Group Project Part 2 - MongoDB

of Systems and Methods for Big and Unstructured Data Course
(SMBUD)
held by
Brambilla Marco
Tocchetti Andrea

Group 14

Banfi Federico
10581441

Carotenuto Alessandro
10803080

Donati Riccardo
10669618

Mornatta Davide
10657647

Zancani Lea
10608972

Academic year 2021/2022

POLITECNICO
MILANO 1863

Contents

1 Problem Specification

During the sanitary emergency due to the Covid-19 pandemic, many programs and applications developed thanks to the use of Big Data proved to be particularly effective in different settings and scenarios, such as the hospital administration, the hospitalizations organization or the analysis of data relating to various clinical cases. Among the various areas that have been supported by these technologies there is also that which concerns the tracking of populations belonging to a given geographical area and the collection of all information regarding the tests carried out and the vaccination status.

Our project aims to create an information system suitable for this specific use case and to do this it is necessary to design a database that allows us to store large amounts of data derived from heterogeneous sources and to carry out targeted queries useful for different purposes. The NOSQL document-based approach, known for being based on managing data saved in JSON-like documents, is the optimal one in this case and MongoDB is the open source database management software that is best suited to accomplish this task thanks to the considerable scalability guaranteed by the automatic data sharding and ease of use thanks to the dynamic schemes developed starting from the archived documents.

2 Hypotesis

The way in which the database was structured and implemented is based on some hypotheses discussed in the design phase.

Considering a typical scenario in which each person can interact either with people who live with him (family members or roommates) or with other people who go to meet voluntarily or not in a limited closed place (for example a gym, a public place, a restaurant, etc.) or in an open place located in a generic position, it was assumed that the infection can occur if during the contact between the two subjects (one of which is potentially infected) the distance between them was particularly close and if the contact duration was at least 15 minutes. In addition, to check the state of health of each individual in relation to the pandemic situation, it is essential to consider whether they have been given a vaccine and/or whether they have recently undergone a test to check for any positivity.

Specifically, in order to simulate the pandemic situation in the most realistic way possible and to optimally manage the available data that an information system directly connected to an app supported by the database will have to process, the following assumptions were taken into consideration:

1. people can go to public places even without a green pass (therefore without a vaccine or a negative test carried out in the last 48 hours),
2. data relating to public places are sent to the system by the manager of the public place itself,
3. to record a greater number of relationships to be processed, while ensuring the consistency and meaningfulness of the stored data, it was decided to limit to an example study case in which the interactions present in the dataset are related to the single city of Milan and for a limited time interval, therefore:
    - the tests dates, the meetings between people and the visits to public places took place from the 15th of October to the 15th of November, in addition, to test the queries more easily, there is a specific date, i.e. the 31st of October, that expressely presents multiple recorded interactions,
    - vaccine administrations were all performed within the last year,
    - all public places are located in Milan,
    - meetings between people are geolocated within the city limits
4. with regard to the various connections that can exist between several people and the infections that can occur within the same family, it was preferred, for design purposes, to consider the concept of ”same residential unit” (i.e. residence) rather than that of ”family”, since belonging to a given family unit does not necessarily imply a constant coexistence (e.g. workers who often travel for a job or off-site students) while a relationship of coexistence between several people, not necessarily related to each other, usually involves a close and inevitable contact,
5. concerning to meetings between two or more people, it was assumed that any contact tracing devices/ apps interact with the system, communicating this information for each meeting: person1, person2, time at which the contact took place, geographical point in where the contact took place,
6. a person is considered infected from the moment he undergoes a test and receives a positive result until he takes another test again but it is negative,
7. in the event that a person undergoes a test and receives a positive result, all people met in the 15 days prior to the test will receive an alert notification on the app ensuring the privacy of the infected person is safeguarded,
8. the possibility has also been envisaged in which a person already vaccinated may still be infected, in line with the cases that actually occurred

3 ER diagram

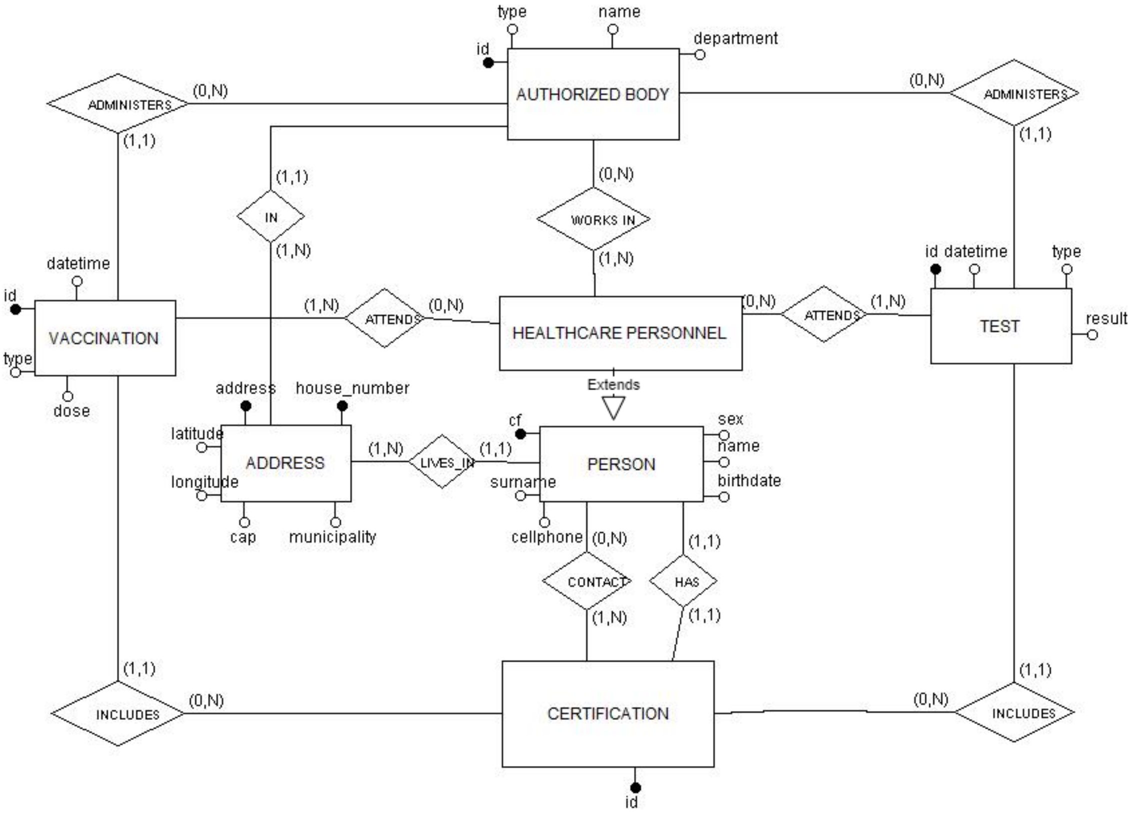

Figure 1: E-R Diagram

Starting from the considerations previously exposed regarding the implementation hypotheses, we have drawn an ER diagram (**Figure 1**) which includes 7 different entities and 4 many-to-many relationships described below in the logical model:

- **Address**(<u>Address</u>, <u>HouseNumber</u>, CAP, Latitude, Longitude, Municipality)
- **AuthorizedBody**(<u>ID</u>, Department, Name, Type)
- **Certification**(<u>ID</u>)
- **HealthcarePersonnel**(<u>CF</u>, Birthdate, Cellphone, Name, Sex, Surname)
- **Person**(<u>CF</u>, Birthdate, Cellphone, Name, Sex, Surname)
- **Test**(<u>ID</u>, Datetime, Result, Type)

- **Vaccination**(ID, Datetime, Dose, Type)
- **AttendsTest**(HealthcarePersonnel.CF, Test.ID)
- **AttendsVaccination**(HealthcarePersonnel.CF, Vaccition.ID)
- **Contact**(Certification.ID, Person.CF)
- **WorksIn**(AuthorizedBody.ID, HealthcarePersonnel.CF)

The **Person** entity describes every possible individual with his own personal data, the **Got** and **GotTested** relationships concern the Covid-related information of each one by means of **Test** and **Vaccine** that represent the actual types of vaccines and tests currently used. The **LivesIn** relationship allow us to keep track of all the people who share the same housing unit specified by the **Home**, while with **WentTo** it is possible to check who was in a specific place identified by **PublicPlace** through an address from a certain time until it went out. Finally, the **Met** relationship is used to keep track of all the people that everyone can meet during the day by recording with whom the meeting took place, when and where based on geographical coordinates, storing data only if the meeting duration was at least 15 minutes accordingly with the hypotheses specified before.

4 Dataset description

One of the most critical parts of working with Big Data is managing large amounts of data collected in large datasets. To test and simulate the use of the database for contact tracing activities, some sample datasets were generated, saved in .csv format and imported into Neo4j through the command: `LOAD CSV FROM "file: ///file.csv" AS ....`

Each dataset is divided into various fields that trace the structure of the tables expressed in the ER model, each one was generated randomly through Python scripts (you can find these ones into the folder `"db"`) and to experiment and perform at best the possible tests on the queries and commands that can be executed thanks to Neo4j the number of entries foreseen for each dataset is in the order of magnitude of the hundreds, as a whole, starting exclusively from the data loaded from the datasets, the database provides:

Homes	101	Residence Relationship	300
People	300	Meetings	726
Public Places	20	Public Places Visits	600
Vaccine Types	4	Vaccinations	526
Test Types	2	Tests	450
TOTAL NODES	**427**	**TOTAL RELATIONSHIPS**	**2602**

5 Queries and Commands

The correct functioning of the information system involves the implementation of some essential commands and queries for the database in order to properly support the app and to ensure the right execution of searches among the data available for statistical or practical purposes.

First of all you need to load the .csv files with the query you can find following the path `"documents/importDB.txt"`

5.1 Queries

5.1.1 Find how many people went without greenpass in a public place

This query allows us to find all the people that went in a public place without the GreenPass and the datetime related.

The output is given by:

- The name and the surname of the person
- The entrance time in the public place
- The name of the public place

5.1.2 Find who lives with an infected individual

This query allows us to detect a ”family contact” with an infected person and obtain the people that need to be quarantined.

The output is given by:

- The house identifier
- The name and the surname of the person
- The name and the surname of the infected cohabitant

5.1.3 Find public place contact with infect people

This query allows us to find the people who went in the same public place (e.g. restaurant, cinema,...) at the same time with infected people starting from 15 days before their positive test.

The output is given by:

- The name and the surname of the person
- The name and the surname of the infected cohabitant
- The datetime of the entrance of the health person
- The datetime of the entrance of the ill person
- The public place

5.1.4 Find who got vaccinated in a temporal range

This query allows us to find all the vaccinated people in the temporal range of October 2021.

The output is given by:

- The name and the surname of the person
- The type of the vaccine
- The datetime of the vaccine

5.1.5 Statistical analysis of the vaccination campaign

This query allows us to compute an analysis on the number of the vaccinated people (and the percentage) grouped by age ranges.

The output is given by:

- Total people
- Total vaccinated people (with %)
- Vaccinated people in age ranges (with %)

5.2 Commands

5.2.1 Federico moves in an other house (CREATE)

In this command we simulate a person moving in an another house.

5.2.2 Delete all the public place records older than 1 year (DELETE)

In this command we simulate the need of removing old records that are not useful anymore.

5.2.3 Name change of public place (UPDATE)

In this command we simulate the need, of a public place manager, of changing name of the activity.

6 UI description & User Guide

6.1 UI description

The design of the information system ended with the development of *ContagionShield*, an application connected to the database with a simple GUI that is very intuitive and easy to use. The UI was created with the Python programming language by means of the libraries: *PySimpleGUI* for the creation of the graphical interface, *Py2neo* to work with Neo4j through the syntax offered by Python and *pandas* to manage the analysis and manipulation of data, you can find these ones into the folder `"contagionshield"`.

As you can see from **Figure 3**, the main screen of the application is divided into two sections: one on the left that allows you to create customizable queries by choosing date, place, time interval and whether to display the vaccinated, non-vaccinated or with negative test, in **Figure 4** there is an example query with the results obtained; the other side section on the right (**Figure 5**) presents some predefined queries, some of them follow the ones specified and described in Chapter 5 of the report.

Finally, as shown in **Figure 6**, it is possible to execute some of the commands already presented in Chapter 5 by reaching the appropriate section of the application by means of the `"Commands"` button in the lower left corner of the main screen. The executable commands allow you to create new meetings between several people by indicating their social security numbers, the date and time, create new visits to public places by specifying who went to a given place and when and in the end you can also delete all the visits to public places recorded in the last year.

6.2 User Guide

In order to run *ContagionShield* it is necessary to verify some requirements and to perform some actions:

- At first you have to check if you have the right Python version installed by using the command: `python --version`, if not you could download it from the official website: https://www.python.org/
- Then make sure your local database is at `localhost:7687` and that it has `"smbud"` as password
- Install the required packages (in the ”contagionshield” folder): `pip install -r requirements.txt`
- Finally make it run by navigating to the folder where you have been saved the materials, then into `"contagionshield"` folder and run: `python contagionshield.py`
- From there you can execute queries about the collected data

7 References & Sources

[1] Course Slides

[2] https://pysimplegui.readthedocs.io/en/latest/call

[3] https://py2neo.org/

[4] https://neo4j.com/docs/cypher-manual/current/

[5] https://neo4j.com/developer/python/

[6] http://iniball.altervista.org/Software/ProgER

[7] https://neo4j.com/developer/cypher/

[8] https://pandas.pydata.org/docs/