConnection przechowującą informacje o połączeniu typu, login, hasło… Zmienną typu string przechowująca nazwę tabeli w bazie mssql.

```
static void BulkCopyToData(ref DataTable  myTable, SqlConnection conn, string name)
 {
   using (SqlTransaction myTran = conn.BeginTransaction())
   {


     using (SqlBulkCopy Bulki = new SqlBulkCopy(conn,SqlBulkCopyOptions.KeepIdentity,myTran))
     {
                Bulki.BatchSize = 10000;
                Bulki.BulkCopyTimeout = 10000;
                Bulki.DestinationTableName = name;


                try
                {
                    Bulki.WriteToServer(myTable);
                    myTran.Commit();
                }
                catch(Exception ex)
                {
                    Console.WriteLine(ex.Message);
                    myTran.Rollback();
                }

            }
        }
    }
```

- Funkcja `AddRowToDataTable()` Funkcja dodaje kolejny wiersz do tablicy typu DataTable na podstawie przekopiowania danych z tablic strukturalnych typu krowy tab1. Funkcja tworzy obiekt typu DataRow = myTable.newRow(), następnie za pomocą pętli for Przekopiowuję dane z pierwszej tablicy tab1. Funkcja dodatkowo zmienia zapis daty w przypadku kiedy w polu data jest 4 elementy.

```
static void AddRowToDataTable(ref DataTable myTable, krowy[] tab1)
{
            Int32 dl = 0;
            DataRow myRow = myTable.NewRow();
            for (int i = 0; i < 24; i++)
            {
                dl = (tab1[i].dane).Length;
                if (dl > 0)
                {
                    if (tab1[i].nazwa[0] == 'd' && tab1[i].nazwa[1] == 'a' && dl == 4)
                        AddZeroToDat(ref tab1[i].dane);

                    myRow[tab1[i].nazwa] = (tab1[i].dane).ToString();

                }
            }

            myTable.Rows.Add(myRow);
}
```

-Funkcja AddZeroToDat()
Funkcja dopisuję cztery zera do daty w przypadku kiedy data składa się tylko z roku itp.

```
 static void AddZeroToDat(ref string tab)
 {
     tab += "0000";
 }
```

- Funkcja GetNextWord() Funkcja wybiera pojedyncze słowa z ciągu słów, jeśli napotka na biały znak, zwraca słowo. Funkcja działa w pętli typu while i przeszukuję dany plik i wybiera z niego pojedyncze słowa do funkcji jako parametr jest przekazana zmienna typu StreamRider().

```
static String GetNextWord(StreamReader sr)
{
       String word = "";
       char c;
       while (sr.Peek() >= 0)
       {
            c = (char)sr.Read();
            if (c.Equals(' ') || c.Equals('\t') || c.Equals('\n') || c.Equals('\r'))
            {
                 break;
            }
            else
              word += c;
     }
     return word;
}
```

- Funkcja CopPiece() Funkcja kopiuje wybrany fragment z tekstu i zwraca go jako parametr, parametrami funkcji są zmienna typu string lan która przechowuję pobrany z pliku wiersz, zmienna typu string zm1 która jest zwracanym elementem czyli wybranym słowem z ciągu, zmienna typu Int32 od która mówi od którego znaku ma zostać pobierany fragment, zmienna typu int32 doo która mówi do którego znaku ma zostać pobrany fragment, i zmienna typu int siz która informuje o długości ciągu znaków(wiersza z pliku), Funkcja po za samym kopiowanie fragmentu sprawdza poprawność danych i poprawia je.

```
static void CopPiece(ref string zm1, ref string lan, Int32 od, Int32 doo, int siz)
{
        zm1 = "";
        int i, z = 1;
        for (i = od; i <= doo; i++)
        {
             if (doo - od == 0 && lan[od] == ' ') break;
             if (i == siz) break;
             if (i == doo) z = 0;
             if (lan[i] == ' ' && lan[i + z] == ' ')
             {

             }
             else
             {
                if (lan[i] == 39)
                   zm1 += '`';
                else
                   zm1 += lan[i];
             }
          }
             zm1 = zm1.Trim();
 }
```

- **Funkcja SprInt()**Funkcja sprawdza poprawność danych typu integer, usuwa zbędne spacje na początku i na końcu słowa, lub w przypadku wystąpienia niewłaściwego znaku w ciągu zwraca puste słowo(NULL).

```
static void SprInt(ref string t)
{
        string zm ="";
        char znak;
        t = t.Trim();
        int siz = t.Length;
        if (siz > 0)
        {
            for (int i = 0; i < siz; i++)
             {
                 znak = t[i];
                 if (znak < 48 || znak > 57)
                 {
                     Console.WriteLine("niewlasciwy int " + t + "    " + znak);
                     }
                     else
                       zm += znak;
                 }
                 t = zm;
             }
             else
              t = "";
}
```

