ECS713 Group Project

Hassan Bashir 170333320,
Sara Nikamalfard 190677491,
Liam Radley 200907291

December 8, 2020

Contents

1 A Brief Guide

This program allows the user to download a relational database filled with 400 location's worth of information about the air quality of various cities around the world, as measured against parameters such as NO_x and CO_2 levels, as well as geographical information about the cities. As such, the database is comprised of three tables – a 'measurements' table, a 'locations' table and a 'parameter' table. The program populates the tables using HTTP requests to 'parameters.json' and to 'measurements.json'. Information for population of both the locations and measurements tables are extracted from 'measurements.json', whereas the parameters table is populated via 'parameters.json'. This database is relational – there are foreign key constraints on both the location and the parameter id used in each measurement. This is shown in the diagram below:

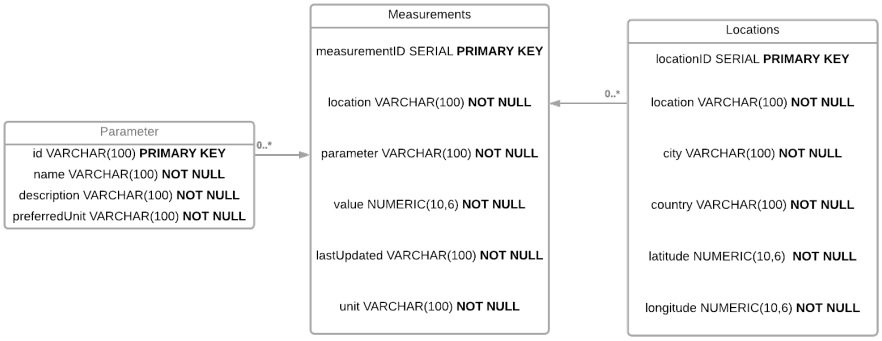

Figure 1: A UML diagram of the underlying database.

The data harvester makes use of a static JSON dump, provided by an API from
OpenAQ.org but stored within the GitHub repository of the program for the purpose of the exercise. This API, when it works, provides access to various JSON dumps of information about air quality that are updated in real time. After experiencing multiple refusals of HTTP requests, we opted to move a static chunk of 400 locations to a JSON file for storage in a way that would not refuse the request. These refusals were experienced on multiple student IPs, a VPN and the university system via RGate.[1]

The running of the program is simple, and can be decomposed into the following steps[2]:

1. Open the *Database.hs* file and, within the getConn function, edit the password value to match that of on your setup for the Postgres user. It is set to "admin" as default.

[1]For completeness, the URLS for the API are `https://api.openaq.org/v1/latest?has_geo=true&order_by=city&limit=400` and `https://api.openaq.org/v1/parameters` for the locations and measurements, and the parameters respectively. This refusal was still in force as of 08/12/2020.

[2]These instructions assume you already have PostgreSQL installed on your machine. If not then consult `https://www.postgresql.org/download/` for advice on how to install and set up PostgreSQL.

2. Open an instance of PostgreSQL shell, and enter the following:
   CREATE DATABASE airqual_db;

3. Open an instance of Command Line, PowerShell or Terminal.

4. Navigate to the *functional-assignment* folder.

5. Enter the following to the prompt:
   stack build
   stack exec functional-assignment-exe
   The program will run, and will display the following options:

   ```
   "Usage: System command [args]"
   "selectLocations
   Selects all locations in ascending order, from locations table"
   "deleteLocations
   Deletes countries, based on user input"
   "joinParameterAndMeasurement
   Joins Parameter and Measurement table to retrieve some data"
   ```

   Insertion of any of these three commands after *stack exec functional-assignment-exe* will cause a different function to run on the database.
   selectLocations prints to the terminal a list of all locations provided in the dataset.
   deleteLocations allows the user to input a country within the database, and remove all appearances of this country from the measurements and locations tables. The validity of this can be verified by opening an SQL shell after running the program.
   joinParameterAndMeasurement. This provides the parameter unit, name and preferred scientific unit of measurement via an inner join.

If you have another HDBC-compatible DBMS installed, such as SQLite, then you can opt to configure the program with this instead. This is done by importing the relevant HDBC module and altering the getConn in *Database.hs* to use the module's connect function, then follow the above from step 3 onwards. The program will then proceed to make two requests to the OpenAQ API, and populate each of the tables with values requested from OpenAQ. The program will then perform a series of queries on the database to extract information about each reading and present it to the user in an easy-to-read format. If using an instance of an SQL shell, and would prefer to make use of database functionality outside of the pre-defined methods, ensure that you run the following line before you perform any queries:
SET CLIENT_ENCODING TO 'utf-8';
This is because you will most likely find that characters that are not part of WIN1252 encoding will be returned in the query. Failure to perform this step will cause queries to also fail.

2 Access to Documentation

Access to documentation is provided on the following relative path from the functional-assignment directory: \.stack-work\install\b5baaaa8\doc\functional-assignment-0.1.0.0\index.html