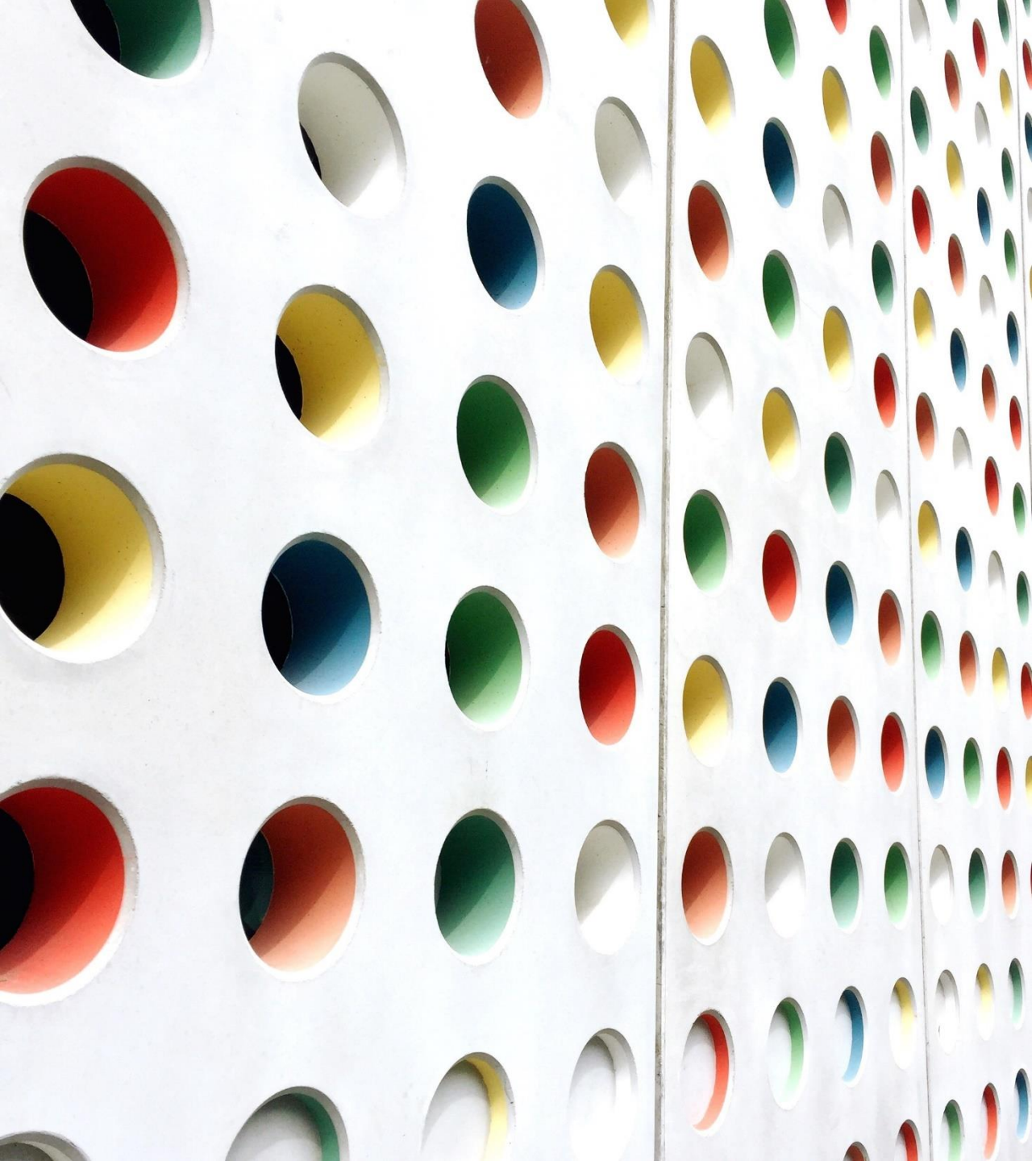


Database Start Here

v3

Alan Lomax

Table of Contents

Before you get Started

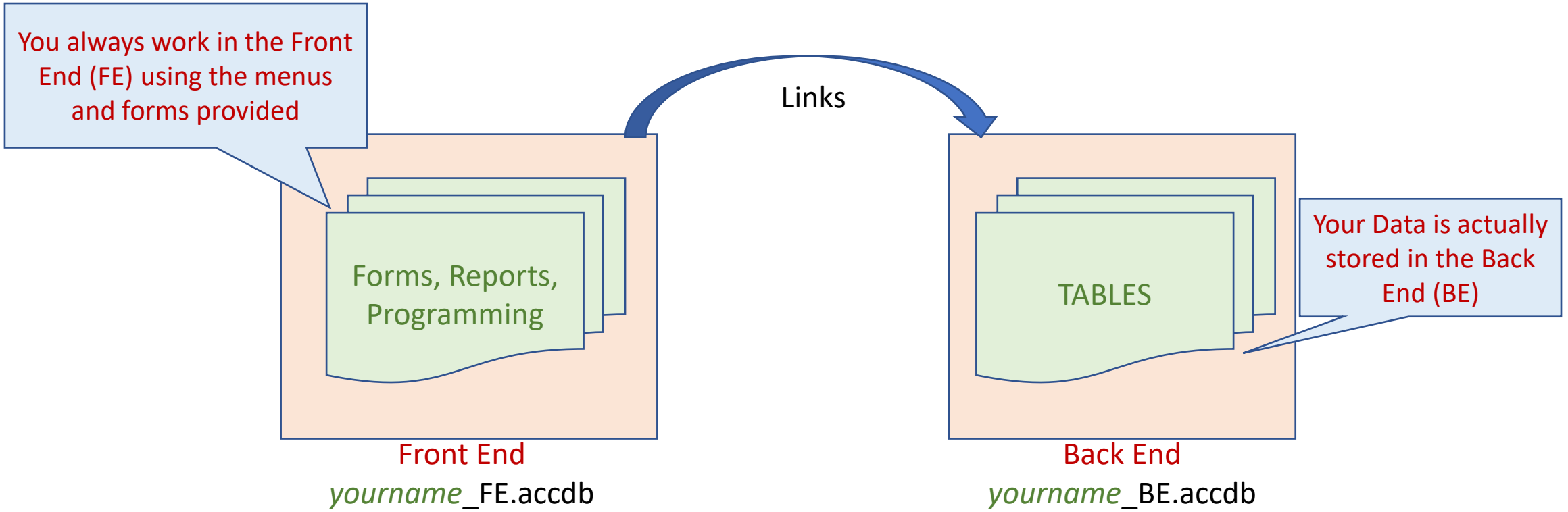
This database is written in and runs with Microsoft Access (2016). You only ***need*** Access if you want to modify the structure and the programming. To just use the database all you need is the Access 'Run Time' which is available from Microsoft here:

https://www.microsoft.com/en-ca/download/details.aspx?id=50040

The files that make up the system are described here:

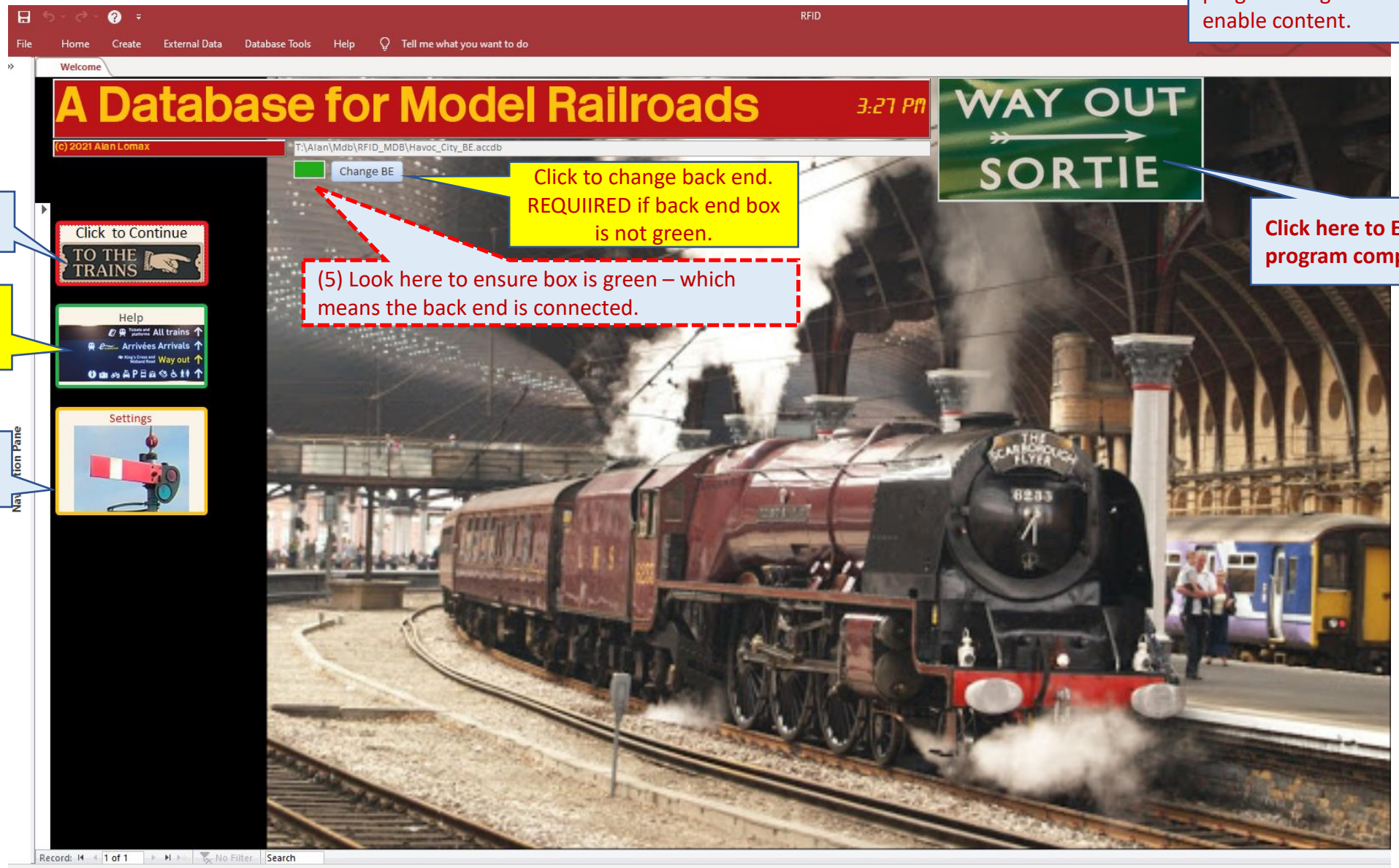
1. This 'Start Here Guide' of course … it is recommended you review it before diving in
2. The Access Database Front End and Back End files. I recommend you name them as a pair. The default name is NEW_FE.accdb and NEW_BE.accdb but you can change NEW to be anything you like.
3. In general you always only ever run the FE file … and it 'attaches itself' to the back end. The first time you run the front end it may complain the back end is missing … and you simply navigate to the back end file and click on it.
4. There is an Executable file for use with the StaRFIshRail RFID system.
   - Additionally there is a library file (*.dll) that this executable needs and there is also a text config file
5. A software library called M2Mqtt.Net.tlb which is included. (for MQTT not Access)
6. IF YOU ARE UPGRADING see the caution note along with the list of changes in the change log at the end of this presentation.

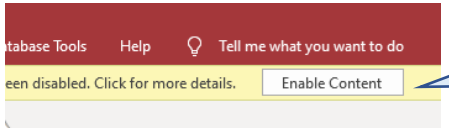
Front End and Back End



IMPORTANT:
Double Click the FE access database and it should launch automatically. It will test for and refresh the links to the BE. **If the two files are not in the same place then it wont know where the BE is and you will be prompted to locate the file.**
On the welcome page (the first page that comes up in the FE) there is a box on the top left that shows where the front end thinks the BE file is. If the box is green the file is found and the two are connected! Access wont ask again unless files are moved or renamed.
There is a button to deliberately break the connection and connect to a different back end (desirable when testing).

The Welcome Screen





You may get a security warning like this. Since this database relies on programming to work you must click enable content.

Click to go to the main forms

** START HERE ** Some built in basic instructional pages

Change or review settings

Click to change back end. REQUIIRED if back end box is not green.

(5) Look here to ensure box is green – which means the back end is connected.