- **Funkcja AddFileToMatrix()**Funkcja dodaje dane z pliku dozbk.txt w którym jest informacja o strukturze pliku(zbkrod.txt), i odczytuję słowo po słowie, plik zawiera 5 kolumn pierwsza kolumna informuje o rodzaju danych (integer, varchar), druga kolumna informuje o ilości znaków w danym słowie, trzecia kolumna informuje od którego znaku rozpoczyna się dane słowo, czwarta kolumna informuje do którego znaku, piąta kolumna zawiera nazwę danego słowa, i spełnia rolę nazwy kolumny w bazie danych, plik dozbkrod.txt zawiera 24 wierszy, co odpowiada 24 kolumnom w bazie Mssql, funkcja wykorzystuję funkcję

```
static void AddFileToMatrix(ref krowy[] tab, int i, string slowo, StreamReader srr)
        {
            if (slowo == "i") slowo = "INT";
            else slowo = "VARCHAR";
            tab[i].rodzaj = slowo;

            slowo = GetNextWord(srr);
            tab[i].liczba_znakow = Convert.ToInt32(slowo);

            slowo = GetNextWord(srr);
            tab[i].od_znaku = Convert.ToInt32(slowo) - 1;

            slowo = GetNextWord(srr);
            tab[i].do_znaku = Convert.ToInt32(slowo) - 1;

            slowo = GetNextWord(srr);
            tab[i].nazwa = slowo;
            tab[i].dane = "";
        }
```

- Funkcja `CreateDataTable()` Funkcja tworzy tabelę typu DataTable i zwraca ją, funkcja tworzy strukturę kolumn tabeli na podstawie tablicy przechowującej dane odnośnie wiersza z pliku zbkrod.txt. Parametrami funkcji są, tablice strukturalne typu krowy które przechowuje listaKolumn1 informacje odnośnie pierwszego wiersza w pliku zbkrod.txt.

```
static DataTable CreateDataTable(krowy[] listaKolumn1, string tablename)
{
       string zm = "";
       string query = "select top 1 *from " + tablename;
       DataTable tab = new DataTable();

       for (int i = 0; i < 24; i++)
       {
              if (listaKolumn1[i].rodzaj[0] == 'V')
              tab.Columns.Add(new DataColumn(listaKolumn1[i].nazwa, typeof(string)));
              else
              tab.Columns.Add(new DataColumn(listaKolumn1[i].nazwa, typeof(int)));

       }


       return tab;
}
```

- Funkcja `SprNr ()` Funkcja sprawdza numer osobnika, pod względem ilości znaków, i poprawności znaków, jeśli nr jest prawidłowy to ustawia w kolumnie flaga w bazie mssql na 1 jeśli nie 0. Dodatkowo funkcja dodaje zera po symbolu kraju jeśli nr nie ma 14 znaków np. PL1234567 zmieni go na PL000001234567. Funkcja jako parametry przyjmuję dwie tablicę typu strukturalnego krowy które reprezentują dwuwierszową strukturę pliku zbkrod.txt, jak i zmienną typu string przechowującą datę oceny.

```
static void SprNr(ref krowy[] tab1, string data)
{

     int dl = (tab1[1].dane).Length;
     tab1[21].dane = "";
     tab1[21].dane = "";
     tab1[23].dane = data;
```

```
if (dl == 14)
   {
      tab1[21].dane = "1";

   }
   else
   {
      if ((tab1[1].dane[0] > 64 && tab1[1].dane[0] < 91) && (tab1[1].dane[1] > 64 && tab1[1].dane[1]
      < 91) && (dl > 2))
      {

                    string nap = "";
                    nap += tab1[1].dane[0];
                    nap += tab1[1].dane[1];

                    for (int i = 0; i < 14 - dl; i++)
                    {
                        nap += '0';
                    }
                    for (int i = 2; i < dl; i++)
                    {
                        if (tab1[1].dane[i] > 47 && tab1[1].dane[i] < 58)
                            nap += tab1[1].dane[i];
                        else
                        {
                            nap = "";
                            break;
                        }
                    }
                    tab1[22].dane = nap;
                }
                tab1[21].dane = "0";

     }

 }
}
```

3.3. Omówienie funkcji głównej main().

Dokładne omówienie funkcji głównej main() znajduję się w pliku Program.cs