# Problem Statement

A client in the area of palliative care in the rural sector is concerned about the multiple potential contact points for patients and their carers. Examples of these carers could be. family, neighbours or friends. There is a desire for a system to ensure a single point of contact for the patients/carers. This point of contact would act as a triage point, and provide advice, refer the patient to a higher level of advice, or advise the patient to seek external treatment. This system would need to store information about the contact, including who called, who answered the contact, and the advice given or actions taken.

# ERD Diagram

Diagram

Description automatically generated

## Design decisions and normalisation

I believe that the **patient** must provide all the information, namely

* Full name and surname
* Date of Birth
* full address, including the region, as it is important to know if the patient lives in rural area.
* home phone, because patients are elderly, sometimes they reject technology, it is difficult for them to adapt to technology due to their advanced age.
* mobile phone
* email (who knows, maybe there are qualified IT specialists among the patients)
* notes - for more information

**Carer**

For the **carer** table, I decided that the following data would be needed

* Full name and surname
* Date of Birth
* Full address, including region and district
* Home phone
* Mobile phone
* Email
* Notes - for additional information

The **contactor** can be both a patient and a carer (if you remove all other persons who can call, but I believe that such persons will not be recorded). Table contactor should contain carerId and patientId. I think it is very important to specify that Contactor should be patient OR carer, I mean on what behalf caller is calling. Can it be a device? Like patient will have a button on special equipment (I am imagining a hospital bed with a button to call a nurse) to make call, if patient cannot talk for example. It will provide special message automatically which will contain just a patientId.

Table **worker** should contain standard information full name and surname, mobile(work) phone. An additional field that I believe should be present is the worker's availability, as it may not be available. At the moment it's Boolean, but maybe it's worth changing to two fields instead of one, one of them is the date of the available from, the other is available to. Additional information such as address, etc. I consider irrelevant in this situation, as it may be stored elsewhere.

**Contact** table has a contactID as a primary key, and contactorID, patientId, workerId as foreign keys referencing the corresponding tables. It has a date and time when the contact was made, what the reason for calling, actions that were taken and any additional notes if required.

**Bridging** table.

There is a bridging table called **PatientCarers**, which has patientId and carerId as both Primary and Foreign keys. The reason for creating this table was that a patient could have multiple carers, or one carer could look after multiple patients. This way, a patient cannot have only one foreign key of a carerId in its table, as it would create repetitive rows. The bridging table would deal with this problem and allow many-to-many relationship between Carer and Patient tables. This table also has an attribute that specifies in what relationship the carer and patient are.

# Security and GDPR

**Privacy Act 2020 and the Privacy Principles** – looks very similar to me, it was not in my plans to violate these principles and laws. There is no point to dive deeper into Privacy act, because all information can be found there

https://www.privacy.org.nz/privacy-act-2020/privacy-principles/

## GDPR

General Data Protection Regulation (GDPR) is the European Union privacy law that came into force in 2018 and requires all the companies that hold or process the personal data of people residing in the European Union will be required to comply with the Regulation (Intersoft Consulting, 2020).

There is a possibility that some of the patients might be EU residents, so that is why the system should be compliant with GDPR. There is a lot of personal data (data, that can be used to identify an individual) stored in the database, such as: names, addressed, dates of births.

To make a database GDPR compliant, I will have to ensure the following steps are implemented:

* **Create and enforce roles and permissions.** By defining roles and permissions on the database level in advance, I can prevent unauthorized access to sensitive personal data.
* **Mask sensitive data.** Data masking within the database ensures that the developers and engineers can use real data when working on their databases, without compromising any individual’s privacy or breaching any articles of the GDPR.
* **Prevent data loss and alteration.** Maintaining the integrity of data is just as important as the security measures used to collect and store it. I plan on implementing hashing and encryption in the database.

## GDPR Compliance

In Iteration 1, the database was not GDPR compliant as it did not have any security measures implemented.

In iteration 2 security measurements were implemented. Roles and users were created, rights were assigned.

# Coding

### Iteration 1

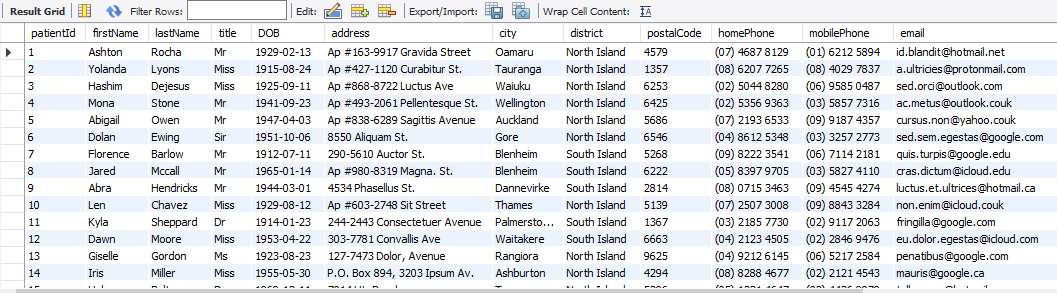
In Iteration 1, I implemented the following tables:

