**In this video we will learn about**  
**1.** Loading 2 or more tables into a DataSet, and giving these tables a meaningful name.  
**2.** Binding the tables in the DataSet to user interface databound controls like GridView, using the Tables collection property of the DataSet object.  
  
**When the following stored procedure is executed, we get 2 result-sets**  
Create procedure spGetProductAndCategoriesData  
as  
Begin  
 Select ProductId, ProductName, UnitPrice   
 from tblProductInventory  
  
 Select CategoryId, CategoryName  
 from tblProductCategories  
End  
  
**Drag and drop 2 GridView controls onto the webform**and change the ID to **GridViewProducts** and **GridViewCategories**. The HTML in the aspx page should be as shown below.  
<asp:GridView ID="GridViewProducts" runat="server">  
</asp:GridView>  
<br />  
<asp:GridView ID="GridViewCategories" runat="server">  
</asp:GridView>  
  
   
  
**Copy and paste the following code into the code behind page.**  
string ConnectionString = ConfigurationManager.ConnectionStrings["DBConnectionString"].ConnectionString;  
using (SqlConnection connection = new SqlConnection(ConnectionString))  
{  
    SqlDataAdapter dataAdapter = new SqlDataAdapter("spGetProductAndCategoriesData", connection);  
    dataAdapter.SelectCommand.CommandType = CommandType.StoredProcedure;  
    DataSet dataset = new DataSet();  
    dataAdapter.Fill(dataset);  
  
    GridViewProducts.DataSource = dataset;  
    GridViewProducts.DataBind();  
  
    GridViewCategories.DataSource = dataset;  
    GridViewCategories.DataBind();  
}   
  
   
  
**When you run the project now, notice that both the gridview controls show the same data.** This is because, by default, the first table from the dataset is used as the data source for both the gridview controls. We actually want to show products data in one gridview control and categories data in the other. To specify the specific DataTable, that you want to bind to a gridview control, use the Tables collection property of the dataset object, as shown below.  
string ConnectionString = ConfigurationManager.ConnectionStrings["DBConnectionString"].ConnectionString;  
using (SqlConnection connection = new SqlConnection(ConnectionString))  
{  
    SqlDataAdapter dataAdapter = new SqlDataAdapter("spGetProductAndCategoriesData", connection);  
    dataAdapter.SelectCommand.CommandType = CommandType.StoredProcedure;  
    DataSet dataset = new DataSet();  
    dataAdapter.Fill(dataset);  
  
    GridViewProducts.DataSource = dataset.Tables[0];  
    GridViewProducts.DataBind();  
  
    GridViewCategories.DataSource = dataset.Tables[1];  
    GridViewCategories.DataBind();  
}  
  
**By default the tables in the DataSet will have table names as Table, Table1, Table2**etc. So if you want to give the tables in the DataSet a meaningful name, use the TableName property as shown below.  
dataset.Tables[0].TableName = "Products";  
dataset.Tables[1].TableName = "Categories";  
  
**These table names can then be used when binding to a GridView control**, instead of using the integral indexer, which makes your code more readable, and maintainable.  
GridViewProducts.DataSource = dataset.Tables["Products"];  
GridViewProducts.DataBind();  
  
GridViewCategories.DataSource = dataset.Tables["Categories"];  
GridViewCategories.DataBind();