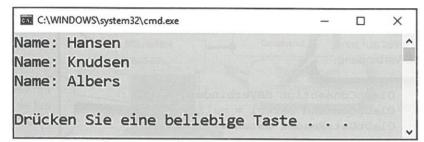
Nach dem Starten wird die Kundentabelle ausgelesen und mithilfe des Datenlesers werden Schritt für Schritt die Werte der 2. Spalte (Index 1) ausgegeben:



## Hinweise:

Der Zugriff auf die Spaltenwerte einer Tabelle mit dem Datenleser erfolgt in Abhängigkeit vom jeweiligen Datentyp. Für jeden Datentyp steht eine geeignete Methode zur Verfügung:

```
GetDateTime(Spaltenindex)GetString(Spaltenindex)GetInt32(Spaltenindex)... weitere Typen
```

Beispielsweise kann die erste Spalte der Kunden-Tabelle mit der Methode GetInt32() ausgelesen werden, da es sich um einen ganzzahligen numerischen Typen (ACCESS-Typ Zahl) handelt:

## 10.1.4 Nicht-Select-Befehle absetzen

Das Auslesen einer beliebigen Tabelle kann mithilfe der oben beschriebenen Anweisungen erfolgen. Möchte man hingegen nicht selektieren, sondern einfügen, ändern oder löschen, so kann ein **ExecuteNonQuery**-Befehl abgesetzt werden. Vorher wird der gewünschte SQL-Befehl in einer Zeichenkette erstellt. In dem folgenden Beispiel wird eine neue Zeile in die Kundentabelle eingefügt, eine bestehende Zeile geändert und eine Zeile gelöscht:

```
using System;
using System. Data;
using System.Data.OleDb;
namespace DB Zugriff CSharp
    class CDBZugriff
                                             Den Verbindungsstring mit
                                            der Provider-Angabe und der
    static void Main(string[] args)
                                              Datenquelle festlegen.
           string verbindungsstring =
                        "Provider=Microsoft.ACE.OLEDB.12.0;
                        Data Source=C:\\Temp\\Kunden.accdb";
           OleDbConnection dBVerbindung = null;
           OleDbCommand befehl = null;
           bool offen = false;
           int anzahl=0;
           try
                dBVerbindung = new
                      OleDbConnection(verbindungsstring);
                dBVerbindung.Open();
                offen = true;
                befehl = dBVerbindung.CreateCommand();
                                   Der SQL-Befehl, um eine
                                     Zeile einzufügen.
               befehl.CommandText = "INSERT INTO Kunden
                                      VALUES (4, 'König', 'Seestr. 5',
                                      'Hamburg', '45621');";
                                  SQL-Befehl absetzen und die
                                  Anzahl der betroffenen Zeilen
                                       zurückerhalten.
                anzahl = befehl.ExecuteNonQuery();
                Console.WriteLine("Anzahl der eingefügten Zeilen: "
                                    + anzahl);
                                            Ein UPDATE-Befehl
                befehl.CommandText = "UPDATE Kunden SET telefon =
                                       '11111' WHERE name = 'Hansen';";
```