

## Practical No 9

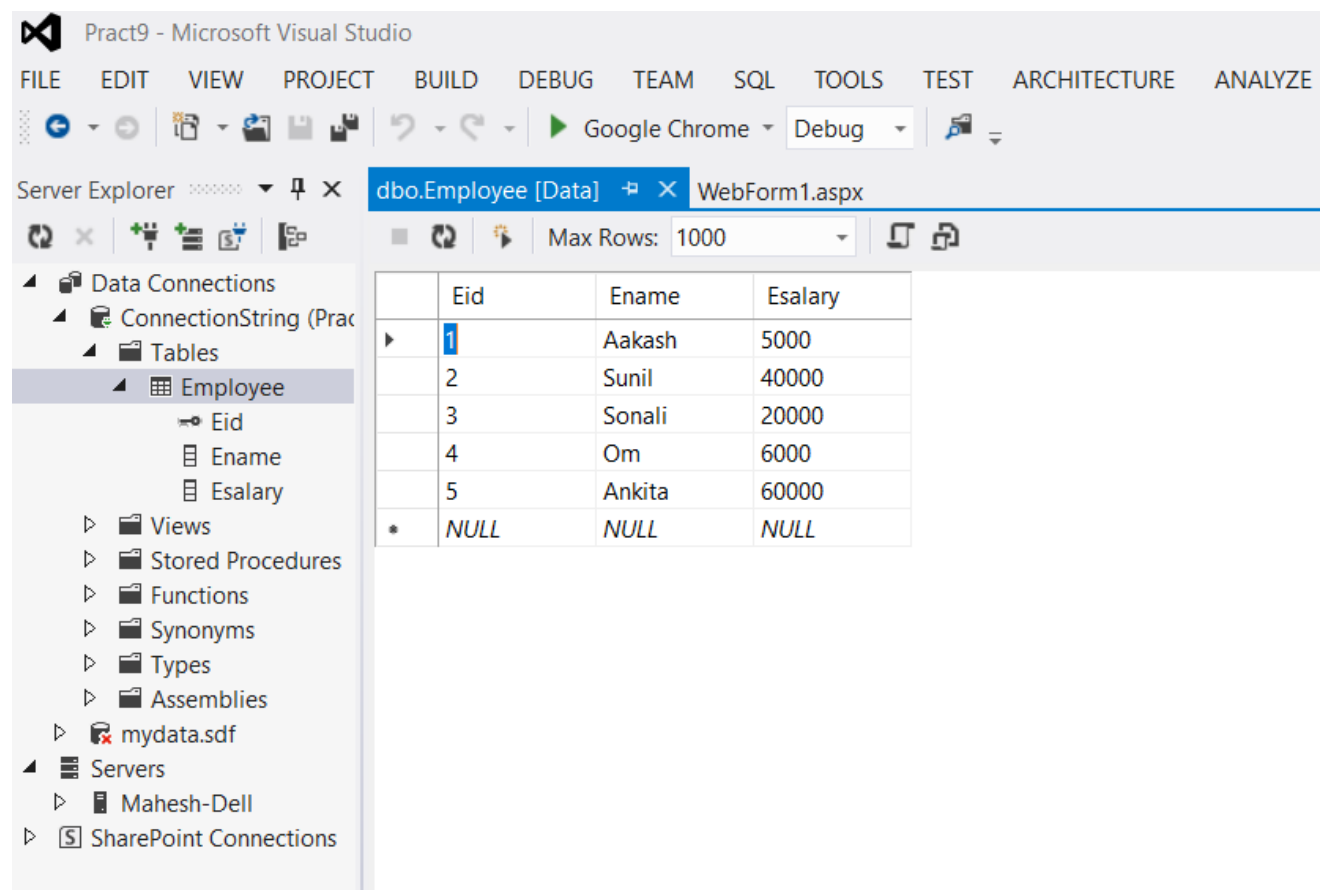
Create a web application to demonstrate use of GridView button column and GridView events.

### Web.config

```
<connectionStrings>
  <add name ="C1" connectionString=" " />
</connectionStrings>
```

### Step 1 :

Create a database with Employee Table:

