***Project to implement remote PHP web client/ MSSQL database server network setup***

*fortuneWalla  
August 16th 2015*

*Created on Friday, May 16th, 2014 at 2:43 pm*

For students learning data science/analytics, accessing data sets stored from databases in remote servers is a necessary skill. People in corporations/institutions have Systems Administrators to set up most of the interfaces for such purposes. However knowing how such a process is setup is a good knowledge. Here PHP is used to demonstrate database access.

Although this project is not related to data science/analytics, it will give a good hands-on experience of client/server networking. These are mere guidelines and not details as specific options & settings will depend on the version of the software & components.  
  
Pre-requisites: Fairly good understanding of computer hardware especially networking, Windows operating system, drivers & software installation.

**Hardware/Networking**

Physical: Atleast two computers with Ethernet/WiFi capability and a wireless/wired router.

Final Setup: Ability to create Workgroups & shared folders. Also ability to ping & telnet the remote computers/servers.

**Server Setup**

There are many versions & editions MSSQL depending on user requirements (The Developer Edition is best for learning & academic use). For data mining features, SQL Server Analysis Services(SSAS) must be installed. To install OLTP/DW AdventureWorks sample databases, enable FILESTREAM and make sure the SQL Full-text Filter Daemon Launcher service is running.

1) If you have the space, do a full install of every feature. It will make it easier to troubleshoot problems.  
2) Create a MSSQL user exclusively for remote access. <http://technet.microsoft.com/en-us/library/aa337545.aspx>  
3) Allow remote connection to MSSQL. <http://technet.microsoft.com/en-us/library/ms191464.aspx>  
4) Enable Shared Memory, TCP/IP, pipes <http://technet.microsoft.com/en-us/library/ms181035.aspx>  
5) In the Windows Firewall, allow incoming/outgoing connections for port 1433 (default MSSQL port.) <http://blogs.msdn.com/b/walzenbach/archive/2010/04/14/how-to-enable-remote-connections-in-sql-server-2008.aspx>

Final Setup: From the Command Prompt in the client, execute ping dbpcip and telnet dbpcip 1433 where dbpcip is the IP address of MSSQL machine. If you can connect with both the commands, the client/server networking is probably working. Although accessing the datebase & tables finally depends on the permissions assigned to the remote user.

**Client Setup**

Keep these details in mind before starting installation.  
1) Some of the types of PHP functions for access to MSSQL. This article only discusses mssql\_() and sqldrv\_().

* mssql\_() Supported by PHP 5.2 (php52) and lower. MSSQL Driver for PHP (SSDPHP) not required
* sqldrv\_() Supported by PHP 5.3 (php53) and higher. SSDPHP required.
* pdo\_sqlsrv\_() Supported by php52 and higher. SSDPHP required.
* pdo\_odbc\_() Supported by php51 and higher. SSDPHP not required.

2) SQL Server Native Client (SSNC): This is needed by PHP to access MSSQL Server.  
SSNC2008: installs on winxp  
SSNC2012: does not install on winxp.  
php52: requires atleast SSNC2008  
php53: requires atleast SSNC2012

3) Problem: php53 onwards requires SSNC2012, but SSNC2012 does not work on winxp. Also php53 does not support mssql\_() functions.

Solution: SSNC2008 works on winxp. Use php 5.2.13 to 5.2.17 as php52 requires SSNC2008. Also php 5.2.x supports mssql\_() functions and sqldrv\_() functions if you install the SSDPHP 2.0.

**Client software**

**1) Internet Information Services (IIS):**  
Official Documentation: <https://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/iiiisin2.mspx?mfr=true>

Step-by-step illustrated guide: Follow until step 10 <http://www.wikihow.com/Configure-IIS-for-Windows-XP-Pro>

You can test if it is working by opening the http://localhost/ address on the web browser. You should also test it on a remote computer using http://clientip/

Final Setup:  
Windows Version: XP  
IIS Version: 5.1

**2) FastCGI:**

FastCGI helps IIS work better with PHP. Install the one which is compatible with your version of IIS & OS.  
Official source: <http://www.iis.net/downloads/microsoft/fastcgi-for-iis>

Final Setup:fcgisetup\_1.5\_rtw\_x86.msi

**3) PHP installation:**

a) Steps to installing PHP with IIS are explained in [www.php.net/manual/en/install.windows.iis6.php](http://fortunewalla.com/2014/05/16/project-to-implement-remote-php-web-client-mssql-database-server-network-setup/www.php.net/manual/en/install.windows.iis6.php)  
b) 5.2 and lower mssql\_() documentation <http://www.php.net/manual/en/book.mssql.php>  
c) 5.3 and higher sqlsrv\_() documentation <http://www.php.net/manual/en/book.sqlsrv.php>  
c) 5.2 and higher pdo\_sqlsrv() documentation <http://www.php.net/manual/en/ref.pdo-sqlsrv.php>  
d) 5.1 and higher pdo\_odbc() documentation <http://www.php.net/manual/en/ref.pdo-odbc.php>