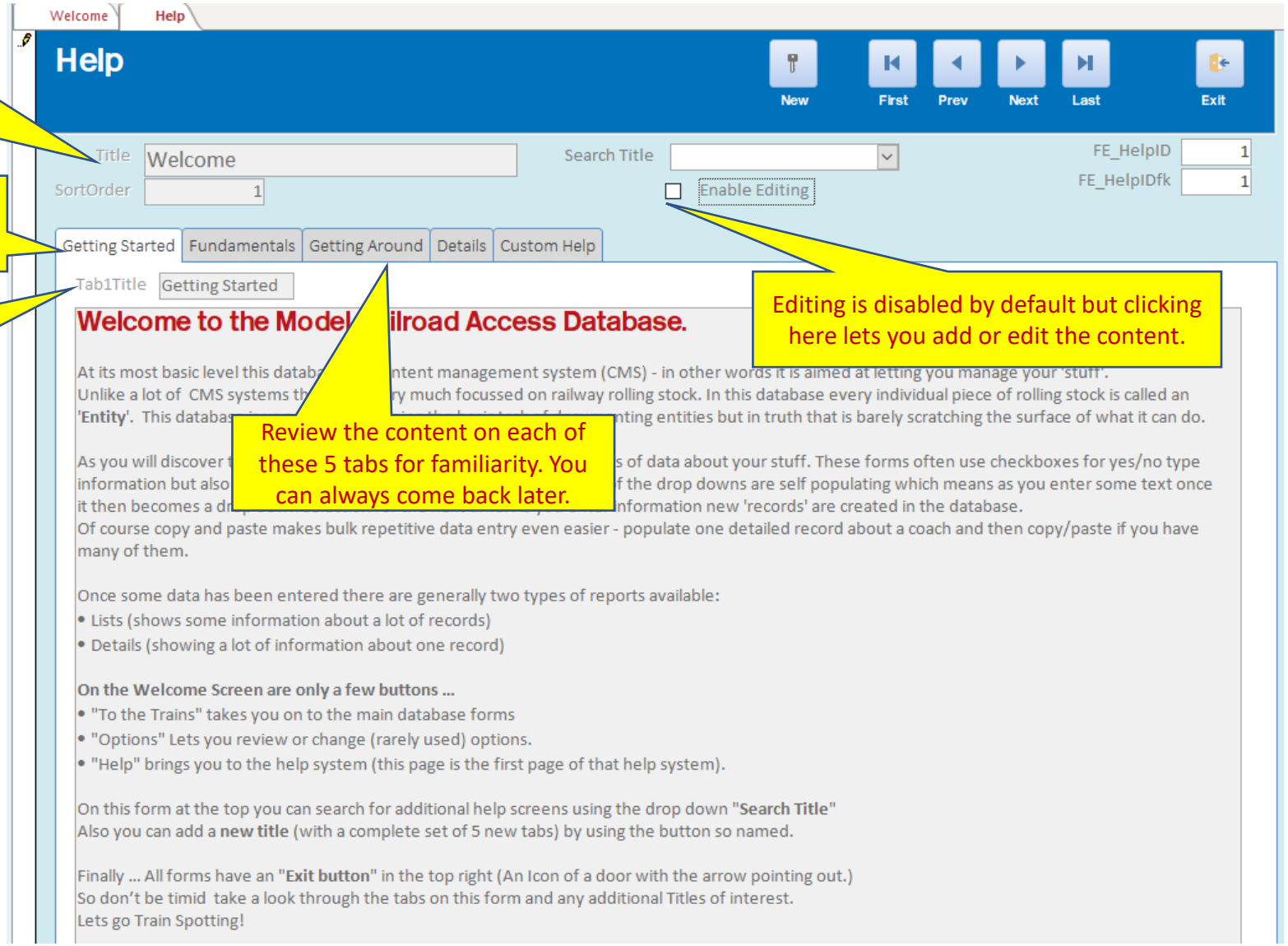
Click here to Exit the program completely

The Help Screen

Help data is saved in the database like other records are.
Each Record has a Title

For Each title there are 5 tabs worth of possible information

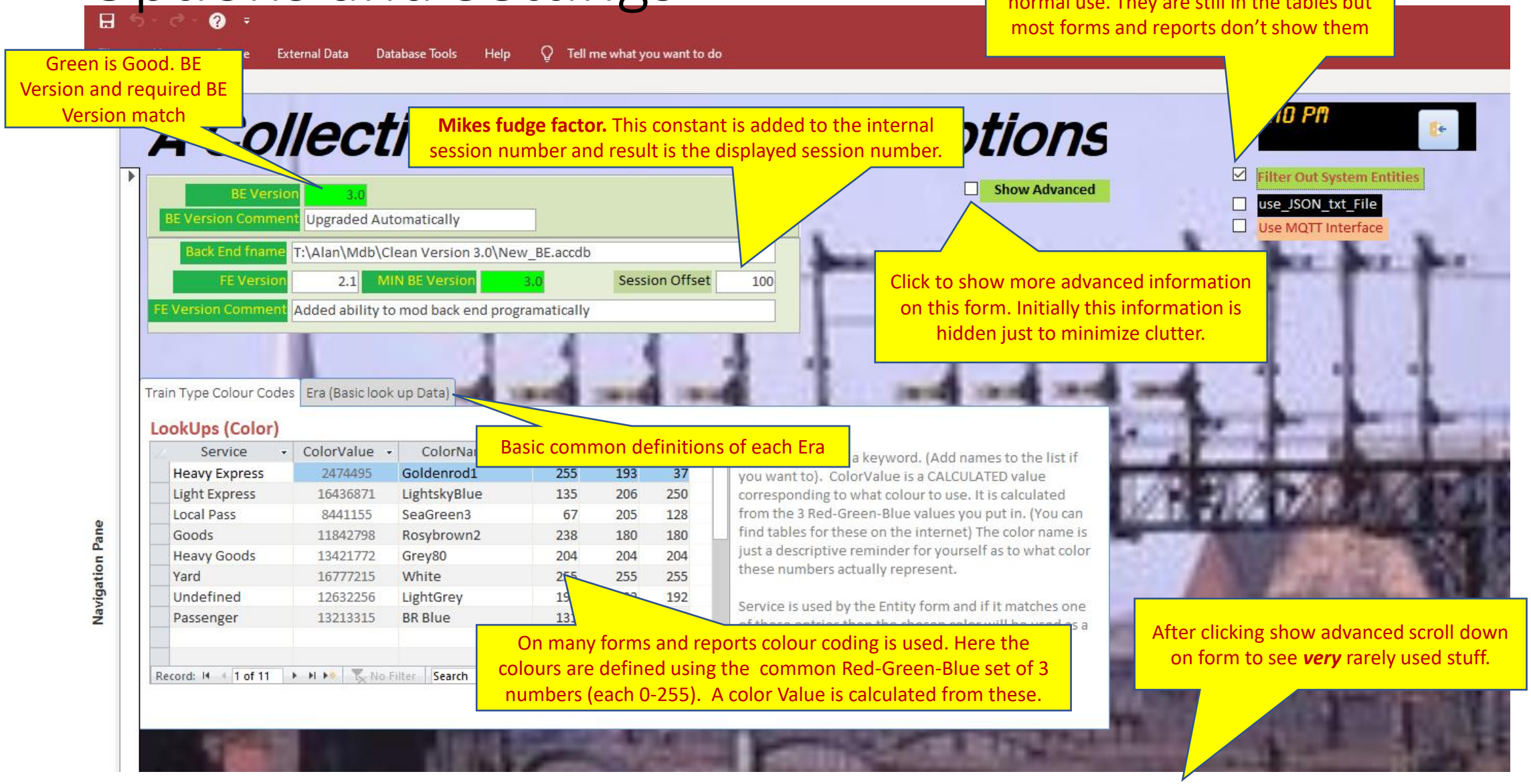
The title for each tab is also part of the data that is saved.



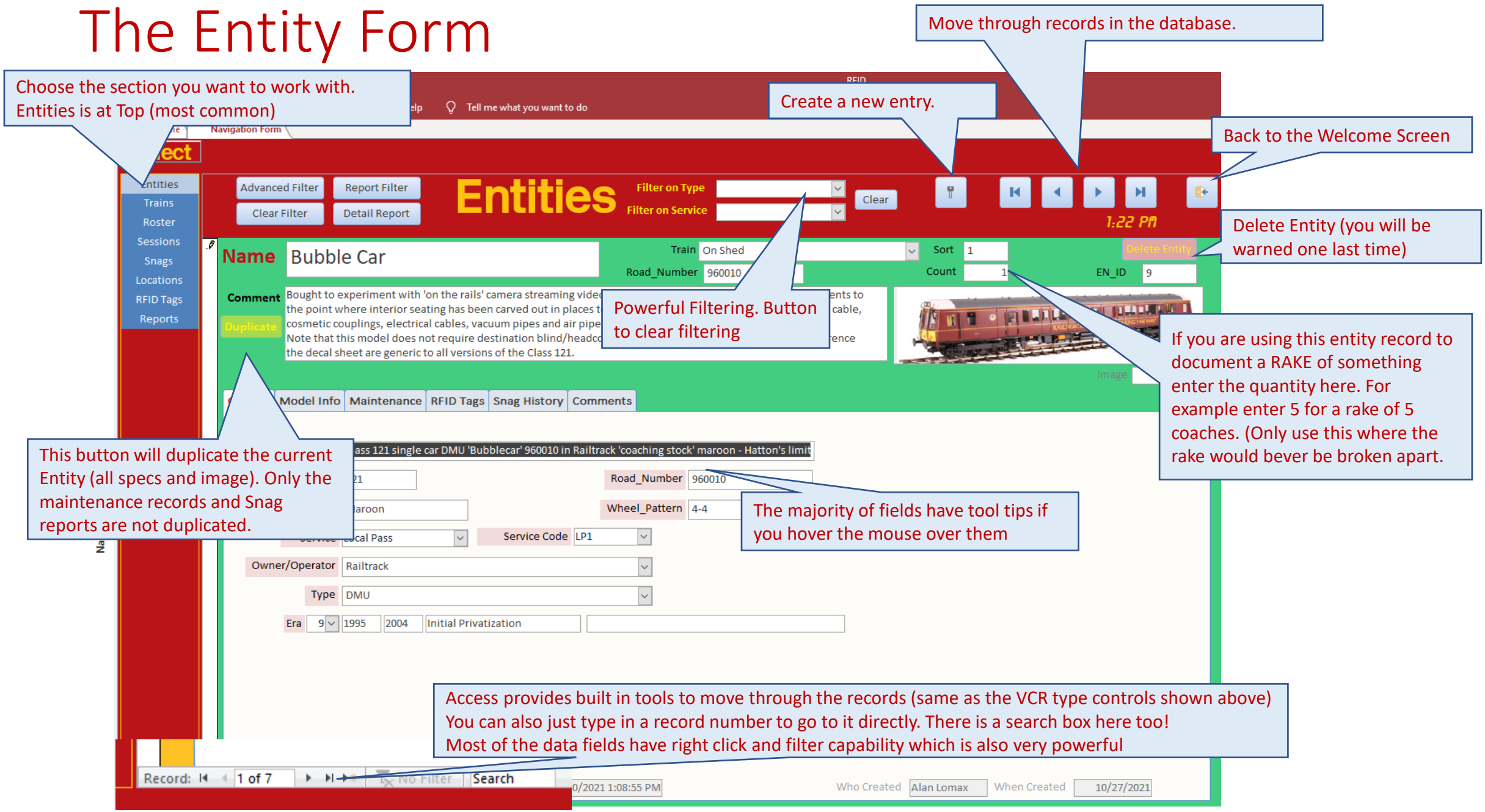
Review the content on each of these 5 tabs for familiarity. You can always come back later.

Editing is disabled by default but clicking here lets you add or edit the content.

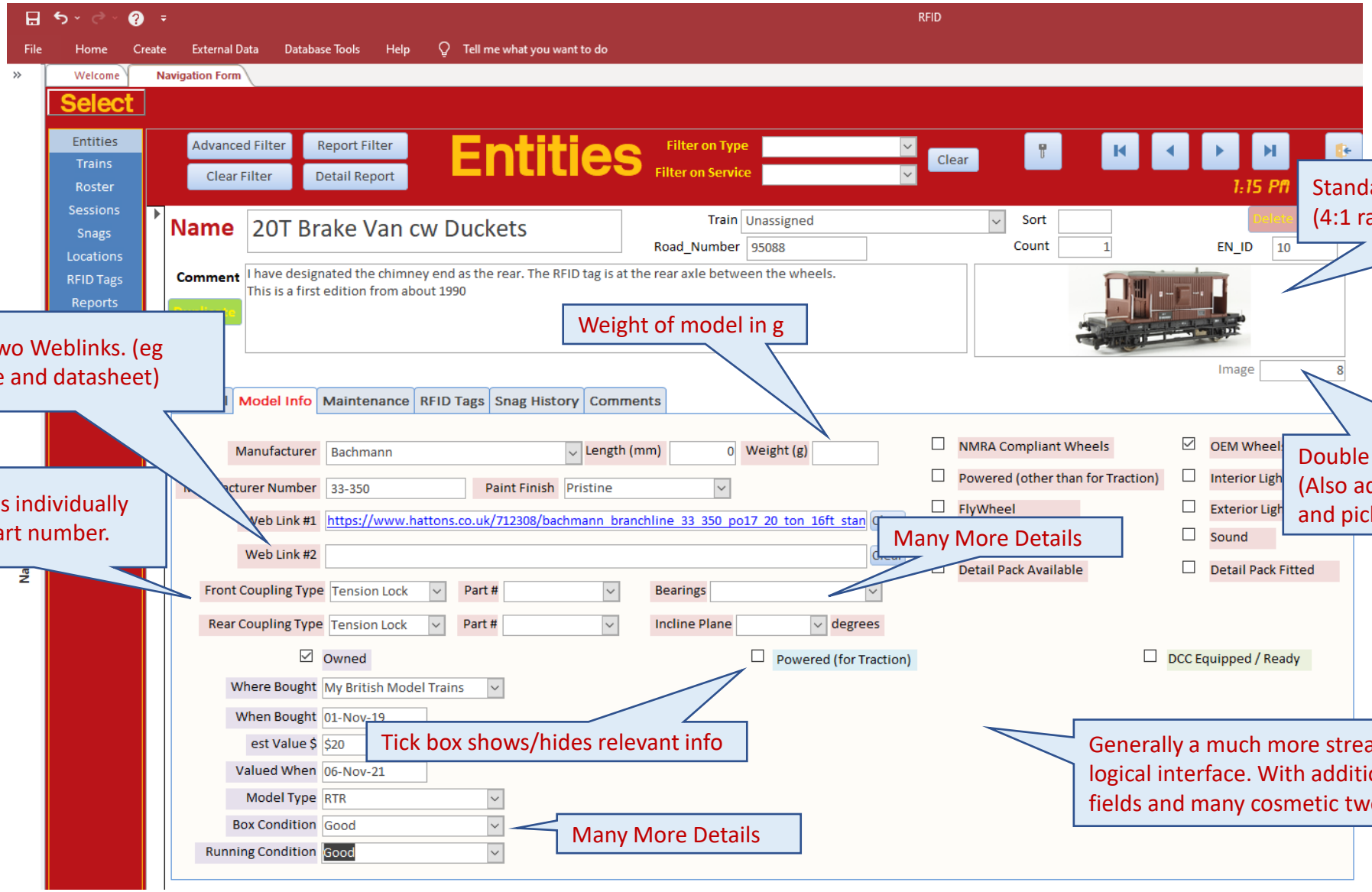
Options and Settings



The Entity Form



Entity Form (More)



Standardized proportions for images (4:1 ratio width to height)

Weight of model in g

Now with Two Weblinks. (eg vendor page and datasheet)

Double click to browse image library. (Also add a new image to the library, and pick the image for this entity.)

Both couplings individually spec'd with part number.

Many More Details

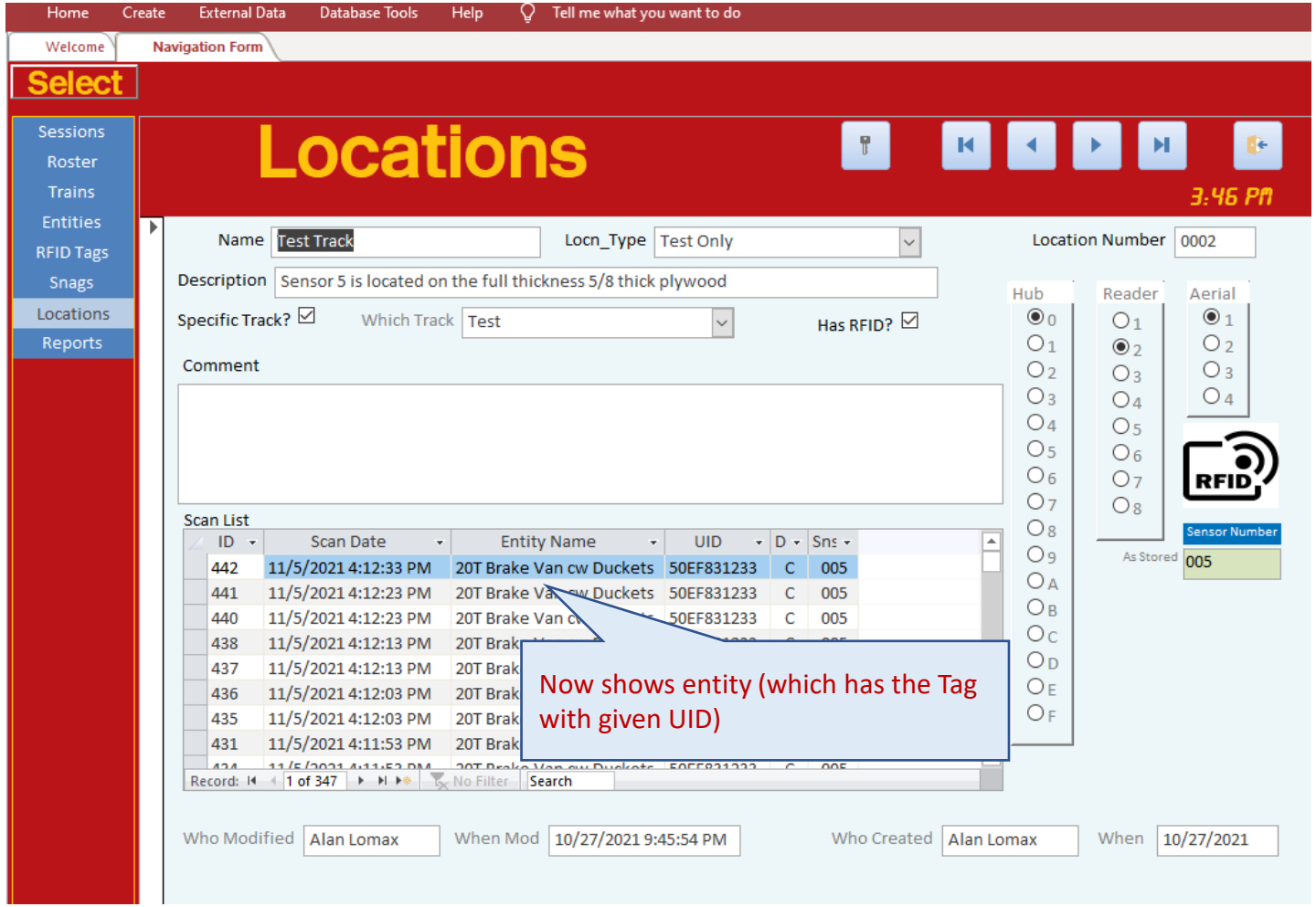
Tick box shows/hides relevant info

Generally a much more streamlined logical interface. With additional data fields and many cosmetic tweaks

Many More Details

Locations and RFID

Generally the RFID TAGs are mounted on your rolling stock. Aerials to read those tags are mounted under the rails. Each Aerial connects to a reader, and each reader to a Hub. Within the database all of this is defined as a ‘location’. Below a location has been defined and the tick box ‘Has RFID’ is selected.



