Unicamp - MC536 - Projeto Final

Equipe Dinossauros, Bancos de dados e Coisas Parecidas (DBDCP)

Davi Gabriel Bandeira Coutinho (183710)
Francisco Vinicius Sousa Guedes (260440)
Márcio Levi Sales Prado (183680)

Motivação e Objetivo

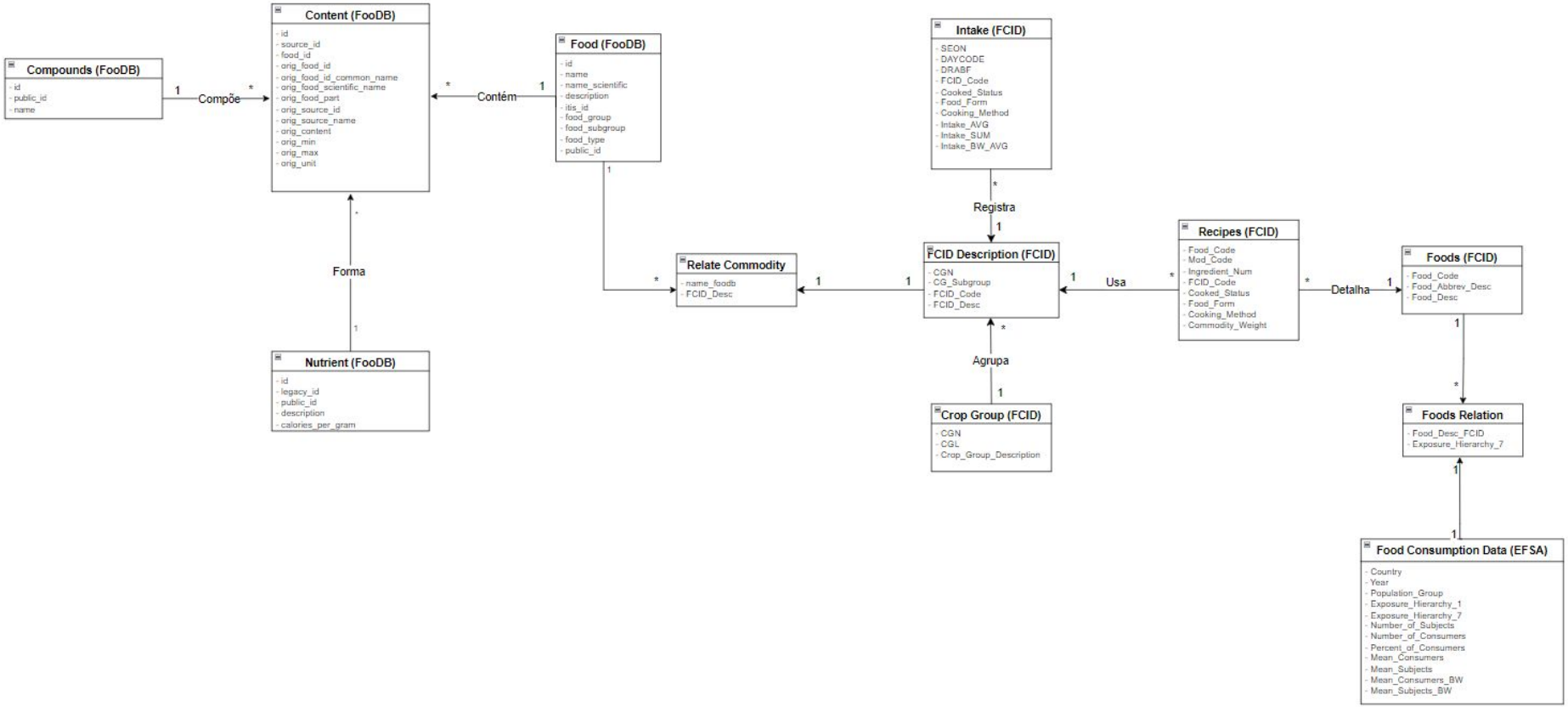
Com a constante aceleração do dia a dia, muitas pessoas optam por preencher suas refeições com comidas rápidas e processadas, fazendo **escolhas não muito saudáveis**.

O desenvolvimento e estudo dos **hábitos alimentares da população** também trazem à tona outras questões, desta vez relacionadas à saúde dos consumidores já que a alimentação de uma pessoa pode nos dizer bastante sobre sua saúde, como por exemplo indicar a **deficiência de algum nutriente**.

Nosso objetivo então é :

fornecer informações nutricionais sobre os hábitos alimentares da população alvo da pesquisa, realizando uma conexão entre as bases de dados de consumo com o FooDB**.** Também buscaremos **trazer informações sobre receitas de alimentos,** a partir da análise dos dados que detalham os ingredientes que as compõem.

Modelo Lógico Conceitual



Modelo Lógico Relacional (Parte 1)

FOOD_CONSUMPTION_DATA(_Country_, _Year_, _Population_Group_, Exposure_Hierarchy_1, _Exposure_Hierarchy_7_, Number_of_Subjects, Number_of_Consumers, Percent_of_Consumers, Mean_Consumers, Mean_Subjects, Mean_Consumers_BW, Mean_Subjects_BW)

FOODS_RELATION(_Food_Desc_FCID_, _Exposure_Hierarchy_7_)
  Food_Desc_FCID chave estrangeira -> FOODS(Food_Desc)
  Exposure_Hierarchy_7 chave estrangeira -> FOOD_CONSUMPTION_DATA(Exposure_Hierarchy_7)

FOODS(_Food_Code_, Food_Abbrev_Desc, Food_Desc)

RECIPES(_Food_Code_, Mod_Code, Ingredient_Num, _FCID_Code_, Cooked_Status, Food_Form, Cooking_Method, Commodity_Weight)

FCID_DESCRIPTION(_FCID_Code_, FCID_Desc, CGN, CG_Subgroup)

INTAKE(_SEQN_, _DAYCODE_, DRABF, FCID_Code, Cooked_Status, Food_Form, Cooking_Method, Intake_AVG, Intake_SUM, Intake_BW_AVG)
  FCID_Code chave estrangeira -> FCID_DESCRIPTION(FCID_Code)

Modelo Lógico Relacional (Parte 2)

CROP_GROUP(_CGN_, _CGL_, Crop_Group_Description)
  CGN chave estrangeira -> FCID_DESCRIPTION(CGN)

RELATE_COMMODITY(_name_foodb_, _FCID_Desc_)
  name_foodb chave estrangeira -> FOOD(name)
  FCID_Desc chave estrangeira -> FCID_DESCRIPTION(FCID_Desc)

FOOD(_id_, name, name_scientific, description, itis_id, food_group, food_subgroup, food_type, public_id)

NUTRIENT(_id_, legacy_id, public_id, description, calories_per_gram)

COMPOUNDS(_id_, public_id, name)

CONTENT(_id_, source_id, _food_id_, orig_food_id, orig_food_id_common_name, orig_food_id_scientific_name, orig_content, orig_min, orig_max, orig_unit)
  id chave estrangeira -> NUTRIENT(id)
  id chave estrangeira -> COMPOUNDS(id)
  food_id chave estrangeira -> FOOD(id)

Matches entre Bases Diferentes (TF-IDF)

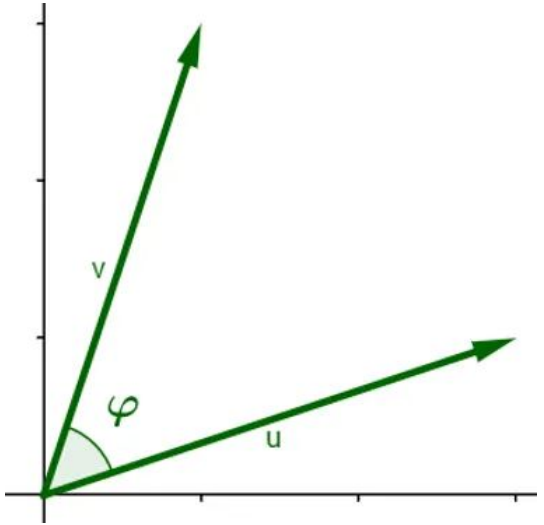
#1: Iniciamos carregando as 2 tabelas que vão ser comparadas.
#2: Desconsideramos letras maiúsculas, igualando-as a letra minúsculas
#3: Criamos o dicionário a partir das palavras que aparecem na base escolhida por referência
#4: Fazemos o encaixe das palavras no dicionário
#5: Criamos o vetor de presença de palavras
#6: Seleciona o melhor vetor, utilizando o arg_max sobre a similaridade do cosseno dada por cosine_similarity(v1, v2)

Count Vectorizer

	blue	bright	sky	sun
Doc1	1	0	1	0
Doc2	0	1	0	1

TD-IDF Vectorizer

	blue	bright	sky	sun
Doc1	0.707107	0.000000	0.707107	0.000000
Doc2	0.000000	0.707107	0.000000	0.707107



Matches entre Bases Diferentes (TF-IDF)

Carregamento

Normalização

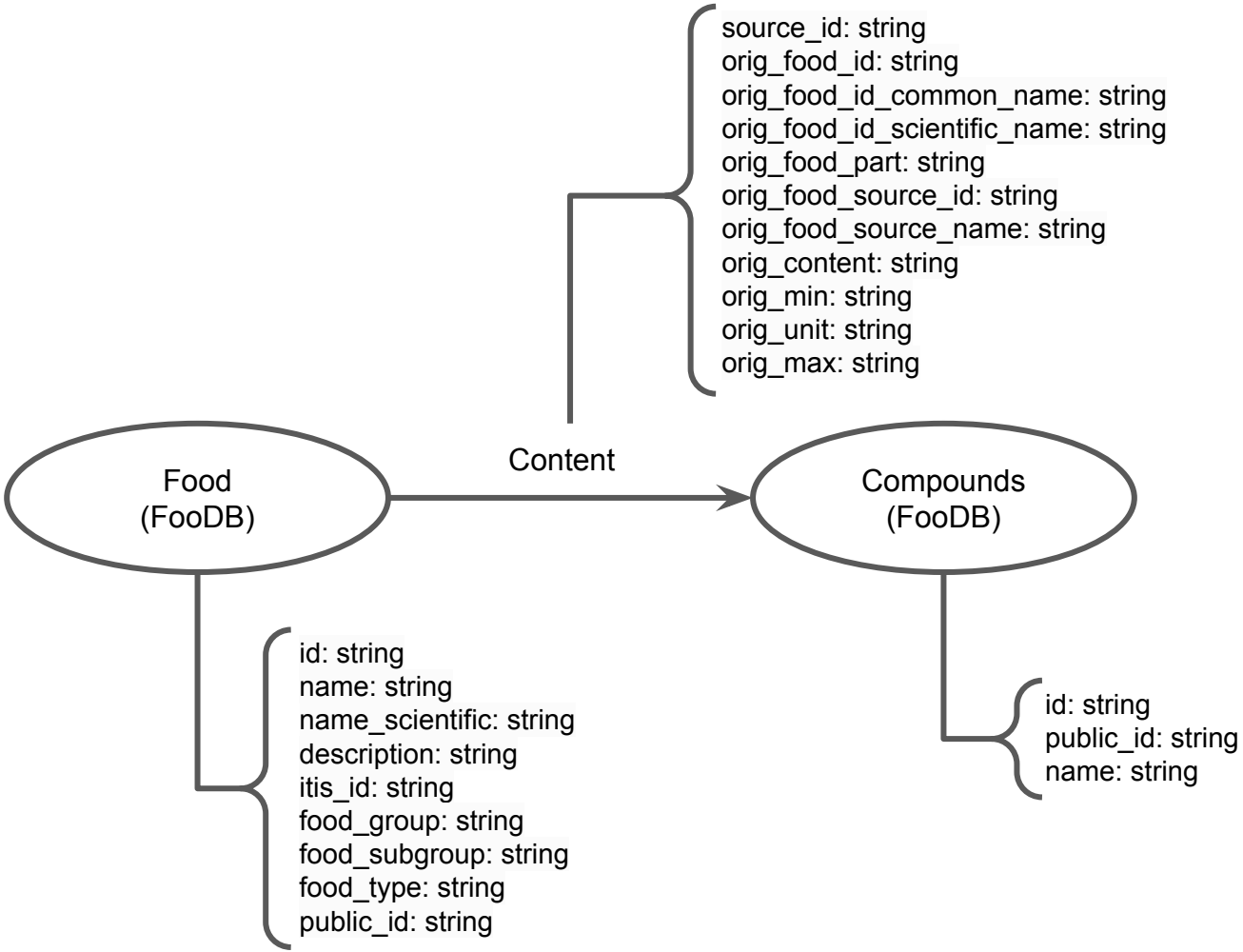
Dicionário e Conversão

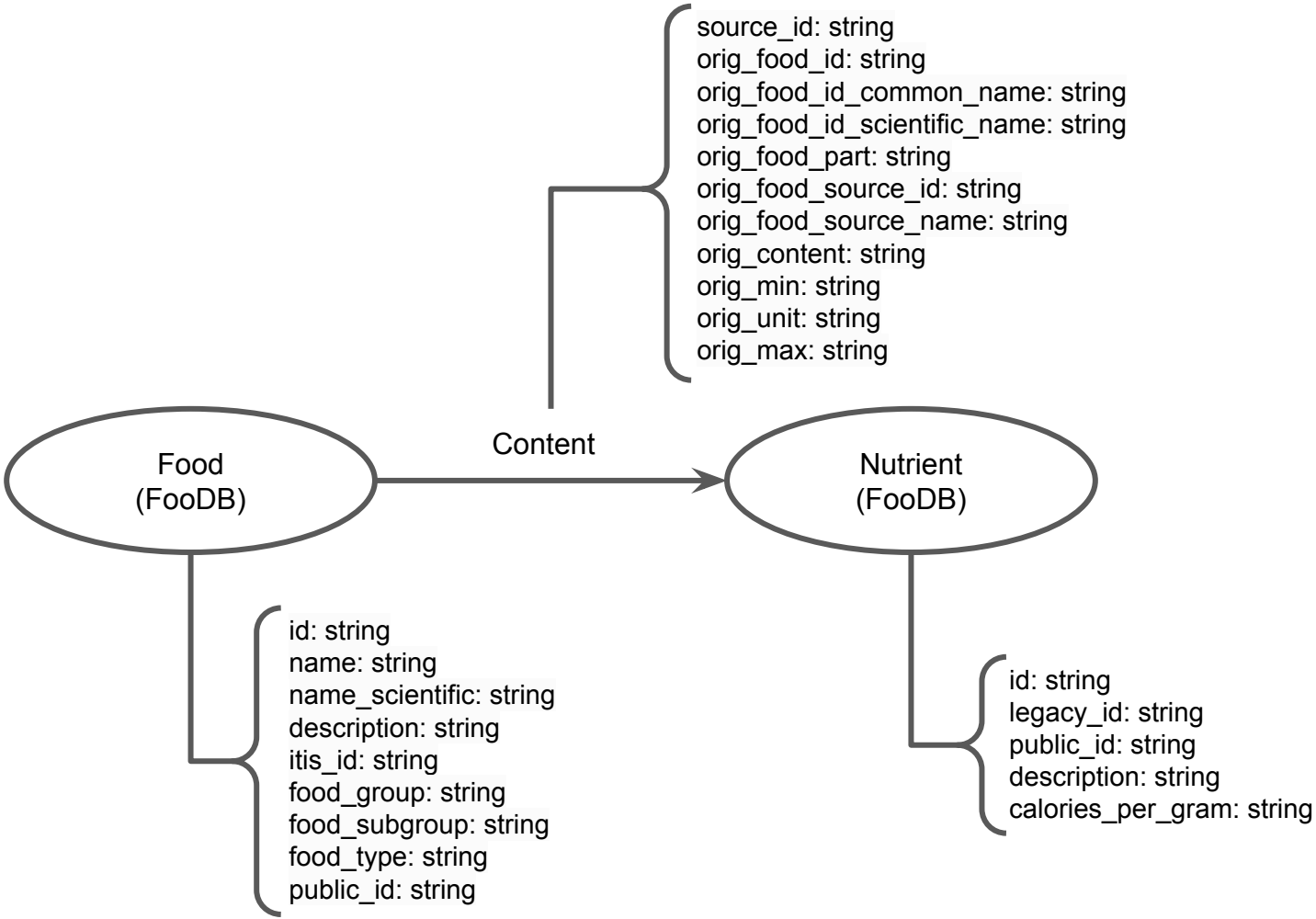
Similaridade de Cosseno

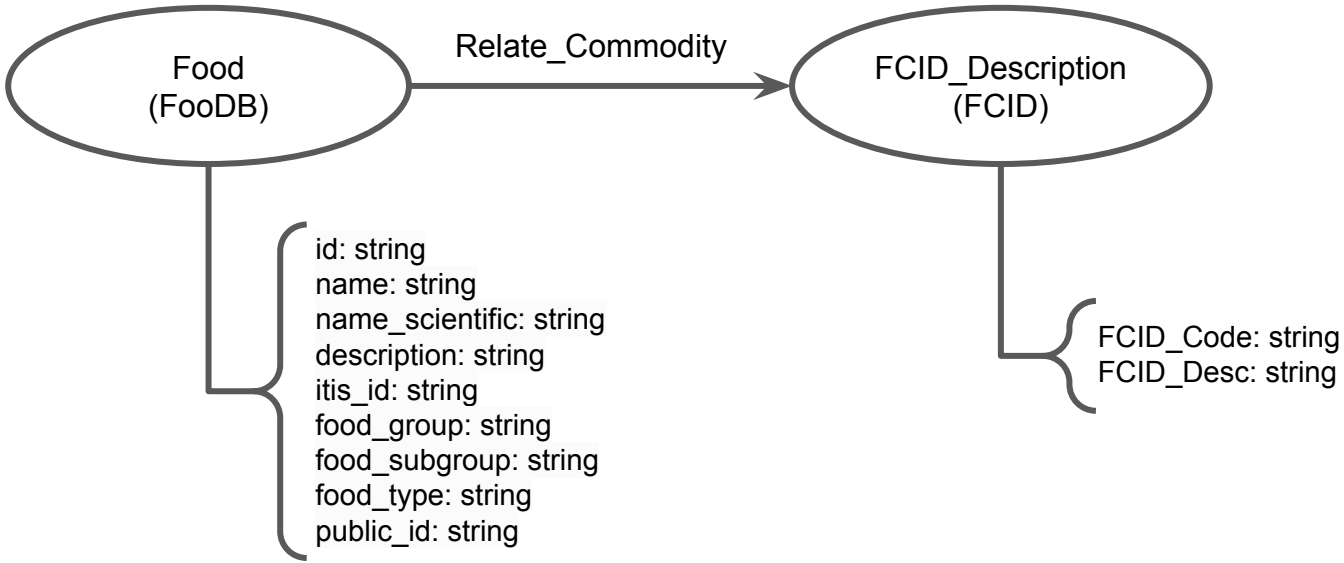
Selecionando o melhor match

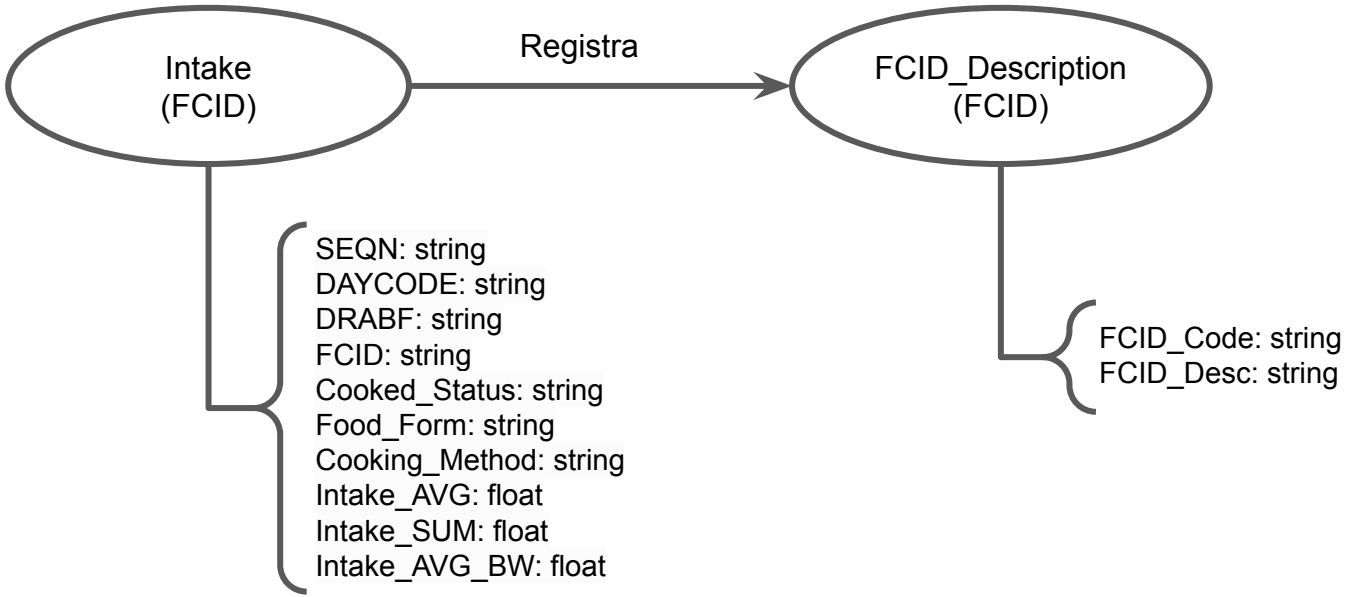
Gerando output

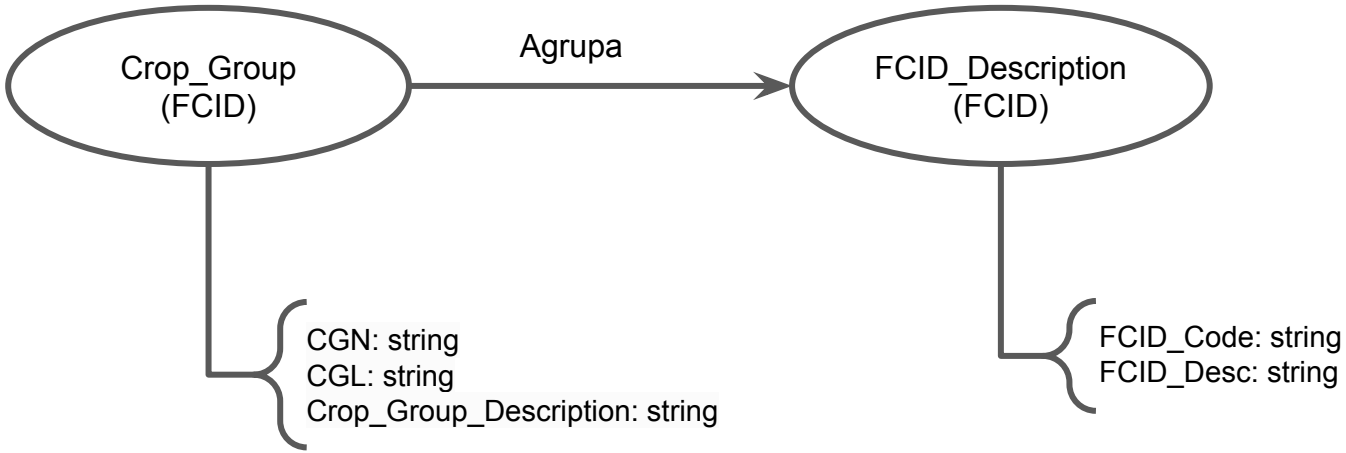
```python
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity

# Carregar dados do FCID e do FoodDB (Configurar o ambiente para estar na raíz do repositório.)
fcid_data = pd.read_csv('bases/fcid/FCID_Code_Description.csv')
fooddb_data = pd.read_csv('bases/foodb/Food.csv')

# Pré-processamento: converter todos os nomes para minúsculas
fcid_data['processed'] = fcid_data['FCID_Desc'].str.lower()
fooddb_data['processed'] = fooddb_data['name'].str.lower()

# Usar TF-IDF para converter os nomes em vetores
vectorizer = TfidfVectorizer()
tfidf_matrix = vectorizer.fit_transform(fcid_data['processed'].tolist() + fooddb_data['processed'].tolist())

# Calcular a similaridade de cosseno
cosine_sim = cosine_similarity(tfidf_matrix[:len(fcid_data)], tfidf_matrix[len(fcid_data):])

# Encontrar a melhor correspondência para cada alimento do FCID
matches = []
for idx, row in enumerate(cosine_sim):
    best_match_idx = row.argmax()
    fcid_food = fcid_data.iloc[idx]['FCID_Desc']
    fooddb_food = fooddb_data.iloc[best_match_idx]['name']
    matches.append((fcid_food, fooddb_food))

# Criar um DataFrame com as correspondências
matches_df = pd.DataFrame(matches, columns=['FCID_Food', 'FoodDB_Food'])

# Salvar o DataFrame em um arquivo CSV
matches_df.to_csv('bases/relação/food_matches.csv', index=False)
```

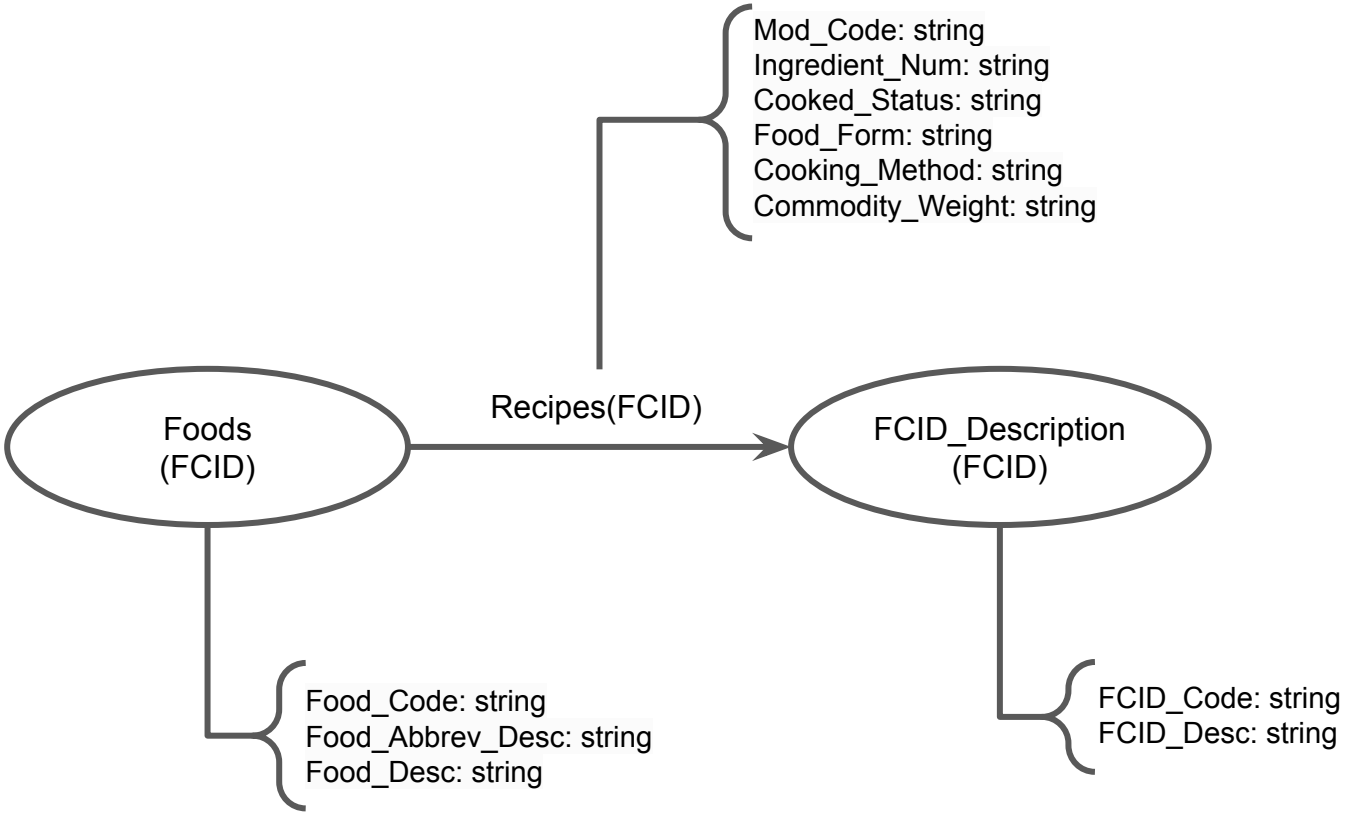

source_id: string
orig_food_id: string
orig_food_id_common_name: string
orig_food_id_scientific_name: string
orig_food_part: string
orig_food_source_id: string
orig_food_source_name: string
orig_content: string
orig_min: string
orig_unit: string
orig_max: string
Content
Food
(FooDB)
Compounds
(FooDB)
id: string
name: string
name_scientific: string
description: string
itis_id: string
food_group: string
food_subgroup: string
food_type: string
public_id: string
id: string
public_id: string
name: string

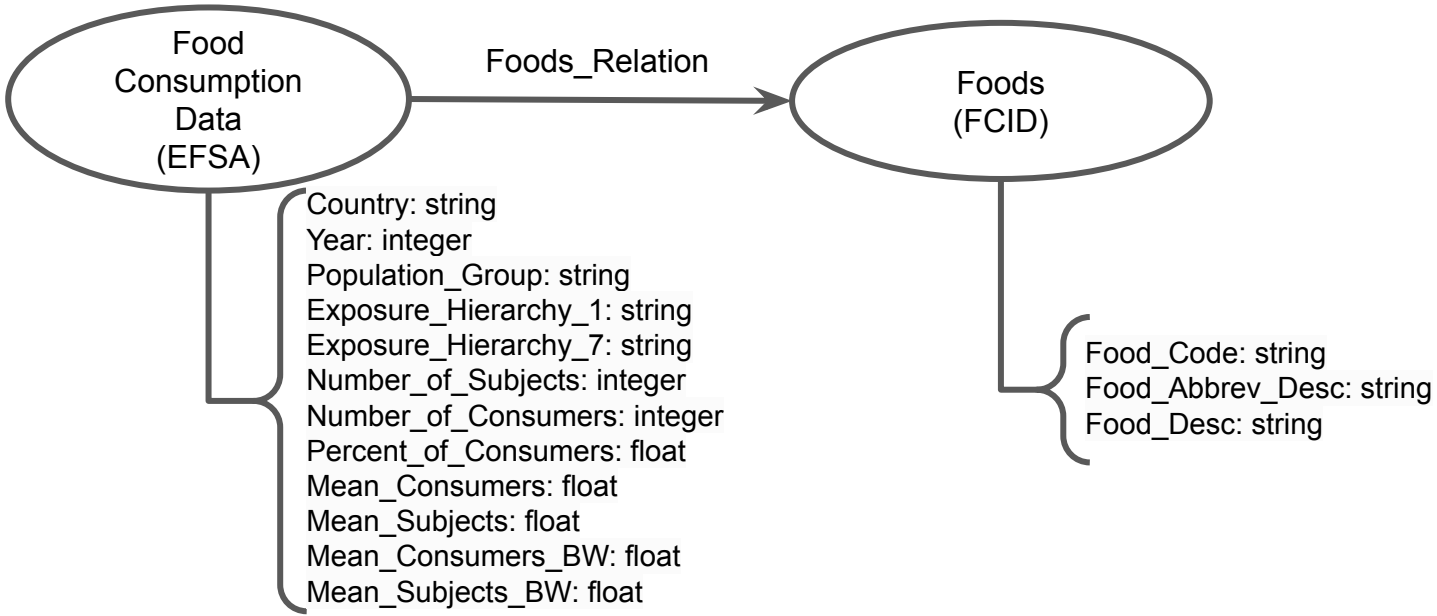

source_id: string
orig_food_id: string
orig_food_id_common_name: string
orig_food_id_scientific_name: string
orig_food_part: string
orig_food_source_id: string
orig_food_source_name: string
orig_content: string
orig_min: string
orig_unit: string
orig_max: string
Content
Food
(FooDB)
Nutrient
(FooDB)
id: string
name: string
name_scientific: string
description: string
itis_id: string
food_group: string
food_subgroup: string
food_type: string
public_id: string
id: string
legacy_id: string
public_id: string
description: string
calories_per_gram: string


Food
(FooDB)
Relate_Commodity
FCID_Description
(FCID)
id: string
name: string
name_scientific: string
description: string
itis_id: string
food_group: string
food_subgroup: string
food_type: string
public_id: string
FCID_Code: string
FCID_Desc: string


Intake
(FCID)
Registra
FCID_Description
(FCID)
SEQN: string
DAYCODE: string
DRABF: string
FCID: string
Cooked_Status: string
Food_Form: string
Cooking_Method: string
Intake_AVG: float
Intake_SUM: float
Intake_AVG_BW: float
FCID_Code: string
FCID_Desc: string


Crop_Group
(FCID)
Agrupa
FCID_Description
(FCID)
CGN: string
CGL: string
Crop_Group_Description: string
FCID_Code: string
FCID_Desc: string

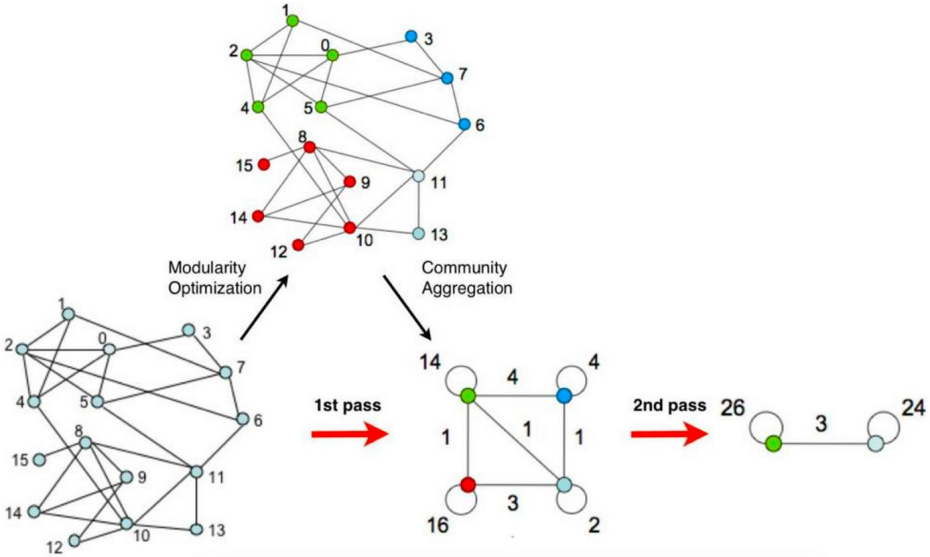

Mod_Code: string
Ingredient_Num: string
Cooked_Status: string
Food_Form: string
Cooking_Method: string
Commodity_Weight: string
Foods
(FCID)
Recipes(FCID)
FCID_Description
(FCID)
Food_Code: string
Food_Abbrev_Desc: string
Food_Desc: string
FCID_Code: string
FCID_Desc: string


Food Consumption Data (EFSA)
Foods_Relation
Foods (FCID)
Country: string
Year: integer
Population_Group: string
Exposure_Hierarchy_1: string
Exposure_Hierarchy_7: string
Number_of_Subjects: integer
Number_of_Consumers: integer
Percent_of_Consumers: float
Mean_Consumers: float
Mean_Subjects: float
Mean_Consumers_BW: float
Mean_Subjects_BW: float
Food_Code: string
Food_Abbrev_Desc: string
Food_Desc: string

Perguntas Implementadas (Modelo de Grafos)

Pergunta 1) No contexto das receitas de alimentos, é possível localizarmos uma comunidade de ingredientes que aparecem juntos em várias receitas?

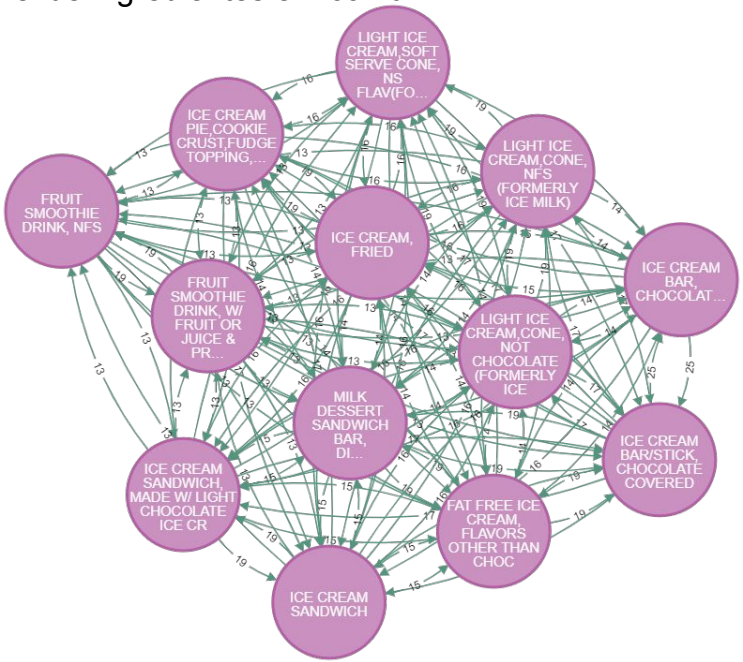
Começamos ligando 2 nós que representam ingredientes sempre que eles estão em uma mesma receita, a fim de aproximar o conceito de 2 ingredientes ligados à ideia de comunidade da pergunta. Utilizamos então, o Algoritmo de Louvain para detectar comunidades nessa nova projeção de grafo:



Perguntas Implementadas (Modelo de Grafos)

Pergunta 2) Dado que uma pessoa gosta de um alimento, é possível prever que ela gostará de um outro alimento?

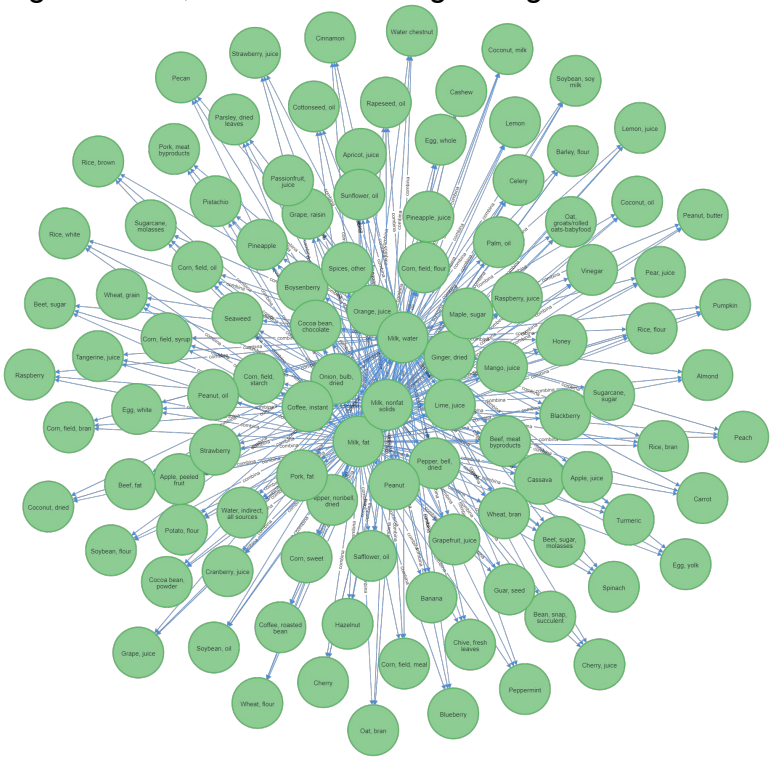
Se uma pessoa gosta de um alimento, certamente é por causa dos ingredientes que estão na receita daquele alimento, e assim ela provavelmente gostaria de um alimento suficientemente similar ao primeiro. Vamos ligar 2 alimentos se suas receitas possuem um número razoável de ingredientes em comum



Perguntas Implementadas (Modelo de Grafos)

Pergunta 3) Qual o ingrediente que mais combina com outros?

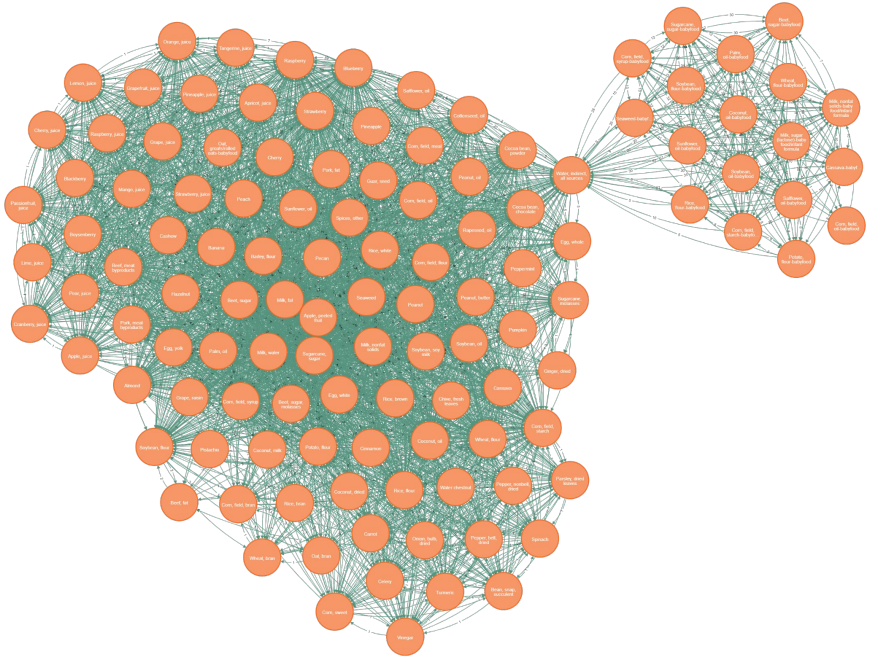
Usamos o conceito de centralidade de grau para detectar ingredientes que estão juntos com outros em muitas receitas.
Achamos 3 nós com centralidade de grau = 105, fornecendo o seguinte grafo isolado



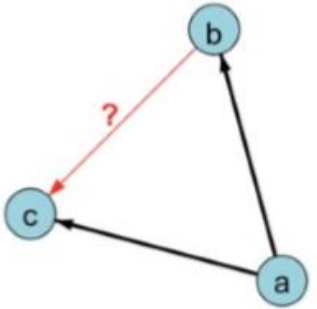
Perguntas Implementadas (Modelo de Grafos)

Pergunta 4) Como se dá a clusterização dos ingredientes, considerando a relação de combinação entre cada par de ingredientes em uma mesma receita?

Analisamos o coeficiente de clusterização local para cada nó, escolhendo apenas aqueles que têm esse coeficiente igual a 1.0 (o máximo), obtivemos um grafo que acaba retratando 2 grupos de comunidades que quase não se intersectam, como segue:



$$C_i := \frac{2 \cdot k(i)}{d_i(d_i - 1)} = \frac{k(i)}{\binom{d_i}{2}}$$



Perguntas não Implementadas (Modelo Relacional)

Pergunta 1)
Como o consumo médio na Europa mudou ao longo dos anos de 1997 até 2019?

Pergunta 2)
Quais os alimentos que mais contribuem para o total calórico de cada receita?

Pergunta 3)
Quais os alimentos que cada grupo populacional mais consome? E qual o perfil calórico desses alimentos?

Pergunta 4)
Para cada grupo de alimento, qual o alimento pertencente a este grupo que é mais consumido?

Pergunta 5)
Qual o alimento mais consumido em cada país, levando em conta o peso do consumidor?
Também, como seria o perfil calórico desses alimentos?

Pergunta 6)
Qual grupo de alimentos é mais utilizado para fazer receitas?