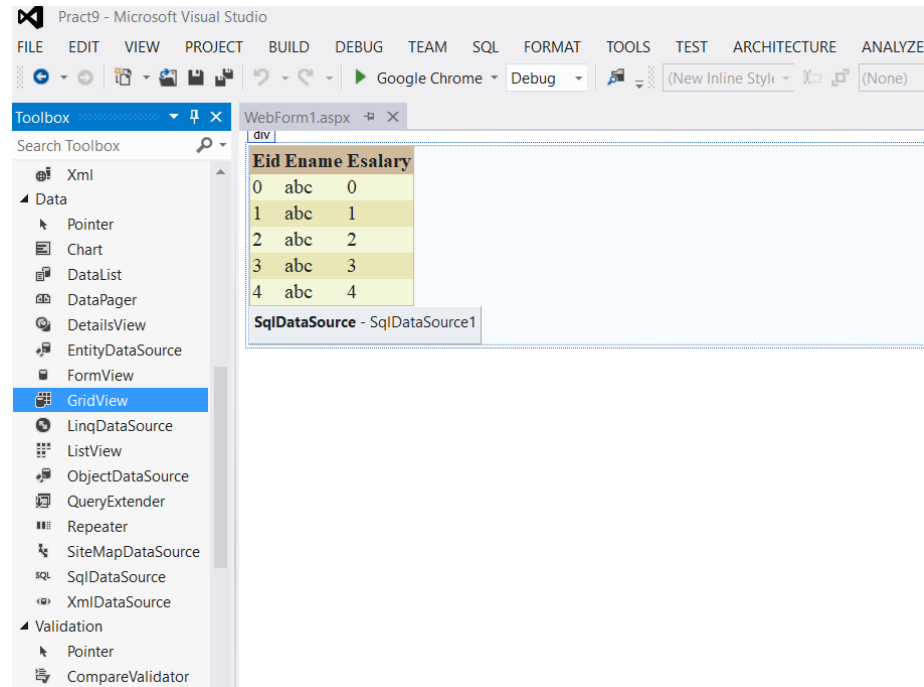


The screenshot shows the Microsoft Visual Studio interface. The title bar indicates the project is 'Pract9 - Microsoft Visual Studio'. The menu bar includes FILE, EDIT, VIEW, PROJECT, BUILD, DEBUG, TEAM, SQL, TOOLS, TEST, ARCHITECTURE, and ANALYZE. The toolbar shows various icons for file operations, debugging, and testing. The Server Explorer on the left shows a tree view of the project structure. Under 'Data Connections', there is a 'ConnectionString (Pract9)' connection. Under 'Tables', there is an 'Employee' table. The 'Employee' table has columns 'Eid', 'Ename', and 'Esalary'. The main window displays the 'WebForm1.aspx' page. The page shows a GridView control displaying the data from the 'Employee' table. The GridView has 5 rows of data and a 'Max Rows' dropdown set to 1000.

	Eid	Ename	Esalary
▶	1	Aakash	5000
	2	Sunil	40000
	3	Sonali	20000
	4	Om	6000
	5	Ankita	60000
*	NULL	NULL	NULL

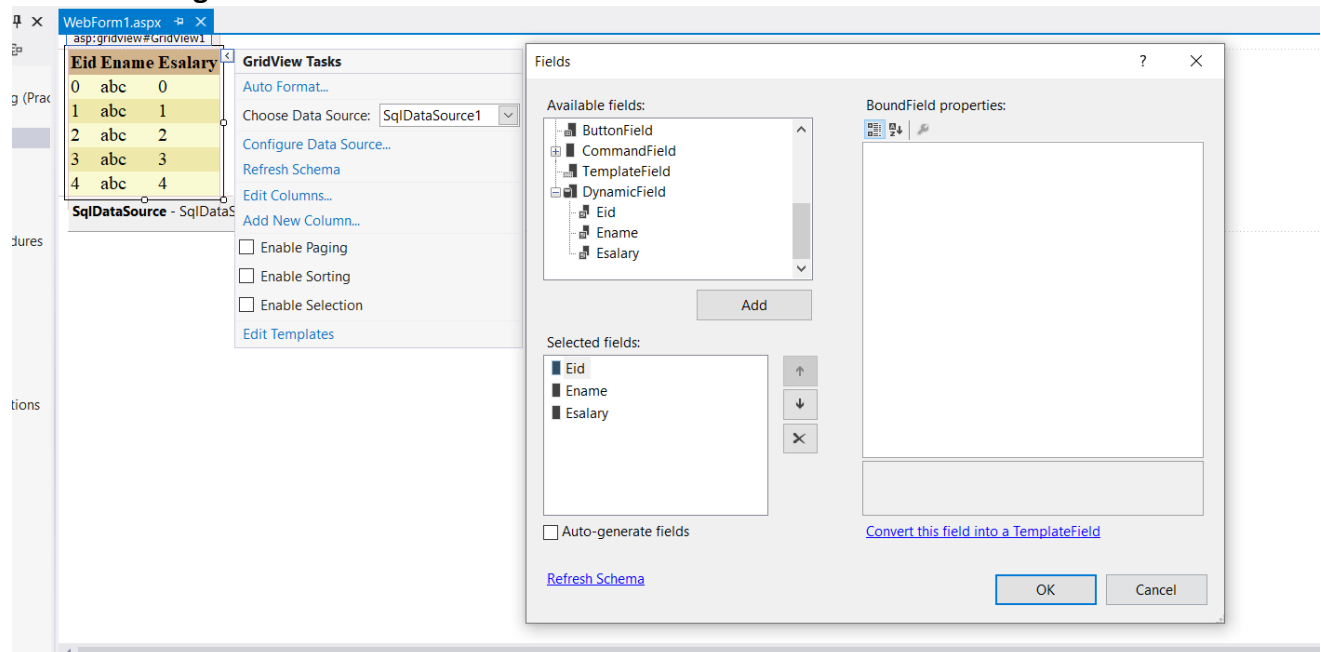
## Step 2 :

**Add a Grid View on WebForm1 and set the data source to the Database1**



## Step 3:

**Click on the right arrow near GridView and select Edit Columns:**



#### Step 4 :

Add Button Field from Available Fields and set the button field Properties:

The screenshot shows the Visual Studio IDE with a GridView control and the 'Fields' dialog box open.

**GridView Tasks:**

- Auto Format...
- Choose Data Source: **SqlDataSource1**
- Configure Data Source...
- Refresh Schema
- Edit Columns...
- Add New Column...
- ☐ Enable Paging
- ☐ Enable Sorting
- ☐ Enable Selection
- Edit Templates

**SqlDataSource - SqlDataSource1**

**Fields**

**Available fields:**

- ButtonField
- CommandField
- TemplateField
- DynamicField
- Eid
- Ename
- Esalary

**Selected fields:**

- Eid
- Ename
- Esalary
- Button

☐ Auto-generate fields

[Refresh Schema](#)

**ButtonField properties:**

- Accessibility**
  - AccessibleHeaderText
- Appearance**
  - ButtonType: **Button**
  - FooterText
  - HeaderImageUrl
  - HeaderText
  - ImageUrl
  - Text: **Show Employee Name**
- Behavior**
  - CausesValidation: False
  - CommandName: **Show**
  - InsertVisible: True
  - ShowHeader: False
  - SortExpression

**CommandName**  
The command associated with the button.

[Convert this field into a TemplateField](#)

**OK** **Cancel**

#### Step 5:

Add Label and remove its Text:

The screenshot shows the Visual Studio IDE with a GridView control. The GridView has a header row with columns 'Eid', 'Ename', and 'Esalary'. Below the header, there are five rows of data. Each row has a button field with the text 'Show Employee Name'.

**body**

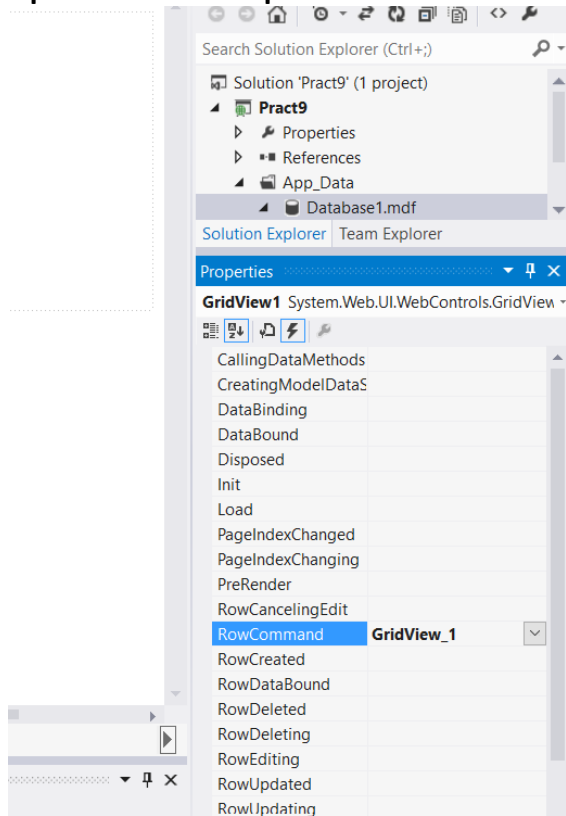
Eid	Ename	Esalary	
0	abc	0	Show Employee Name
1	abc	1	Show Employee Name
2	abc	2	Show Employee Name
3	abc	3	Show Employee Name
4	abc	4	Show Employee Name

**SqlDataSource - SqlDataSource1**

**[Label1]**

### Step 6 :

Open GridView Properties And set the Row Command and press enter:



### Step 7 :

Write a following code in WebForm1.aspx.cs :

```
protected void GridView_1(object sender, GridViewCommandEventArgs e)
{
    if (e.CommandName == "Show")
    {
        int index = int.Parse(e.CommandArgument.ToString());
        GridViewRow row = GridView1.Rows[index];
        String name = row.Cells[1].Text;
        Label1.Text = "Selected Employee name is " + name;
    }
}
```