

# Culinária ao Redor do Mundo: Análise de Receitas

Equipe União Vasko:

- \* Breno Shigeki Guimarães Nishimoto - 220599
- \* Gabriel de Carvalho Silva Nascimento - 222103
  - \* Henrique Marques de Martim - 248333
  - \* Leandro Henrique Silva Resende - 213437
- \* Mateus da Costa e Silva Rios Alves de Andrade - 230806
  - \* Matheus Mantovani Meneghel - 230906

# Motivação e Contexto

O tema do projeto foi definido como “O que as pessoas comem ao redor do mundo”. A ideia para o projeto surgiu de uma análise do sistema de comida a quilo brasileiro, em que uma mesma fileira de pratos possuem alimentos de receitas de origens diversas, como o spaghetti italiano ao lado do sushi japonês que por sua vez está ao lado do kibe, que possui origens no Oriente Médio. Diante tal situação, chegamos a diversas perguntas como “o quão semelhante pode ser uma refeição entre as diversas regiões do mundo?” e “como essa diversidade de pratos molda a dieta macromolecular de cada povo?”.



# Bases Utilizadas

GFG04

DINO


## FooDB

<https://foodb.ca>



- FooDB é uma grande base de dados sobre alimentos, sua química, seus ingredientes e nutrientes.
- Alimentação é um dos principais fatores que definem a saúde e a qualidade de vida das pessoas. Dados sobre a constituição de alimentos podem ajudar a relacionar alimentos com saúde.
- Site bem documentado e com interface Web de acesso a dados.

Download: CSV, XML, JSON | API: JSON



Food  
Leek  
Allium porrum

Classification

- Vegetables
- Onion-family vegetables

Macronutrients

- Fiber
- Proteins
- ...

Composition

- Water
- D-Fructose
- ...

GPABD

## CulinaryDB

<https://cosylab.iiitd.edu.in/culinarydb/>



- CulinaryDB é um grande repositório de receitas e ingredientes. As mais de 40 mil receitas contêm informações como a região de origem (dentre as 20 catalogadas), o nome e os ingredientes. São mais de 450 mil registros no total.
- O site apresenta também alguns gráficos com estatísticas básicas sobre os dados.
- Ao integrar os ingredientes e receitas aos bancos de dado base, é possível analisar padrões ao redor do mundo e, por exemplo, entender melhor padrões de macronutrientes nas diferentes regiões do mundo.

Download: CSV



Recipe  
Pão de Queijo  
(Brazilian Cheese Bread)

