TP 2: Data Pipelines with a Remote Storage Solution

EFREI Course 2024-2025

Introduction

This lab will guide you through building a complete data pipeline using LocalStack as a Data Lake infrastructure. You will learn to structure your data in the following zones:

- **Raw**: Storage of raw data as downloaded from Kaggle.
- **Staging**: Cleaning and preprocessing of data to remove errors.
- **Curated**: Transformation of data for use.

In this lab, I deliberately change the vocabulary from bronze/silver/gold to raw/staging/curated, which are also often used, so that you become comfortable with both.

You will use the data from the Kaggle PFAM challenge (click here to be redirected to the dataset) as in Lab 1, while using LocalStack to manage the storage and organization of the data.

Exercise 1: Environment Configuration

Objective

Configure LocalStack, AWS CLI and prepare a Python environment to manage a Data Lake.

Steps

As before, go to the GitHub repo `Data-Lakes`. You can either fork the `tp2-student` branch or create a remote to it and pull it into your local repo. Be careful which method you choose if you want to keep a local copy of your work from TP1!

Follow the instructions in the README to install the necessary dependencies and get started.

GitHub - Data-Lakes TP2

Exercise 2: Data Integration into Raw

Objective

Combine CSV files from the `train`, `test`, and `dev` subfolders and upload them to the raw bucket.

Instructions

1. Study the existing code in `src/unpack_data.py`.
2. Follow the instructions in the file to modify the code to integrate S3 and LocalStack.
3. Run the script with:
   `python src/unpack_data.py -input_dir ./data/raw -bucket_name raw -output_file_name combined_raw.csv`

Exercise 3: Data Preprocessing into Staging

Objective

Download the combined data from the raw bucket, preprocess it, and upload it into the staging bucket.

Instructions

1. Study the existing code in `src/preprocess.py`.
2. Modify the code to:
   - Download the data from the remote raw bucket.
   - Accelerate the manual split into train/dev/test with Numba.
   - Upload the resulting files to the staging bucket.
3. Run the script with:
   `python src/preprocess_to_staging.py -bucket_raw raw -bucket_staging staging -input_file combined_raw.csv -output_prefix preprocessed`

Exercise 4: Data Preparation for Curated

Objective

Perform final transformations of the data to prepare them for use with AI models.

Instructions

1. Create a new file `src/process_to_curated.py`.
2. Implement the following steps:
   - Download the preprocessed data (train) from staging.
   - Tokenize the `sequence` column using the tokenizer of `facebook/esm2_t6_8M_UR50D`.
   - Add the tokenized sequences to the metadata.
   - Upload the final file to the curated bucket.
3. Run the script with:
   `python src/process_to_curated.py -bucket_staging staging -bucket_curated curated -input_file preprocessed_train.csv -output_file tokenized_train.csv`

Exercise 5: Creating and Running the DVC Pipeline

Objective

Create a DVC pipeline to automatically run all steps.

Instructions

1. Modify the `dvc.yaml` file with the following steps:
   - `start_localstack`: Launch LocalStack.
   - `unpack`: Run `src/unpack_data.py`.
   - `preprocess`: Run `src/preprocess_to_staging.py`.
   - `process`: Run `src/process_to_curated.py`.
2. Use hints to define `cmd`, `deps`, and `outs`.
3. Add a DVC remote pointing to LocalStack S3:
   ```
   dvc remote add -d localstack-s3 s3://
   dvc remote modify localstack-s3 endpointurl http://localhost:4566
   ```
4. Run the pipeline with:
   ```
   dvc repro
   ```

Conclusion

At the end of this lab, you will have learned to:

- Configure a remote storage Data Lake infrastructure with LocalStack.
- Implement a complete pipeline to organize and process data.
- Automate and version data transformations with DVC.

Don't hesitate to ask your questions!