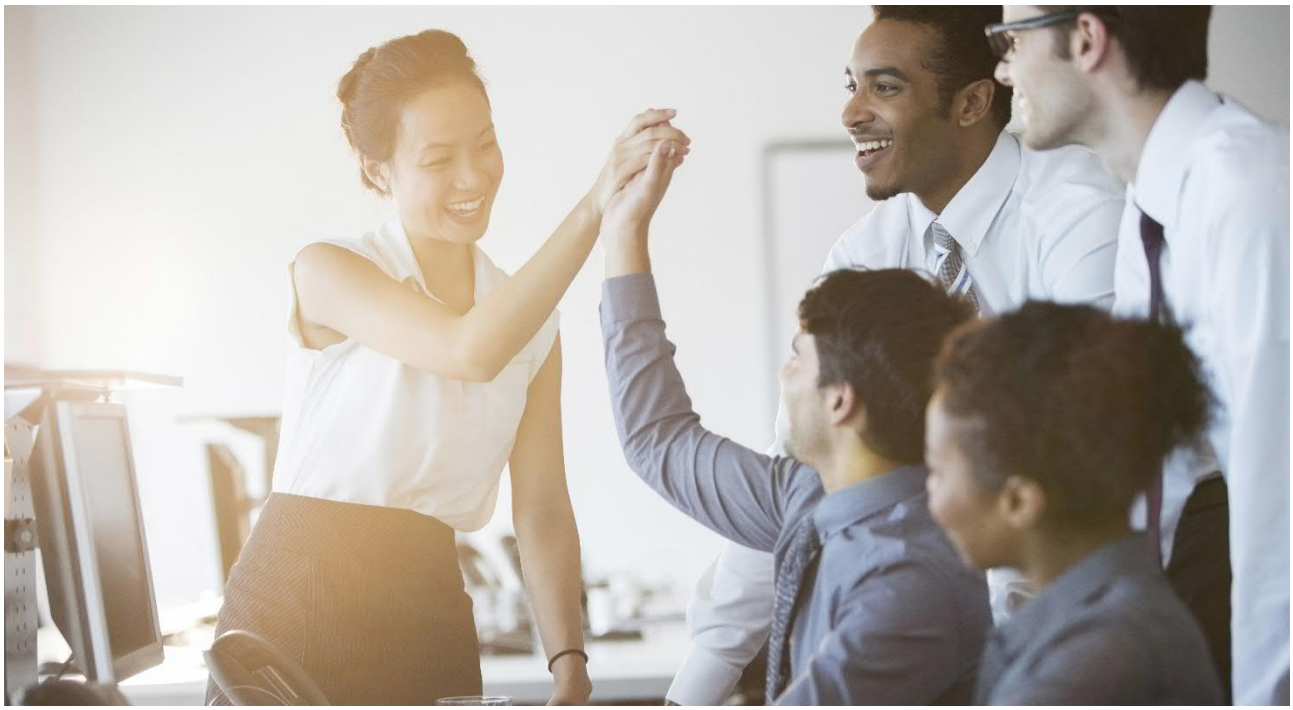


Exercise Solution: Metadata

Service Layer

PUBLIC

INTRODUCTION

In this exercise, you will perform the following tasks:

1. Create User-defined Table
2. Create User-defined Fields
3. Create User-defined Key
4. Create User-defined Object
5. Insert records into the User-defined Object

PREREQUISITE:

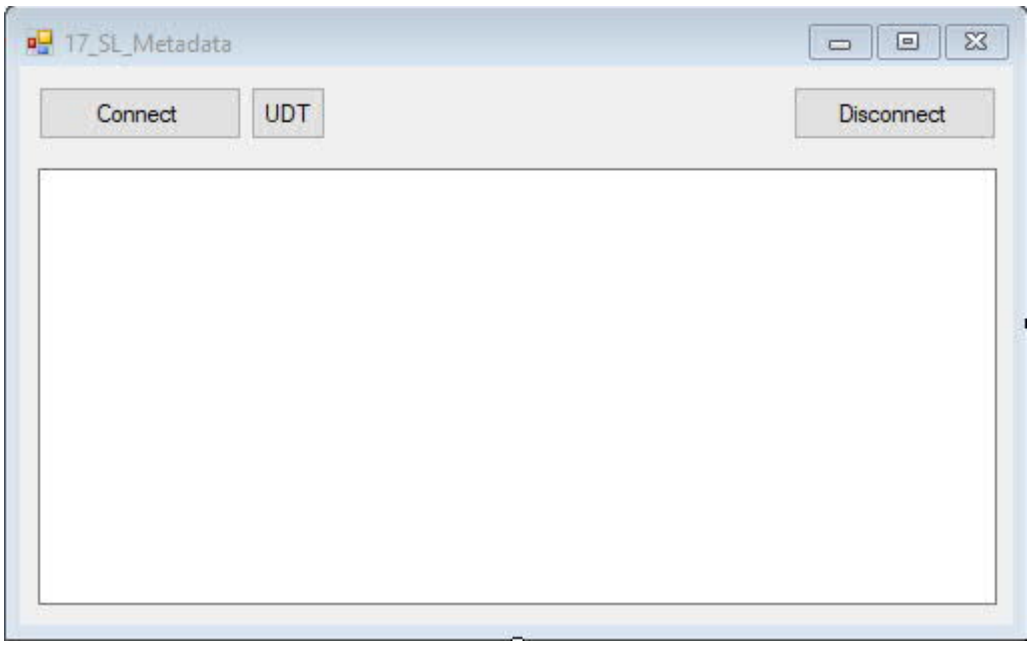
- Use the demo database for SAP Business One, version for SAP HANA
- Credentials: User code: **manager**

GUIDELINES:

The screenshots provided here are for your reference only and may differ from the actual screenshots in your system.

1. TASK - CREATE USER-DEFINED TABLE

1.1. On your Visual Studio project create a new button called “UDT”



1.2. Define a variable for the EventArg class – ensure it is defined as a member of the add-on application class or globally.

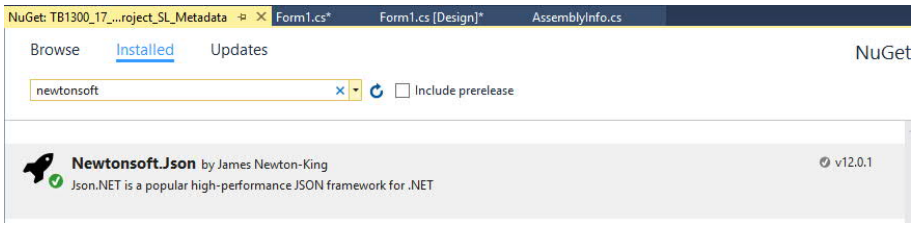
```
public ReceivingResponseEventArgs webResponse = null;
```

1.3. Define a class for handling the errors from web service response.

```
public class wsResponseError
 {
     public err error { get; set; }

     public class err
     {
         public string code { get; set; }
         public msg message { get; set; }
     }

     public class msg
     {
         public string lang { get; set; }
         public string value { get; set; }
     }

 }
```

1.4. Install NuGet packages - Newtonsoft.Json

Project → Manage NuGet Packages



1.5. Create a generic function to handle the User-defined Table creation.

```
        private void AddUserDefinedTable(string TableName, string Description, string Type)
        {
            try
            {
                Uri httpAction = new Uri("/UserTablesMD", UriKind.Relative);
                BodyOperationParameter[] body = new BodyOperationParameter[3];
                body[0] = new BodyOperationParameter("TableName", TableName);
                body[1] = new BodyOperationParameter("TableDescription", Description);
                body[2] = new BodyOperationParameter("TableType", Type);
                MyServiceLayer.SAPB1.UserTablesMD UserDefinedTable = null;

                UserDefinedTable =
(MyServiceLayer.SAPB1.UserTablesMD)slContext.Execute<MyServiceLayer.SAPB1.UserTablesMD>(http
Action, "POST", true, body).SingleOrDefault();

                if (webResponse.ResponseMessage.StatusCode == 201)
                    txtMain.AppendText("Table created, TableName = " + TableName +
System.Environment.NewLine);
            }
            catch (Exception ex)
            {
                if (webResponse.ResponseMessage.StatusCode <= 199 ||
webResponse.ResponseMessage.StatusCode >= 300)
                {
                    txtMain.AppendText("StatusCode = " +
webResponse.ResponseMessage.StatusCode + System.Environment.NewLine);
                    wsResponseError currentError =
Newtonsoft.Json.JsonConvert.DeserializeObject<wsResponseError>(ex.InnerException.Message);
                    txtMain.AppendText("ErrorCode = " + currentError.error.code +
System.Environment.NewLine);
                    txtMain.AppendText("ErrorMessage = " + currentError.error.message.value
+ System.Environment.NewLine);
                }
                else
                {
                    txtMain.AppendText(ex + System.Environment.NewLine);
                }
                webResponse = null;
                return;
            }
        }
```

1.6. Assign the *ResponseMessage* from class *ReceivingResponseEventArgs* to the *webResponse* class. This can be done in the ***SLReceivingResponse*** function.

```
void SLReceivingResponse(object sender, ReceivingResponseEventArgs e)
 {
     if (null == e.ResponseMessage)
         return;

     string strMessage = e.ResponseMessage.GetHeader("Set-Cookie");

     if (!string.IsNullOrEmpty(strMessage))
     {
         CookieString = strMessage.Replace(',', ';');
     }
     webResponse = e;

         }
```

1.7. Add a User-Defined Table (use namespace “TB1_” as a prefix…), but do not add any fields to the table yet. The newly created button for UDF should call the function ***AddUserDefinedTable***.

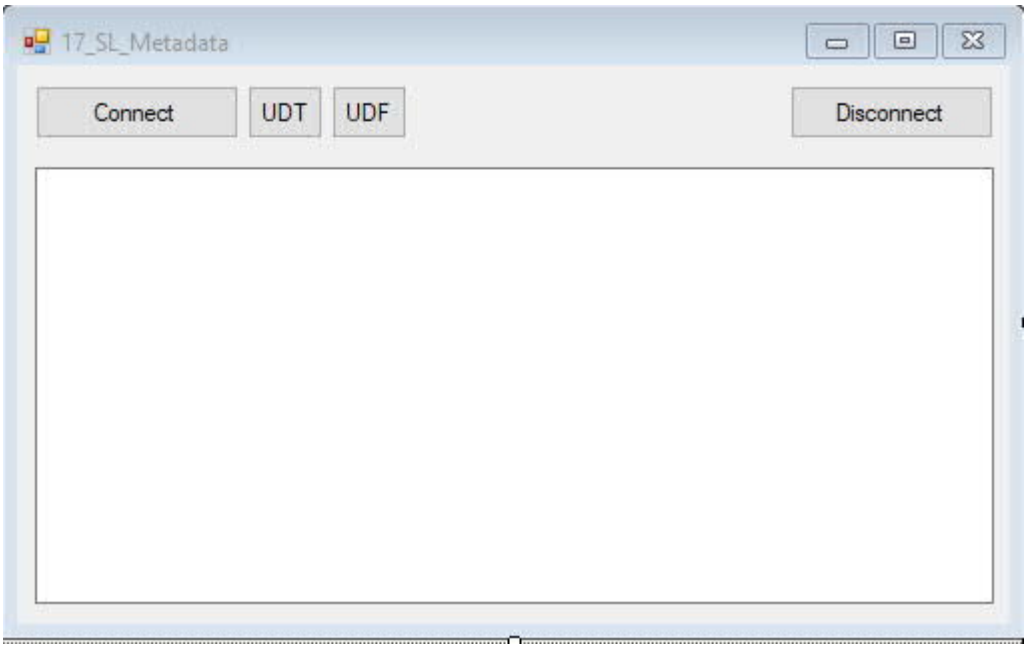
Table name: TB1_TABLE2
Table description: UDO Document
Table type: Document

Table name: TB1_TABLE3
Table description: UDO Document Row
Table type: Document Lines

```
private void btnUDT_Click(object sender, EventArgs e)
 {
     AddUserDefinedTable("TB1_MYTABLE2", "UDO Document", "bott_Document");
     AddUserDefinedTable("TB1_MYTABLE3", "UDO Document Row", "bott_DocumentLines");
  }
```

2. TASK – CREATE USER-DEFINED FIELDS

2.1. On your Visual Studio project create a new button called “UDF”



2.2. Create a generic function to handle the User-defined Field creation.

```
        private void AddUserDefinedField(string Name, string Type, int Size, string
Description, string TableName)
        {
            try
            {
                Uri httpAction = new Uri("/UserFieldsMD", UriKind.Relative);
                BodyOperationParameter[] body = new BodyOperationParameter[5];
                body[0] = new BodyOperationParameter("Name", Name);
                body[1] = new BodyOperationParameter("Type", Type);
                body[2] = new BodyOperationParameter("Size", Size);
                body[3] = new BodyOperationParameter("Description", Description);
                body[4] = new BodyOperationParameter("TableName", TableName);
                MyServiceLayer.SAPB1.UserFieldMD UserDefinedField = null;

                UserDefinedField =
(MyServiceLayer.SAPB1.UserFieldMD)slContext.Execute<MyServiceLayer.SAPB1.UserFieldMD>(httpAc
tion, "POST", true, body).SingleOrDefault();

                if (webResponse.ResponseMessage.StatusCode == 201)
                    txtMain.AppendText("Field created, FieldName = " + Name +
System.Environment.NewLine);
            }
            catch (Exception ex)
            {
                if (webResponse.ResponseMessage.StatusCode <= 199 ||
webResponse.ResponseMessage.StatusCode >= 300)
                {
                    txtMain.AppendText("StatusCode = " +
webResponse.ResponseMessage.StatusCode + System.Environment.NewLine);
```

```
                    wsResponseError currentError =
Newtonsoft.Json.JsonConvert.DeserializeObject<wsResponseError>(ex.InnerException.Message);
                    txtMain.AppendText("ErrorCode = " + currentError.error.code +
System.Environment.NewLine);
                    txtMain.AppendText("ErrorMessage = " + currentError.error.message.value
+ System.Environment.NewLine);
                }
                else
                {
                    txtMain.AppendText(ex + System.Environment.NewLine);
                }
                webResponse = null;
                return;
            }
        }
```

2.3. Add the following User-Defined Fields to your new User-Defined Table.

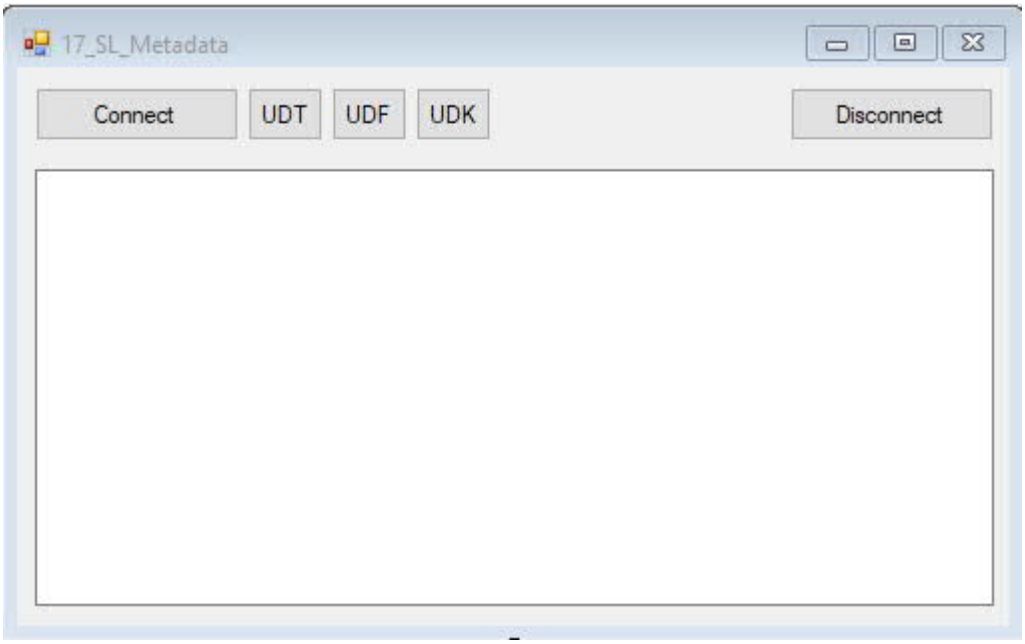
You will need to create an instance of the UserTablesMD object in order to add a field to the User Table. It is recommended that after you create your table you set this object variable to “Nothing” so that its properties do not inadvertently carry forward to the next table or field you are creating.