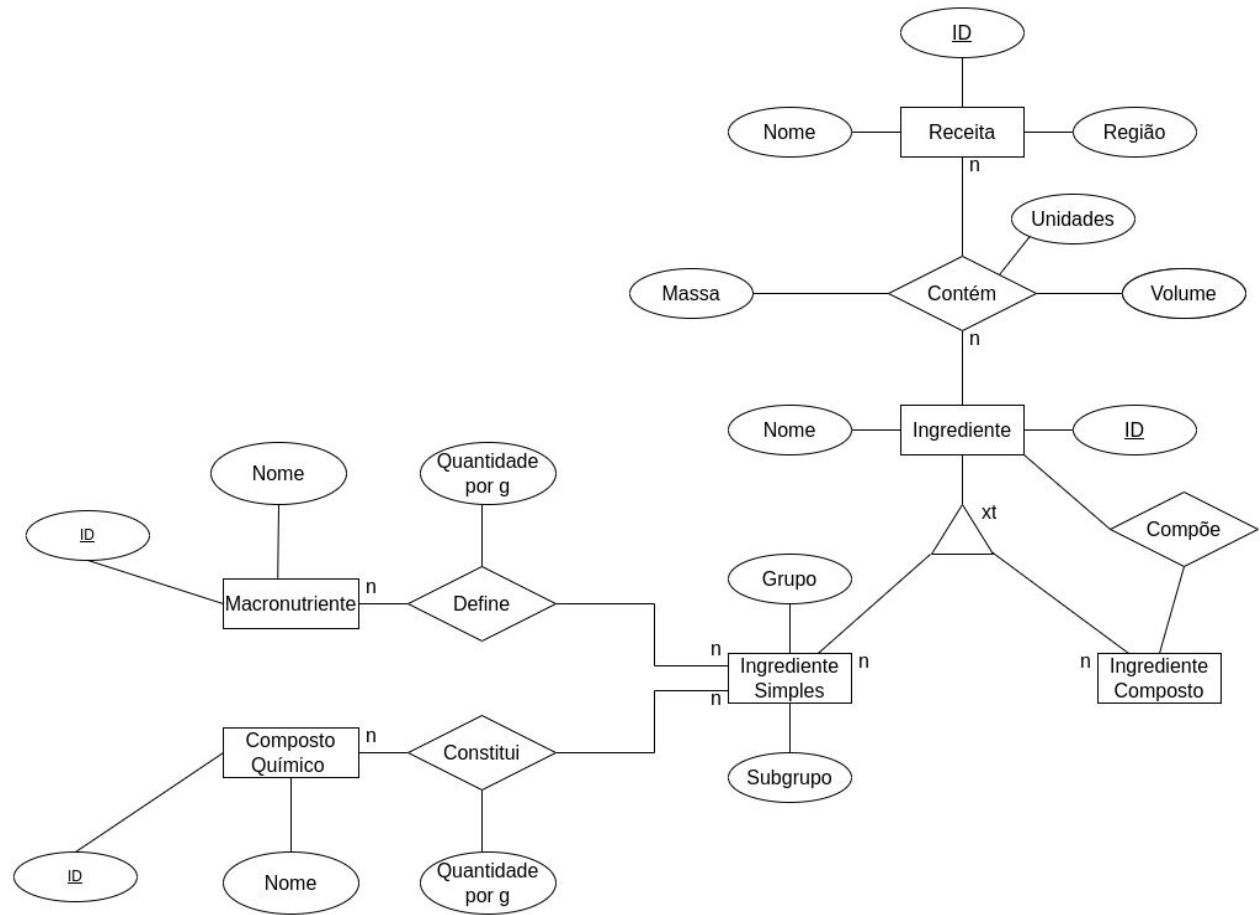
Ingredients

- cheese romano
- egg
- salt
- ...

Regional Cuisine

- South America

# Modelo Conceitual



# Modelo Lógico Relacional

```
IngredienteComposto(_ID_, Nome)
IngredienteSimples(_ID_, Nome, Grupo, Subgrupo)
Receita(_ID_, Nome, Regiao)
CompostoQuimico(_ID_, Nome)
Macronutriente(_ID_, Nome)

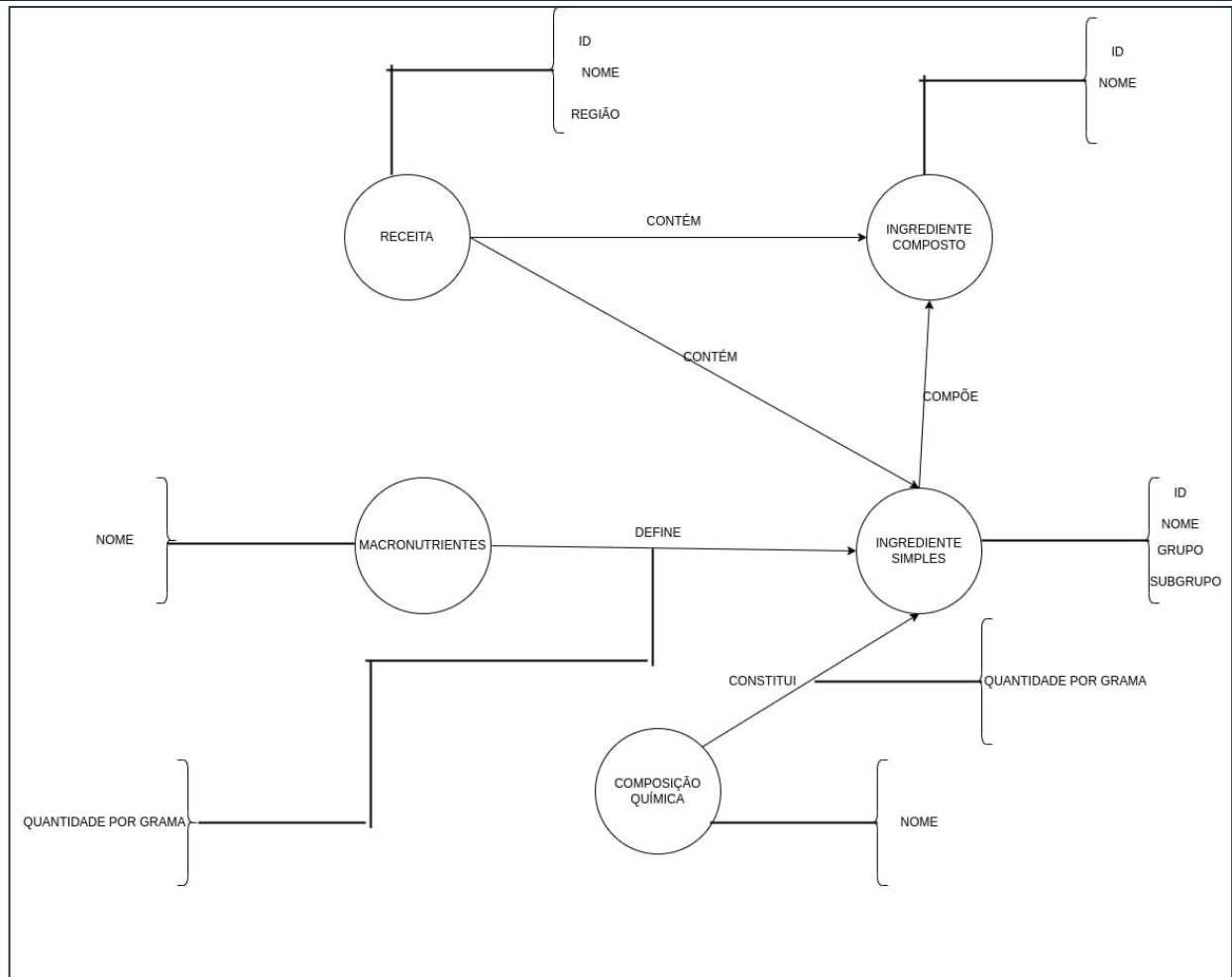
Contem(Receita_ID, Ingrediente_ID, Volume, Unidades, Massa)
    Receita_ID chave estrangeira -> Receita(ID)
    Ingrediente_ID chave estrangeira -> Ingrediente(Nome)

Constitui(_IngredienteS_ID_, _CQ_ID_, QuantidadeGrama)
    IngredienteS_ID chave estrangeira -> IngredienteSimples(ID)
    CQ_ID chave estrangeira -> CompostoQuimico(ID)

Define(_IngredienteS_ID_, _Macro_ID_, QuantidadeGrama)
    Ingrediente_ID chave estrangeira -> IngredienteSimples(ID)
    Macro_ID chave estrangeira -> Macronutriente(ID)

Compoe(_Ingrediente1_ID_, _Ingrediente2_ID_)
    Ingrediente1_ID chave estrangeira -> Ingrediente(ID)
    Ingrediente2_ID chave estrangeira -> Ingrediente(ID)
```

# Modelo Lógico de Grafos



# Modelo Lógico de Grafos

```
Ingredientes-Simples(_id_, nome, grupo, subgrupo)
Composicao-Quimica(_nome_)
Macronutrientes(_nome_)
Receita(_id_, nome, regioao)
Ingrediente-Composto(_id_, nome)
Define(_idis_, _nome-macro_, quantidade-grama)
    idis chave estrangeira -> Ingredientes-Simples(id)
    nome-macro chave estrangeira -> macronutrientes(nome)
Constitui(_idis_, _nomecq_, quantidade-grama)
    idis chave estrangeira -> Ingredientes-Simples(id)
    nomecq chave estrangeira -> Composicao-Quimica(nome)
Compoe(_idis_, _idic_)
    idis chave estrangeira -> Ingredientes-Simples(id)
    idic chave estrangeira -> Ingrediente-Composto(id)
Contems(_idrec_, _idis_)
    idirec chave estrangeira -> Receita(id)
    idis chave estrangeira -> Ingredientes-Simples(id)
Contemc(_IDrec_, _IDic_)
    idirec chave estrangeira -> Receita(id)
    idic chave estrangeira -> Ingrediente-Composto(id)
```

# Operações de preparo do dataset

- Integração de dados de bancos distintos
- Extração de dados dentro de strings
- Tratamento de inconsistências
- Transformação dos dados para análise



# Integração

- Juntamos logicamente os dados no ponto dos ingredientes. Criamos um mapeamento de ingredientes do CulinaryDB para o FooDB, de forma a permitir uma análise quanto à categoria e valor nutricional dos ingredientes e receitas.
- Integração dos dados de um dataset com o outro usando o código de Hamming normalizado para no mínimo 85% de semelhança, evitando falsos positivos. Mesmo assim, sobraram cerca de 300 (~30%) ingredientes que foram associados manualmente.

```
1 def checar_par(ing_cdb, ing_fdb, sem_par: dict[str, str], pares, cdb_nome='Aliased Ingredient Name', cdb_id='Entity ID') -> bool:
2     if (s := hamming.normalized_similarity(ing_cdb[cdb_nome].lower(), ing_fdb["name"].lower())) >= 0.85:
3         sem_par.pop(ing_cdb[cdb_id], None)
4         if ing_cdb[cdb_id] in pares.keys():
5             print(f"{ing_cdb[cdb_id], ing_cdb[cdb_nome]} de {pares[ing_cdb[cdb_id]]} para {(ing_cdb[cdb_id], ing_fdb['name'])}")
6             pares[ing_cdb[cdb_id]] = (ing_cdb[cdb_nome], ing_fdb['id'], ing_fdb["name"], ing_fdb["food_group"], ing_fdb["food_subgroup"])
7             return True
8     return False
```

# Integração

- Com a ligação dos ingredientes concluída, pudemos relacionar as receitas com os novos ingredientes.
- Os ingredientes do CulinaryDB remanescentes são apenas os ingredientes compostos que não possuem par no FoodB; todos os ingredientes simples foram pareados ou removidos.

```
1 def ligar_ingredientes_receita():
2     with (open("data/interim/ingredientes_final.csv") as ing_f,
3           open("data/interim/ingredientes_compostos_final.csv") as ingc_f,
4           open("data/processed/receitas.csv") as rec_f,
5           open("data/external/04_Recipe-Ingredients_Aliases.csv") as rec_ing_f,
6           open("data/processed/ingredientes_receitas.csv", "w") as out_f,
7           open("data/interim/ing_rec_removidos.csv", "w") as removidos_f):
8         ing_reader = csv.DictReader(ing_f, lineterminator='\n')
9         ingc_reader = csv.DictReader(ingc_f, lineterminator='\n')
10        rec_reader = csv.DictReader(rec_f, lineterminator='\n')
11        rec_ing_reader = csv.DictReader(rec_ing_f, lineterminator='\n')
12        out_writer = csv.DictWriter(out_f, fieldnames=["id_ingredient", "id_receita", "volume", "massa",
13        "unidade"], lineterminator='\n')
14        removidos_writer = csv.DictWriter(removidos_f, fieldnames=["id_ing_cdb", "id_receita", "nome"],
15        lineterminator='\n')
16
17        out_writer.writeheader()
18
19        ingredientes = {x["id_cdb"]: x for x in ing_reader}
20        ingredientes_compostos = {x["id_cdb"]: x for x in ingc_reader}
21        map_final: dict[(str, str), dict] = dict()
```



# Tratamento

O que fizemos com os dados inconsistentes?

	A	B	C	D	E	F
1	id_cdb	nome_cdb	id_fdb	nome_fdb	grupo_fdb	subgrupo_fdb
2	0	Egg	633	Eggs	Eggs	Eggs
3	2	Bread	1019	White bread	Cereals and cereal products	Cereals
4	4	Wheaten Bread	836	Wheat bread	Cereals and cereal products	Leavened breads
5	6	Wholewheat Bread	836	Wheat bread	Cereals and cereal products	Leavened breads
6	7	Wort	268	Beer	Beverages	Fermented beverages
7	8	Arrack	630	Liquor	Beverages	Distilled beverages
8	10	Bantu Beer	268	Beer	Beverages	Fermented beverages
9	11	Brandy	630	Liquor	Beverages	Distilled beverages
10	12	Anise Brandy	630	Liquor	Beverages	Distilled beverages
11	13	Apple Brandy	630	Liquor	Beverages	Distilled beverages
12	14	Armagnac Brandy	630	Liquor	Beverages	Distilled beverages
13	15	Blackberry Brandy	630	Liquor	Beverages	Distilled beverages
14	16	Cherry Brandy	630	Liquor	Beverages	Distilled beverages
15	17	Cognac Brandy	630	Liquor	Beverages	Distilled beverages
16	18	Papaya Brandy	630	Liquor	Beverages	Distilled beverages

# Tratamento

Eliminação das colunas que não serão importantes na análise, para facilitar a pesquisa e melhorar a visualização dos dados.

```
1 def limpa_compound():
2     with open('data/external/Compound.csv') as csv_file:
3         csv_reader = csv.reader(csv_file, delimiter=',')
4         header = ["id", "nome"]
5         next(csv_reader, None)
6         with open('data/processed/compostos.csv', mode='w') as new_csv_file:
7             csv_writer = csv.writer(new_csv_file, delimiter=',', quotechar='"', quoting=csv.QUOTE_MINIMAL, lineterminator='\n')
8             csv_writer.writerow(header)
9             for row in csv_reader:
10                 csv_writer.writerow([row[0], row[2]])
```

# Tratamento

```
1 def limpa_nutrient():
2     with open('data/external/Nutrient.csv') as csv_file:
3         csv_reader = csv.reader(csv_file, delimiter=',')
4         header = ["id", "nome"]
5         next(csv_reader, None)
6         with open('data/processed/nutrientes.csv', mode='w') as new_csv_file:
7             csv_writer = csv.writer(new_csv_file, delimiter=',', quotechar='"', quoting=csv.QUOTE_MINIMAL, lineterminator='\n')
8             csv_writer.writerow(header)
9             for row in csv_reader:
10                 csv_writer.writerow([row[0], row[4]])
```

```
1 def limpa_receita():
2     with open('data/external/01_Recipe_Details.csv') as csv_file:
3         csv_reader = csv.reader(csv_file, delimiter=',')
4         header = ["id", "titulo", "regiao"]
5         next(csv_reader, None)
6         with open('data/processed/receitas.csv', mode='w') as new_csv_file:
7             csv_writer = csv.writer(new_csv_file, delimiter=',', quotechar='"', quoting=csv.QUOTE_MINIMAL, lineterminator='\n')
8             csv_writer.writerow(header)
9             for row in csv_reader:
10                 csv_writer.writerow([row[0], row[1], row[3]])
```

# Perguntas de Pesquisa/Análise

→ Perguntas que irão ser implementadas

- 1)Quais as estruturas bioquímicas mais frequentes em cada região?
- 2)Quais regiões possuem receitas com ingredientes majoritariamente vegetais?
- 3)Quais regiões possuem a maior média de gorduras por receita?

# 1)Quais as estruturas bioquímicas mais frequentes em cada região?

```
1 DROP VIEW IF EXISTS EstruturasFrequentes;
2 CREATE VIEW EstruturasFrequentes AS
3 SELECT
4     r.regiao AS Regiao,
5     c.nome AS NomeComposto,
6     COUNT(*) AS Quantidade
7 FROM
8     Receitas r
9 JOIN
10     Ingredientes_receitas ir ON r.id = ir.id_receita
11 JOIN
12     Compostos_content cc ON ir.id_ingrediente = cc.id_ingrediente
13 JOIN
14     Compostos c ON cc.id_composto = c.id
15 GROUP BY
16     r.regiao, c.nome
17 ORDER BY
18     COUNT(*) DESC;
19
20 SELECT regiao, nomecomposto, quantidade
21 FROM (
22     SELECT regiao, nomecomposto, quantidade, ROW_NUMBER() OVER ( PARTITION BY regiao ORDER BY quantidade
23         DESC) as rn
24     FROM EstruturasFrequentes
25 ) x
26 WHERE rn <= 3;
```



	Aa	regiao	Aa	nomecomposto	#	quantidade
		26 Uniques		3 Uniques		128 - 122,565
1		Africa		Sodium		6,252
2		Africa		Iron		6,252
3		Africa		Zinc		6,248
4		Australia & NZ		Iron		3,523
5		Australia & NZ		Sodium		3,522
6		Australia & NZ		Zinc		3,518
7		British Isles		Iron		8,013
8		British Isles		Sodium		8,007
9		British Isles		Zinc		7,974
10		Canada		Iron		7,789
11		Canada		Sodium		7,787
12		Canada		Zinc		7,756
13		Caribbean		Iron		9,755
14		Caribbean		Sodium		9,752
15		Caribbean		Zinc		9,735
16		China		Iron		8,634
17		China		Sodium		8,630
18		China		Zinc		8,611
19		DACH Countries		Iron		3,978
20		DACH Countries		Sodium		3,977
21		DACH Countries		Zinc		3,950
22		Eastern Europe		Iron		4,466
23		Eastern Europe		Sodium		4,466
24		Eastern Europe		Zinc		4,441
25		France		Iron		20,944
26		France		Sodium		20,914
27		France		Zinc		20,845
28		Greece		Sodium		7,826
29		Greece		Iron		7,826
30		Greece		Zinc		7,818
31		Indian Subcontinent		Iron		26,932
32		Indian Subcontinent		Sodium		26,912
33		Indian Subcontinent		Zinc		26,906

34	Italy	Iron	57,728
35	Italy	Sodium	57,707
36	Italy	Zinc	57,567
37	Japan	Iron	4,377
38	Japan	Sodium	4,372
39	Japan	Zinc	4,364
40	Korea	Iron	2,589
41	Korea	Sodium	2,589
42	Korea	Zinc	2,584
43	Mexico	Iron	22,119
44	Mexico	Sodium	22,118
45	Mexico	Zinc	22,088
46	Middle East	Sodium	8,685
47	Middle East	Iron	8,685
48	Middle East	Zinc	8,674
49	Misc.: Belgian	Zinc	128
50	Misc.: Belgian	Iron	128
51	Misc.: Belgian	Sodium	128
52	Misc.: Central America	Iron	137
53	Misc.: Central America	Zinc	137
54	Misc.: Central America	Sodium	137
55	Misc.: Dutch	Iron	302
56	Misc.: Dutch	Sodium	302
57	Misc.: Dutch	Zinc	300
58	Misc.: Portugal	Iron	1,135
59	Misc.: Portugal	Sodium	1,134
60	Misc.: Portugal	Zinc	1,130
61	Scandinavia	Iron	3,096
62	Scandinavia	Sodium	3,091
63	Scandinavia	Zinc	3,072
64	South America	Iron	2,311
65	South America	Sodium	2,310
66	South America	Zinc	2,301

67	South East Asia	Sodium	5,647
68	South East Asia	Iron	5,647
69	South East Asia	Zinc	5,644
70	Spain	Iron	6,903
71	Spain	Sodium	6,900
72	Spain	Zinc	6,894
73	Thailand	Iron	6,115
74	Thailand	Sodium	6,114
75	Thailand	Zinc	6,112
76	USA	Iron	122,565
77	USA	Sodium	122,461
78	USA	Zinc	122,009

## 2) Quais regiões possuem receitas com ingredientes majoritariamente vegetais?

```
Title

1 DROP VIEW IF EXISTS RegionVegetablePercentageRanking;
2 CREATE VIEW RegionVegetablePercentageRanking AS
3 SELECT
4     r.regiao AS nome_regiao,
5     COUNT(DISTINCT CASE WHEN i.grupo NOT IN ('Milk and milk products', 'Snack foods', 'Aquatic foods',
6         'Animal foods') THEN ir.id_ingrediente END) AS total_vegetais,
7     COUNT(DISTINCT ir.id_ingrediente) AS total_ingredientes,
8     COUNT(DISTINCT ir.id_receita) AS total_receitas,
9     (COUNT(DISTINCT CASE WHEN i.grupo NOT IN ('Milk and milk products', 'Snack foods', 'Aquatic foods',
10         'Animal foods') THEN ir.id_ingrediente END) * 100.0) / COUNT(DISTINCT ir.id_ingrediente) AS
11     percentagem_vegetais
12 FROM
13     Receitas r
14 JOIN
15     Ingredientes_receitas ir ON r.id = ir.id_receita
16 JOIN
17     Ingredientes i ON ir.id_ingrediente = i.id
18 GROUP BY
19     r.regiao
20 ORDER BY
21     percentagem_vegetais DESC;
22 SELECT * FROM RegionVegetablePercentageRanking LIMIT 22;
```

index	NOME_REGIAO	TOTAL_VEGETAIS	TOTAL_INGREDIENTES	TOTAL_RECEITAS	PORCENTAGEM_VEGETAIS
0	Misc.: Dutch	57	66	40	86.36363636363636
1	DACH Countries	179	212	487	84.43396226415095
2	Middle East	215	256	993	83.984375
3	Australia & NZ	202	241	494	83.81742738589212
4	Canada	242	292	1112	82.87671232876713
5	Indian Subcontinent	241	291	4054	82.81786941580756
6	Korea	134	162	301	82.71604938271605
7	Mexico	250	304	3137	82.23684210526316
8	Scandinavia	166	202	404	82.17821782178218
9	Eastern Europe	166	202	565	82.17821782178218
10	Thailand	182	222	664	81.98198198198199
11	Misc.: Central America	50	61	14	81.9672131147541
12	Misc.: Belgian	40	49	15	81.63265306122449
13	British Isles	220	270	1073	81.48148148148148
14	Africa	202	248	650	81.45161290322581
15	China	200	246	940	81.30081300813008
16	South East Asia	182	225	611	80.88888888888889
17	Greece	186	230	934	80.8695652173913
18	Misc.: Portugal	97	121	138	80.16528925619835
19	Italy	284	357	7501	79.55182072829132
20	Japan	190	239	578	79.4979079497908
21	Caribbean	221	278	1102	79.49640287769785

### 3)Quais regiões possuem a maior média de gorduras por receita?

```
Title

1 DROP VIEW IF EXISTS RegionFatAverageRanking;
2 CREATE VIEW RegionFatAverageRanking AS
3 SELECT
4     r.regiao AS nome_regiao,
5     AVG(nc.quantidade) AS media_gordura
6 FROM
7     Receitas r
8 JOIN
9     Ingredientes_receitas ir ON r.id = ir.id_receita
10 JOIN
11     Nutrientes_content nc ON ir.id_ingrediente = nc.id_ingrediente
12 WHERE
13     nc.id_nutriente = 1
14 GROUP BY
15     r.regiao
16 ORDER BY media_gordura DESC;
17 SELECT * FROM RegionFatAverageRanking LIMIT 22;
```

index	NOME_REGIAO	MEDIA_GORDURA
0	Misc.: Dutch	21326.884736842105
1	Scandinavia	19353.324574083643
2	France	17173.067878333193
3	British Isles	17108.132793709516
4	DACH Countries	17060.838876102785
5	Eastern Europe	16353.001610197904
6	Misc.: Belgian	16135.756790123458
7	Canada	15341.187846889941
8	USA	14996.273418855028
9	Italy	13821.445315528272
10	Australia & NZ	13535.283883495144
11	Indian Subcontinent	13467.79639313689
12	Middle East	12257.380754716918
13	Greece	12227.171196264626
14	South America	12011.708659217871
15	Misc.: Portugal	11823.665217391303
16	Mexico	11607.517253678232
17	Spain	11288.560922557623
18	Africa	11007.57716301899
19	Caribbean	10875.610227272737
20	Misc.: Central America	10617.594845360823
21	Japan	7651.733659730716

# Perguntas de Pesquisa/Análise

→ Perguntas que **NÃO** irão ser implementadas

- 1)Quais as combinações de ingredientes mais frequentes em cada região?
- 2)Existem similaridades entre as receitas das mais diversas regiões do globo?
- 3)Quais regiões com maior diversidade de subgrupos alimentícios?

# Perguntas de Pesquisa/Análise

→ Análises a serem exploradas por meio de bancos de grafos

Podemos construir uma rede de ingredientes fazendo a projeção dos ingredientes do FooDB que se interligam por meio das receitas do CulinaryDB em que aparecem

