ROBIN Database

User Manual

3D Flex/Fixed Generic Version 2.1.7.0
August 2016
ROBIN Radar Systems B.V.

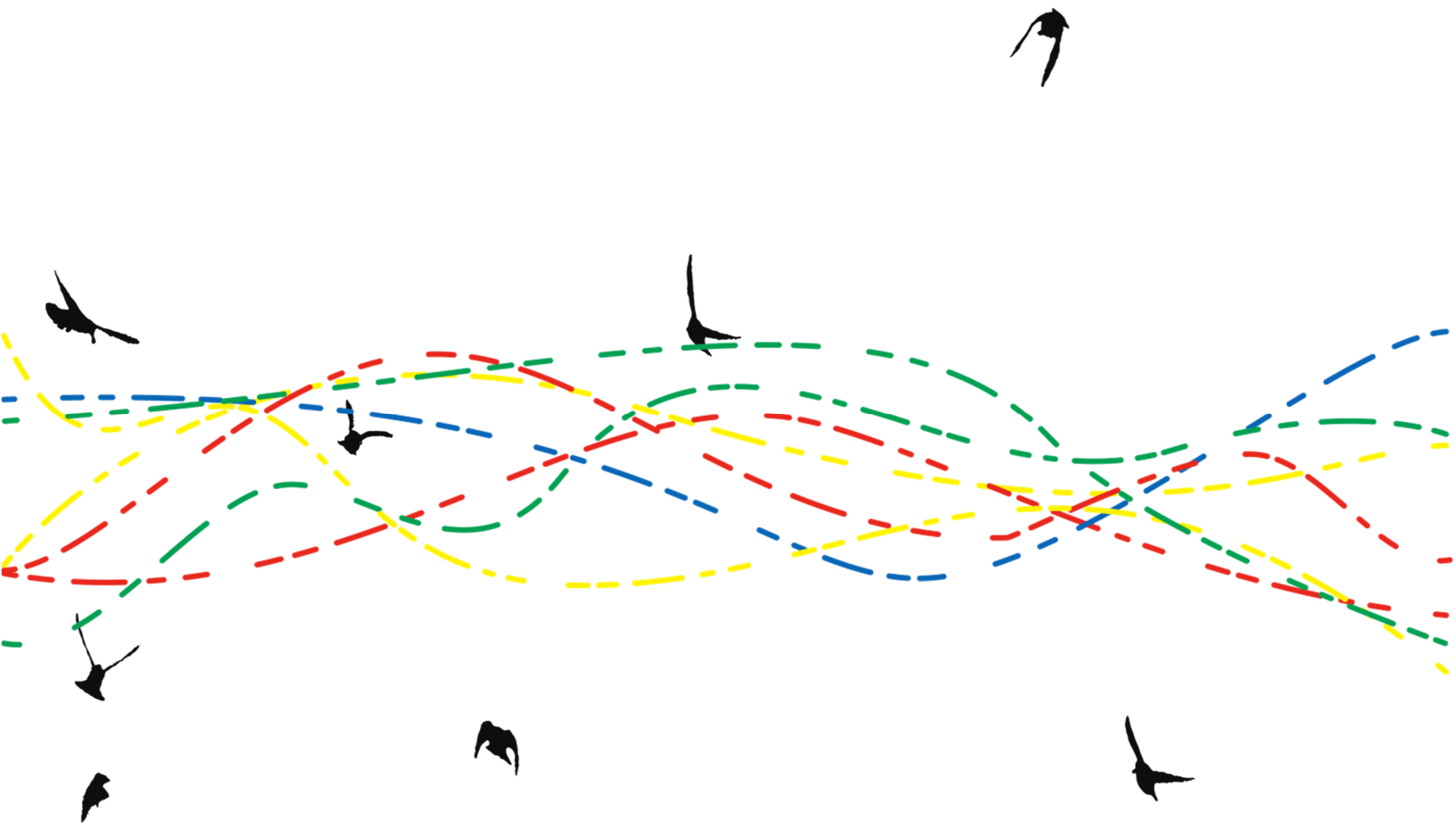


Table of Contents

History description

Version	Date	Author	Comments
2.1.0.0	March 2013	René Somer	Initial PostgreSQL version.
2.1.0.2	June 2014	René Somer	Rework after reviews.
2.1.1.0	Aug 2014	René Somer	Updated for latest DB changes
2.1.2.1	Sep 2014	René Somer	Updated for latest DB changes
2.1.6.5	Jul 2015	René Somer	Updated for latest DB changes and disabling snapshot use.
2.1.6.6	January 2016	René Somer	Updated to match rel 2.1.6.
2.1.7.0	August 2016	René Somer	Updated to match rel 2.1.7 (db-rev.9)

1 Introduction

1.1 Purpose

The purpose of this database user manual document is twofold. The first is to give a basic view on the structure of the database itself. It shortly describes all values in the tables and shows the relation between these tables to help users to get a basic understanding of the Robin database. The second purpose of the document is to explain the basic tooling for handling the database data.

1.2 Default database schema

The radar system uses the PostgreSQL database to store all necessary data. By default the database contains the following schemas.

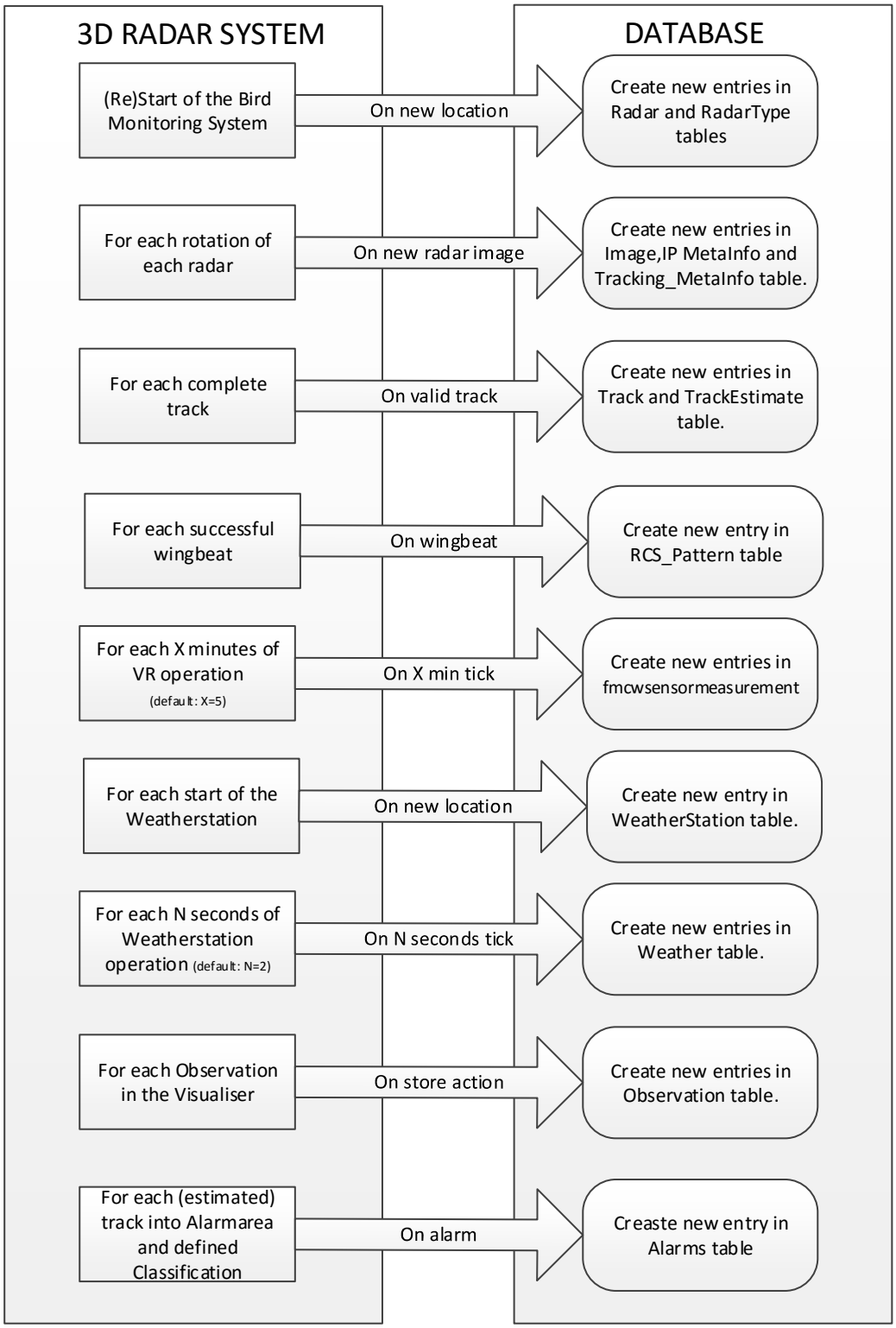
- Public schema, this schema contains the tables in which the Robin 3D system insert all data. This is the main schema for the bird data.
- Config schema, this schema contains the configuration of the system.
- Spatial schema, this schema may contain overlay areas for the BirdMonitor.

