ALTIR

AIRCRAFT MAINTENANCE PLANNING APPLICATION

v1.2.8

ADMINISTRATION GUIDE

Moscow
2019

Content

1. Terms and abbreviations

ROM – permanent storage device.

RAM – random access memory.

OS – operating system.

DB – database.

DBMS – database management system.

JDBC – Java-based applications standard of interaction with DBMS.

AC – aircraft.

ENG – engine.

APU – auxiliary power unit.

MAINT – maintenance.

MLG – main landing gear.

NLG – nose landing gear.

SN – since new.

2. Scope

This guide is intended for ALTIR administrators. Guide provides information about:
- application purpose and functionality,
- installing of application,
- setting up a connection to non-default database,
- application settings,
- user management,
- data import and export,
- data backup,
- troubleshooting.

3. Application purpose

ALTIR is designed to plan the maintenance of aircraft and their components in accordance with operating time and regulations of maintenance of airline. The application can be used by engineers of planning and control departments of aviation companies, as well as by staff of other companies serving the aircraft fleet, to monitor the status of the aircraft and to plan service.

ALTIR has ability to scale and is intended to small, medium-sized enterprises, local or regional offices of large companies with 5–20 users.

Application interface is designed as typical tables to facilitate transition from file-based aircraft maintenance to centralized. Functionality of the application allows you to perform batch creation and modification of components, forms and tasks on several aircraft.

4. Target functionality

- Aircraft, engine, APU and component flight time accounting;
- Component database;
- Component statuses;
- Wheel repairing database;
- Maintenance planning;
- Maintenance database;
- Aircraft, engine, APU and component flight time reports;
- Reports of performed maintenance;
- Selection of tasks, maintenance forms by resource residues and service life;
- Selection of components by resource residues and service life.

5. Base functionality

- User administration;
- View or edit mode;
- Logging data changes;
- Import / export to *.xls format;
- Data backup;
- UI localization support;
- Appearance customization.

6. System requirements

- Screen resolution of at least 1024*768 pixels.

- Windows, Linux or MacOS X operating system.

- Installed Java virtual machine with support of Java SE 8 (recommend Oracle JRE 8 or later).

- Free space in ROM: 500 Mb or more.

- Free space in RAM: 1 Gb or more.

7. Application installation

Full information about installation of ALTIR (including recommendations for preparing for installation, SQL queries for creating tables in non-default databases) is given in the *Installation guide*.

General information about installation

ALTIR distributes in compressed file that contains a set of files required for the functioning of the application:
- executable file
  (`ALTIR_*.exe` for Windiws,
  `ALTIR_*.jar` for Linux,
  `ALTIR_*.app` for MacOS X);
- default database file (`altir.sqlite3`).

Application launch:
1) unpack compressed file into the folder (local folder, folder on a mapped network drive, or a network folder with shared access);
2) set read and write permissions for that folder (administrator privileges is not required);
3) to use non-default database you should create `Altir_db_settings.ini` file in the same folder and edit it according to instruction (see par. 8);
4) run (execute) the executable file.

ALTIR contains following built-in user accounts (account data is stored at database):

- Administrator account
  Login: `admin`
  Password: `admin`

- Guest account
  Login: `guest`
  Password: `guest`

8. Database connection settings

ALTIR uses `altir.sqlite3` file in application folder as the default database.

To change database connection settings you should place (create) text configuration file in UTF-8 encoding named `Altir_db_settings.ini` in folder of executable file of application. This file must contain description of at least one connection according to following pattern:

```
[connection title]
type = database type
link = database address
name = database name
schema = database schema
login = login
password = password
```

where:

connection title – an arbitrary string that will identify the connection in application interface;

database type – one of strings: MYSQL, POSTGRESQL, SQLITE, FIREBIRD, matched to the type of DBMS;

database address – 1) DBMS host's network address and port number in the form //host:port/ (for MySQL, PostgreSQL, Firebird) or 2) absolute or relative address of the database file in file system (for SQLite);

database name – database name in DBMS (if necessary);

database schema – name of database schema in DBMS (if necessary);

login – name of DBMS user who has read and write access to specified database (if necessary);

password – password of specified user (if necessary);

Example:

```
[first database (mysql)]
type = MYSQL
link = //server1.local:3306/
name = altir_db
login = admin
password = adminpassword
```

or

```
[second database (postgresql)]
type = POSTGRESQL
link = //server2.local:5432/
name = altir_db_reserv
schema = public
login = admin
password = adminpassword
```
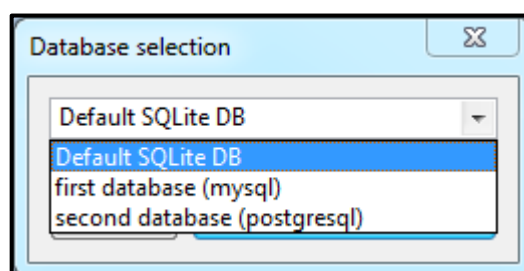
or

```
[third database (firebird)]
type = FIREBIRD
link = //server3.local:3050/
name = altir_db_reserv_2
login = admin
password = adminpassword
```

Following description corresponds to default database:

```
[Default SQLite DB]
type = SQLITE
link = ./altir.sqlite3
```
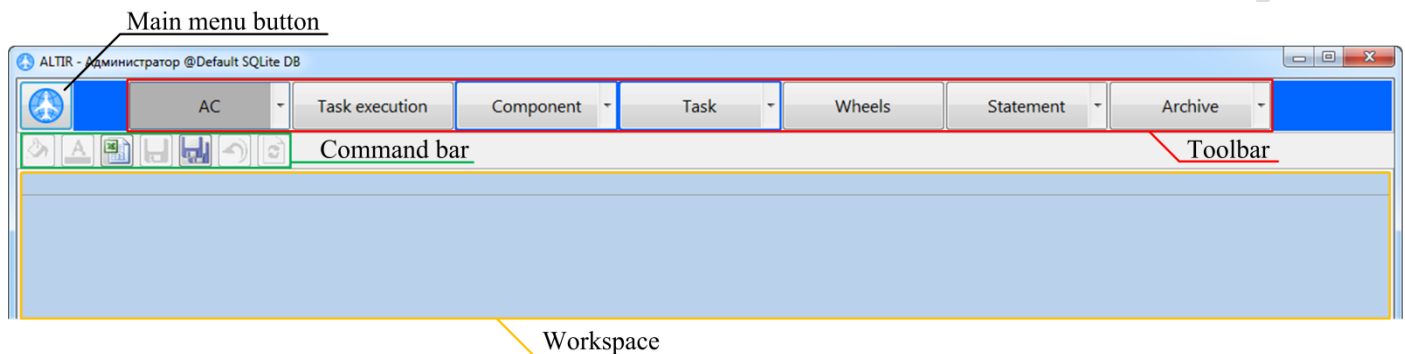
If file contains description of several connections, the application will prompt the user to select connection to be used.

9. Graphical user interface

Interface of main window contains:
- Main menu button;
- Toolbar to perform actions with aircraft, its components and tasks, to carry out their selection by service life, etc;
- Command bar;
- Workspace of selected tool.

Main menu items open windows that allow you to: 1) configure tables view, configure whole the application for different types of aircraft (item «Settings»), 2) manage user accounts (item «Users»), 3) view the protocol of user actions of creating, modifying and deleting of aircraft data (item «Protocol»).

Main menu items «Users» and «Protocol» are available only for accounts with administrator rights.
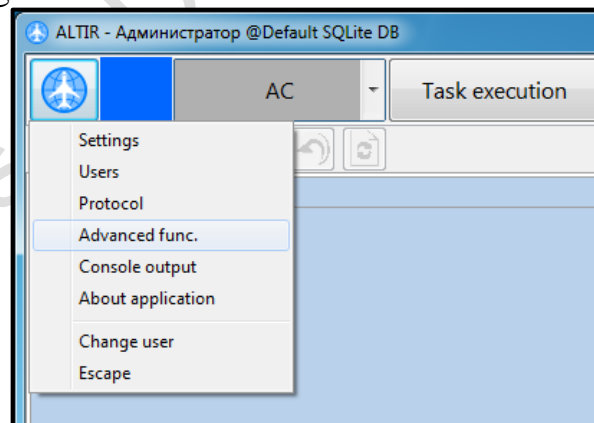
Toolbar contains seven buttons to invoke workspaces and tool menu. Pressing the button with name of tool activates its working area, pressing the arrow on the right side of button brings up tool menu. Tool button with active work area is highlighted in dark gray (only one work area can be active at once). «**Component**» and «**Task**» tools have no workspaces and are outlined in blue.

Command bar contains buttons for calling frequently used functions:

- changing background color of selected row of active table,

- changing font color of selected row of active table,

- exporting active table to Excel,

- saving of selected aircraft or wheels information,

- saving of all aircrafts and wheels information,

- undoing the last change of active table,

- refreshing active table.

Look for description of tools' workspace in *User guide*.

10. Settings

Settings window is called from the main menu of the application.
Settings are divided into two groups:
1) «Common» – define functionality of the application;
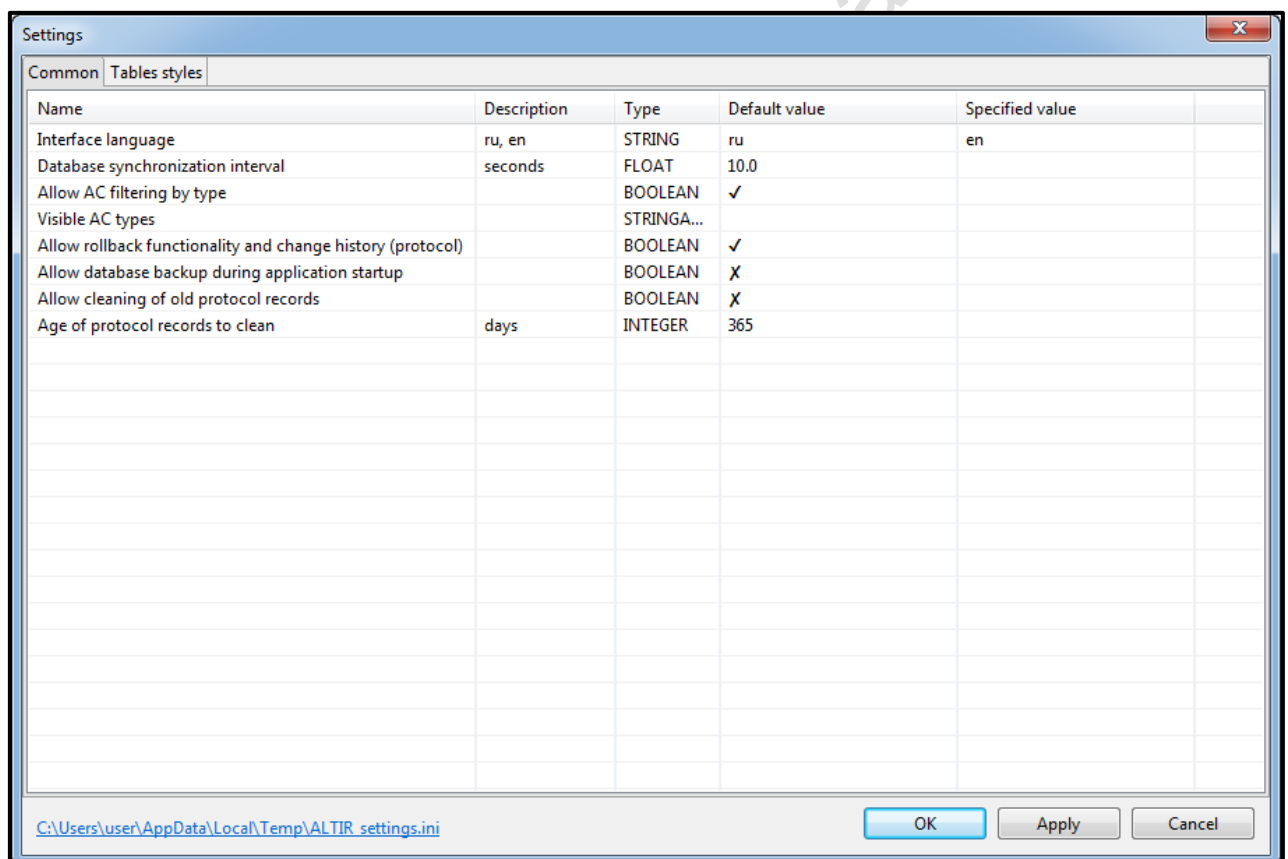2) «Table style» – define appearance of tables in workspace.

Common settings:

Setting name	Default value	Remark
Interface language	matches OS language	Missed localization file for default language or selected language will cause for English localization apply (only English and Russian localizations are available in this version).
Database synchronization interval	10 sec.	Interval to accesses the database to load occurred changes.
Visible AC types		List of aircraft types in form of strings separated by commas. Data of aircrafts of those types reads from database during synchronization, and, accordingly, are

		visible in the application. Works when filtering by aircraft type is enabled.
Allow AC filtering by type	yes	Enables or disables filtering by aircraft type. Enabled filtering and empty type list will cause all aircraft to load.
Allow rollback functionality and change history (protocol)	no	Enables or disables functionality of data changes rollback (for tables only).
Allow database backup during application startup	no	Enables or disables database backup functionality (for SQLite only).
Allow cleaning of old protocol records	no	Enables or disables functionality of deleting of protocol records. Request sends during synchronization with database.
Age of protocol records to clean	365 days	

To change the setting, enter the corresponding value in the cell of «Custom value» column.

Changing of common settings is available only for users with «Administrator» role. Application restart required to apply common settings.

Name	Description	Type	Default value	Specified value
Interface language	ru, en	STRING	ru	en
Database synchronization interval	seconds	FLOAT	10.0	
Allow AC filtering by type		BOOLEAN	✓	
Visible AC types		STRINGA...		
Allow rollback functionality and change history (protocol)		BOOLEAN	✓	
Allow database backup during application startup		BOOLEAN	✗	
Allow cleaning of old protocol records		BOOLEAN	✗	
Age of protocol records to clean	days	INTEGER	365	

C:\Users\user\AppData\Local\Temp\ALTIR_settings.ini

10

Table style settings include settings of all tables:
- row height,
- title height,
- background color,
- title background color,
- title text color.

as well as settings of background color, text color and width for each column of tables in AC tabs.

Application restart not required to apply table style settings.

11. User management

User management window is called from the main menu of the application.
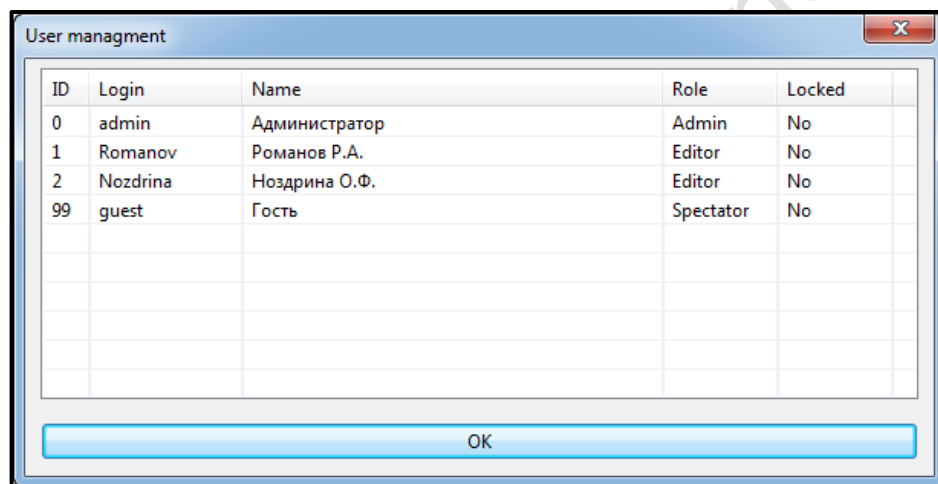
ALTIR contains following built-in user accounts (account data is stored at database):
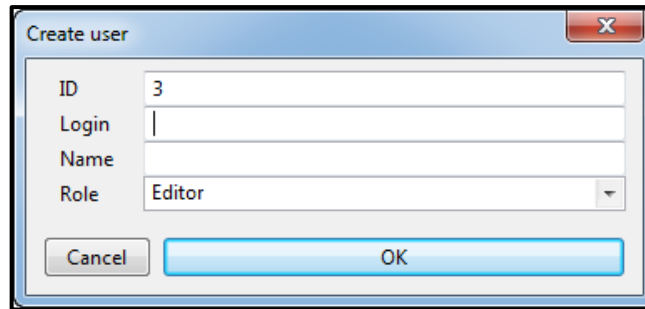
- Administrator account
  ID: 0
  Login: admin
  Password: admin
  Role: Admin

- Guest account
  ID: 99
  Login: guest
  Password: guest
  Role: Spectator

Right-click at table area of dialog box and select action «Create» to add new user.

PIN required to add, change or lock user account (PIN is provided at application purchase).

Description of the fields of «Create user» window

«**ID**» – user's sequence number (defined by administrator).
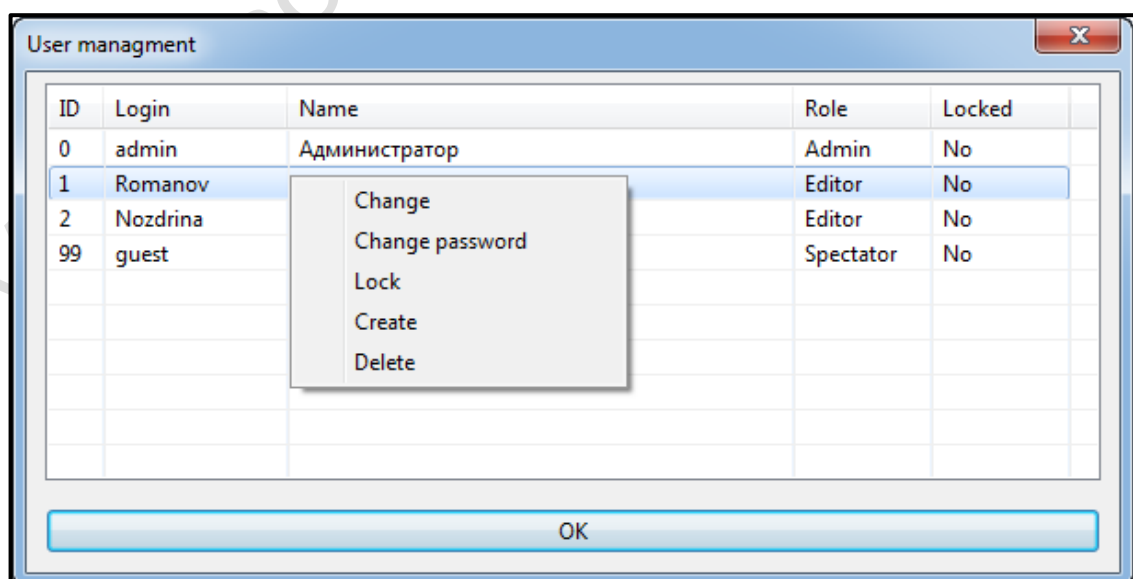«**Login**» – a string identifier required for authorization in the application.
«**Name**» – user name and/or surname.
«**Role**» – determines the level of user access to the application functionality.
User with «Administrator» role (Admin) has access to all functionality of the application. User with «Editor» role (Editor) has no access to Protocol, User management and Line editor. User with «Guest» role (Spectator) has only access to view aircraft data and export it to *.xls format.

New user is created with a blank password. Right-click at user's account line and select action «Change password» to change password. Password is required for authorization in the application.

In addition, «User management» window allows you to perform actions to change account data, block and remove account.
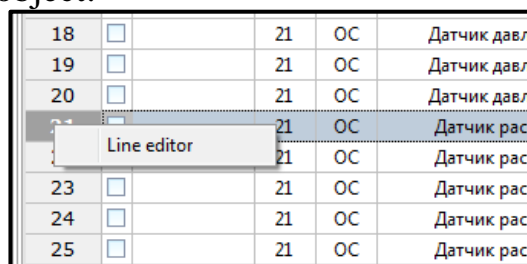
Built-in administrator account cannot be changed, deleted or blocked. Built-in guest account cannot be edited or deleted. Built-in administrator and guest accounts passwords can be changed.
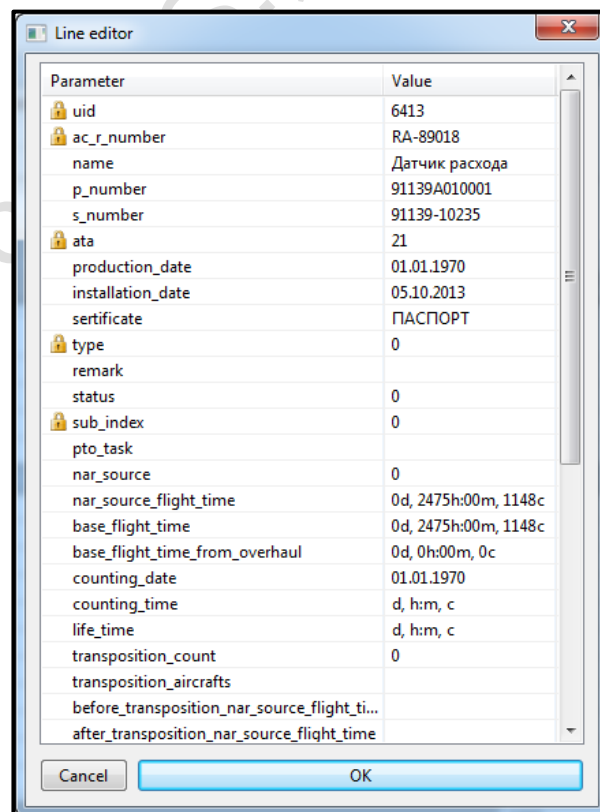
12. Line editor

Functionality of line editor is designed to change data of objects (lines) in AC tabs, which are not displayed not in table columns neither at object change dialog boxes, but affect the calculated values (for example: the number of component swaps between ACs, flight time of target engine at moment of creating of task or component, etc.).

Line editor is called from context menu on right-click at header (number) of the line of corresponding object.

Data fields of the object, whose changing may affect other objects (lines), are inaccessible for changing (has a lock icon).
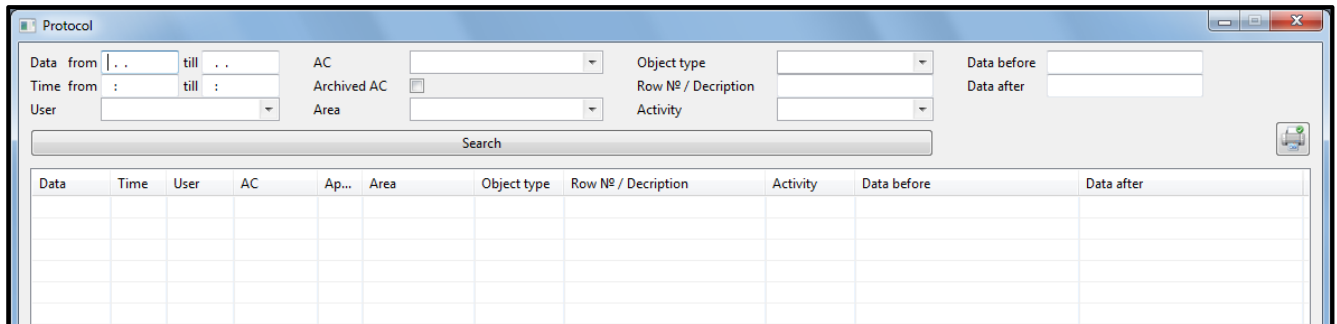

Line editor is available only for users with «Administrator» role.

13. Data changing protocol

Protocol allows to track user actions in the application.
Data changing protocol window is called from the main menu.

Description of protocol table columns

«**Date**» – contains date of data change.
«**Time**» – contains time (hours: minutes) of change.
«**User**» – contains name of user who made the change.
«**AC**» – contains registration number of aircraft in which the change was made.
«**Archive AC**» – indicates that the change was made in archived AC.
«**Area**» – indicates area of the application in which the change was made:
    «**Active tab**» – if the change was made in tabs of active aircraft;
    «**Archive tab**» – if the change was made in tabs of archive aircraft or archive elements;
    «**Dialog**» – if the change was made through any dialog box of the application (dialogs of creating, deleting, restoring components, forms, tasks, etc.);
    «**Wheels**» – if the change was made in wheels tool workspace;
    «**Line editor**» – if the change was made in line editor;
    «**Color palette**» – for object (line) background or text color change.
«**Type**» – contains type of changed object (line).
«**Row № / Description**» – is used to display a short description of modified object (line).
«**Action**» – contains type of action that was performed on object change.
«**Data before**» – contains information before change.
«**Data after**» – contains information after change.

Protocol window has a set of filters to search for change records:
- Filter by date and filter by time allows you to select changes for a certain period of time.
- «User» filter allows you to display only changes that were made by selected user.
- «AC» filter is designed to display changes for aircraft with specified registration number.

- «Archive AC» filter allows you to display changes that were made when AC was in archive.
- «Area» filter is used to display changes made in specified area of application.
- «Object type» filter allows you to display changes of objects of specified type.
- «Row № / Description» filter allows you to search for an object based on its name.
- «Action» filter allows you to display changes by the type of action performed to change object (line).
- «Data before» filter allows you to search for changes of object based on the data before the change.
- «Data after» filter allows you to search for changes of the object based on the data after the change.

Click search button to display information about changes in the protocol table.

14. Data import/export

Import

There are no special tools for data import in this version of the application.

AC tabs' tables support standard mechanism of inserting data from OS clipboard. For correct insertion, data must contain system line separator characters and tab characters to separate cells in line.

Export

As a tool for data export, the application offers functionality for outputting table contents to *.xls files. This functionality is provided by the capabilities of NatTable library, and allows you to get copy of exported table with used fonts, sizes and colors of all table elements.

To export active table to *.xls, click export button at command bar:

AC tabs' tables can be exported both individually and together. While exporting only one table it is possible to export selected rows.

AC tabs' tables (as well as protocol table, aircraft engines table, statements' tables) support standard mechanism for copying data to OS clipboard. Copied data is presented in standard format: lines are separated by system characters of line separator (usually «\r\n»), cells inside line are separated by tab character («\t»). Column headers of table are automatically added to the first row.

15.   Data backup

Database backup functionality is applicable only to SQLite databases. To activate it use the appropriate application setting (see p. 10).

When the functionality is enabled, the application, at startup, before the first loading of data from database makes a copy of database file. A copy is placed at: `%CD%/backup/` (where `%CD%` – path of application executable file). Name of copy contains current date and name of source database file. If a file with the same name already exists, no copying is performed.

