zxweather

Database Structure

Document Number: DAZW-DB02

January 2013

This document provides an overview of the database structure used by zxweather.

Revision/Update Information:	This is a new manual.
Operating System:	Any
Software Version:	zxweather 0.2

© Copyright David Goodwin, 2012, 2013.

Use, reproduction and modification of this document is permitted subject to the terms of the GNU Free Documentation License, Version 1.3 or any later vesion published by the Free Software Foundation. See `http://www.gnu.org/copyleft/fdl.html` for full license text.

This document was prepared using LaTeX 2$_\varepsilon$

Contents

1 Introduction **1**
- 1.1 Changes from v0.1 . . . 1
  - 1.1.1 New Tables . . . 1
  - 1.1.2 Changes to Views . . . 2
  - 1.1.3 New functions . . . 2
  - 1.1.4 New trigger functions . . . 2
  - 1.1.5 Changes to Notification Channels . . . 2
- 1.2 Related Documents . . . 2

2 Tables **3**
- 2.1 station_type . . . 3
  - 2.1.1 Applications . . . 3
  - 2.1.2 Columns . . . 3
- 2.2 station . . . 4
  - 2.2.1 Applications . . . 4
  - 2.2.2 Columns . . . 4
- 2.3 sample . . . 5
  - 2.3.1 Applications . . . 5
  - 2.3.2 Columns . . . 5
  - 2.3.3 Indices . . . 7
  - 2.3.4 Triggers . . . 7
  - 2.3.5 WH1080 Implementation Notes . . . 7
- 2.4 wh1080_sample . . . 8
  - 2.4.1 Applications . . . 8
  - 2.4.2 Columns . . . 8
  - 2.4.3 Triggers . . . 9
- 2.5 live_data . . . 9
  - 2.5.1 Applications . . . 9
  - 2.5.2 Columns . . . 10
- 2.6 db_info . . . 11
  - 2.6.1 Columns . . . 12
  - 2.6.2 Defined values . . . 12

3 Views **13**
- 3.1 latest_record_number . . . 13
- 3.2 Daily, Montly and Yearly Records . . . 13
  - 3.2.1 Columns . . . 13

4 Functions and Trigger Functions **15**
- 4.1 Functions . . . 15
  - 4.1.1 apparent_temperature . . . 15
  - 4.1.2 dew_point . . . 15
  - 4.1.3 minimum_version_string . . . 16
  - 4.1.4 version_check . . . 16
  - 4.1.5 wind_chill . . . 17
  - 4.1.6 wind_direction_to_degrees . . . 17
- 4.2 Trigger Functions . . . 18
  - 4.2.1 compute_sample_values . . . 18
  - 4.2.2 compute_wh1080_sample_values . . . 18
  - 4.2.3 live_data_update . . . 18

5 Domains **19**
5.1 rh_percentage . . . 19
5.2 wind_direction . . . 19

6 Notification Channels **21**
6.1 live_data_updated . . . 21
6.1.1 Applications . . . 21
6.2 new_sample . . . 21
6.2.1 Applications . . . 21
6.3 update_complete . . . 22
6.3.1 Applications . . . 22

7 Versioning **23**
7.1 Backwards compatibility . . . 23
7.2 Forwards compatibility . . . 23
7.2.1 Checking compatibility . . . 24
7.3 Handling v0.1.x databases . . . 24

1

Introduction

zxweather uses a PostgreSQL database to store weather data and for some communication between components. No other RDMBS is supported in any way.

This manual describes the schema used by zxweather version 0.2. It is divided up into five chapters which define the various parts of the database:

- Chapter 2: Tables
- Chapter 3: Views
- Chapter 4: Functions and Trigger Functions
- Chapter 5: Domains
- Chapter 6: Notification Channels

For documentation on the database schema used by zxweather version 0.1, see document `DAZW-DB01`.