Final Setup: php-5.2.17-nts-Win32-VC6-x86.msi

**4) MSSQL Driver for PHP:**

Official Documentation: <http://technet.microsoft.com/en-us/library/dn425064%28v=sql.10%29.aspx>

Installation help: <http://www.iis.net/learn/application-frameworks/install-and-configure-php-on-iis/install-the-sql-server-driver-for-php>

Final Setup: SQLSRV20.exe

**5) SQL Server Native Client:**

Use this page to decide what components are needed for your setup. <http://msdn.microsoft.com/en-us/library/cc296170%28SQL.105%29.aspx>

Final Setup: sqlncli2k8r2x86.msi

**6) Overview:** The entire process can be very broadly summarised as:  
  
IIS -> FastCGI -> PHP 5.3 -> sqlsrv\_() -> SSDPHP 3.0 -> SSNC -> MSSQL  
IIS -> FastCGI -> PHP 5.2 -> sqlsrv\_() -> SSDPHP 2.0 -> SSNC -> MSSQL  
IIS -> FastCGI -> PHP 5.2 -> mssql\_() -> No SSDPHP -> SSNC -> MSSQL

**Testing the final setup:**

**1) PHP System Information Test:** If PHP is properly installed, the following code should execute:

**SOURCE CODE:**

<?php echo phpinfo(); ?>

You should get PHP system information which includes information about msql\_() and sqlsrv\_() extensions. Only relevant partial output shown below.

**OUTPUT:**

|  |  |
| --- | --- |
| Registered PHP Streams | php, file, data, http, ftp, compress.zlib, compress.bzip2, https, ftps, zip, **sqlsrv** |

**cgi-fcgi**

|  |  |  |
| --- | --- | --- |
| **Directive** | **Local Value** | **Master Value** |
| cgi.check\_shebang\_line | 1 | 1 |
| cgi.fix\_pathinfo | 1 | 1 |
| cgi.force\_redirect | 0 | 0 |
| cgi.nph | 0 | 0 |
| cgi.redirect\_status\_env | *no value* | *no value* |
| cgi.rfc2616\_headers | 0 | 0 |
| fastcgi.impersonate | 1 | 1 |
| fastcgi.logging | 0 | 0 |

**msql**

|  |  |
| --- | --- |
| MSQL Support | enabled |
| Allow Persistent Links | yes |
| Persistent Links | 0/unlimited |
| Total Links | 0/unlimited |

**mssql**

|  |  |
| --- | --- |
| **MSSQL Support** | **enabled** |
| Active Persistent Links | 0 |
| Active Links | 0 |
| Library version | 7.0 |

|  |  |  |
| --- | --- | --- |
| **Directive** | **Local Value** | **Master Value** |
| mssql.allow\_persistent | On | On |
| mssql.batchsize | 0 | 0 |
| mssql.compatability\_mode | Off | Off |
| mssql.connect\_timeout | 5 | 5 |
| mssql.datetimeconvert | On | On |
| mssql.max\_links | Unlimited | Unlimited |
| mssql.max\_persistent | Unlimited | Unlimited |
| mssql.max\_procs | Unlimited | Unlimited |
| mssql.min\_error\_severity | 10 | 10 |
| mssql.min\_message\_severity | 10 | 10 |
| mssql.secure\_connection | Off | Off |
| mssql.textlimit | Server default | Server default |
| mssql.textsize | Server default | Server default |
| mssql.timeout | 60 | 60 |

**pdo\_sqlsrv**

|  |  |
| --- | --- |
| **pdo\_sqlsrv support** | **enabled** |

|  |  |  |
| --- | --- | --- |
| **Directive** | **Local Value** | **Master Value** |
| pdo\_sqlsrv.log\_severity | 0 | 0 |

**sqlsrv**

|  |  |
| --- | --- |
| **sqlsrv support** | **enabled** |

|  |  |  |
| --- | --- | --- |
| **Directive** | **Local Value** | **Master Value** |
| sqlsrv.LogSeverity | 0 | 0 |
| sqlsrv.LogSubsystems | 0 | 0 |
| sqlsrv.WarningsReturnAsErrors | On | On |

**2) Remote MSSQL Access Test:** If phpinfo() indicates that all the drivers & extensions are installed properly, then test to see if the database server & client are properly registered.

