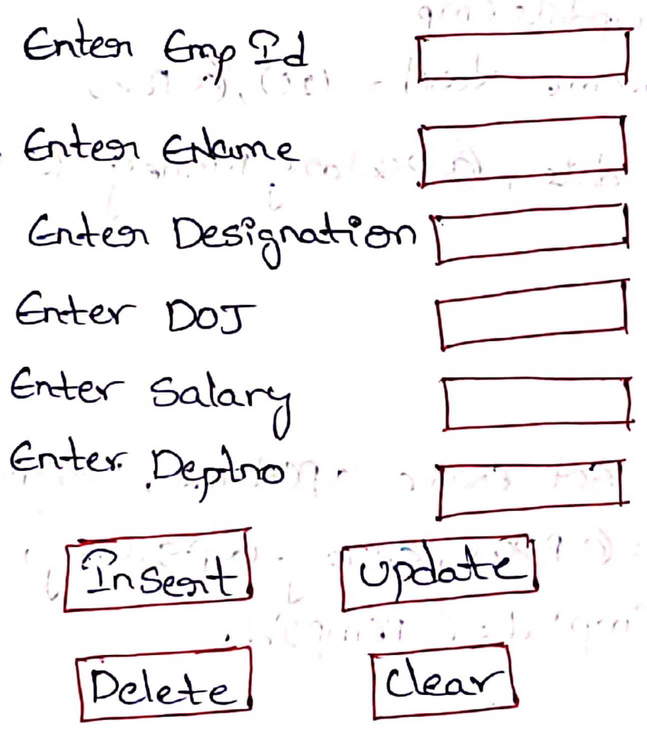
```
sql parameter P1 = new sql parameter ("@pEmpId",
                          SqlDbType.int);
P1. Value = txtEmpId.Text;
Cmd. parameters. Add(P1);
int Rows = Cmd. ExecuteNonQuery();
con.close();
Message box. show (Rows +" Record(s) deleted");
}
```

* Example to pass multiple parameters to the stored Procedures:



→ Go to Sql Server. Write the following code to insert the stored procedure

```
Create procedure Insert Emp
@pempId int, @pEName Varchar(20), @PDesignation
Varchar(20), @PDOJ date, @psalary Money, @pdeptNo int.
As
Begin
Insert into Empdetails Values (@pEmpId, @pEName,
@Pdesignation, @PDOJ, @Psalary, @PdeptNo)
END
```

Write following code to update stored procedure:

```
Create procedure updateEmp
@pEmpId int, @pEName Varchar(20), @pdesignation
Varchar(20), @PDOJ date, @psalary Money, @pdeptNo
int.
AS
Begin
Update Empdetails set EName=@pEName, Designation=
@Pdesignation, DOJ =@PDOJ, Salary=@psalary, DeptNo=
@PdeptNo Where EmpId =@pEmpId.
END;
```

~~Go to Visual Studio~~

→ Go to Visual studio and Write the following Code:-

```
using System.Data.Sqlclient;

string sqlconstring = "Server= ; UserId= ;"
Sql Connection Con; Sql command cmd; sqlparameter P1;
int Rows;
```

```
Private void form11_load (object Sender, EventArgs e)
{
 Con = new SqlConnection (sqlconstring);
}
Private void btnInsert_Click (     "     )
{
Cmd = new sqlCommand ("Insert Emp", con);
Cmd.CommandType = CommandType.Stored procedure;
P1 = new sql parameter ("@pEmpId", SqlDbType.Int);
P1.Value = txtempId.Text;
Cmd.parameters.Add (P1);
P1 = new sql parameter ("@pEName", sqlDbType.Varchar,
                                                    20);
P1.Value = txtEName.Text;
Cmd.parameters.Add (P1);
P1 = New sql parameter ("@pDesignation", sqlDbType.
                                        Varchar, 20);
P1.Value = txtDesignation.Text;
Cmd.parameters.Add (P1);
P1 = New sql parameter ("@pDOJ", SqlDbType.Date);
P1.Value = txtDOJ.Text;
Cmd.parameters.Add (P1);
P1 = New sql parameter ("@pSalary", sqlDbType.Money);
P1.Value = txtSalary.Text;
Cmd.parameters.Add (P1);
P1 = New Sql parameter ("@pdeptNo", SqlDbType.Int);
P1.Value = txtdeptNo.Text;
Cmd.parameters.Add (P1);
Con.open();
```

```
  Rows= Cmd. ExecuteNon Query ();
   Con. Close();
  Message box. show (Rows +"Record(s) Inserted");
}
Private void btnupdate_click (        "       )
{
 Cmd=new sql command ("Update emp", Con);
```

