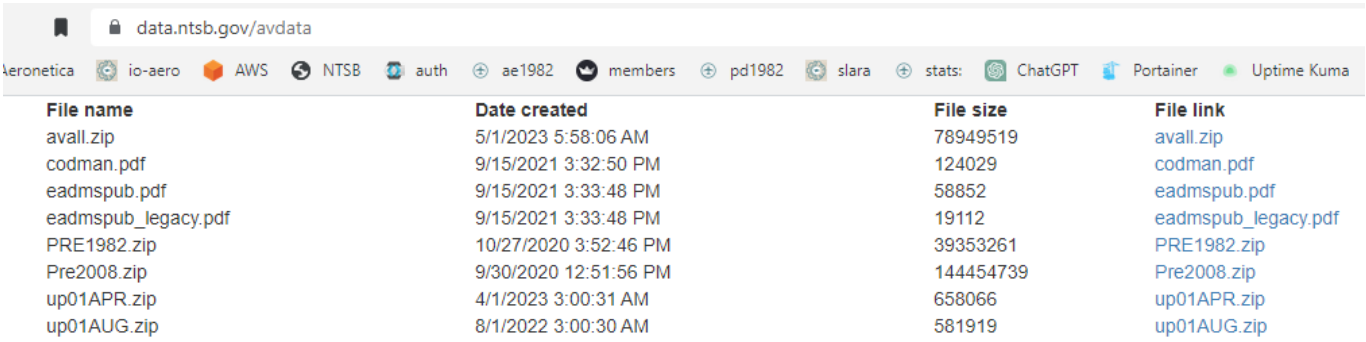
# How to add **NTSB** accident files

Aviation accident data provided by **NTSB** can be found at the following link{:target="_blank"}:


data.ntsb.gov/avdata

Aeronetica io-aero AWS NTSB auth ae1982 members pd1982 slara stats: ChatGPT Portainer Uptime Kuma

| File name | Date created | File size | File link |
|---|---|---|---|
| avall.zip | 5/1/2023 5:58:06 AM | 78949519 | avall.zip |
| codman.pdf | 9/15/2021 3:32:50 PM | 124029 | codman.pdf |
| eadmspub.pdf | 9/15/2021 3:33:48 PM | 58852 | eadmspub.pdf |
| eadmspub_legacy.pdf | 9/15/2021 3:33:48 PM | 19112 | eadmspub_legacy.pdf |
| PRE1982.zip | 10/27/2020 3:52:46 PM | 39353261 | PRE1982.zip |
| Pre2008.zip | 9/30/2020 12:51:56 PM | 144454739 | Pre2008.zip |
| up01APR.zip | 4/1/2023 3:00:31 AM | 658066 | up01APR.zip |
| up01AUG.zip | 8/1/2022 3:00:30 AM | 581919 | up01AUG.zip |

**NTSB** provides the following files:

- `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

In the database table `io_processed_files` you can find the files already processed by **IO-AVSTATS**:

| file_name | first_processed | last_processed | counter |
|---|---|---|---|
| Pre2008 | '2022-11-05 10:44:38.868' | '2022-11-05 11:49:12.766' | 2 |
| avall | '2022-11-05 11:59:49.223' | '2022-11-05 20:31:00.556' | 2 |
| data/Countries_States/countries_states.json | '2022-11-27 14:15:55.929' | '2022-11-27 14:15:57.546' | 3 |
| uszips.xlsx | '2022-11-27 15:25:29.741' | '2022-11-29 13:40:41.456' | 2 |
| uscities.xlsx | '2022-11-27 15:08:39.447' | '2022-11-29 13:54:55.259' | 4 |
| zip_code_database.xls | '2022-11-27 14:57:37.929' | '2022-11-29 14:26:07.066' | 3 |
| up01NOV | '2022-12-12 11:57:31.862' | | 1 |
| up08NOV | '2022-12-12 12:05:13.850' | | 1 |
| up15NOV | '2022-12-12 12:07:16.557' | | 1 |
| up22NOV | '2022-11-27 13:40:18.094' | '2022-12-12 12:08:15.915' | 2 |
| up01DEC | '2022-12-12 13:26:03.872' | | 1 |
| up08DEC | '2022-12-12 13:29:14.703' | | 1 |

Any file provided by **NTSB** can be processed several times with the process described in the following, as long as one processes also afterward all newer files again.

All processing tasks can be executed with the `run_io_avstats` script. For a more detailed description of these tasks, see under **Operation**.

## 1. Every first of the month

### 1.1 Prepare the database

- Stop the Docker container `io_avstats_db`.
- Remove the directory `data/postgres`
- Unzip the latest file `data/yy.mm.dd_postgres_Pre2008_20.09.30.zip`

- Start the Docker container `io_avstats_db` with `s_d_c - Set up the IO-AVSTATS-DB PostgreSQL database container`

### 1.2 Processing modified non-**NTSB** data sources

The following steps are performed only if the source files have changed:

| Source(s) | Version | Task | Description |
|---|---|---|---|
| Airports{:target="_blank"} | 10.08.2023 | l_a_p | Load airport data into PostgreSQL |
| NPIAS{:target="_blank"} | 2023-2027 NPIAS | | |
| Runways{:target="_blank"} | 10.08.2023 | | |
| NTSB{:target="_blank"} | October 2013 (4.6) | a_o_c | Load aviation occurrence categories into PostgreSQL |
| geodatos{:target="_blank"} | n/a | l_c_s | Load country and state data into PostgreSQL |
| NTSB | n/a | l_s_e | Load sequence of events data into PostgreSQL |
| United States Cities Database{:target="_blank"} | v1.76 | l_s_d | Load simplemaps data into PostgreSQL |
| US Zip Codes Database{:target="_blank"} | v1.82 | | |
| ZIP Code Database{:target="_blank"} | 42'735 active | l_z_d | Load ZIP Code Database data into PostgreSQL |

#### 1.2.1 Backup the file directory `data/postgres`

- Stop the Docker container `io_avstats_db`.
- Zip the file directory `postgres` in the file directory `data` - result is the file `postgres.zip`.
- Rename the file `data/postgres.sql` to `yy.mm.dd_postgres_upDDMON.zip`.
- Start the Docker container `io_avstats_db` with `s_d_c - Set up the IO-AVSTATS-DB PostgreSQL database container`

### 1.3 **NTSB** file `avall.zip`

The following task must be performed on the first of each month:

| Task | Description |
|---|---|
| l_n_a | Load NTSB MS Access database data into PostgreSQL |

## 2. **NTSB** file `upDDMON.zip`

The following tasks must be performed every month on the 1st, 8th, 15th and 22nd:

| No. | Task | Description |
|---|---|---|
| either | u_p_d | Complete processing of a modifying MS Access file |
| or 1. | l_n_a | Load NTSB MS Access database data into PostgreSQL |
| 2. | l_c_d | **optional** Load data from a correction file into PostgreSQL |
| 3. | c_l_l | Correct decimal US latitudes and longitudes |
| 4. | f_n_a | Find the nearest airports |
| 5. | v_n_d | Verify selected NTSB data |
| 6. | r_d_s | Refresh the PostgreSQL database schema |

## 3. Repository io-avstats

- Create the log file `docs/yyyy_mm_dd_log_upddMON.md`
- Modify the release notes file `docs/release_notes.md`
- Modify the word document `upload/IO-AVSTATS.docx`
- Create the pdf file `upload/IO-AVSTATS.pdf`
- Modify the word document `upload/IO-AVSTATS-DB.docx`
- Create the pdf file `upload/IO-AVSTATS-DB.pdf`
- Modify the configuration file `mkdocs.yml`
- Run `make pipenv-dev`
- Run `make final`
- Create a pull request
- Create the new release yy.mm.dd

## 4. Backup

The following steps are used to back up the database **IO-AVSTATS-DB**:

| No. | Description |
|---|---|
| 1. | Backup the file directory `data/postgres` |
| 2. | Update the Google Drive |

- Stop the Docker container `io_avstats_db`.
- Zip the file directory `postgres` in the file directory `data` - result is the file `postgres.zip`.
- Rename the file `data/postgres.sql` to `yy.mm.dd_postgres_upDDMON.zip`.
- Create a copy of the file `data/yy.mm.dd_postgres_upDDMON.zip` with the name `latest_postgres.zip`.

## 5. Create new application images

| Task | Description |
|---|---|
| c_d_i | Create or update an application Docker image |

# 6. Create the new cloud zip file

| Task | Description |
|---|---|
| c_f_z | Zip the files for the cloud |

# 7 Optional data quality checks

### 7.1 Event completeness

**Query**:

```
SELECT count(*) "Count",
       'Events Total' "Description"
  FROM events e
 UNION
SELECT count(*) ,
       'Events Total with Fatalities'
  FROM events e
 WHERE inj_tot_f > 0
 UNION
SELECT count(*) ,
       'Events US'
  FROM events e
 WHERE ev_state IS NOT NULL
   AND ev_state IN (SELECT state
                      FROM io_states is2)
 UNION
SELECT count(*) ,
       'Events US with Fatalities'
  FROM events e
 WHERE inj_tot_f > 0
   AND ev_state IS NOT NULL
   AND ev_state IN (SELECT state
                      FROM io_states is2)
 UNION
SELECT count(*) "Count",
       'Events Total since 1982' "Description"
  FROM events e
 WHERE ev_year >= 1982
 UNION
SELECT count(*) ,
       'Events Total with Fatalities since 1982'
  FROM events e
 WHERE ev_year >= 1982
   AND inj_tot_f > 0
 UNION
SELECT count(*) ,
       'Events US since 1982'
  FROM events e
 WHERE ev_year >= 1982
   AND ev_state IS NOT NULL
```

```
    AND ev_state IN (SELECT state
                     FROM io_states is2)
 UNION
SELECT count(*) ,
       'Events US with Fatalities since 1982'
  FROM events e
 WHERE ev_year >= 1982
   AND inj_tot_f > 0
   AND ev_state IS NOT NULL
   AND ev_state IN (SELECT state
                     FROM io_states is2)
 UNION
SELECT count(*) "Count",
       'Events Total since 2008' "Description"
  FROM events e
 WHERE ev_year >= 2008
 UNION
SELECT count(*) ,
       'Events Total with Fatalities since 2008'
  FROM events e
 WHERE ev_year >= 2008
   AND inj_tot_f > 0
 UNION
SELECT count(*) ,
       'Events US since 2008'
  FROM events e
 WHERE ev_year >= 2008
   AND ev_state IS NOT NULL
   AND ev_state IN (SELECT state
                     FROM io_states is2)
 UNION
SELECT count(*) ,
       'Events US with Fatalities since 2008'
  FROM events e
 WHERE ev_year >= 2008
   AND inj_tot_f > 0
   AND ev_state IS NOT NULL
   AND ev_state IN (SELECT state
                     FROM io_states is2)
 ORDER BY 2
```

### 7.2 Latitude & longitude

**Query Total:**:

```
SELECT count(*) "Count",
       io_dec_lat_lng_actions
  FROM events
 WHERE io_dec_lat_lng_actions IS NOT NULL
 GROUP BY io_dec_lat_lng_actions
 ORDER BY io_dec_lat_lng_actions
```

**Query Total since 1982:**:

```
SELECT count(*) "Count",
       io_dec_lat_lng_actions
  FROM events
 WHERE ev_year >= 1982
   AND io_dec_lat_lng_actions IS NOT NULL
 GROUP BY io_dec_lat_lng_actions
 ORDER BY io_dec_lat_lng_actions
```

**Query Total since 2008:**:

```
SELECT count(*) "Count",
       io_dec_lat_lng_actions
  FROM events
 WHERE ev_year >= 2008
   AND io_dec_lat_lng_actions IS NOT NULL
 GROUP BY io_dec_lat_lng_actions
 ORDER BY io_dec_lat_lng_actions
```

### 7.3 Issue summary

**Query Total:**:

```
SELECT count(*) "Count",
       'Latitude deviation' "Description"
  FROM events e
 WHERE io_dec_latitude_deviating IS NOT NULL
 UNION
SELECT count(*),
       'Longitude deviation'
  FROM events e
 WHERE io_dec_longitude_deviating IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid Latitude'
  FROM events e
 WHERE io_invalid_latitude IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid Longitude'
  FROM events e
 WHERE io_invalid_longitude IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US City'
```

```
  FROM events e
 WHERE io_invalid_us_city IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US City & Zipcode'
  FROM events e
 WHERE io_invalid_us_city_zipcode IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US State'
  FROM events e
 WHERE io_invalid_us_state IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US Zipcode'
  FROM events e
 WHERE io_invalid_us_zipcode IS NOT NULL
 ORDER BY 2
```

**Query US until 2008:**:

```
SELECT count(*) "Count",
       'Latitude deviation' "Description"
  FROM events e
 WHERE ev_year < 2008
   AND io_dec_latitude_deviating IS NOT NULL
 UNION
SELECT count(*),
       'Longitude deviation'
  FROM events e
 WHERE ev_year < 2008
   AND io_dec_longitude_deviating IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid Latitude'
  FROM events e
 WHERE ev_year < 2008
   AND io_invalid_latitude IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid Longitude'
  FROM events e
 WHERE ev_year < 2008
   AND io_invalid_longitude IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US City'
  FROM events e
 WHERE ev_year < 2008
   AND io_invalid_us_city IS NOT NULL
 UNION
SELECT count(*) ,
```

```
       'Invalid US City & Zipcode'
  FROM events e
 WHERE ev_year < 2008
   AND io_invalid_us_city_zipcode IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US State'
  FROM events e
 WHERE ev_year < 2008
   AND io_invalid_us_state IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US Zipcode'
  FROM events e
 WHERE ev_year < 2008
   AND io_invalid_us_zipcode IS NOT NULL
 ORDER BY 2
```

**Query US Accidents since 2008:**:

```
SELECT count(*) "Count",
       'Latitude deviation' "Description"
  FROM events e
 WHERE ev_year >= 2008
   AND io_dec_latitude_deviating IS NOT NULL
 UNION
SELECT count(*),
       'Longitude deviation'
  FROM events e
 WHERE ev_year >= 2008
   AND io_dec_longitude_deviating IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid Latitude'
  FROM events e
 WHERE ev_year >= 2008
   AND io_invalid_latitude IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid Longitude'
  FROM events e
 WHERE ev_year >= 2008
   AND io_invalid_longitude IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US City'
  FROM events e
 WHERE ev_year >= 2008
   AND io_invalid_us_city IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US City & Zipcode'
```

```
  FROM events e
 WHERE ev_year >= 2008
   AND io_invalid_us_city_zipcode IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US State'
  FROM events e
 WHERE ev_year >= 2008
   AND io_invalid_us_state IS NOT NULL
 UNION
SELECT count(*) ,
       'Invalid US Zipcode'
  FROM events e
 WHERE ev_year >= 2008
   AND io_invalid_us_zipcode IS NOT NULL
 ORDER BY 2
```