**DOT Net Driver for OpenGauss:**

**Introduction:**

openGauss is an open-source relational database management system that is released with the Mulan PSL v2. with the kernel derived from PostgreSQL, openGauss is built on Huawei's years of experience in the database field and continuously provides competitive features tailored to enterprise-grade scenarios.

We need following dll to be use:

1. Microsoft.Bcl.AsyncInterfaces.dll
2. Npgsql.dll
3. Opengauss.dll
4. System.Runtime.CompilerServices.Unsafe.dll
5. System.Text.Json.dll
6. System.Threading.Channels.dll

All dll are in zipped folder. Please extract the zipped folder and keep all dll in any drive as your wish. Right click on your project and select all dll from your drive as a reference.

It required dot net framework v4.7.2

**Steps to Use this DLL in Windows:**

Step1: Put this DLL’s into application.

Step2: Give namespace

Step3: Create object of class.

Step4: Using that object use the methods inside it.

**Steps to Use this DLL in Linux(Cent-OS)**

Step1: Put this DLL’s into application.

Step2: Give namespace

Step3: Create object of class.

Step4: Using that object use the methods inside it.

Step5: Give the path of DLL’s in .csproj file.

1. **Connection:**

We have connection method name as OpenConnection to establish a proper connection to the database. This method required only one parameter. If connection established successfully it will execute queries otherwise it will give an error related to connection.

Ex:

string conn=”place your connection string here”;

OpenGauss objIn = new OpenGauss();

string connection = objIn.OpenConnection(conn);

where Opengauss= class file

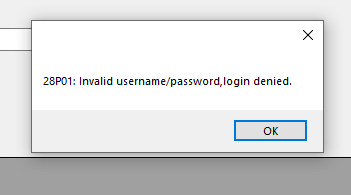
objIn= Object of class

Openconnection= method name which authenticate the connection.

1. if connection string is correct it will show records from the database



1. if anything wrong with connection it will show below message



1. **DDL and DML Commands:**
2. DDL Commands
3. Create
4. Drop
5. Alter
6. Truncate
7. Rename
8. **DML Commands**
9. Insert
10. Update
11. Delete

We create the common method name as ExecuteQuery to execute all this commands.

**Method Name: ExecuteQuery:**

This method required two parameter query and connection and returns an integer value. For all **DML Commands** it will return 1 if operation is successful, indicating that row is affected and 0 if operation is failed. If the operation is successful it will insert the data into database and if it is failed it will give an error.

For all **DDL commands** it will return -1 if operation is successful and 0 if operation is failed. Using this result, we can handle exceptions.

The query contains the operation information to be performed. It required suitable connection string to connect openGauss database. If the method gets proper query and connection, it will perform the operation successfully and gives the result.

Ex:

OpenGauss objIn = new OpenGauss();

string row = objIn.ExecuteQuery(query, conn);

if (row == "1")

{

“Display any user friendly message if success”

}

Else{

“Display any user friendly message if operation failed”

}

where objIn== object of class opengauss

1. **DQL Command**
2. **Select**

The DQL command contains the select statement which gives all the data from the table.

**Method Name: ExecuteSelect:**

This method also required two parameter and it will return a data table having data from a selected table. When we are passing query to the method it is required to pass table and in the query. Using that query and connection it will give data from mentioned table. If it finds wrong table name or improper connection, it will throw an error.it will count the rows from the table,if it is more than 0 then it will display the result.

Ex:

DataTable ds = objIn.ExecuteSelect(query,conn);

if (ds.Rows.Count > 0)

{

“display data into gridview”

}

Else{

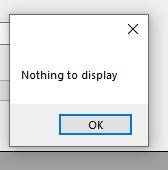
“if no data is there display empty result”

}

if count>0 it will display below result from database:



If count <0 it will display message



1. **Commit and Rollback:**

We have method for commit and rollback. Rollback and Commit are transaction statements that are called Data Control Language and are used to ensure the integrity of data in databases.

**Method Name: ExecuteTransactionQuery:**

This method required three parameters. We need to pass two queries defining table name in which we are performing operation. The third parameter is connection string. If everything is correct with method, it will return integer values 2 otherwise 0. If we pass wrong table name or wrong query it will automatically rollback previous data without affecting the result.

Ex:

string row = objIn.ExecuteTransactionQuery(query1, query2, conn);

if (row == "2")

{

“Transaction successful”

}

Else

{

“Transaction failed”

}

1. **Stored Procedure:**

When we are working with stored procedure all the command returns -1 if operation is successful otherwise 0.

Ex:

string query= "CALL sp\_InsertData(‘”+paramter1+”’, ‘”+paramter1+”’…)

string row = objIn.ExecuteSPQuery(query, conn);

if (row=="-1")

{

“Display success message”

}

else

{

“Display failure message”

}