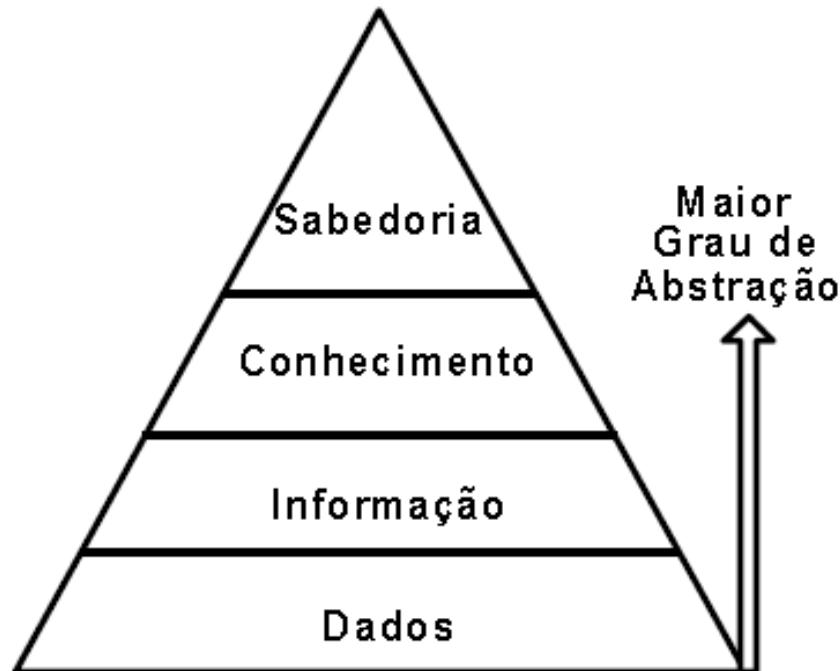


Banco de Dados 2

13 – Descoberta de Conhecimento em Bancos de Dados (KDD)

Dados, Informação e Conhecimento



- Os **DADOS** são os registros soltos, aleatórios, sem quaisquer análises.
- **Dados** são códigos que constituem a **matéria prima da informação**, ou seja, é a **informação não tratada** que ainda não apresenta relevância.
- Eles representam um ou

Informação

- A **INFORMAÇÃO** seria **qualquer estruturação ou organização desses dados**.
- Ela é **um registro**, em suporte físico ou intangível, **disponível à assimilação crítica para produção de conhecimento**.
- **Informação** é, portanto, o **material de que é feito o conhecimento**, após posicionamento crítico do indivíduo.
- Além disso, a **informação** é **derivada dos dados** que, sem um sentido ou contexto, significam muito pouco.

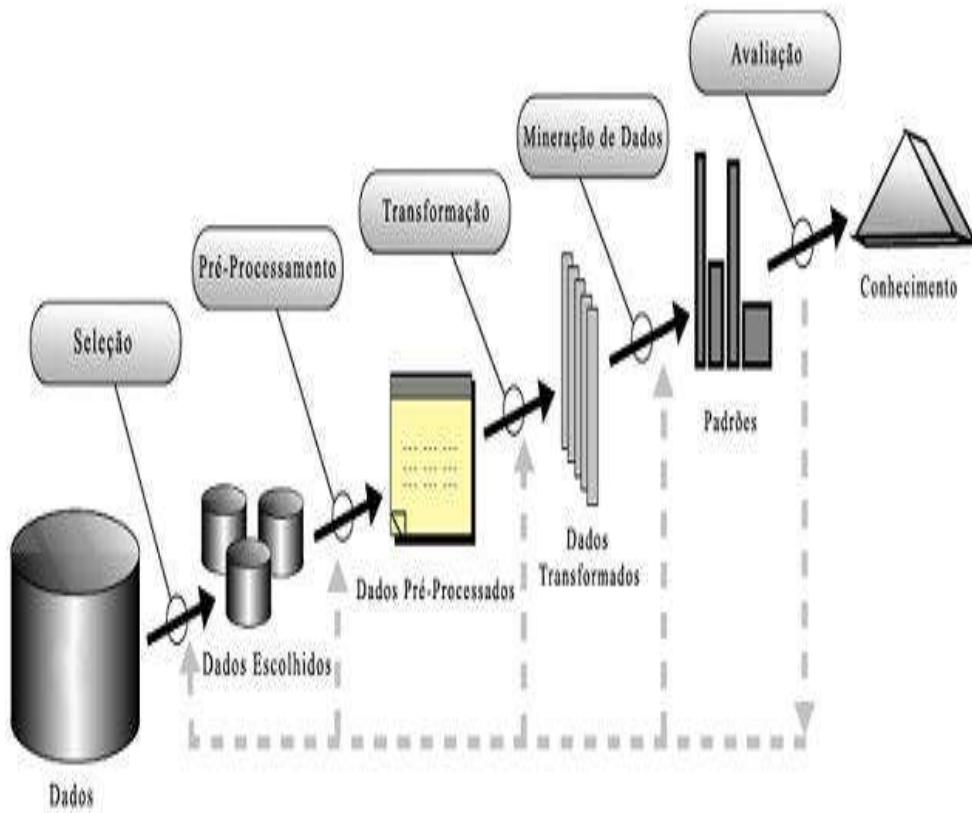
Conhecimento

- **Informação não é Conhecimento**, informação é diferente de Conhecimento.
- O **Conhecimento** é a **informação processada e transformada** em **experiência** pelo indivíduo.
- O **conhecimento** é a capacidade que, o processamento da informação adicionado ao repertório individual, nos dá, de agir e prever o resultado dessa ação.

Conhecimento

- **Aprendizagem** seria, então, toda exposição a novas informações que, a partir dai, modificam o nosso comportamento e relacionamento com o meio-ambiente que nos rodeie.
- Se **informação** é **dado trabalhado**, então **conhecimento** é **informação trabalhada**.
- A **sabedoria** é a boa aplicação do **conhecimento**.

Mineração de Dados e Reconhecimento de Padrões



KDD (*Knowledge Discovery in Databases*) – em português, **Descoberta de Conhecimento em Bancos de Dados** –, que permite a extração não trivial de conhecimento

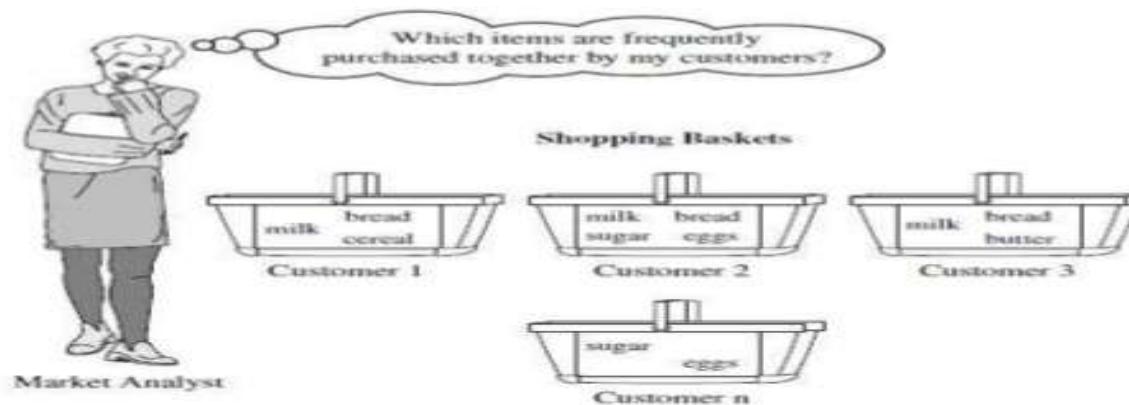
KDD

- A expressão ***data mining*** surgiu pela primeira vez em **1990** em comunidades de bases de dado.
- A **mineração de dados (*data mining*)** é a etapa de análise do processo conhecido como **KDD (Knowledge Discovery in Databases)**.

KDD e Reconhecimento de Padrões

INSTITUTO DE INFORMÁTICA - UFG

Motivação



KDD e Reconhecimento de Padrões

- O processo de Data Mining localiza **padrões** através da judiciosa **aplicação** de **processos** de **generalização**, algo que é conhecido como *indução*.
- **Padrões** são unidades de informação que se repetem, ou então são sequências de informações que dispõe de uma **estrutura** que se repete.

KDD e Reconhecimento de Padrões

Sequência original: ABCXYABCZKABDKCABCTUABEWLABCWO

- Observe atentamente essa **sequência de letras** e tente encontrar alguma coisa relevante.
- **Passo 1:** A primeira etapa é perceber que existe uma sequência de letras que se repete bastante.
- Encontramos as **sequências "AB"** e **"ABC"** e observamos que elas ocorrem com frequência superior

KDD e Reconhecimento de Padrões

Passo 2: Após determinarmos as sequências "ABC" e "AB", verificamos que elas *segmentam* o padrão original em diversas unidades independentes:

"ABCXY"
"ABCZK"
"ABDKC"
"ABCTU"
"ABEWL"
"ABCWO"

Sequência original: ABCXYZKABDKCABCTUABEWLABCWO

Passo 3: Fazem-se agora induções, que geram algumas *representações genéricas* dessas unidades:

"ABC??" "ABD??" "ABE??" e "AB???",
onde "?" representa qualquer letra

No final desse processo, toda a sequência original foi substituída por regras genéricas indutivas⁵ que simplificou (reduziu) a informação original a algumas expressões simples. Se você compreendeu esta explicação até aqui, então você acaba de conhecer um dos pontos essenciais do Data Mining: como se pode fazer para extrair certos padrões de dados brutos. Contudo, mais importante do que simplesmente obter essa redução (compressão) de informação, esse processo nos permite gerar formas de *predizer* futuras ocorrências de padrões. Este é exatamente o ponto onde este processo começa a mostrar o seu valor.

KDD e Reconhecimento de Padrões

- Uma das **expressões** encontradas nos diz que toda vez que encontramos a **sequência "AB"**, podemos **inferir** que iremos **encontrar mais três caracteres** e isto completaria um "**padrão**".
 - Nesta forma abstrata ainda pode ficar difícil de perceber a relevância deste resultado.
 - Por isso vamos usar uma representação mais próxima da realidade.
- Considere que a **letra 'A'** esteja **representando um item qualquer de um registro comercial**.
 - Por exemplo, a **letra 'A'** poderia significar "**aquisição de pão**" em uma transação de supermercado.
 - A **letra 'B'** poderia, por exemplo, significar "**aquisição de leite**".

KDD e Reconhecimento de Padrões

- A letra '**C**' é um indicador de que o **leite** que foi adquirido é do tipo **desnatado**.
- É interessante notar que a obtenção de **uma regra** com as **lettras "AB"** quer dizer, na prática, que toda vez que alguém comprou **pão**, também comprou **leite**.
- **Esses dois atributos** estão **associados** e isto foi revelado pelo processo de **descoberta de padrões**.

KDD e Reconhecimento de Padrões

- Esta associação já nos fará pensar em colocar "**leite**" e "**pão**" **mais próximos** um do outro no **supermercado**, pois assim estaríamos facilitando a **aquisição conjunta desses dois produtos**.
- Mas a coisa pode ir além disso, bastando continuar nossa exploração da **indução**. É o que faremos a seguir.

KDD e Reconhecimento de Padrões

- Suponha que a letra **X** queira dizer "**manteiga sem sal**", e a letra '**Z**' signifique "**manteiga com sal**".
- A letra '**T**' poderia significar "**margarina**".
- Parece que poderíamos tentar unificar todas essas letras através de um único conceito, uma idéia que resuma uma característica essencial de todos esses itens.
- Introduzimos a letra '**V**', que significaria "**manteiga/margarina**", ou "**coisas que passamos no pão**".
- Fizemos uma **indução orientada a atributos**, substituímos uma série de valores distintos (mas similares) por *um nome só*.

KDD e Reconhecimento de Padrões

- Observe que ao fazer isso **estamos perdendo um pouco das características dos dados originais.**
- Após essa transformação, já não sabemos mais o que é manteiga e o que é margarina.
- Essa **perda de informação** é fundamental na **indução** e é um dos fatores que permite o aparecimento de **padrões mais gerais**.

KDD e Reconhecimento de Padrões

Qual a vantagem de assim proceder? Basta codificar nossa sequência original substituindo a letra V em todos os lugares devidos. Assim fica essa sequência transformada:

ABCYABCVKABDKCABCVUABEWLABCVO

Daqui, nosso sistema de Data Mining irá extrair, entre outras coisas, a expressão "ABCV", que nos irá revelar de pronto algo muito interessante:

A maioria dos usuários que adquiriram pão e leite desnatado *também adquiriram manteiga ou margarina.*

De posse desta regra, fica fácil imaginar uma disposição nas prateleiras do supermercado para incentivar ainda mais este hábito⁷. Em linguagem mais lógica, pode-se dizer que pão e leite estão associados (implicam) na aquisição de manteiga:

Pão, Leite \Rightarrow Manteiga

O lado da esquerda desta expressão (Pão, Leite) é chamado de *Antecedente*, e o lado da direita de *Consequente*.

KDD e Reconhecimento de Padrões

- As letras ‘A’, ‘B’, ‘C’, ‘X’ e ‘Y’ são, cada uma isoladamente, dados.
- O conjunto ‘ABCXY’ representando itens comprados por um consumidor em um dado momento, com ‘A’ significando ‘Pão’, ‘B’ significando ‘Leite’ etc., corresponde a uma informação.

KDD e Reconhecimento de Padrões

- Essas informações estão armazenadas em um **Sistema Gerenciador de Banco de Dados (SGBD)**, como o **PostgreSQL** ou o **MySQL**.
- **Algoritmos de Aprendizagem de Máquina (*Machine Learning*)**, presentes em um KDD, ao identificarem **padrões** nas informações, conseguem **extrair** conhecimento desses **Bancos de Dados**.

KDD

- **Descoberta de Conhecimento em Bancos de Dados (KDD – Knowledge Discovery in Databases)** é o **processo** de identificar **padrões válidos, novos, potencialmente úteis e comprehensíveis** em grandes **volumes** de dados **armazenados** em **bancos de dados**.
- Em outras palavras, o **KDD** não é apenas “**analisar dados**”, mas **transformar grandes quantidades de dados brutos em conhecimento útil para apoiar decisões**.

KDD

- **Etapas principais do KDD:** Embora diferentes autores variem um pouco, em geral o processo é composto por:
 - **Seleção** – escolha dos dados relevantes no banco.
 - **Pré-processamento** – tratamento de dados ausentes, ruidosos ou inconsistentes.
 - **Transformação** – organização e padronização dos dados para análise (normalização, redução de dimensionalidade, etc.).
 - **Mineração de Dados (Data Mining)** – aplicação de algoritmos para encontrar padrões, regras de associação, classificações, agrupamentos, etc.
 - **Interpretação e Avaliação** – validação dos padrões descobertos, analisando sua relevância e utilidade.
 - **Apresentação do Conhecimento** – comunicação dos resultados de forma comprehensível (relatórios, dashboards, gráficos).

Exemplo de KDD

- **Cenário:**
- Imagine um **supermercado** com **registros de compras (transações)**.
- Cada **transação** é a **lista de itens comprados por um cliente** em um dado momento.

Exemplo de KDD

Transação	Itens Comprados
T1	Pão, Manteiga, Leite
T2	Pão, Manteiga
T3	Pão, Leite
T4	Pão, Manteiga, Café
T5	Manteiga, Leite

Exemplo de KDD

- Na etapa de mineração de dados aplicaremos o **algoritmo de aprendizagem de máquina (Machine Learning) Apriori.**
- O **Apriori** funciona em **três passos principais:**

Exemplo de KDD

- **1. Gerar conjuntos frequentes de itens:**
- Calcula-se o **suporte (frequência relativa)** de cada **item ou conjunto de itens**.
- **Exemplo** (considerando **5 transações**):
- **Suporte(Pão) = 4/5 = 80%**
- **Suporte(Manteiga) = 4/5 = 80%**
- **Suporte(Leite) = 3/5 = 60%**
- **Suporte(Pão, Manteiga) = 3/5 = 60%**

Exemplo de KDD

- O **pão**, por exemplo, ocorre em 4 das 5 transações analisadas.
- O **leite**, em 3 das 5 transações.
- Assim, o pão ocorre em 80% das transações ($4/5$) e o leite em 60% das transações ($3/5$).
- Se definirmos um **limiar mínimo de suporte = 50%**, os **itens** acima passam no corte.

Exemplo de KDD

- **2. Gerar regras de associação:**
- Com base nos conjuntos frequentes, o **Apriori cria regras** do tipo: $A \rightarrow B$.
- (“**Se** o cliente compra A, **então** tende a comprar B”).

Exemplo de KDD

- A regra **Pão → Manteiga** apresenta **Suporte(Pão, Manteiga) = 3/5 = 60%**.
- A regra **Pão → Leite** apresenta **Suporte(pão, leite) = 2/5 = 40%**.
- A regra **Pão → Café** apresenta **Suporte(pão, café) = 1/5 = 20%**.
- A regra **Manteiga → Pão** apresenta **Suporte(manteiga, pão) = 3/5 = 60%**.

Exemplo de KDD

- A regra **Manteiga → Leite** apresenta **Suporte(manteiga, leite) = 2/5 = 40%**.
- A regra **Manteiga → Café** apresenta **Suporte(manteiga, café) = 1/5 = 20%**.
- A regra **Leite → Pão** apresenta **Suporte(leite, pão) = 2/5 = 40%**.
- A regra **Leite → Manteiga** apresenta **Suporte(leite, manteiga) = 2/5 = 40%**.

Exemplo de KDD

- A regra **Leite → Café** apresenta **Suporte(leite, café) = 0/5 = 0%**.
- A regra **Café → Leite**, do mesmo jeito, apresenta **Suporte(café, leite) = 0/5 = 0%**.
- A regra **Café → Pão**, da mesma forma, apresenta **Suporte(café, pão) = 1/5 = 20%**.
- A regra **Café → Manteiga** também apresenta **Suporte(café, manteiga) = 1/5 = 20%**.

Exemplo de KDD

- **3. Avaliar as regras:**
- As métricas mais usadas são:
- Suporte: frequência de ocorrências do conjunto.
- **Suporte(Pão ⇒ Manteiga) = Suporte(Pão, Manteiga) = 60%.**
- Confiança: probabilidade de comprar B dado que comprou A.
- **Confiança(Pão ⇒ Manteiga) = Suporte(Pão, Manteiga) / Suporte(Pão) = 60% / 80% = 75%**

Exemplo de KDD

- Lift: mede o quanto A e B ocorrem juntos acima do acaso.
- $\text{Lift}(\text{Pão} \Rightarrow \text{Manteiga}) = \text{Confiança} / \text{Suporte}(\text{Manteiga}) = 75\% / 80\% = 0,9375$
- (nesse caso, como está próximo de 1, a relação é fraca; se fosse >1 , mostraria forte correlação).

Exemplo de KDD

- A regra **Pão ⇒ Manteiga (suporte 60%, confiança 75%)** indica que 3 em cada 4 clientes que compram pão também levam manteiga.
- O supermercado pode usar esse padrão para: Colocar pão e manteiga próximos na gôndola.
- Criar promoções conjuntas (“leve pão e ganhe desconto na manteiga”).
- Assim, o **Apriori transforma dados transacionais em conhecimento de negócio útil — exatamente a essência do KDD.**

Implementação do Exemplo

The screenshot shows a SQL query editor interface. At the top, there are two tabs: "Query" (which is selected) and "Query History". Below the tabs, the SQL code for creating a table is displayed:

```
1 -- Tabela de transações (cada compra feita por um cliente)
2 CREATE TABLE transacao (
3     id_transacao INT,
4     item VARCHAR(50)
5 );
6
```

Below the code, there are two more tabs: "Data output" and "Messages" (which is selected). The "Messages" tab displays the following output:

CREATE TABLE

Query returned successfully in 92 msec.

Implementação do Exemplo

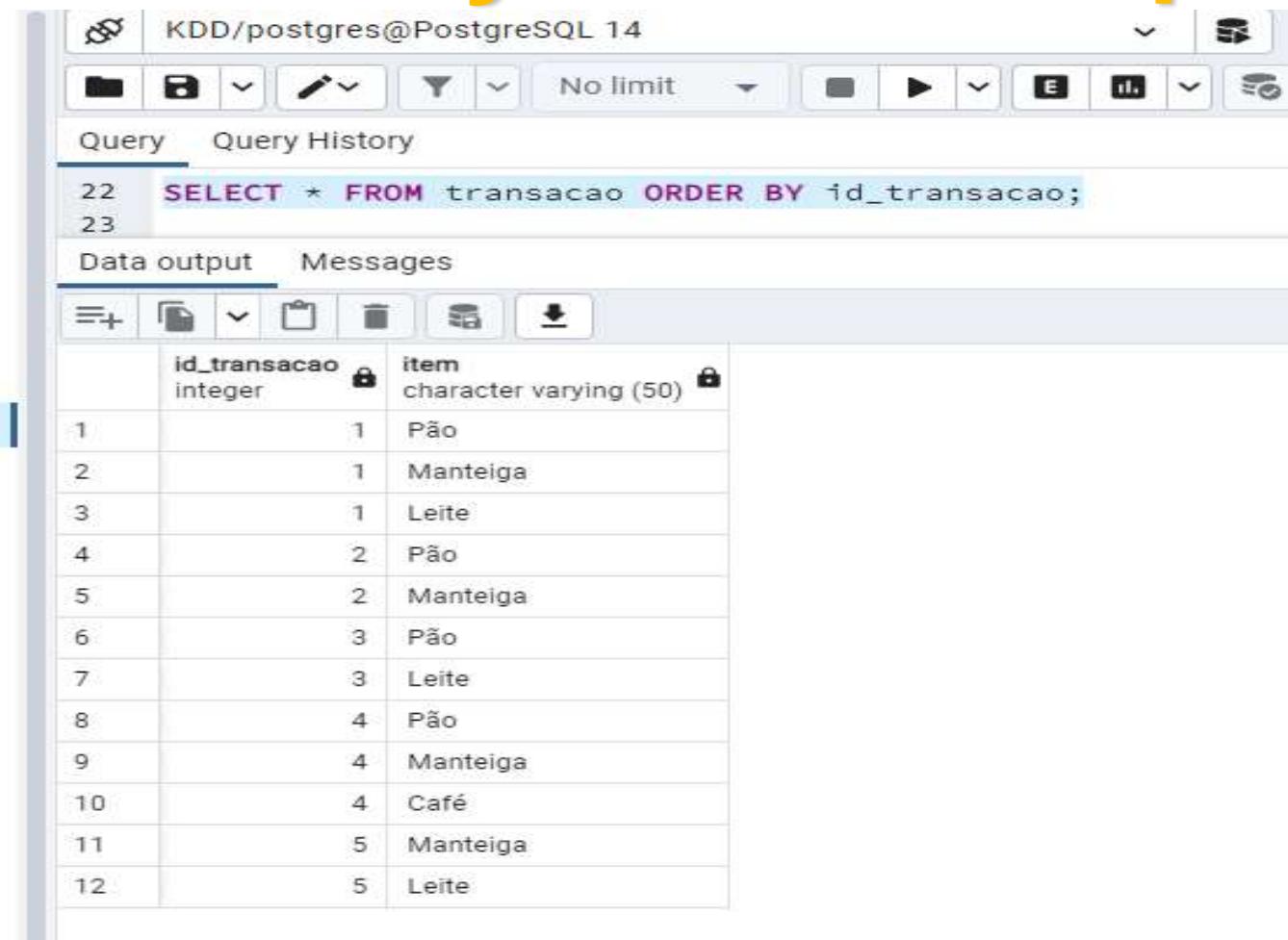
```
1 -- Tabela de transações (cada compra feita por um cliente)
2 CREATE TABLE transacao (
3     id_transacao INT,
4     item VARCHAR(50)
5 );
6
7 -- Inserindo dados de exemplo
8 INSERT INTO transacao VALUES
9 (1, 'Pão'),
10 (1, 'Manteiga'),
11 (1, 'Leite'),
12 (2, 'Pão'),
13 (2, 'Manteiga'),
14 (3, 'Pão'),
15 (3, 'Leite'),
16 (4, 'Pão'),
17 (4, 'Manteiga'),
18 (4, 'Café'),
19 (5, 'Manteiga'),
20 (5, 'Leite');
21
```

Data output Messages

INSERT 0 12

Query returned successfully in 52 msec.

Implementação do Exemplo



The screenshot shows the pgAdmin 4 interface connected to a PostgreSQL 14 database. The current connection is KDD/postgres@PostgreSQL 14. The toolbar includes standard database management icons. Below the toolbar, there are tabs for 'Query' (which is selected) and 'Query History'. A code editor window displays the following SQL query:

```
22  SELECT * FROM transacao ORDER BY id_transacao;  
23
```

Below the code editor is a 'Data output' tab, which is currently active, showing the results of the query in a grid format. The results are as follows:

	id_transacao integer	item character varying (50)
1	1	Pão
2	1	Manteiga
3	1	Leite
4	2	Pão
5	2	Manteiga
6	3	Pão
7	3	Leite
8	4	Pão
9	4	Manteiga
10	4	Café
11	5	Manteiga
12	5	Leite

Implementação do Exemplo

```
23  
24 -- Número total de transações  
25 WITH total AS (  
26     SELECT COUNT(DISTINCT id_transacao) AS total_transacoes FROM transacao  
27 )  
28 SELECT  
29     item,  
30     COUNT(DISTINCT id_transacao) AS freq,  
31     ROUND(100.0 * COUNT(DISTINCT id_transacao) / (SELECT total_transacoes FROM total), 2) AS suporte_percentual  
32 FROM transacao  
33 GROUP BY item  
34 ORDER BY suporte_percentual DESC;  
35
```

Data output Messages



	item character varying (50)	freq bigint	suporte_percentual numeric
1	Manteiga	4	80.00
2	Pão	4	80.00
3	Leite	3	60.00
4	Café	1	20.00

```
36
37 WITH total AS (
38     SELECT COUNT(DISTINCT id_transacao) AS total_transacoes FROM transacao
39 )
40 SELECT
41     t1.item AS item_a,
42     t2.item AS item_b,
43     COUNT(DISTINCT t1.id_transacao) AS freq,
44     ROUND(100.0 * COUNT(DISTINCT t1.id_transacao) / (SELECT total_transacoes FROM total), 2) AS suporte_percentual
45 FROM transacao t1
46 JOIN transacao t2
47     ON t1.id_transacao = t2.id_transacao
48     AND t1.item < t2.item -- evita pares duplicados e reflexivos
49 GROUP BY t1.item, t2.item
50 ORDER BY suporte_percentual DESC;
51
```

Data output Messages



	item_a character varying (50)	item_b character varying (50)	freq bigint	suporte_percentual numeric
1	Manteiga	Pão	3	60.00
2	Leite	Manteiga	2	40.00
3	Leite	Pão	2	40.00
4	Café	Manteiga	1	20.00
5	Café	Pão	1	20.00

```

52
53 WITH suporte_ambos AS (
54     SELECT COUNT(*) AS freq
55     FROM (
56         SELECT id_transacao
57         FROM transacao
58         WHERE item IN ('Pão', 'Manteiga')
59         GROUP BY id_transacao
60         HAVING COUNT(DISTINCT item) = 2
61     ) s
62 ),
63 suporte_pao AS (
64     SELECT COUNT(DISTINCT id_transacao) AS freq
65     FROM transacao
66     WHERE item = 'Pão'
67 )
68 SELECT
69     sa.freq AS suporte_pao_manteiga,
70     sp.freq AS suporte_pao,
71     ROUND(100.0 * sa.freq / NULLIF(sp.freq,0), 2) AS confianca_percent
72 FROM suporte_ambos sa
73 CROSS JOIN suporte_pao sp;
74

```

Data output Messages



	suporte_pao_manteiga bigint	suporte_pao bigint	confianca_percent numeric
1	3	4	75.00

Implementação do Exemplo

```
75 -- Se quiser gerar suporte/confiança para todos os pares ordenados de itens de uma vez:  
76 WITH total AS (  
77     SELECT COUNT(DISTINCT id_transacao) AS total_transacoes FROM transacao  
78 ),  
79 item_freq AS (  
80     SELECT item, COUNT(DISTINCT id_transacao) AS freq  
81     FROM transacao  
82     GROUP BY item  
83 ),  
84 pair_freq AS (  
85     SELECT t1.item AS item_a, t2.item AS item_b, COUNT(DISTINCT t1.id_transacao) AS freq  
86     FROM transacao t1  
87     JOIN transacao t2  
88     ON t1.id_transacao = t2.id_transacao  
89     AND t1.item <> t2.item  
90     GROUP BY t1.item, t2.item  
91 )
```

Implementação do Exemplo

```
90     GROUP BY t1.item, t2.item
91
92 )  
93 SELECT
94     pf.item_a,
95     pf.item_b,
96     pf.freq AS suporte_pares,
97     ROUND(100.0 * pf.freq / (SELECT total_transacoes FROM total), 2) AS suporte_percent,
98     ifreq.freq AS suporte_item_a,
99     ROUND(100.0 * pf.freq / NULLIF(ifreq.freq,0), 2) AS confianca_percent,
100    ROUND( (pf.freq::numeric / (SELECT total_transacoes FROM total))
101          / ( (SELECT freq FROM item_freq WHERE item = pf.item_b)::numeric / (SELECT total_transacoes FROM total) )
102          , 4) AS lift
103 FROM pair_freq pf
104 JOIN item_freq ifreq ON ifreq.item = pf.item_a
105 ORDER BY confianca_percent DESC;
```

Implementação do Exemplo

Data output Messages

The screenshot shows a database interface with a toolbar at the top containing icons for new, open, save, delete, refresh, and export. Below the toolbar is a table with the following data:

	item_a character varying (50)	item_b character varying (50)	suporte_pares bigint	suporte_percent numeric	suporte_item_a bigint	confianca_percent numeric	lift numeric
1	Café	Manteiga	1	20.00	1	100.00	0.2500
2	Café	Pão	1	20.00	1	100.00	0.2500
3	Pão	Manteiga	3	60.00	4	75.00	0.7500
4	Manteiga	Pão	3	60.00	4	75.00	0.7500
5	Leite	Manteiga	2	40.00	3	66.67	0.5000
6	Leite	Pão	2	40.00	3	66.67	0.5000
7	Manteiga	Leite	2	40.00	4	50.00	0.6667
8	Pão	Leite	2	40.00	4	50.00	0.6667
9	Manteiga	Café	1	20.00	4	25.00	1.0000
10	Pão	Café	1	20.00	4	25.00	1.0000

Dados, Informações ...

- Os conceitos **dados** (data), **informação** (information), **conhecimento** (knowledge) e **sabedoria** (wisdom) estão relacionados, mas têm níveis diferentes de abstração e valor.
- Eles costumam ser explicados pela chamada **pirâmide DIKW** (Data → Information → Knowledge → Wisdom).

Dados, Informações ...

- **1. Dados (Data):**
- O que são: **Elementos brutos, fatos isolados, sem contexto ou interpretação.**
- Exemplo: **35, azul, 2025-09-08, Maria.**
- Analogia: **Letras soltas de um alfabeto.**

Dados, Informações ...

- **2. Informação (Information):**
- O que é: **Dados processados, organizados e contextualizados de forma a terem significado.**
- Exemplo: “A temperatura hoje é 35°C.” (o dado 35 agora está **contextualizado** como **temperatura**).
- **Analogia:** Palavras formadas a partir das letras.

Dados, Informações ...

- **3. Conhecimento (Knowledge):**
- O que é: **Conjunto de informações analisadas, interpretadas e compreendidas, permitindo gerar entendimento e experiência.**
- Exemplo: “**Quando a temperatura chega a 35°C, é provável que haja aumento no consumo de bebidas geladas.**”
- **Analogia: Frases que transmitem uma ideia ou conceito.**

Dados, Informações ...

- **4. Sabedoria (Wisdom):**
- O que é: Capacidade de aplicar o conhecimento de forma ética, crítica e útil para tomar decisões acertadas.
- Exemplo: “Diante da previsão de 35°C, aumentaremos o estoque de bebidas geladas para evitar falta e desperdício.”
- Analogia: Um texto coerente que traz reflexão e guia ação.

Um Outro Exemplo

- **Segmentação de clientes em uma academia:**
- Imagine que uma academia registrou informações sobre seus clientes. Para simplificar, usaremos dois atributos: Idade e Frequência semanal de visitas (quantas vezes por semana o cliente vai treinar).

Um Outro Exemplo

Os dados fictícios (10 clientes):

Cliente	Idade	Frequência (vezes/semana)
C1	18	5
C2	20	4
C3	22	3
C4	25	2
C5	30	1
C6	35	2
C7	40	1
C8	45	1
C9	50	2
C10	60	1

Um Outro Exemplo

- **Aplicando o algoritmo de Machine Learning K-Means (K=3 clusters):**
- 1. **Escolha do número de clusters (K=3) // Queremos separar clientes em 3 grupos distintos.**
- 2. **Inicialização:** O algoritmo escolhe 3 centróides iniciais (pontos de referência no espaço “idade × frequência”).
- 3. **Atribuição:** Cada cliente é atribuído ao cluster cujo centróide está mais próximo (usando distância euclidiana).
- 4. **Recalcular centróides:** Depois que todos os clientes são atribuídos, o algoritmo recalcula a posição dos centróides como a média dos pontos de cada grupo.
- 5. **Iteração:** Os passos 3 e 4 se repetem até os centróides pararem de mudar significativamente (convergência).

Um Outro Exemplo

- **1) Inicialização:**
- **Escolhemos 3 centróides iniciais (como se fossem “representantes” provisórios dos grupos):**
- **A = C1 = (18, 5)**
- **B = C5 = (30, 1)**
- **C = C10 = (60, 1)**
- **Distância usada: euclidiana.** Para comparar “quem está mais perto”, podemos usar as distâncias ao quadrado (evita raiz):

$$d^2((x, y), (a, b)) = (x - a)^2 + (y - b)^2$$

Um Outro Exemplo

- 2) Iteração 1 — Atribuição aos centróides (com $A=(18,5)$, $B=(30,1)$, $C=(60,1)$):

Um Outro Exemplo

Cliente	Idade (x)	Frequência/semana (y)
C1	18	5
C2	20	4
C3	22	3
C4	25	2
C5	30	1
C6	35	2
C7	40	1
C8	45	1
C9	50	2
C10	60	1

Um Outro Exemplo

- Cliente **C4**: para qual cluster será designado?
- **C4(25,2)** tem **25 anos** e **5 idas semanais** à academia.
- O cluster 1 é liderado por C1(18,5):
- $D^2(C4, C1) = (25 - 18)^2 + (2 - 5)^2$
- $D^2(C4, C1) = (7)^2 + (-3)^2 = 49 + 9 = 58$

Um Outro Exemplo

- O cluster 2 é liderado por **C2(20,4)**:
- $D^2(C4, C2) = (25 - 20)^2 + (2 - 4)^2$
- $D^2(C4, C2) = (5)^2 + (-2)^2 = 25 + 4 = 29$
- O cluster 3 é liderado por **C3(22,3)**:
- $D^2(C4, C3) = (25 - 22)^2 + (2 - 3)^2$
- $D^2(C4, C3) = (3)^2 + (-1)^2 = 9 + 1 = 10$

Um Outro Exemplo

- A distância de C4 para C1 (cluster 1) é 58.
- A distância de C4 para C2 (cluster 2) é 29.
- A distância de C4 para C3 (cluster 3) é 10.
- Portanto, **C4** será transferido para o **cluster 3** (a menor distância – o mais próximo).

Um Outro Exemplo

- O mesmo deve ser feito com C5, C6, ..., C10.
- Cada um será designado para um dos 3 clusters.
- Cliente **C5**: para qual cluster será designado?
- **C5(30,1)** tem **30 anos** e **1 ida semanal** à academia.

Um Outro Exemplo

- O cluster 1 é liderado por C1(18,5):
- $D^2(C5, C1) = (30 - 18)^2 + (1 - 5)^2$
- $D^2(C5, C1) = (12)^2 + (-4)^2 = 144 + 16 = 160$
- O cluster 2 é liderado por C2(20,4):
- $D^2(C5, C2) = (30 - 20)^2 + (1 - 4)^2$
- $D^2(C5, C2) = (6)^2 + (-3)^2 = 36 + 9 = 45$

Um Outro Exemplo

- O cluster 3 é liderado por **C3(22,3)**:
- $D^2(C5, C3) = (30 - 22)^2 + (1 - 3)^2$
- $D^2(C5, C3) = (8)^2 + (-2)^2 = 64 + 4 = 68$
- Distância de C5 para cluster 1: 160
- Distância de C5 para cluster 2: 45
- Distância de C5 para cluster 3: 68
- **C5 vai para o cluster 2!**

Um Outro Exemplo

- Cluster 1 {C1}
- Cluster 2 {C2, C5}
- Cluster 3 {C3, C4}
- Cliente C6: para qual cluster será designado?
- C6(35,2) tem 35 anos e 2 idas semanais à academia.

Um Outro Exemplo

- O cluster 1 é liderado por C1(18,5):
- $D^2(C_6, C_1) = (35 - 18)^2 + (2 - 5)^2$
- $D^2(C_6, C_1) = (17)^2 + (-3)^2 = 289 + 9 = 298$
- O cluster 2 é liderado por C2(20,4):
- $D^2(C_6, C_2) = (35 - 20)^2 + (2 - 4)^2$
- $D^2(C_6, C_2) = (15)^2 + (-2)^2 = 225 + 4 = 229$

Um Outro Exemplo

- O cluster 3 é liderado por C3(22,3):
- $D^2(C6, C3) = (35 - 22)^2 + (2 - 3)^2$
- $D^2(C6, C3) = (13)^2 + (-1)^2 = 169 + 1 = 170$
- Distância de C6 para cluster 1: 298
- Distância de C6 para cluster 2: 229
- Distância de C6 para cluster 3: 170
- C6 vai para o cluster 3!

Um Outro Exemplo

- Cluster 1 {C1}
- Cluster 2 {C2, C5}
- Cluster 3 {C3, C4, C6}
- Cliente C7: para qual cluster será designado?
- C7(40,1) tem 40 anos e 1 ida semanal à academia.

Um Outro Exemplo

- O cluster 1 é liderado por C1(18,5):
- $D^2(C7, C1) = (40 - 18)^2 + (1 - 5)^2$
- $D^2(C7, C1) = (22)^2 + (-4)^2 = 484 + 16 = 500$
- O cluster 2 é liderado por C2(20,4):
- $D^2(C7, C2) = (40 - 20)^2 + (1 - 4)^2$
- $D^2(C7, C2) = (20)^2 + (-3)^2 = 400 + 9 = 409$

Um Outro Exemplo

- O cluster 3 é liderado por C3(22,3):
- $D^2(C7, C3) = (40 - 22)^2 + (1 - 3)^2$
- $D^2(C7, C3) = (18)^2 + (-2)^2 = 324 + 4 = 328$
- Distância de C7 para cluster 1: 500
- Distância de C7 para cluster 2: 409
- Distância de C7 para cluster 3: 328
- **C7 vai para o cluster 3!**

Um Outro Exemplo

- Cluster 1 {C1}
- Cluster 2 {C2, C5}
- Cluster 3 {C3, C4, C6, C7}
- Cliente C8: para qual cluster será designado?
- C8(45,1) tem 45 anos e 1 ida semanal à academia.

Um Outro Exemplo

- O cluster 1 é liderado por C1(18,5):
- $D^2(C8, C1) = (45 - 18)^2 + (1 - 5)^2$
- $D^2(C8, C1) = (27)^2 + (-4)^2 = 729 + 16 = 745$
- O cluster 2 é liderado por C2(20,4):
- $D^2(C8, C2) = (45 - 20)^2 + (1 - 4)^2$
- $D^2(C8, C2) = (25)^2 + (-3)^2 = 625 + 9 = 634$

Um Outro Exemplo

- O cluster 3 é liderado por C3(22,3):
- $D^2(C8, C3) = (45 - 22)^2 + (1 - 3)^2$
- $D^2(C8, C3) = (23)^2 + (-2)^2 = 529 + 4 = 533$
- Distância de C8 para cluster 1: 745
- Distância de C8 para cluster 2: 634
- Distância de C8 para cluster 3: 533
- **C8 vai para o cluster 3!**

Um Outro Exemplo

- Cluster 1 {C1}
- Cluster 2 {C2, C5}
- Cluster 3 {C3, C4, C6, C7, C8}
- Cliente C9: para qual cluster será designado?
- C9(50,2) tem 50 anos e 2 idas semanais à academia.

Um Outro Exemplo

- O cluster 1 é liderado por **C1(18,5)**:
- $D^2(C9,C1) = (50 - 18)^2 + (2 - 5)^2$
- $D^2(C9,C1) = (32)^2 + (-3)^2 = 1024 + 9 = 1033$
- O cluster 2 é liderado por **C2(20,4)**:
- $D^2(C9,C2) = (50 - 20)^2 + (2 - 4)^2$
- $D^2(C9,C2) = (30)^2 + (-2)^2 = 900 + 4 = 904$

Um Outro Exemplo

- O cluster 3 é liderado por C3(22,3):
- $D^2(C9,C3) = (50 - 22)^2 + (2 - 3)^2$
- $D^2(C9,C3) = (28)^2 + (-1)^2 = 784 + 1 = 785$
- Distância de C8 para cluster 1: 1033
- Distância de C8 para cluster 2: 904
- Distância de C8 para cluster 3: 785
- C9 vai para o cluster 3!

Um Outro Exemplo

- Cluster 1 {C1}
- Cluster 2 {C2, C5}
- Cluster 3 {C3, C4, C6, C7, C8, C9}
- Cliente C10: para qual cluster será designado?
- C10(60,1) tem 60 anos e 1 ida semanal à academia.

Um Outro Exemplo

- O cluster 1 é liderado por $C1(18,5)$:
- $D^2(C10,C1) = (60 - 18)^2 + (1 - 5)^2$
- $D^2(C10,C1) = (42)^2 + (-4)^2 = 1764 + 16 = 1780$
- O cluster 2 é liderado por $C2(20,4)$:
- $D^2(C10,C2) = (60 - 20)^2 + (1 - 4)^2$
- $D^2(C10,C2) = (40)^2 + (-3)^2 = 1600 + 9 = 1609$

Um Outro Exemplo

- O cluster 3 é liderado por C3(22,3):
- $D^2(C10, C3) = (60 - 22)^2 + (1 - 3)^2$
- $D^2(C10, C3) = (38)^2 + (-2)^2 = 1444 + 4 = 1448$
- Distância de C10 para cluster 1: 1780
- Distância de C10 para cluster 2: 1609
- Distância de C10 para cluster 3: 1448
- **C10 vai para o cluster 3!**

Um Outro Exemplo

- Cluster 1 {C1}
- Cluster 2 {C2, C5}
- Cluster 3 {C3, C4, C6, C7, C8, C9, c10}
- Percebemos que não resultou em uma equilibrada distribuição de clientes.
- O cluster 3 recebeu a maioria dos clientes, exceto os centróides (C1 e C2) e o cliente C5.
- Não é um bom resultado!

Um Outro Exemplo

- 3) Iteração 1 — Recalcular centróides (média de cada grupo):
 - Cluster 1: {C1(18,5)}
 - Cluster 1: Média(18,5).
 - Cluster 2: {C2(20,4), C5(30,1)}
 - Cluster 2: Média(25, 2.5)

Um Outro Exemplo

- Cluster 3: {C3(22,3), C4(25,2), C6(35,2),
C7(40,1), C8(45,1), C9(50,2), C10(60,1)}
- Cluster 3: Média($277/7$, $12/7$).
- Cluster 3: Média(39.57, 1.71).

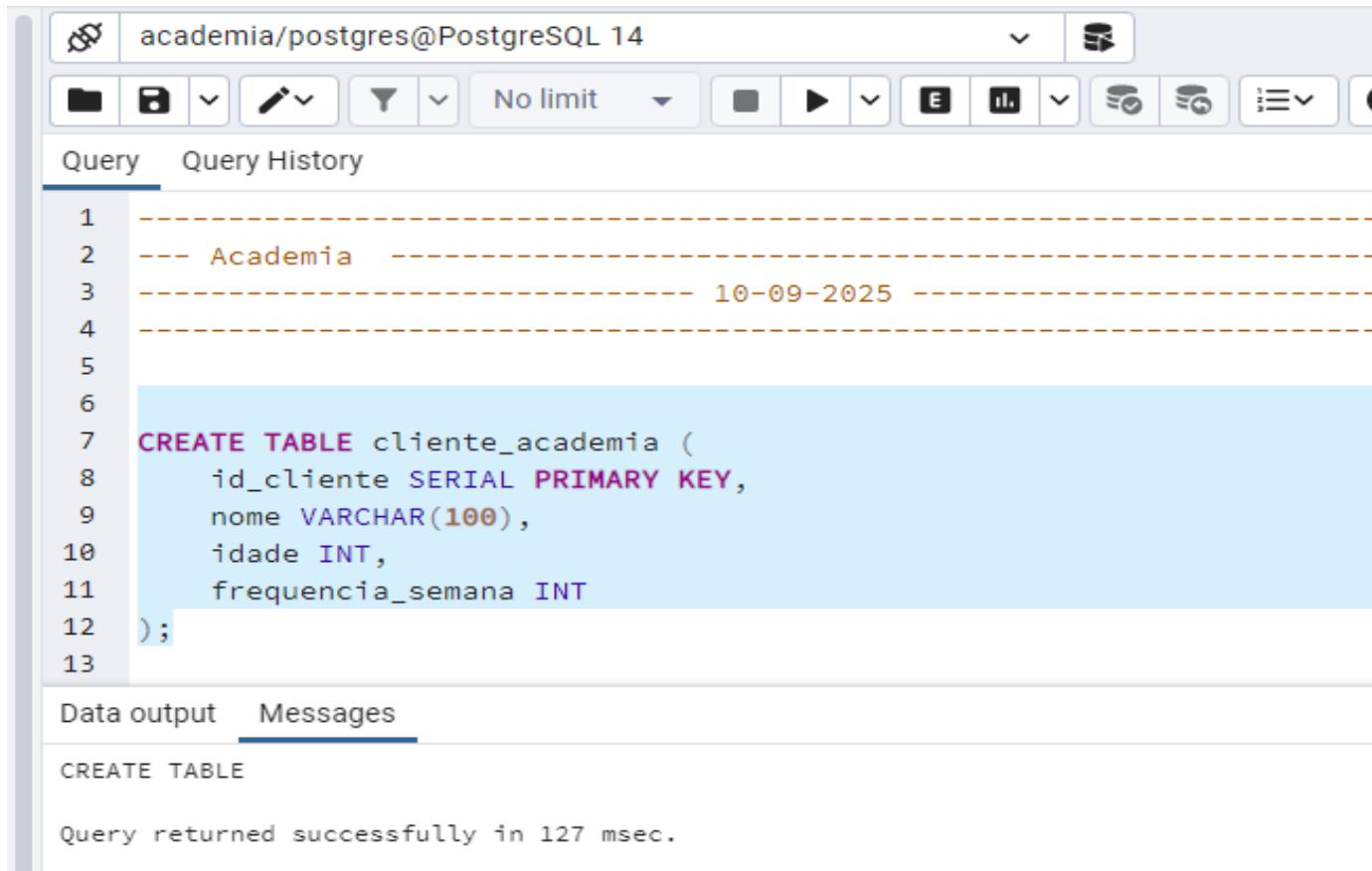
Um Outro Exemplo

- 4) Iteração 2 — Nova atribuição:
- Centróide do cluster 1: **média(18,5).**
- Novo centróide do cluster 2: **média(25, 2.5).**
- Novo centróide do cluster 3: **média(39.57, 1.71).**

Um Outro Exemplo

- Agora devemos refazer os cálculos da distância euclidiana para efetuar uma nova distribuição dos clientes (C1, C2, C3, ..., C10) pelos diferentes clusters.
- Então efetuamos novamente o cálculo de novas médias para determinar novos centróides.
- Em seguida redistribuímos novamente os clientes pelos clusters existentes.
- Caso não ocorra uma mudança nessa distribuição, o algoritmo se encerra.
- Caso contrário repetimos a etapa de cálculo das novas médias e redistribuição dos clientes.

Implementação do Exemplo 2



The screenshot shows the pgAdmin 4 interface for PostgreSQL 14. The connection is set to 'academia/postgres@PostgreSQL 14'. The toolbar includes various icons for database management. The main window has two tabs: 'Query' (selected) and 'Query History'. The 'Query' tab contains the following SQL code:

```
1 ---  
2 --- Academia ---  
3 --- 10-09-2025 ---  
4 ---  
5  
6  
7 CREATE TABLE cliente_academia (  
8     id_cliente SERIAL PRIMARY KEY,  
9     nome VARCHAR(100),  
10    idade INT,  
11    frequencia_semana INT  
12 );  
13
```

The code is highlighted in blue, indicating it is a SQL statement. Below the code, the 'Messages' tab shows the execution results:

CREATE TABLE

Query returned successfully in 127 msec.

Implementação do Exemplo 2

```
6
7 CREATE TABLE cliente_academia (
8     id_cliente SERIAL PRIMARY KEY,
9     nome VARCHAR(100),
10    idade INT,
11    frequencia_semana INT
12 );
13
14 -- Exemplo de dados
15 INSERT INTO cliente_academia (nome, idade, frequencia_semana) VALUES
16 ('Ana', 22, 5),
17 ('Bruno', 35, 3),
18 ('Carla', 28, 4),
19 ('Diego', 42, 2),
20 ('Elisa', 19, 6),
21 ('Fernando', 55, 1),
22 ('Gabriela', 31, 3),
23 ('Henrique', 40, 2);
24
```

Data output Messages

INSERT 0 8

Query returned successfully in 84 msec.

Implementação do Exemplo 2

```
24  
25 CREATE OR REPLACE FUNCTION exportar_csv_academia(caminho TEXT)  
26 RETURNS void AS $$  
27 BEGIN  
28     EXECUTE format(  
29         $f$  
30         COPY (  
31             SELECT idade, frequencia_semana  
32             FROM cliente_academia  
33         )  
34         TO %L  
35         WITH CSV HEADER;  
36         $f$, caminho  
37     );  
38 END;  
39 $$ LANGUAGE plpgsql;  
40
```

Data output Messages

CREATE FUNCTION

Query returned successfully in 67 msec.

Implementação do Exemplo 2

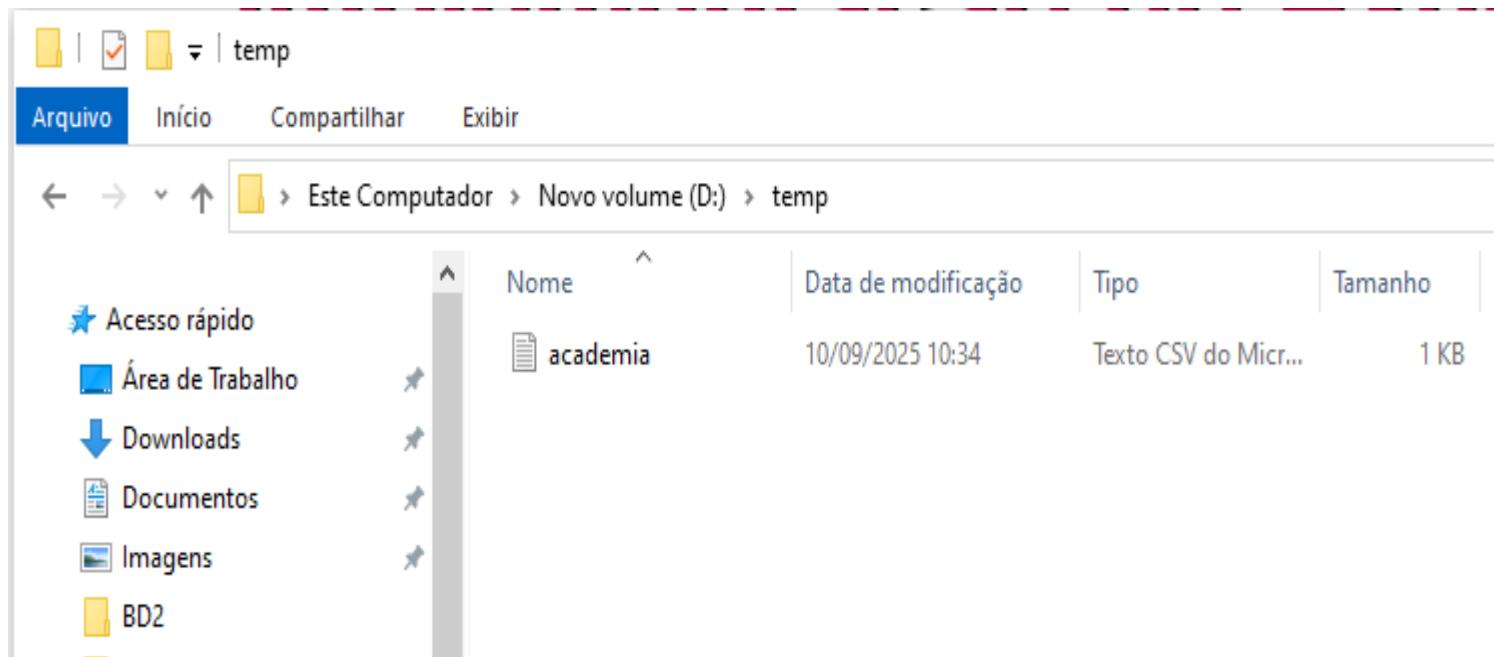
```
24
25 CREATE OR REPLACE FUNCTION exportar_csv_academia(caminho TEXT)
26 RETURNS void AS $$ 
27 BEGIN
28     EXECUTE format(
29         $f$ 
30         COPY (
31             SELECT idade, frequencia_semana
32             FROM cliente_academia
33         )
34         TO %L
35         WITH CSV HEADER;
36         $f$, caminho
37     );
38 END;
39 $$ LANGUAGE plpgsql;
40
41 SELECT exportar_csv_academia('D:/temp/academia.csv');
42
```

Data output Messages

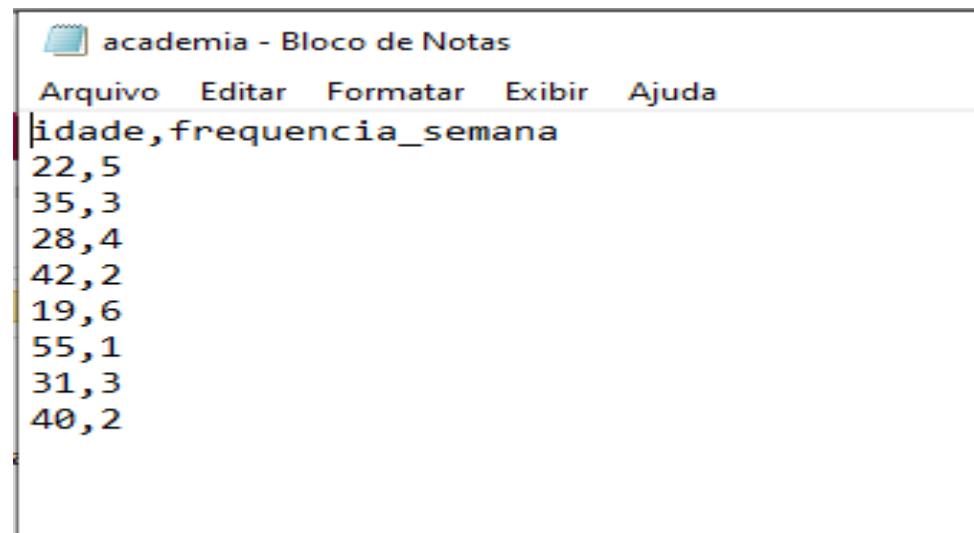


	exportar_csv_academia	locked
1		

Implementação do Exemplo 2



Implementação do Exemplo 2



The image shows a screenshot of a Windows Notepad window. The title bar reads "academia - Bloco de Notas". Below the title bar is a menu bar with options: Arquivo, Editar, Formatar, Exibir, and Ajuda. The main content area of the notepad contains a list of data points, each consisting of two values separated by a comma. The data points are:

idade	frequencia_semana
22	5
35	3
28	4
42	2
19	6
55	1
31	3
40	2

Implementação do Exemplo 2

The screenshot shows a web browser window with the URL ilyakuzovkin.com/csv2arff/. The page title is "CSV2ARFF" and the subtitle is "Online converter from .csv to WEKA .arff". On the left, a sidebar has a blue header "CSV2ARFF" and two items: "1. Upload" and "2. Generate". The main content area has a large "Upload" heading, a note about the file size limit, and form fields for "Filename" (with a button "Escolher arquivo" and placeholder "Nenhum arquivo escolhido"), "Delimiter" (with a value ","), and a "Submit" button.

O Google Chrome não é seu navegador padrão

Definir como padrão

CSV2ARFF

Online converter from .csv to WEKA .arff

Upload

Current file size limit is 100 MBytes.

Filename Nenhum arquivo escolhido

Delimiter

Submit

Implementação do Exemplo 2

CSV2ARFF

Online converter from .csv to WEKA .arff

Upload

Current file size limit is 100 MBytes.

Filename academia.csv

Delimiter

Implementação do Exemplo 2

Online converter from .csv to WEKA .arff

Configure

Your file was uploaded.

Now we will carefully save it and parse.

Meanwhile you can choose the parameters of your dataset.

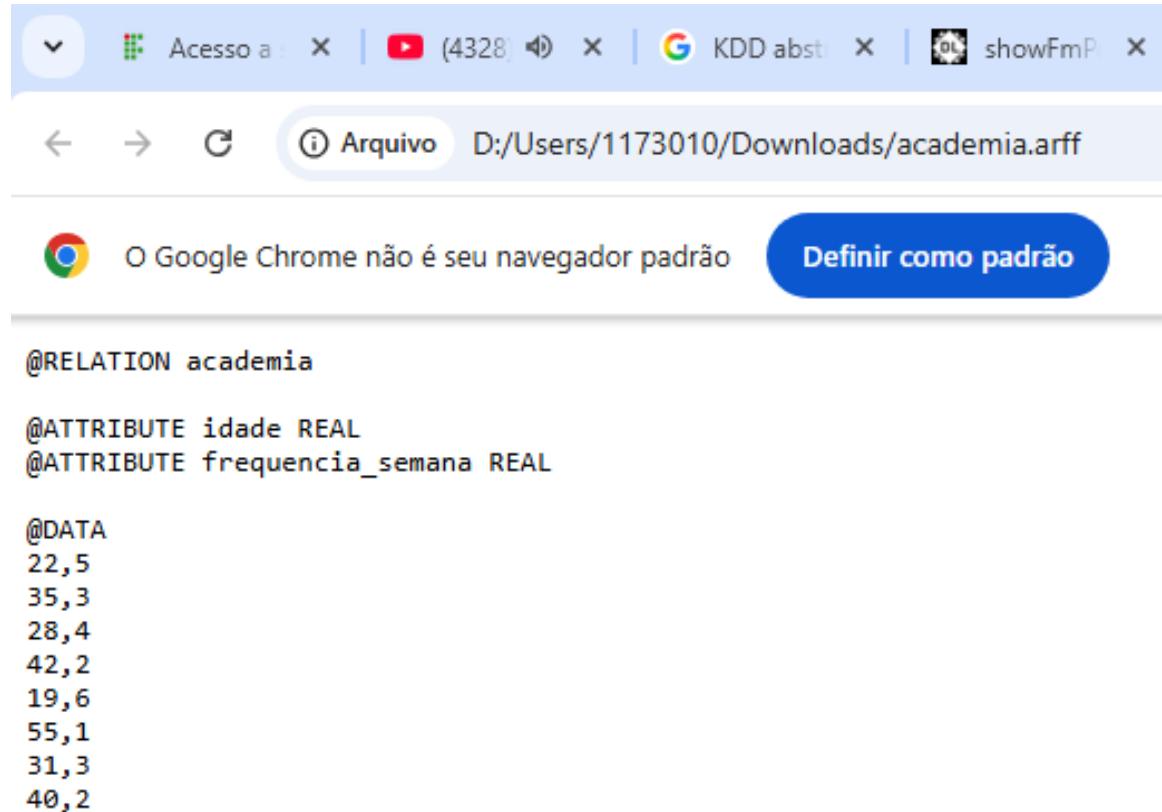
The last column, where most probably your class label is, should be Nominal.

First row contains labels

First row	Second row	Type
idade	22	<input checked="" type="radio"/> Numeric <input type="radio"/> Nominal <input type="radio"/> skip
frequencia_semana5		<input checked="" type="radio"/> Numeric <input type="radio"/> Nominal <input type="radio"/> skip

[Generate my ARFF](#)

Implementação do Exemplo 2



A screenshot of a web browser window displaying an ARFF (Attribute-Relationship File Format) file. The browser's address bar shows the file path: D:/Users/1173010/Downloads/academia.arff. The content of the page is the ARFF code:

```
@RELATION academia  
  
@ATTRIBUTE idade REAL  
@ATTRIBUTE frequencia_semana REAL  
  
@DATA  
22,5  
35,3  
28,4  
42,2  
19,6  
55,1  
31,3  
40,2
```

The browser interface includes standard navigation buttons (back, forward, search), a refresh button, and a status bar message: "O Google Chrome não é seu navegador padrão" with a "Definir como padrão" button.

Implementação do Exemplo 2



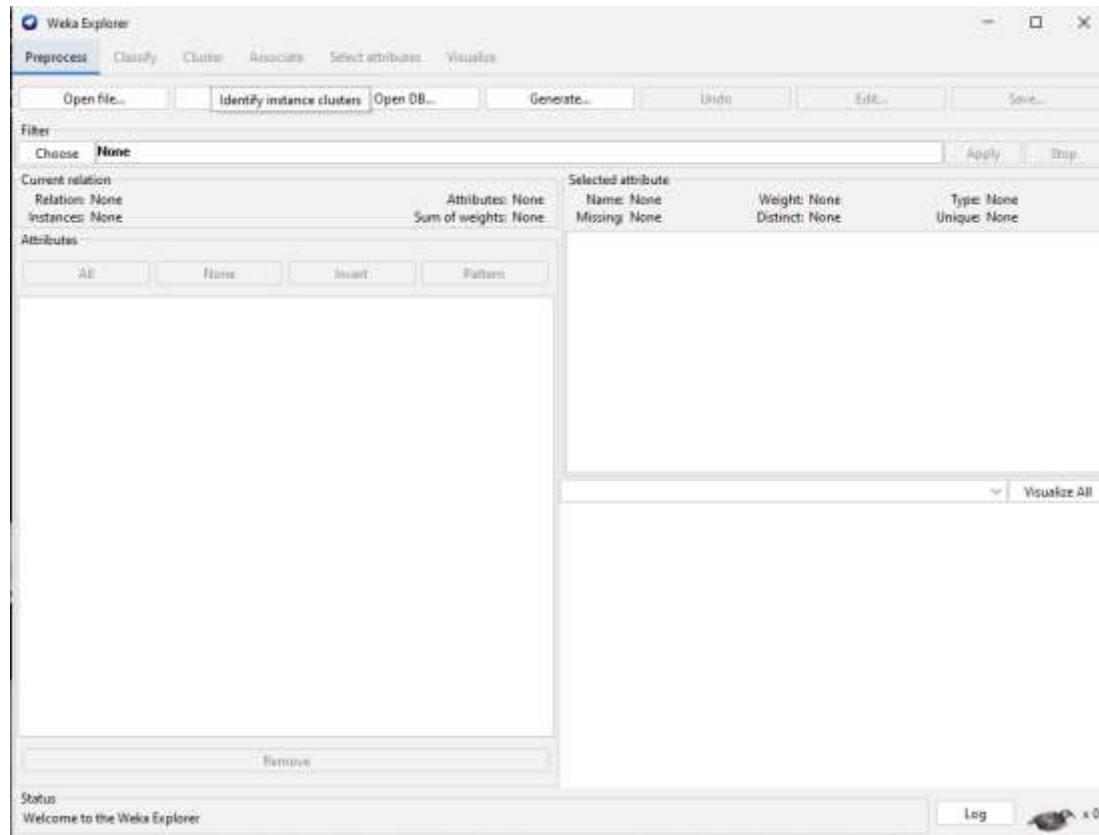
**Agora, de posse
do arquivo .ARFF,
vamos abrir o
WEKA.**

Implementação do Exemplo 2



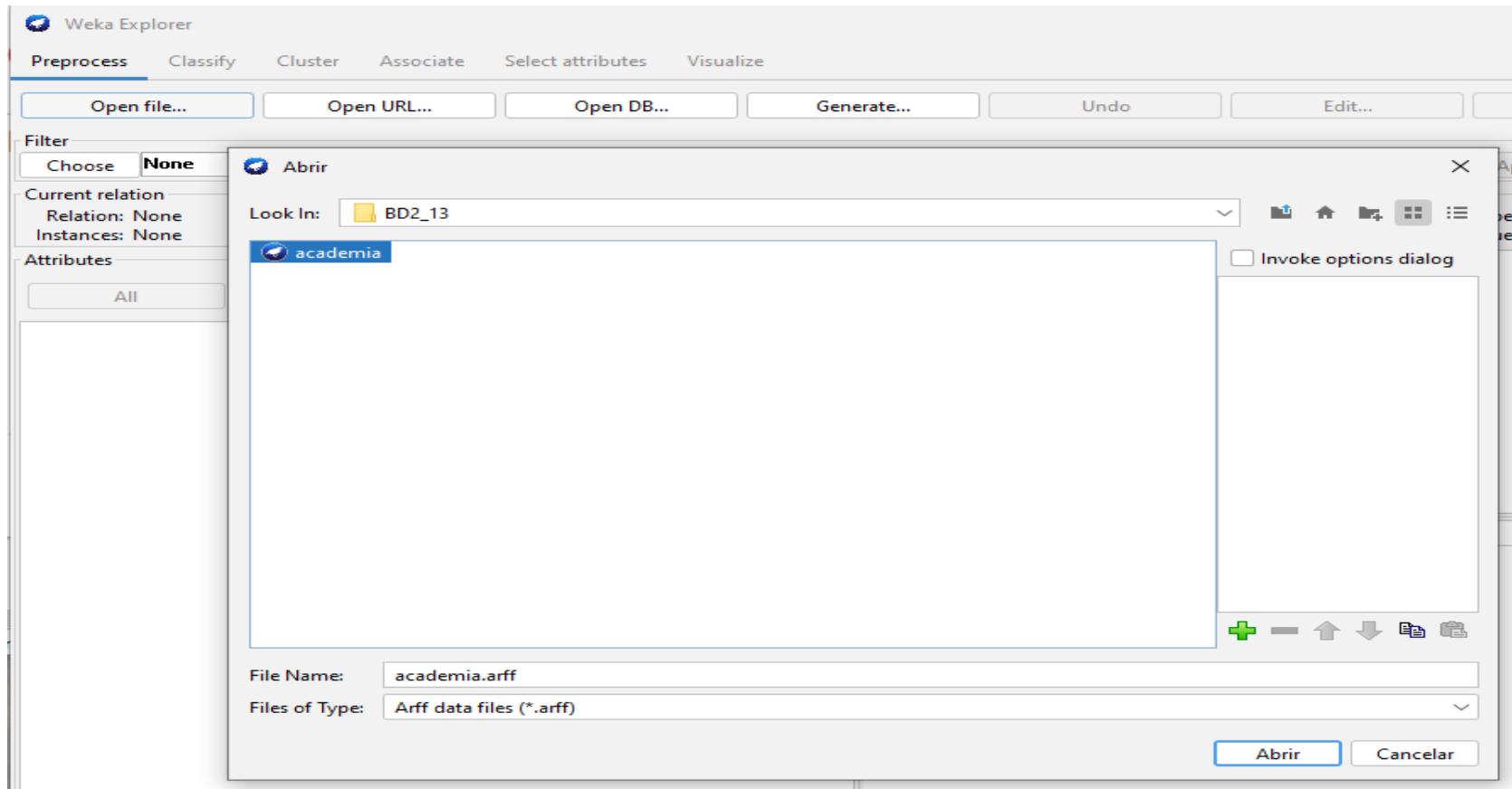
Clique no botão EXPLORER.

Implementação do Exemplo 2

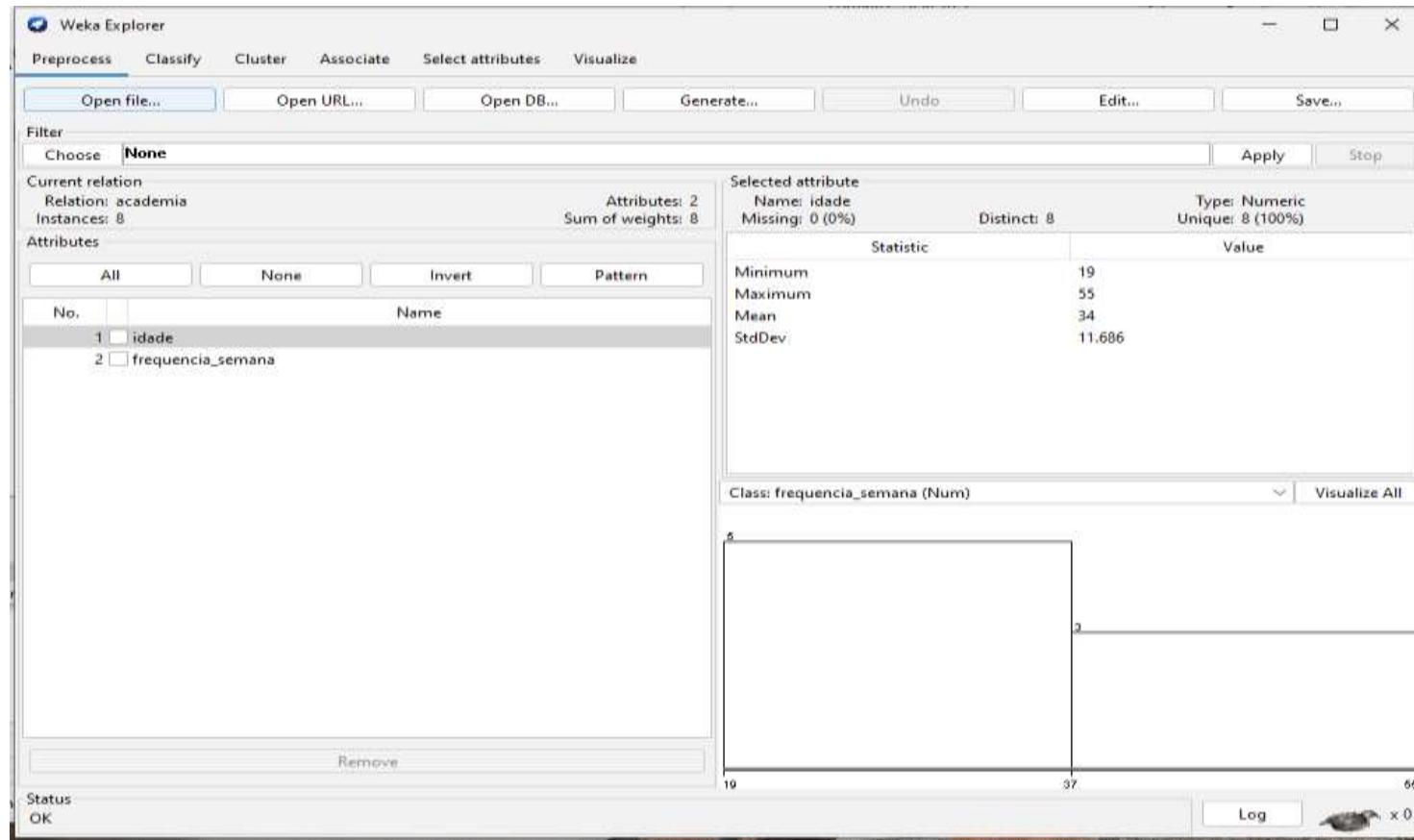


Clique no botão **OPEN FILE**
e selecione o arquivo **ARFF**.

Implementação do Exemplo 2



Implementação do Exemplo 2



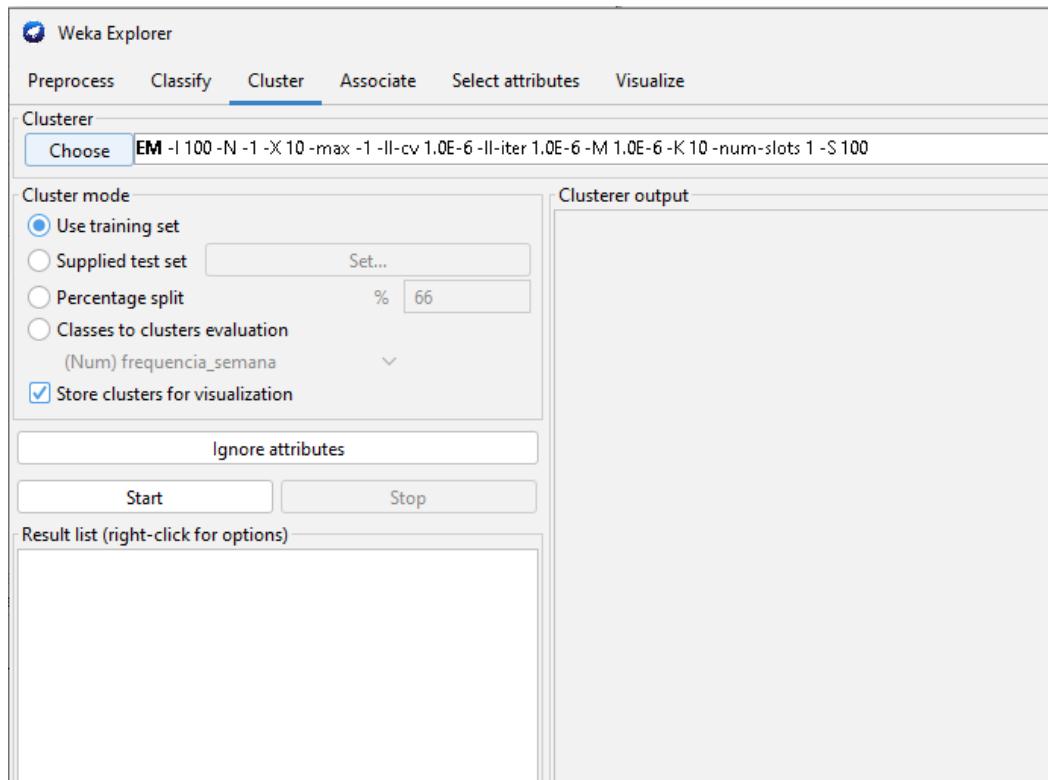
Coluna **idade** selecionada.

Idade mínima 19 anos.

Idade máxima 55 anos.

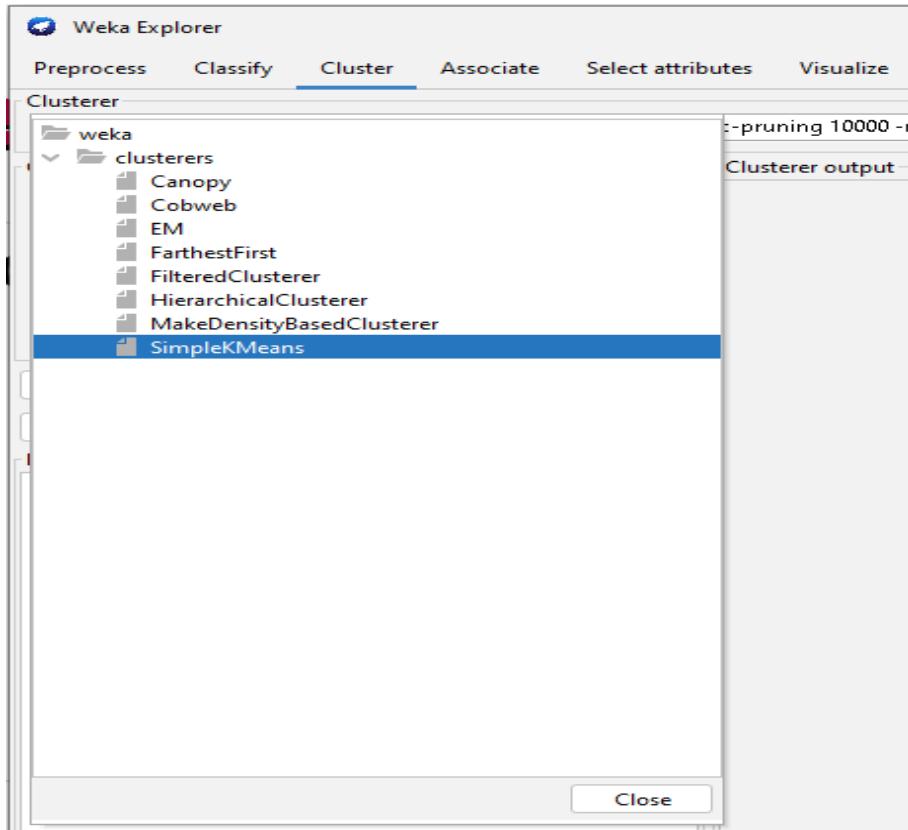
Selecione a aba **cluster** (ou agrupamento).

Implementação do Exemplo 2



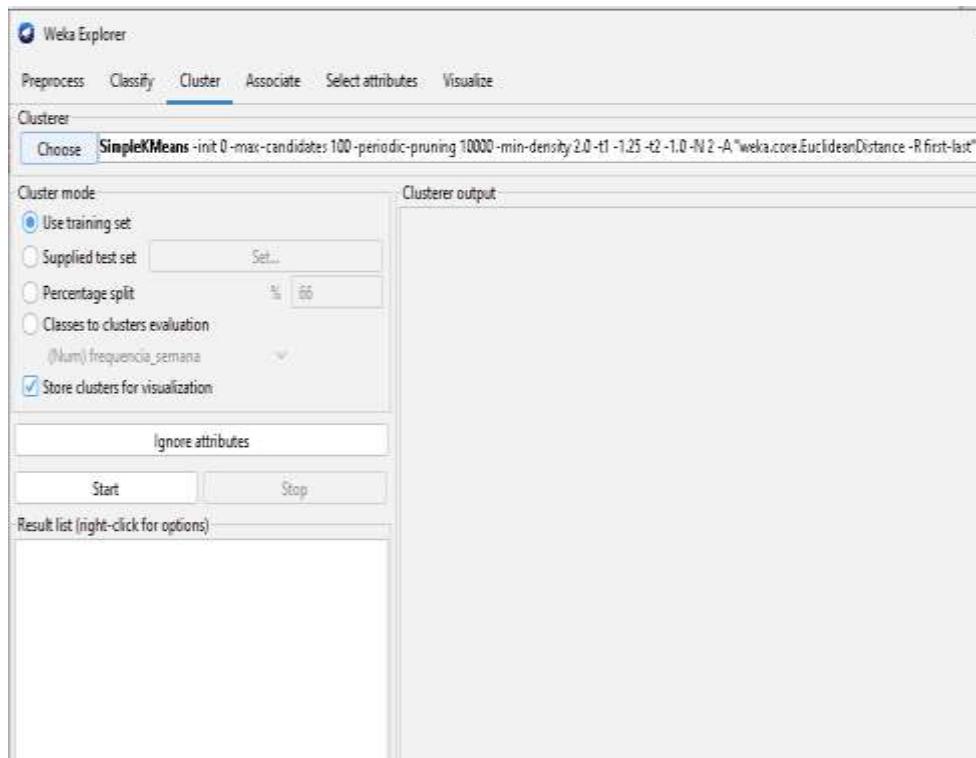
Agora clique no **botão choose** para selecionar o **algoritmo de aprendizagem de máquina** a ser utilizado.

Implementação do Exemplo 2



Selecione o
**SimpleKMeans (K-
Means).**

Implementação do Exemplo 2



Como nosso arquivo apresenta poucos clientes a serem agrupados (apenas 8 instâncias) vamos usar todos eles no treinamento.

O teste será feito com

Implementação do Exemplo 2

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Clusterer

Choose **SimpleKMeans** -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 2 -A "weka.core.EuclideanDistance -R first-last" -l 500 -num-slots 1

Cluster mode

Use training set
 Supplied test set Set...
 Percentage split % 66
 Classes to clusters evaluation (Num) frequencia_semana
 Store clusters for visualization

Ignore attributes
Start Stop

Result list (right-click for options)

13:24:35 - SimpleKMeans

Clusterer output

=====

Number of iterations: 4
Within cluster sum of squared errors: 0.5322057613166724

Initial starting points (random):

Cluster 0: 55,1
Cluster 1: 35,3

Missing values globally replaced with mean/mode

Final cluster centroids:

Attribute	Full Data	Cluster# 0	Cluster# 1
	(8.0)	(3.0)	(5.0)
idade	34	45.6667	27
frequencia_semana	3.25	1.6667	4.2

Time taken to build model (full training data) : 0 seconds

--- Model and evaluation on training set ---

Clustered Instances

0	3 (38%)
1	5 (63%)

Status OK

Log x 0

Implementação do Exemplo 2

```
Clusterer output
=====
Number of iterations: 4
Within cluster sum of squared errors: 0.5322057613168724

Initial starting points (random):

Cluster 0: 55,1
Cluster 1: 35,3

Missing values globally replaced with mean/mode

Final cluster centroids:
          Cluster#
Attribute      Full Data      0        1
                  (8.0)   (3.0)   (5.0)
=====
idade           34    45.6667    27
frequencia_semana  3.25    1.6667    4.2

Time taken to build model (full training data) : 0 seconds

== Model and evaluation on training set ==

Clustered Instances

0      3 ( 38%)
1      5 ( 63%)
```

Note que o algoritmo distribuiu os clientes (instâncias) entre 2 clusters (agrupamentos).

K = 2.

Implementação do Exemplo 2

```
Clusterer output

Scheme:      weka.clusterers.SimpleKMeans -init 0 -max-candidate
Relation:    academia
Instances:   8
Attributes:  2
            idade
            frequencia_semana
Test mode:   evaluate on training data

*** Clustering model (full training set) ***

kMeans
=====

Number of iterations: 4
Within cluster sum of squared errors: 0.5322057613168724

Initial starting points (random):

Cluster 0: 55,1
Cluster 1: 35,3

Missing values globally replaced with mean/mode

Final cluster centroids:

          Cluster#
Attribute   Full Data      0        1
              (8.0)   (3.0)   (5.0)
=====
idade           34   45.6667    27
frequencia_semana  3.25   1.6667   4.2
```

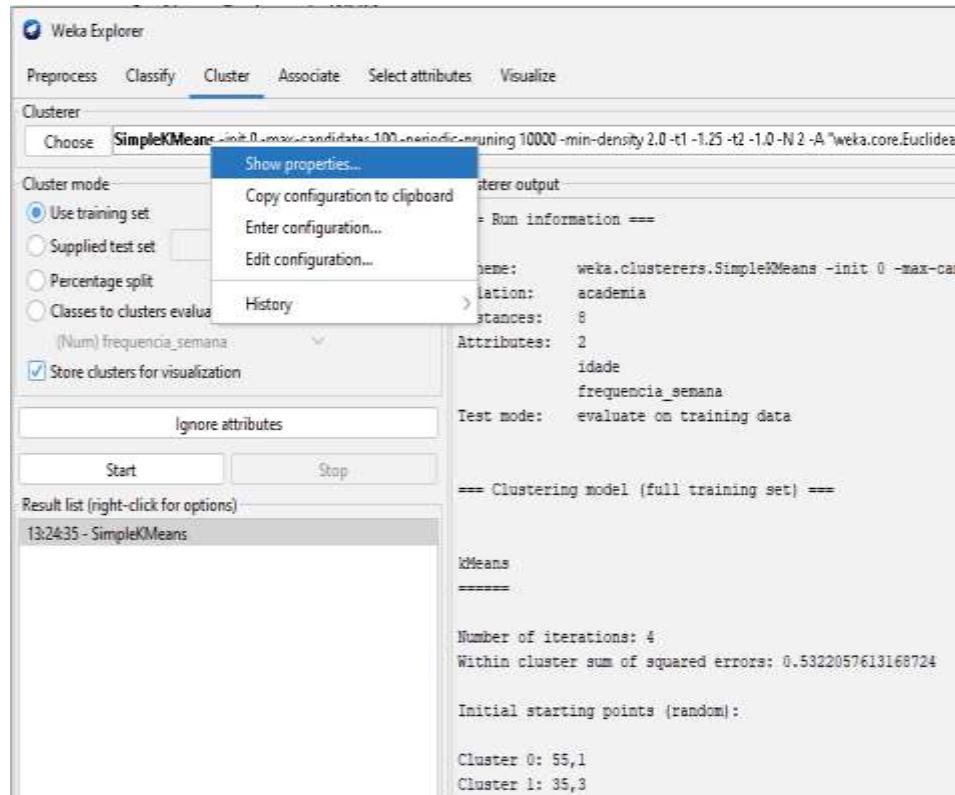
Note que os centróides iniciais foram:

- Cluster 0: idade=55, freq= 1.
- Cluster 1: idade= 35, freq=3.

Mas, no final, ficaram:

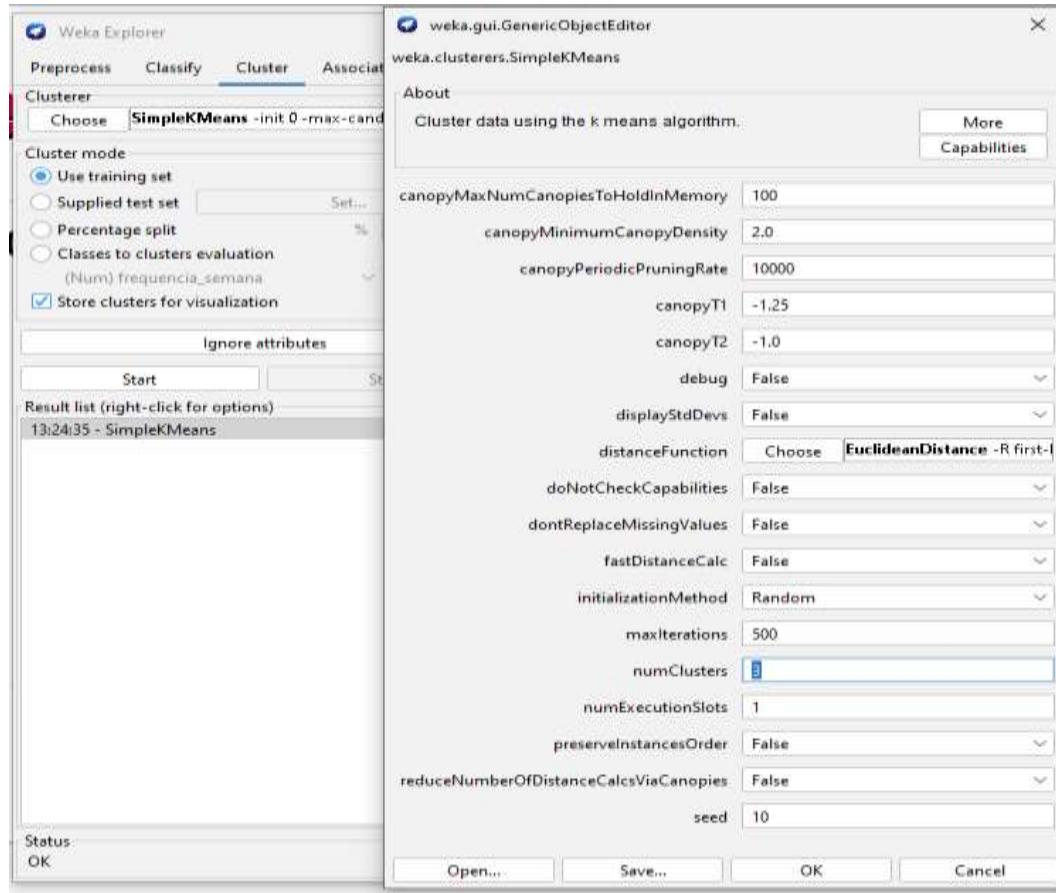
- Cluster 0: idade=45.67, freq= 1.67
- Cluster 1: idade=27, freq= 4.2

Implementação do Exemplo 2



Clique com o botão direito do mouse sobre o nome do algoritmo e selecione “Show Properties ...”.

Implementação do Exemplo 2



Altere **numClusters**
(o número de agrupamentos – k)
para **3**.

Clique em **ok**.

Volte a executar
(START).

Implementação do Exemplo 2

```
Clusterer output

--- Run information ---

Scheme:      weka.clusterers.SimpleKMeans -init 0 -max-candidates 100
Relation:     academia
Instances:    8
Attributes:   2
              idade
              frequencia_semana
Test mode:    evaluate on training data

--- Clustering model (full training set) ---


kMeans
=====

Number of iterations: 4
Within cluster sum of squared errors: 0.2095061728395062

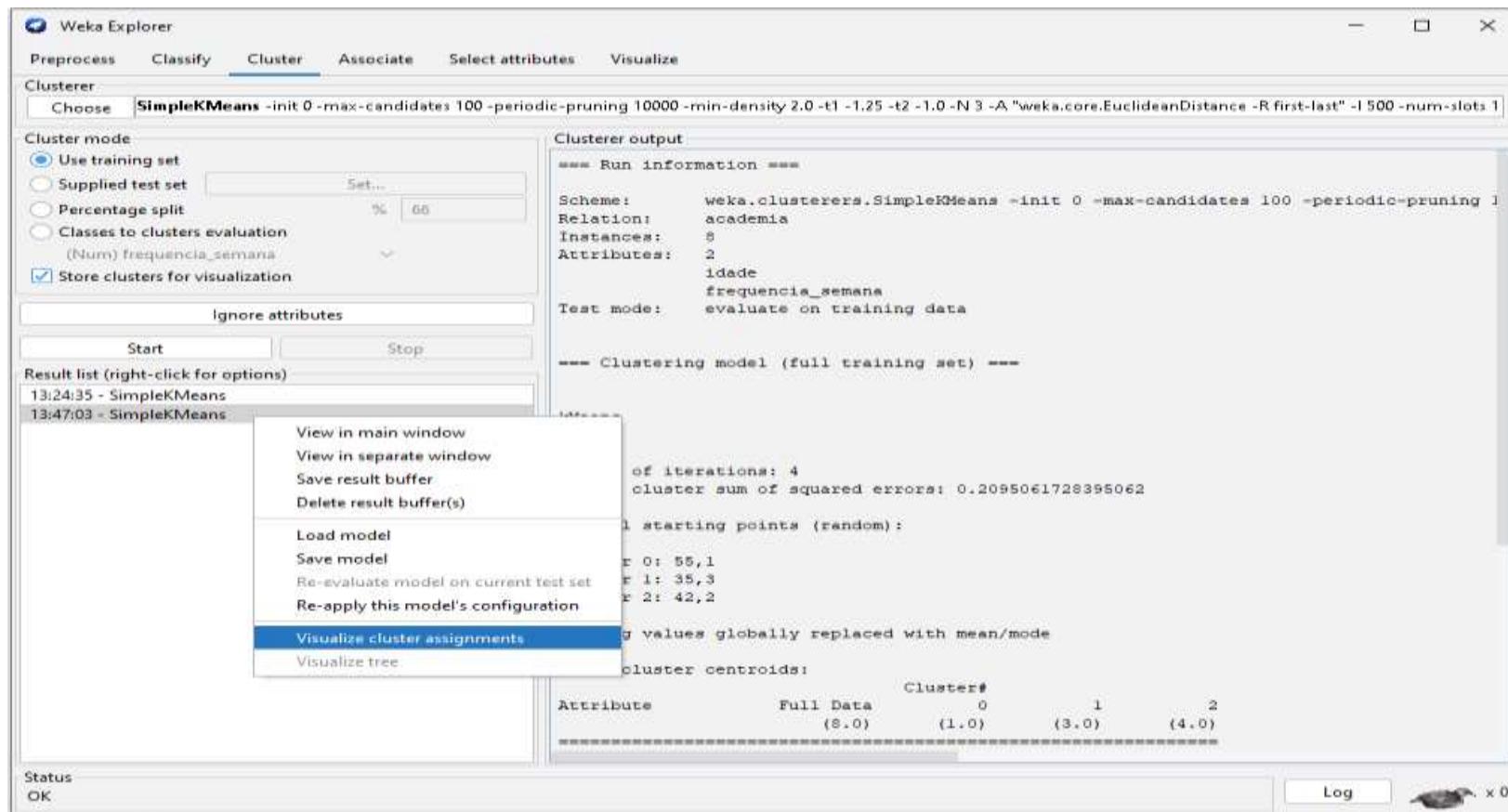
Initial starting points (random):

Cluster 0: 55,1
Cluster 1: 35,3
Cluster 2: 42,2
```

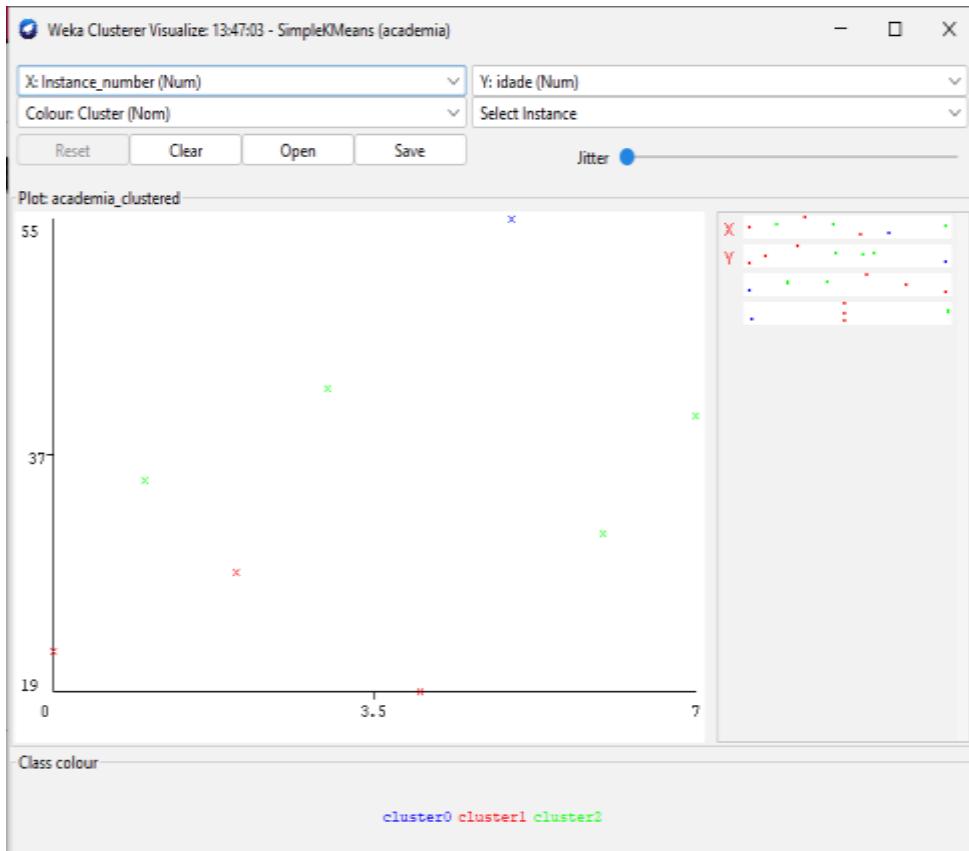
Agora as instâncias (clientes da academia) foram distribuídos entre 3 agrupamentos.

Clusters 0, 1 e 2, respectivamente com os centróides iniciais (55,1 – 35,3 – 42,2).

Implementação do Exemplo 2



Implementação do Exemplo 2



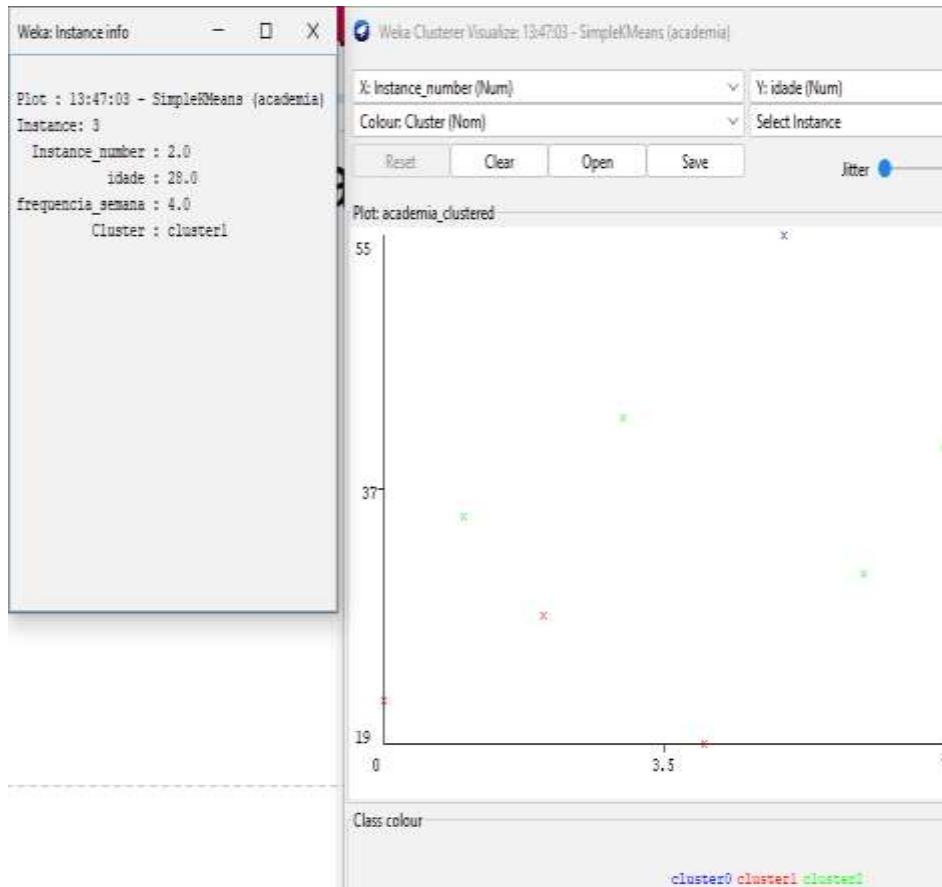
'X' são as instâncias.

'X' azul pertence ao cluster 0.

'X' vermelho pertence ao cluster 1.

'X' verde pertence ao cluster 2.

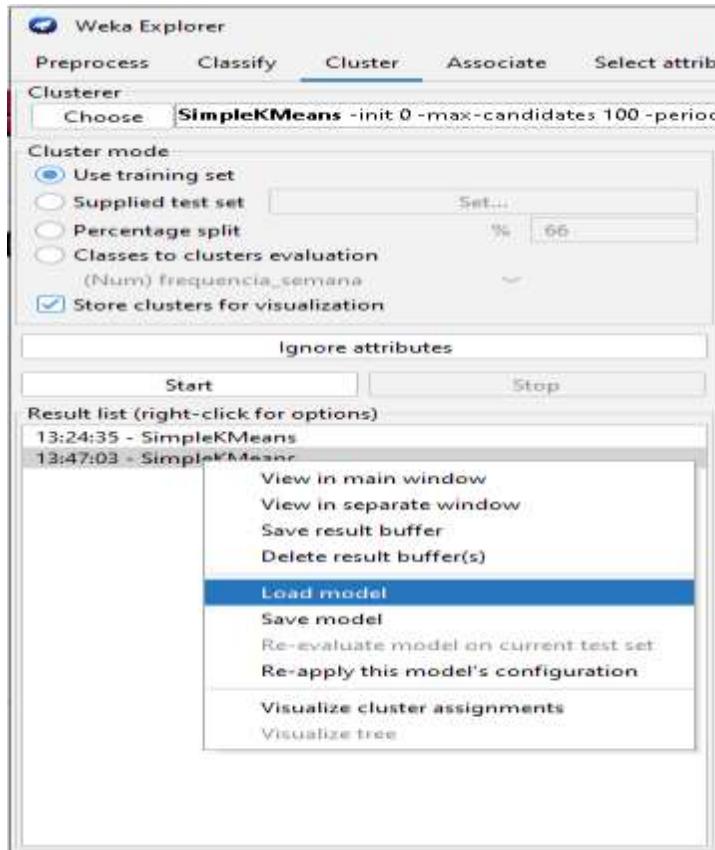
Implementação do Exemplo 2



Clique 2 vezes rapidamente sobre uma das instâncias e o weka lhe dará informações sobre a mesma.

Cliquei sobre a segunda instância vermelha da esquerda para a direita.

Implementação do Exemplo 2



Agente

- Um **Agente** pode ser definido como:
- Uma **entidade autônoma (software ou hardware)** que percebe o seu **ambiente** por meio de **sensores**, e age sobre **esse ambiente** através de **atuadores**, de forma a atingir **objetivos** ou maximizar uma função de **desempenho**, podendo **interagir** e **cooperar com outros agentes ou sistemas**.

Agente

De forma mais técnica, em inteligência artificial e sistemas distribuídos, um agente pode ser descrito como uma tupla:

$$Agente = \langle E, S, A, \pi \rangle$$

onde:

- **E**: conjunto de estados possíveis do ambiente
- **S**: conjunto de sensores (informações que o agente pode perceber)
- **A**: conjunto de ações disponíveis (atuadores)
- **π (pi)**: função de decisão ou política, que mapeia percepções em ações:

$$\pi : Percepções \rightarrow Ações$$

Características comuns de agentes

- **Autonomia** → atuam sem intervenção humana direta.
- **Reatividade** → respondem a mudanças no ambiente.
- **Proatividade** → tomam iniciativa para alcançar metas.
- **Sociabilidade** → podem interagir com outros agentes ou sistemas.

Agente

Exemplos de agentes:

1. Agentes de Software (IA):

Assistentes virtuais (como Siri, Alexa, ChatGPT): percebem comandos de voz/texto e executam ações (responder perguntas, ligar dispositivos, etc.).

Agentes de negociação em comércio eletrônico: monitoram preços e fazem ofertas automáticas.

Agente

Agentes de recomendação (Netflix, Spotify): percebem preferências do usuário e sugerem conteúdo.

2. Agentes Robóticos (IA + mundo físico):

Robô aspirador inteligente: sensores (câmeras, proximidade), atuadores (motores), política (limpar o ambiente de forma eficiente).

Agente

Carros autônomos: percebem o tráfego via câmeras/LiDAR e atuam controlando direção, freio e aceleração.

3. Agentes em Sistemas Distribuídos:

Agentes móveis: pequenos programas que “viajam” entre servidores coletando e processando informações.

Agente

Agentes de monitoramento de rede: verificam tráfego, detectam falhas e acionam correções automáticas.

Agentes de segurança cibernética: detectam intrusões e executam respostas (como isolar máquinas).

Tipos de Agentes

1. Agentes Simples (Reativos)

- **Definição:** funcionam baseados em regras **condição → ação**.
- **Funcionamento:** percebem o ambiente em um dado momento e reagem imediatamente, sem memória nem raciocínio complexo.
- **Modelo formal:**

$$f : \text{Percepção} \rightarrow \text{Ação}$$

- **Características:**
 - Alta velocidade de resposta.
 - Não consideram o histórico de percepções.
 - Não planejam nem antecipam consequências.
- **Exemplo:**
 - **Termostato:** se a temperatura < 20°C, ligar o aquecedor; caso contrário, desligar.
 - **Robô aspirador básico:** se bater em obstáculo, mudar de direção.

Tipos de Agentes

2. Agentes com Estado (Reativos Modelados)

- Definição: semelhantes aos simples, mas mantêm uma memória do estado **interno** para interpretar o ambiente.
- Funcionamento: usam percepções passadas para formar uma visão mais consistente do mundo.
- Exemplo:
 - Robô aspirador avançado: guarda o mapa do ambiente para evitar limpar o mesmo local várias vezes.

Tipos de Agentes

3. Agentes Cognitivos (Deliberativos)

- **Definição:** possuem uma representação simbólica ou lógica do ambiente e **planejam** ações para alcançar metas.
- **Funcionamento:**
 1. Constroem um **modelo do mundo**.
 2. Usam raciocínio/plano para decidir ações.
- **Modelo formal:**

$$\text{Planejador} : \text{Objetivos} \times \text{Conhecimento} \rightarrow \text{Plano}$$

- **Características:**
 - Mais "inteligentes", mas mais lentos.
 - Capazes de **prever** consequências de ações.
- **Exemplo:**
 - **Carro autônomo:** decide não só frear, mas também mudar de faixa para evitar acidentes.
 - **Agentes de jogos de xadrez** (como o Stockfish): planejam jogadas futuras, não apenas reagem à jogada do oponente.

Tipos de Agentes

4. Agentes de Aprendizado (Inteligentes Adaptativos)

- Definição: agentes que aprendem com experiência e melhoram sua política de ação ao longo do tempo.
- Base matemática:
 - Muitas vezes modelados por aprendizado por reforço:

$$\pi^* = \arg \max_{\pi} \mathbb{E}[Recompensa | \pi]$$

- Características:
 - Ajustam o comportamento conforme feedback.
 - Podem lidar com ambientes incertos e dinâmicos.
- Exemplo:
 - AlphaGo (Google DeepMind): aprendeu a jogar Go superando campeões humanos.
 - Assistentes virtuais modernos: aprendem preferências do usuário ao longo do tempo.

Tipos de Agentes

Tipo de Agente	Memória	Planejamento	Aprendizado	Exemplo
Simples (Reativo)	Não	Não	Não	Termostato
Com Estado	Sim	Não	Não	Aspirador com mapa
Cognitivo (Deliberativo)	Sim	Sim	Não	Carro autônomo
De Aprendizado	Sim	Sim	Sim	AlphaGo

Agente BDI

- Um **Agente BDI** é um agente **cognitivo deliberativo** cuja arquitetura é inspirada em **modelos de raciocínio humano**, baseada em três componentes fundamentais:
- **1. Beliefs (Crenças)**
 - Representam o **conhecimento** que o agente tem sobre o mundo, que pode ser incompleto ou até incorreto.
 - Inclui informações recebidas de sensores, comunicação com outros agentes ou memória interna.
 - Ex.: “Está chovendo”, “Tenho 20 litros de combustível”, “O cliente pediu uma entrega”.

Agente BDI

- **2. Desires (Desejos):**
 - Representam os **objetivos** ou estados de coisas que o agente gostaria de alcançar.
 - Nem todos os desejos podem ser satisfeitos ao mesmo tempo (podem ser conflitantes).
 - Ex.: “Chegar ao destino”, “Maximizar o lucro”, “Atender todos os pedidos do cliente”.

Agente BDI

- **3. Intentions (Intenções):**
- São os **desejos que o agente escolheu perseguir ativamente**, ou seja, os objetivos **comprometidos**.
- Determinam os **planos de ação** que o agente efetivamente executará.
- Ex.: “Seguir pela rota A para entregar a encomenda”, “Reducir o preço do produto em 10%”.

Agente BDI

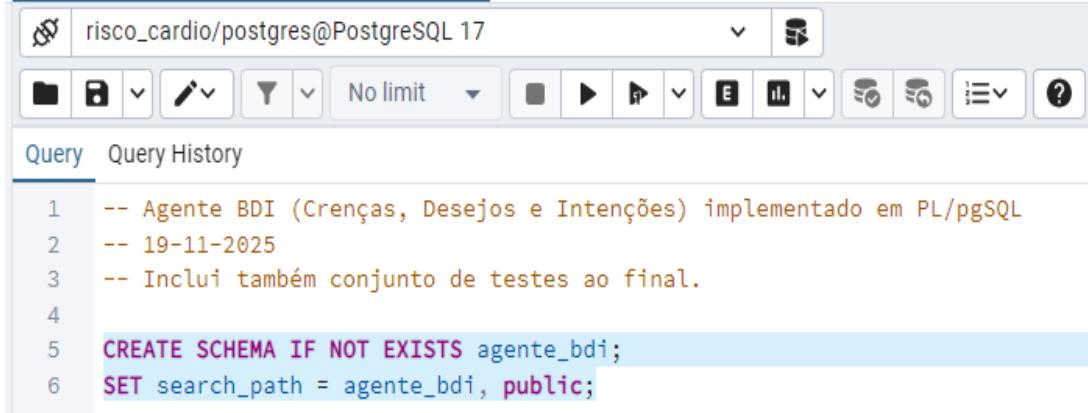
Funcionamento de um Agente BDI:

Um **ciclo de raciocínio BDI** pode ser descrito assim:

1. **Percepção** → o agente observa o ambiente (sensores, mensagens).
2. **Atualização de Crenças** → o agente ajusta suas *Beliefs*.
3. **Geração de Desejos** → o agente cria ou revisa seus *Desires*.
4. **Filtragem de Desejos em Intenções** → o agente seleciona um subconjunto viável de *Desires*, transformando-os em *Intentions*.
5. **Execução de Planos** → com base nas *Intentions*, o agente escolhe **planos** de ações e os executa.
6. **Laço contínuo** → o ciclo se repete, ajustando crenças, desejos e intenções de acordo com mudanças no ambiente.

Proposta de um Agente BDI

- Proporemos a implementação de um **agente simplificado** de avaliação do **risco cardiológico**.
- Seus **sensores** percebem a **pressão arterial (sistólica e diastólica)**, a **altura** e o **peso** dos pacientes.



```
-- Agente BDI (Crenças, Desejos e Intenções) implementado em PL/pgSQL
-- 19-11-2025
-- Inclui também conjunto de testes ao final.

CREATE SCHEMA IF NOT EXISTS agente_bdi;
SET search_path = agente_bdi, public;
```

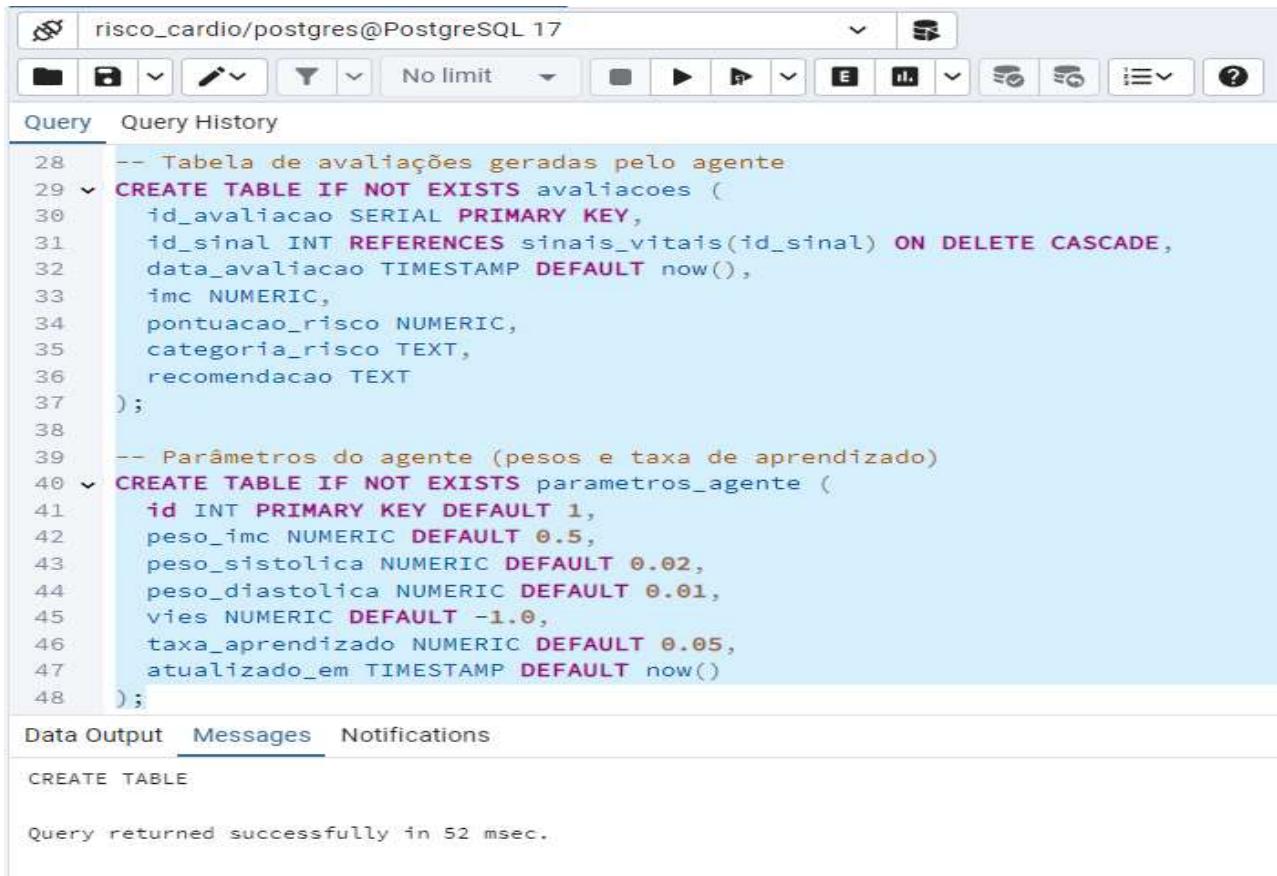
Proposta de um Agente BDI

The screenshot shows a PostgreSQL client window with the following details:

- Connection:** risco_cardio/postgres@PostgreSQL 17
- Toolbar:** Includes icons for file, copy, paste, search, and various database operations.
- Query History:** Shows the history of queries run.
- Text Area:** Displays the SQL code for creating two tables:

```
9
10    -- Tabela de pacientes
11    CREATE TABLE IF NOT EXISTS pacientes (
12        id_paciente SERIAL PRIMARY KEY,
13        nome TEXT,
14        data_nascimento DATE
15    );
16
17    -- Tabela de sinais vitais
18    CREATE TABLE IF NOT EXISTS sinais_vitais (
19        id_sinal SERIAL PRIMARY KEY,
20        id_paciente INT REFERENCES pacientes(id_paciente) ON DELETE CASCADE,
21        data_medicao TIMESTAMP DEFAULT now(),
22        peso_kg NUMERIC NOT NULL,
23        altura_m NUMERIC NOT NULL,
24        pressao_sistolica INT NOT NULL,
25        pressao_diastolica INT NOT NULL
26    );
27
```
- Data Output:** Shows the result of the CREATE TABLE command: "CREATE TABLE".
- Messages:** Shows the message: "Query returned successfully in 94 msec."

Proposta de um Agente BDI



The screenshot shows a PostgreSQL client interface with the following details:

- Connection:** risco_cardio/postgres@PostgreSQL 17
- Toolbar:** Includes icons for file operations, search, and various database management functions.
- Tab Bar:** Shows "Query" and "Query History".
- Code Area:** Displays the following SQL code:

```
28 -- Tabela de avaliações geradas pelo agente
29 CREATE TABLE IF NOT EXISTS avaliacoes (
30     id_avaliacao SERIAL PRIMARY KEY,
31     id_sinal INT REFERENCES sinais_vitais(id_sinal) ON DELETE CASCADE,
32     data_avaliacao TIMESTAMP DEFAULT now(),
33     imc NUMERIC,
34     pontuacao_risco NUMERIC,
35     categoria_risco TEXT,
36     recomendacao TEXT
37 );
38
39 -- Parâmetros do agente (pesos e taxa de aprendizado)
40 CREATE TABLE IF NOT EXISTS parametros_agente (
41     id INT PRIMARY KEY DEFAULT 1,
42     peso_imc NUMERIC DEFAULT 0.5,
43     peso_sistolica NUMERIC DEFAULT 0.02,
44     peso_diastolica NUMERIC DEFAULT 0.01,
45     viés NUMERIC DEFAULT -1.0,
46     taxa_aprendizado NUMERIC DEFAULT 0.05,
47     atualizado_em TIMESTAMP DEFAULT now()
48 );
```

Data Output: CREATE TABLE

Messages: Query returned successfully in 52 msec.

Proposta de um Agente BDI

The screenshot shows a pgAdmin 4 interface with a query editor window. The title bar says 'risco_cardio/postgres@PostgreSQL 17'. The toolbar has various icons for database management. The query history tab is selected. The main area contains the following SQL code:

```
38
39 -- Parâmetros do agente (pesos e taxa de aprendizado)
40 CREATE TABLE IF NOT EXISTS parametros_agente (
41     id INT PRIMARY KEY DEFAULT 1,
42     peso_imc NUMERIC DEFAULT 0.5,
43     peso_sistolica NUMERIC DEFAULT 0.02,
44     peso_diastolica NUMERIC DEFAULT 0.01,
45     vies NUMERIC DEFAULT -1.0,
46     taxa_aprendizado NUMERIC DEFAULT 0.05,
47     atualizado_em TIMESTAMP DEFAULT now()
48 );
49 INSERT INTO parametros_agente(id) SELECT 1 WHERE NOT EXISTS (SELECT 1 FROM parametros_agente);
50
```

The status bar at the bottom shows 'Data Output' is active, with 'Messages' and 'Notifications' also listed. The output pane shows the result of the last query: 'INSERT 0 1'. Below that, it says 'Query returned successfully in 49 msec.'

padrão de inicialização segura: garante que uma tabela tenha exatamente um registro padrão, sem duplicações.

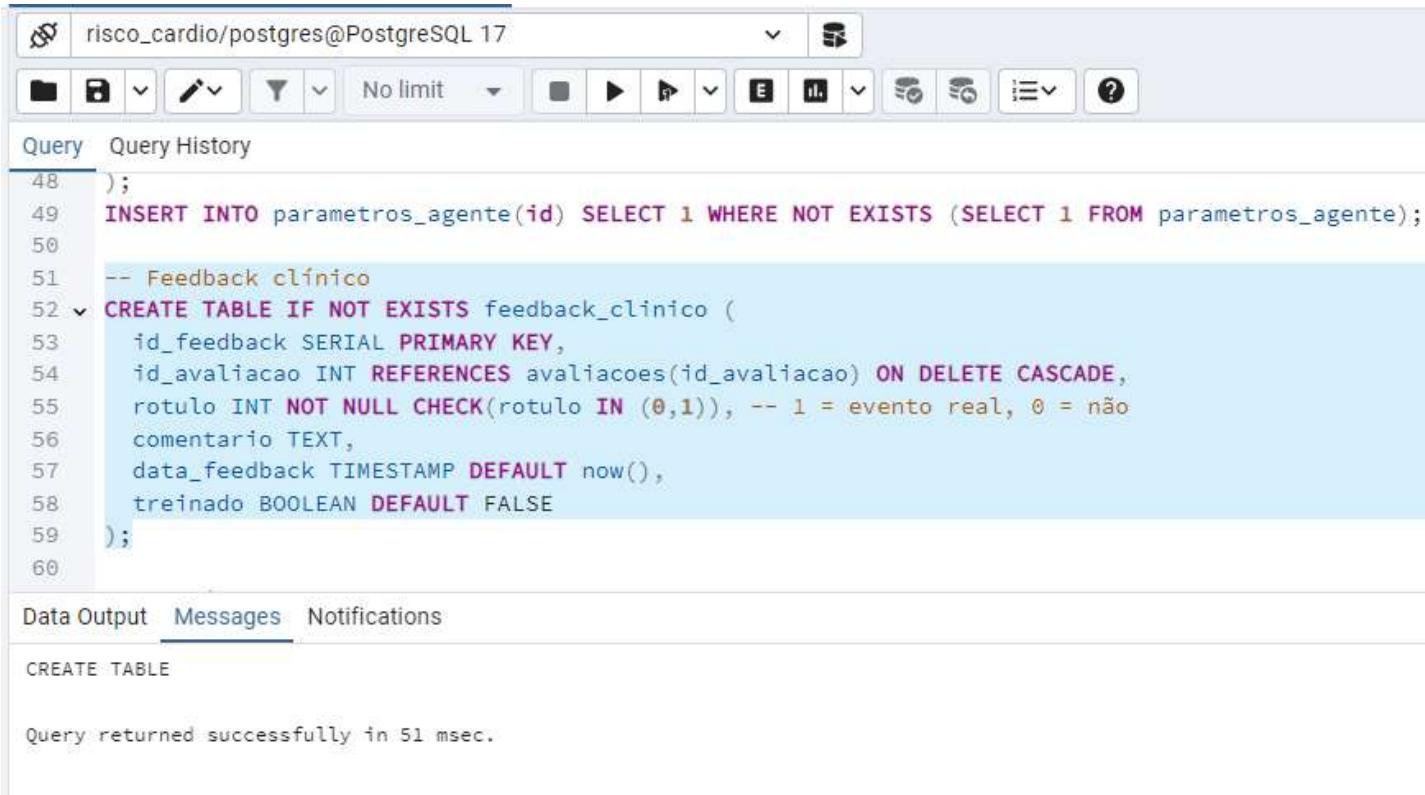
Verifica se a tabela parametros_agente está vazia.

