```
OleDbConnection dBVerbindung = null;
OleDbCommand befehl = null;
bool offen = false;

try
{
    dBVerbindung =
        new OleDbConnection(verbindungsstring);
        dBVerbindung.Open();
        offen = true;

        befehl = dBVerbindung.CreateCommand();
        befehl.CommandText = "SELECT * FROM Kunden";

        OleDbDataAdapter da = new OleDbDataAdapter(befehl);

        DataSet ds = new DataSet();
        da.Fill(ds);

    for (int i = 0; i < ds.Tables[0].Rows.Count; i++)
    {
        Console.WriteLine(
        ds.Tables[0].Rows[i]["name"].ToString());
    }
```

Ein `OleDBDataAdapter`-Objekt instanzieren und das `OleDbCommand`-Objekt übergeben.

Ein `DataSet`-Objekt anlegen.

Mit der Methode `Fill()` wird das `DataSet`-Objekt mit den Datensätzen aus der Datenbank gefüllt, die der oben angegebenen Abfrage entsprechen.

Das Ergebnis der Abfrage kann über das `Tables`-Array des `DataSet`-Objektes angesprochen werden. Zusätzlich wird die gewünschte Zeile und Spalte angegeben:

```
C:\WINDOWS\system32\cmd.exe
Name: Hansen
Name: Knudsen
Name: Albers

Drücken Sie eine beliebige Taste . . .
```

```
ds.Tables[0].Rows[0]["name"] = "Laufer";
```

Eine Änderung der Daten erfolgt dann über eine einfache Zuweisung!


```
DataRow zeile = ds.Tables[0].NewRow();
zeile["id"] = 10;
zeile["name"] = "Kaiser";
ds.Tables[0].Rows.Add(zeile);

ds.Tables[0].Rows[1].Delete();

OleDbCommandBuilder cmb =
        new OleDbCommandBuilder(da);
da.Update(ds);
    }
    catch(Exception ausnahme)
    {
        Console.WriteLine("Datenbankfehler: "
                + ausnahme.Message);
    }
    finally
    {
        if (offen == true) dBVerbindung.Close();
    }
}
}
```

Das Hinzufügen eines Datensatzes erfolgt mit einem `DataRow`-Objekt und der Methode `Add()`.

Das Löschen erfolgt über die `Delete`-Methode!

Die Synchronisierung mit der Datenbank erfordert ein `CommandBuilder`-Objekt, das die erforderlichen `Update`-Befehle zur Verfügung stellt.

Nach der Synchronisierung sieht die Kunden-Tabelle so aus:

Kunden

id	name	strasse	ort	telefon
1	Laufer	Baumallee 1	Hamburg	123456
3	Albers	Paulistr. 8	Hamburg	111222
10	Kaiser			

Datensatz: 3 von 3 | Kein Filter | Suchen

10.2 Den Datenbankassistenten von Visual C# nutzen

10.2.1 Eine Datenbank einbinden

Die Entwicklungsumgebung Visual C# bietet einen Assistenten an, mit dem Datenbanken automatisiert in ein Projekt eingebunden werden können. Damit verkürzt sich die Entwicklungszeit im Vergleich zu den oben beschriebenen Methoden beträchtlich. Allerdings hat der Entwickler damit auch weniger Freiheiten, da der Assistent viel Quellcode automatisch generiert.

In einem ersten Schritt muss dem Projekt eine Datenquelle hinzugefügt werden. Das geschieht über den Menüpunkt *„Projekt → Neue Datenquelle hinzufügen…“*: