1. INITIAL PROJECT DESCRIPTION

We are going to create a database which use is going to be directly related to aliments. The goal of our database is to provide a determined patient with a diet following its main characteristics (age, height, weight) Therefore, we will need a group of nutritionists. They are the ones who will be in charge of the development of the diets. We will storage data for aliments mainly, and we will use this data in order to create new diets for our patients in the nutritional center.

We will use a relational database, a type of database that stores and provides access to data points that are related to one another.

In order to generate data for our database, we will use csv files for some of our tables. For the rest of the tables, we will try to generate data automatically with the tools we will see in class (Python)

1. RDBMS deployment

In the schema of our database, we will have the following tables: Food, Patient, Nutritionist, Diet and List\_Of\_Food. One Diet will consist of different kinds of Food, a Diet will be created by a Nutritionist (although a Nutrionist can create more than one) and a Nutritionist will take care of one or more Patients. The table List\_Of\_Food is created as a consequence of the relation between Diet and Food (a relation M:N creates a new table including the primary keys of both tables included in the relation)

Firstly, we have introduced the table Food, which has the food ID as the primary key. The rest of columns of the table are related with information such as the energy in kcal and KJ that provides the food, the quantity of Protein and Fiber in food in every 100 grams…

Then the table to register the information about the Patient has been created and fulfilled with relevant information about it in order to organize the new diet. It also stores the current diet that the person is following and the intolerances, information that can be quite useful in order to design a new diet.

The table of the Nutritionist is created also with an ID as the primary key. It also contains information about the name of the nutritionist and the specialty that he or she has (obesity, for example)

For the diets we have another table. As an identifier, we have and ID for the diet (primary key) Also, a description and a Creator are columns for this table. The Description will contain the different steps to follow in order to do the diet.

The data which has been introduced in the table Food has been obtained by a csv file in https://www.fao.org/infoods/infoods/tablas-y-bases-de-datos/es/

1. QUERY DESIGN

· Food with more than 20 grams of protein:

* SELECT NutritionalDB.Food.Name

FROM NutritionalDB.Food WHERE Food.Protein > 20;

Tabla

Descripción generada automáticamente

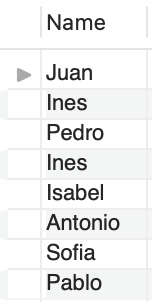
· All the names of patients treated by the doctor Pablo Moreno Garcia-Espina:

* SELECT NutritionalDB.Patient.Name

FROM NutritionalDB.Patient JOIN NutritionalDB.Nutritionist

ON Patient.Doctor = Nutritionist.idNutritionist

WHERE Nutritionist.FullName = 'Pablo Moreno Garcia-Espina';



· Description of the diets created by the doctor Susana Rocio Fernandez Giaccomassi

* SELECT NutritionalDB.Diet.Description

FROM NutritionalDB.Diet JOIN NutritionalDB.Nutritionist

ON Diet.Creator = Nutritionist.idNutritionist

WHERE Nutritionist.FullName = 'Susana Rocio Fernandez Giaccomassi';

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

· Obtain the description and the name of their creator of all diets with more than 10 foods:

* SELECT NutritionalDB.Diet.Description as 'Diets with more than

ten foods', NutritionalDB.Nutritionist.FullName as 'Creator'

FROM NutritionalDB.Diet JOIN NutritionalDB.Nutritionist

ON Diet.Creator = Nutritionist.idNutritionist

WHERE NutritionalDB.Diet.idDiet IN (

SELECT NutritionalDB.Diet.idDiet

FROM NutritionalDB.ListOfFood

GROUP BY idDiet HAVING COUNT(\*) > 10

);

Texto

Descripción generada automáticamente

· Get the names and protein content of foods that have more protein than the average protein of all foods:

* SELECT NutritionalDB.Food.Name, NutritionalDB.Food.Protein

FROM NutritionalDB.Food

WHERE NutritionalDB.Food.Protein > (

SELECT AVG(NutritionalDB.Food.Protein)

FROM NutritionalDB.Food

Tabla

Descripción generada automáticamente);

1. RDBMS Optimization

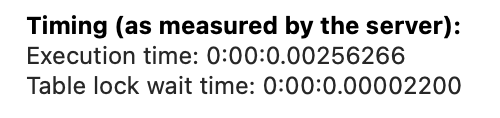
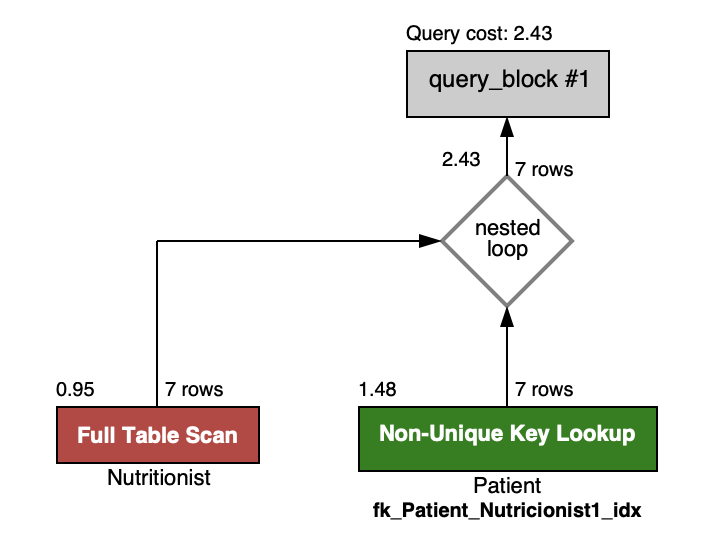
Firstly, we have changed the first query in order to make a deeper search. We can also search in the database according to the type of food, as we made a join between the table of Food and the table of Type of Food.

In order to make the search easier when it comes to diets related with sporty people, we have decided to design two indexes: one with Name and Kcal and another with Protein and Name. This index becomes useful with our first query, as we are looking firstly at the food with protein’s value of more than 20 grams.

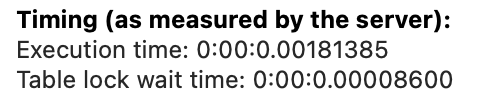
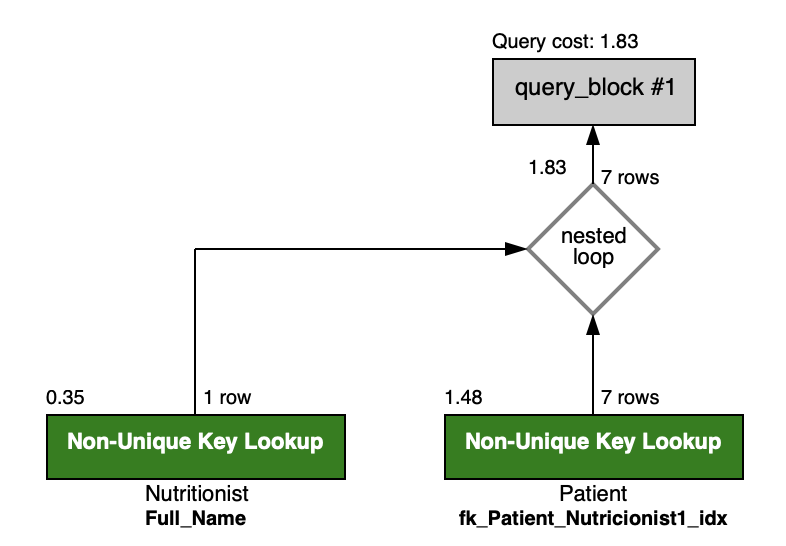
For the previous index, when we try to execute the first query apparently the index is not used. That could be because the improvement when the index is used is not considerably high when you compare it with the performance without the index.

Furthermore, as we will frequently look for the different diets that one nutritionist has created, we will create an index for the column “Full Name” in the table Nutritionist. We already have an index for the id of a nutritionist, but we will typically get a diet of a nutritionist by his concrete name. With this index we will get an advantage in the second and third query, as they use the full name of the nutritionist in order to make the query.

This picture shows the time which lasts the second query without the index.



This picture shows the time which lasts the second query with the index.



We see how the time elapsed in the second time is lower than in the first time. The existence of the index makes the performance of the query higher.

We will also do queries which involve the name of the different patients we have in our database. Therefore, it would be suitable too to make an index with the name of the patients. Every time that we do a query in which the name of the patient is involved, we will get an advantage in terms of time, as we can see before.

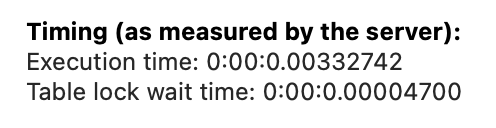
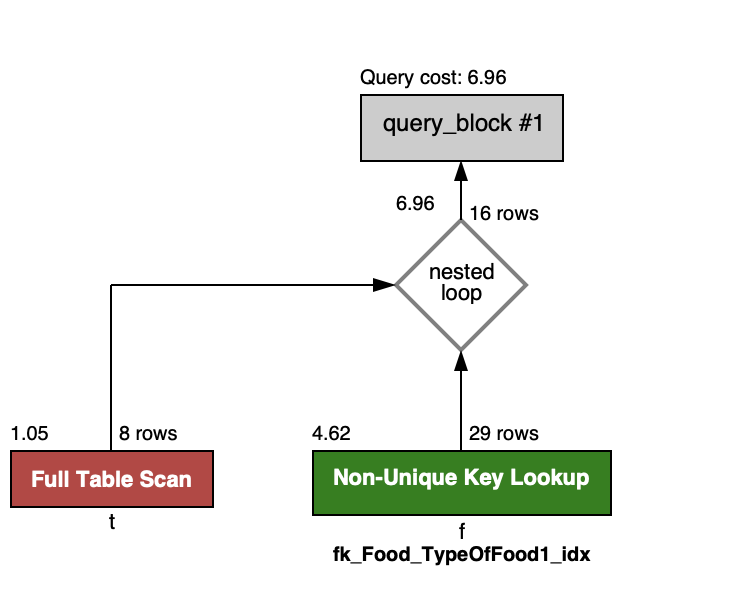
We think that nutritionists may want to filter the aliments by the nutritional information and origin, so the query would be:

* SELECT f.Name

FROM Food f

JOIN TypeOfFood t ON f.TypeOfFood = t.Type

WHERE t.Origin = 'animal' AND (f.Protein > 15 OR f.Iron > 5);



In this case, we have created an index for the table Food including the columns Protein and Iron, but it did not used it, neither the protein index created before.