1.1 Changes from v0.1

zxweather version 0.2 introduced significant database changes to allow multiple different weather stations of different hardware types to all log to the same database. This resulted in a number of new tables and changes to all existing tables and views. It also introduces schema versioning minimum application versions to be specified allowing applications to determine if they are compatible with the database or not.

1.1.1 New Tables

These tables are new in the v0.2 schema:

- `db_info` (section 2.6)
- `station` (section 2.2)
- `station_type` (section 2.1)
- `wh1080_sample` (section 2.4)

1.1.2 Changes to Views

All views now have a `station_id` column which indicates which station the particular row belongs to. The `latest_record_number` view now includes multiple rows - one for each station.

1.1.3 New functions

These functions are new in the v0.2 schema:

- `minimum_version_string(character varying)` (section 4.1.3)
- `version_check(varchar, integer, integer, integer)` (section 4.1.4)
- `wind_direction_to_degrees(character varying)` (section 4.1.6)

1.1.4 New trigger functions

These trigger functions are new in the v0.2 schema:

- `compute_wh1080_sample` (section 4.2.2)

1.1.5 Changes to Notification Channels

Notifications about data updates are now sent with a payload containing a station code. This station code is the code for the station that caused the notification to be generated.

1.2 Related Documents

The following documents describe the rest of zxweather.

DAZW-IG01	zxweather Installation Reference, version 0.1
DAZW-UR02	zxweather Upgrade Installation Guide, version 0.2
DAZW-WG01	zxweather WH1080 Utilities Users Guide, version 0.1

2

Tables

This chapter documents the database tables used by zxweather.

2.1 station_type

This table stores information about different station hardware types. It typically contains only two rows:

1. *FOWH1080* - Fine Offset WH1080-compatible hardware. Any samples logged against a station of this type must have an associated record in the `wh1080_sample` table.
2. *GENERIC* - Generic hardware. It is assumed that there are no hardware-specific tables that need to be replicated by the db_push tool.

2.1.1 Applications

The following applications use this table:

- db_push
- wh1080 update tools

2.1.2 Columns

station_type_id

The primary key.

code

An 8-character code used to uniquely identify the station type.

title

A descriptive name for the station type.

2.2 station

This table contains information about a specific weather station in the database. This includes some configuration data such as its sample interval and if live data is available for it.

2.2.1 Applications

The following applications use this table:

- Web Interface
- Desktop Client
- db_push
- wh1080 update tools

2.2.2 Columns

station_id

The primary key.

code

A five character unique identifier for the station. This is used as a payload for station-specific notification messages in the database. It also forms part of the URL for the station in the zxweather web interface.

Its value should never be changed once set as doing so will break any existing links to pages within the zxweather web interface.

title

A name for the weather station that can be displayed to users.

description

A full description for the weather station that can be displayed to users.

station_type_id

The type of hardware that this weather station runs on.

This is a foreign key to the `station_type` table.

sample_interval

How often new samples are logged to the database (in seconds).

This is used in the web interface for calculating maximum cache ages and for finding gaps in the data.

live_data_available

If this weather station makes live data available in the `live_data` table. If this is set to true then there must be a record in that table for this station. That record should be updated when ever new data is available.

If it is set to false then clients should use the most recent sample record for the station as a substitute for live data.

2.3 sample

This is core weather history table. It contains only weather data with any hardware-specific data living in other tables. Some columns are computed automatically by a trigger on insert.