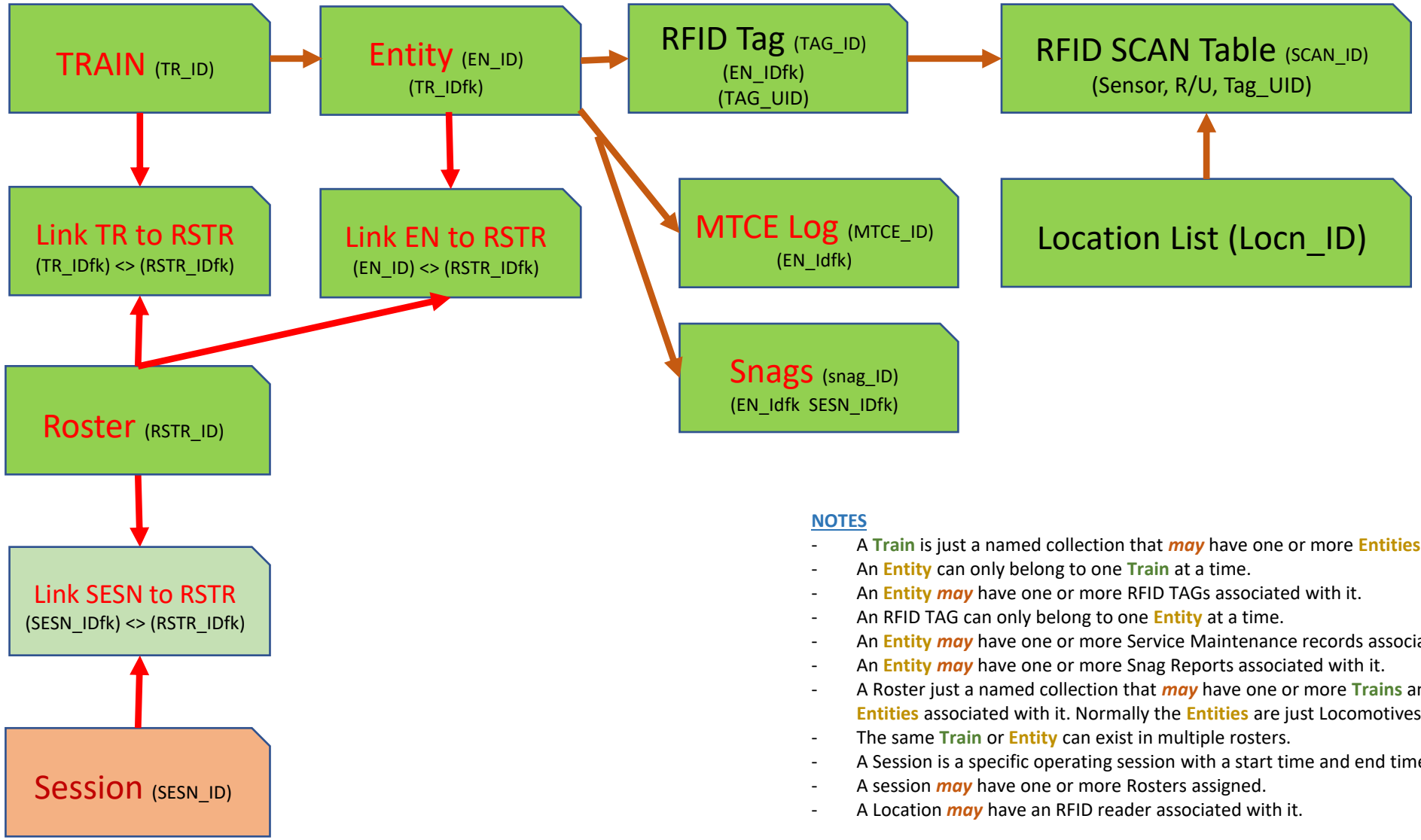
Ticking “Has RFID” will unhide the fields that allows a Hub reader and Aerial to be selected which results in a particular sensor # according to the design of the StaRFIshRail system.

The Bottom line is a location can have a sensor associated with it and if it does this page will show the Scans Records for that location..

Additional Material

Additional material is FYI only

Database Main Table Design



NOTES

- A **Train** is just a named collection that ***may*** have one or more **Entities** associated with it.
- An **Entity** can only belong to one **Train** at a time.
- An **Entity** ***may*** have one or more RFID TAGs associated with it.
- An RFID TAG can only belong to one **Entity** at a time.
- An **Entity** ***may*** have one or more Service Maintenance records associated with it.
- An **Entity** ***may*** have one or more Snag Reports associated with it.
- A Roster just a named collection that ***may*** have one or more **Trains** and ***may*** have one or more **Entities** associated with it. Normally the **Entities** are just Locomotives but this is not required.
- The same **Train** or **Entity** can exist in multiple rosters.
- A Session is a specific operating session with a start time and end time (to be set loosely and wide).
- A session ***may*** have one or more Rosters assigned.
- A Location ***may*** have an RFID reader associated with it.

Where to Now?

I recommend looking through the forms for familiarity.

During testing I created and deleted several dummy records. (Flying Scotsman theme)

You could do similar just create records to see where the info shows up on various forms and reports. You can delete the records when done to clean up the database.

The central pillar for the database is without a doubt the Entity. There are forms for entering data, reports for listing and reviewing the data plus many sub forms.

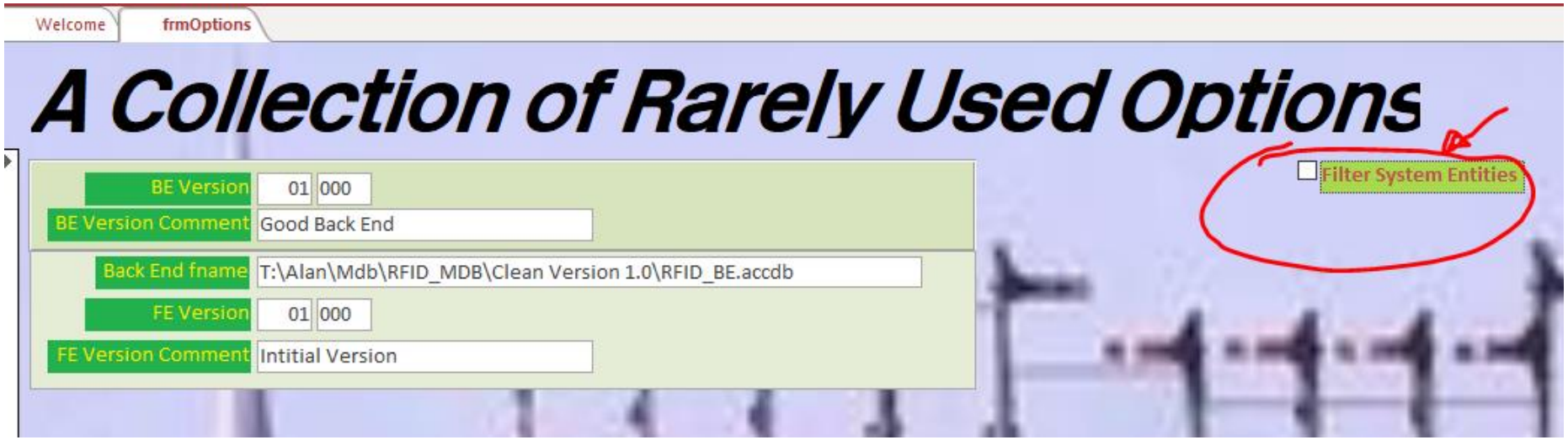
A sub form shows how one set of data relates to another .. for example a “Train” is composed of related “entities”, or an “Entity” has “Maintenance records”.

There are three entities that are ‘**system**’ related and as such are not really dummy records. They are named ‘Blank’, ‘System’, and ‘Unassigned’. Initially these entities are filtered out on mist forms and reports. (effectively ‘hidden’) There is an option on the settings screen to remove the filter and it is worth getting familiar with this behaviour.

All of these activities will help you get familiar with the database layout and operation.

System Entities

Another **system** record is simply called ‘Undefined’ which is used when RFID tags are scanned for the first time. Your workflow to install RFID tags should go like this: a) document entity in database b) Attach tag to an entity (unknown UID at this point) c) scan entity / tag. When first scanned a new RFID tag is created and it will be assigned (in the database) to the ‘Undefined’ entity. In the database you look at ‘Undefined’ and then reassign this tag away from the ‘Undefined’ entity and over to the correct entity that you just built.
The final **system** entity is called ‘system’ and it can be used for logging maintenance records not about your model railroad stuff but about your system in general. You could of course create your own entities for keeping track of other items also.
The three **system** entities are not required to be visible. There is an option on the ‘options page’ to effectively hide the three **system** entities from your forms and reports. You tick the setting and then the next time you go to the entity form there will be three less entities. See below:



RFID MQTT and Access

This database was written with the ability to read and process RFID tags.

1. Specifically a small ‘client program’ is provided for subscribing to especially formatted MQTT messages and loading them behind the scenes into a back end. (Specifically to the table: tbl_JSON_Raw)
2. The RFID system used publishes MQTT messages up to every 1/10 of a second as tags pass over the aerials. The message consists of the UID of the tag, the sensor # (aerial) and also the type of read (A new read, a ongoing read, or a loss of read)
3. Additionally the RFID Hub will send a ‘keep alive’ message every 20 seconds.
4. If configured (by default it is turned off) you can ask the Database to auto start the client program on the options page. Generally it is best if you manually launch the client first to ensure it connects, subscribes and is receiving MQTT messages. Once verified to be properly configured only then should it be auto started.
5. For test purposes the client can also log the MQTT messages to a text file for analysis.

Database MQTT Interface (Unified View)

MS Access

RFID SCAN Table (SCAN_ID)
(Sensor, R/U, Tag_UID)

SQL Insert (JSON) One Line at a time from recordset

Read JSON as a record set from tbl_json_raw processing and deleting Rows as we go. Do every 10 sec when main form is open.

tbl_JSON_raw
Each new row is a payload having all reads from previous 1/10 second

This data path was only used during testing to load the JSON messages and test Access processing. A checkbox remains in options to reenable it.

OR

Standalone EXE program

MQTT Client Code

Append / Write one line to Access Table

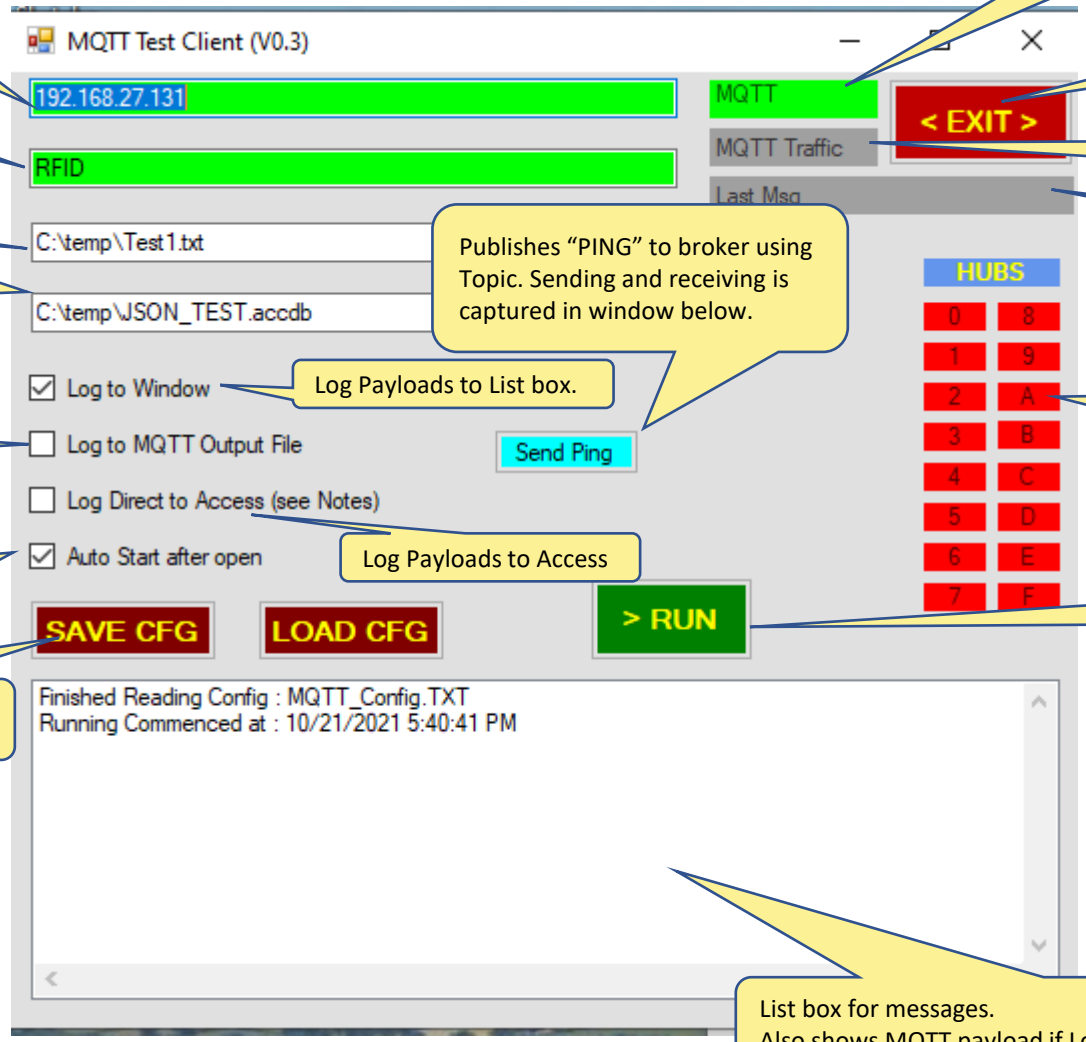
If not exist Create file Append / Write one line

User Interface

JSON Text file
- Each payload on a new line
- Each line has all reads from previous 1/10 second

Configuration Settings
MQTT_Config.TXT

MQTT RFID Tool



IP Address or URL of MQTT Broker
Green when connected.

MQTT Topic to subscribe to.
Green when subscribed.

Name of Text file where payloads will be logged. (Can Locate using the button)
Green when logging

Name of Access BE Database where payloads will be logged.
(Can Locate using the button)
Green when logging

Log Payloads to txt file.

Auto start on open
(once configuration file is set of course).

Load and Save Configuration file (All of the above settings). See MQTT_Config.TXT

Green when connected.
Red if disconnected

If you don't know what this is I can't help you

Green when traffic in last 20 seconds.
Yellow if no traffic in last 20 seconds.
Red if no traffic in 45 seconds

Time of most recent message

Publishes "PING" to broker using Topic. Sending and receiving is captured in window below.

Log Payloads to List box.

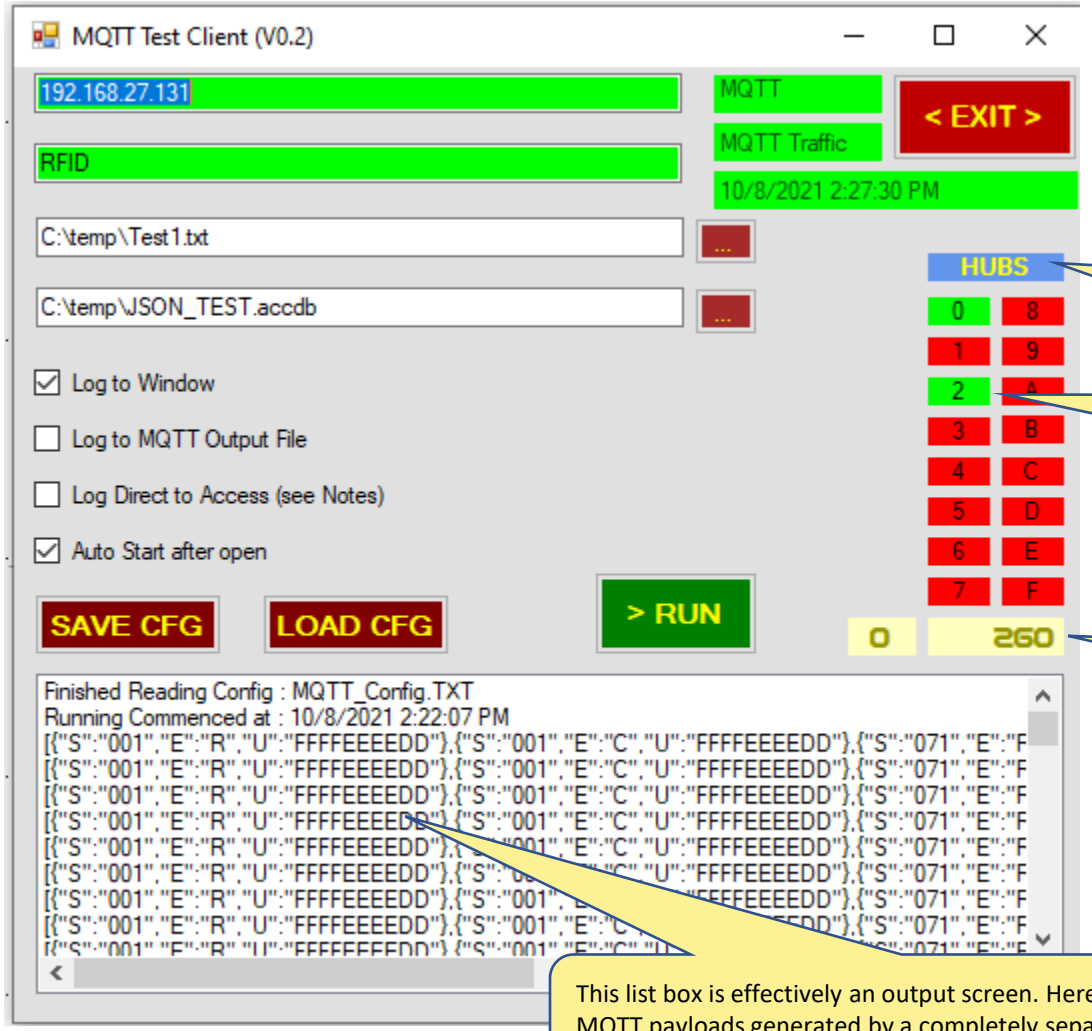
Log Payloads to Access

Which Hubs are reporting in. (0-F to match HUB address switch)
Green < 20 seconds
Yellow for > 20 seconds
Red for > 45 seconds

Manually start
Or restart if configuration has changed.

List box for messages.
Also shows MQTT payload if Log to Window is checked.

Viewing Hub Message Counts



Click Here and two additional boxes will appear below the Hub Status lights.

Click the box again to hide them away.

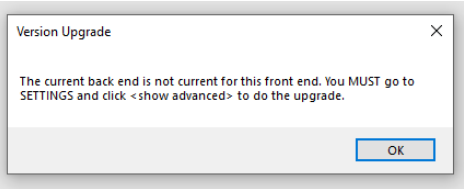
Select a HUB to monitor by clicking one of these boxes. Note Green hubs have sent messages in last 45 seconds. (Keep alive are sent every 20 seconds). Effectively this shows who is connected.

These two boxes show the currently selected HUB and the number of payloads received from that HUB. Here Hub 0 has sent 260 messages.

This list box is effectively an output screen. Here it is showing 'dummy' MQTT payloads generated by a completely separate Python program and being published to a MQTT broker running on a Raspberry Pi

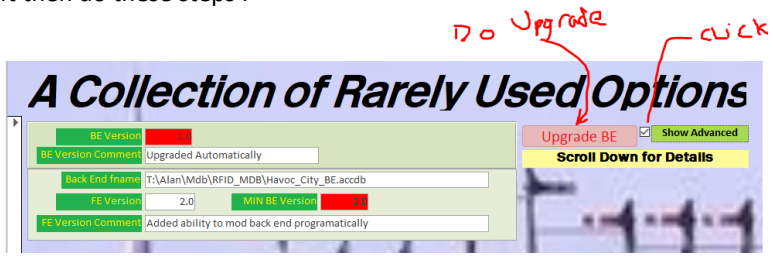
Changelog

- **CAUTION –** If you are upgrading **do NOT overwrite your back end database file**.
- You only need the new front end. The back end (with your data in it) will be upgraded in place.
- When you start the new Front End it first checks your Back End version and you will get the message box below left if an upgrade is needed.



If you get a message box like shown at left then do these steps :

- ❑ **Click OK**
- ❑ **Go to the settings screen**
- ❑ **Click Show Advanced and**
- ❑ **press the Button to upgrade BE**



- If you have several BE files you will need to connect to each BE and upgrade them one at a time. When complete the version numbers should be green once again.

Main Changes

- Bug Fixes resolving issues found during testing