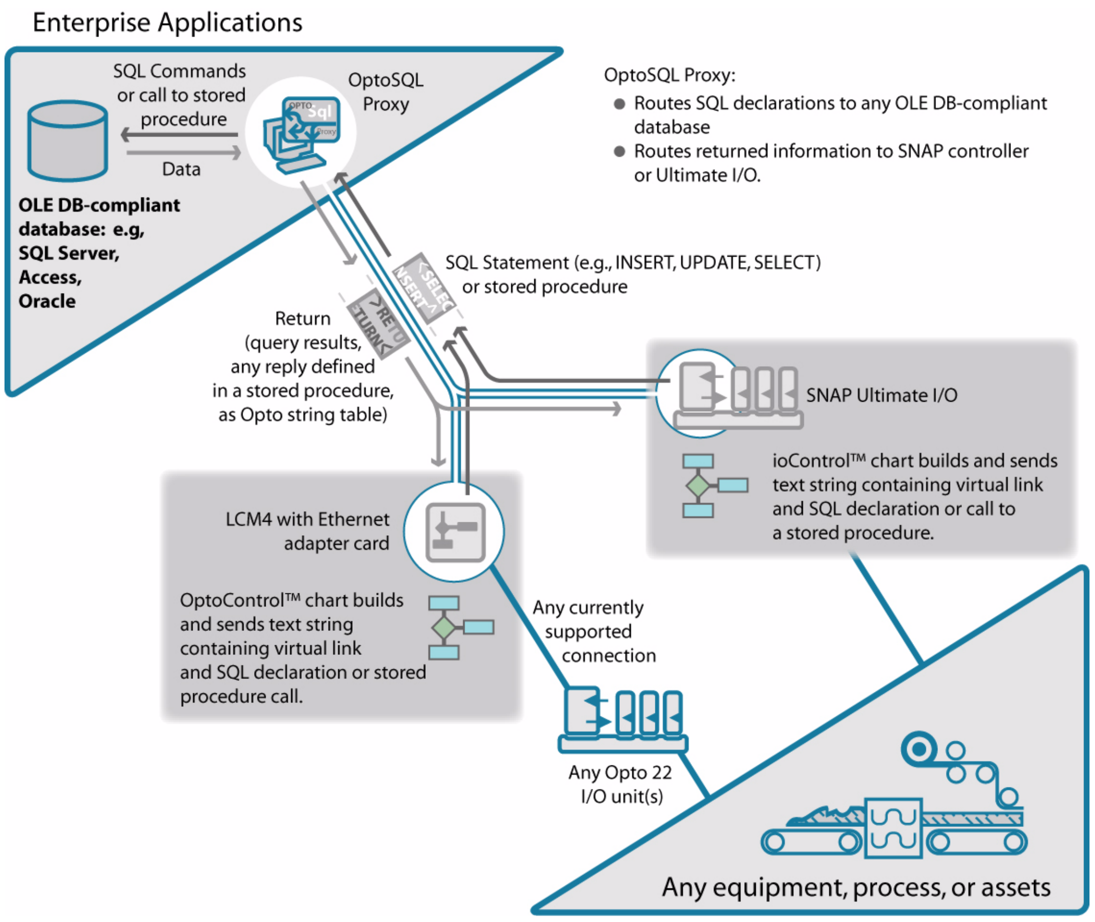
APPLICATION NOTE:
OPTOSQL™ PROXY (FORM 1428)



Enterprise Applications
SQL Commands or call to stored procedure
OptoSQL Proxy
Data
OLE DB-compliant database: e.g, SQL Server, Access, Oracle
OptoSQL Proxy:
Routes SQL declarations to any OLE DB-compliant database
Routes returned information to SNAP controller or Ultimate I/O.
SQL Statement (e.g., INSERT, UPDATE, SELECT) or stored procedure
Return (query results, any reply defined in a stored procedure, as Opto string table)
SNAP Ultimate I/O
ioControl™ chart builds and sends text string containing virtual link and SQL declaration or call to a stored procedure.
LCM4 with Ethernet adapter card
OptoControl™ chart builds and sends text string containing virtual link and SQL declaration or stored procedure call.
Any currently supported connection
Any Opto 22 I/O unit(s)
Any equipment, process, or assets

Contents

Application Note: OptoSQL Proxy
Form 1428-040311 (March 2004)

OptoSQL Proxy

Overview

OptoSQL Proxy is a free utility that allows our SNAP Ultimate I/O and SNAP-LCE controllers to send data to and request data from an OLE-DB data source. It was built to show how to send data to a Microsoft SQL Server but uses features supported by any OLE-DB data source. We also include the source code for customers who wish to adapt the program to their particular application. Hence, we do not support it as a product; rather we provide it as an example.

OptoSQL Proxy requires the .NET framework, and runs as a service on any suitably equipped workstation or server.

Components

OptoSQL Proxy—This is a service that runs on a workstation or server and relays TCP communications between Opto 22 Ethernet-based controllers and OLE DB data sources. This service requires the .NET framework.

Virtual Link Manager—This is the user interface to OptoSQL Proxy, allowing you to define, edit, and remove links to OLE DB data sources. The Virtual Link Manager runs on the same workstation where the OptoSQL Proxy service is installed.

OptoProxy Tester—This is a debugging tool that allows you to test the virtual links defined in OptoSQL proxy and to test commands to the OLE DB data source.

Sample ioControl strategies—Two sample strategies are available. Both strategies are configured for a SNAP Ultimate I/O Learning Center. ULC_DataToAccess.idb demonstrates how to read and write to an Access 2002 database, which is also included. ULC_SqlServer.idb demonstrates how to read and write to an SQL Server.

What OptoSQL Proxy Does

OptoSQL Proxy operates as a service on any networked, Windows-based workstation and relays communication initiated by Opto 22 Ethernet-based controllers (e.g., SNAP Ultimate I/O or SNAP-LCM4) to and from an OLE DB-compliant data source. Communication is event-based: the SNAP controller makes a connection to OptoSQL Proxy, sends information, and then disconnects. OptoSQL Proxy may reside on the same workstation as the OLE DB-compliant data source or on a different workstation. Using OptoSQL Proxy, a SNAP controller can send data to and request data from any OLE DB data source.

The examples in this application note use Microsoft SQL Server as the data source. As OptoSQL Proxy is a utility, source code is available for adapting OptoSQL Proxy to applications other than Microsoft SQL Server.

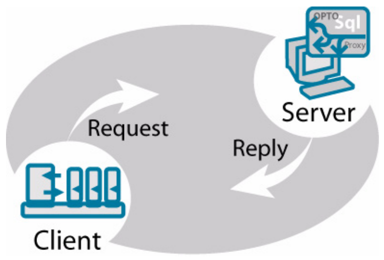
Disclaimer: Opto 22 offers the source code as an example of how to write a proxy service. All implementations must be tested according to the application developer's safety standards.