2.3.1 Applications

The following applications use this table:

- data collection tools (such as wh1080d)
- web interface
- db_push
- Desktop Client

2.3.2 Columns

sample_id

This is the primary key.

download_timestamp

When the record was downloaded from the weather station. It is not used directly for anything.

time_stamp

The time this sample was taken by the weather station.

indoor_relative_humidity

The relative humidity at the base station. Its type is the `rh_percentage` domain type (see section 5.1). Values are 0-99%.

indoor_temperature

The temperature at the base station in degrees Celsius.

relative_humidity

The relative humidity at the outdoor unit. Its type is the `rh_percentage` domain type (see section 5.1). Values are 0-99%.

temperature

The temperature at the outdoor unit in degrees Celsius.

dew_point

Dew point. This columns value is calculated on insert by a trigger using the dew_point function (Section 4.1.2). It is calculated from the `temperature` and `relative_humidity` columns.

wind_chill

Wind chill. This columns value is calculated on insert by a trigger using the wind_chill function (Section 4.1.5). It is calculated from the `temperature` and `average_wind_speed` columns.

apparent_temperature

Apparent temperature. This columns value is calculated on insert from the following columns by a trigger using the `apparent_temperature` function (section 4.1.1):

- `temperature`
- `relative_humidity`
- `average_wind_speed`

absolute_pressure

The absolute pressure in hectopascals (hPa, same as the millibar).

average_wind_speed

Average wind speed in metres per second.

gust_wind_speed

Gust wind speed in metres per second.

wind_direction

Wind direction in degrees. Zero is North.

rainfall

The rainfall for this record in millimetres.

WH1080-compatible hardware does not directly provide rainfall values in its samples. It is calculated as the difference from the `total_rain` from the previous sample. If the `rain_overflow` column is set then the actual value is used instead. Values are multiples of 0.3mm.

station_id

This is the ID of the station the sample came from. It is a foreign key to the `station` table.

2.3.3 Indices

This table has the following indices:

- `idx_time_stamp` on `time_stamp` column.

2.3.4 Triggers

The `compute_sample_values` trigger function is executed for each row inserted. This trigger calculates values for columns such as `dew_point`. See section 4.2.1 for more details on this trigger function.

2.3.5 WH1080 Implementation Notes

All data specific to WH1080-compatible hardware is stored in the `wh1080_sample` table. Each sample that comes from WH1080-compatible hardware should have a matching record in that table to enable values such as rainfall to be calculated.

time_stamp column

For WH1080 hardware, the `time_stamp` column is calculated from the computers clock by the wh1080 update tool. As such the time stamp is only approximate and may be out by a minute or two.

If the `last_in_batch` column in the `wh1080_sample` table is `true` then the timestamp value in the associated `sample` record was determined by waiting for the live data record on the weather station to change. The live data records `sample_time` field is then used to compute the timestamp for this record.

If the `last_in_batch` column is `false` then the records timestamp value was calculated based on the its `record_number`, the `sample_interval` of the previous record and the `time_stamp` of the previous record.

2.4 wh1080_sample

This table contains data specific to WH1080-compatible hardware. Each row in this table is associated with a row in the `sample` table. It is not used by the web interface or desktop client but its contents is replicated by the db_push tool.

2.4.1 Applications

The following applications use this table

- wh1080 update tools
- db_push

2.4.2 Columns

sample_id

The primary key.

This column is a foreign key to the `sample` table.

sample_interval

This is the number of minutes since the previous sample on the weather station. It is used for calculating timestamps. It is not used directly for anything else.

record_number

The history slot on the weather station this record was downloaded from. Its range is 0-4079. It is only used to track which records have already been downloaded from the weather station when performing updates and for calculating timestamps.

last_in_batch

If this record was the final record in a batch of records downloaded from the weather station. This records timestamp is the one used to calculate all other records downloaded in the batch.

invalid_data

If the record on the weather station had the invalid data status flag set. The WH1080 documentation refers to this as both "rain counter value is not valid" and "no sensor data received"

wind_direction

Wind direction. Its type is the `wind_direction` domain type. See section 5.2 for valid values.

total_rain

Total rainfall since it last overflowed. This is used only for calculating the values in the sample tables `rainfall` column. Values are a multiple of 0.3. Unit is the millimetre.

rain_overflow

If an overflow in the `total_rain` counter has occurred. Used for calculating values in the sample tables `rainfall` column.