* Carer
* Patient
* Contactor
* Contact
* Worker
* PatientCarer(bridging Table)

I generated sample data on generatedata.com and inserted it into tables:

**Patient, Carer, Worker** tables I have filled with data by using a very simple tool Data Table Wizard Import,which is part of Mysql WorkBench!

**Patient**



**Contact**

Graphical user interface, application

Description automatically generated

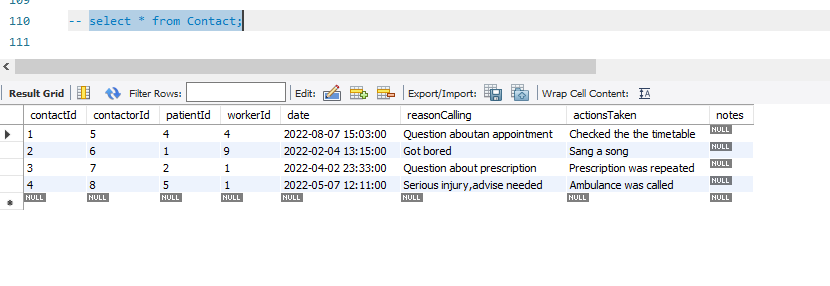
**Carer**

Graphical user interface, application

Description automatically generated

Unfortunately, I could not do the same for the rest of the table, due to tool restrictions. I had to do it manually by inserting data with a query.

**Contact**



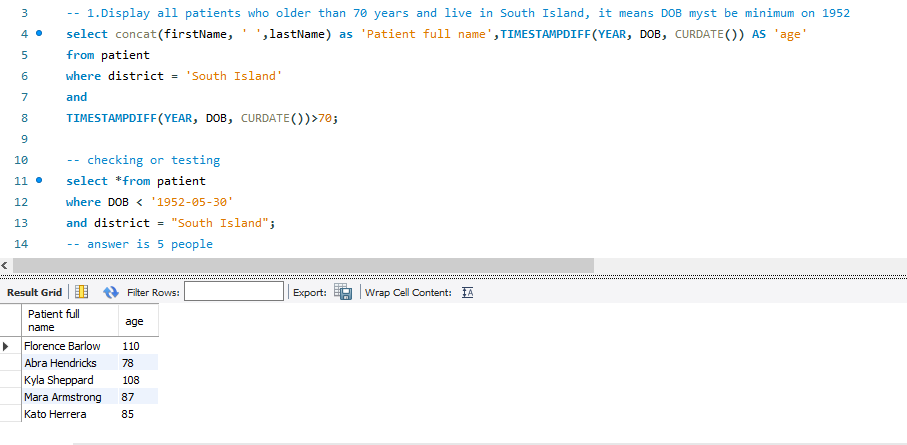
Example of inserting

Text

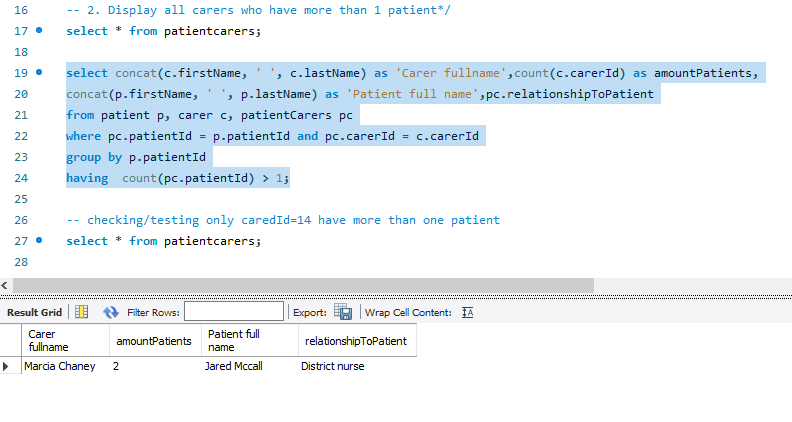
Description automatically generated

## **Queries**

Query 1. Display all patients who older than 70 and live in South Island

****

Query 2. Display all carers who has more than 1 patient



Query 3.Display how many times particular carer made a contact.

Graphical user interface, text, application

Description automatically generated

Query 4. Displaying important information about patient through a VIEW

Graphical user interface, text, application

Description automatically generated

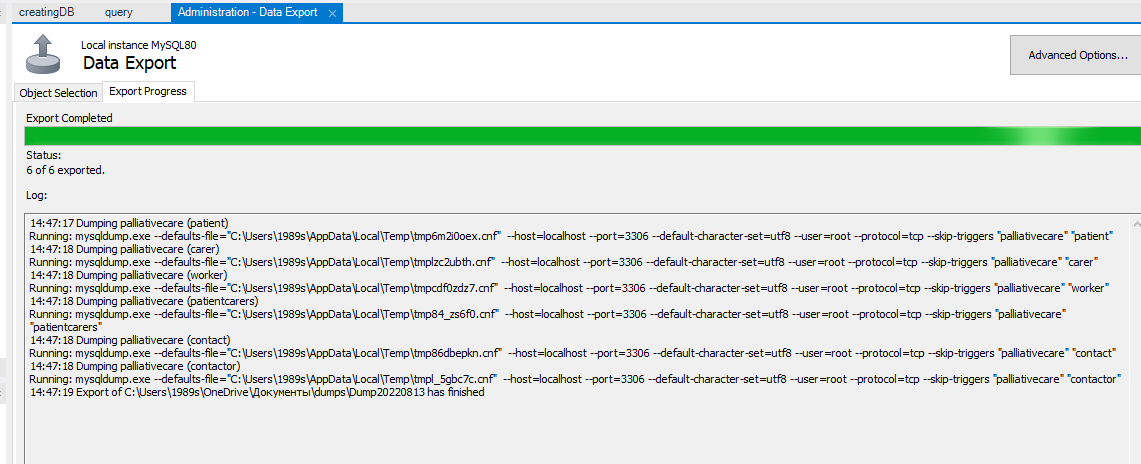
**Graphical user interface, table

Description automatically generated**

# Backup

A backup of all personal and general data should be made at the beginning, immediately after the information has been received, as unforeseen circumstances can happen at any time.

**Prove of a backup**



Text

Description automatically generated

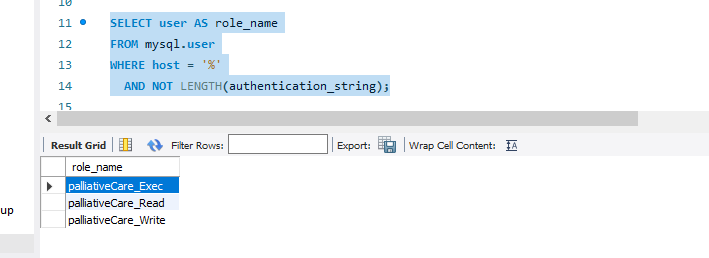
Unfortunately, my version of Mysql does not have a proper tool MysqlEnterpriseBackupRecovery.I had to export (dump it) it instead.

# Security/coding

**Creating roles for a database**

A picture containing text

Description automatically generated

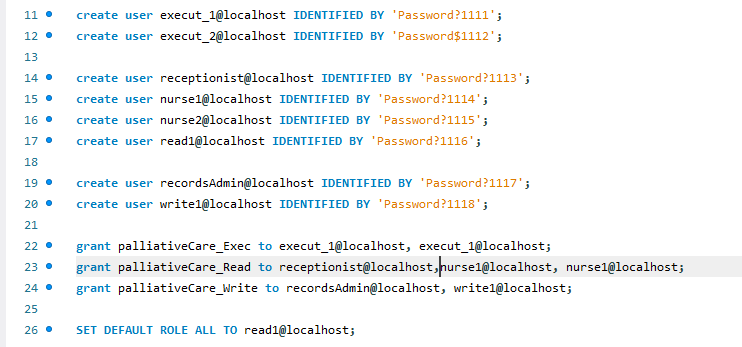


**Granting access rights to roles**

Graphical user interface, text, application

Description automatically generated

**Creating users and granting roles to them**



Prove

Graphical user interface, text, application

Description automatically generated

# Evaluation

The hardest part is always getting started.

**Iteration 1** performance was satisfying. It was hard to divide all the work into equal iterations as some things like security or views are hard to do without fully implementing the database.

It was difficult to come up with a relevant field for tables, cutting off everything unnecessary and non-relevant. Since I have no experience specifically in this industry, it is quite possible that I missed some important information. I hope that in the next iteration I will correct my shortcomings.

For the future Iteration I am planning to secure all data, do a backup again and draw a normalization diagram.

**Iteration 2** was easier and quicker. There is no alterations to database itself or ERD. But there are significant changes to security. The database is now more GDPR-compliant with the roles and user accounts created and assigned. In the next iteration, I plan to look into encrypting and creating more views and indexes to check queries for performance.