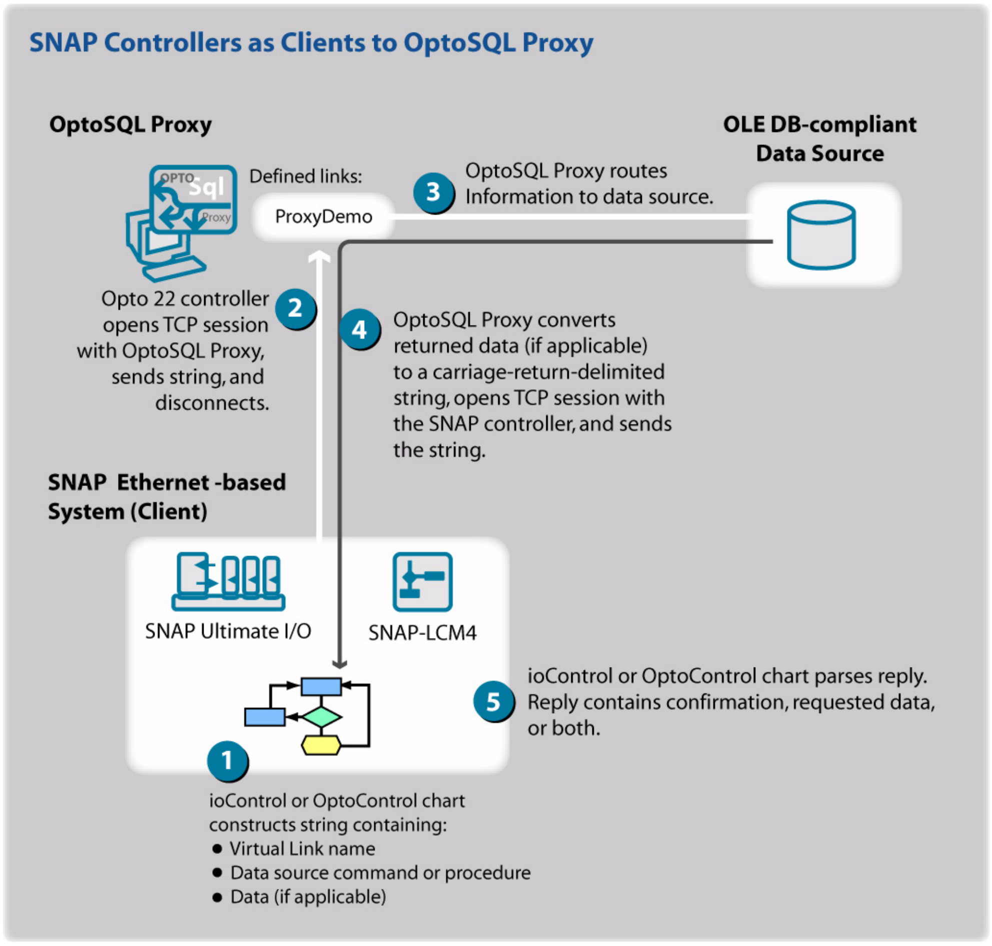
What is an OLE DB-Compliant Data Source?

An OLE DB-compliant data source is any application that supports Object Linking and Embedding (OLE) for data bases. Examples are Microsoft SQL Server, Oracle, and MSDE. This application note describes using OptoSQL proxy with Microsoft SQL Server.

SNAP Controllers are Clients to OptoSQL Proxy

Client and server, as used in this application note, are determined by which device initiates communication. The SNAP controller initiates communication with OptoSQL Proxy, so the controller is the client and OptoSQL Proxy is the server. This application note only describes applications in which the SNAP controller is the client.



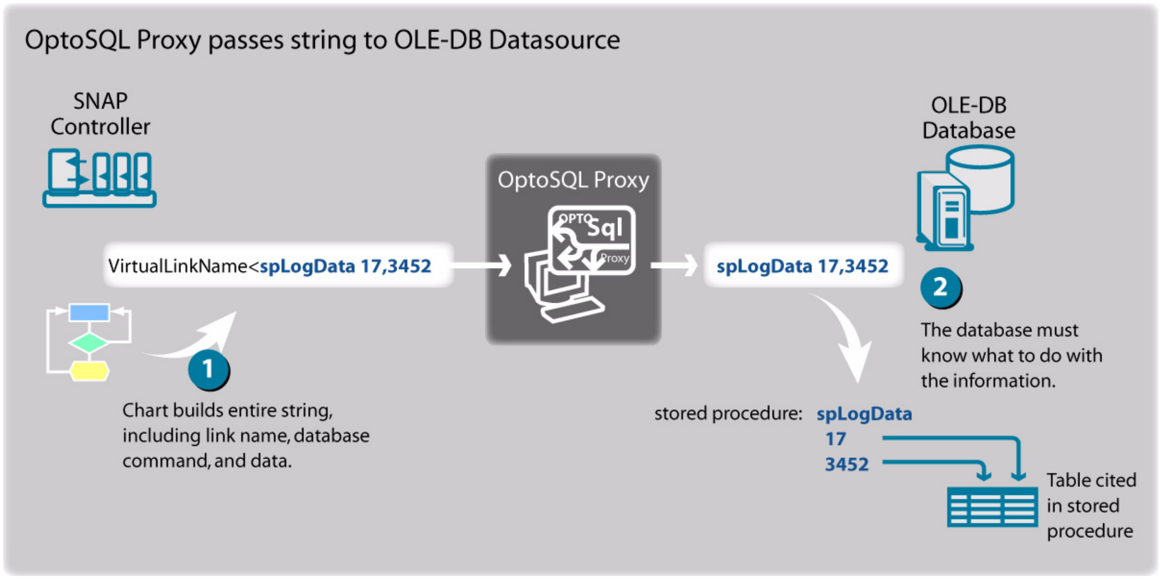

SNAP Controllers as Clients to OptoSQL Proxy
OptoSQL Proxy
OLE DB-compliant Data Source
Defined links:
ProxyDemo
3 OptoSQL Proxy routes Information to data source.
2 Opto 22 controller opens TCP session with OptoSQL Proxy, sends string, and disconnects.
4 OptoSQL Proxy converts returned data (if applicable) to a carriage-return-delimited string, opens TCP session with the SNAP controller, and sends the string.
SNAP Ethernet -based System (Client)
SNAP Ultimate I/O
SNAP-LCM4
5 ioControl or OptoControl chart parses reply. Reply contains confirmation, requested data, or both.
1 ioControl or OptoControl chart constructs string containing:
● Virtual Link name
● Data source command or procedure
● Data (if applicable)

OptoSQL Proxy Relays Strings

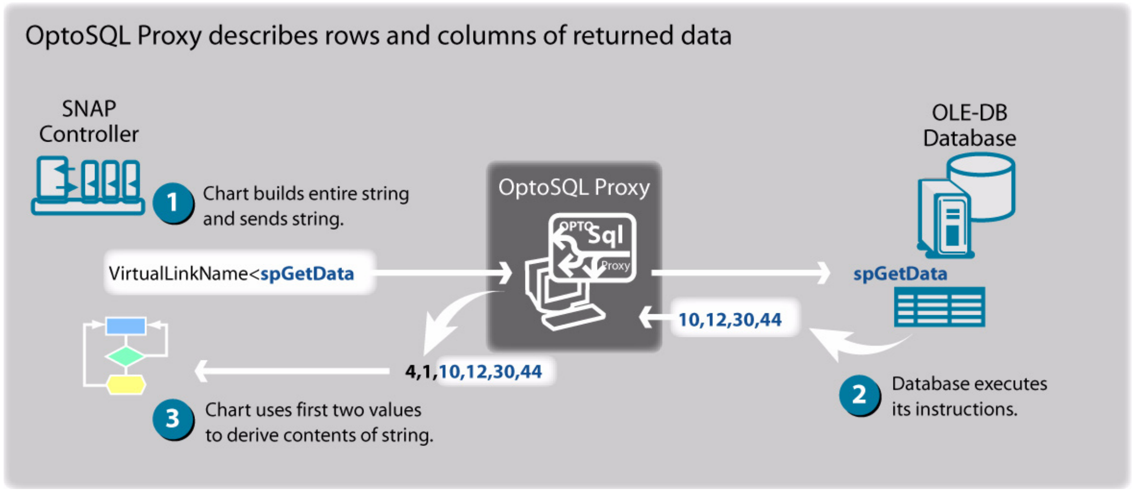
Your control strategy sends information in string format to the database and this information must be in a format that the database understands. For example, if your database supports the ANSI standard Structured Query Language (SQL), then you can design a control strategy that sends the following SQL statement to the database:

```
INSERT into DataLOG(Field1,Field2) values (123,233)
```

For this standard SQL statement to work in a database, there must be a table named *DataLOG* with columns named *Field1* and *Field2*. This is the case with all strings sent to OptoSQL proxy. The receiver, e.g., the database, must know what to do with the strings. The diagrams below and the sample ioControl strategies included with this application note send strings that call stored procedures in SQL Server. A stored procedure is a set of SQL commands stored in the database that can be executed by calling the stored procedure name.


OptoSQL Proxy passes string to OLE-DB Datasource
SNAP Controller
OptoSQL Proxy
OLE-DB Database
VirtualLinkName<spLogData 17,3452
spLogData 17,3452
1
Chart builds entire string, including link name, database command, and data.
2
The database must know what to do with the information.
stored procedure: spLogData
17
3452
Table cited in stored procedure

The diagram below shows data being returned from a database. When data is returned, OptoSQL Proxy gathers information on how many rows and columns of data are being returned and puts this information in the first two values of the string sent to the controller.


OptoSQL Proxy describes rows and columns of returned data
SNAP Controller
OLE-DB Database
1 Chart builds entire string and sends string.
OptoSQL Proxy
VirtualLinkName<spGetData
spGetData
10,12,30,44
4,1,10,12,30,44
2 Database executes its instructions.
3 Chart uses first two values to derive contents of string.

The control strategy must be explicitly programmed to write the returned data to a table variable and the table variable must be configured to support the number of returned values and the length of each. How to do this is explained in Strategy Design and demonstrated by the sample strategy.

Getting Started

Sample Files

Accompanying this application note are the following files:

Sample ioControl strategy for SNAP Ultimate I/O Learning Centers

- ULC_SqlServer.idb

Sample OptoControl strategy for SNAP LCM4 controllers

- LCM4_SqlServer

Installing and Running the OptoSQL Proxy

1. Run Setup.exe

Setup.exe installs the following:

- OptoSQL Proxy—Service that routes communications between SNAP controllers and vitual links defined in Opto Virtual Link Manager.
- Opto Virtual Link Manager—User interface for configuring OptoSQL Proxy and virtual links.
- Opto Proxy Tester—Utility for testing any virtual link defined in Opto Virtual Link Manager.

2. Start Opto Virtual Link Manager.

a. Run OptoVLM.exe:

Start → *Programs* → *Opto 22* → *OptoSQL Proxy* → *OptoVLM*

USING THE OPTOSQL PROXY VIRTUAL LINK MANAGER

What You Need

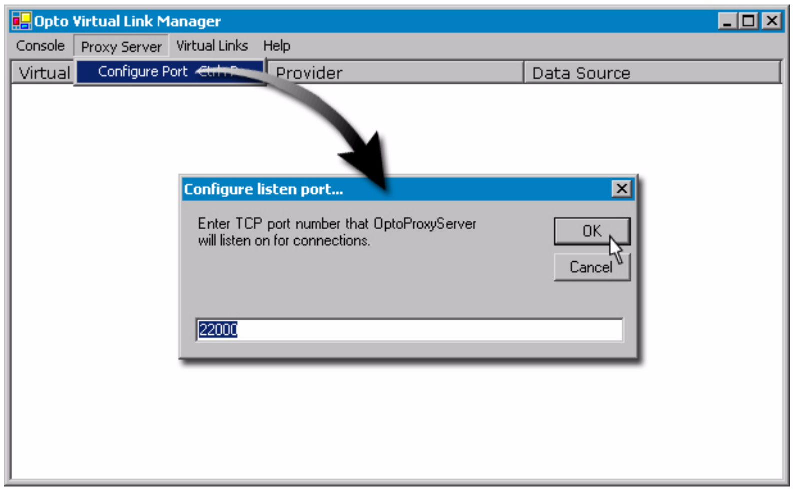
- You will need an open port on the workstation on which OptoSQL Proxy is listening. The default port is 22000.
- You will need an SQL Server (for this application note only). Each data source will have its own security settings, recognized users, and data infrastructure.

Configuring the OptoSQL Proxy Port

The OptoSQL Proxy listens at the default port 22000. You can easily change this port, and once changed, you will need to stop and restart the service before the change will take effect.

NOTE: When assigning a port, make sure you choose an unused port.

1. Choose Configure Port under the Proxy Server menu.



Enter TCP port number that OptoProxyServer will listen on for connections.

22000

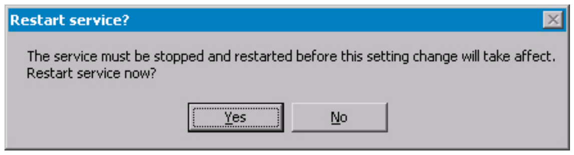
The currently configured port will appear in the Configure Listen Port dialog box.

2. Type a port number and click *OK*.

You will be prompted to restart the service.

3. Stop and restart the service.

- Choose *Yes* to restart the service.



The service must be stopped and restarted before this setting change will take affect.
Restart service now?

If you choose *No*, you will have to restart the service before the change in port takes effect.

Configuring a Virtual Link

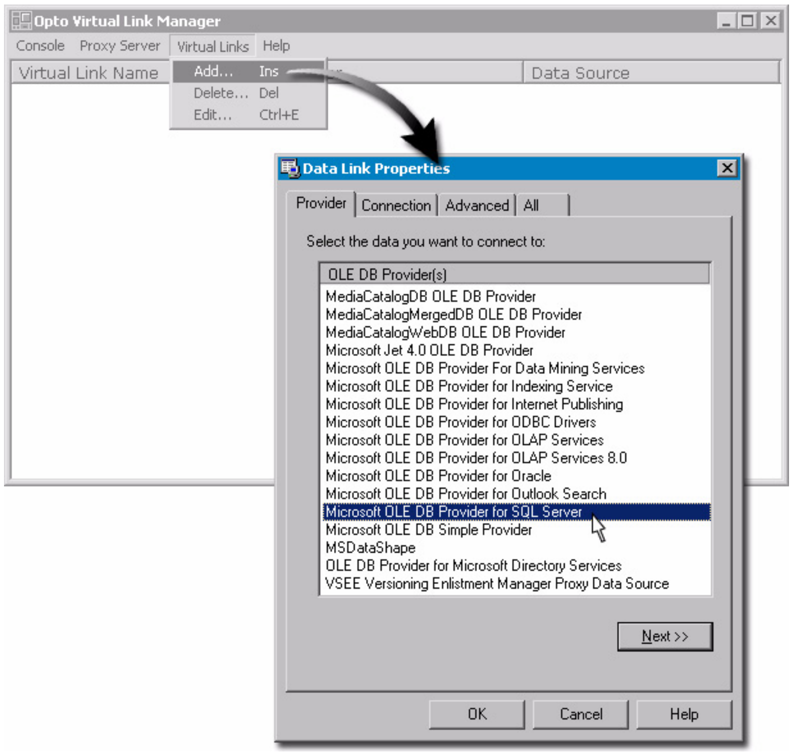
A virtual link is a name residing within the OptoSQL Proxy that is associated with an OLE DB-compliant data source. Any Opto 22 Ethernet-based contoller can send data to the OLE DB-compliant data source by referencing the virtual link.

Configuing the link requires the following:

- A virtual link name (This is an arbitrary name that you create and use within your OptoControl or ioControl code.)
- A data source type
- Paramaters specific to the data source type. For example, if you choose an SQL Data Source, the additional parameters will define the name of a specific database and security access.

1. Choose provider type.

a. Choose *Add* under the Virtual Links menu.



Select the data you want to connect to:

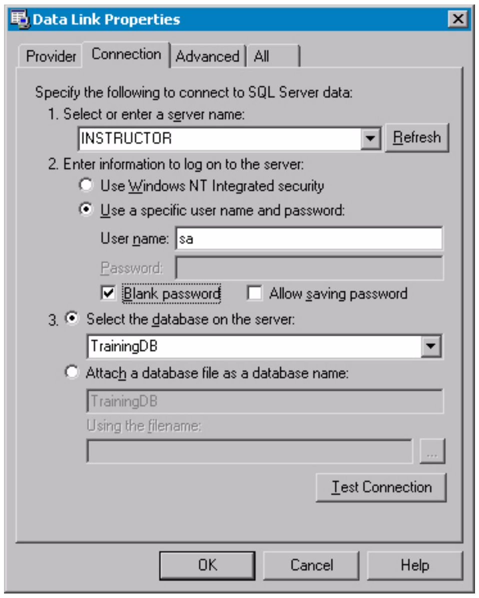
OLE DB Provider(s)
MediaCatalogDB OLE DB Provider
MediaCatalogMergedDB OLE DB Provider
MediaCatalogWebDB OLE DB Provider
Microsoft Jet 4.0 OLE DB Provider
Microsoft OLE DB Provider For Data Mining Services
Microsoft OLE DB Provider for Indexing Service
Microsoft OLE DB Provider for Internet Publishing
Microsoft OLE DB Provider for ODBC Drivers
Microsoft OLE DB Provider for OLAP Services
Microsoft OLE DB Provider for OLAP Services 8.0
Microsoft OLE DB Provider for Oracle
Microsoft OLE DB Provider for Outlook Search
Microsoft OLE DB Provider for SQL Server
Microsoft OLE DB Simple Provider
MSDataShape
OLE DB Provider for Microsoft Directory Services
VSEE Versioning Enlistment Manager Proxy Data Source

NOTE: At this point it is very important to choose the correct type of data source, as the remaining configuration options will vary according to the Provider selected here. This application note uses an OLE DB provider for SQL Server.

b. Select *Microsoft OLE DB Provider for SQL Server* and click *Next*.

2. Define the connection.

For this step, you will need to know the SQL Server and a method of access. Below is an example of how the information may appear using the MS SQL server in the Opto 22 Training Room.



Specify the following to connect to SQL Server data:
1. Select or enter a server name: INSTRUCTOR
2. Enter information to log on to the server:
( ) Use Windows NT Integrated security
(●) Use a specific user name and password:
User name: sa
Password:
[✓] Blank password [ ] Allow saving password
3. (●) Select the database on the server: TrainingDB
( ) Attach a database file as a database name: TrainingDB
Using the filename:

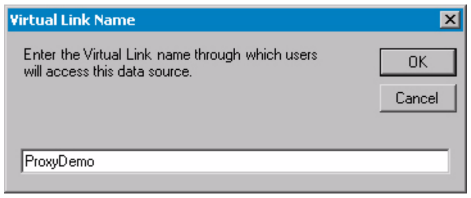
Make the following selections according to your SQL Server:

a. Select *Server* (all accessable servers will be listed under the Select Server dropdown list). If the server does not appear, type its name or IP address.
b. Provide Security log-on information. The SQL Server used in this example is set up to accept SA as a user name and a blank password.
c. Select the Database within the Server.

   The databases of the selected server are listed under the dropdown list entitled Select the Database on the Server.
d. Click *OK*.

3. Provide a proxy name.

a. Type a name of the link.



Enter the Virtual Link name through which users will access this data source.
ProxyDemo

   The samples provided with this application note use ProxyDemo as the link name.
b. Click *OK*.

At this point, the configuration of the Virtual link for an SQL Server is complete. Any connections to your data source can be made by using the link you've defined. You can define additional links, too.

4. Close the Virtual Link Manager.

OptoProxy Tester

About the Proxy Tester

The OptoProxy Tester can be run from any workstation on the network and be used to test the virtual link defined at any other workstation. The Proxy tester allows you to define the IP address and port of OptoSQL Proxy.

To use OptoProxyTester, you should have already configured links using the Virtual Link manager. (**NOTE:** Virtual link manager creates links on the computer where it was used.)

Using the OptoProxy Tester

1. Run OptoProxyTester.

a. Run OptoVLM.exe:

Start ⟶ *Programs* ⟶ *Opto 22* ⟶ *OptoSQL Proxy* ⟶ *OptoProxyTester*

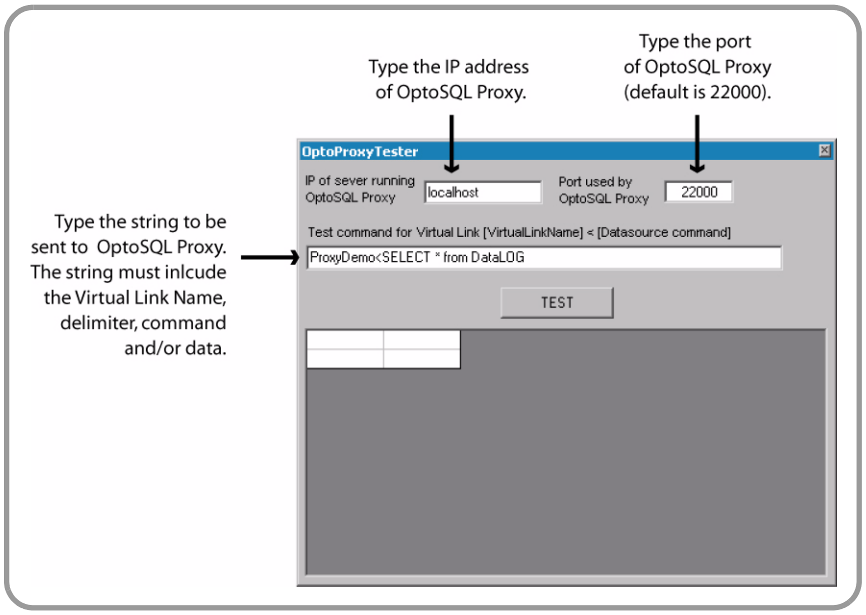
2. Assign IP address and port.

a. Type the IP Address of the computer running OptoSQL Proxy or leave the field as is if you defined the links on your workstation (localhost).
b. Type a port or leave as the default. (The default port is 22000.)

3. Type a string in the test command field.

The string you type must be supported by your data source. For example, if you are using an SQL database, you could type the following, substituting [TableName] with the name of a table from your database:

```
ProxyDemo<SELECT * from DataLOG
```


Type the IP address of OptoSQL Proxy.
Type the port of OptoSQL Proxy (default is 22000).
Type the string to be sent to OptoSQL Proxy. The string must inlcude the Virtual Link Name, delimiter, command and/or data.
OptoProxyTester
TEST

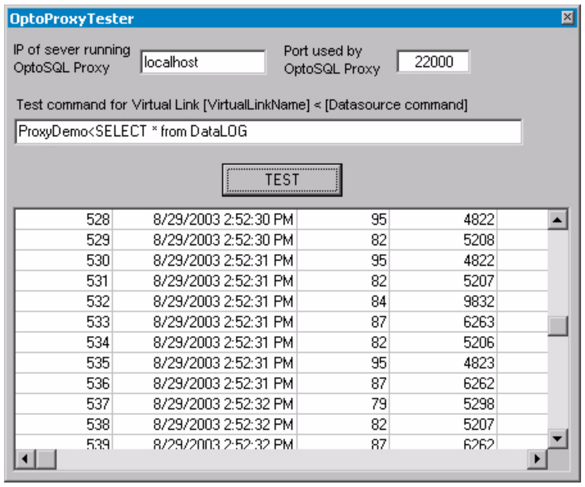
IP of sever running OptoSQL Proxy: localhost

Port used by OptoSQL Proxy: 22000

Test command for Virtual Link [VirtualLinkName] < [Datasource command]

ProxyDemo<SELECT * from DataLOG

4. **Click Test.**



IP of sever running OptoSQL Proxy: localhost

Port used by OptoSQL Proxy: 22000

Test command for Virtual Link [VirtualLinkName] < [Datasource command]

ProxyDemo<SELECT * from DataLOG

528	8/29/2003 2:52:30 PM	95	4822
529	8/29/2003 2:52:30 PM	82	5208
530	8/29/2003 2:52:31 PM	95	4822
531	8/29/2003 2:52:31 PM	82	5207
532	8/29/2003 2:52:31 PM	84	9832
533	8/29/2003 2:52:31 PM	87	6263
534	8/29/2003 2:52:31 PM	82	5206
535	8/29/2003 2:52:31 PM	95	4823
536	8/29/2003 2:52:31 PM	87	6262
537	8/29/2003 2:52:32 PM	79	5298
538	8/29/2003 2:52:32 PM	82	5207
539	8/29/2003 2:52:32 PM	87	6262

You can use OptoProxyTester to test any string that would be sent from your control strategy. A key difference is that OptoProxyTester will add an End-of-String character to the end of your string. The string you build in OptoControl or ioControl will have to include the End-of-String character. The sample strategies show how to do this.

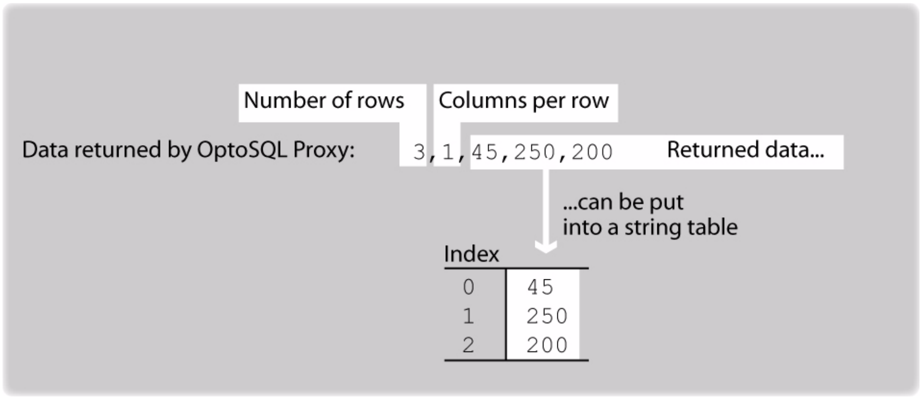
Strategy Design

Sample Strategy

Using standard ioControl or OptoControl commands, any strategy can communicate with an OLE-DB-compliant data source through OptoSQL Proxy. Sending data and requesting data are the same operation. In both cases, the strategy sends information to OptoSQL Proxy and receives a reply as a string table. The returned data contains at least two fields, and more if data is returned. In the first two fields, OptoSQL Proxy inserts integers desribing the number of rows and columns of the returned data. If these first fields contain 0 and 0, then no data follows. If, for example, three values are returned as data, the following carriage-return-delimited string may be returned:

```
3 1 45 250 200
```

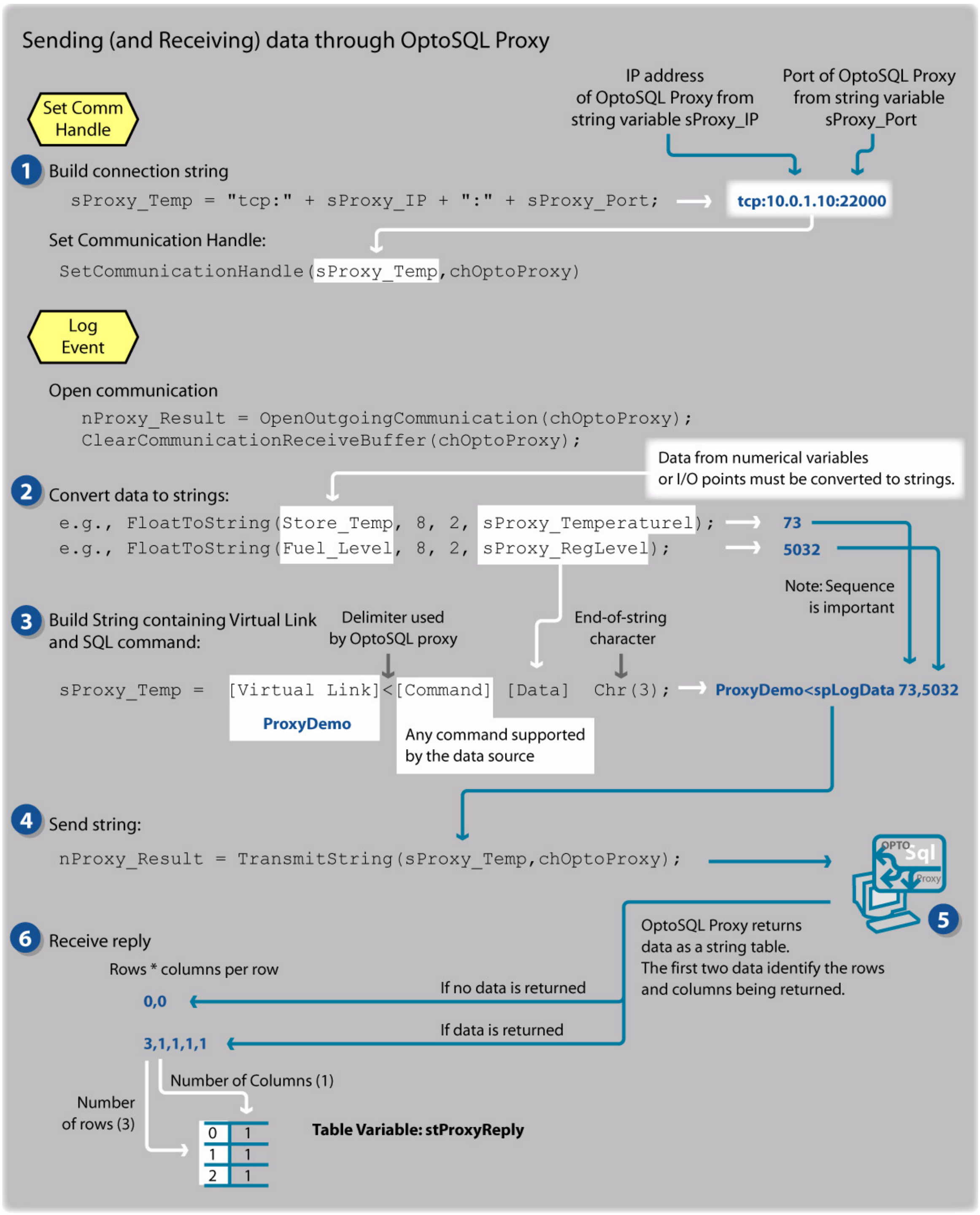
This string is interpreted as follows:



The following section examines the main components of the ioControl sample strategy built for the SNAP Ultimate I/O Learning Center. The same methods apply to OptoControl.

How the Sample Strategy works

The following diagram explains only the main parts of the sample chart OptoSQLProxy_Interface in order to show which components need to be customized for your application.


Sending (and Receiving) data through OptoSQL Proxy
Set Comm Handle
IP address of OptoSQL Proxy from string variable sProxy_IP
Port of OptoSQL Proxy from string variable sProxy_Port
1 Build connection string
tcp:10.0.1.10:22000
Set Communication Handle:
Log Event
Open communication
Data from numerical variables or I/O points must be converted to strings.
2 Convert data to strings:
73
5032
Note: Sequence is important
3 Build String containing Virtual Link and SQL command:
Delimiter used by OptoSQL proxy
End-of-string character
ProxyDemo<spLogData 73,5032
ProxyDemo
Any command supported by the data source
4 Send string:
5
OptoSQL Proxy returns data as a string table. The first two data identify the rows and columns being returned.
6 Receive reply
Rows * columns per row
If no data is returned
0,0
If data is returned
3,1,1,1,1
Number of Columns (1)
Number of rows (3)
Table Variable: stProxyReply

```
sProxy_Temp = "tcp:" + sProxy_IP + ":" + sProxy_Port;

SetCommunicationHandle(sProxy_Temp,chOptoProxy)

nProxy_Result = OpenOutgoingCommunication(chOptoProxy);
ClearCommunicationReceiveBuffer(chOptoProxy);

e.g., FloatToString(Store_Temp, 8, 2, sProxy_Temperature1);
e.g., FloatToString(Fuel_Level, 8, 2, sProxy_RegLevel);

sProxy_Temp = [Virtual Link]<[Command] [Data] Chr(3);

nProxy_Result = TransmitString(sProxy_Temp,chOptoProxy);
```

0	1
1	1
2	1

1 Before data can be sent to OptoSQL a Communication Handle must be defined. This example contains the IP Address and Port as separate string variables:

- sProxy_IP
- sProxy_Port

To use this code in your application, make sure these variables contain the IP address and port number of OptoSQL Proxy.

2 The strategy communicates via TCP, sending information in string format. Therefore, all data sent must be converted to strings. The sample strategy sends three values: Fuel Level (analog input), Store Temperature (analog input), and UioID (string variable). UioID is a string and doesn't need to be converted. Store Temperature and Fuel Level are converted to strings contained in the these variables:

- sProxy_RegLevel
- sProxy_StoreTemp

Your application must have a FloatToString or NumberToString command for each numerical value to be sent to the data source. Create a unique string variable to contain each converted value.

3 The string sProxy_Temp contains the data the strategy will send to OptoSQL Proxy. The following considerations apply:

- The string doesn't contain the IP and Port of OptoSQL Proxy as this formation is contained in the Communication Handle chOptoProxy.
- The virtual link must match a link defined in OptoSQL Proxy (using the Virtual Link Manager).
- A less-than symbol separates the link from the string to be relayed to the data source.
- All characters between the less-than and the end-of-string characters are passed to the data source. These characters may make up any command and data set recognized by the data source. The sample strategy shows how the string may appear using ANSI standard SQL commands or stored procedures (for Microsoft SQL Server).

4 The data string is sent by way of TCP, using the Communication Handle chOptoProxy. All data sent to the OptoSQL proxy can use the same Communication Handle. To send data to different OLE DB-compliant data sources, define additional virtual links within OptoSQL Proxy. If you have different versions of OptoSQL Proxy, each would require a unique Communication Handle. (A unique communication handle is needed only for each OptoSQL Proxy with an unique IP address.)

5 OptoSQL Proxy is a service that listens at a specified port , and parses messages for the virtual link name, command or procedure, and data. OptoSQL proxy must be running.

6 If the command or procedure returns data, OptoSQL proxy encodes the data as a carriage-return-delimited string. Use ioControl command *Receive String* with default end-of-message character to receive the reply (ioControl version 5.0 or higher). You can use command Receive String Table once the first two strings are received and the number of elements has been computed.

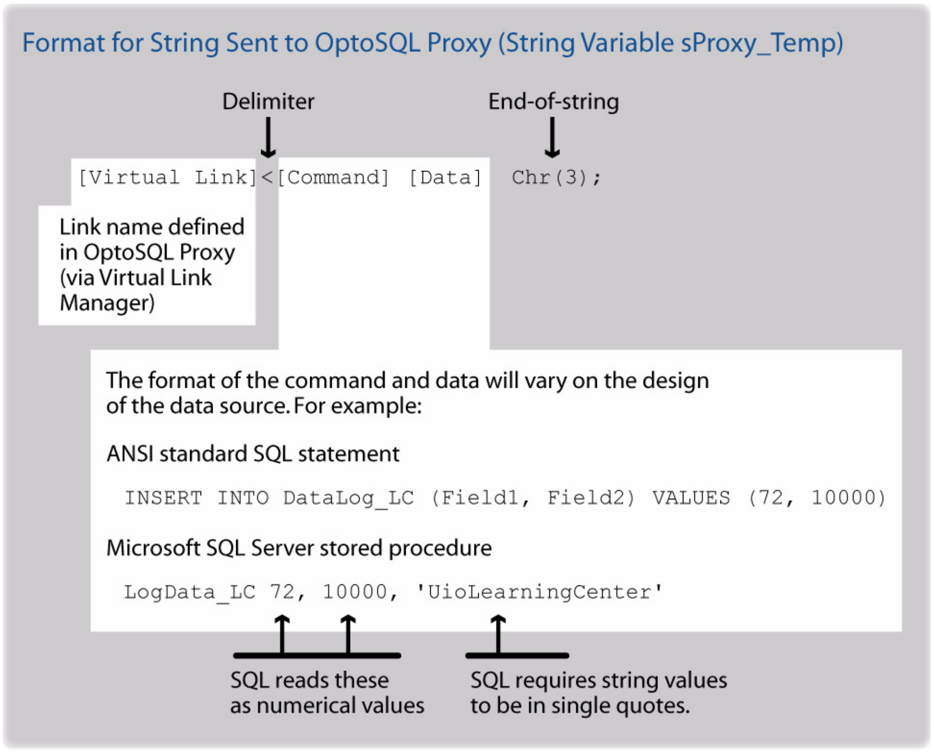
Preparing Data: Data-to-String Conversions

Some strategies for converting data.

ioControl Data Float example	Command	SQL
7.897848e-029	NumberToString FloatToString FloatToFormattedHexString	Float
Integer32	NumberToString	INT
Integer64		BIGINT
String		

Sending Data: ioControl Command String

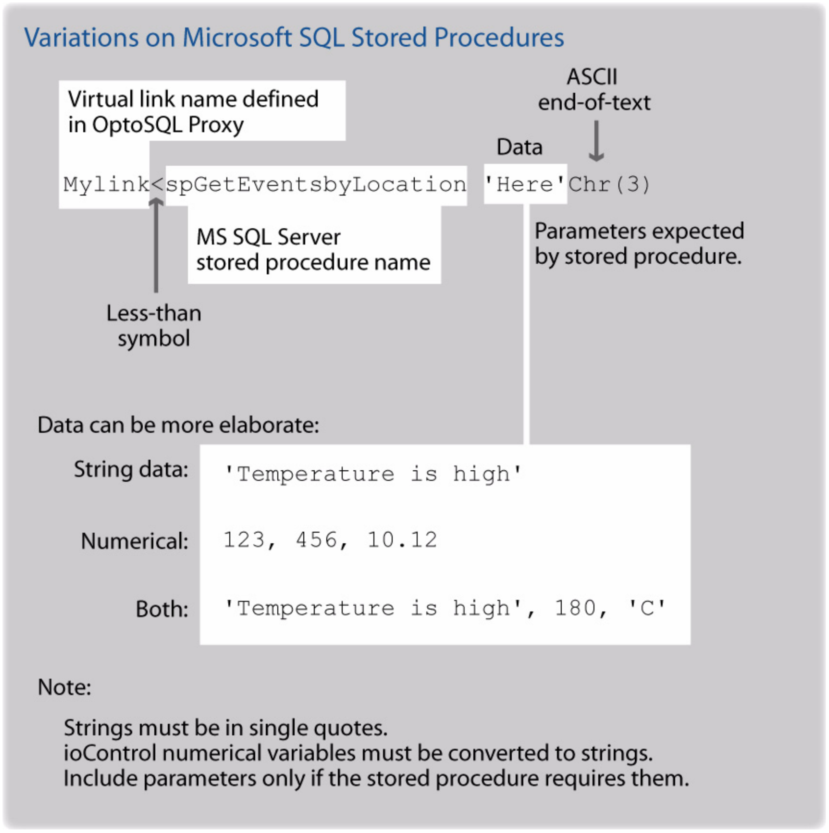
Any string of data can be sent to an OLE DB-compliant data source. The limit is based on the data source. The sample strategy puts the data into a variable sProxy_Temp. The diagrams below show possible content for this variable.


Format for String Sent to OptoSQL Proxy (String Variable sProxy_Temp)
Delimiter
End-of-string
[Virtual Link]<[Command] [Data] Chr(3);
Link name defined in OptoSQL Proxy (via Virtual Link Manager)
The format of the command and data will vary on the design of the data source. For example:
ANSI standard SQL statement
INSERT INTO DataLog_LC (Field1, Field2) VALUES (72, 10000)
Microsoft SQL Server stored procedure
LogData_LC 72, 10000, 'UioLearningCenter'
SQL reads these as numerical values
SQL requires string values to be in single quotes.

Microsoft SQL Server supports stored procedures, which allow the string contructed in your strategy to be more concise than an equivalent ANSI standard SQL statement. The sample files rely on stored procedures but you will find the equivalent SQL statement included in the sample code (commented out). For example, here are two ways to send the same information from the sample strategy:

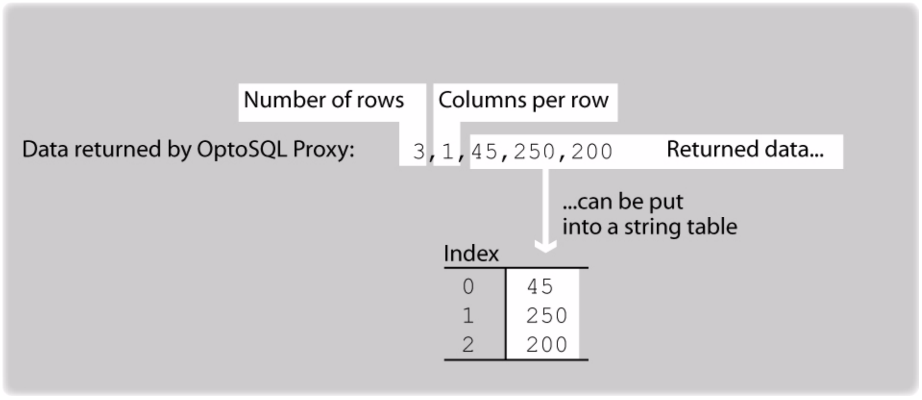
```
//  sProxy_Temp = "ProxyDemo<Insert DataLog (Temperature,FuelLevel,UioID)
values (" + sProxy_StoreTemp + ","+ sProxy_RegLevel +",'"+ UioID +"')" +
Chr(3);


// Use STored Procedure:
   sProxy_Temp = "ProxyDemo<spLogData" + sProxy_StoreTemp + ","+
sProxy_RegLevel +",'"+ UioID + "'"+Chr(3);
```

Here are some examples of how stored procedures would be formatted:

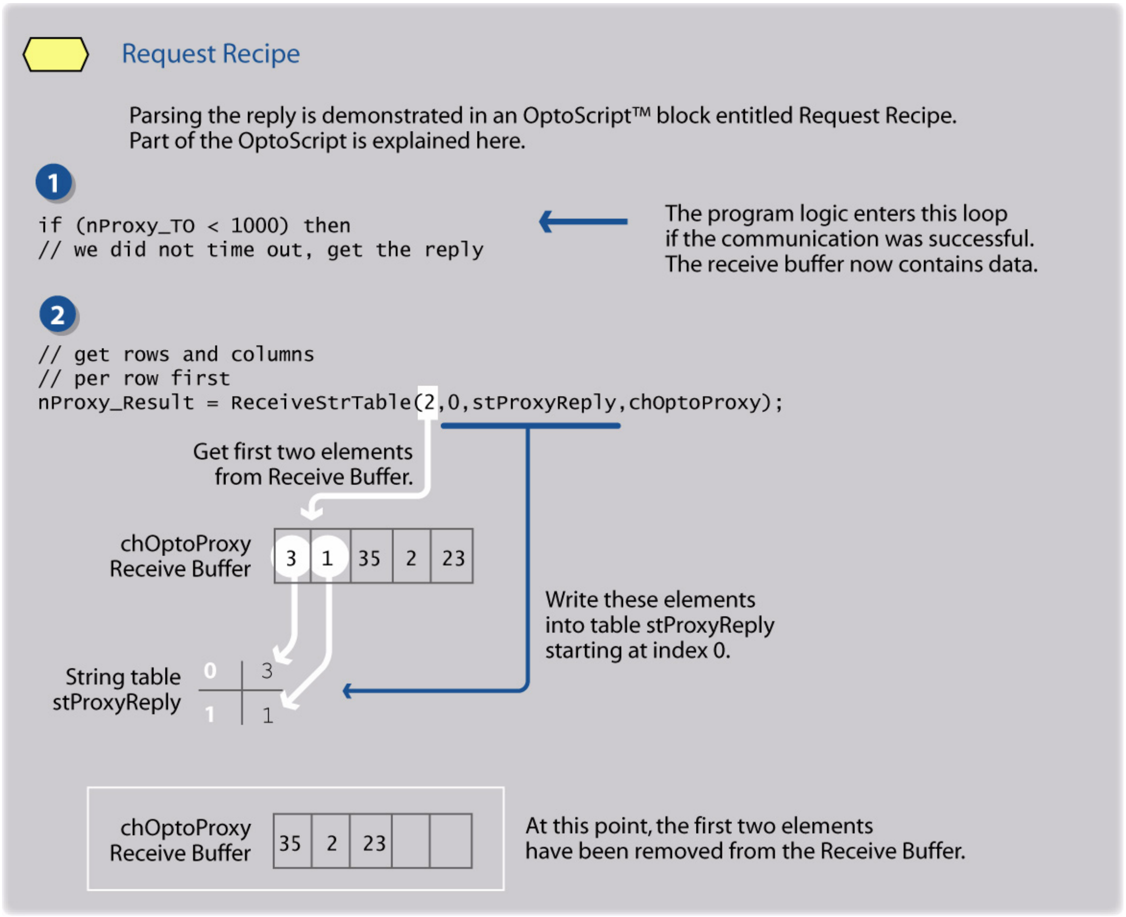

Variations on Microsoft SQL Stored Procedures
Virtual link name defined in OptoSQL Proxy
ASCII end-of-text
Data
Mylink<spGetEventsbyLocation 'Here'Chr(3)
MS SQL Server stored procedure name
Parameters expected by stored procedure.
Less-than symbol
Data can be more elaborate:
String data: 'Temperature is high'
Numerical: 123, 456, 10.12
Both: 'Temperature is high', 180, 'C'
Note:
Strings must be in single quotes.
ioControl numerical variables must be converted to strings.
Include parameters only if the stored procedure requires them.

RECEIVING DATA

If the communcation is successful, OptoSQL Proxy always replies with a carriage-return-delimited string. The first two elements of this string describe how many rows and columns are returned from the data source. If no data is returned, these first two elements are 0 and 0. If data is returned, these two elements contain information describing how to parse the data.

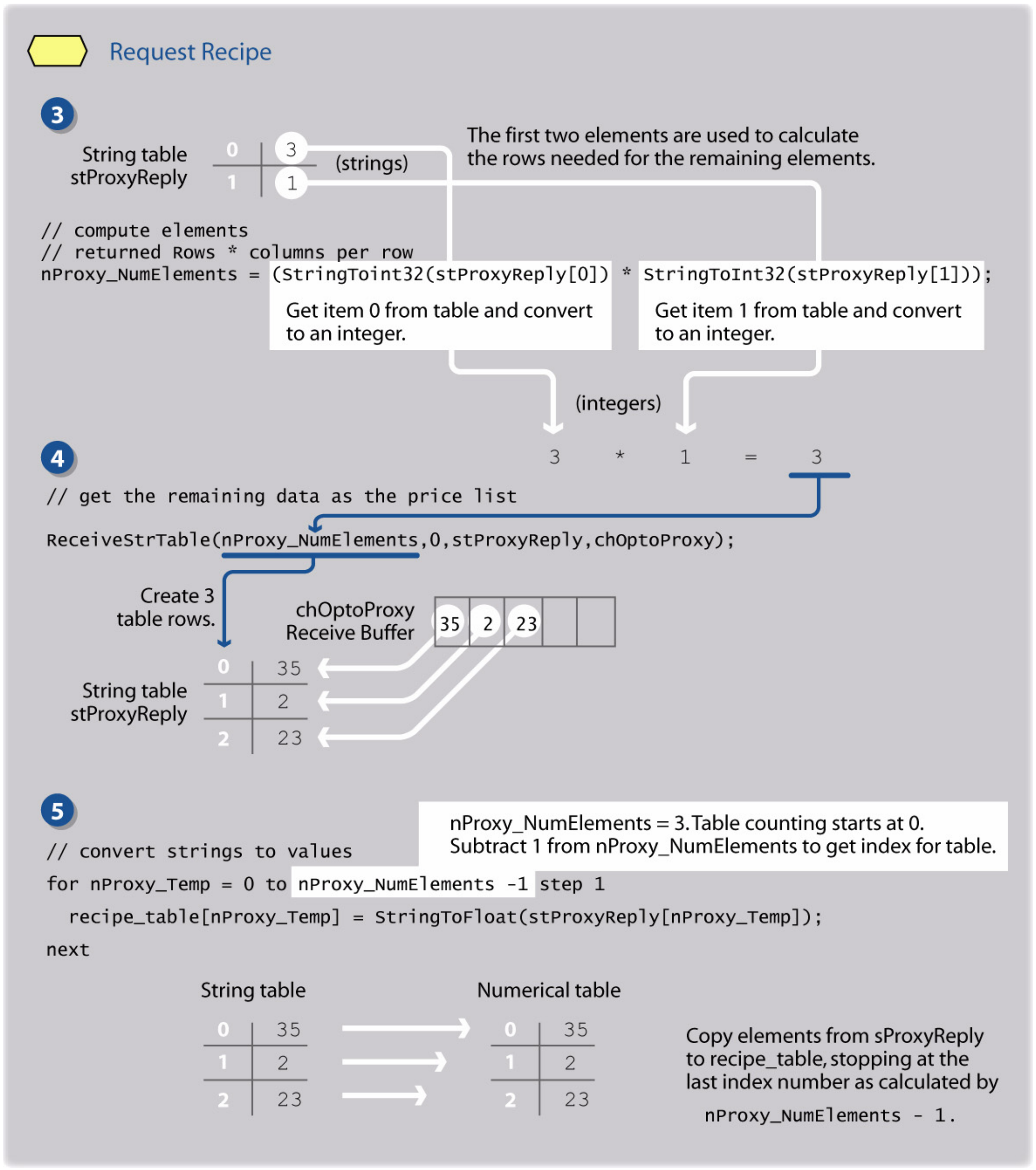


The following diagrams show how the sample strategy parses the returned data in the OptoScript block *Request Recipe*.


Request Recipe
Parsing the reply is demonstrated in an OptoScript™ block entitled Request Recipe. Part of the OptoScript is explained here.
1
2
The program logic enters this loop if the communication was successful. The receive buffer now contains data.
Get first two elements from Receive Buffer.
chOptoProxy Receive Buffer
3 1 35 2 23
Write these elements into table stProxyReply starting at index 0.
String table stProxyReply
0 3
1 1
chOptoProxy Receive Buffer
35 2 23
At this point, the first two elements have been removed from the Receive Buffer.

```
if (nProxy_TO < 1000) then
// we did not time out, get the reply
```

```
// get rows and columns
// per row first
nProxy_Result = ReceiveStrTable(2,0,stProxyReply,chOptoProxy);
```



The first two elements are used to calculate the rows needed for the remaining elements.

```
// compute elements
// returned Rows * columns per row
nProxy_NumElements = (StringToint32(stProxyReply[0]) * StringToInt32(stProxyReply[1]));
```

Get item 0 from table and convert to an integer.

Get item 1 from table and convert to an integer.

```
// get the remaining data as the price list

ReceiveStrTable(nProxy_NumElements,0,stProxyReply,chOptoProxy);
```

nProxy_NumElements = 3. Table counting starts at 0. Subtract 1 from nProxy_NumElements to get index for table.

```
// convert strings to values
for nProxy_Temp = 0 to nProxy_NumElements -1 step 1
  recipe_table[nProxy_Temp] = StringToFloat(stProxyReply[nProxy_Temp]);
next
```

Copy elements from sProxyReply to recipe_table, stopping at the last index number as calculated by `nProxy_NumElements - 1.`

The sample strategy shows one example of how to get the returned data. Specific details will vary according to your application.

REFERENCE

IOCONTROL SAMPLE: ULC_SQLSERVER

As OptoSQL Proxy connects SNAP controllers to OLE-DB-complient data sources, a demonstration and sample code of this capability is incomplete without a data source.

The following section describes the database used in the ioControl strategy ULC_SqlServer.idb for Ultimate I/O Learning Centers.

OptoSQL Proxy: Virtual Link to SQL Server

The demonstation uses a link called ProxyDemo. A different link name can be used as long as the sample ioControl chart is modified to use the new link name.

Tables

This information must be recreated on the SQL Server. Alternatively, the SQL statements sent by ioControl can be modified to match the structure of the database.

Datalog_LC

Column Name	Data Type	Length	Allow Nulls	Comment
LogID	int	4	no	Properties: Identy–yes; Identity Seed–1; Identity increment–1
Temperature	decimal	9	no	
FuelLevel	decimal	9	no	
UioID	text	16	no	

Recipe_LC

Column Name	Data Type	Length	Allow Nulls
OutsideLight	int	4	no
InsideLight	int	4	no
FreezerDoorLight	decimal	9	no

Stored Procedures

spLogData_LC

```
ALTER PROCEDURE dbo.spLogData_LC
      @st as float,
      @rl as float,
      @ui as text



 AS

      INSERT INTO DataLog_LC (Temperature,FuelLevel,UioID) VAL-
UES(@st,@rl,@ui)

      RETURN
```

spGetRecipe

```
ALTER PROCEDURE dbo.spGetRecipe

AS

      select * from Recipe
      return
```

SQL Insert Command

Form

```
INSERT [INTO] table_or_view [(column_list)] data_values
```

Examples

```
INSERT INTO DataLog_LC (Temperature, FuelLevel,UioID) VALUES (72,
10000, 'UioLearningCenter')
```

```
ProxyDemo<INSERT INTO DataLog_LC (Temperature, FuelLevel,UioID) VALUES
(72, 10000, 'UioLearningCenter')
```

SELECT COMMAND

Form

```
SELECT columns [INTO new_Table] FROM table [WHERE search_condition]
[GROUP BY expression] [HAVING search_condition] [ORDER BY expression
[ASC/DESC]]
```

Examples

```
SELECT * FROM DataLog WHERE LogID>'10' ORDER BY LogID ASC
SELECT logID,StoreTemp FROM DataLog ORDER BY LogID ASC
SELECT Settings FROM Recipe ORDER BY VariableID ASC
ProxyDemo<SELECT logID,StoreTemp FROM DataLog ORDER BY LogID ASC
ProxyDemo<SELECT *  FROM DataLog WHERE LogID>'27' ORDER BY LogID ASC
```