



IO-AVSTATS - Aviation Event Statistics

Manual

IO-Aero Team

Feb 18, 2024

| | | |
|----------|--------------------------------------|-----------|
| 1 | General Documentation | 1 |
| 1.1 | Introduction | 1 |
| 1.2 | Requirements | 1 |
| 1.3 | Installation | 2 |
| 1.4 | Configuration IO-AVSTATS | 4 |
| 1.5 | Configuration Logging | 6 |
| 1.6 | First Steps | 6 |
| 1.7 | Advanced Usage | 9 |
| 1.8 | Data Sources | 17 |
| 1.9 | PostgreSQL Administration | 25 |
| 2 | Master Data Logs | 27 |
| 3 | Transaction Data Logs | 29 |
| 4 | About | 31 |
| 4.1 | Release Notes | 31 |
| 4.2 | End-User License Agreement | 44 |
| 4.3 | Repository | 45 |
| 4.4 | Version | 45 |

General Documentation

1.1 Introduction

The Aviation Event Statistics (**IO-AVSTATS**) includes the following applications:

- **ae1982** - Aviation Event Analysis
- **pd1982** - IO-AVSTATS-DB Database Profiling
- **slara** - Association Rule Analysis

1.2 Requirements

The required software is listed below. Regarding the corresponding software versions, you will find the detailed information in the [Release Notes](#).

1.2.1 Operating System

Continuous delivery / integration (CD/CI) runs on Ubuntu and development is also done with Windows 10/11.

For the Windows operating systems, only additional the functionality of the `make` tool must be made available, e.g. via [Make for Windows](#)

The command-line shells supported are:

| Operating system | Command-line shell(s) |
|------------------|-----------------------|
| Ubuntu | bash |
| Windows 10/11 | cmd and PowerShell |

For Ubuntu, the end-of-line character and the execution authorization may need to be adjusted for the shell scripts. If the `dos2Unix` program is installed, the necessary adjustments can be made using the script `./scripts/run_prep_bash_scripts.sh` (Ubuntu).

1.2.2 Python

This project utilizes Python 3.10, which introduces significant enhancements in type hinting and type annotations. These improvements provide a more robust and clear definition of function parameters, return types, and variable types, contributing to improved code readability and maintainability. The use of Python 3.10 ensures compatibility with these advanced typing features, offering a more structured and error-resistant development environment.

1.2.3 AWS Command Line Interface

The AWS CLI is employed in this project to facilitate access to private Python libraries hosted on Amazon CodeArtifact, a fully managed artifact repository service. This integration allows for seamless retrieval and management of project dependencies, ensuring a streamlined and secure development workflow. Utilizing the AWS CLI ensures efficient and reliable access to the necessary Python libraries, enhancing the overall build and deployment process within the AWS ecosystem.

1.2.4 Docker Desktop

The project employs PostgreSQL for data storage and leverages Docker images provided by PostgreSQL to simplify the installation process. Docker Desktop is used for its ease of managing and running containerized applications, allowing for a consistent and isolated environment for PostgreSQL. This approach streamlines the setup, ensuring that the database environment is quickly replicable and maintainable across different development setups.

1.2.5 MS Access Database Engine

This Software consists of a set of components that facilitate the transfer of data between existing Microsoft Office files such as Microsoft Office Access (*.mdb and *.accdb) files and Microsoft Office Excel (*.xls, *.xlsx, and *.xlsb) files to other data sources. Connectivity to existing text files is also supported.

1.2.6 DBeaver Community - optional

DBeaver is recommended as the user interface for interacting with the PostgreSQL database due to its comprehensive and user-friendly features. It provides a flexible and intuitive platform for database management, supporting a wide range of database functionalities including SQL scripting, data visualization, and import/export capabilities. Additionally, the project includes predefined connection configurations for DBeaver, facilitating a hassle-free and streamlined setup process for users.

1.3 Installation

1.3.1 Python

The project repository contains a `scripts` directory that includes operating system-specific installation scripts for Python, ensuring a smooth setup across various environments.

- **Ubuntu:** For users on Ubuntu, the `run_install_python.sh` script is provided. This Bash script is created to operate within the default shell environment of Ubuntu, facilitating the Python installation process.
- **Windows:** The `run_install_python.bat` script is tailored for users on Windows systems. It is designed to be run in the Command Prompt and automates the Python installation process on Windows.

These scripts are named according to the convention `run_install_python.<ext>`, where `<ext>` corresponds to the script extension appropriate for the target operating system and shell environment (e.g., `.bat` for Windows or `.sh` for Ubuntu Bash). Users are recommended to execute the script matching their OS to ensure an efficient Python setup.

1.3.2 AWS Command Line Interface

Within the project's `scripts` directory, you will find a set of scripts specifically designed for the installation of the AWS Command Line Interface (AWS CLI). These scripts facilitate the installation process on different operating systems, ensuring a consistent and reliable setup.

- **Ubuntu:** Ubuntu users should utilize the `run_install_aws_cli.sh` script. This script is a Bash script that simplifies the AWS CLI installation on Ubuntu systems by setting up the necessary repositories and installing the CLI via `apt-get`.
- **Windows:** The `run_install_aws_cli.bat` script is intended for Windows users. It automates the process of downloading and installing the latest version of the AWS CLI in the Windows Command Prompt environment.

Each script is named following the pattern `run_install_aws_cli.<ext>`, with `<ext>` being the respective script extension suitable for the target operating system and shell environment (e.g., `.bat` for Windows CMD or `.sh` for Ubuntu Bash). Users are advised to execute the corresponding script for their operating system to achieve an optimal AWS CLI installation experience.

1.3.3 Docker Desktop

The `scripts` directory contains scripts that assist with installing Docker Desktop on Ubuntu, facilitating an automated and streamlined setup.

- **Ubuntu:** The `run_install_docker.sh` script is available for Ubuntu users. This Bash script sets up Docker Desktop on Ubuntu systems by configuring the necessary repositories and managing the installation steps through the system's package manager.
- **Windows:** For Windows users, it is recommended to download and install Docker Desktop using the traditional installer available at [Docker Desktop for Windows](#). This approach guarantees the most stable version and is tailored to integrate seamlessly with Windows-specific features and configurations.

Please select and execute the appropriate script for your operating system from the `scripts` directory. Windows users should follow the provided link to obtain the official installer for a guided installation experience.

1.3.4 MS Access Database Engine

- **Ubuntu Bash Shell:** The necessary software can be downloaded with the package manager `apt` as follows:

```
sudo apt-get update -y
sudo apt-get install -y unixodbc-dev
```

- **Windows:** The software can be downloaded from [here](#) and then installed according to the instructions provided.

1.3.5 DBeaver - optional

DBeaver is an optional but highly recommended tool for this software as it offers a user-friendly interface to gain insights into the database internals. The project provides convenient scripts for installing DBeaver on Ubuntu.

- **Ubuntu:** For Ubuntu users, the `run_install_dbeaver.sh` script facilitates the installation of DBeaver. This Bash script automates the setup process, adding necessary repositories and handling the installation seamlessly.
- **Windows:** Windows users are advised to download and install DBeaver using the official installer from the DBeaver website at [DBeaver Download](#). The installer ensures that DBeaver is properly configured and optimized for Windows environments.

To install DBeaver, locate the appropriate script in the `scripts` directory for Ubuntu. If you're a Windows user, please use the provided link to access the official installer for an intuitive installation experience.

1.3.6 Python Libraries

The project's Python dependencies are managed partly through Conda and partly through pip/pipenv. To facilitate a straightforward installation process, a Makefile is provided at the root of the project.

- **Development Environment:** Run the command `make conda-dev` from the terminal to set up a development environment. This will install the necessary Python libraries using Conda and pip/pipenv as specified for development purposes.
- **Production Environment:** Execute the command `make conda-prod` for preparing a production environment. It ensures that all the required dependencies are installed following the configurations optimized for production deployment.

The Makefile targets abstract away the complexity of managing multiple package managers and streamline the environment setup. It is crucial to have both Conda and the appropriate pip tools available in your system's PATH to utilize the Makefile commands successfully.

1.4 Configuration IO-AVSTATS

For the administration of the configuration parameters of **IO-AVSTATS** the tool [dynaconf](#) is used. The file `settings.io_aero.toml` is available as the configuration file. The names of **IO-AVSTATS** related environment variables must include the prefix `IO_AERO`. Layered environments are supported. The `test` layer is used for the automated tests.