Field Name	Field Description	Field Type	Field EditSize
udf1	field 01	db_Alpha	20
udf2	field 02	db_Alpha	20

```
    private void btnUDF_Click(object sender, EventArgs e)
     {
         AddUserDefinedField("udf1", "db_Alpha", 20, "field 01", "@TB1_MYTABLE2");
         AddUserDefinedField("udf2", "db_Alpha", 20, "field 02", "@TB1_MYTABLE3");
     }
```

3. TASK – CREATE USER-DEFINED KEY

3.1. On your Visual Studio project create a new button called “UDK”



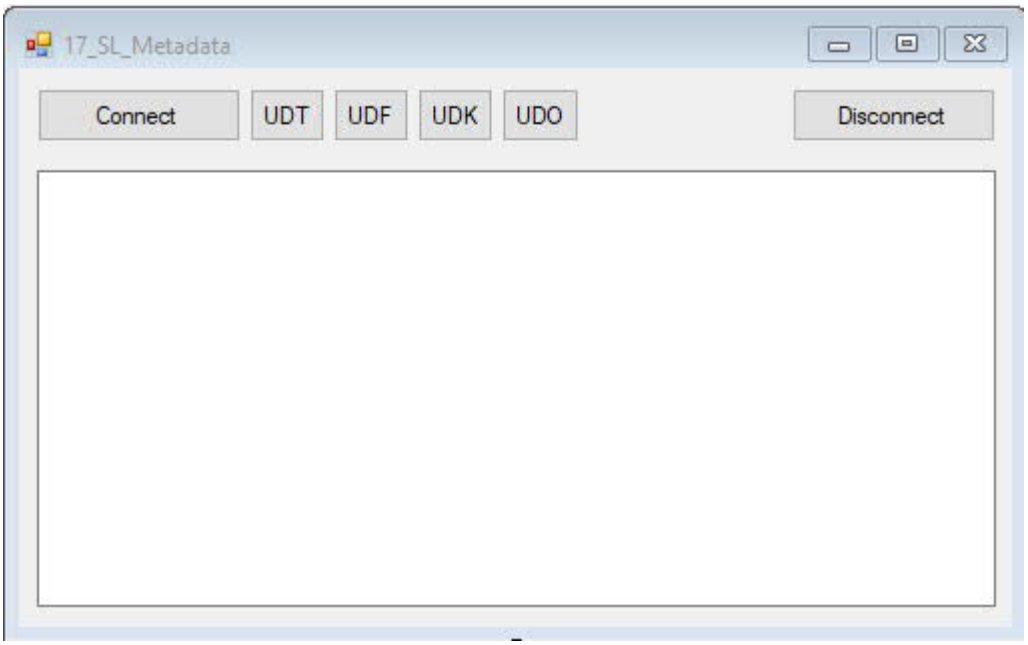
3.2. Defined the button click function to create the User-defined Key

```
private void btnUDK_Click(object sender, EventArgs e)
        {
            var httpWebRequest =
WebRequest.Create("https://YourHanaServerAddress:50000/b1s/v1/UserKeysMD") as
HttpWebRequest;
            try
            {
                httpWebRequest.Method = "POST";
                httpWebRequest.AllowAutoRedirect = false;
                httpWebRequest.Timeout = 30 * 1000;
                httpWebRequest.ServicePoint.Expect100Continue = false;
                httpWebRequest.CookieContainer = new CookieContainer();

                string[] cookieItems = CookieString.Split(';');
                foreach (var cookieItem in cookieItems)
                {
                    string[] parts = cookieItem.Split('=');
                    if (parts.Length == 2)
                    {
                        httpWebRequest.CookieContainer.Add(httpWebRequest.RequestUri, new
Cookie(parts[0].Trim(), parts[1].Trim()));
                    }
                }

                using (var streamWriter = new
StreamWriter(httpWebRequest.GetRequestStream()))
                {
```

```
                    string json = "{\"TableName\": \"@TB1_MYTABLE2\", \"KeyIndex\": 0,
\"KeyName\": \"key1\", \"Unique\": \"tYES\", \"UserKeysMD_Elements\": [{\"ColumnAlias\":
\"udf1\"}]}";

                    streamWriter.Write(json);
                    streamWriter.Flush();
                    streamWriter.Close();
                }

                var httpResponse = (HttpWebResponse)httpWebRequest.GetResponse() as
HttpWebResponse;
                string responseContent = null;
                if (httpResponse.StatusCode == HttpStatusCode.Created)
                    using (var streamReader = new
StreamReader(httpResponse.GetResponseStream()))
                    {
                        responseContent = streamReader.ReadToEnd();
                        var oResult =
Newtonsoft.Json.JsonConvert.DeserializeObject<System.Collections.Generic.IDictionary<string,
object>>(responseContent);
                        var keyName = oResult["KeyName"].ToString();
                        txtMain.AppendText("Key created, KeyName =  " + keyName +
System.Environment.NewLine);
                    }
                else
                    txtMain.AppendText("Error: " + System.Environment.NewLine);

            }
            catch (Exception ex)
            {
                txtMain.AppendText(ex + System.Environment.NewLine);
                return;
            }

}
```

4. TASK – CREATE USER-DEFINED OBJECT

4.1. On your Visual Studio project create a new button called “UDO”



4.2. Defined the button click function to create the User-defined Object

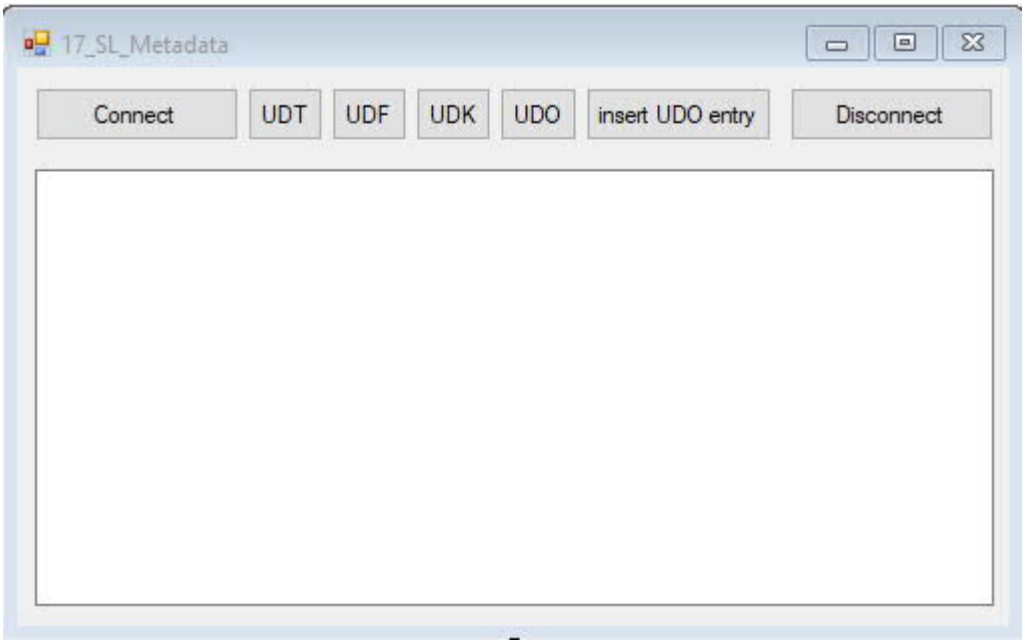
```
private void btnUDO_Click(object sender, EventArgs e)
        {
            try
            {
                var httpWebRequest =
WebRequest.Create("https://YourHanaServerAddress:50000/b1s/v1/UserObjectsMD") as
HttpWebRequest;
                httpWebRequest.Method = "POST";
                httpWebRequest.AllowAutoRedirect = false;
                httpWebRequest.Timeout = 30 * 1000;
                httpWebRequest.ServicePoint.Expect100Continue = false;
                httpWebRequest.CookieContainer = new CookieContainer();

                string[] cookieItems = CookieString.Split(';');
                foreach (var cookieItem in cookieItems)
                {
                    string[] parts = cookieItem.Split('=');
                    if (parts.Length == 2)
                    {
                        httpWebRequest.CookieContainer.Add(httpWebRequest.RequestUri, new
Cookie(parts[0].Trim(), parts[1].Trim()));
                    }
                }

                using (var streamWriter = new
StreamWriter(httpWebRequest.GetRequestStream()))
                {
                    string json = "{\"Code\": \"MyOrder\", \"Name\": \"My Orders\",
\"TableName\": \"TB1_MYTABLE2\", \"ObjectType\": \"boud_Document\", " +
```



```
                        "\"CanDelete\": \"tYES\", \"CanFind\": \"tYES\",
\"UseUniqueFormType\": \"tYES\", \"EnableEnhancedForm\": \"tYES\", \"RebuildEnhancedForm\":
\"tYES\", " +
                        "\"UserObjectMD_ChildTables\": [{\"TableName\": \"TB1_MYTABLE3\",
\"ObjectName\": \"MyOrderLines\"}], " +
                        "\"UserObjectMD_FindColumns\": [ " +
                        "{\"Code\": \"MYTABLE2\", \"ColumnNumber\": \"1\", \"ColumnAlias\":
\"DocNum\", \"ColumnDescription\": \"DocNum\"}, " +
                        "{\"Code\": \"MYTABLE2\", \"ColumnNumber\": \"2\", \"ColumnAlias\":
\"CreateDate\", \"ColumnDescription\": \"CreateDate\"}," +
                        "{\"Code\": \"MYTABLE2\", \"ColumnNumber\": \"3\", \"ColumnAlias\":
\"UpdateDate\", \"ColumnDescription\": \"UpdateDate\"}, " +
                        "{\"Code\": \"MYTABLE2\", \"ColumnNumber\": \"4\", \"ColumnAlias\":
\"U_udf1\", \"ColumnDescription\": \"U_udf1\"}]}";

                    streamWriter.Write(json);
                    streamWriter.Flush();
                    streamWriter.Close();
                }

                var httpResponse = (HttpWebResponse)httpWebRequest.GetResponse() as
HttpWebResponse;
                string responseContent = null;
                if (httpResponse.StatusCode == HttpStatusCode.Created)
                {
                    using (var streamReader = new
StreamReader(httpResponse.GetResponseStream()))
                    {
                        responseContent = streamReader.ReadToEnd();
                        var oResult =
Newtonsoft.Json.JsonConvert.DeserializeObject<System.Collections.Generic.IDictionary<string,
object>>(responseContent);
                        var entry = oResult["Code"].ToString();
                        txtMain.AppendText("UDO created, UDOCode = " + entry +
System.Environment.NewLine);
                    }
                }
                else
                    txtMain.AppendText("Error: " + System.Environment.NewLine);
            }
            catch (Exception ex)
            {
                txtMain.AppendText(ex + System.Environment.NewLine);
                return;
            }

        }
```