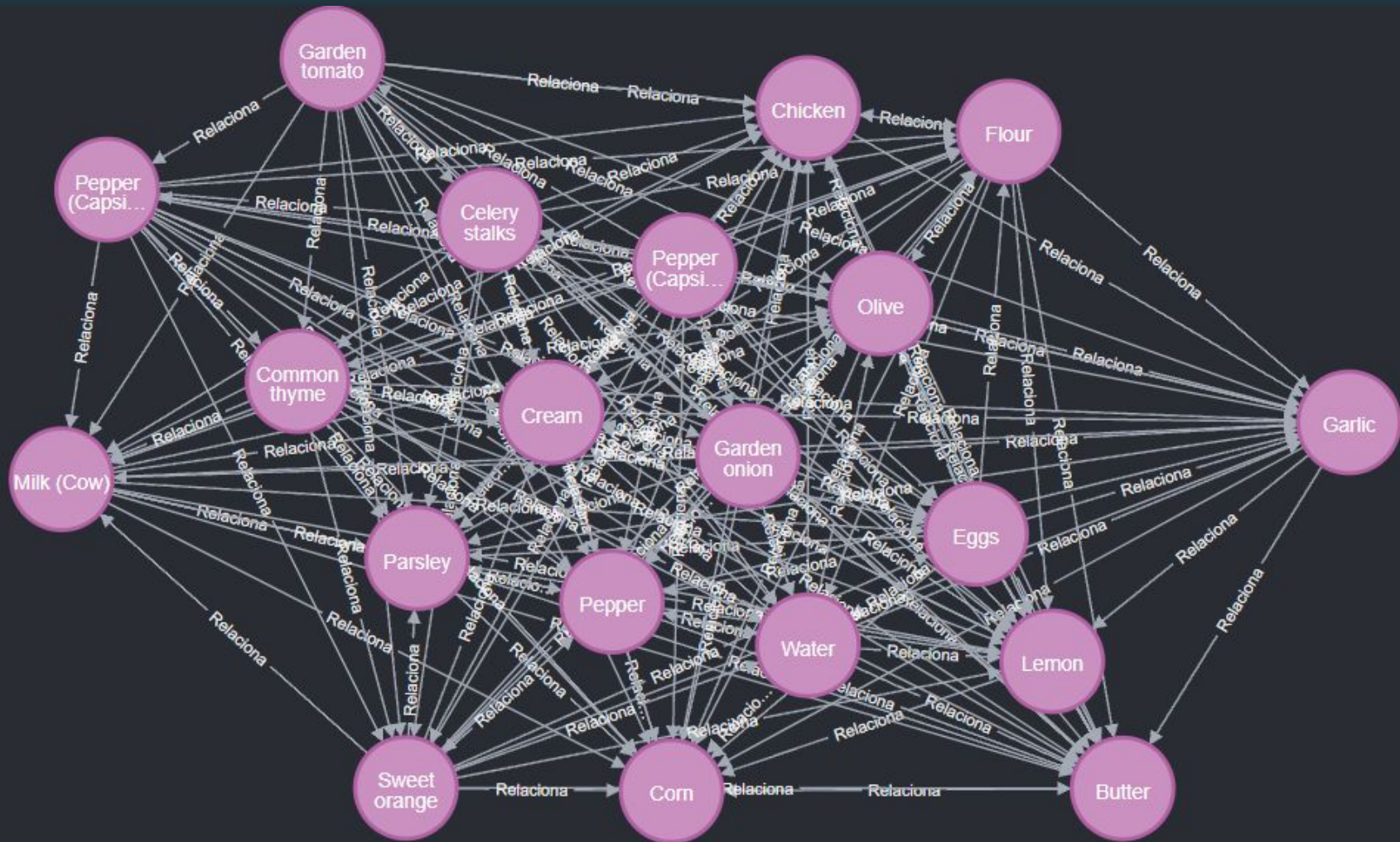
- 1) Distância grande entre dois ingredientes implica em baixa compatibilidade/disponibilidade na mesma região?
- 2) O que ingredientes considerados "hubs" têm em comum?

## Ingredientes que possuem mais ligações com outros ingredientes

```
1 MATCH (i:Ingrediente)
2 WHERE i.grupo <> 'Baking goods'
3 WITH i, [(i)-[:Relaciona]-() | 1] AS relationships
4 WITH i, REDUCE(s = 0, rel IN relationships | s + rel) AS degree
5 ORDER BY degree DESC
6 LIMIT 20
7 RETURN i, degree;
8
```

Exluímos ingredientes do grupo “Baking goods”, como óleo de cozinha e sal, pois são respostas muito óbvias





# Evolução do Projeto

- Modelos conceitual e lógico pouco alterados
- Dificuldades no pré-processamento dos dados (integrabilidade baixa)
- Tratamento de dados em python
- Falta de dados para responder perguntas (pesquisa)
- Ferramentas online não suportavam as grandes quantidades de dados (ajustes)

Fim