Deployment Data Cleanup

D Goldsmith, R Wilkins

December 17, 2012

1 Introduction

Below is a description of the data summary process, how the information is processed an yields calculated.

It takes the form of a walk though of the yield calculation process, and includes code snippets from the R script used to generate the data.

The document was generated in R and Latex, using the Sweave plugin.

2 Summary Table

A new table added to the database, the summary table is intended to hold summary statistics on deployments. This means that future work can avoid having to process entire data sets when dealing with yield, or other summarised functions.

The summary table takes the same form as the reading table, with an additional *summary type* column, these summary types are taken from a lookup table in the database.

Database Rows, and expected inputs are given below

Row	Type	Description
Time	PK,Required	Timestamp of summary, In general I would expect this to use midnight to summarise a complete day. However, if more detailed summaries (such as hourly) are needed, this should not be a problem.
nodeId	PK,Required	Id of node that this summary is from
sensorTypeId	PK	Id of sensor that this summary is from, this can be left NULL to indicate whole node summary samples (for example yield)
summaryTypeId	FK	Id of summary type.
locationId	FK	Id of location this node is from, to keep parity with the reading table
value	float	Value of the summary
textValue	string(30)	Optional text description of the summary, for example “Hot” if we are dealing with exposure graphs.

Table 1: Summary Table Description

3 Scripts

This section has a description of the scripts used process the data, and combine all samples into one database.

These scripts are designed to work with the new format (location aware) database format.

They can found in the *dataclense* directory of the *cogent-house/djgoldsmith-devel* repository.

processCC.py Transfers current cost data from the old style sqlite database, into the new format database.

processAr.py Transfers data from an Archrock postgresql database into the new format database.

getStats.R R script that calculates yields for each deployment in a given database

calcKwh.R R script to caluclate KwH usage from current cost readings.

Further details of these scripts are given below

4 getStats.R

This script calculates summary statistics for all houses in a given database. The statistics are output in two formats.

- *.csv* file with summary output for this database
- update rows in the *summary* table given these statistics

To run the script modify the source file with the relevant database access name. Then run the script through R.

4.1 Script initialisation

- Load the relevant R librarys
- Connects to the database
- Loads the Relevant Lookup tables into memory.
  - Houses Table
  - Sensor Table (For Calibration)
  - Sensor Type Table
  - Summary Type Table

```
> #Load Relevant Libraries
> library(RMySQL)
> library(ggplot2)
> library(plyr)
> library(xtable)
> #Setup Database Connection
> drv ← dbDriver("MySQL")
> con ← dbConnect(drv,dbname="mainStore",user="chuser")
> #Load the Relevant lookup tables into memeory
> allHouses ←  dbGetQuery(con,statement="SELECT * FROM House WHERE address != 'ERROR-DATA'")
> summaryData ← dbReadTable(con,"SummaryType")
> calibrationData ← dbReadTable(con,"Sensor")
> sensorType ← dbReadTable(con,"SensorType")
> ##Sensors we are interested in (For Yield Calculateions)
> sensorTypeList ← subset(sensorType,
+                            name=="Temperature" |
+                            name=="Humidity" |
+                            name=="Light PAR" |
+                            name=="Light TSR" |
+                            name=="CO2" |
+                            name=="Air Quality" |
+                            name=="VOC" |
+                            name=="Battery Voltage" |
+                            name=="Power")
> #Create a temporary table to hold summary informtion
> houseData ← data.frame(address = allHouses$address,
+                         dbStart = NA,
+                         dbEnd = NA,
+                         dataStart = NA,
+                         dataEnd = NA,
+                         totalNodes = NA,
+                         coNodes = NA,
+                         yield = NA,
+                         yieldSD = NA,
+                         yieldMin = NA,
+                         yieldMax = NA,
+                         totalSamples = NA,
+                         yieldDays = NA
+                         )
> #Choose a house (In this case 131 Jodrell Street)
> i=16
> THEHOUSE ← allHouses[i,]
> hseName ← THEHOUSE$address
```

At the end of this we have 1) A connection to the Database 2) A collection of lookup tables used later in the application 3) One main dataframe, to hold the summary information generated during the summarisation process.

When initialised, the main dataframe *houseData* should look something like this

	address	dbStart	dbEnd	dataStart	dataEnd	totalNodes	coNodes	yield	yieldSD	yieldMin	yieldMax	totalSamples	yie
1	5 Elm Road												
2	158 Trevelyan Crescent												
3	1 Avon Road												
4	10 Southam Gardens												
5	73 St Peters Road												
6	28 Hastings Road												

Table 2: Initialised Summary Table