

## Proposed Database Changes June 2019

### Devices and Sensors

The current **devices** table includes a pointer to a **device\_types** table, in turn, which contains information about the sensors.

#### devices

Field	Type	Null	Key	Default	Extra
device_id	int(11)	NO	PRI	NULL	auto_increment
device_name	varchar(16)	NO	UNI	NULL	
device_type	int(11)	NO	MUL	NULL	
owner_id	int(11)	YES	MUL	NULL	
device_latitude	double	YES		53.725383	
device_longitude	double	YES		-0.336571	
device_altitude	double	YES		NULL	

#### device\_types

Field	Type	Null	Key	Default	Extra
device_type	int(11)	NO	PRI	NULL	auto_increment
processor	text	YES		NULL	
Connection	varchar(8)	YES		NULL	
particle_sensor	text	YES		NULL	
temp_sensor	text	YES		NULL	
power	text	YES		NULL	
Software	text	YES		NULL	
Other	text	YES		NULL	

Note that the device\_types table is non-scalable because a separate column is present for each sensor type contained in the device. To make this scalable it is planned that we remove the columns particle\_sensor and temp\_sensor . The information can then be contained in two additional tables as follows:-

#### device\_sensors

This table enables a device to include multiple sensors and have sensors easily added/removed as required

Device_id	Sensor_id
10	1
10	3
22	2
22	4

## sensors

This table is a unique list of sensors e.g.

Sensor_id	Type	Description
1	BME280	Pressure, Temperature, Humidity
2	BME680	Pressure, Temperature, Humidity, VOC
3	PMS7003	PM1.0, PM2.5, PM10
4	SDS011	PM2.5,PM10
5	NE06M	GNSS
6	DHT11	Humidity, Temperature
7	BMP280	Pressure, Temperature
8	DS18B20	Temperature
9	MICS6814	NO2,CO,NH3

This means that we can add new sensors to the sensors table and not have to change the columns of device\_types.

### Changing Over

The current tables can be left as-is until the Web API software is changed to use the new tables.

## New Data Types

In order to include NO2, CO, NH3 from other sources of information we need to add these to the list of accepted values in the dbLoader settings.py and also add them to the reading\_value\_types table.

### Changing Over

Add new reading types to the reading\_value\_types table and note the ids of the newly added types.

Edit settings.py and add these to the types\_id dictionary including any aliases (e.g. temperature can be written in full or simply as temp and thus far is the only types\_id alias)

Restart dbLoader so it reads the new types\_id dictionary.

## Changes to enable locating the nearest device to a given location

It has been suggested that we use a Point() datatype to speed up searching for devices within a given region. Adding a persistent column calculated using Point(device\_latitude,device\_longitude) will satisfy that requirement and has the advantage that when device\_longitude or device\_latitude are changed the new column is automatically updated. This would only happen if a device is moved.

An index is also required on the new column for reasons of speed. The preferred index is a SPATIAL index which is a 2D index designed for geometry searching but that requires mariaDB 10.2. Since the current devices table is quite small we can get away with a simple 1D index.

### Changing Over

Run the following SQL commands :-

```
alter table devices add column loc Point as (Point(device_latitude,device_longitude)) persistent;
```

```
alter table devices add index loc_idx (loc(25))
```

The devices table is small and these execute very quickly.