Same code to write in update what we have to write in Insert code.

```
 Message box. show (Rows + "Record(s) Updated);
}
Private void btn delete_click (       "       )
{
 Cmd = new sql Command ("Delete Emp1", Con);
 Cmd. Command Type= Command Type. Stored procedure;
Sql parameter P1=new Sql parameter ("@pEmpId", SqlDb
P1. Value = txt EmpId. Text;                  Type. Int);
Cmd. parameters. Add(P1);
 Con. open();
~~int~~ Rows= Cmd. ExecuteNon Query ();
Con. Close();
Message box. show (Rows +"Record(s) deleted");
}
```

```
Private Void btnclear_click (          )
{
foreach (Control c in this.controls)
  {
   if (c is Textbox)
    c.Text = "";
}
```

* In the above Example When We are using Cmd. parameters. Add (P1); All P1 Associated Properties Values Will be added to parameters collection. and Internally a collection table Will be created like.

ParamName	Datatype	Size	Value	~~type~~ direction
@PEmpId	Int		108	Input
@PEName	Varchar	20	Anand	Input
@PDesignation	Varchar	20	DB-Programmer	Input
@PDOJ	date		01/01/2023	Input
@Psalary	Money		65000	Input
@Pdept No.	Int		30	Input

* When front End Encounters cmd. Executenonquery(); method Then this parameters collection table Will be sent to database along With stored procedure Name. At database side, database Engine Will

Cross Verify the stored procedure Name, Will Map the parameters and their datatypes, Win Store the Values into the padameters and them Executy stored procedure.

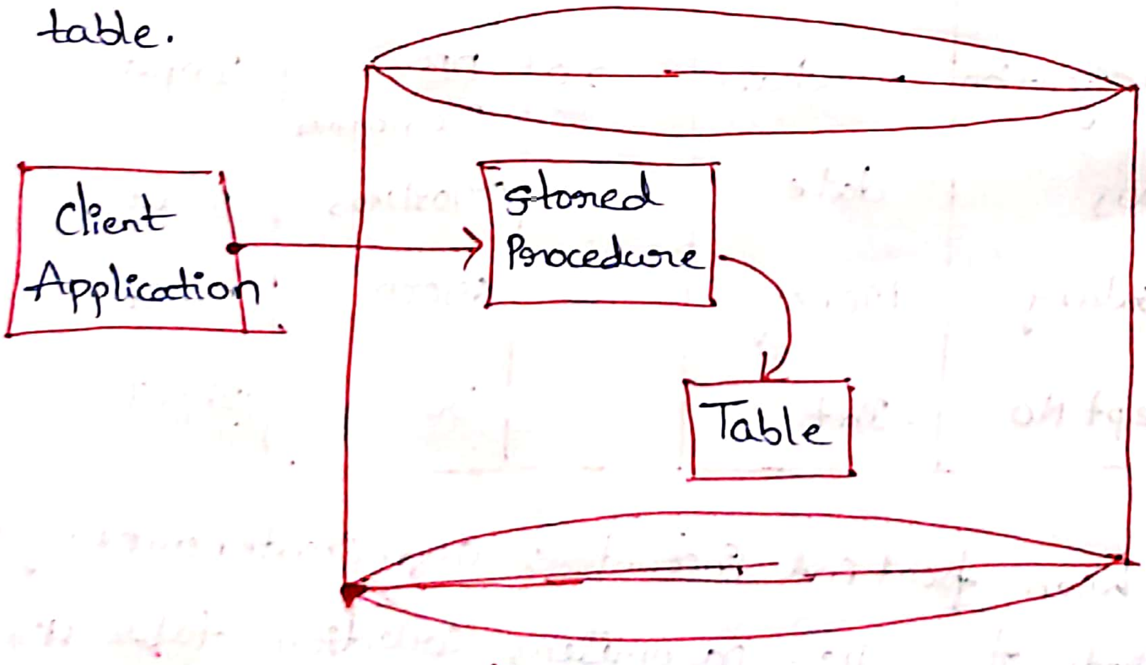
Parameters direction in stored procedures:-

* Stored procedures have parameters that supports three types of directions.

1. Input
2. Output
3. Input output

1. Input:-

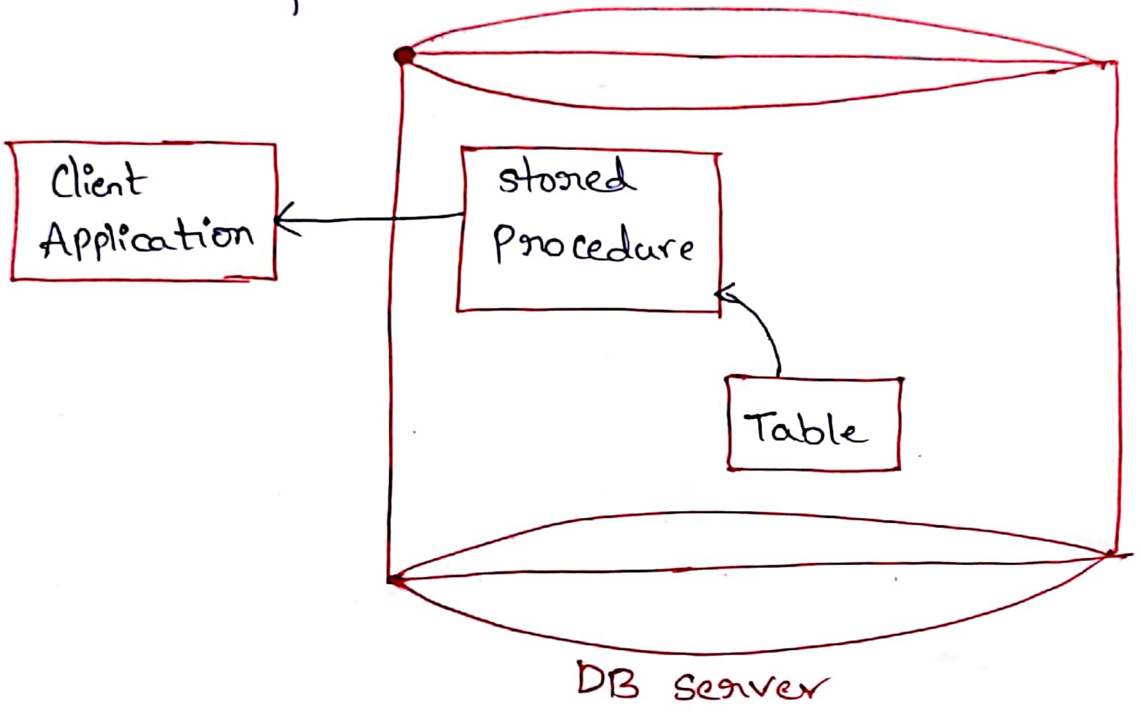
→ Input direction parameters are capable to take the data from client Application to Stored procedure and stored procedure to table.



DB Server

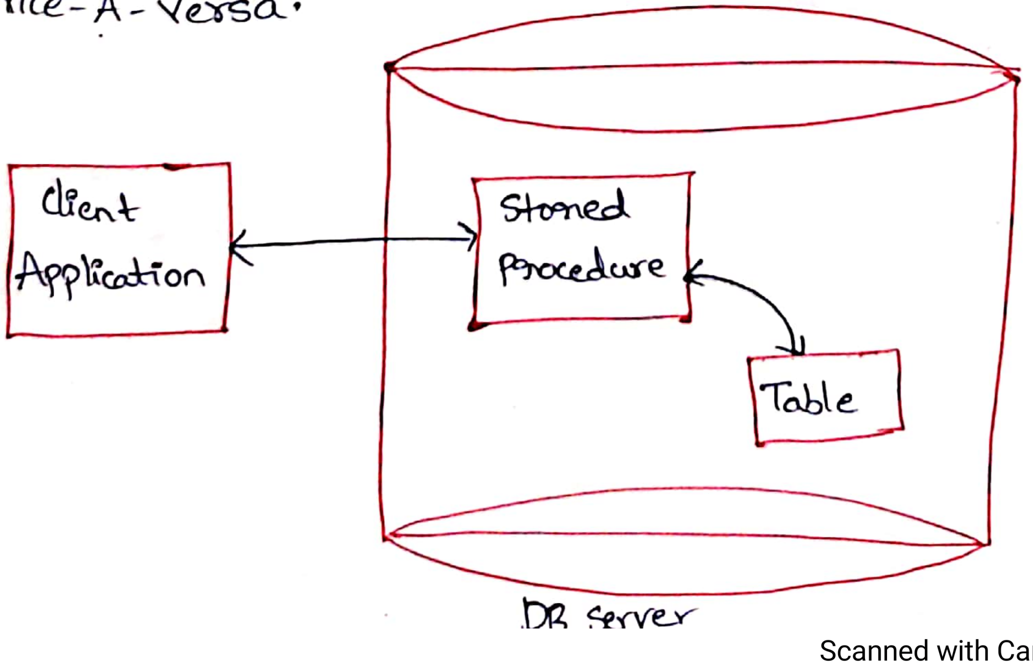
2. Output:-

Output direction parameters are capable to transfer the data from Table to storeprocedure and stored procedure to client Application.



3. Input output:-

These parameters are capable to transfer the data from client Application to stored ~~vice versa~~. Procedure, stored procedure to Table and Vice-A-Versa.



client Application
Stored Procedure
Table
DB server