If there is no `backup` folder, the application will try to create it automatically.

It is recommended to activate backup functionality at one of workstations. Thus, the backup will be performed once a day, at first start of the application.

To backup other types of databases, use their DBMS built-in backup tools.

16.   Troubleshooting

The application can report errors that occur during incorrect user actions or for other reasons, using three types of messages:

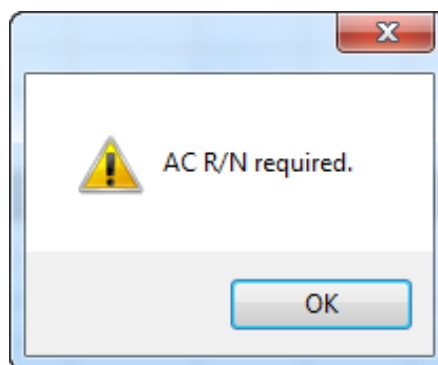### 1) Information message about incorrectly entered or missing data

Source: dialog windows;
Cause: requested data is missed or entered incorrectly (usually the message contains the reason of error);
Message content: error cause;
Way to respond: input valid data.
Example: registration number doesn't specified while aircraft creation:

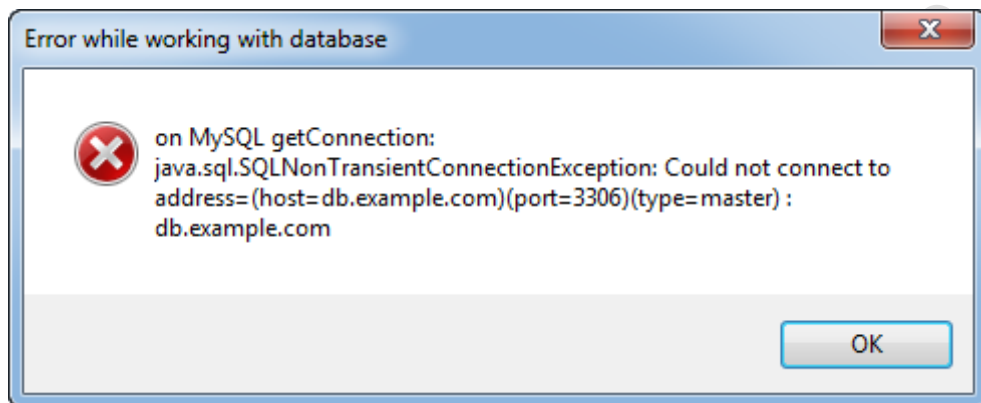## 2) **Error message while working with database**

<u>Source</u>: DB connection libraries;
<u>Cause</u>: incorrect settings of application or database, lack of connection to database, damaged database, etc;
<u>Message content</u>: name and description of error according to documentation for JDBC and corresponding DBMS;
<u>Way to respond</u>: action according to documentation for JDBC and corresponding DBMS.
Example: no connection with MySQL database:

Links to documentation for JDBC and supported DBMS:

System	Section	Link
JDBC	Java JDBC API	https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/
	java.sql Exception Summary	https://docs.oracle.com/javase/8/docs/api/java/sql/package-summary.html
SQLite	Result and Error Codes	https://www.sqlite.org/rescode.html
MySQL / MariaDB	Client Error Message Reference	https://dev.mysql.com/doc/refman/5.6/en/client-error-reference.html
	MariaDB Error Codes	https://mariadb.com/kb/en/library/mariadb-error-codes/
PostgreSQL	PostgreSQL Error Codes	https://www.postgresql.org/docs/9.6/errcodes-appendix.html
Firebird	Exception Codes and Messages	https://firebirdsql.org/file/documentation/reference_manuals/fblangref25-en/html/fblangref25-appx02-errorcodes.html

3) Unexpected error message

Source: unknown in advance;
Cause: unknown in advance;
Message content: information about error location;
Way to respond: sending error information to developers.
Example: