crunchy data

Quel est le contraire d'un hot-dog ?

Trouvons la réponse avec Postgres et l'IA

Karen Jex | Senior Solutions Architect @ Crunchy Data

PG Session 16 | Paris | Février 2024

crunchy data

crunchy data

Corn Dog

Quel est le contraire d'un ~~hot-dog~~ ?

Trouvons la réponse avec Postgres et l'IA

Karen Jex | Senior Solutions Architect @ Crunchy Data

PG Session 16 | Paris | Février 2024

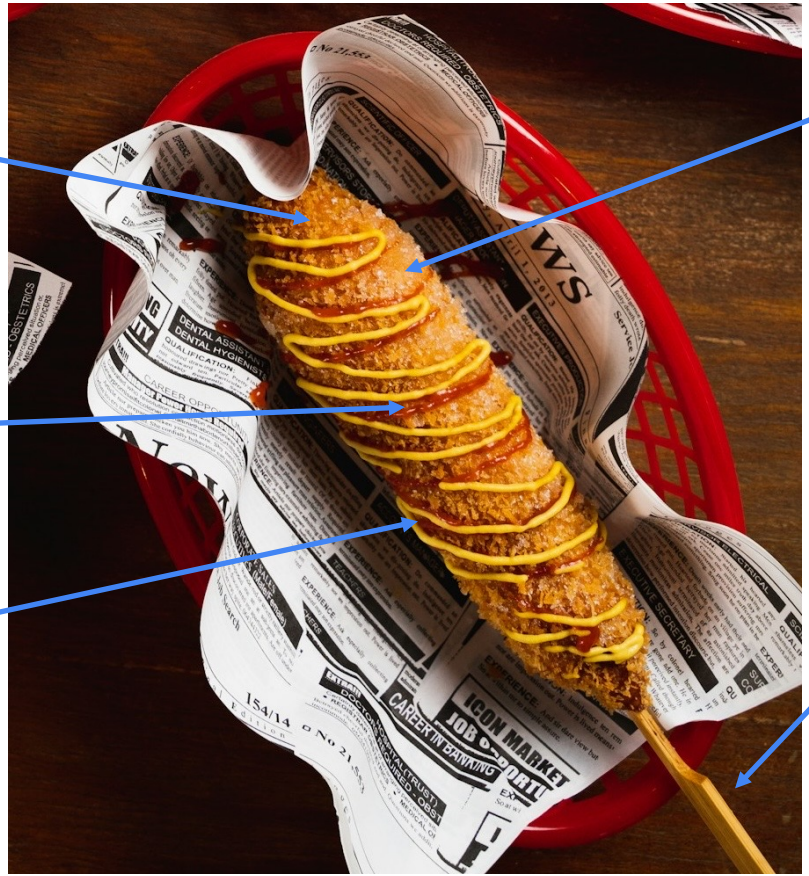
crunchy data

Qu'est-ce qu'un Corn Dog ?

Pâte de pain

ketchup

moutarde



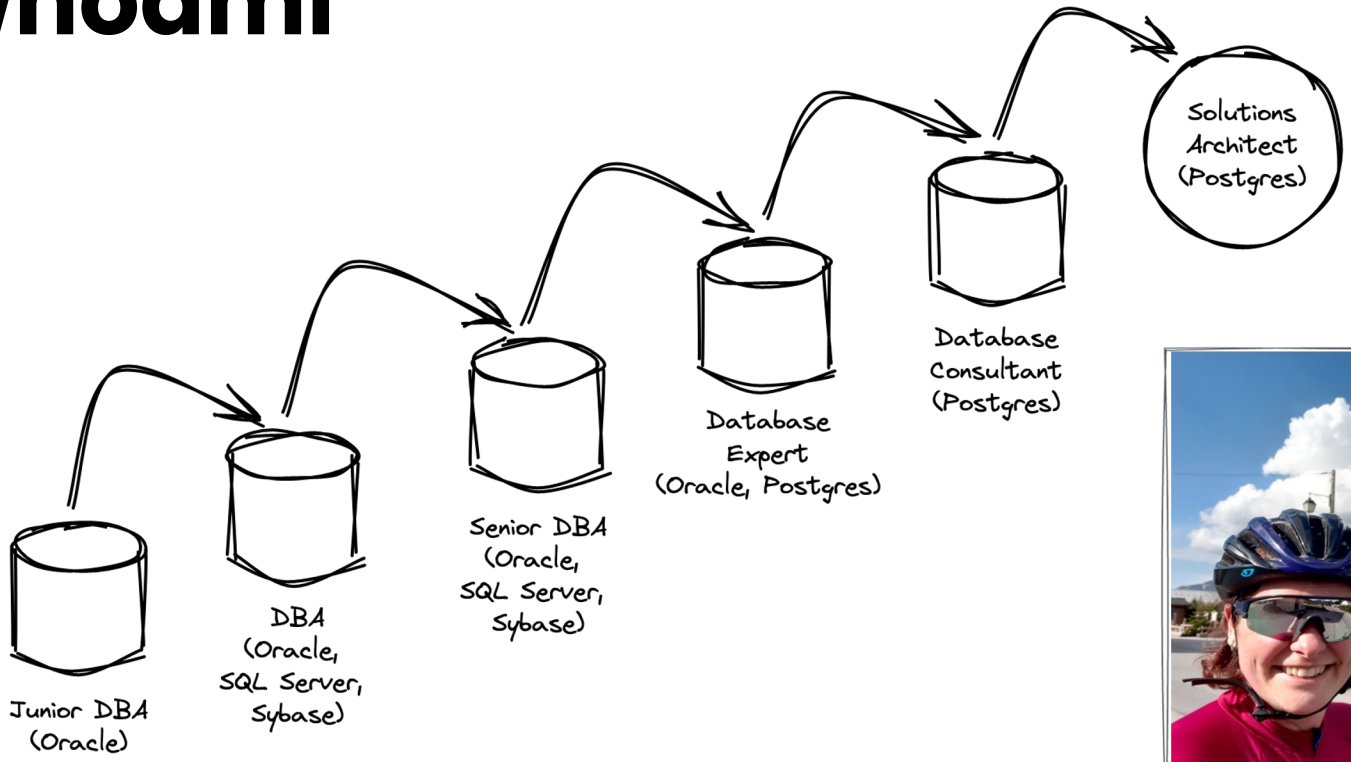
Saucisse à l'intérieur

bâtonnet

Photo by D. Pham on Unsplash

crunchy data

whoami



crunchy data

Qu’est-ce qui nous attend?

- Magie 🪄
- API OpenAI
- pgVector
- l’IA dans Postgres
- Solution sur-conçue à un problem familier
- Réponse à la question ~~sur la vie, l’Univers et le reste~~

  “Quel est le contraire d’un Corn Dog?”

pgvector

https://github.com/pgvector/pgvector

- Extension open source Postgres
- Stocker les vecteurs avec vos données
- Recherche de similarité vectorielle “voisin le plus proche”
- exacte ou approximative
- Votre choix de langage (C, C++, Go, Python, Ruby, Rust…)
- Toutes les fonctionalités Postgres

OpenAI

https://platform.openai.com/docs/introduction

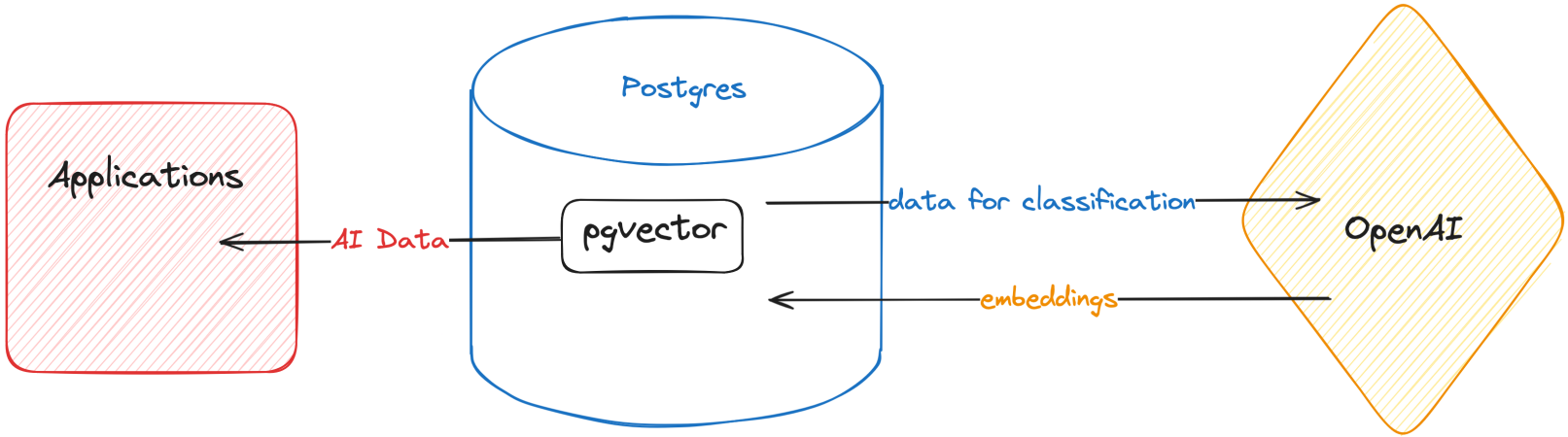
- Entreprise de produits et de services IA
- API avec plusieurs fonctionalités
- Dont embedding (classification)

```
→  ~ curl https://api.openai.com/v1/embeddings \
  -H "Content-Type: application/json" \
  -H "Authorization: Bearer $OPENAI_API_KEY" \
  -d '{
    "input": "Your text string goes here",
    "model": "text-embedding-ada-002"
  }'

{
  "object": "list",
  "data": [
    {
      "object": "embedding",
      "index": 0,
      "embedding": [
        -0.007009038,
        -0.0053659794,
        0.011887812,
        -0.024931476,
```

crunchy data

l’IA dans Postgres



crunchy data

Demo (prérequis) : pgvector

```
# exemple – v0.6.0 sur macOS

git clone --branch v0.6.0 https://github.com/pgvector/pgvector.git
cd pgvector
make
sudo make install
```

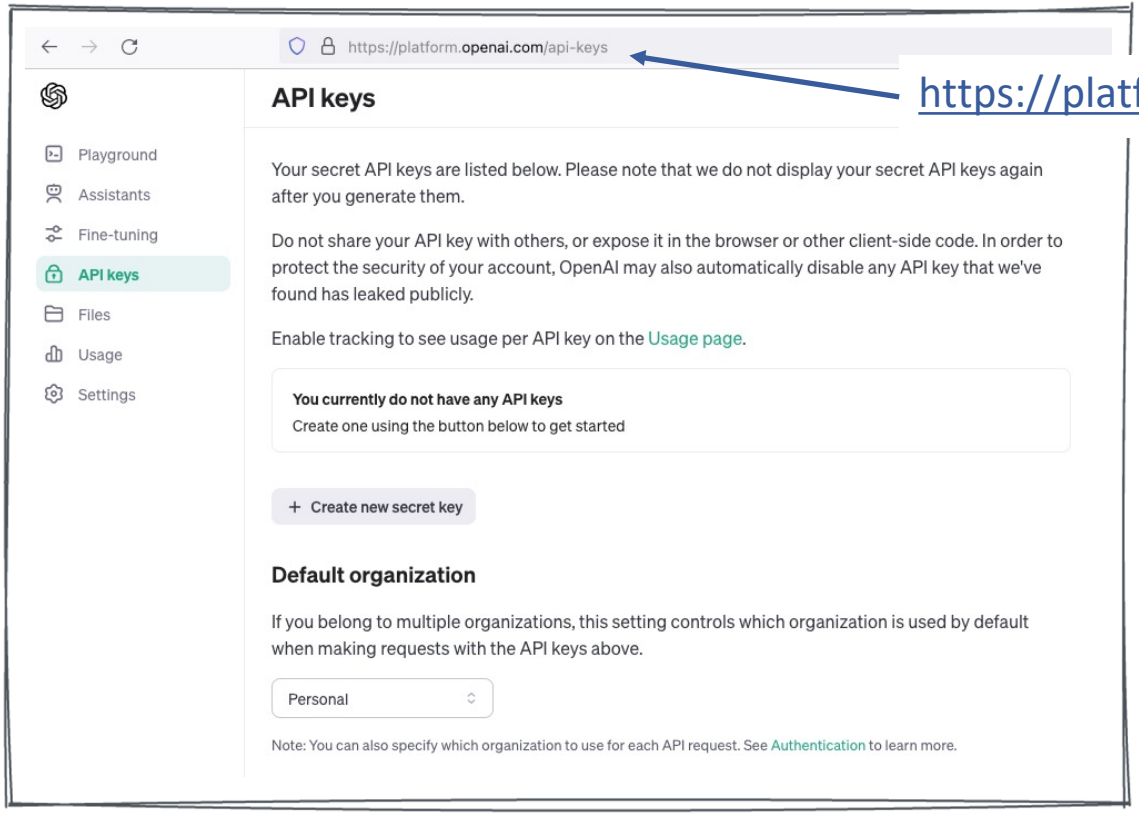
Demo (prérequis) : pgvector

```
recipes=# CREATE EXTENSION vector;

recipes=# \dx vector

                                    List of installed extensions
  Name  | Version | Schema |                     Description
--------+---------+--------+-----------------------------------------------------
 vector | 0.6.0   | public | vector data type and ivfflat and hnsw access methods
(1 row)
```

Demo (prérequis) : OpenAI



https://platform.openai.com/api-keys

crunchy data

Demo (prérequis) : Stocker les Données

```
connection_string = ENV['DATABASE_URL']
DB = Sequel.connect(connection_string)
DB.create_table? :recipes do
  primary_key :id
  String :name
  String :description
  Vector :embedding
end
recipes_xml = Nokogiri::XML(File.read('ArmedForcesRecipes.xml'))
for recipe_xml in recipes_xml.xpath('/*/recipe')
  recipe_id = DB[:recipes].insert(
    name: recipe_xml["description"],
    description: recipe_xml.xpath(".//XML_MEMO1")[0]&.text )
end
```

crunchy data

Demo (prérequis) : Stocker les Données

```
SELECT COUNT(*) FROM recipes;

 count
-------
   720
```

Demo (prérequis) : Stocker les Données

```
Table "public.recipes"
   Column    |  Type   | Collation | Nullable |              Default
-------------+---------+-----------+----------+-----------------------------------
 id          | integer |           | not null | generated by default as identity
 name        | text    |           |          |
 description | text    |           |          |
 embedding   | vector  |           |          |
Indexes:
    "recipes_pkey" PRIMARY KEY, btree (id)
```

Les Données

SANDWICHES No.N 034 00

CORN DOG

Yield 100 **Portion** 1 Sandwich

Calories	Carbohydrates	Protein	Fat	Cholesterol	Sodium	Calcium
258 cal	20 g	8 g	16 g	35 mg	674 mg	49 mg

Ingredient	Weight	Measure	Issue
FRANKFURTERS	10 lbs		
FLOUR,WHEAT,GENERAL PURPOSE	3-1/3 lbs	3 qts	
CORN MEAL	1-2/3 lbs	1 qts 1-1/2 cup	
BAKING POWDER	1-1/3 oz	2-2/3 tbsp	
SALT	1 oz	1 tbsp	
SUGAR,GRANULATED	3-1/2 oz	1/2 cup	
MUSTARD,DRY	3-1/8 oz	1/2 cup	
MILK,NONFAT,DRY	3-5/8 oz	1-1/2 cup	
WATER	3-7/8 lbs	1 qts 3-1/2 cup	
EGGS,WHOLE,FROZEN	9-5/8 oz	1-1/8 cup	
OIL,SALAD	5-3/4 oz	3/4 cup	

Method

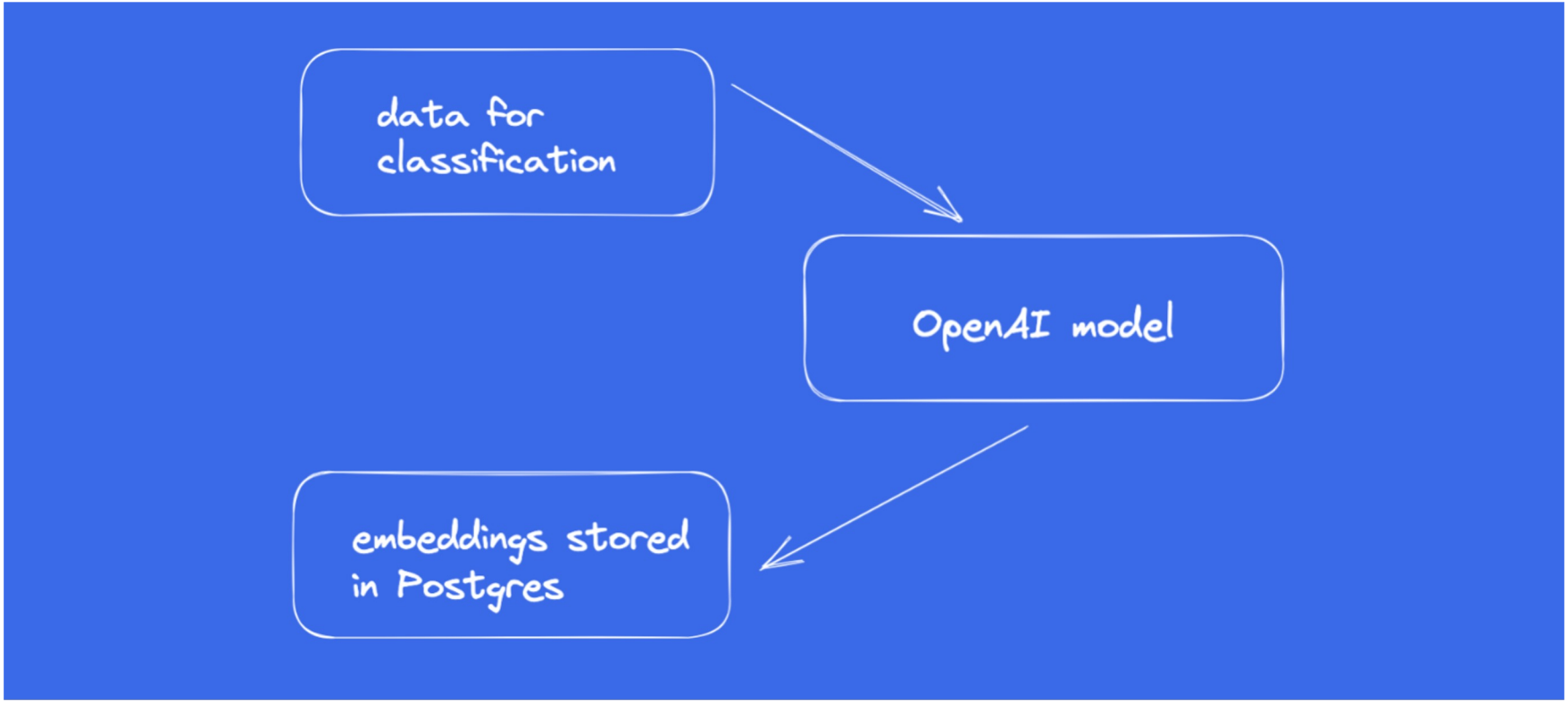
1 Insert 1 stirring stick lengthwise into each thawed frankfurter. Dry surface of frankfurter with paper towel.
2 Combine flour, cornmeal, baking powder, salt, sugar, mustard flour and milk.
3 Add water, eggs and salad oil or melted shortening to dry ingredients. Blend well.
4 Dip frankfurters in cornmeal mixture; allow excess batter to drain slightly; fry 2 to 4 minutes or until golden brown in 375 F. deep fat. CCP: Internal temperature must reach 145 F. or higher for 15 seconds.
5 Drain on absorbent paper.
6 Serve hot. CCP: Hold for service at 140 F. or higher.

Notes

1 18-3/4 lbs frozen corn dogs may also be used.

crunchydata

Classification des Données



crunchy data

Classification des Données

```
openai = OpenAI::Client.new(access_token: ENV['OPENAI_API_KEY'])
while recipe = DB[:recipes].where(embedding: nil).exclude(description:
nil).first do
  submitted_value = recipe[:description].gsub(/\n/, ' ')
    response = openai.embeddings(
      parameters: {
        model: 'text-embedding-ada-002',
        input: submitted_value })
begin
  embedding_value = response["data"][0]["embedding"].to_s
  DB[:recipes].where(id: recipe[:id]).update(embedding: embedding_value)
rescue
  puts [$!, response].inspect
end
```

crunchy data

Qu’est-ce qu’un Embedding ?

Vecteurs

Coordonnées 2D	[x,y]	2 dimensions
Coordonnées 3D	[x,y,z]	3 dimensions
Vecteur pgvector	[x,y,z,…,a]	jusqu’à 16k dimensions

crunchy data

Qu’est-ce qu’un Embedding ?

```
[
   0.0030384033,  -0.028696792, 0.00008155421, -0.018161833,  -0.028052075,
    0.026584314,  -0.019163202, -0.0053600725, -0.022674857,   0.011001351,
    0.031796925,   0.007366242,  0.025198856,  -0.00827159,    0.002122767,
    0.006406024,   0.003249308, -0.0039849035,  0.014595309,  -0.007949231,
   -0.028861402,  -0.014595309,  0.006447176,  0.0017069585,   0.012606287,
  -0.0011625494,   0.005963638, -0.009698199,  0.009245525,  -0.017613137,
    0.025349747,   0.020672116,  0.012139896, -0.009485579,  -0.025733836,
   -0.03039775,    0.011351145, 0.0075514265,  0.02312753,   -0.006711236,
   0.0034619279,  -0.00458504,  0.0017575413, -0.031330533, -0.0042318166,
 ... 1436 more items
]
```

Proximité

Trouver des Recettes Similaires

Image par Matteo Orlandi from Pixabay

crunchydata

Trouver des Recettes Similaires

```
SELECT
 recipe_1.id AS pizza_id,
 recipe_1.name AS pizza_name,
 recipe_2.id AS similar_recipe_id,
 recipe_2.name AS similar_recipe_name
FROM
 (SELECT * FROM recipes WHERE name = 'Pizza')
recipe_1,
 recipes AS recipe_2
ORDER BY recipe_1.embedding <=> recipe_2.embedding
LIMIT 4;
```

crunchy data

Résultats

```
 pizza_id | pizza_name | recipe_id |    recipe_name
----------+------------+-----------+-------------------
      431 | Pizza      |       431 | Pizza
      431 | Pizza      |       433 | Pizza, 12 in, fzn
      431 | Pizza      |       126 | Chicken, parmesan
      431 | Pizza      |       435 | Pizza, treats
(4 rows)
```

Trouver des Recettes Similaires

```
SELECT
 recipe_2.id AS similar_recipe_id,
 recipe_2.name AS similar_recipe_name
FROM
 (SELECT * FROM recipes WHERE name = 'Pizza')
recipe_1,
 recipes AS recipe_2
WHERE LOWER(recipe_2.name) NOT LIKE '%pizza%'
ORDER BY recipe_1.embedding <=> recipe_2.embedding
LIMIT 4;
```

crunchy data

Résultats

```
 recipe_id |             recipe_name
-----------+-------------------------------------
       126 | Chicken, parmesan
        31 | Beef, ground, hamburger, w/parmesan
       211 | Dish, eggplant, parmesan
       229 | Dish, lasagna
(4 rows)
```

Opérateurs Vectorielles

<=>	distance cosinus (cosine)
<->	distance Euclidienne (Euclidean)
<#>	produit scalaire (inner product)

Fonctions Vectorielles

l2_distance(v1, v2)	distance Euclidienne
cosine_distance(v1, v2)	distance cosinus
inner_product(v1, v2)	produit scalaire
l1_distance(v1, v2)	distance de Manhattan

Opérateurs & Fonctions Vectorielles

addition	+
soustraction	-
multiplication	*
moyenne	avg(vector)
somme	sum(vector)

Quel est le contraire d’un corn dog ?

SANDWICHES No.N 034 00

CORN DOG

Yield 100 **Portion** 1 Sandwich

Calories	Carbohydrates	Protein	Fat	Cholesterol	Sodium	Calcium
258 cal	20 g	8 g	16 g	35 mg	674 mg	49 mg

Ingredient	Weight	Measure	Issue
FRANKFURTERS	10 lbs		
FLOUR,WHEAT,GENERAL PURPOSE	3-1/3 lbs	3 qts	
CORN MEAL	1-2/3 lbs	1 qts 1-1/2 cup	
BAKING POWDER	1-1/3 oz	2-2/3 tbsp	
SALT	1 oz	1 tbsp	
SUGAR,GRANULATED	3-1/2 oz	1/2 cup	
MUSTARD,DRY	3-1/8 oz	1/2 cup	
MILK,NONFAT,DRY	3-5/8 oz	1-1/2 cup	
WATER	3-7/8 lbs	1 qts 3-1/2 cup	
EGGS,WHOLE,FROZEN	9-5/8 oz	1-1/8 cup	
OIL,SALAD	5-3/4 oz	3/4 cup	

Method

1 Insert 1 stirring stick lengthwise into each thawed frankfurter. Dry surface of frankfurter with paper towel.
2 Combine flour, cornmeal, baking powder, salt, sugar, mustard flour and milk.
3 Add water, eggs and salad oil or melted shortening to dry ingredients. Blend well.
4 Dip frankfurters in cornmeal mixture; allow excess batter to drain slightly; fry 2 to 4 minutes or until golden brown in 375 F. deep fat. CCP: Internal temperature must reach 145 F. or higher for 15 seconds.
5 Drain on absorbent paper.
6 Serve hot. CCP: Hold for service at 140 F. or higher.

Notes

1 18-3/4 lbs frozen corn dogs may also be used.

Trouver des Recettes pas Similaires

```
SELECT
 recipe_2.id AS recipe_id,
 recipe_2.name AS recipe_name
FROM
 (SELECT * FROM recipes
  WHERE name = 'Corn Dog' LIMIT 1) recipe_1,
 recipes AS recipe_2
WHERE recipe_2.description IS NOT NULL
ORDER BY recipe_1.embedding <=> recipe_2.embedding
DESC
LIMIT 1;
```

crunchy data

Résultats

```
 id  |    name    | id  |          name
-----+------------+-----+------------------------
 165 | Corn Dog | 538 | Salad, green, tossed
(1 row)
```

Résultats

Photo by D. Pham on Unsplash

Image par Pexels de Pixabay

crunchy data

Moteurs de Recommandation

- eCommerce “pensez à acheter également…”
- Media/Entertainment “regarder cette nouvelle serie”
- health & fitness “essayez cette recette plus saine”
- social media “vous allez adorer ce post”
- travel “voici l’hôtel de vos rêves”

crunchy data

Considérations

- Stockage
- Performance
- Ressources

Taille d’un Vecteur

4 * dimensions + 8 octets

```
SELECT vector_dims(embedding)
FROM recipes
WHERE name = 'Pizza';

 vector_dims
-------------
        1536
```

Embedding OpenAI : 4 * 1536 + 8 = 24584 octets = 24 Ko

crunchy data

Stockage

720 embeddings:

720 * 24Kb = 17Mo

1 million embeddings:

1 000 000 * 24Kb = 23Gb

Performance

```
                     QUERY PLAN
-------------------------------------------------------------------------------------
 Limit  (cost=296.79..296.80 rows=1 width=60)
   ->  Sort  (cost=296.79..298.59 rows=719 width=60)
         Sort Key: ((recipes.embedding <=> recipe_2.embedding)) DESC
         ->  Nested Loop  (cost=0.00..293.20 rows=719 width=60)
               ->  Limit  (cost=0.00..143.00 rows=1 width=76)
                     ->  Seq Scan on recipes  (cost=0.00..143.00 rows=1 width=76)
                           Filter: (name = 'Corn Dog'::text)
               ->  Seq Scan on recipes recipe_2  (cost=0.00..141.20 rows=719 width=44)
                     Filter: (description IS NOT NULL)
(9 rows)
```

Les Index

- Recherche de similarité “voisin le plus proche”
- “Nearest neighbor search”
- Exacte : precise, moins rapide
- Approximative : moins precise, plus rapide
- Types d’Index : IVFFlat ou HNSW

Types d’Index

IVFFlat

- Création plus rapide
- Utilise moins de mémoire
- Performance moins bien
- Créer sur table déjà alimentée

Types d’Index

IVFFlat

- Création plus rapide
- Utilise moins de mémoire
- Performance moins bien
- Créer sur table déjà alimentée

HNSW

- Création moins rapide
- Utilise plus de mémoire
- Meilleure performance
- Créer sur table vide

- Créer un index par opérateur distance

crunchy data

Performance – les Index

```
SELECT name FROM recipes
ORDER BY
  embedding <-> (
  SELECT embedding
  FROM recipes
  WHERE id = 151 ) -- chocolate chip cookies!
  LIMIT 5;

           name
-------------------------
 Cookies, chocolate chip
 Cookies, crisp chocolate
 Cookies, chocolate drop
 Bar, toffee, crisp
 Cookies, peanut butter
(5 rows)
```

crunchy data

Performance - les Index

```
                        QUERY PLAN
-----------------------------------------------------------------------------
 Limit  (cost=142.25..142.26 rows=5 width=30)
   InitPlan 1 (returns $0)
     ->  Index Scan using recipes_pkey on recipes
recipes_1  (cost=0.28..8.29 rows=1 width=32)
           Index Cond: (id = 151)
   ->  Sort  (cost=133.96..135.76 rows=720 width=30)
         Sort Key: ((recipes.embedding <-> $0))
         ->  Seq Scan on recipes  (cost=0.00..122.00 rows=720 width=30)
(7 rows)
```

crunchy data

Performance – les Index

```
CREATE INDEX ON recipes
USING hnsw (embedding vector_l2_ops)
WITH (m = 4, ef_construction = 10);
```

Performance - les Index

```
CREATE INDEX ON recipes
USING hnsw (embedding vector_l2_ops)
WITH (m = 4, ef_construction = 10);

ERROR:  column does not have dimensions

ALTER TABLE recipes ALTER COLUMN embedding
SET DATA TYPE vector(1536);
```

Stockage

Les index HNSW sont grand :