5. TASK – INSERT RECORDS INTO THE USER-DEFINED OBJECT

5.1. On your Visual Studio project create a new button called “UDO”



5.2. Defined the button click function to insert entries to the User-defined Object

```
private void btnInsertUDOEntry_Click(object sender, EventArgs e)
        {
            var httpWebRequest = WebRequest.Create("https://
YourHanaServerAddress:50000/b1s/v1/MyOrder") as HttpWebRequest;

            try
            {
                httpWebRequest.Method = "POST";
                httpWebRequest.AllowAutoRedirect = false;
                httpWebRequest.Timeout = 30 * 1000;
                httpWebRequest.ServicePoint.Expect100Continue = false;
                httpWebRequest.CookieContainer = new CookieContainer();

                string[] cookieItems = CookieString.Split(';');
                foreach (var cookieItem in cookieItems)
                {
                    string[] parts = cookieItem.Split('=');
                    if (parts.Length == 2)
                    {
                        httpWebRequest.CookieContainer.Add(httpWebRequest.RequestUri, new
Cookie(parts[0].Trim(), parts[1].Trim()));
                    }
                }

                using (var streamWriter = new
StreamWriter(httpWebRequest.GetRequestStream()))
                {
                    string currentTimeStamp = DateTime.Now.ToString("HH:mm:ss.fff");
```



```
                    string json = "{\"U_udf1\": \"" + currentTimeStamp + "\",
\"MyOrderLinesCollection\":[{\"U_udf2\": \"my comment\"}]}";

                    streamWriter.Write(json);
                    streamWriter.Flush();
                    streamWriter.Close();
                }

                var httpResponse = (HttpWebResponse)httpWebRequest.GetResponse() as
HttpWebResponse;
                string responseContent = null;
                if (httpResponse.StatusCode == HttpStatusCode.Created)
                    using (var streamReader = new
StreamReader(httpResponse.GetResponseStream()))
                    {
                        responseContent = streamReader.ReadToEnd();
                        var oResult =
Newtonsoft.Json.JsonConvert.DeserializeObject<System.Collections.Generic.IDictionary<string,
object>>(responseContent);
                        var udoObject = oResult["Object"].ToString();
                        var udoDocNum = oResult["DocNum"].ToString();
                        txtMain.AppendText(udoObject + " created, DocNum =  " + udoDocNum +
System.Environment.NewLine);
                    }
                else
                    txtMain.AppendText("Error: " + System.Environment.NewLine);

            }
            catch (Exception ex)
            {
                txtMain.AppendText(ex + System.Environment.NewLine);
                return;
            }

        }
```

www.sap.com