SBFspot V3.0 – Linux (Raspberry Pi) Quick Reference

Introduction

Unlike SBFspot 2.x, version 3.0 no longer uploads to PVoutput.org. This functionality is now in the hands of an upload service (Windows) or daemon (Linux)
SBFspot retrieves the data from the inverter and stores it in a SQLite or MySQL database
The service / daemon reads the data from the database and sends it to PVoutput.org.
This approach has a few advantages such as:

- Backlog upload (if pi crashes or internet connection is down, missing data will be sent to PVoutput.org when connection is restored.)
- If consumption data is available, it can also be sent to PVoutput.org

Which database?

Currently there are 2 databases supported: SQLite and MySQL.

SQLite is a simple system without a lot of bells and whistles (only 1 file) and is the preferred choice for use on a Raspberry Pi (SQLite is also used on Android smartphones and in popular software such as Firefox, Google Chrome...)

SQLite is a software library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. SQLite is the most widely deployed SQL database engine in the world. The source code for SQLite is in the public domain.

MySQL is a client/server system with a lot more possibilities (security, user management, …)

Many of the world's largest and fastest-growing organizations including Facebook, Google, Adobe, Alcatel Lucent and Zappos rely on MySQL to save time and money powering their high-volume Web sites, business-critical systems and packaged software.

Both systems are open source software and available for Windows and Linux.
On Raspberry Pi you can use both SQLite or MySQL, but I discourage to install MySQL (server) on Rpi (I know it's possible, but I didn't test it)

So, these are the options:

- SQLite database on Rpi
- MySQL database server on Windows/Linux system and MySQL client on Raspberri Pi

Backup

This installation procedure should not touch existing ones as we will use other directories, but it's always a good idea to have a backup. So, before proceeding with this SBFspot installation, make a copy of your .cfg file(s) and other files you want to keep.

SBFspot & SQLite

Installation

Install Bluetooth:
```
sudo apt-get --no-install-recommends install bluetooth libbluetooth-dev
```
To avoid installation of printer/scanner/other BT device drivers, the use of ***--no-install-recommends*** switch is needed (saves about 75Mb disk space)

Remark: even if you're using Speedwire, you need to install bluetooth stuff.

Install boost:
```
sudo apt-get install libboost-all-dev
```

Install SQLite:
```
sudo apt-get install sqlite3
sudo apt-get install libsqlite3-dev
```

Create directories:
```
mkdir /home/pi/sbfspot.3
mkdir /home/pi/smadata
mkdir /home/pi/smadata/logs
```

Download and copy sourcecode to /home/pi/sbfspot.3
From Linux host to RPi:
```
scp /home/pi/SBFspot_SRC_300_Linux_Win32.tar.gz pi@192.168.x.y: /home/pi/sbfspot.3
```

From Windows PC to RPi:
Get a copy of psftp.exe (http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html) and copy it to your %SystemRoot%\system32 folder.

```
C:\TEMP>psftp pi@192.168.x.y
Using username "pi".
pi@192.168.x.y's password:
Remote working directory is /home/pi
psftp> cd sbfspot.3
Remote directory is now /home/pi/sbfspot.3
psftp> put SBFspot_SRC_300_Linux_Win32.tar.gz
local:SBFspot_SRC_300_Linux_Win32.tar.gz =>
remote:/home/pi/sbfspot.3/SBFspot_SRC_300_Linux_Win32.tar.gz
psftp> exit
```

Untar the sourcecode:
```
cd /home/pi/sbfspot.3
tar -xvf SBFspot_SRC_300_Linux_Win32.tar.gz
```

Compile and install SBFspot with SQLite support in /usr/local/bin/sbfspot.3:
```
cd /home/pi/sbfspot.3/SBFspot
sudo make install_sqlite
```

When you see this warning, simply ignore it: *"Warning: swp{b} use is deprecated for this architecture"*

Don't worry too much about the SWP warnings, it's deprecated because it's expensive performance wise (especially on armv7 where it has to be implemented through an illegal instruction trap in the kernel) but it should still do the right thing.
See also http://www.raspberrypi.org/forums/viewtopic.php?t=63190&p=468543

Create an empty database

```
cd /home/pi/smadata
sqlite3 SBFspot.db < /home/pi/sbfspot.3/SBFspot/CreateSQLiteDB.sql
```

This should create /home/pi/smadata/SBFspot.db

You can also copy the pre-built SBFspot_Empty.db from `/home/pi/sbfspot.3/SBFspot` to /home/pi/smadata
Rename it to SBFspot.db

As a quick test, you can already execute a simple query:

```
sqlite3 SBFspot.db
SQLite version 3.7.13 2012-06-11 02:05:22
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> select * from config;
SchemaVersion|1
sqlite>.quit
```

Configuration

There are quite some changes made to the config file, so the best way is to start with a new one. Normally, there should be a default SBFspot.cfg in the install directory (/usr/local/bin/sbfspot.3)
References to PVoutput are removed and there are a few keys added to configure your SQL db

The default config file is well documented, but here are the most important changes:

- All settings for PVoutput removed: PVoutput_SID, PVoutput_Key, VoltLogging, InverterTemp, InverterTempMapTo and CumulativeEnergy
- Added SQL_Database (SQLite & MySQL)
- Added SQL_Hostname, SQL_Username and SQL_Password (MySQL only)

To edit the config file, execute the following command:

```
sudo nano /usr/local/bin/sbfspot.3/SBFspot.cfg
```

Don’t forget to set the Timezone (it’s on my TODO list to get it from /etc/timezone)

Testing

```
/usr/local/bin/sbfspot.3/SBFspot -v -finq -nocsv
```

Remark: don’t use -u (upload is removed)

Watch out for “Error: Can't open SQLite db [/home/pi/smadata/SBFspot.db] : unable to open database file” in the output.

Now we can check if there is something in the db (don’t forget the ‘;’ at the end of each query):

```
cd /home/pi/smadata
sqlite3 SBFspot.db
SQLite version 3.7.13 2012-06-11 02:05:22
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> select * from vwspotdata;
```

```
2014-08-26 10:32:08|2014-08-26 10:30:00|SB4000TL|SB 4000TL-
20|2100276197|127|56|0.37|0.339|345.3|166.73|178|0|0|0.769|0.0|0.0|232.98|0.0|0.0|183|1
78|0.0|332|14969473|49.96|15967.7|15502.0|64.7059|OK|Closed|36.84
sqlite> select * from inverters;
2100276197|SB4000TL|SB 4000TL-
20|03.01.05.R|1409041928|178|0.332|14969.5|15967.7|15502.0|OK|Closed|0.0
sqlite> .quit
```

Remark: make sure you are in the smadata directory (or provide the fullpath to the db) If not, you will create an empty db

If you're not too familiar with SQL syntax, there are plenty of examples and tutorials on the internet.

For automation and fine-tuning of SBFspot, I refer to the online documentation (https://sbfspot.codeplex.com/documentation)

UploadDaemon & SQLite

Installation

Install curl:
```
sudo apt-get install libcurl3-dev
```

Compile and install SBFspotUploadDaemon with SQLite support in /usr/local/bin/sbfspot.3:
```
cd /home/pi/sbfspot.3/SBFspotUploadDaemon
sudo make install_sqlite
```

Configuration

SBFspotUploadDaemon has its proper configuration file. All the settings are well documented.

The difference with previous versions of SBFspot is the multi SID feature. In a multi inverter plant, you can map each inverter to a PVoutput SID. The syntax is:
```
PVoutput_SID=SerialNo_1:SID_1,SerialNo_2:SID_2
```

To edit the config file, execute the following command:
```
sudo nano /usr/local/bin/sbfspot.3/SBFspotUpload.cfg
```

Testing

Start the daemon:
```
/usr/local/bin/sbfspot.3/SBFspotUploadDaemon
```

If all goes well, you should see a logfile in /home/smadata/logs:
```
[23:37:26] INFO: : Starting...
[23:37:57] INFO: : Uploading 30 datapoints, starting with 20140611,12:10,13775864,2256 => OK (200)
[23:39:36] INFO: : Uploading 30 datapoints, starting with 20140611,14:40,13782249,3432,,,62.94,234.95 => OK (200)
[23:41:14] INFO: : Uploading 30 datapoints, starting with 20140611,17:10,13788507,2736,,,60.94,236.69 => OK (200)
[23:42:51] INFO: : Uploading 25 datapoints, starting with 20140611,19:40,13793775,1704,,,54.31,237.53 => OK (200)
```

A few minutes later, your PVoutput graphs should be updated.

To see running processes:
```
ps aux
```

Stop daemon:
```
sudo killall SBFspotUploadDaemon
```

Autostart

```
sudo nano /etc/rc.local
```
Add these lines:
```
#Start SBFspotUploadDaemon
sudo /usr/local/bin/sbfspot.3/SBFspotUploadDaemon
```

Cleanup

When everything is running fine, you can remove the source folder:
`sudo rm -r /home/pi/sbfspot.3/`

Tweaking

By default, Inverter Temperature (V5) and Uac1 (V6) are uploaded to PVoutput. This can be relatively easy changed by modifying the view used for PVoutput data upload.

```
CREATE VIEW vwPvoData AS
       SELECT dd.Timestamp,
              dd.Name,
              dd.Type,
              dd.Serial,
              dd.TotalYield AS V1,
              dd.Power AS V2,
              cons.EnergyUsed AS V3,
              cons.PowerUsed AS V4,
              spot.Temperature AS V5,
              spot.Uac1 AS V6,
              NULL AS V7,
              NULL AS V8,
              NULL AS V9,
              NULL AS V10,
              NULL AS V11,
              NULL AS V12,
              dd.PVoutput
         FROM vwDayData AS dd
              LEFT JOIN vwAvgSpotData AS spot
                     ON dd.Serial = spot.Serial AND dd.Timestamp = spot.Nearest5min
              LEFT JOIN vwAvgConsumption AS cons
                     ON dd.Timestamp = cons.Nearest5min
        ORDER BY dd.Timestamp DESC;
```

Suppose we want ambient temperature (Weather Underground) instead of inverter temperature and we want DC voltage instead of AC. In the example below this is achieved by replacing the data for V5 and V6.

Spot.Temperature -> NULL
Spot.Uac1 -> Spot.Udc1

Since SQLite doesn't allow us to alter a view, we have to delete it first:

```
DROP VIEW IF EXISTS vwPvoData;
```

Then, recreate the view with our changes:

```
CREATE VIEW vwPvoData AS
       SELECT dd.Timestamp,
              dd.Name,
              dd.Type,
              dd.Serial,
              dd.TotalYield AS V1,
              dd.Power AS V2,
              cons.EnergyUsed AS V3,
              cons.PowerUsed AS V4,
              NULL AS V5,
              spot.Udc1 AS V6,
              NULL AS V7,
              NULL AS V8,
              NULL AS V9,
              NULL AS V10,
              NULL AS V11,
              NULL AS V12,
              dd.PVoutput
         FROM vwDayData AS dd
              LEFT JOIN vwAvgSpotData AS spot
                     ON dd.Serial = spot.Serial AND dd.Timestamp = spot.Nearest5min
              LEFT JOIN vwAvgConsumption AS cons
                     ON dd.Timestamp = cons.Nearest5min
```

```
        ORDER BY dd.Timestamp DESC;
```

Remark: You can only change the V5..V12 parameters. V1..V4 are fixed. Also, make sure all V1..V12 are present.

For compatibility with previous SBFspot versions, V6 can be changed as follows:

VoltLogging in config file	**Replacement in vwPvoData**
NONE (disabled)	NULL AS V6
MAX(AC)	MAX(spot.Uac1,spot.Uac2,spot.Uac3) AS V6
AC(PH1)	Uac1 AS V6
MAX(DC)	MAX(spot.Udc1,spot.Udc2) AS V6
DC(ST1)	Udc1 AS V6

When in PVoutput donation mode, you can make use of V7..V12
The next example shows the query to upload Udc1(V7), Udc2(V8), Pdc1(V9) and Pdc2(V10)

```
CREATE VIEW vwPvoData AS
       SELECT dd.Timestamp,
              dd.Name,
              dd.Type,
              dd.Serial,
              dd.TotalYield AS V1,
              dd.Power AS V2,
              cons.EnergyUsed AS V3,
              cons.PowerUsed AS V4,
              spot.Temperature AS V5,
              spot.Uac1 AS V6,
              spot.Udc1 AS V7,
              spot.Udc2 AS V8,
              spot.Pdc1 AS V9,
              spot.Pdc2 AS V10,
              NULL AS V11,
              NULL AS V12,
              dd.PVoutput
         FROM vwDayData AS dd
              LEFT JOIN vwAvgSpotData AS spot
                     ON dd.Serial = spot.Serial AND dd.Timestamp = spot.Nearest5min
              LEFT JOIN vwAvgConsumption AS cons
                     ON dd.Timestamp = cons.Nearest5min
        ORDER BY dd.Timestamp DESC;
```

With a bit more experience you can include gas/water usage and let SBFspotUploader do the upload. The consumption table can be used as a starting point.

SBFspot & MySQL Client

Installation

Install Bluetooth:
```
sudo apt-get --no-install-recommends install bluetooth libbluetooth-dev
```
To avoid installation of printer/scanner/other BT device drivers, the use of ***--no-install-recommends*** switch is needed (saves about 75Mb disk space)

Remark: even if you're using Speedwire, you need to install bluetooth stuff.

Install boost:
```
sudo apt-get install libboost-all-dev
```

Install MySQL Client:
```
sudo apt-get install libmysqlclient-dev
```

Create directories:
```
mkdir /home/pi/sbfspot.3
mkdir /home/pi/smadata
mkdir /home/pi/smadata/logs
```

Download and copy sourcecode to /home/pi/sbfspot.3
From Linux host to RPi:
```
scp /home/pi/SBFspot_SRC_300_Linux_Win32.tar.gz pi@192.168.x.y: /home/pi/sbfspot.3
```

From Windows PC to RPi:
Get a copy of psftp.exe (http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html) and copy it to your %SystemRoot%\system32 folder.

```
C:\TEMP>psftp pi@192.168.x.y
Using username "pi".
pi@192.168.x.y's password:
Remote working directory is /home/pi
psftp> cd sbfspot.3
Remote directory is now /home/pi/sbfspot.3
psftp> put SBFspot_SRC_300_Linux_Win32.tar.gz
local:SBFspot_SRC_300_Linux_Win32.tar.gz =>
remote:/home/pi/sbfspot.3/SBFspot_SRC_300_Linux_Win32.tar.gz
psftp> exit
```

Untar the sourcecode:
```
cd /home/pi/sbfspot.3
tar -xvf SBFspot_SRC_300_Linux_Win32.tar.gz
```

Compile and install SBFspot with SQLite support in /usr/local/bin/sbfspot.3:
```
cd /home/pi/sbfspot.3/SBFspot
sudo make install_sqlite
```

When you see this warning, simply ignore it: *"Warning: swp{b} use is deprecated for this architecture"*
Don't worry too much about the SWP warnings, it's deprecated because it's expensive performance wise (especially on armv7 where it has to be implemented through an illegal instruction trap in the kernel) but it should still do the right thing.

See also http://www.raspberrypi.org/forums/viewtopic.php?t=63190&p=468543

Create an empty database

Like already said in the introduction, it is not recommended to install MySQL server on your Raspberry Pi.
Oh, you already did? No problem, you can proceed, but I didn't test it.
Anyway, I provided 2 scripts: 1 to create an empty database and another to create a user/password
Unfortunately, I can only give an example for MySQL Server installed on a Windows Server:

Create the database (assuming MySQL 5.6 is installed in C:\Program Files\MySQL\MySQL Server 5.6

```
"C:\Program Files\MySQL\MySQL Server 5.6\bin\mysql.exe" "--defaults-
file=C:\ProgramData\MySQL\MySQL Server 5.6\my.ini" "-uroot" "-p" <CreateMySQLDB.sql
```

Create the user/password

```
"C:\Program Files\MySQL\MySQL Server 5.6\bin\mysql.exe" "--defaults-
file=C:\ProgramData\MySQL\MySQL Server 5.6\my.ini" "-uroot" "-p" <CreateMySQLUser.sql
```

As a quick test, you can already logon and execute a simple query (on your RPi):

```
mysql --host=192.168.1.xx --user=SBFspotUser --password=SBFspotPassword
```

When you get this error, you can fix this by setting up an account for the combination of client host name and user name that you are using when trying to connect:

```
ERROR 1130 (HY000): Host 'raspberrypi.lan' is not allowed to connect to this MySQL
server
```

Logon as root and create a user SBFspotUser@'raspberrypi.lan'

```
CREATE USER 'SBFspotUser'@'raspberrypi.lan' IDENTIFIED BY 'SBFspotPassword';
GRANT DELETE,INSERT,SELECT,UPDATE ON SBFspot.* TO 'SBFspotUser'@'raspberrypi.lan';
FLUSH PRIVILEGES;
```

Now let's try again:

```
mysql --host=192.168.1.xx --user=SBFspotUser --password=SBFspotPassword
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 13
Server version: 5.6.13 MySQL Community Server (GPL)

Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> select * from sbfspot.config;
+---------------+-------+
| Key           | Value |
+---------------+-------+
| SchemaVersion | 1     |
+---------------+-------+
1 row in set (0.00 sec)
```

Configuration

There are quite some changes made to the config file, so the best way is to start with a new one. Normally, there should be a default SBFspot.cfg in the install directory (/usr/local/bin/sbfspot.3)
References to PVoutput are removed and there are a few keys added to configure your SQL db

The default config file is well documented, but here are the most important changes:

- All settings for PVoutput removed: PVoutput_SID, PVoutput_Key, VoltLogging, InverterTemp, InverterTempMapTo and CumulativeEnergy
- Added SQL_Database (SQLite & MySQL)
- Added SQL_Hostname, SQL_Username and SQL_Password (MySQL only)

To edit the config file, execute the following command:

```
sudo nano /usr/local/bin/sbfspot.3/SBFspot.cfg
```

```
SQL_Database=SBFspot
SQL_Hostname=SBF-PC (IP-Address or MySQL Computer Name)
SQL_Username=SBFspotUser
SQL_Password=SBFspotPassword
```

Don't forget to set the Timezone (it's on my TODO list to get it from /etc/timezone)

Testing

```
/usr/local/bin/sbfspot.3/SBFspot -v -finq -nocsv
```

Remark: don't use -u (upload is removed)

Watch out for "Error: Can't open MySQL db [SBFspot] : Unknown MySQL server host 'SBFPC'" in the output.

Now we can check if there is something in the db (don't forget the ';' at the end of each query):

```
mysql> use sbfspot;
Database changed
mysql> select * from inverters;
+------------+----------------------+--------------+------------+------------+----------+--------+--------+-
-------------+------------+--------+----------+-------------+
| Serial     | Name                 | Type         | SW_Version | TimeStamp  | TotalPac | EToday | ETotal |
OperatingTime | FeedInTime | Status | GridRelay | Temperature |
+------------+----------------------+--------------+------------+------------+----------+--------+--------+-
-------------+------------+--------+----------+-------------+
| 2100276197 | SB4000TL - 2100276197 | SB 4000TL-20 | 03.01.05.R | 1409173895 |        0 |     16 |  14988 |
15991.1 |    15524.7 | OK     | N/A      |           0 |
+------------+----------------------+--------------+------------+------------+----------+--------+--------+-
-------------+------------+--------+----------+-------------+
1 row in set (0.00 sec)
```

The output looks a bit sloppy, but there are better tools like MySQL Workbench

If you're not too familiar with SQL syntax, there are plenty of examples and tutorials on the internet.

For automation and fine-tuning of SBFspot, I refer to the online documentation (https://sbfspot.codeplex.com/documentation)

UploadDaemon & MySQL

Installation

Install curl:
```
sudo apt-get install libcurl3-dev
```

Compile and install SBFspotUploadDaemon with MySQL support in /usr/local/bin/sbfspot.3:
```
cd /home/pi/sbfspot.3/SBFspotUploadDaemon
sudo make install_mysql
```

Configuration

SBFspotUploadDaemon has its proper configuration file. All the settings are well documented.

The difference with previous versions of SBFspot is the multi SID feature. In a multi inverter plant, you can map each inverter to a PVoutput SID. The syntax is:
```
PVoutput_SID=SerialNo_1:SID_1,SerialNo_2:SID_2
```

To edit the config file, execute the following command:
```
sudo nano /usr/local/bin/sbfspot.3/SBFspotUpload.cfg
```

Testing

Start the daemon:
```
/usr/local/bin/sbfspot.3/SBFspotUploadDaemon
```

If all goes well, you should see a logfile in /home/smadata/logs:
```
[23:37:26] INFO: : Starting...
[23:37:57] INFO: : Uploading 30 datapoints, starting with 20140611,12:10,13775864,2256 => OK (200)
[23:39:36] INFO: : Uploading 30 datapoints, starting with 20140611,14:40,13782249,3432,,,62.94,234.95 => OK (200)
[23:41:14] INFO: : Uploading 30 datapoints, starting with 20140611,17:10,13788507,2736,,,60.94,236.69 => OK (200)
[23:42:51] INFO: : Uploading 25 datapoints, starting with 20140611,19:40,13793775,1704,,,54.31,237.53 => OK (200)
```

A few minutes later, your PVoutput graphs should be updated.

To see running processes:
```
ps aux
```

Stop daemon:
```
sudo killall SBFspotUploadDaemon
```

Autostart

```
sudo nano /etc/rc.local
```
Add these lines:
```
#Start SBFspotUploadDaemon
sudo /usr/local/bin/sbfspot.3/SBFspotUploadDaemon
```

Cleanup

When everything is running fine, you can remove the source folder:
`sudo rm -r /home/pi/sbfspot.3/`

Tweaking

By default, Inverter Temperature (V5) and Uac1 (V6) are uploaded to PVoutput. This can be relatively easy changed by modifying the view used for PVoutput data upload.

```
CREATE VIEW vwPvoData AS
       SELECT dd.Timestamp,
              dd.Name,
              dd.Type,
              dd.Serial,
              dd.TotalYield AS V1,
              dd.Power AS V2,
              cons.EnergyUsed AS V3,
              cons.PowerUsed AS V4,
              spot.Temperature AS V5,
              spot.Uac1 AS V6,
              NULL AS V7,
              NULL AS V8,
              NULL AS V9,
              NULL AS V10,
              NULL AS V11,
              NULL AS V12,
              dd.PVoutput
         FROM vwDayData AS dd
              LEFT JOIN vwAvgSpotData AS spot
                     ON dd.Serial = spot.Serial AND dd.Timestamp = spot.Nearest5min
              LEFT JOIN vwAvgConsumption AS cons
                     ON dd.Timestamp = cons.Nearest5min
        ORDER BY dd.Timestamp DESC;
```

Suppose we want ambient temperature (Weather Underground) instead of inverter temperature and we want DC voltage instead of AC. In the example below this is achieved by replacing the data for V5 and V6.

Spot.Temperature -> NULL
Spot.Uac1 -> Spot.Udc1

MySQL makes it possible to alter a view, but for compatibility with SQLite, we delete it first:

```
USE SBFspot;
DROP VIEW IF EXISTS vwPvoData;
```

Then, recreate the view with our changes:

```
CREATE VIEW vwPvoData AS
       SELECT dd.Timestamp,
              dd.Name,
              dd.Type,
              dd.Serial,
              dd.TotalYield AS V1,
              dd.Power AS V2,
              cons.EnergyUsed AS V3,
              cons.PowerUsed AS V4,
              NULL AS V5,
              spot.Udc1 AS V6,
              NULL AS V7,
              NULL AS V8,
              NULL AS V9,
              NULL AS V10,
              NULL AS V11,
              NULL AS V12,
              dd.PVoutput
         FROM vwDayData AS dd
              LEFT JOIN vwAvgSpotData AS spot
                     ON dd.Serial = spot.Serial AND dd.Timestamp = spot.Nearest5min
              LEFT JOIN vwAvgConsumption AS cons
```

```
                ON dd.Timestamp = cons.Nearest5min
        ORDER BY dd.Timestamp DESC;
```

Remark: You can only change the V5..V12 parameters. V1..V4 are fixed. Also, make sure all V1..V12 are present.

For compatibility with previous SBFspot versions, V6 can be changed as follows:

VoltLogging in config file	**Replacement in vwPvoData**
NONE (disabled)	NULL AS V6
MAX(AC)	MAX(spot.Uac1,spot.Uac2,spot.Uac3) AS V6
AC(PH1)	Uac1 AS V6
MAX(DC)	MAX(spot.Udc1,spot.Udc2) AS V6
DC(ST1)	Udc1 AS V6

When in PVoutput donation mode, you can make use of V7..V12
The next example shows the query to upload Udc1(V7), Udc2(V8), Pdc1(V9) and Pdc2(V10)

```
CREATE VIEW vwPvoData AS
       SELECT dd.Timestamp,
              dd.Name,
              dd.Type,
              dd.Serial,
              dd.TotalYield AS V1,
              dd.Power AS V2,
              cons.EnergyUsed AS V3,
              cons.PowerUsed AS V4,
              spot.Temperature AS V5,
              spot.Uac1 AS V6,
              spot.Udc1 AS V7,
              spot.Udc2 AS V8,
              spot.Pdc1 AS V9,
              spot.Pdc2 AS V10,
              NULL AS V11,
              NULL AS V12,
              dd.PVoutput
         FROM vwDayData AS dd
              LEFT JOIN vwAvgSpotData AS spot
                     ON dd.Serial = spot.Serial AND dd.Timestamp = spot.Nearest5min
              LEFT JOIN vwAvgConsumption AS cons
                     ON dd.Timestamp = cons.Nearest5min
        ORDER BY dd.Timestamp DESC;
```

With a bit more experience you can include gas/water usage and let SBFspotUploader do the upload. The consumption table can be used as a starting point.

Disclaimer

Although I did my very best to make this document as comprehensive as possible, there is still a chance I forgot something.

Sorry for this.

I tested all the commands and examples, but there is still a chance it doesn't run like it should on your computer.

Sorry for this.

But don't worry. We're there to help.

If you like this tool, join the SBFspot Team and/or give a small donation

THANKS

https://sbfspot.codeplex.com/

V 1.0 (29-AUG-2014) Initial version by *SBF*