AfterWork

Data Pipelines with Redis

Background Information

As a telecommunications data engineer, you have been tasked with building a pipeline that can efficiently extract, transform, and load data from CSV files into a Postgres database. The data to be extracted is related to customer call logs, which contain information about the duration, cost, and destination of customer calls. The extracted data needs to be transformed to ensure it is in the correct format and structure for storage in the database. The pipeline should also cache data using Redis to speed up the data extraction and transformation.

Guidelines

You can follow the steps below:

- Start by creating a new Python file and importing the necessary libraries (pandas, psycopg2, and redis).
- Create a Redis client object and connect it to the Redis Labs cloud instance.
- Create an extract function that reads the CSV files using pandas and caches the data in Redis.
- Create a transform function that cleans, structures, and formats the extracted data.
- Create a load function that connects to the Postgres database using psycopg2 and loads the transformed data into the database.
- Combine the extract, transform, and load functions into a single data pipeline that extracts data from a CSV file, transforms it and loads it into a Postgres database.
- Test the pipeline with a sample dataset to ensure it works correctly.

Sample CSV Files

We’ve provided a sample CSV file (customer_call_logs.csv) that you can use for this data pipeline. Files for this project can be downloaded from here (link).

Deliverables

We will be expected to deliver a GitHub repository with the following:

- A python file for the data pipeline.
- Documentation of the pipeline.
  - Highlight at least 3 best practices used during the implementation.
  - Recommendations for deployment and running the pipeline with a cloud-based provider.