2.4.3 Triggers

The `compute_wh1080_sample_values` trigger function is executed for each row inserted. This trigger calculates a number of values in the associated sample record. See section 4.2.2 for more details on this trigger function.

2.5 live_data

This table stores live data from all weather stations. It only ever contains one record for each weather station. It is updated by data collection tools such as the `wh1080d` program when ever there is new live data. Its set of fields is a subset of those in the sample table.

This table does not provide access to live rainfall data. Calculating this for WH1080 hardware would be difficult as it requires access to the previous samples rainfall. The wh1080d application updates live data before downloading and inserting any new history records so the previous samples rainfall data is not available from the live_data update trigger or within the wh1080d application.

2.5.1 Applications

The following applications use this table:

- wh1080d (database update daemon)
- web interface
- desktop client
- db_push

2.5.2 Columns

All columns except wind direction and invalid data allow nulls.

station_id

The station this live data record is for. This column doubles as the tables primary key restricting the table to one record for each station.

download_timestamp

When this data was downloaded from the weather station.

indoor_relative_humidity

The relative humidity at the base station. Values are 0-99%.

indoor_temperature

The temperature at the base station in degrees Celsius.

relative_humidity

The relative humidity at the outdoor unit. Values are 0-99%.

temperature

The temperature at the outdoor unit in degrees Celsius.

dew_point

Dew point. This columns value is calculated on update by a trigger using the dew_point function (Section 4.1.2). It is calculated from the `temperature` and `relative_humidity` columns.

wind_chill

Wind chill. This columns value is calculated on update by a trigger using the wind_chill function (Section 4.1.5). It is calculated from the `temperature` and `average_wind_speed` columns.

apparent_temperature

Apparent temperature. This columns value is calculated on update from the following functions by a trigger using the `apparent_temperature` function (Section 4.1.1):

- `temperature`
- `relative_humidity`
- `average_wind_speed`

absolute_pressure

The absolute pressure in hectopascals (hPa, same as the millibar).

relative_pressure

Calculated relative pressure in hectopascals (hPa, same as the millibar).

This column is for future use. Its value is not currently calculated.

average_wind_speed

Average wind speed in metres per second.

gust_wind_speed

Gust wind speed in metres per second.

wind_direction

Wind direction in degrees. Zero is North.

2.6 db_info

This table stores basic information about the database in a key/value format. It was introduced in the v0.2 schema to allow identification of the database version.

2.6.1 Columns

k

This is a 10-character unique key.

v

Data. Field type is character varying (no limit)

2.6.2 Defined values

The following keys are defined in zxweather v0.2.

Key	Description
DB_VERSION	The database version. The schema described by this manual is version 2.
MIN_VER_MAJ	The minimum zxweather major version number.
MIN_VER_MIN	The minimum zxweather minor version number.
MIN_VER_REV	The minimum zxweather revision number.
*_MIN_VER_MAJ	Application-specific minimum major version number.
*_MIN_VER_MIN	Application-specific minimum minor version number.
*_MIN_VER_REV	Application-specific minimum revision version number.

All of these keys are used by applications to determine if they are compatible with a particular zxweather database. See chapter 7 for more details on how this works.

3

Views

3.1 latest_record_number

This view is used by the wh1080 database update utilities to determine what the most recent record in the database is.

3.2 Daily, Montly and Yearly Records

These views provide minimum and maximum weather conditions (temperature, humidity, etc) for each day, month or year in the database. The views are:

- daily_records
- monthly_records
- yearly_records

The PostgreSQL query planner seems to have difficulty with the queries behind these views when more than one column is included in a where clause. This has been observed in PostgreSQL version 9.1.

3.2.1 Columns

- `date_stamp`: The date the record is for (daily_records and monthly_records only).
- `year_stamp`: The year the record is for. Present only in the yearly_records view.
- `station_id`: The station this data is for.
- `total_rainfall`: Total rainfall during this day, month or year.
- `max_gust_wind_speed`: Maximum gust wind speed for the period.
- `max_gust_wind_speed_ts`: Most recent timestamp for the maximum gust wind speed record.
- `max_average_wind_speed`: Maximum average wind speed for the period.
- `max_average_wind_speed_ts`: Most recent timestamp for the maximum wind speed record.
- `min_absolute_pressure`: Minimum absolute pressure during the period.

- `min_absolute_pressure_ts`: Most recent timestamp for the minimum absolute pressure record.
- `max_absolute_pressure`: Maximum absolute pressure during the period.
- `max_absolute_pressure_ts`: Most recent timestamp for the maximum absolute pressure record.
- `min_apparent_temperature`: Minimum apparent temperature during the period.
- `min_apparent_temperature_ts`: Most recent timestamp for the minimum apparent temperature record.
- `max_apparent_temperature`: Maximum apparent temperature during the period.
- `max_apparent_temperature_ts`: Most recent timestamp for the maximum apparent temperature record.
- `min_wind_chill`: Minimum wind chill during the period.
- `min_wind_chill_ts`: Most recent timestamp for the minimum wind chill record.
- `max_wind_chill`: Maximum wind chill during the period.
- `max_wind_chill_ts`: Most recent timestamp for the maximum wind chill record.
- `min_dew_point`: Minimum dewpoint during the period.
- `min_dew_point_ts`: Most recent timestamp for the minimum dewpoint record.
- `max_dew_point`: Maximum dewpoint during the period.
- `max_dew_point_ts`: Most recent timestamp for the maximum dewpoint record.
- `min_temperature`: Minimum outdoor temperature during the period.
- `min_temperature_ts`: Most recent timestamp for the minimum outdoor temperature record.
- `max_temperature`: Maximum outdoor temperature during the period.
- `max_temperature_ts`: Most recent timestamp for the maximum outdoor temperature record.
- `min_humidity`: Minimum outdoor relative humidity during the period.
- `min_humidity_ts`: Most recent timestamp for the minimum outdoor relative humidity record.
- `max_humidity`: Maximum outdoor relative humidity during the period.
- `max_humidity_t`: Most recent timestamp for the maximum outdoor relative humidity record.

4

Functions and Trigger Functions

The database contains a number of stored procedures for computing values that don't come directly from the weather station. This chapter gives an overview of those functions.

4.1 Functions

4.1.1 apparent_temperature

Calculates the Apparent Temperature using the formula used by the Australian Bureau of Meteorology.

The formula used is:

$$AT = Ta + 0.33e - 0.7ws - 4.00$$

Where Ta is the dry bulb temperature (°C), e is the water vapour pressure in hPa and ws is the Wind speed (m/s) at an elevation of 10 metres.

The vapour pressure is calculated from the temperature and relative humidity using the formula:

$$e = rh/100 * 6.105 * 2.718281828^{17.27Ta/(237.7+Ta)}$$

where Ta is the dry bulb temperature (°C), rh is the relative humidity (%).

See the Australian Bureau of Meteorology website for more details:
`http://www.bom.gov.au/info/thermal_stress/`

This function is *immutable*.

Parameters

- temperature, real
- wind_speed, real
- relative_humidity, real

4.1.2 dew_point

Calculates the approximate dew point given temperature and relative humidity. The calculation is based on the August-Roche-Magnus approximation for the saturation vapour pressure of water in air as a function of temperature. It is valid for input temperatures 0 to 60 °C and dew points 0 to 50 °C

The formula is:

$$T_d = \frac{b\,\gamma(T, RH)}{a - \gamma(T, RH)}$$

where

$$\gamma(T, RH) = \frac{a\,T}{b + T} + \ln(RH/100)$$

where the temperatures are in degrees Celsius. The constants are a = 17.271 and b = 237.7 °C

See the article on Wikipedia for more details: `http://en.wikipedia.org/wiki/Dewpoint`

This function is *immutable*.

