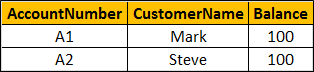
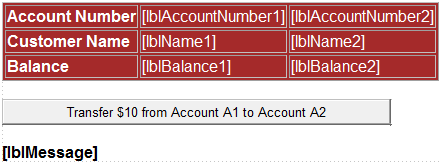
In this video we will discuss **how to implement Transactions in ADO.NET**  
  
**What is a Transaction**  
A Transaction ensures that either all of the database operations succeed or all of them fail. This means the job is never half done, either all of it is done or nothing is done. Let's understand this with an example.   
  
   
  
We will be using the following **Accounts** table in this demo   
   
  
The table has got 2 Accounts (A1 and A2). We want to design a web application to **transfer $10 from Account A1 to Account A2**. The design of the webform should be as shown below.   
   
  
When we click **"Transfer $10 from Account A1 to Account A2"** button, we should subtract 10 from A1 account and add 10 to A2 account. So there will be 2 database UPDATE statements. What do you think will happen if only the first update statement is executed successfully and not the second statement. $10 is deducted from the first account, but not added to the second account. This is definitely not desirable. Either both the statements should succeed or both of them should fail. If one succeeds and other fails we should also rollback the changes made by the first statement to maintain the integrity of the data. This can be achieved using transactions in ado.net.  
  
**Step 1 :** Create the Accounts table using the following SQL script

Create Table Accounts

(

     AccountNumber nvarchar(10) primary key,

     CustomerName nvarchar(50),

     Balance int

)

GO

Insert into Accounts values('A1', 'Mark', 100)

Insert into Accounts values('A2', 'Steve', 100)

GO

**Step 2 :** Create a new empty asp.net web application. Name it Demo.  
  
**Step 3 :** Include connection string in web.config file

<connectionStrings>

  <add name="CS"

           connectionString="server=.;database=Sample;integrated security=SSPI"/>

</connectionStrings>

**Step 4 :** Add a WebForm. Copy and paste the following HTML.

<div style="font-family: Arial">

<table border="1" style="background: brown; color: White">

    <tr>

        <td>

            <b>Account Number </b>

        </td>

        <td>

            <asp:Label ID="lblAccountNumber1" runat="server"></asp:Label>

        </td>

        <td>

            <asp:Label ID="lblAccountNumber2" runat="server"></asp:Label>

        </td>

    </tr>

    <tr>

        <td>

            <b>Customer Name </b>

        </td>

        <td>

            <asp:Label ID="lblName1" runat="server"></asp:Label>

        </td>

        <td>

            <asp:Label ID="lblName2" runat="server"></asp:Label>

        </td>

    </tr>

    <tr>

        <td>

            <b>Balance </b>

        </td>

        <td>

            <asp:Label ID="lblBalance1" runat="server"></asp:Label>

        </td>

        <td>

            <asp:Label ID="lblBalance2" runat="server"></asp:Label>

        </td>

    </tr>

</table>

<br />

<asp:Button ID="btnTransfer" runat="server"

            Text="Transfer $10 from Account A1 to Account A2"

            OnClick="btnTransfer\_Click" />

<br />

<br />

<asp:Label ID="lblMessage" runat="server" Font-Bold="true"></asp:Label>

</div>

**Step 5 :** Copy and paste the following code in the code-behind file.

using System;

using System.Configuration;

using System.Data.SqlClient;

namespace Demo

{

    public partial class WebForm1 : System.Web.UI.Page

    {

        protected void Page\_Load(object sender, EventArgs e)

        {

            if (!IsPostBack)

            {

                GetAccountsData();

            }

        }

        private void GetAccountsData()

        {

            string cs = ConfigurationManager.ConnectionStrings["CS"].ConnectionString;

            using (SqlConnection con = new SqlConnection(cs))

            {

                SqlCommand cmd = new SqlCommand("Select \* from Accounts", con);

                con.Open();

                SqlDataReader rdr = cmd.ExecuteReader();

                while (rdr.Read())

                {

                    if (rdr["AccountNumber"].ToString() == "A1")

                    {

                        lblAccountNumber1.Text = "A1";

                        lblName1.Text = rdr["CustomerName"].ToString();

                        lblBalance1.Text = rdr["Balance"].ToString();

                    }

                    else

                    {

                        lblAccountNumber2.Text = "A2";

                        lblName2.Text = rdr["CustomerName"].ToString();

                        lblBalance2.Text = rdr["Balance"].ToString();

                    }

                }

            }

        }

        protected void btnTransfer\_Click(object sender, EventArgs e)

        {

            string cs = ConfigurationManager.ConnectionStrings["CS"].ConnectionString;

            using (SqlConnection con = new SqlConnection(cs))

            {

                con.Open();

                // Begin a transaction. The connection needs to

                // be open before we begin a transaction

                SqlTransaction transaction = con.BeginTransaction();

                try

                {

                    // Associate the first update command with the transaction

                    SqlCommand cmd = new SqlCommand

                        ("Update Accounts set Balance = Balance - 10 where AccountNumber = 'A1'"

                        , con, transaction);

                    cmd.ExecuteNonQuery();

                    // Associate the second update command with the transaction

                    cmd = new SqlCommand

                        ("Update Accounts set Balance = Balance + 10 where AccountNumber = 'A2'"

                        , con, transaction);

                    cmd.ExecuteNonQuery();

                    // If all goes well commit the transaction

                    transaction.Commit();

                    lblMessage.ForeColor = System.Drawing.Color.Green;

                    lblMessage.Text = "Transaction committed";

                }

                catch

                {

                    // If anything goes wrong, rollback the transaction

                    transaction.Rollback();

                    lblMessage.ForeColor = System.Drawing.Color.Red;

                    lblMessage.Text = "Transaction rolled back";

                }

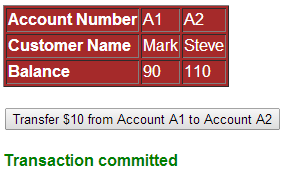
            }

            GetAccountsData();

        }

    }

}

**Testing :** Run the apllication and click the**"Transfer $10 from Account A1 to Account A2"** button. Notice that $10 is deducted from Account A1 and added to Account A2 and the transaction is committed.   
   
  
Let's now deliberately introduce a change that would crash the application at run time after executing the first update statement.   
  
CHANGE THE FOLLOWING LINE  
cmd = new SqlCommand("Update Accounts set Balance = Balance + 10 where AccountNumber = 'A2'", con, transaction);  
  
TO  
cmd = new SqlCommand("Update Accounts1 set Balance = Balance + 10 where AccountNumber = 'A2'", con, transaction);  
  
Run the apllication again and click the **"Transfer $10 from Account A1 to Account A2"** button. Notice that the transaction is rolled back and the data integrity is not lost.   