8+ Go pour 1M lignes d’embeddings IA

de préférence en mémoire

Plus d’infos :

https://www.crunchydata.com/blog/hnsw-indexes-with-postgres-and-pgvector

Performance - paramètres

```
SET hnsw.ef_search = 5;     -- default 40
```

Performance - les Index

```
                                  QUERY PLAN
------------------------------------------------------------------------------------------
 Limit  (cost=96.47..99.67 rows=5 width=30)
   InitPlan 1 (returns $0)
     ->  Index Scan using recipes_pkey on recipes recipes_1  (cost=0.28..8.29 rows=1
width=32)
           Index Cond: (id = 151)
   ->  Index Scan using recipes_embedding_idx on recipes  (cost=88.18..549.18 rows=720
width=30)
         Order By: (embedding <-> $0)
(6 rows)
```

crunchy data

Performance - les Index

```
SELECT name FROM recipes
ORDER BY
  embedding <-> (
  SELECT embedding
  FROM recipes
  WHERE id = 151 ) -- chocolate chip cookies!
  LIMIT 5;

           name
-------------------------
 Cookies, chocolate chip
 Cookies, crisp chocolate
 Cookies, chocolate drop
 Bar, toffee, crisp
 Cookies, peanut butter
(5 rows)
```

crunchy data

Performance – Plus d’Infos

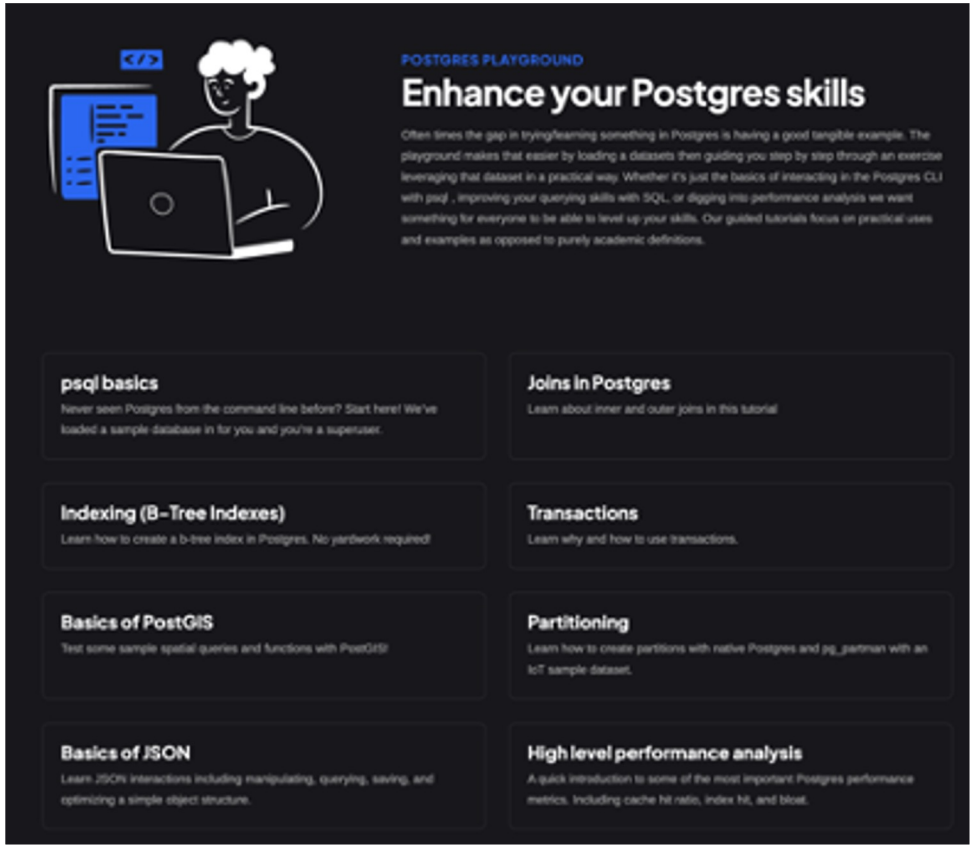
- Index : jusqu’à 2000 dimensions
- Réduction de la Dimensionnalité
- Séparation physique des données
- Caching

Un Grand Merci à mes collegues !

- **Chris Winslett** : blog “What’s Postgres Got To Do With AI?”
- Code: https://github.com/Winslett/rails-postgres-ai-workshop
- Crunchy Data Blog : “Postgres AI Series”
  https://www.crunchydata.com/blog/topic/ai
- **Bob Pacheco** : lightning talk @ OpenShift Commons
  https://www.youtube.com/watch?v=1ddxwZWSgtY

crunchy data

Postgres Playground



crunchy data

crunchy data

Merci !

Karen Jex | @karenhjex | karen.jex@crunchydata.com

crunchy data