Parameters

- temperature, real
- relative_humidity, integer

4.1.3 minimum_version_string

Returns the oldest version of the specified application that is compatible with this database. Its value is derived from the `db_info` table (section 2.6). If the application does not have a minimum version number entry in there then the minimum zxweather version is returned instead.

See chapter 7 (Versioning) for more details on how database and application versioning works.

Parameters

- application, character varying

4.1.4 version_check

This function checks if the specified application version is compatible with the database. This is done by looking up the application-specific minimum version in the `db_info` table (section 2.6) and comparing it to the supplied version number.

If the application does not have a minimum version number in the `db_info` table then the minimum zxweather version is used instead.

See chapter 7 (Versioning) for more details on how database and application versioning works.

Parameters

- application, character varying
- major, integer
- minor, integer
- revision, integer

4.1.5 wind_chill

Calculates the North American wind chill using the following formula:

$$T_{wc} = 13.12 + 0.6215T_a - 11.37V^{+0.16} + 0.3965T_aV^{+0.16}$$

where T_{wc} is the wind chill index in degrees Celsius, T_a is the air temperature in degrees Celsius and V is the wind speed at 10 metres in kilometres per hour.

The Wind chill temperature is only defined for air temperatures below 10 °C and wind speeds above 4.8 km/h.

See the article on Wikipedia for more details: `http://en.wikipedia.org/wiki/Wind_chill`

This function is *immutable*.

Parameters

- temperature, real
- wind_speed, real

4.1.6 wind_direction_to_degrees

Converts the supplied wind direction string to a value in degrees. This is used primarily by WH1080-related code as that station records wind direction as a string.

Conversion table

Value	In degrees
N	0
NNE	22.5
NE	45
ENE	67.5
E	90
ESE	112.5
SE	135
SSE	157.5
S	180
SSW	202.5
SW	225
WSW	247.5
W	270
WNW	292.5
NW	315
NNW	337.5

All other values are converted to `null`.

Parameters

- wind_direction character varying

4.2 Trigger Functions

4.2.1 compute_sample_values

Computes values for the following fields when a new record is inserted:

- dew_point (see dew_point function, section 4.1.2)
- wind_chill (see wind_chill function, section 4.1.5)
- apparent_temperature (see apparent_temperature function, section 4.1.1)

4.2.2 compute_wh1080_sample_values

Calculates any values that need calculating for WH1080-compatible hardware. Its calculations are based off the data stored in the hardware-specific `wh1080_sample` table. This is currently limited wind direction (calculated with `wind_direction_to_degrees`, section 4.1.6) and rainfall which is discussed below.

Rainfall calculation

The rainfall value in the `sample` table is calculated using the following query:

```
1  select into NEW.rainfall
2         case when NEW.total_rain - prev.total_rain >= 0 then
3             NEW.total_rain - prev.total_rain
4         else
5             NEW.total_rain + (19660.8 - prev.total_rain)
6         end as rainfall
7  from sample prev
8  -- find the previous sample:
9  where time_stamp = (select max(time_stamp)
10                     from sample ins
11                     where ins.time_stamp < NEW.time_stamp)
```

The value 19660.8 is the maximum rainfall accumulator value (65536 * 0.3mm). Rainfall values are calculated based on the total rainfall value of the new record and the previous record in the database.

4.2.3 live_data_update

Computes values for live_data table updates.

It computes values for the following fields:

- dew_point
- wind_chill
- apparent_temperature

It also generates a notification on channel live_data_updated,

It is only used on the live_data table.

5

Domains

Domains are used on a number of columns to simplify constraints, etc. This chapter gives an overview of those domains and their valid values.

5.1 rh_percentage

Used for columns containing relative humidity percentages. It is an `integer` field.

This domain used to enforce a not null constraint and values between 0 and 99. This was removed in zxweather v0.1.2 to allow for invalid data logged by WH1080-type hardware. The check constraint is now directly on the sample table so that it is executed after trigger functions are executed to tidy up the data.

5.2 wind_direction

Used for columns containing wind directions. Valid values are:

- N
- NNE
- NE
- ENE
- E
- ESE
- SE
- SSE
- S
- SSW
- SW
- WSW
- W

- WNW
- NW
- NNW
- INV (used for invalid values).

its type is `character varying(3)`

6

Notification Channels

zxweather uses the PostgreSQL NOTIFY feature to notify components of the system that new data has arrived. This chapter covers the notification channels used by zxweather and their meaning.

6.1 live_data_updated

A notification is broadcast on this channel when data in the live_data table has been updated. The notification is sent automatically from the live_data_update trigger function. The payload is the station code for the station that just received updated live data.

Clients interested in displaying live data should listen on this channel rather than polling the database.

6.1.1 Applications

The following applications use this channel:

- zxweather Desktop (GUI) Client

6.2 new_sample

A notification is broadcast on this channel when a new sample has been inserted into the database. This notification is sent automatically from the compute_sample_values trigger function. The payload is the station code for the station that received the new sample

Clients interested in displaying history samples should listen on this channel rather than polling the database.

6.2.1 Applications

The following applications use this channel:

- zxweather Database Replicator

6.3 update_complete

A notification with no payload is broadcast on this channel by the zxweather WH1080 Update Daemon when it has finished updating the live_data table and inserting any new records into the sample table. The payload is the station code for the station that the WH1080 Update Daemon is running for.

This channel does not specify what has been updated. Listen on the live_data_updated and new_sample channels as well to determine what has been updated.

If there are new history samples then the notification sequence will be:

- live_data_updated
- new_sample
- update_complete

If there are no new history samples then there will be no notification on the new_sample channel.

6.3.1 Applications

The following applications use this channel:

- wh1080d (WH1080 Update Daemon) - issues notifications when it has finished doing work.
- zxweather Database Replicator - waits for a notification on this channel before sending data to the remote database. A notification on the new_sample channel instructs it to send new samples in addition to sending live data.

7

Versioning

The database stores data to allow applications to determine if they are forwards or backwards compatible.

The data is:

- A schema version number for controlling backwards-compatibility
- Minimum application versions for forwards-compatibility.

7.1 Backwards compatibility

When ever changes are made to the database schema its version number is incremented. This is stored in the `db_info` table (section 2.6) as `DB_VERSION`.

This allows applications to determine if a given database supports the features the application requires.

As an example, all zxweather v0.2 applications require the database schema version number to be 2 or greater.

7.2 Forwards compatibility

Forwards compatibility is controlled through minimum application version numbers stored in the `db_info` table. The minimum zxweather version number is broken up into its three components and stored under the following keys:

- `MIN_VER_MAJ` - minimum major version
- `MIN_VER_MIN` - minimum minor version
- `MIN_VER_REV` - minimum revision number

Other applications may store their minimum version using the following scheme:

- `APPNAME_MIN_VER_MAJ`
- `APPNAME_MIN_VER_MIN`
- `APPNAME_MIN_VER_REV`

where APPNAME is a short unique name for the application in uppercase. If these keys are not found in the table then the zxweather version numbers are used instead.

This all allows older versions of the various individual zxweather components to be blacklisted by future databases by increasing the minimum zxweather version number and introducing lower application-specific version numbers for those older components which are still compatible.

7.2.1 Checking compatibility

Minimum version numbers can be retrieved in string form using the `minimum_version_string` function (section 4.1.3).

Application version compatibility checks can be performed by passing the application name and version components to the `version_check` function (section 4.1.4). This function will return true if the passed in version number meets or exceeds the minimum version stored in the database.

7.3 Handling v0.1.x databases

Version 1 of the database schema (used by zxweather 0.1.x) did not carry the `db_info` table. Checking for these old versions can be achieved by querying `INFORMATION_SCHEMA.TABLES` for the presence of the `db_info` table.