**SOURCE CODE:**

<?php

$serverName = "dbpcip, 1433"; //serverName\instanceName, portNumber (default is 1433)

$connectionInfo = array( "Database"=>"master", "UID"=>"userName", "PWD"=>"passWord");

$conn = sqlsrv\_connect( $serverName, $connectionInfo);

if( $conn ) {

echo "Connection established.  
";

}else{

echo "Connection could not be established.  
";

die( print\_r( sqlsrv\_errors(), true));

}

if( $client\_info = sqlsrv\_client\_info( $conn)) {

foreach( $client\_info as $key => $value) {

echo $key.": ".$value."  
";

}

} else {

echo "Error in retrieving client info.  
";

}

$server\_info = sqlsrv\_server\_info( $conn);

if( $server\_info )

{

foreach( $server\_info as $key => $value) {

echo $key.": ".$value."  
";

}

} else {

die( print\_r( sqlsrv\_errors(), true));

}

?>

**OUTPUT:**

**Connection established.**

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| DriverDllName: | sqlncli10.dll |
| DriverODBCVer: | 03.52 |
| DriverVer: | 10.50.1600 |
| ExtensionVer: | 2.0.1802.200 |
| SQLServerVersion: | 10.50.1600 |
| SQLServerName: | DBPC |

**3) Test of SQL Query:** Once a connection is established, the final test is to see whether query execution is possible or not. This is a quick crude code. But the fact that it retrieves data from master.dbo.spt\_monitor table shows the setup & the connection works.

**SOURCE CODE:**

<?php

$serverName = "dbpcip, 1433"; //serverName\instanceName, portNumber (default is 1433)

$connectionInfo = array( "Database"=>"master", "UID"=>"userName", "PWD"=>"passWord");

$conn = sqlsrv\_connect( $serverName, $connectionInfo);

error\_reporting(-1);

if( $conn ) {

echo "Connection established.  
";

}else{

echo "Connection could not be established.  
";

die( print\_r( sqlsrv\_errors(), true));

}

/\* SQL Query \*/

$sql="select \* from dbo.spt\_monitor";

$results = sqlsrv\_query( $conn, $sql );

if( $results === false) {

die( print\_r( sqlsrv\_errors(), true) );

}

echo "MSSQL master.dbo.spt\_monitor TABLE  
";

echo "

<table border=1>

<tr>

<th>cpu\_busy</th>

<th>io\_busy</th>

<th>idle</th>

<th>pack\_received</th>

<th>pack\_sent</th>

<th>connections</th>

<th>pack\_errors</th>

<th>total\_read</th>

<th>total\_write</th>

<th>total\_errors</th>

</tr>";

while ($row = sqlsrv\_fetch\_array($results))

{

$cpu\_busy=$row[1];

$io\_busy=$row[2];

$idle=$row[3];

$pack\_received=$row[4];

$pack\_sent=$row[5];

$connections=$row[6];

$pack\_errors=$row[7];

$total\_read=$row[8];

$total\_write=$row[9];

$total\_errors=$row[10];

echo "

<tr>

<td>$cpu\_busy</td>;

<td>$io\_busy</td>;

<td>$idle</td>;

<td>$pack\_received</td>;

<td>$pack\_sent</td>;

<td>$connections</td>;

<td>$pack\_errors</td>;

<td>$total\_read</td>;

<td>$total\_write</td>;

<td>$total\_errors</td>;

</tr>";

}

echo "</table>";

?>

**OUTPUT:**

**Connection established.**

**MSSQL master.dbo.spt\_monitor TABLE**

; ; ; ; ; ; ; ; ; ;

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **cpu\_busy** | **io\_busy** | **idle** | **pack\_received** | **pack\_sent** | **connections** | **pack\_errors** | **total\_read** | **total\_write** | **total\_errors** |
| 9 | 7 | 792 | 28 | 28 | 14 | 0 | 0 | 0 | 0 |

**4) Incorrect SSDPHP version:**

Problem: If you use php53/SSDPHP 3.0 code but use SSNC2008 instead of SSNC 2012, you get an error message.

Could not connect. Array ( [0] => Array ( [0] => IMSSP [SQLSTATE] => IMSSP

[1] => -49 [code] => -49 [2] => This extension requires the Microsoft SQL Server 2012 Native Client.

Access the following URL to download the Microsoft SQL Server 2012 Native Client ODBC driver for x86: http://go.microsoft.com/fwlink/?LinkId=163712

[message] => This extension requires the Microsoft SQL Server 2012 Native Client.

Access the following URL to download the Microsoft SQL Server 2012 Native Client ODBC driver for x86: http://go.microsoft.com/fwlink/?LinkId=163712 )

[1] => Array ( [0] => IM002 [SQLSTATE] => IM002 [1] => 0 [code] => 0 [2] =>

[Microsoft][ODBC Driver Manager] Data source name not found and no default driver specified

[message] => [Microsoft][ODBC Driver Manager] Data source name not found and no default driver specified ) )

Solution: Install php52/SSDPHP 2.0 & SSNC2008

**Summary:**

Although this project takes a lot of time to setup, troubleshoot & tweak all the settings, the end result is a client/server setup one can experiment with. Once the database server is setup properly, various types of software can be configured to retrieve/store data. PHP can be used to create web applications for doing data analysis using MSSQL.

To access data from MSSQL using php52, mssql\_() functions is used. But php53 onwards uses only sqldrv\_() functions to access MSSQL. Hence it is better to learn sqldrv\_() functions. However knowing how to use the older functions will help when dealing with legacy systems having php52.