1.4.1 Available Parameters

| Parameter | Description |
|--|---|
| <code>correction_work_dir</code> | file directory containing the files with the manual corrections |
| <code>database_commit_size</code> | number of rows processed before a progress message is created |
| <code>download_chunk_size</code> | chunk size for download from the NTSB website |
| <code>download_file_aviation_occurrence_categories_xlsx</code> | name of the file containing data of aviation occurrence categories |
| <code>download_file_countries_states_json</code> | name of the file containing data of countries and states |
| <code>download_file_faa_airports_xlsx</code> | name of the file containing data of airports |
| <code>download_file_faa_npias_xlsx</code> | name of the file containing National Plan of Integrated Airport Systems |
| <code>download_file_faa_runways_xlsx</code> | name of the file containing data of runways |
| <code>download_file_main_phases_of_flight_xlsx</code> | name of the file containing data of main phases of a flight |
| <code>download_file_sequence_of_events_xlsx</code> | name of the file containing data of sequence of events |
| <code>download_file_simplemaps_us_cities_xlsx</code> | simplemaps: name of the zipped US city file |
| <code>download_file_simplemaps_us_zips_xlsx</code> | simplemaps: name of the zipped US zip code file |
| <code>download_file_zip_codes_org_xls</code> | ZIP Code Database: name of the unzipped US zip code file |
| <code>download_timeout</code> | seconds to wait for the server to send data |

| | |
|------------------------------|---|
| download_url_ntsb_prefix | prefix of the download link for the NTSB data sets |
| download_work_dir | working directory for the processing of NTSB data sets |
| is_runtime_environment_local | local execution environment - unlike Docker |
| is_verbose | display progress messages for processing |
| max_deviation_latitude | maximum decimal deviation of the latitude in the database table even |
| max_deviation_longitude | maximum decimal deviation of the longitude in the database table even |
| odbc_connection_string | connection string for the MS Access ODBC driver |
| postgres_connection_port | database port number |
| postgres_container_name | container name |
| postgres_database_schema | database schema name |
| postgres_dbname | database name |
| postgres_dbname_admin | administration database name |
| postgres_host | database server hostname |
| postgres_password | database password |
| postgres_password_admin | administration database password |
| postgres_password_guest | guest database password |
| postgres_pgdata | file directory on the host for the database files |
| postgres_user | database username |
| postgres_user_admin | administration database username |
| postgres_user_guest | guest database username |
| postgres_version | requested PostgreSQL version from DockerHub |
| razorsql_jar_file_windows | name of the jar file (Windows version) |
| razorsql_java_path_windows | name of the Java file (Windows version) |
| razorsql_profile | name of the RazorSQL connection profile |
| razorsql_reference_dir | file directory of the database schema reference file |
| razorsql_reference_file | file name of the database schema reference file |

1.4.2 Example

```
[default]
check_value = "default"
correction_work_dir = "data/correction"
database_commit_size = 10000
download_chunk_size = 524288
...

[test]
check_value = "test"
correction_work_dir = "data/correction_test"
download_work_dir = "data/download_test"
postgres_connection_port = 5433
postgres_container_name = "io_avstats_db_test"
postgres_password = "postgres_password"
postgres_password_admin = "postgres_password_admin"
```

```
postgres_pgdata ="data/postgres_test"
```

1.4.3 Notes

The configuration parameters in the configuration files can be overridden with corresponding environment variables, e.g. the environment variable `IO_AERO_IS_VERBOSE` overrides the configuration parameter `is_verbose`.

1.5 Configuration Logging

In **IO-AVSTATS** the Python standard module for logging is used - details can be found [here](#).

The file `logging_cfg.yaml` controls the logging behaviour of the application.

Default content:

```
version: 1

disable_existing_loggers: False

formatters:
  simple:
    format: "%(asctime)s [%(name)s] [%(module)s.py ] %(levelname)-5s
%(funcName)s:%(lineno)d %(message)s"
  extended:
    format: "%(asctime)s [%(name)s] [%(module)s.py ] %(levelname)-5s
%(funcName)s:%(lineno)d \n%(message)s"

handlers:
  console:
    class: logging.StreamHandler
    level: INFO
    formatter: simple
  file_handler:
    class: logging.FileHandler
    level: INFO
    filename: logging_io_aero.log
    formatter: extended

root:
  level: DEBUG
  handlers: [ console, file_handler ]
```

1.6 First Steps

To get started, you'll first need to clone the repository, which contains essential scripts for various operating systems. After cloning, you will use these scripts to install the necessary foundational software. Finally, you will complete the repository-specific installation to set up your environment correctly. Detailed instructions for each of these steps are provided below.

1.6.1 Cloning the Repository

Start by cloning the *io-avstats* repository. This repository contains essential scripts and configurations needed for the project.

```
git clone https://github.com/io-aero/io-avstats
```

1.6.2 Install Foundational Software

Once you have successfully cloned the repository, navigate to the cloned directory. Within the *scripts* folder, you will find scripts tailored for various operating systems. Proceed with the subsection that corresponds to your operating system for further instructions.

Ubuntu

To set up the project on an Ubuntu system, the following steps should be performed in a terminal window within the repository directory:

a. Grant Execute Permission to Installation Scripts

Provide execute permissions to the installation scripts:

```
chmod +x scripts/*.sh
```

b. Install Python, pip, and pipenv

Run the script to install Python, pip, and pipenv:

```
./scripts/run_install_python.sh
```

c. Install AWS Command Line Interface

Execute the script to install the AWS CLI:

```
./scripts/run_install_aws_cli.sh
```

d. Install Miniconda and the Correct Python Version

Use the following script to install Miniconda and set the right Python version:

```
./scripts/run_install_miniconda.sh
```

e. Install Docker Desktop

To install Docker Desktop, run:

```
./scripts/run_install_docker.sh
```

f. Optionally Install DBeaver

If needed, install DBeaver using the following script:

```
./scripts/run_install_dbeaver.sh
```

g. Close the Terminal Window

Once all installations are complete, close the terminal window.

Windows 10/11

To set up the project on a Windows 10/11 system, the following steps should be performed in a command prompt (cmd) within the repository directory:

a. Install Python, pip, and pipenv

Run the script to install Python, pip, and pipenv:

```
scripts/run_install_python.bat
```

b. Install AWS Command Line Interface

Execute the script to install the AWS CLI:

```
scripts/run_install_aws_cli.bat
```

c. Install Miniconda and the Correct Python Version

Use the following script to install Miniconda and set the right Python version:

```
scripts/run_install_miniconda.bat
```

d. Close the Command Prompt

Once all installations are complete, close the command prompt.

e. Install Docker Desktop

To install Docker Desktop, download the software from here:

<https://www.docker.com/products/docker-desktop/>

and follow the installation instructions.

f. Optionally Install DBeaver

If needed, install DBeaver, download the software from here:

<https://dbeaver.io/>

and follow the installation instructions.

1.6.3 Repository-Specific Installation

After installing the basic software, you need to perform installation steps specific to the *io-avstats* repository. This involves setting up project-specific dependencies and environment configurations. To perform the repository-specific installation, the following steps should be performed in a command prompt or a terminal window (depending on the operating system) the repository directory.

Setting Up the Python Environment

To begin, you'll need to set up the Python environment using Miniconda and Pipenv, both of which are already pre-installed. You can use the provided Makefile for managing the environment.

a. For **production** use, run the following command:

```
make conda-prod
```

b. For **software development**, use the following command:

```
make conda-dev
```

These commands will create and configure a virtual environment for your Python project, ensuring a clean and reproducible development or production environment. The virtual environment is automatically activated by the Makefile, so you don't need to activate it manually.

System Testing with Unit Tests

If you have previously executed *make conda-dev*, you can now perform a system test to verify the installation using *make test*. Follow these steps:

a. Run the System Test:

Execute the system test using the following command:

```
make tests
```

This command will initiate the system tests using the previously installed components to verify the correctness of your installation.

b. Review the Test Results:

After the tests are completed, review the test results in the terminal. Ensure that all tests pass without errors.

If any tests fail, review the error messages to identify and resolve any issues with your installation.

Running system tests using *make tests* is a valuable step to ensure that your installation is working correctly, and your environment is properly configured for your project. It helps identify and address any potential problems early in the development process.

Downloading Database Files (Optional)

Database files can be downloaded from the IO-Aero Google Drive directory *io_aero_data/io-xpa/-database/io_xpa_db* to your local repository directory *data*. Before extracting, if a *postgres* directory exists within the *data* directory, it should be deleted.

Follow these steps to manage the database files:

a. *Access the IO-Aero Google Drive Directory:*

Navigate to the IO-Aero Google Drive and locate the directory *io_aero_data/io-xpa/database/io_xpa_db*.

b. *Download Database Files:*

Download the necessary database files from the specified directory to your local repository directory *data*.

c. *Delete Existing postgres Directory (if present):*

If a directory named *postgres* already exists within the *data* directory, you should delete it to avoid conflicts.

d. *Extract Database Files:*

The downloaded database files are in an archive format (ZIP) and should be extracted in the *data* directory. After completing these steps, the database files should reside in the *data* directory of your local repository and will be ready for use.

Creating the Docker Container with PostgreSQL DB

To create the Docker container with PostgreSQL database software, you can use the provided *run_io_avstats* script. Depending on your operating system, follow the relevant instructions below:

A. *Ubuntu (sh):*

```
./scripts/run_io_avstats.sh s_d_c
```

b. *Windows 10/11 (cmd):*

```
scripts\run_io_avstats.cmd s_d_c
```

These commands will initiate the process of creating the Docker container with PostgreSQL database software.

1.7 Advanced Usage

The main tool for operating **IO-AVSTATS** is the ``*run_io_avstats*`` script. The script is available in a Windows command line version and in a Linux bash shell version.

1.7.1 Overview

The following tasks can be executed with this script:

| Code | Task | Additional parameter(s) |
|-------|---|-----------------------------|
| a_o_c | Load aviation occurrence categories into PostgreSQL | |
| c_d_c | Run Docker Compose tasks - Cloud | clean, down, logs or up |
| c_d_i | Create or update an application Docker image | all or single Streamlit app |
| c_d_s | Create the IO-AVSTATS-DB PostgreSQL database schema | |
| c_f_z | Zip the files for the cloud | |
| c_l_l | Correct decimal US latitudes and longitudes | |
| c_p_d | Cleansing PostgreSQL data | |
| f_n_a | Find the nearest airports | |
| l_a_p | Load airport data into PostgreSQL | |

| | | |
|---------|--|-----------------------------------|
| l_c_d | Load data from a correction file into PostgreSQL | -e / -excel |
| l_c_s | Load country and state data into PostgreSQL | |
| l_n_a | Load NTSB MS Access database data into PostgreSQL | -m / -msaccess |
| l_s_d | Load simplemaps data into PostgreSQL | |
| l_s_e | Load sequence of events data into PostgreSQL | |
| l_z_d | Load ZIP Code Database data into PostgreSQL | |
| r_d_s | Refresh the PostgreSQL database schema | |
| r_s_a | Run a Streamlit application | single Streamlit app, e.g. ae1982 |
| s_d_c | Set up the IO-AVSTATS-DB PostgreSQL database container | |
| u_d_s | Update the IO-AVSTATS-DB PostgreSQL database schema | |
| u_p_d | Complete processing of a modifying MS Access file | -m / -msaccess |
| v_n_d | Verify selected NTSB data | |
| version | Show the IO-AVSTATS version | |

1.7.2 Detailed task list

a_o_c - Load aviation occurrence categories into PostgreSQL

Purpose

Load the definition of the valid CICTT codes.

Data Source

The data source can be found on the NTSB website here:

- [AVIATION OCCURRENCE CATEGORIES - DEFINITIONS AND USAGE NOTES](#)

The NTSB provides the data in a pdf file which must then be converted to MS Excel format `xlsx` before processing.

Implementation

```
CREATE TABLE public.io_aviation_occurrence_categories (  
    cictt_code varchar(10) NOT NULL,  
    identifier varchar(100) NOT NULL,  
    definition text NOT NULL,  
    first_processed timestamp NOT NULL,  
    last_processed timestamp NULL,  
    last_seen timestamp NULL,  
    CONSTRAINT io_aviation_occurrence_categories_pkey PRIMARY KEY (cictt_code)  
);
```

...

c_d_c - Run Docker Compose tasks - Cloud

Purpose

Manage the Docker containers needed in the cloud:

- **portainer**: container management
- **IO-AVSTATS-DB**: PostgreSQL database
- Application **ae1982**: Aircraft Events since 1982
- Application **pd1982**: Profiling Data since 1982
- Application **slara**: Association Rule Analysis
- **load_balancer**: load balancer NGINX

Processing Options

- clean - Remove all containers and images
- down - Stop Docker Compose
- logs - Fetch the logs of a container
- up - Start Docker Compose

c_d_i - Create or update an application Docker image

Purpose

Create the application-specific Docker images and store them on DockerHub.

Processing Options

- all - All Streamlit applications
- ael1982 - Aircraft Accidents in the US since 1982
- pdl1982 - Profiling Data for the US since 1982
- slara - Association Rule Analysis

c_d_s - Create the IO-AVSTATS-DB PostgreSQL database schema

Purpose

Create the database schema including the following steps, among others:

1. creation of a new database user, and
2. creation of a new database, and
3. creation of database objects such as database tables and so on.

The following parameters are used when creating the database schema:

- *postgres_dbname_admin* - administration database name
- *postgres_password_admin* - administration database password
- *postgres_user_admin* - administration database username

Subsequently, the task *u_d_s* (Update the PostgreSQL database schema) is also executed.

c_f_z - Zip the files for the cloud

Purpose

Collect and zip the elements needed for the cloud to run the **IO-AVSTATS** application there. The result is contained in the file **cloud.zip**.

c_l_1 - Correct decimal US latitudes and longitude

Purpose

An attempt is made to calculate missing decimal longitudes and latitudes using the database tables ``io_lat_lng`` and ``io_states``.

Implementation

1. In the database table ``events`` the values in the columns ``io_dec_lat_lng_actions``, ``io_dec_latitude``, ``io_dec_longitude`` and ``io_latlong_acq`` are deleted.
2. All rows in the database table ``events`` are processed where at least one of the columns ``dec_latitude`` or ``dec_longitude`` is empty or 0 and the column ``ev_country`` has the content ``USA``.
 - 2.1 An erroneous swapping of latitude and longitude is corrected.
 - 2.2 An attempt is made to calculate a missing column ``dec_latitude`` from the column ``latitude`` and a missing column ``dec_longitude`` from the column ``longitude``.

- 2.3 An attempt is made to calculate a missing column ``dec_latitude`` or ``dec_longitude`` from the column ``ev_site_zipcode``.
- 2.4 It tries to calculate a missing column ``dec_latitude`` or ``dec_longitude`` from the column ``ev_city``.
- 2.5 An attempt is made to calculate a missing column ``dec_latitude`` or ``dec_longitude`` from the column ``ev_state``.
- 2.6 For a missing column ``dec_latitude`` resp. ``dec_longitude`` the center of the USA is assumed.

c_p_d - Cleansing PostgreSQL data

Purpose

Clean up data the abnormalities in the database. This includes the following activities:

- remove trailing whitespace in string data types (trimming),
- converting string data types that contain only whitespace to NULL (nullifying).

As a result, a much simplified processing of the data is possible, e.g. for comparisons.

On the one hand, the task can be executed explicitly with the ``run_io_avstats_db`` script (task ``c_p_d``) and, on the other hand, it always runs after loading NTSB MS Access data into the PostgreSQL database (task ``l_n_a`` and ``u_p_d``).

f_n_a - Find the nearest airports

- TODO

Purpose

l_a_p - Load airport data into PostgreSQL

- TODO

Purpose

Data Source

Implementation

l_c_d - Load data from a correction file into PostgreSQL

- TODO

Purpose

Data Source

Implementation

This task allows files containing aviation accident data to be downloaded from the NTSB download site. These files are there as MS Access databases in a compressed format. The following subtasks are executed:

1. A connection to the NTSB download page is established.
2. The selected file is downloaded to the local system in chunks.
3. The downloaded file is then unpacked.
4. A script with the database schema definition is created with RazorSQL from the downloaded database.
5. The newly created script is then compared with a reference script for matching.

1_c_s - Load country and state data into PostgreSQL

- TODO

Purpose

Data Source

Implementation

1_n_a - Load NTSB MS Access database data into PostgreSQL

Purpose

This task allows files containing aviation event data to be downloaded from the **NTSB** download site. These files are there as MS Access databases in a compressed format. The following subtasks are executed:

1. A connection to the **NTSB** download page is established.
2. The selected file is downloaded to the local system in chunks.
3. The downloaded file is then unpacked.
4. A script with the database schema definition is created with RazorSQL from the downloaded database.
5. The newly created script is then compared with a reference script for matching.

Subsequently, the downloaded data can be loaded into the PostgreSQL database with the task 1_n_a (Load NTSB MS Access database data into PostgreSQL).

Data Sources

- **Pre2008.zip**: data set for 1982 through 2007
- **avall.zip**: data set from 2008 to the present
- **upDDMON.zip**: monthly supplements on the 1st, 8th, 15th and 22nd

Implementation

The PostgreSQL database **IO-AVSTATS-DB** completely maps the database schema of the **NTSB** MS Access database.

1_s_d - Load simplemaps data into PostgreSQL

- TODO

Purpose

Data Source

Implementation

This task transfers the data from an NTSB MS Access database previously downloaded from the NTSB website to the PostgreSQL database. The same MS Access database can be processed several times with this task without any problems, since only the changes are newly transferred to the PostgreSQL database. The initial loading is done with both MS Access databases Pre2008 and avall. After that only the monthly updates are then transferred.

1_s_e - Load sequence of events data into PostgreSQL

- TODO

Purpose

Data Source

Implementation

l_z_d - Load ZIP Code Database data into PostgreSQL

- TODO

Purpose

Data Source

Implementation

This task transfers the data from an NTSB MS Access database previously downloaded from the NTSB website to the PostgreSQL database. The same MS Access database can be processed several times with this task without any problems, since only the changes are newly transferred to the PostgreSQL database. The initial loading is done with both MS Access databases Pre2008 ubd avall. After that only the monthly updates are then transferred.

r_d_s - Refresh the PostgreSQL database schema

- TODO

Hereby changes can be made to the database schema. The task can be executed several times without problems, since before a change is always first checked whether this has already been done.

1. Materialized database view

- ``io_app_ae1982`` - provides the data for processing the task ``c_l_I`` (Correct decimal US latitudes and longitudes).

Example protocol:

```
Progress update 2022-...:09.337180 : INFO.00.004 Start Launcher. Progress update
2022-...:09.342679 : INFO.00.001 The logger is configured and ready. Progress update
2022-...:09.352180 : INFO.00.005 Argument task='r_d_s'. Progress update 2022-...:09.352180 :
-----
: INFO.00.071 Refreshing the database schema. Progress update 2022-...:09.352180 :
-----
: INFO.00.069 Materialized database view is refreshed: io_app_ae1982. Progress update
2022-...:19.366370 : -----
Progress update 2022-...:19.366370 : 10,187,690,800 ns - Total time launcher. Progress update
2022-...:19.366370 : INFO.00.006 End Launcher. Progress update 2022-...:19.366370 :
=====
```

r_s_a - Run the IO-AVSTATS application

- TODO

s_d_c - Set up the PostgreSQL database container

- TODO

u_d_s - Update the PostgreSQL database schema

- TODO

Hereby changes can be made to the database schema. The task can be executed several times without problems, since before a change is always first checked whether this has already been done.

1. New database tables:

- ``io_countries``: contains latitude and longitude of selected countries.

- `io_lat_lng`: used to store the **simplemaps** and **United States Zip Codes.org** data.
- `io_states`: contains the identification, name, latitude and longitude of all US states.

2. Extensions for database tables:

2.1 Database table `events`.

- The columns `io_city`, `io_country`, `io_latitude`, `io_longitude`, `io_site_zipcode` and `io_state` to store manual corrections.
- The columns `io_deviating_dec_latitude`, `io_deviating_dec_longitude`, `io_invalid_latitude`, `io_invalid_longitude`, `io_invalid_us_city`, `io_invalid_us_state` and `io_invalid_us_zipcode` for documenting data plausibility (task `v_nd`).
- the columns `io_dec_lat_lng_actions`, `io_dec_latitude` and `io_dec_longitude` to store corrected decimal latitude and longitude values.

3. New database views:

- `io_lat_lng_issues` - provides the data for processing the task `c_1_I` (Correct decimal US latitudes and longitudes).
- `io_accidents_us_1982` - provides event data for aviation accidents in the U.S. since 1982.

u_p_d - Complete processing of a modifying MS Access file

- TODO

v_n_d - Verify selected NTSB data

Purpose

This task can be used to perform a plausibility check for the following columns in the database table `events`:

- `dec_latitude`,
- `dec_longitude`,
- `ev_state`,
- `ev_site_zipcode`,
- `latitude`,
- `longitude`,

and the combination of:

- `ev_state` and `ev_city`,
- `ev_state`, `ev_city` and `ev_site_zipcode`.

The results of the check are stored in the following columns:

- `io_deviating_dec_latitude` (absolute difference),
- `io_deviating_dec_longitude` (absolute difference),
- `io_invalid_latitude` (true),
- `io_invalid_longitude` (true),
- `io_invalid_us_city` (true),
- `io_invalid_us_city_zipcode` (true),
- `io_invalid_us_state` (true),

- ``io_invalid_us_zipcode`` (true).

The tests are performed according to the following logic:

- ``io_deviating_dec_latitude``: Absolute difference between ``dec_latitude`` and ``latitude`` exceeding a given limit in ``max_deviation_latitude``.
- ``io_deviating_dec_longitude``: Absolute difference between ``dec_longitude`` and ``longitude`` exceeding a given limit ``max_deviation_longitude``.
- ``io_invalid_latitude``: Can the latitude in the ``latitude`` column be converted to its decimal equivalent?
- ``io_invalid_longitude``: Can the longitude in the ``longitude`` column be converted to its decimal equivalent?
- ``io_invalid_us_city``: For country *USA* and the given state, is the specified value in the ``ev_city`` column an existing city?
- ``io_invalid_us_city_zipcode``: For country *USA* and the given state, are the specified values in the ``ev_city`` column and in the ``ev_site_zipcode`` column an existing city?
- ``io_invalid_us_state``: For country *USA*, is the specified value in the ``ev_state`` column a valid state identifier?
- ``io_invalid_us_zipcode``: For country *USA*, is the specified value in the ``ev_site_zipcode`` column an existing zip code?

version - Show the ``IO-AVSTATS`` version

- TODO

1.7.3 First installation

The initial load in a fresh Windows environment requires the execution of the following tasks in the given order:

- ``c_d_s`` - Create the IO-AVSTATS-DB PostgreSQL database schema
- ``l_c_s`` - Load country and state data into PostgreSQL
- ``l_a_p`` - Load airport data into PostgreSQL
- ``a_o_c`` - Load aviation occurrence categories into PostgreSQL
- ``l_s_e`` - Load sequence of events data into PostgreSQL
- ``l_s_d`` - Load simplemaps data into PostgreSQL
- ``l_z_d`` - Load ZIP Code Database data into PostgreSQL
- ``u_p_d`` - Complete processing of a modifying MS Access file: ``Pre2008``

1.7.4 Regular updates

Every 1st of the month

1. Stop the Docker container ``IO-AVSTATS-DB``
2. Restore the current state of Pre2008
3. Start the Docker container ``IO-AVSTATS-DB``
4. Process the current ``avall`` file with code ``l_n_a``

Every 1st, 8th, 15th and 22nd

- Process the current ``upDDMON`` file with code ``u_p_d``

1.8 Data Sources

1.8.1 AVIATION OCCURRENCE CATEGORIES

The CICTT codes used in the `Aviation_Occurrence_Categories/aviation_occurrence_categories.xlsx` file is taken from [this document](#).

AVIATION OCCURRENCE CATEGORIES

DEFINITIONS AND USAGE NOTES

October 2013 (4.6)

Aviation s...

1.8.2 CICTT_SOE_MAP.csv

The content of the database table *io_sequence_of_events* is created based on this csv file.

| eventsoe_no | meaning | CICTT_Code | CICTT_Description |
|-------------|--------------------------------|------------|-------------------------|
| 0 | Unknown or undetermined | UNK | Unknown |
| 10 | Aircraft loading event | RAMP | Ground Handling |
| 20 | Aircraft servicing event | RAMP | Ground Handling |
| 30 | Preflight or dispatch event | | |
| 40 | Aircraft maintenance event | | |
| 50 | Aircraft inspection event | | |
| 60 | Attempted remediation/recovery | | |
| 70 | Airport occurrence | ADRM | Aerodrome |
| 80 | Ground handling event | RAMP | Ground Handling |
| 81 | AC/prop/rotor contact w person | RAMP | Ground Handling |
| 82 | Prop/jet/rotor blast/suction | RAMP | Ground Handling |
| 90 | Abnormal runway contact | ARC | Abnormal Runway Contact |
| 91 | Tailstrike | ARC | Abnormal Runway Contact |
| 92 | Hard landing | ARC | Abnormal Runway Contact |
| 93 | Dragged wing/rotor/float/other | ARC | Abnormal Runway Contact |
| 94 | Landing gear collapse | ARC | Abnormal Runway Contact |

1.8.3 FAA Airports

The FAA provides data on airports in the United States in the form of a csv file. This data is updated by the FAA at irregular intervals.

 Federal Aviation Administration

Airports

 **Private Member** ⓘ
Federal Aviation Administration - AIS

Summary

Airports of the United States, Puerto Rico, and Virgin Islands.

[View Full Details](#)

[Download](#)

Details

 **Dataset**
Feature Layer

 **23 February 2023**
Info Updated

 **23 February 2023**
Data Updated

 **22 July 2016**
Published Date

 **20,096 Records**
[View data table](#)

 **Public**
Anyone can see this content


 **Custom License**
[View license details](#)

20,096 records




1.8.4 FAA Runways

The FAA provides data on runways in the United States in the form of a csv file. This data is updated by the FAA at irregular intervals.


Federal Aviation Administration

Runways


Private Member ⓘ
 Federal Aviation Administration - AIS








Summary

Runways of the United States, Puerto Rico, and Virgin Islands.


[View Full Details](#)

[Download](#)

Details

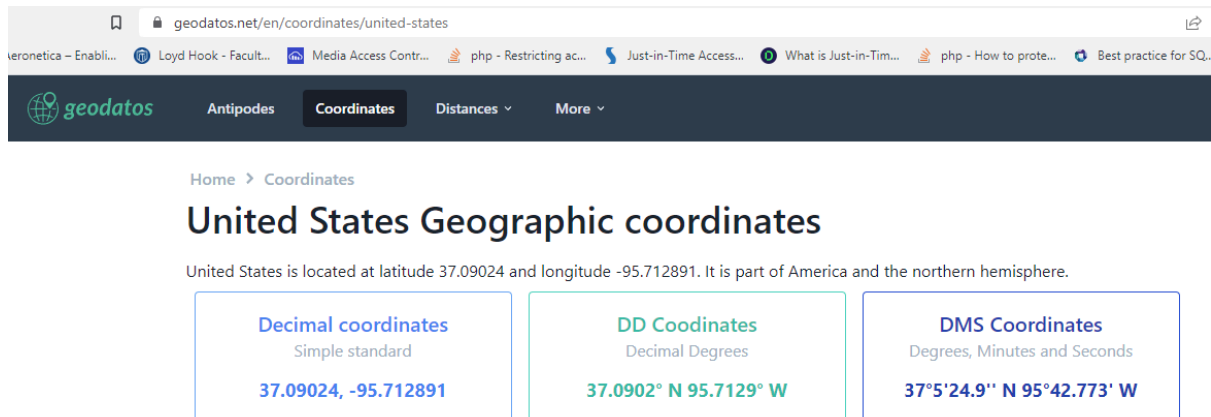
- 
Dataset
 Feature Layer
- 
As Needed
 Info Updated: 23 February 2023
- 
As Needed
 Data Updated: 23 February 2023
- 
23 February 2023 at 10:41
 Published Date
- 
23,998 Records
[View data table](#)
- 
Public
 Anyone can see this content
- 
Custom License
[View license details](#)

23,998 records



1.8.5 geodatos

The decimal latitude and longitude for the USA used in the `Countries_States/countries_states.json` file is taken from the [geodatos website](#).



geodatos.net/en/coordinates/united-states

Antipodes Coordinates Distances More

Home > Coordinates

United States Geographic coordinates

United States is located at latitude 37.09024 and longitude -95.712891. It is part of America and the northern hemisphere.

| Decimal coordinates Simple standard | DD Coordinates Decimal Degrees | DMS Coordinates Degrees, Minutes and Seconds |
|--|-----------------------------------|---|
| 37.09024, -95.712891 | 37.0902° N 95.7129° W | 37°5'24.9" N 95°42.773' W |

1.8.6 National Plan of Integrated Airport Systems (NPIAS)

The FAA provides data on preferred airports in the United States in the form of an MS Excel file. This data is updated by the FAA at irregular intervals.



United States Department of Transportation

Federal Aviation Administration

About Jobs News Search

Aircraft Air Traffic Airports Pilots & Airmen Data & Research Regulations Space

Home / Airports / Planning & Capacity / National Plan of Integrated Airport Systems (NPIAS)

Overview

Current NPIAS

Previous NPIAS

National Plan of Integrated Airport Systems (NPIAS)

The National Plan of Integrated Airport Systems (NPIAS) identifies nearly 3,300 public-use airports that are included in the national airport system, the roles they currently serve, and the amounts and types of airport development eligible for Federal funding under the [Airport Improvement Program \(AIP\)](#) over the next 5 years. The FAA is required to publish a 5-year estimate of AIP eligible development every other year.

The NPIAS contains all commercial service airports, all reliever airports, and selected public-owned general aviation airports.

- Subscribe to this page
- [Current NPIAS](#)
- [Previous NPIAS](#)

1.8.7 NTSB

The main data used in IO-AVSTATS is provided by the National Transportation Safety Board. On the accident data page is the link [Downloadable data sets](#), which contains the aviation accident data in MS Access format for free download.

| File | Created (mm/dd/yyyy) | Description |
|--------------------|----------------------|---|
| <i>avall.zip</i> | current | Data from January 1, 2008 to today |
| <i>PRE1982.zip</i> | 10/27/2020 | unknown |
| <i>Pre2008.zip</i> | 09/30/2020 | Data from January 1, 1982 to December 31, 2007 |
| <i>upDDMON.zip</i> | current | New additions and updates until DD day in the month MON |

The schemas of the two databases *avall* and *Pre2008* are identical except for the two new optional columns *dec_latitude* and *dec_longitude* in the database table *events* of the database *avall*.

Database Schema Comparison

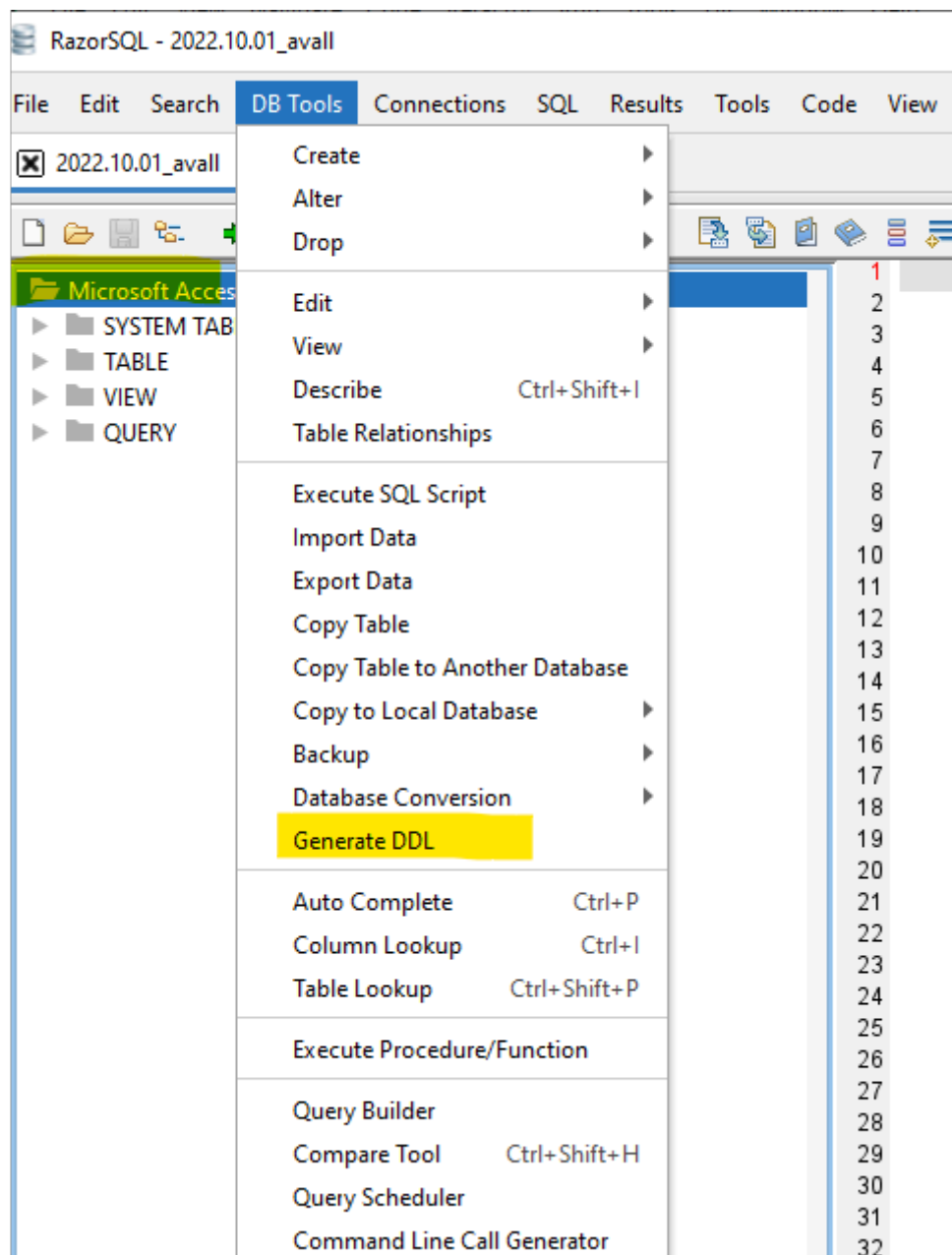
Before any new **NTSB** data set can be processed, the database schema of the new data set must first be compared with the database schema of the previous version of the **NTSB** data set *avall.mdb*.

Procedure:

- The RazorSQL program is started.
- Under 'Connections' select 'Add Connection Profile'.
- Select *MS Access* under *Add Connection Profile*.
- *Connection Type* is ODBC (Direct).
- In addition, an entry must be made at *Connection Profile Name* and *Database File*: - Connection Profile Name: IO-AVSTATS - Database File: IO-AVSTATS.mdb
- After pressing the *CONNECT* button you will be connected to the selected MS Access database.



- After opening the database, a SQL script with the DDL statements can be generated in the menu under *DB Tools* and *Generate DDL*.



- This SQL script can be saved as a file for further processing in the menu under *File* and *Save As*.

In case of any discrepancy, these must be implemented into the existing PostgreSQL database.

Table Processing Order

Based on the foreign keys (FK) present in the database schema, the following processing sequence results when creating the database schema and loading or updating the database tables:

Level 1 - without FK - events

Level 2 - FK: *ev_id* - aircraft - dt_events - NTSB_Admin

Level 3 - FK: *ev_id* & *Aircraft_Key* - dt_aircraft - engines - Events_Sequence - Findings - Flight_Crew - injury - narratives - Occurrences

Level 4 - FK: *ev_id* & *Aircraft_Key* & *crew_no* - dt_Flight_Crew - flight_time

Level 4 - FK: *ev_id* & *Aircraft_Key* & *Occurrence_No* - seq_of_events

Issues

The problem database tables listed below are not included in the PostgreSQL database schema.

Empty database tables

The following database tables are included in the Entity Relationship Diagram (ERD), but they do not contain any data in the database: - Occurrences - seq_of_events

Legacy database tables

The following database tables are included in the database but are missing from the ERD: - Country - ct_iaids - ct_seqevt - eADMSPUB_DataDictionary - states

Inconsistent database data - foreign key

- database table *ntsb_admin*: database table *event* has no row with *ev_id* = 20210527103155 (data source: *avall.zip*) - as a consequence, the foreign key to the events table had to be removed

Inconsistent database data - country USA and state

```
SELECT ev_state,
       count (*)
FROM events
WHERE ev_country = 'USA'
      AND NOT ev_state IN (SELECT state
                           FROM io_states ius)
GROUP BY ev_state
ORDER BY ev_state
```

| ev_state | count |
|----------|-------|
| AO | 17 |
| CB | 1 |
| GM | 45 |
| GU | 8 |
| OF | 14 |
| PO | 15 |
| PR | 112 |
| UN | 3 |
| VI | 6 |

Inconsistent database data - invalid USA latitude

```
SELECT count (*)
FROM io_lat_lng_issues
WHERE io_dec_lat_lng_actions LIKE '%ERROR.00.920%'
```

| count |
|-------|
| 430 |

Inconsistent database data - invalid USA longitude

```
SELECT count (*)
FROM io_lat_lng_issues
WHERE io_dec_lat_lng_actions LIKE '%ERROR.00.921%'
```

| count |
|-------|
| 462 |

1.8.8 opendatasoft

The decimal latitudes and longitudes for the US states used in

the `Countries_States/countries_states.json` file are taken from the [opendatasoft](https://public.opendatasoft.com/explore/dataset/us-state-boundaries/export/) website.

The screenshot shows the Opendatasoft website interface for the 'US State Boundaries' dataset. The top navigation bar includes 'EXPLORE', 'MAP BUILDER', 'API', and 'CHART BUILDER'. The dataset is titled 'US State Boundaries' and has 56 records. The 'Export' button is highlighted. The 'Flat file formats' section lists CSV, JSON, and Excel, each with a 'Whole dataset' download link. The 'Geographic file formats' section lists GeoJSON, Shapefile, and KML, each with a 'Whole dataset' download link. The 'Filters' section on the left shows a search bar and a list of state names with their respective record counts.

1.8.9 simplemaps

In order to fill in the missing decimal latitudes and longitudes, free available data from **simplemaps** is used. simplemaps offers postal codes and cities with their latitude and longitude.

The [US Zip Codes Database](#) link provides latitude and longitude for selected US zip codes:

| zip | lat | lng |
|-------|----------|-----------|
| 00601 | 18.18027 | -66.75266 |
| 00602 | 18.36075 | -67.17541 |
| 00603 | 18.45744 | -67.12225 |
| 00606 | 18.16585 | -66.93716 |
| 00610 | 18.2911 | -67.12243 |
| 00611 | 18.27698 | -66.80688 |
| 00612 | 18.41283 | -66.7051 |
| 00616 | 18.41878 | -66.6679 |
| 00617 | 18.44598 | -66.56006 |
| 00622 | 17.98892 | -67.1566 |
| 00623 | 18.08429 | -67.15336 |
| 00624 | 18.05905 | -66.71932 |

The [United States Cities Database](#) link provides latitude and longitude for selected US cities:

| city_ascii | state_id | state_name | county_fips | county_name | lat | lng |
|---------------|----------|----------------------|-------------|----------------------|---------|-----------|
| New York | NY | New York | 36081 | Queens | 40.6943 | -73.9249 |
| Los Angeles | CA | California | 06037 | Los Angeles | 34.1141 | -118.4068 |
| Chicago | IL | Illinois | 17031 | Cook | 41.8375 | -87.6866 |
| Miami | FL | Florida | 12086 | Miami-Dade | 25.784 | -80.2101 |
| Dallas | TX | Texas | 48113 | Dallas | 32.7935 | -96.7667 |
| Houston | TX | Texas | 48201 | Harris | 29.786 | -95.3885 |
| Philadelphia | PA | Pennsylvania | 42101 | Philadelphia | 40.0077 | -75.1339 |
| Atlanta | GA | Georgia | 13121 | Fulton | 33.7628 | -84.422 |
| Washington | DC | District of Columbia | 11001 | District of Columbia | 38.9047 | -77.0163 |
| Boston | MA | Massachusetts | 25025 | Suffolk | 42.3188 | -71.0852 |
| Phoenix | AZ | Arizona | 04013 | Maricopa | 33.5722 | -112.0892 |
| Detroit | MI | Michigan | 26163 | Wayne | 42.3834 | -83.1024 |
| San Francisco | CA | California | 06075 | San Francisco | 37.7558 | -122.4449 |
| Seattle | WA | Washington | 53033 | King | 47.6211 | -122.3244 |
| San Diego | CA | California | 06073 | San Diego | 32.8313 | -117.1222 |
| Minneapolis | MN | Minnesota | 27053 | Hennepin | 44.9635 | -93.2678 |

1.8.10 United States Zip Codes.org

In order to fill in the missing decimal latitudes and longitudes, free available data from **United States Zip Codes.org** is used. **United States Zip Codes.org** offers more complete postal codes but only with estimated latitudes and longitudes.

| zip | type | decommissioned | primary_city | acceptable_cities | state | country | latitude | longitude |
|-------|----------|----------------|--------------|-------------------|-------|---------|----------|-----------|
| 00501 | UNIQUE | 0 | Holtsville | | NY | US | 40.81 | -73.04 |
| 00544 | UNIQUE | 0 | Holtsville | | NY | US | 40.81 | -73.04 |
| 00601 | STANDARD | 0 | Adjuntas | | PR | US | 18.16 | -66.72 |
| 00602 | STANDARD | 0 | Aguada | | PR | US | 18.38 | -67.18 |
| 00603 | STANDARD | 0 | Aguadilla | Ramey | PR | US | 18.43 | -67.15 |
| 00604 | PO BOX | 0 | Aguadilla | Ramey | PR | US | 18.43 | -67.15 |
| 00605 | PO BOX | 0 | Aguadilla | | PR | US | 18.43 | -67.15 |
| 00606 | STANDARD | 0 | Maricao | | PR | US | 18.18 | -66.98 |
| 00610 | STANDARD | 0 | Anasco | | PR | US | 18.28 | -67.14 |
| 00611 | PO BOX | 0 | Angeles | | PR | US | 18.28 | -66.79 |
| 00612 | STANDARD | 0 | Arecibo | | PR | US | 18.45 | -66.73 |
| 00613 | PO BOX | 0 | Arecibo | | PR | US | 18.45 | -66.73 |
| 00614 | PO BOX | 0 | Arecibo | | PR | US | 18.45 | -66.73 |
| 00616 | STANDARD | 0 | Bajadero | | PR | US | 18.42 | -66.67 |
| 00617 | STANDARD | 0 | Barceloneta | | PR | US | 18.45 | -66.56 |
| 00622 | STANDARD | 0 | Boqueron | | PR | US | 17.99 | -67.15 |
| 00623 | STANDARD | 0 | Cabo Rojo | | PR | US | 18.08 | -67.14 |

1.9 PostgreSQL Administration

IO-AVSTATS uses the [PostgreSQL DBMS](#) for data management. PostgreSQL is a very powerful relational database system where the SQL language can be extended by procedural add-ons like [PL/pgSQL](#) or [PL/Python](#).

IO-AVSTATS provides all the tools to import the data from the MS Access databases available on the ****NTSB**** website and the data from the basic flat files available on [simplemaps United States Cities Database](#) and [US Zip Codes Database](#) websites.

1.9.1 Docker

The PostgreSQL Docker community provides PostgreSQL images suitable for **IO-AVSTATS** on [DockerHub](#). The official image can be found [here](#). The `c_d_l` task available in the `run_io_avstats_db` script downloads a selected PostgreSQL DBMS image from DockerHub and creates, configures and starts a Docker container.

1.9.2 Parameterization

The parameters can be defined either via environment variables or in the `settings.io_aero.toml` file. Details can be found under *Configuration* and *IO-AVSTATS*.

The following parameters are used when downloading the Docker image and creating the Docker container:

- `postgres_connection_port` - the database IP address
- `postgres_container_name` - the container name
- `postgres_dbname_admin` - the administration database name
- `postgres_password_admin` - the administration database password
- `postgres_pgdata` - the file directory on the host for the database files
- `postgres_user_admin` - the administration database username
- `postgres_version` - the requested PostgreSQL version from DockerHub

When using environment variables, they must contain the prefix `IO_AERO_`, e.g., `IO_AERO_POSTGRES_USER`.

1.9.3 Backup & Restore

Since the **IO-AVSTATS** database contains only statistical data, it is subject to a relatively low frequency of change. The following three events can lead to a change in the **IO-AVSTATS** database:

1. a new change file on the **NTSB** download site, or
2. an evolution of the database software or schema, or
3. a new PostgreSQL version that requires database migration.

This does not require sophisticated methods for backing up and restoring the **IO-AVSTATS** database, especially since the database contents reside in a dedicated local file directory. For data backup it is therefore sufficient to create a copy of the file directory with the **IO-AVSTATS** database before a change event. This copy can then replace the corrupted **IO-AVSTATS** database in the event of an error.

Very important: before any backup or restore, the PostgreSQL Docker container must be stopped first!

Master Data Logs

Transaction Data Logs

4.1 Release Notes

4.1.1 Version 24.02.15

Release Date: 15.02.2024

Applied Software

| Software | Version | Remark | Status |
|----------------|---------|--------|---------|
| AWS CLI | 2.15.20 | | upgrade |
| Docker Desktop | 4.27.1 | | upgrade |
| Python | 3.10.11 | | |
| RazorSQL | 10.5.3 | | |

Windows-specific Software

Important: All software components should be installed in the 64 bit version!

| Software | Version | Remark | Status |
|---|-----------|--------|--------|
| 7-Zip | 23.01 | | |
| Make for Windows | 3.81 | | |
| MS Access Database Engine 2016 Redis-tributable | 8/11/2020 | | |

Processed files

NTSB - National Transport Safety Board

Data source up15FEB.zip

- Download link: 02/15/2024 3:00:23 AM

4.1.2 Version 24.02.08

Release Date: 08.02.2024

Applied Software

| Software | Version | Remark | Status |
|----------------|---------|--------|---------|
| AWS CLI | 2.15.16 | | upgrade |
| Docker Desktop | 4.26.1 | | |
| PostgreSQL | 16.1 | | |

| | | | |
|----------|---------|--|---------|
| Python | 3.10.11 | | |
| RazorSQL | 10.5.3 | | upgrade |

Windows-specific Software

Important: All software components should be installed in the 64 bit version!

| Software | Version | Remark | Status |
|---|-----------|--------|--------|
| 7-Zip | 23.01 | | |
| The LLVM Compiler Infrastructure | 17.0.6 | | |
| Make for Windows | 3.81 | | |
| MS Access Database Engine 2016 Redis-tributable | 8/11/2020 | | |
| Visual Studio Community 2022 | 2022 | | |

Minimal Requirements Visual Studio Community 2022

Installation details

- Visual Studio core editor
- ▾ Desktop development with C++
 - ▾ Included
 - ✓ C++ core desktop features
 - ▾ Optional
 - ☒ MSVC v143 - VS 2022 C++ x64/x86 build t...
 - ☐ C++ ATL for latest v143 build tools (x86 &...
 - ☐ Windows 11 SDK (10.0.22000.0)
 - ☐ Security Issue Analysis
 - ☐ Just-In-Time debugger
 - ☐ C++ profiling tools
 - ☐ C++ CMake tools for Windows
 - ☐ Test Adapter for Boost.Test
 - ☐ Test Adapter for Google Test
 - ☐ Live Share
 - ☐ IntelliCode
 - ☐ C++ AddressSanitizer
 - ☐ vcpkg package manager
 - ☐ C++ MFC for latest v143 build tools (x86...
 - ☐ C++ Modules for v143 build tools (x64/x8...
 - ☐ Windows 11 SDK (10.0.22621.0)
 - ☐ C++/CLI support for v143 build tools (Late...
 - ☒ C++ Clang tools for Windows (15.0.1 - x64...
 - ☐ JavaScript diagnostics
 - ☐ Incredibuild - Build Acceleration
 - ☐ Windows 10 SDK (10.0.20348.0)
 - ☒ Windows 10 SDK (10.0.19041.0)
 - ☐ Windows 10 SDK (10.0.18362.0)
 - ☐ MSVC v142 - VS 2019 C++ x64/x86 build t...
 - ☐ MSVC v141 - VS 2017 C++ x64/x86 build t...
 - ☐ MSVC v140 - VS 2015 C++ build tools (v1...
 - ☐ Windows App SDK C++ Templates

Processed files

NTSB - National Transport Safety Board

Data source up08FEB.zip

- Download link: 02/08/2024 3:00:18 AM

4.1.3 Version 24.02.01

Release Date: 01.02.2024

Applied Software

| Software | Version | Remark | Status |
|----------------|---------|--------|---------|
| AWS CLI | 2.15.16 | | upgrade |
| Docker Desktop | 4.26.1 | | |
| PostgreSQL | 16.1 | | |
| Python | 3.10.11 | | |
| RazorSQL | 10.5.3 | | upgrade |

Windows-specific Software

Important: All software components should be installed in the 64 bit version!

| Software | Version | Remark | Status |
|---|-----------|--------|--------|
| 7-Zip | 23.01 | | |
| The LLVM Compiler Infrastructure | 17.0.6 | | |
| Make for Windows | 3.81 | | |
| MS Access Database Engine 2016 Redis-tributable | 8/11/2020 | | |
| Visual Studio Community 2022 | 2022 | | |

Minimal Requirements Visual Studio Community 2022

Installation details

- Visual Studio core editor
- ▾ Desktop development with C++
 - ▾ Included
 - ✓ C++ core desktop features
 - ▾ Optional
 - ☒ MSVC v143 - VS 2022 C++ x64/x86 build t...
 - ☐ C++ ATL for latest v143 build tools (x86 &...
 - ☐ Windows 11 SDK (10.0.22000.0)
 - ☐ Security Issue Analysis
 - ☐ Just-In-Time debugger
 - ☐ C++ profiling tools
 - ☐ C++ CMake tools for Windows
 - ☐ Test Adapter for Boost.Test
 - ☐ Test Adapter for Google Test
 - ☐ Live Share
 - ☐ IntelliCode
 - ☐ C++ AddressSanitizer
 - ☐ vcpkg package manager
 - ☐ C++ MFC for latest v143 build tools (x86...
 - ☐ C++ Modules for v143 build tools (x64/x8...
 - ☐ Windows 11 SDK (10.0.22621.0)
 - ☐ C++/CLI support for v143 build tools (Late...
 - ☒ C++ Clang tools for Windows (15.0.1 - x64...
 - ☐ JavaScript diagnostics
 - ☐ Incredibuild - Build Acceleration
 - ☐ Windows 10 SDK (10.0.20348.0)
 - ☒ Windows 10 SDK (10.0.19041.0)
 - ☐ Windows 10 SDK (10.0.18362.0)
 - ☐ MSVC v142 - VS 2019 C++ x64/x86 build t...
 - ☐ MSVC v141 - VS 2017 C++ x64/x86 build t...
 - ☐ MSVC v140 - VS 2015 C++ build tools (v1...
 - ☐ Windows App SDK C++ Templates

Processed files

FAA - Aeronautical data Delivery Service

Data source Airports

- Download link: [Version: 01/25/2024](#)

Data source Runways

- Download link: [Version: 01/25/2024](#)

simplemaps - Interactive Maps & Data

Data source United States Cities Database

- Download link: [Version: 1.78](#)

Data source US Zip Codes Database

- Download link: [Version: 1.84](#)

NTSB - National Transport Safety Board

Data source avall.zip

- Download link: [02/01/2024 06:15:30 AM](#)

Data source up01FEB.zip

- Download link: [02/01/2024 3:00:41 AM](#)

4.1.4 Version 24.01.22

Release Date: 22.01.2024

Modified Features

- new column 'cm_inPc' in database table 'findings'

Applied Software

| Software | Version | Remark | Status |
|----------------|---------|--------|--------|
| AWS CLI | 2.15.3 | | |
| Docker Desktop | 4.26.1 | | |
| PostgreSQL | 16.1 | | |
| Python | 3.10.11 | | |
| RazorSQL | 10.5.2 | | update |

Windows-specific Software

Important: All software components should be installed in the 64 bit version!

| Software | Version | Remark | Status |
|--|-----------|--------|--------|
| 7-Zip | 23.01 | | |
| The LLVM Compiler Infrastructure | 17.0.6 | | |
| Make for Windows | 3.81 | | |
| MS Access Database Engine 2016 Redistributable | 8/11/2020 | | |
| Visual Studio Community 2022 | 2022 | | |

Minimal Requirements Visual Studio Community 2022

Installation details

- Visual Studio core editor
- ▾ Desktop development with C++
 - ▾ Included
 - ✓ C++ core desktop features
 - ▾ Optional
 - ☒ MSVC v143 - VS 2022 C++ x64/x86 build t...
 - ☐ C++ ATL for latest v143 build tools (x86 &...
 - ☐ Windows 11 SDK (10.0.22000.0)
 - ☐ Security Issue Analysis
 - ☐ Just-In-Time debugger
 - ☐ C++ profiling tools
 - ☐ C++ CMake tools for Windows
 - ☐ Test Adapter for Boost.Test
 - ☐ Test Adapter for Google Test
 - ☐ Live Share
 - ☐ IntelliCode
 - ☐ C++ AddressSanitizer
 - ☐ vcpkg package manager
 - ☐ C++ MFC for latest v143 build tools (x86...
 - ☐ C++ Modules for v143 build tools (x64/x8...
 - ☐ Windows 11 SDK (10.0.22621.0)
 - ☐ C++/CLI support for v143 build tools (Late...
 - ☒ C++ Clang tools for Windows (15.0.1 - x64...
 - ☐ JavaScript diagnostics
 - ☐ Incredibuild - Build Acceleration
 - ☐ Windows 10 SDK (10.0.20348.0)
 - ☒ Windows 10 SDK (10.0.19041.0)
 - ☐ Windows 10 SDK (10.0.18362.0)
 - ☐ MSVC v142 - VS 2019 C++ x64/x86 build t...
 - ☐ MSVC v141 - VS 2017 C++ x64/x86 build t...
 - ☐ MSVC v140 - VS 2015 C++ build tools (v1...
 - ☐ Windows App SDK C++ Templates

Processed files

NTSB - National Transport Safety Board

Data source up22JAN.zip

- Download link: 01/22/2024 3:00:11 AM

4.1.5 Version 24.01.15

Release Date: 15.01.2024

New Features

- IO-AVSTATS includes IO-AVSTATS-DB

Modified Features

- new IO-TEMPLATE-APP version

Applied Software

| Software | Version | Remark | Status |
|----------------|---------|--------|--------|
| AWS CLI | 2.15.3 | | |
| Docker Desktop | 4.26.1 | | |
| PostgreSQL | 16.1 | | |
| Python | 3.10.11 | | |
| RazorSQL | 10.5.1 | | |

Windows-specific Software

Important: All software components should be installed in the 64 bit version!

| Software | Version | Remark | Status |
|---|-----------|--------|--------|
| 7-Zip | 23.01 | | |
| The LLVM Compiler Infrastructure | 17.0.6 | | |
| Make for Windows | 3.81 | | |
| MS Access Database Engine 2016 Redis-tributable | 8/11/2020 | | |
| Visual Studio Community 2022 | 2022 | | |

Minimal Requirements Visual Studio Community 2022

Installation details

- Visual Studio core editor
- ▾ Desktop development with C++
 - ▾ Included
 - ✓ C++ core desktop features
 - ▾ Optional
 - ☒ MSVC v143 - VS 2022 C++ x64/x86 build t...
 - ☐ C++ ATL for latest v143 build tools (x86 &...
 - ☐ Windows 11 SDK (10.0.22000.0)
 - ☐ Security Issue Analysis
 - ☐ Just-In-Time debugger
 - ☐ C++ profiling tools
 - ☐ C++ CMake tools for Windows
 - ☐ Test Adapter for Boost.Test
 - ☐ Test Adapter for Google Test
 - ☐ Live Share
 - ☐ IntelliCode
 - ☐ C++ AddressSanitizer
 - ☐ vcpkg package manager
 - ☐ C++ MFC for latest v143 build tools (x86...
 - ☐ C++ Modules for v143 build tools (x64/x8...
 - ☐ Windows 11 SDK (10.0.22621.0)
 - ☐ C++/CLI support for v143 build tools (Late...
 - ☒ C++ Clang tools for Windows (15.0.1 - x64...
 - ☐ JavaScript diagnostics
 - ☐ Incredibuild - Build Acceleration
 - ☐ Windows 10 SDK (10.0.20348.0)
 - ☒ Windows 10 SDK (10.0.19041.0)
 - ☐ Windows 10 SDK (10.0.18362.0)
 - ☐ MSVC v142 - VS 2019 C++ x64/x86 build t...
 - ☐ MSVC v141 - VS 2017 C++ x64/x86 build t...
 - ☐ MSVC v140 - VS 2015 C++ build tools (v1...
 - ☐ Windows App SDK C++ Templates

Processed files

NTSB - National Transport Safety Board

Data source up15JAN.zip

- Download link: 01/15/2024 3:00:19 AM

4.1.6 Version 24.01.08

Release Date: 08.01.2024

Applied Software

| Software | Version | Remark | Status |
|----------------|---------|--------|--------|
| AWS CLI | 2.15.3 | | update |
| Docker Desktop | 4.26.1 | | |
| PostgreSQL | 16.1 | | |
| Python | 3.10.11 | | |
| RazorSQL | 10.5.1 | | |

Windows-specific Software

Important: All software components should be installed in the 64 bit version!

| Software | Version | Remark | Status |
|---|-----------|--------|--------|
| 7-Zip | 23.01 | | |
| The LLVM Compiler Infrastructure | 17.0.6 | | |
| Make for Windows | 3.81 | | |
| MS Access Database Engine 2016 Redis-tributable | 8/11/2020 | | |
| Visual Studio Community 2022 | 2022 | | |

Minimal Requirements Visual Studio Community 2022

Installation details

- Visual Studio core editor
- ▾ Desktop development with C++
 - ▾ Included
 - ✓ C++ core desktop features
 - ▾ Optional
 - ☒ MSVC v143 - VS 2022 C++ x64/x86 build t...
 - ☐ C++ ATL for latest v143 build tools (x86 &...
 - ☐ Windows 11 SDK (10.0.22000.0)
 - ☐ Security Issue Analysis
 - ☐ Just-In-Time debugger
 - ☐ C++ profiling tools
 - ☐ C++ CMake tools for Windows
 - ☐ Test Adapter for Boost.Test
 - ☐ Test Adapter for Google Test
 - ☐ Live Share
 - ☐ IntelliCode
 - ☐ C++ AddressSanitizer
 - ☐ vcpkg package manager
 - ☐ C++ MFC for latest v143 build tools (x86...
 - ☐ C++ Modules for v143 build tools (x64/x8...
 - ☐ Windows 11 SDK (10.0.22621.0)
 - ☐ C++/CLI support for v143 build tools (Late...
 - ☒ C++ Clang tools for Windows (15.0.1 - x64...
 - ☐ JavaScript diagnostics
 - ☐ Incredibuild - Build Acceleration
 - ☐ Windows 10 SDK (10.0.20348.0)
 - ☒ Windows 10 SDK (10.0.19041.0)
 - ☐ Windows 10 SDK (10.0.18362.0)
 - ☐ MSVC v142 - VS 2019 C++ x64/x86 build t...
 - ☐ MSVC v141 - VS 2017 C++ x64/x86 build t...
 - ☐ MSVC v140 - VS 2015 C++ build tools (v1...
 - ☐ Windows App SDK C++ Templates

Processed files

NTSB - National Transport Safety Board

Data source up08JAN.zip

- Download link: 01/08/2024 3:00:17 AM

4.1.7 Version 24.01.01

Release Date: 01.01.2024

Applied Software

| Software | Version | Remark | Status |
|----------------|---------|--------|--------|
| AWS CLI | 2.15.3 | | |
| Docker Desktop | 4.26.1 | | |
| PostgreSQL | 16.1 | | |
| Python | 3.10.11 | | |
| RazorSQL | 10.5.1 | | |

Windows-specific Software

Important: All software components should be installed in the 64 bit version!

| Software | Version | Remark | Status |
|---|-----------|--------|--------|
| 7-Zip | 23.01 | | |
| The LLVM Compiler Infrastructure | 17.0.6 | | |
| Make for Windows | 3.81 | | |
| MS Access Database Engine 2016 Redis-tributable | 8/11/2020 | | |
| Visual Studio Community 2022 | 2022 | | |

Minimal Requirements Visual Studio Community 2022

Installation details

- Visual Studio core editor
- ▾ Desktop development with C++
 - ▾ Included
 - ✓ C++ core desktop features
 - ▾ Optional
 - ☒ MSVC v143 - VS 2022 C++ x64/x86 build t...
 - ☐ C++ ATL for latest v143 build tools (x86 &...
 - ☐ Windows 11 SDK (10.0.22000.0)
 - ☐ Security Issue Analysis
 - ☐ Just-In-Time debugger
 - ☐ C++ profiling tools
 - ☐ C++ CMake tools for Windows
 - ☐ Test Adapter for Boost.Test
 - ☐ Test Adapter for Google Test
 - ☐ Live Share
 - ☐ IntelliCode
 - ☐ C++ AddressSanitizer
 - ☐ vcpkg package manager
 - ☐ C++ MFC for latest v143 build tools (x86...
 - ☐ C++ Modules for v143 build tools (x64/x8...
 - ☐ Windows 11 SDK (10.0.22621.0)
 - ☐ C++/CLI support for v143 build tools (Late...
 - ☒ C++ Clang tools for Windows (15.0.1 - x64...
 - ☐ JavaScript diagnostics
 - ☐ Incredibuild - Build Acceleration
 - ☐ Windows 10 SDK (10.0.20348.0)
 - ☒ Windows 10 SDK (10.0.19041.0)
 - ☐ Windows 10 SDK (10.0.18362.0)
 - ☐ MSVC v142 - VS 2019 C++ x64/x86 build t...
 - ☐ MSVC v141 - VS 2017 C++ x64/x86 build t...
 - ☐ MSVC v140 - VS 2015 C++ build tools (v1...
 - ☐ Windows App SDK C++ Templates

Processed files

NTSB - National Transport Safety Board

Data source avall.zip

- Download link: 01/01/2024 06:23:50 AM

Data source up01JAN.zip

- Download link: 01/01/2024 3:00:17 AM

4.2 End-User License Agreement

4.2.1 End-User License Agreement (EULA) of IO-Aero Software

This End-User License Agreement (“EULA”) is a legal agreement between you and **IO-Aero**.

This **EULA** agreement governs your acquisition and use of our **IO-Aero Software** (“Software”) directly from **IO-Aero** or indirectly through a **IO-Aero** authorized reseller or distributor (a “Reseller”).

Please read this **EULA** agreement carefully before completing the installation process and using the **IO-Aero Software**. It provides a license to use the **IO-Aero Software** and contains warranty information and liability disclaimers.

If you register for a free trial of the **IO-Aero Software**, this **EULA** agreement will also govern that trial. By clicking “accept” or installing and/or using the **IO-Aero Software**, you are confirming your acceptance of the Software and agreeing to become bound by the terms of this **EULA** agreement.

If you are entering into this **EULA** agreement on behalf of a company or other legal entity, you represent that you have the authority to bind such entity and its affiliates to these terms and conditions. If you do not have such authority or if you do not agree with the terms and conditions of this **EULA** agreement, do not install or use the Software, and you must not accept this **EULA** agreement.

This **EULA** agreement shall apply only to the Software supplied by **IO-Aero** herewith regardless of whether other software is referred to or described herein. The terms also apply to any **IO-Aero** updates, supplements, Internet-based services, and support services for the Software, unless other terms accompany those items on delivery. If so, those terms apply.

License Grant

IO-Aero hereby grants you a personal, non-transferable, non-exclusive licence to use the **IO-Aero Software** on your devices in accordance with the terms of this **EULA** agreement.

You are permitted to load the **IO-Aero Software** (for example a PC, laptop, mobile or tablet) under your control. You are responsible for ensuring your device meets the minimum requirements of the **IO-Aero Software**.

You are not permitted to:

- Edit, alter, modify, adapt, translate or otherwise change the whole or any part of the Software nor permit the whole or any part of the Software to be combined with or become incorporated in any other software, nor decompile, disassemble or reverse engineer the Software or attempt to do any such things
- Reproduce, copy, distribute, resell or otherwise use the Software for any commercial purpose
- Allow any third party to use the Software on behalf of or for the benefit of any third party
- Use the Software in any way which breaches any applicable local, national or international law
- use the Software for any purpose that **IO-Aero** considers is a breach of this **EULA** agreement Intellectual Property and Ownership

IO-Aero shall at all times retain ownership of the Software as originally downloaded by you and all subsequent downloads of the Software by you. The Software (and the copyright, and other intellectual property rights of whatever nature in the Software, including any modifications made thereto) are and shall remain the property of **IO-Aero**.

IO-Aero reserves the right to grant licences to use the Software to third parties.

Termination

This **EULA** agreement is effective from the date you first use the Software and shall continue until

terminated. You may terminate it at any time upon written notice to **IO-Aero**.

It will also terminate immediately if you fail to comply with any term of this **EULA** agreement. Upon such termination, the licenses granted by this **EULA** agreement will immediately terminate, and you agree to stop all access and use of the Software. The provisions that by their nature continue and survive will survive any termination of this **EULA** agreement.

Governing Law

This **EULA** agreement, and any dispute arising out of or in connection with this **EULA** agreement, shall be governed by and construed in accordance with the laws of the United States.

4.3 Repository

[Link to the repository](#)

4.4 Version

24.2.15

