SpaNPortal

Siaminou Ioanna

12/6/2021

Project Outline

Introduction

Architecture

Implementation

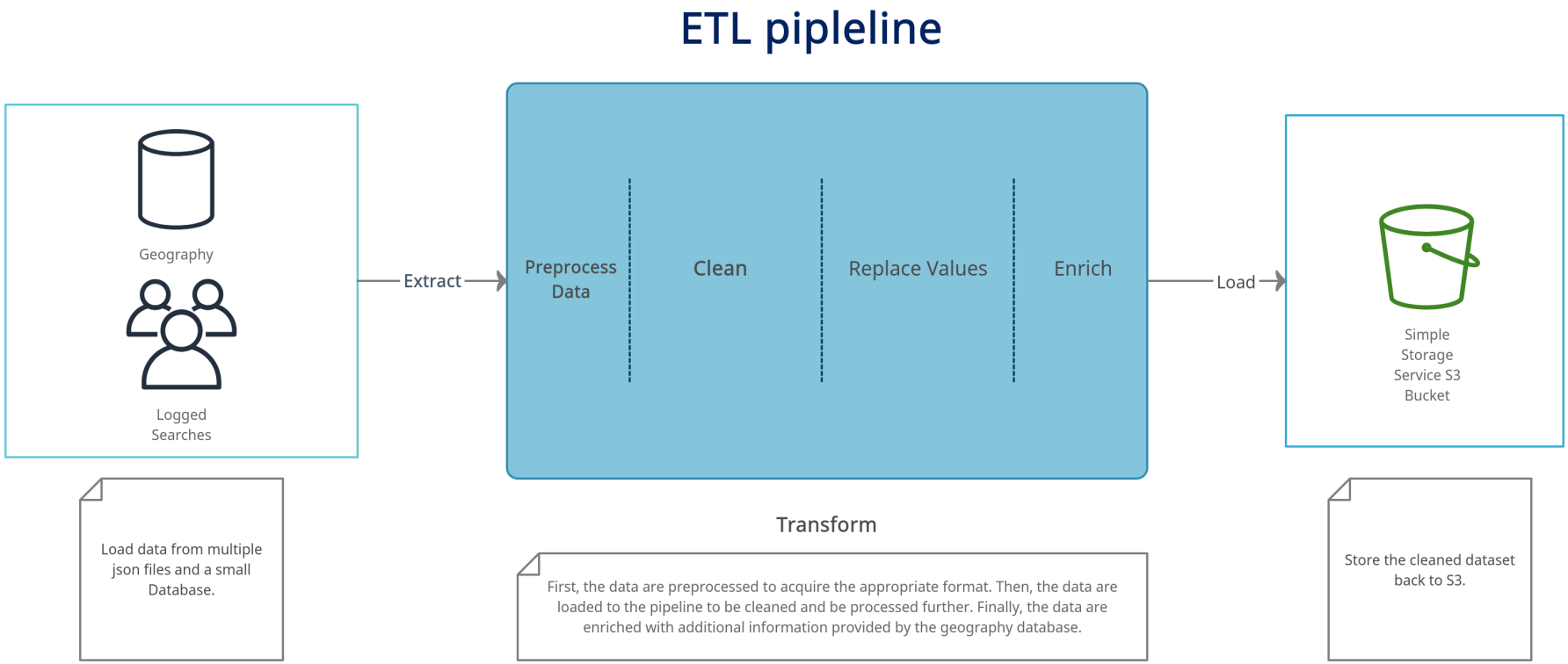
Context

1. SpaN is a company that provides an online portal for real estate services.
2. Strategy department of SpaN needs to process the logged property searches acquired by users based on certain criteria.
3. The design of a data lake and and the appropriate pipeline is necessary for better analysis of the data by the Strategy team.
4. The project consists of three distinct parts (Architecture, Implementation, Database Design).

Data & Dataflows

1. The data provided, consisted of a relational database with geographical information, and a series of files, containing user searches logged in a json format.
2. After the preprocessing, appropriate tranformations of the data, extraction of the useful attributes according to the given specifications.
3. The data are enriched with information provided by the second data source, the geography database.
4. Finally, the final dataset is stored in the appropriate format, in order to be stored used efficiently.
5. An overview of my approach follows in the next slide.

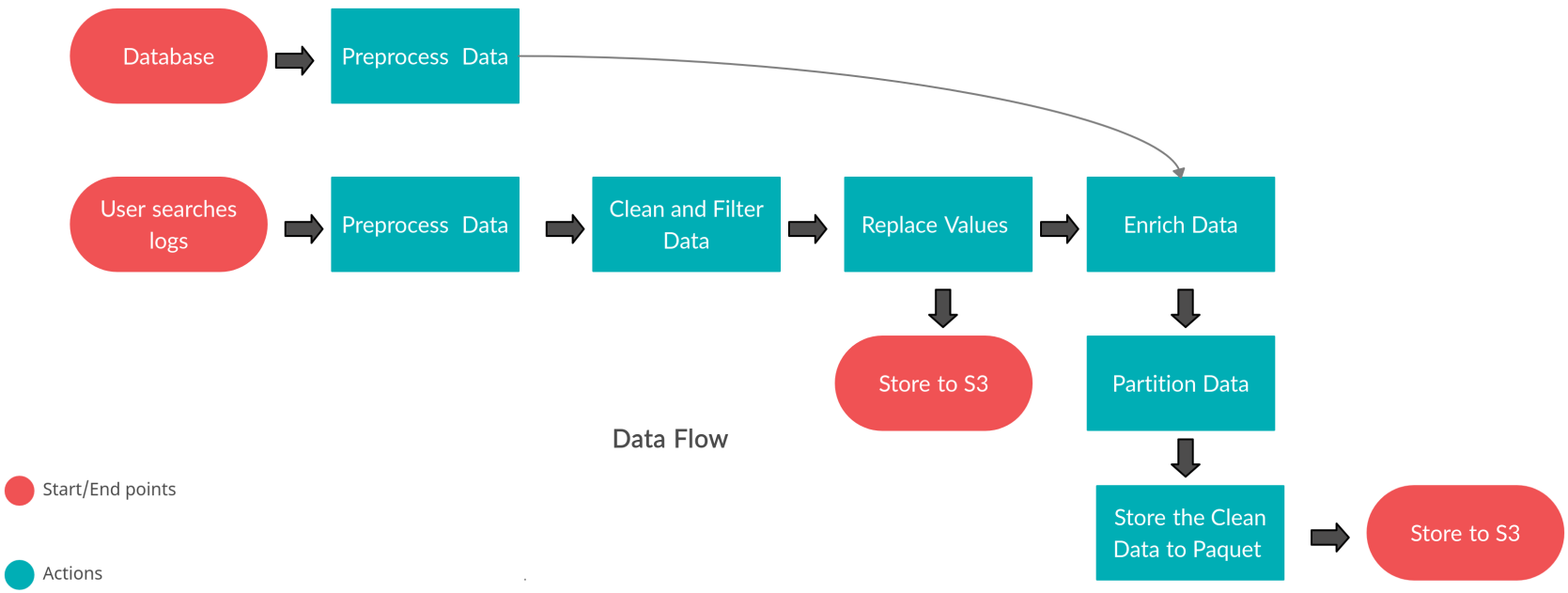
Architecture Diagram



How data flows through the pipeline?

1. After the extraction of the useful data from the data sources, the data are ingested to the transform stage.
2. First, the duplicate records are removed along with the null records.
3. The data are then filtered, to remove any record that correspond to *brokerIDs*.
4. Null values for specific columns are replaced.
5. And finally the data are enriched with additional information.

Data Flow Diagram



Assignment implementation

1. The project is implemented using Python programming language (version: Python 3.8.5).
2. Pyspark is used since it is can perform computation and transformations on data efficiently.
3. The project is accompanied with 2 bash scripts.
   - To activate a conda environment.
   - To manage the data directories.
4. The code is available on my github profile (https://github.com/ninasiam).

Some Notes

- The data are preprocessed, in a separate application available in the project directory.
- Due to a datatype mismatch (false/False) and the interpretation of json objects from pyspark, the json files could not be loaded correctly.
- I chose to process them separately, using spark RDDs, to keep the pipeline as clean as possible.
- I am pretty sure that a more efficient way exists (a parameter on pyspark json read function maybe), but at the moment this approach seemed to me straightforward.

More on the implementation

- The main script is the etl_main(), where the sparkSession is initialized.
- The pipeline functionality (Export, Transform, Load) is implemented on the ETL_fun.py file.
- The json file preprocessing is performed by preprocess_data.py.

Output files

- The output datasets (cleaned, enriched) are saved as a parquet files.
- The Parquet is a column oriented storage file available to any project on the Hadoop ecosystem.
- Aggregation queries are less time consuming compared to row oriented schemes.