I was not sure which project I wanted to do so I made 3 project proposals.

Project proposal 1

Without a good weather history you can not make a good weather forecasts. The weather history should be up-to-date to make good predictions.

I want to build a pipeline that takes in live raw data from multiple types of weather stations and then nicely store this into a central system. This allows the data scientists to, at any moment, download the last X hours of good quality combined weather data as a parquet file to build their models. Another use case of this pipeline could be a real-time global weather monitoring dashboard used by logistics companies to manage shipping routes and schedules efficiently.

The royal Netherlands Meteologial institute of the ministry of infrastructure and water management (KNMI) provides a notification service (link). This notification service sends out push messages when KNMI updates or inserts files into their dataset. These notifications contain a link. You can send a get request to this link to get back another temporary link. Using this second temporary link you can download an .h5 (link) file.

I have never heard of .h5 files, but it seems to be a binary format. Apparently .h5 files are organized in a hierarchical way like a file system that can contain many types of files like images and json files or arrays of numbers. This means the data model of the dataset is semi structured. Each “file” also contains metadata and attributes with the actual data.

I have tested to download one .h5 file, and it was about 70kb. While I was running the notification client there were about 12 messages in about an hour.

The datasets come from KNMI but KNMI has many datasets they send notifications about. There are datasets in KNMI about different types of weather like temperature or sun radiation collected by different weather stations.

I picked this dataset because I think the real time aspect of data updates coming from multiple sources combined with the binary format of the data will be a good challenge.

Project proposal 2

I picked a dataset about Brazil regional spotify charts. Find it here (link) The data was collected from the Spotify api. This dataset lists the top 100 tracks for date ranges in 2021-2023 for 17 regions of Brazil. The dataset also contains 3 files that contain the features and metadata for each track in the time series. This way the csv files only have to store the id of the track and there is not a lot of data duplication. This is a structured zip file with a bunch of csv files of about 19 MB.

This dataset would need to be updated every quarter to stay relevant with the latest top tracks.

Each csv file with the top tracks has 100 rows. In total there are 5190 unique tracks, 487 different genres and 2056 artists. Because the Spotify api gives out data in csv files this data has been preprocessed by someone else.

I want to first turn these csv files into parquet files and then into a single database file. This allows the data scientist to do joins on the top 100 track data of a day while immediately getting the track feature data.

An interesting use case of this dataset is to make a model that can predict the features of top tracks throughout the year. Which features are important when might also differ for different genres. This way an artist could know which features to increase in the tracks they produce.

I choose this project as I have some experience with using the spotify api but in this dataset they preset the data as csv instead as json files and it is interesting that there is a time series.

If this is not challenging enough it will be cool to combine this dataset with a similar dataset from another music streaming service. The features of the tracks came from the spotify api. This means that I could annotate the tracks in the other dataset from the other streaming service with the same features as this one.

Project proposal 3

I picked a dataset that has data about 30.000 Data Breaches. Find it here (link)

This is a relational dataset. The data is stored as a single table in a csv file of 9 Kb. The description on kaggle states it is:

"a compilation of data from various sources detailing data breaches. These sources include press reports, government news releases, and mainstream news articles."

The csv file has these headers `,Entity,Year,Records,Organization type,Method,Sources`

The dataset would need to updated about once a year to stay relevant. But because it is historical data, fewer updates would still make it relevant.

In the end I want to produce a parquet file where every column of the data has the same type. This is harder than you might think. I looked into the csv for a bit and I found many different types of values in the Records column. Most values are integers like 91000 or 114000 but also things like:

- "unknown"
- "G20 world leaders"
- "19 years of data"
- "63 stores"
- ""
- "Source code compromised"

This will be interesting to clean up. Also, the sources is a list of numbers that seem to point to another file format like `[5]` or `[3][1]`.

I picked this data because I like cyber security and this dataset seemed to have a csv file that is pretty bad. The cleaned dataset could be used to get more insights in data leakage trends. The way data is leaked and if those ways have an impact on the size of the leakage. The dataset could also be used for a visualization of leakage over time.

If this is not challenging enough I can combine this data with stock price data of the companies in the years before and after they had the databreach. This could be used to investigate if data breaches (and the severity) can be correlated on the stock price of a company.

One thing to do to facilitate this analysis for the data scientist is to include a column with a severity score from 0 to 1 calculated on the stock price data.