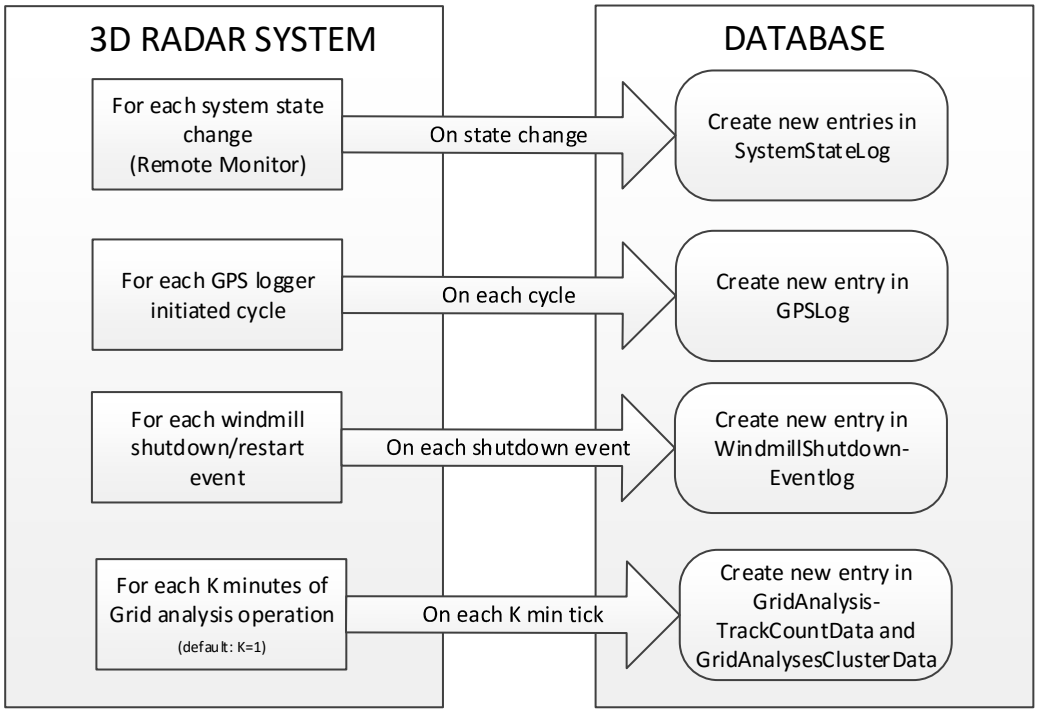
This document handles only the tables which may be of interest of the user who does the bird data analyses.

2 Interface description

2.1 Main database entry events

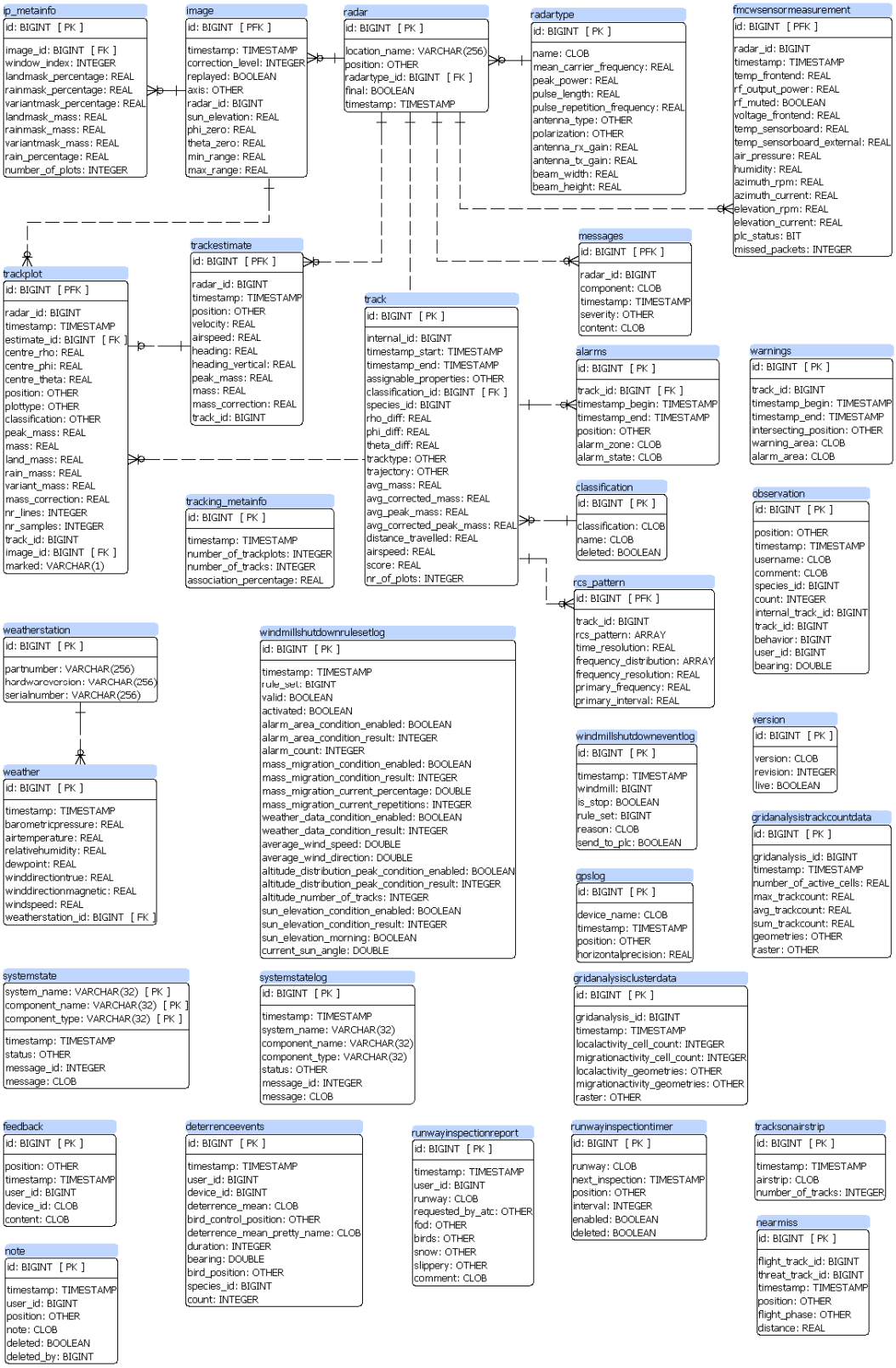
The following events from the 3D radar system cause entries in the database tables.




3D RADAR SYSTEM
DATABASE
For each system state change (Remote Monitor)
On state change
Create new entries in SystemStateLog
For each GPS logger initiated cycle
On each cycle
Create new entry in GPSLog
For each windmill shutdown/restart event
On each shutdown event
Create new entry in WindmillShutdown-Eventlog
For each K minutes of Grid analysis operation
(default: K=1)
On each K min tick
Create new entry in GridAnalysis-TrackCountData and GridAnalysesClusterData

2.2 Database relational schemas (Public)

This shows the tables of the database schema public. These tables are the main data-tables for system and track analyses.



2.2.1 Radar

This table contains information about the radar(s) including its name, location and a reference to its setting in the RadarType table. Typically this data is inserted upon each restart of the Robin Bird Monitoring system.

Radar		
ID	*bigint*	PK
LOCATION_NAME	*character varying(256)*	Name of the radar location
POSITION	*Geometry*	Position Latitude / Longitude / Altitude
RADARTYPE_ID	*bigint*	Reference to the radar type
FINAL	*Boolean*	If no more data may be added referencing this radar
TIMESTAMP	*timestamp without time zone*	Timestamp of creation

2.2.2 RadarType

The RadarType table contains information about a particular configuration of the radar. Like the Radar table, data entries in this table are inserted upon each restart of the Robin Bird Monitoring system.

RadarType		
ID	*bigint*	PK (Primary Key)
NAME	*text*	Name (= specification filename)
MEAN_CARRIER_FREQUENCY	*real*	Mean carrier frequency [Ghz]
PEAK_POWER	*real*	Peak power [W]
PULSE_LENGTH	*real*	Pulse length [s]
PULSE_REPETITION_FREQUENCY	*real*	Pulse repetition frequency [Hz]
ANTENNA_TYPE	*t_antenna|_type('slotted','parabolic','helical')*	Antenna type
POLARIZATION	*t_polarization('horizontal','vertical','circular')*	Polarization
ANTENNA_RX_GAIN	*real*	Antenna RX gain [dB] (Receiving)
ANTENNA_TX_GAIN	*real*	Antenna TX gain [dB] (Transmiting)
BEAM_WIDTH	*real*	Beam width [rad]
BEAM_HEIGHT	*real*	Beam height [rad]

2.2.3 Image

The Image table contains information about each radar image acquired by the system. This information is inserted into the database for each rotation of the particular radar. E.g.: For a 20 RPM VR radar the system will add twenty images per minute to the database.

Image		
ID	*bigint*	PK
TIMESTAMP	*timestamp without time zone*	Time the image was recorded [UTC]
CORRECTION_LEVEL	*Integer*	Current noise level measured by radar (original 15 bit value)
REPLAYED	*Boolean*	Indicates if the image was replayed by the file simulation mode
AXIS	*t_axis('phi',theta,'time')*	Horizontal or Vertical or time
RADAR_ID	*bigint*	Reference to radar that recorded the image
SUN_ELEVATION	*real*	Elevation of the sun at 'timestamp' [rad]
PHI_ZERO	*real*	Bearing of the image [rad]
THETA_ZERO	*real*	Elevation of the image [rad]
MIN_RANGE	*real*	Start range of the image [m]
MAX_RANGE	*real*	End range of the image [m]

2.2.4 IP_MetaInfo

The IP_MetaInfo table contains Meta information about each radar image acquired by the system.

IP_MetaInfo		
ID	*bigint*	PK
IMAGE_ID	*bigint*	reference to Image this record belongs to (see par 2.2.3)
WINDOW_INDEX	*integer*	subwindow index (not used)
LANDMASK_PERCENTAGE	*real*	percentage of total area covered by landmask [0-1]
RAINMASK_ PERCENTAGE	*real*	percencage of total area covered by rainmask [0-1]
VARIANTMASK_ PERCENTAGE	*real*	percencage of total area covered by variantmask [0-1]
LANDMASK_MASS	*real*	'mass' of the land image relative to the original image (landmass/image mass) [fraction]
RAINMASK_MASS	*real*	'mass' of the rain image relative to the original image (rainmass/image mass) [fraction]
VARIANTMASK_MASS	*real*	'mass' of the variant image relative to the original image (variantmass/image mass) [fraction]
RAIN_PERCENTAGE	*real*	Percentage of rain [0…1]
NUMBER_OF_PLOTS	*integer*	number of raw plots

2.2.5 Messages

The Messages table contains information about special messages sent by one of the radar systems.

Messages		
ID	*bigint*	PK
RADAR_ID	*bigint*	References the radar this message comes from
COMPONENT	*text*	References the component this message comes from
TIMESTAMP	*timestamp without time zone*	The time of this message
SEVERITY	*t_severity('fatal','error','warning','info','debug')*	Severity of this message
CONTENT	*text*	Content of the message

2.2.6 FMCWsensormeasurement

The FMCWsensormeasurement table contains logging data from the internal sensors of the VR (FMCW) radar-head.

FMCWsensormeasurement		
ID	*bigint*	PK
RADAR_ID	*bigint*	References the radar this log comes from
TIMESTAMP	*timestamp without time zone*	Time of the log
TEMP_FRONTEND	*real*	Temperature of the front-end [C]
RF_OUTPUT_POWER	*real*	Output power of the front-end [dBm]
RF_MUTED	*boolean*	Mute state of the front-end (shows a "1" if the RF is muted)
VOLTAGE_FRONTEND	*real*	Voltage of the front-end [V]
TEMP_SENSORBOARD	*real*	Temperature of the sensor board [C]
TEMP_SENSORBOARD_EXTERNAL	*real*	Temperature of the external sensor of the sensor board [C]
AIR_PRESSURE	*real*	Air pressure inside the unit [kPa]
HUMIDITY	*real*	Humidity inside the unit [%]
AZIMUTH_RPM	*real*	Angular velocity of the azimuth motor [RPM]
AZIMUTH_CURRENT	*real*	Current of the azimuth motor [A]
ELEVATION_RPM	*real*	Angular velocity of the elevation motor [RPM]
ELEVATION_CURRENT	*real*	Current of the elevation motor [A]
PLC_STATUS	*bit(1)*	Status of the PLC
MISSED_PACKETS	*integer*	Network parameter, Number of missed packets in the measuring interval

2.2.7 **Track**

This table contains information about the individual tracks identified by the Robin tracking system.

Track		
ID	*bigint*	PK
INTERNAL_ID	*bigint*	internal robin track id (used for debugging)
TIMESTAMP_START	*timestamp without time zone*	Start time of the track
TIMESTAMP_END	*timestamp without time zone*	Time the track was lost or dropped
ASSIGNABLE_PROPERTIES	*t_assignableproperty[]*	Assignable track properties (e.g. propelled, large, adsb)
CLASSIFICATION_ID	*bigint*	classification
SPECIES_ID	*bigint*	Reference to the classified bird species
RHO_DIFF	*real*	Straight line distance travelled (polar notation of vector from first to last object)
PHI_DIFF	*real*	Bearing of straight line distance travelled (polar notation of vector from first to last object)
THETA_DIFF	*real*	Elevation of straight line distance travelled (polar notation of vector from first to last object)
TRACK_TYPE	*tracktype*	Dimension measured of track. RaAz (=HR), RaEl (=VR), RaAzEl (=HR+VR combined), Ra (FMCW-Staring).
TRAJECTORY	*geometry*	Polyline of all estimates in the track
AVG_MASS	*real*	average mass of all plots,
AVG_CORRECTED_MASS	*real*	Range corrected average mass of all objects [dB]
AVG_PEAK_MASS	*real*	average peak mass from all plots,
AVG_CORRECTED_PEAK_MASS	*real*	Range corrected average peak mass of all objects [dB]
DISTANCE_TRAVELLED	*real*	total distance travelled along the track path
AIRSPEED	*real*	Average airspeed [m/s]
SCORE	*real*	Score of the tracking algorithm (0-1)
NR_OF_PLOTS	*Integer*	number of plots [#]

2.2.8 **TrackPlot**

The table TrackPlot is not used by the system, but it is kept here for possible backwards compatibility issues for previous releases.

TrackPlot		
ID	*Bigint*	PK
RADAR_ID	*Bigint*	Radar reference that contributed this information
TIMESTAMP	*timestamp without time zone*	Time of the log
ESTIMATE_ID	*bigint*	Reference to the applicable track estimate
CENTRE_RHO	*Real*	Detection range of object center of gravity (radar centered)
CENTRE_PHI	*real*	Detection bearing of object center of gravity (radar centered)
CENTRE_THETA	*real*	Detection elevation of object center of gravity (radar centered)
POSITION	*geometry*	Detection latitude / longitude / altitude of plot centre of gravity
PLOTTYPE	*t_plottype*	Dimension of the plot (range/azimuth (HR), range/elevation (VR))
CLASSIFICATION	*t_plotclassification*	Type of plot (part of blob, large, weak etc.)
PEAK_MASS	*real*	Peak mass of object
MASS	*real*	Estimated mass of object [value * samples] (probably not valid)
LAND_MASS	*real*	Mass of land mask beneath the plot
RAIN_MASS	*real*	Mass of rain mask beneath the plot
VARIANT_MASS	*real*	Mass of variant mask beneath the plot
MASS_CORRECTION	*real*	Range correction for mass numbers [dB]
NR_LINES	*integer*	Number of lines
NR_SAMPLES	*integer*	Number of samples
TRACK_ID	*bigint*	References the track the plot belongs to
IMAGE_ID	*bigint*	References the image the plot was identified in
MARKED	*character varying(1)*	Classification of plot given by the plot algorithm (above land, free, etc.)

Note: Mass is the SNR ratio of the total object. All the cells covered by the objects are summated in the linear domain (voltage) and the sum of their contribution is converted to dB. This value is written to the database.

2.2.9 TrackEstimate

The TrackEstimate is the new "TrackObject" table and contains all the individual estimates of a particular track. The combination of Track and TrackEstimate completely describe a flight path of an object or a group of objects (e.g. birds) detected by the system.

TrackEstimate		
ID	*bigint*	PK
RADAR_ID	*real*	Radar reference that contributed this information
TIMESTAMP	*real*	Time of estimate
POSITION	*real*	Tracker estimate of the latitude / longitude / altitude
VELOCITY	*geometry*	Tracker estimate of the track velocity [m/s]
AIRSPEED	*real*	Wind corrected velocity [m/s]
HEADING	*real*	Tracker estimate of the track bearing [rad]
HEADING_VERTICAL	*real*	Tracker estimate of the track elevation bearing [rad]
PEAK_MASS	*real*	Peak mass value of plot
MASS	*real*	Estimated mass of estimate [value * samples]
MASS_CORRECTION	*integer*	Range correction for mass numbers [dB]
TRACK_ID	*real*	References the track the estimate belongs to

2.2.10 Tracking_MetaInfo

The Tracking_MetaInfo table contains Meta information about each tracking iteration acquired by the system.

Tracking_MetaInfo		
ID	*bigint*	PK
TIMESTAMP	*timestamp without time zone*	Date and hour of the entry
NUMBER_OF_TRACKPLOTS	*integer*	number of associated plots
NUMBER_OF_TRACKS	*integer*	Total number of tracks in the track manager
ASSOCIATION_PERCENTAGE	*real*	Percentage of image ob plot ejcts which are associated to tracks [0-1]

2.2.11 Alarms

The Alarms table contains the alarms caused by objects in an alarm area.

Alarms		
ID	*bigint*	PK
TRACK_ID	*bigin*	Reference to track that produced alarm
TIMESTAMP_BEGIN	*timestamp without time zone*	Begin Time of alarm (UTC)
TIMESTAMP_END	*timestamp without time zone*	End Time of alarm (UTC)
POSITION	*Geometry*	Point of entry of the the alarm zone (lon/lat/alt)
ALARM_ZONE	*text*	Name of the alarm zone
ALARM_STATE	*text*	State of the alarm (No assessment, No alarm, Alarm)

2.2.12 Classification

The Classification table contains the text strings for the possible classification number used in the Track table.

Classification		
ID	*bigint*	PK
CLASSIFICATION	*text*	Classification identifier
NAME	*text*	Classification display name
DELETED	*bigint*	If true, this classification is hidden from the user interfaces.
DEFAULT_COLOR	*text*	The default classification color value for if an application has not set his own.

2.2.13 RCS_Pattern

The RCS_Pattern table contains the RCS data as measured by the VR.

RCS_Pattern		
ID	*bigint*	PK
TRACK_ID	*bigint*	References the track this RCS pattern belongs to
RCS_PATTERN	*real[]*	The RCS pattern
TIME_RESOLUTION	*real*	Time resolution of the samples in the RCS pattern [s]
FREQUENCY_DISTRIBUTION	*real[]*	Frequency distribution of the RCS pattern (FFT)
FREQUENCY_RESOLUTION	*real*	Resolution of the frequency distribution [Hz/sample]
PRIMARY_FREQUENCY	*real*	The primary frequency of the distribution [Hz]
PRIMARY_INTERVAL	*real*	The primary interval time of the distribution [s]

2.2.14 Observation

The Observation table contains information about the observations made by the user.

Observation		
ID	*bigint*	PK
POSITION	*geometry*	Latitude/Longitude/Altitude
TIMESTAMP	*timestamp without time zone*	time
USERNAME	*text*	observer name
COMMENT	*text*	comment
SPECIES_ID	*bigint*	references the species if it was observed
COUNT	*integer*	number of birds counted
INTERNAL_TRACK_ID	*bigint*	references the internal track_id of a track
TRACK_ID	*bigint*	references the internal track_id of a track
BEHAVIOR	*bigint*	reference behavior
USER_ID	*bigint*	reference the user
BEARING	*double precision*	direction

2.2.15 SystemState

The SystemState table is used to show the current state of the system components. It is filled by the Remote Monitor service application.

SystemState		
TIMESTAMP	*timestamp without time zone*	Timestamp of log event
SYSTEM_NAME	*character varying (32)*	Name of system
COMPONENT_NAME	*character varying (32)*	Component name
COMPONENT_TYPE	*character varying(32)*	Component type
STATUS	*T_status*	Status of component (OK,WARNING, ERROR, CRITICAL or MAINTENANCE)
MESSAGE_ID	*integer*	Unique identifier of the message
MESSAGE	*text*	Message on error or changes

2.2.16 SystemStateLog

The SystemState table is used to log the state of the system components. It is filled by the Remote Monitor service application.

SystemStateLog		
ID	*bigint*	PK
TIMESTAMP	*timestamp without time zone*	Timestamp of log event
SYSTEM_NAME	*character varying (32)*	Name of the system
COMPONENT_NAME	*character varying (32)*	Component name
COMPONENT_TYPE	*character varying (32)*	Component type
STATUS	*T_status*	Status of component (OK,WARNING, ERROR, CRITICAL or MAINTENANCE)
MESSAGE_ID	*integer*	Unique identifier of the message
MESSAGE	*text*	Message on error or changes

2.2.17 Weather

The Weather table contains the data measured by the weather station.

Weather		
ID	*bigint*	PK
TIMESTAMP	*timestamp without time zone*	timestamp
BAROMETRICPRESSURE	*real*	Barometric pressure
AIRTEMPERATURE	*real*	Air temperature
RELATIVEHUMIDITY	*real*	Relative humidity
DEWPOINT	*real*	Dew point (if available)
WINDDIRECTIONTRUE	*real*	True wind direction (compensated for magnetic declination)
WINDDIRECTIONMAGNETIC	*real*	Wind direction w.r.t. magnetic north
WINDSPEED	*real*	Wind speed
WEATHERSTATION_ID	*integer*	Reference to weather station

2.2.18 WeatherStation

The WeatherStation table stores information about the weather station itself.

WeatherStation		
ID	*bigint*	PK
PARTNUMBER	*character varying(256)*	Manufacturer Part number
HARDWAREVERSION	*character varying(256)*	Manufacturer Hardware version
SERIALNUMBER	*character varying(256)*	Manufacturer Serial number

2.2.19 GPSlog

The GPSlog table logs the gps data from an optional gpslogging device.
This table is only filled in when an extra GPS logging application is installed on the system, otherwise it is empty.

GPSlog		
ID	*bigint*	PK
DEVICE_NAME	*text*	Name of the GPS device
TIMESTAMP	*timestamp without time zone*	The time of entry
POSITION	*geometry*	Recorded position
HORIZONTALPRECISION	*real*	Precision accurancy [m]
USER_ID	*bigint*	Current user

2.2.20 WindmillShutdownEventlog

The Windmill shutdown event-log table stores log information about the shutdown activities of the configured windmills.
This table is only filled in when a windmill-shutdown application is installed on the system, otherwise it is empty.

WindmillShutdownEventlog		
ID	*Bigint*	PK
TIMESTAMP	timestamp without time zone	The time of entry
WINDMILL	bigint	Windmill that is stopped/started
IS_STOP	*boolean*	Did we send stop or start command
RULE_SET	bigint	The windmill shutdown rule set that is active
REASON	*pg_catalog, "default"*	Reason the windmill is started or stopped
SEND_TO_PLC	*Boolean*	Whether the signal is send to the plc or not

2.2.21 WindmillShutdownRulesetLog

The WindmillShutdownRulesetLog is the log for the Bird Analysis windmill shutdown rule set.

WindmillShutdownRulesetLog		
ID	*bigserial*	Primary key [PK]
TIMESTAMP	timestamp without time zone	Timestamp
RULE_SET	*bigint*	References the rule set this log entry belongs to
VALID	*boolean*	Whether all enabled conditions of the rule set are result
ACTIVATED	*boolean*	The rule set is valid for at least the activation delay
ALARM_AREA_CONDITION_ENABLED	*boolean*	Is alarm area condition enabled
ALARM_AREA_CONDITION_RESULT	*integer*	Is alarm area condition result
ALARM_COUNT	*integer*	Current Alarm count
MASS_MIGRATION_CONDITION_ENABLED	*boolean*	Is mass migration condition enabled
MASS_MIGRATION_CONDITION_RESULT	*integer*	Is mass migration condition result
MASS_MIGRATION_CURRENT_PERCENTAGE	*double precision*	The current percentage of mass migration cells
MASS_MIGRATION_CURRENT_REPETITIONS	*integer*	The number of repetitions that the percentage migration cells is above the threshold
WEATHER_DATA_CONDITION_ENABLED	*Boolean*	Is Weather data condition enabled
WEATHER_DATA_CONDITION_RESULT	*integer*	Is Weather data condition result
AVERAGE_WIND_SPEED	*double precision*	Average wind speed [m]
AVERAGE_WIND_DIRECTION	*double precision*	Average wind direction [rad]
ALTITUDE_DISTRIBUTION_PEAK_CONDITION_ENABLED	*Boolean*	Is altitude distribution peak condition enabled
ALTITUDE_DISTRIBUTION_PEAK_CONDITION_RESULT	*integer*	Is altitude distribution peak condition result
ALTITUDE_NUMBER_OF_TRACKS	*integer*	Number of tracks
SUN_ELEVATION_CONDITION_ENABLED	*Boolean*	Sun elevation condition enabled
SUN_ELEVATION_CONDITION_RESULT	*integer*	Sun elevation condition result
SUN_ELEVATION_MORNING	*Boolean*	Is the sun angle calculated in the morning or afternoon
CURRENT_SUN_ANGLE	*double precision*	Current sun angle [rad]

2.2.22 GridAnalysisClusterData

The Gridanalysisclusterdata table stores the data for grid analysis.

GridAnalysisClusterData		
ID	*bigint*	PK
GRIDANALYSIS_ID	*bigint*	Identification of the Grid Analysis
TIMESTAMP	timestamp without time zone	The time of entry
LOCALACTIVITY_CELL_COUNT	*integer*	Number of grid cells with local activity
MIGRATIONACTIVITY_CELL_COUNT	*Integer*	Number of grid cells with migration activity
LOCALACTIVITY_GEOMETRIES	*geometry*	Polygon representation of local activity clusters
MIGRATIONACTIVITY_GEOMETRIES	*geometry*	Polygon representation of migration activity clusters.
RASTER	*raster*	Raster representation of grid

2.2.23 GridAnalysisTrackCountData

The Gridanalysistrackcountdata table stores the track count data used for grid analysis.

GridAnalysisTrackCountData		
ID	*bigint*	PK
GRIDANALYSIS_ID	*bigint*	FP to corresponding grid analysis
TIMESTAMP	timestamp without time zone	The time of entry
NUMBER_OF_ACTIVE_CELLS	*real*	Active cell count
MAX_TRACKCOUNT	*real*	Max of all grid cells
AVG_TRACKCOUNT	*real*	Avg of active grid cells
SUM_TRACKCOUNT	*real*	Sum of all grid cells
GEOMETRIES	*geometry*	Polygon representation of grid
RASTER	*raster*	Raster representation of grid.

2.2.24 Version

The Version table contains data about the database version.

Version		
ID	*bigint*	PK
VERSION	*text*	Current database version
REVISION	*integer*	Current database update revision
LIVE	*boolean*	true if this is a live database. DB management might use this and also a live database will not be used for replay data.

2.2.25 FeedBack

The FeedBack table contains the comments (feedback) of the mobile viewer user.

FeedBack		
ID	*bigint*	PK
POSITION	*geometry*	Position
TIMESTAMP	*timestamp without time zone*	The time of this feedback message
USER_ID	*bigint*	The user this feedback message comes from (reference to user table)
DEVICE_ID	*text*	The device this feedback message comes from
CONTENT	*text*	Content of the feedback message

2.2.26 DeterrenceEvents

The DeterrenceEvents table contains the logging of all executed deterrence activities by the mobile viewer user.

DeterrenceEvents		
ID	*bigint*	PK
TIMESTAMP	*timestamp without time zone*	Timestamp
USER_ID	*bigint*	User that triggered the deterrence device
DEVICE_ID	*bigint*	Deterrence device that performed the deterrence
DETERRENCE_MEAN	*text*	Deterrence mean that was used
BIRD_CONTROL_POSITION	*geometry*	Position of user triggering the event
DETERRENCE_MEAN_PRETTY_NAME	*text*	Deterrence mean pretty name that was used
DURATION	*integer*	Duration
BEARING	*double precision*	Bearing of the birds based on the BIRD_POSITION
BIRD_POSITION	*geometry*	Position of the birds that are chased away
SPECIES_ID	*bigint*	Species that are chased away
COUNT	*integer*	Number of species that were spotted

2.2.27 NearMiss

The NearMiss table is used as a Risk Indicator and shows the near misses between aircraft and other track.

NearMiss		
ID	*bigserial*	PK
FLIGHT_TRACK_ID	*bigint*	Reference to the track of the aircraft
THREAT_TRACK_ID	*bigint*	Reference to track that produced the nearmiss
TIMESTAMP	*timestamp without time zone*	Time of nearmiss (UTC)
POSITION	*geometry*	Position of possible intersection
FLIGHT_PHASE	t_flightphase	The phase in which the plane was acting.(e.g. Landing, Takeoff)
DISTANCE	real	Closest measured distance between aircraft and threat
AIRCRAFT_SPEED	real	Speed of the aircraft

2.2.28 Note

The Note table contains the notes (position and textual comment) created in the mobile viewer.

Note		
ID	*bigserial*	PK
TIMESTAMP	*timestamp without time zone*	Timestamp of entry
USER_ID	*bigint*	The user who created this note
POSITION	*geometry*	The location of where this note is made
NOTE	*text*	The text
DELETED	*Boolean*	The note is deleted and not visible
DELETED_BY	*bigint*	User that deleted the note

2.2.29 RunwayInspectionReport

The RunwayInspectionReport table contains the data entered by the user during runway inspection.

RunwayInspectionReport		
ID	*bigserial*	PK
TIMESTAMP	*timestamp without time zone*	Timestamp of entry
USER_ID	*bigint*	The user who created this report
RUNWAY	*text*	The runway
REQUESTED_BY_ATC	*t_runwayinspectionresult*	Is the inspection requested by ATC
FOD	*t_runwayinspectionresult*	Is FOD detected
BIRDS	*t_runwayinspectionresult*	Are there birds
SNOW	*t_runwayinspectionresult*	Is there snow on the runway
SLIPPERY	*t_runwayinspectionresult*	Is the runway slippery
COMMENT	*text*	Comment

2.2.30 RunwayInspectionTimer

The RunwayInspectionTimer table contains the date/time of when a new runway inspection is needed for the particular runway.

FeedBack		
ID	*bigserial*	Primary key
RUNWAY	*text*	Name of the runway
NEXT_INSPECTION	*timestamp without time zone*	Time the next inspection should be executed
POSITION	*geometry*	The location of the runway
INTERVAL	*integer*	Interval between two inspections
ENABLED	*Boolean*	If true, this timer is active
DELETED	*Boolean*	

2.2.31 TracksOnAirstrip

The TracksOnAirstrip table is used as a Risk Indicator and shows number of tracks on an airstrip.

TracksOnAirstrip		
ID	*bigserial*	PK
TIMESTAMP	*timestamp without time zone*	Time of measurement (UTC)
AIRSTRIP	*text*	The airstrip
NUMBER_OF_TRACKS	*integer*	Number of tracks that are above the airstrip

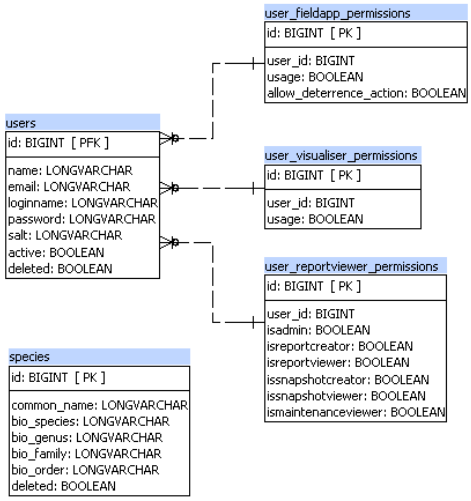
2.2.32 **Warnings**

The Warnings table contains the warnings caused by objects in a warning area.

Warnings		
ID	*bigserial*	PK
TRACK_ID	*bigint*	Reference to track that produced warning
TIMESTAMP_BEGIN	*timestamp without time zone*	Begin Time of warning (UTC)
TIMESTAMP_END	*timestamp without time zone*	End Time of warning (UTC)
INTERSECTING_POSITION	*geometry*	Expected point of entry of the alarm zone (lon/lat/alt)
WARNING_AREA	*text*	Name of the warning area
ALARM_AREA	*text*	Name of the alarm area

2.3 Database relational schema (config)

The config schema stores the Robin 3D configuration system data. This paragraph only describes the user applicable tables.



2.3.1 **Users**

The Users table contains the account data of all users.

Users		
ID	*bigint*	PK
NAME	*text*	User display name
EMAIL	*text*	User email address
LOGINNAME	*text*	User login name
PASSWORD	*text*	Family name Encrypted password
SALT	*text*	Encryption salt
ACTIVE	*Boolean*	This column indicates whether this row should be displayed in the default GUI.
DELETED	*Boolean*	If true, the user is not displayed and not allowed to login

2.3.1.1 User_fieldapp_permissions

The user_fieldapp_permissions table contains the fieldapp (mobile viewer) permissions of the users.

User_fieldapp_permissions		
ID	*bigserial*	PK
USER_ID	*Bigint*	Reference to the user
USAGE	*Boolean*	If the user is allowed to use the FieldApp
ALLOW_DETERRENCE ACTION	*Boolean*	If the user is allowed to perform deterrence actions

2.3.1.2 User_reportviewer_permissions

The User_reportviewer_permissions table contains the report viewer permissions of the users

User_reportviewer_permissions		
ID	*bigserial*	PK
USER_ID	*Bigint*	Reference to the user
ISADMIN	*Boolean*	If the user is Report Viewer administrator
ISREPORTCREATOR	*Boolean*	If the user is allow to create reports
ISREPORTVIEWER	*Boolean*	If the user is allow to view reports
ISSNAPSHOTCREATOR	*Boolean*	If the user is allow to create snapshots (OBSOLETE)
ISSNAPSHOTVIEWER	*Boolean*	If the user is allow to view snapshots (OBSOLETE)
ISMAINTENANCEVIEWER	*Boolean*	If the user is allow to view the maintenance log

2.3.1.3 User_visualiser_permissions

The user_visualiser_permissions table contains the visualiser permissions of the users.

User_visualiser_permissions		
ID	*bigserial*	PK
USER_ID	*Bigint*	Reference to the user
USAGE	*Boolean*	If the user is allowed to use the Visualiser

2.3.2 Species

The Species table contains the data of the user defined bird-species. The user can create his own species and these are stored in this table. These species are shown in the visualiser observation tab.

Species		
ID	*bigint*	PK
COMMON_NAME	*text*	Common name
BIO_SPECIES	*text*	Species name
BIO_GENUS	*text*	Genus name
BIO_FAMILY	*text*	Family name
BIO_ORDER	*text*	Order name
DELETED	*boolean*	This column indicates whether this row should be displayed in the default GUI.

3 DBManagement Manual

With an ever growing database but without a database management tool, ultimately the hard disk will get full and the radar system may stop functioning. Additional a growing database will cause a degrading database performance when the database gets huge. The performance will degrade especially when inserting live data and even worse when extracting and/or searching data with sophisticated SQL scripts. Therefore a Database Management (DBM) tool is foreseen. This tool can be monitored by the ReportViewer. The manual for this tool can be found in the Robin Configuration user manual.

4 Database Tools and Tips

This chapter describes tools and tips to use the database itself. It describes the use of pgAdmin III Query Browser as far as needed to setup and maintain the radar system. It handles the setup of the database and provides various tips which can be useful for experienced users.

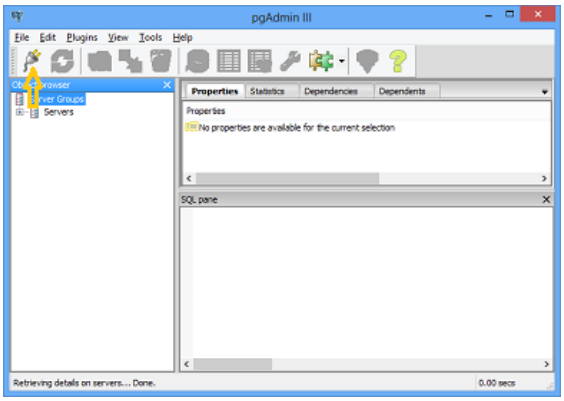
4.1 pgAdmin III Query Browser

The pgAdmin III Query Browser is a graphical interactive client for creating, executing, and optimizing PostgreSQL commands. This paragraph will explain the basic steps to start with PostgreSQL and pgAdmin III, for more detailed information about PostgreSQL see: http://www.postgresql.org/docs/current/static/index.html

4.1.1 Setup a connection

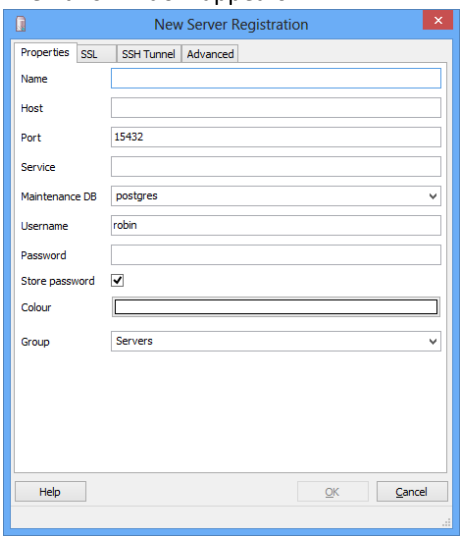
During the installation of the Robin Radar Software package the connection is already created, but for completeness the procedure to connect to a database in PostgreSQL is explained here. It is assumed that pgAdmin III is already installed (if not use the following link to download pgAmin III for Windows, http://www.postgresql.org/download/windows/).

Start pgAdmin III and see the following main window appear.

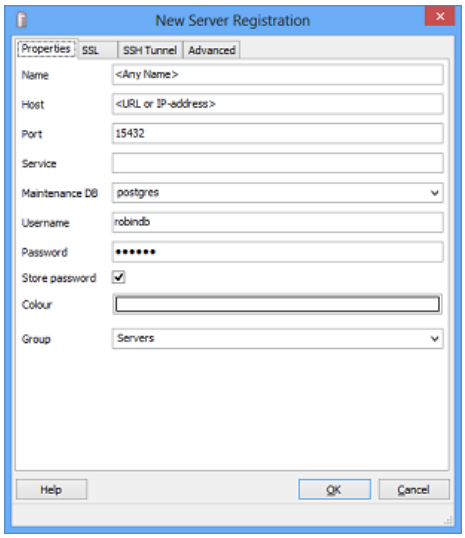


Press the connect button as shown by the arrow.

Then this window appears.



Fill in as shown here →



The following information is required to make a connection:

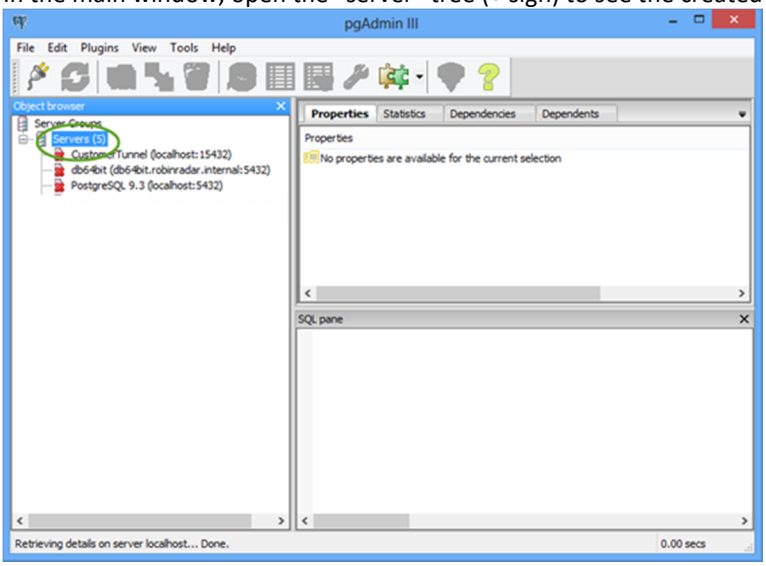
- *Name*: Any alias-name which makes it easy to remember the connected database
- *Host*: Network host name or IP of the computer on which the Robin database is running.
- *Port*: By default, the connection takes place via the TCP/IP protocol, where the Robin default port 15432 for PostgreSQL is used. (Standard PostgreSQL port is 5432)
- *Username* and P*assword*: The PostgreSQL user name and the associated password. Normally this is robindb/robindb.

Then press the OK button to create the connections.

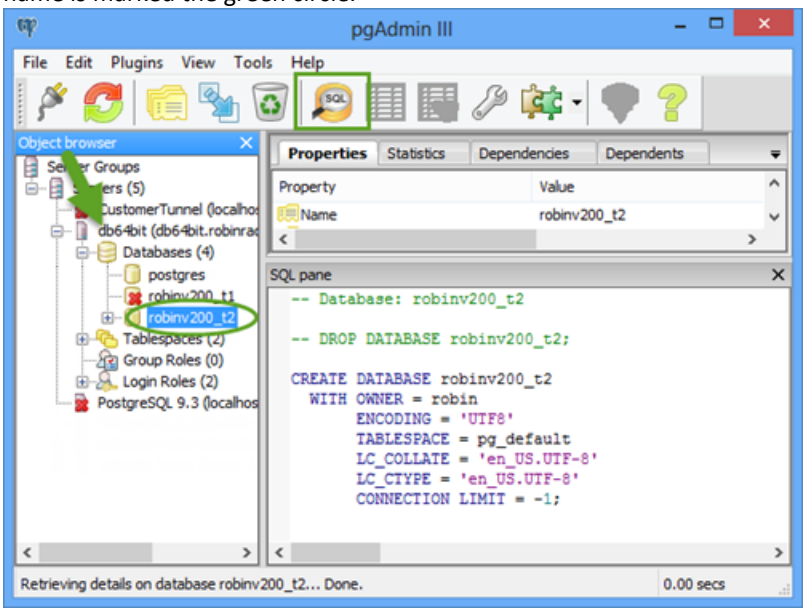
4.1.2 Start of the query browser

To start the browser, follow next steps:

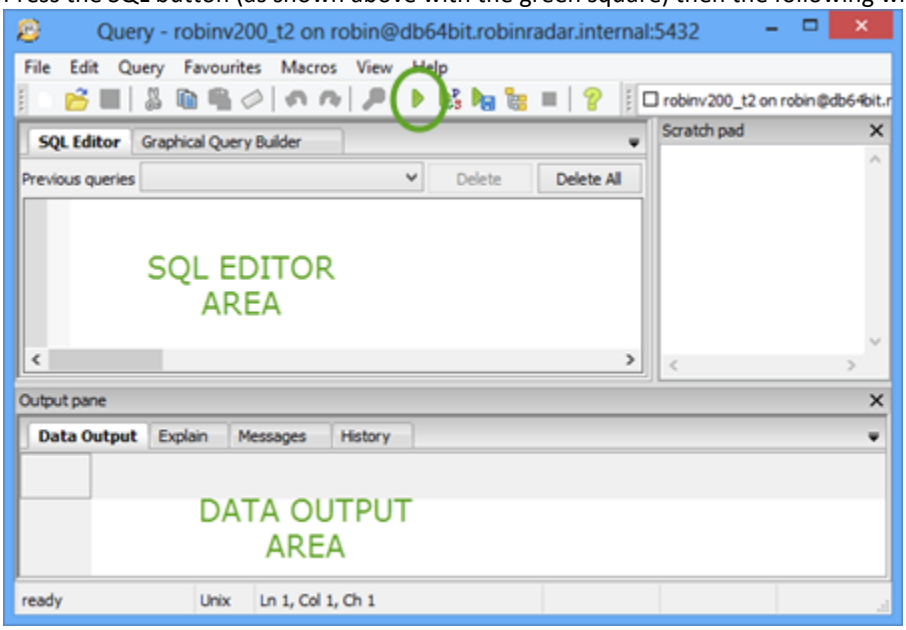
1) In the main window, open the “server” tree (+ sign) to see the created connection(s).



2) Then double click on the connected server (green arrow) and select your database name. Below in the example such a name is marked the green circle.



3) Press the SQL button (as shown above with the green square) then the following window appears.



It shows the two most important areas to use for basic SQL commands as described in this manual. The “SQL Editor Area” is where the SQL commands are entered and after execution the resulting table is shown in the “Data Output Area”. With the “Execute Query” Button (marked with the green circle) the SQL command is executed.

4.1.3 Practical example queries

In the SQL Editor Area the commands (SQL-queries) can be entered. Some useful example queries are shown here.
Change the <dbname> with the name of your database (schema) but leave the quotes!

To show the latest 20 images, enter:

- **SELECT * FROM public.Image order by ID desc limit 20;**

To show the latest 20 tracks with readable latitude/longitude position, enter:

- **SELECT * FROM public.Track order by ID desc Limit 20;**

To show the latest 20 observations (if any), enter:

- **SELECT *,st_astext(position) FROM public.Observation order by ID desc Limit 20;**

To show the radar names and the latitude/longitude position, enter:

- **SELECT id, location_name, st_astext(position) as "Lat Lon Z" FROM public.Radar order by ID desc Limit 20;**

To show the latest 20 logs of the FMCW status (for 3DFlex systems only), enter:

- **SELECT * FROM public.fmcwsensormeasurement order by ID desc Limit 20;**

To show the weather information, enter:

- **SELECT * FROM public.Weather order by ID desc Limit 20;**

Some remarks on the use of PostgreSQL:

1) Do not forget to use the ‘;’ at the end of the line.
2) To make sure which schema you use make sure that it is in front of the table, here the standard ‘public’ schema is used.

4.2 Data handling

The database as described in the ReportViewer user manual, is the database as installed on the radar system. The Robin Radar system can be delivered with the following options to retrieve and use the data.

1) Direct local database use
2) Database replication use

This chapter will show the (dis-)advantages of each option. And if not described in the applicable manual, more detail about the use of the options is explained here.

4.2.1 Direct local database use

"Direct local database use" provides the ability to directly use the database on the database server of the Radar system. The main user interface for this is the Robin Radar ReportViewer to generate data reports. Although the actual user interface of the ReportViewer application is described in the ReportViewer user manual, the usefulness of the ReportViewer is described here.

The ReportViewer is a quick tool, pre-installed on the system and easy to access from other locations (e.g. work-office) by use of the internet and a web browser. The ReportViewer has the advantage that it is running directly on the radar database computer, and queries are run directly on the database. This is also a disadvantage because: if either the database grows too big, or if the queries are done on large tables, it may become slow. Another issue, directly related to Database Management, is that if DBM is installed, the "archive" database is limited to e.g. three (default) months. Any report is limited to the maximum of the local archive database.

4.2.2 Database Replication use

The optimal use of the database is to have the Robin Database Replication (Scraper) option. Replication means that continuously (through the internet) a copy of the radar database is created on a separate database server. This server may be located at the customer office or in a (Robin) datacenter. The replicated data can contains more data, and can grow up to one year of data. If an Ubuntu-Linux replication server is used, the ReportViewer may also be installed there and all reports may be executed within the customer office. Besides the ReportViewer also GIS (Geographical Information System) application to analyze the replicated data are possible.

4.3 QGIS Basics

This paragraphs shows the basic steps to connect and use of QGIS with the Robin database (PostgreSQL).

4.3.1 Introduction

The Purpose for this hands-on is:

- To know how to get robin-geometry data from PostgreSQL and to make this visible in QGIS.
- To have an idea of the type of calculations/operations you can do with a real GIS (and not
- Google Earth) can do.

Why use QGIS:

- GIS software is common use for presentations of geodata.
- A lot of the Robin data is (or contains) geodata., e.g. road-adherence, overlays, tracks and observations.

What is GIS?

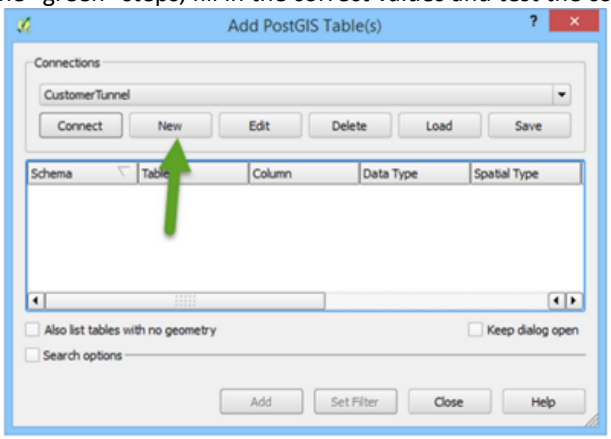
- Raster (.tif): 2D data
- Shape (.kml, .shp): points, line and polygons.
- PostGIS: interface for PostgreSQL database.
- Spatial query: query where you use an area as filter.
- Plugin: QGIS gets it power from the many plugins, say functions.
  The plugins are made by the QGIS community.

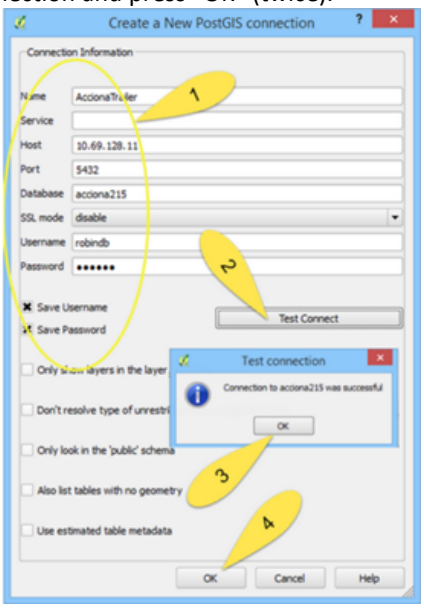
4.3.2 Installation of QGIS

Before continue you must install QGIS on a computer from which you have access (may be through the internet of course) to the applicable PostgreSQL database. Download from the internet http://www.qgis.org/en/site/forusers/download.html the QGIS version for your Operating system. Install it on your computer as it will be proposed by the QGIS Setup application.

4.3.3 Make a new DB connection

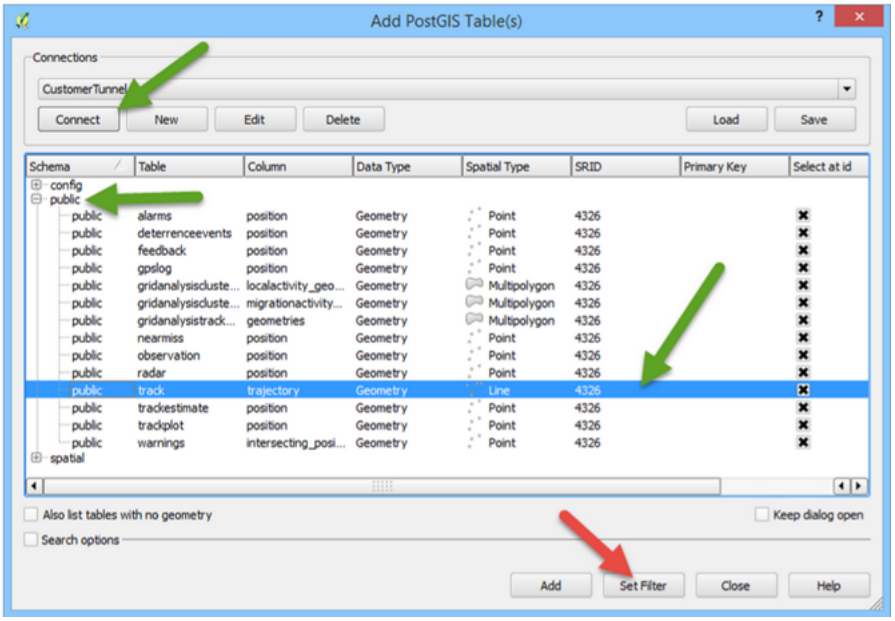
Just follow the “green” steps, fill in the correct values and test the connection and press “OK” (twice).



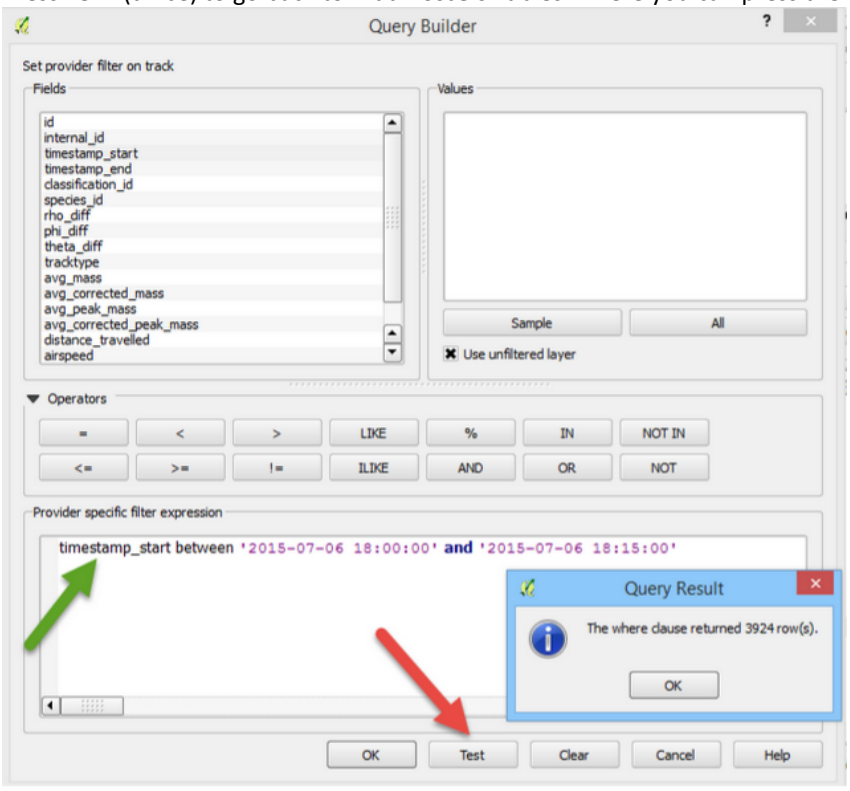


4.3.4 Connect and select

Follow the arrows:

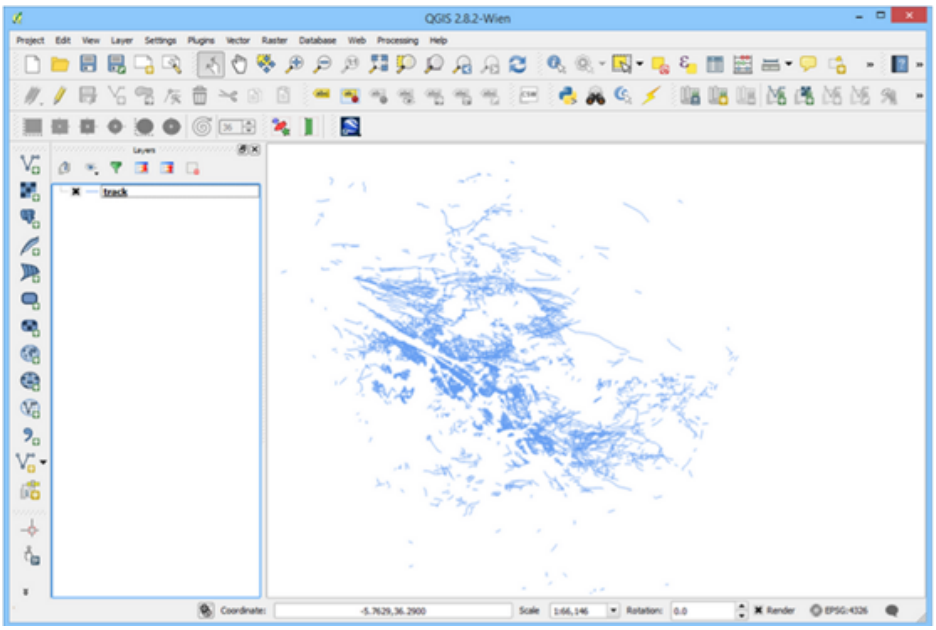


And create a filter as e.g. as shown here, use the “Test” button to check if you really have data.
Press “Ok” (twice) to go back to ‘Add PostGis Tables’ where you can press the “Add”. button (see above).



4.3.5 First QGIS result

When everything is done right then the first results are shown. Remark: QGIS is rather slow in showing data… it can take a few minutes.



4.4 QGIS Plugin setup

QGIS is based on the many plugins which are available from the GIS-community. As QGIS itself it is free software and created by different people for different purposes, be aware that plugins may be slow or not with the best… user interface.
Here how to install plugins: