using Oracle.DataAccess.Client;

previously add reference Oracle.DataAccess browse where Oracle.DataAccess.dll is

Agregue referencia a projecto donde se encuentra Convergys.CommandCenter.DataAccess para utilizar funciones de connection, etc

namespace WFMnow.app.alerts.SearchAgentAlert

{

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

{

**// IMPORTANTE INICIALIZAR VARIABLE**

Database database = new Database("CommandConnection");

// no podra referirlo si no agrefo en referncias Convergys.CommandCenter.DataAccess

En donde se encuentra clase sealed Database

namespace Convergys.CommandCenter.DataAccess

{

public sealed class Database

DATASET

internal IList<AmnestyEntity> GetAmnesties()

{

AmnestyEntity amnestyEntity = null;

IList<AmnestyEntity> amnestyEntities = null;

StringBuilder selectSQL = new StringBuilder();

selectSQL.Append(" Select c.client\_descr client\_name, l.ecp\_lob\_name, ");

selectSQL.Append(" to\_char(a.start\_date,'mm/dd/yyyy') start\_date, ");

selectSQL.Append(" to\_char(a.start\_date,'HH:MI AM') ||' - '||to\_char(a.start\_date + 1/48,'HH:MI AM') as interval, ");

selectSQL.Append(" a.amnesty\_id ");

selectSQL.Append(" from CC\_AMNESTY a, REF\_LOB l, REF\_CLIENT c ");

selectSQL.Append(" where a.lob\_id = l.ecp\_lob\_id ");

selectSQL.Append(" and l.client\_id = c.client\_id ");

selectSQL.Append(" order by lower(c.client\_descr),lower(l.ecp\_lob\_name) ");

using (OracleCommand commandEmp = database.GetSqlStringCommand(selectSQL.ToString()))

{

using (OracleConnection connection = database.CreateConnection())

{

try

{

connection.Open();

using (OracleDataReader reader = database.ExecuteReader(commandEmp, connection))

{

if (reader.HasRows)

{

amnestyEntities = new List<AmnestyEntity>();

while (reader.Read())

{

amnestyEntity = new AmnestyEntity();

amnestyEntity.ClientName = reader["client\_name"].ToString();

amnestyEntity.LobName = reader["ecp\_lob\_name"].ToString();

amnestyEntity.strStartDate = reader["start\_date"].ToString(); ;

amnestyEntity.Interval = reader["interval"].ToString();

amnestyEntity.AmnestyID = reader["amnesty\_id"].ToInt64();

amnestyEntities.Add(amnestyEntity);

}

}

}

}

finally

{

database.CloseConnection(connection);

}

}

}

return amnestyEntities;

}

SCALAR

private bool validateparams(string met, string notmet, string exclstr)

{

bool result = false;

using (OracleCommand command = database.GetSqlStringCommand(met.ToString()))

{

using (OracleConnection connection = database.CreateConnection())

{

try

{

connection.Open();

using (OracleDataReader reader = database.ExecuteReader(command, connection, true))

{

result = true;

}

//connection.Open();

using (OracleDataReader reader = database.ExecuteReader(command, connection, true))

{

result = true;

}

//connection.Open();

using (OracleDataReader reader = database.ExecuteReader(command, connection, true))

{

result = true;

}

}

catch (Exception ex)

{

ExceptionLogging.LogException(ex);

throw new BusinessLogicException("Epression is not valid");

//return false;

}

finally

{

database.CloseConnection(connection);

}

}

}

return result;

}I

STORE PROCEDURES

nternal IList<CTEntity> GetCTs(string lobIDs, string linkedOnly, string siteIDs)

{

IList<CTEntity> CTList = null;

string strErrMsg = "Error";

using (OracleCommand command = database.GetStoredProcCommand("PKG\_COMMON.GetCTs"))

{

database.AddInParameter(command, "p\_LOBids", OracleDbType.Varchar2, lobIDs);

database.AddInParameter(command, "p\_linkedonly", OracleDbType.Varchar2, linkedOnly);

if (siteIDs != null)

{

database.AddInParameter(command, "p\_Siteids", OracleDbType.Varchar2, siteIDs);

}

database.AddCursorOutParameter(command, "p\_CTs");

database.AddOutParameter(command, "P\_errmsg", OracleDbType.Varchar2, 4000);

using (OracleConnection connection = database.CreateConnection())

{

try

{

connection.Open();

using (OracleDataReader reader = database.ExecuteReader(command, connection))

{

strErrMsg = database.GetParameterValue(command, "p\_errmsg").ToString();

if (strErrMsg.Trim().Length > 0 && strErrMsg != "null")

{

throw new BusinessLogicException(strErrMsg);

}

else

{

if (reader.HasRows)

{

CTList = new List<CTEntity>();

}

while (reader.Read())

{

CTList.Add(new CTEntity

{

CTID = reader["CT\_ID"].ToInt64(),

LobID = reader["lob\_id"].ToInt64(),

IEXServer = reader["App\_Code"].TryToString(),

CustomerID = reader["customer\_id"].ToInt64(),

//,AlertLevel = Convert.ToChar(dr["alert\_level"])

CTName = reader["App\_Code"].TryToString() + "-" + reader["customer\_id"].ToInt64() + "-" + reader["CT\_ID"].ToInt64(),

CTKey = reader["App\_Code"].TryToString() + "~" + reader["Customer\_ID"].ToInt64() + "~" + reader["CT\_ID"].ToInt64(),

SiteID = reader["Site\_ID"].ToInt64(),

SiteName = reader["Site\_Descr"].TryToString()

});

}

}

}

}

finally

{

database.CloseConnection(connection);

}

}