Só insere o valor se a tabela estiver vazia

Se a tabela estiver vazia → o SELECT retorna o valor 1 → esse valor é inserido na coluna id

Se já existir algum registro → SELECT não retorna nada →

Proposta de um Agente BDI



The screenshot shows a PostgreSQL query editor interface from pgAdmin 4. The title bar indicates the connection is to 'risco_cardio/postgres@PostgreSQL 17'. The toolbar includes various icons for database management tasks. The main area is divided into tabs: 'Query' (which is selected), 'Query History', 'Data Output', 'Messages', and 'Notifications'. The 'Query' tab contains the following PostgreSQL code:

```
48 );
49 INSERT INTO parametros_agente(id) SELECT 1 WHERE NOT EXISTS (SELECT 1 FROM parametros_agente);
50
51 -- Feedback clínico
52 CREATE TABLE IF NOT EXISTS feedback_clinico (
53     id_feedback SERIAL PRIMARY KEY,
54     id_avaliacao INT REFERENCES avaliacoes(id_avaliacao) ON DELETE CASCADE,
55     rotulo INT NOT NULL CHECK(rotulo IN (0,1)), -- 1 = evento real, 0 = não
56     comentario TEXT,
57     data_feedback TIMESTAMP DEFAULT now(),
58     treinado BOOLEAN DEFAULT FALSE
59 );
60
```

The 'Data Output' tab shows the result of the query:

```
CREATE TABLE
```

The message 'Query returned successfully in 51 msec.' is displayed in the 'Messages' tab.

Proposta de um Agente BDI



The screenshot shows a pgAdmin 4 interface for PostgreSQL 17. The connection is risco_cardio/postgres@PostgreSQL 17. The toolbar includes standard database management icons. Below the toolbar, there are tabs for 'Query' (selected) and 'Query History'. The main area displays a SQL script:

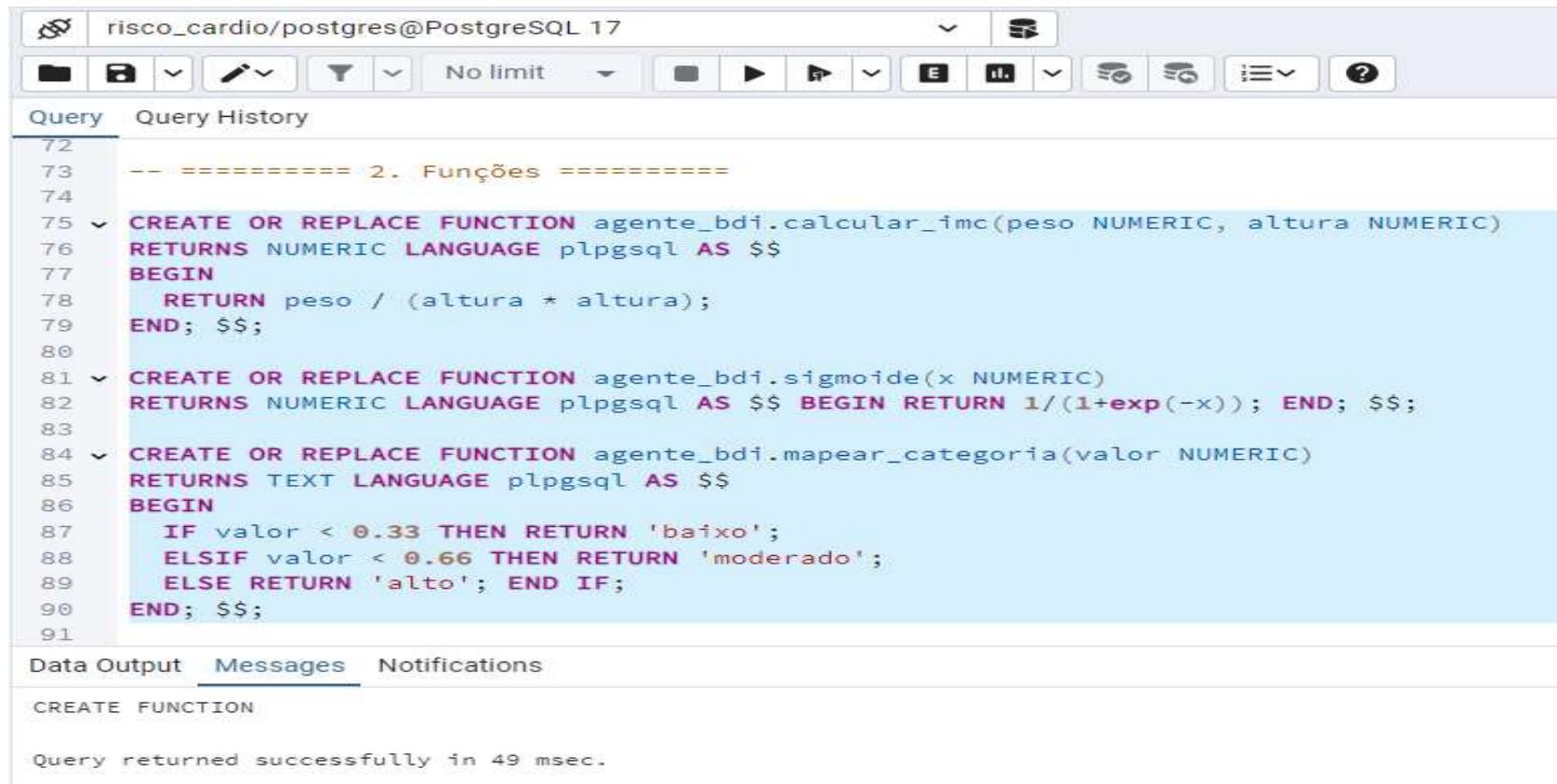
```
60
61  -- Histórico de ajustes
62  CREATE TABLE IF NOT EXISTS historico_parametros (
63      id_historico SERIAL PRIMARY KEY,
64      data_mudanca TIMESTAMP DEFAULT now(),
65      peso_imc NUMERIC,
66      peso_sistolica NUMERIC,
67      peso_diastolica NUMERIC,
68      vies NUMERIC,
69      taxa_aprendizado NUMERIC,
70      motivo TEXT
71  );
72
```

Below the code, the 'Messages' tab is selected, showing the output of the query:

CREATE TABLE

Query returned successfully in 59 msec.

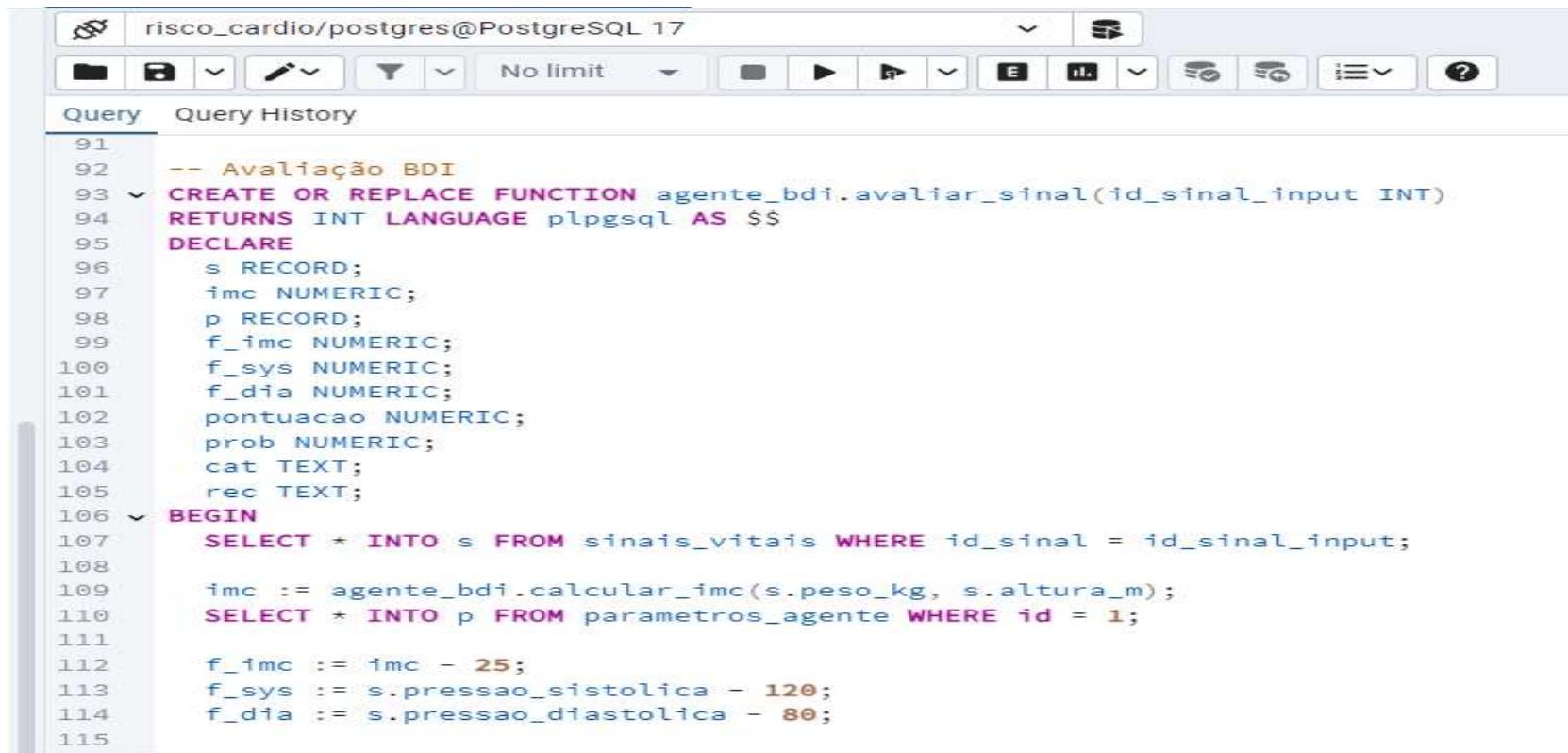
Proposta de um Agente BDI



The screenshot shows a PostgreSQL client window with the following details:

- Connection:** risco_cardio/postgres@PostgreSQL 17
- Toolbar:** Includes icons for file operations, search, and various database management functions.
- Query History:** Shows the history of queries run in the session.
- Query Editor:** Displays the code for three functions:
 - 72 - 80:** `CREATE OR REPLACE FUNCTION agente_bdi.calcular_imc(peso NUMERIC, altura NUMERIC)`
 - 81 - 83:** `CREATE OR REPLACE FUNCTION agente_bdi.sigmoide(x NUMERIC)`
 - 84 - 91:** `CREATE OR REPLACE FUNCTION agente_bdi.mapear_categoria(valor NUMERIC)`
- Data Output:** Shows the result of the last query: `CREATE FUNCTION`.
- Messages:** Shows the message: `Query returned successfully in 49 msec.`

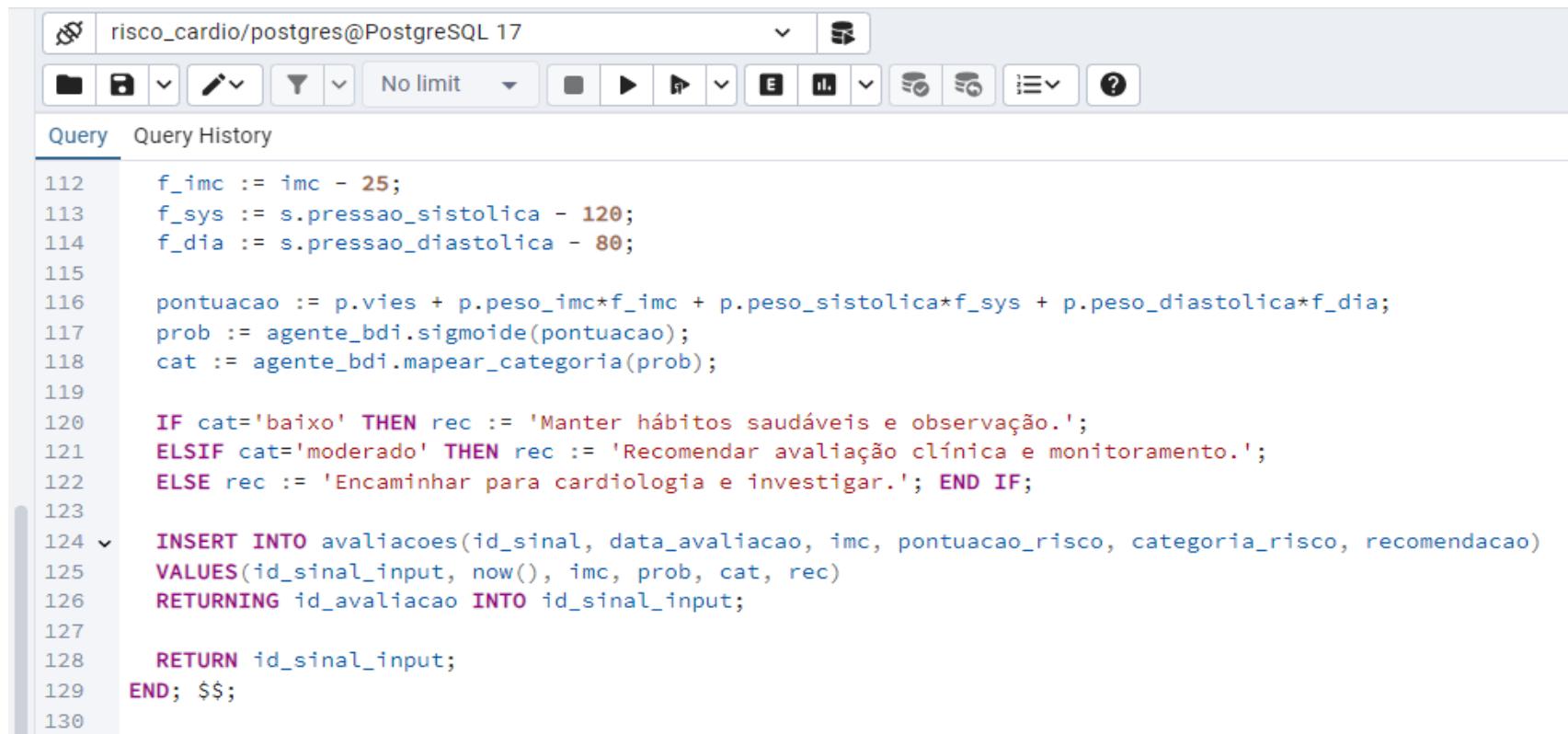
Proposta de um Agente BDI



The screenshot shows a PostgreSQL client interface with a query editor. The title bar reads "risco_cardio/postgres@PostgreSQL 17". The toolbar includes various icons for file operations, search, and navigation. The main area is a "Query" tab showing a script for creating a function named "agente_bdi.avaliar_sinal". The script uses the plpgsql language and declares several variables (s, imc, p, f_imc, f_sys, f_dia, pontuacao, prob, cat, rec) of type RECORD, NUMERIC, or TEXT. It begins by selecting a record 's' from the 'sinais_vitais' table where the input signal ID matches the parameter. It then calculates the IMC using a function 'calcular_imc' and selects parameters for the agent ('p') from the 'parametros_agente' table. Finally, it performs some assignments: setting 'f_imc' to 'imc - 25', and calculating 'f_sys' and 'f_dia' based on systolic and diastolic blood pressure values.

```
91
92 -- Avaliação BDI
93 ✓ CREATE OR REPLACE FUNCTION agente_bdi.avaliar_sinal(id_sinal_input INT)
94 RETURNS INT LANGUAGE plpgsql AS $$ 
95 DECLARE
96     s RECORD;
97     imc NUMERIC;
98     p RECORD;
99     f_imc NUMERIC;
100    f_sys NUMERIC;
101    f_dia NUMERIC;
102    pontuacao NUMERIC;
103    prob NUMERIC;
104    cat TEXT;
105    rec TEXT;
106 ✓ BEGIN
107     SELECT * INTO s FROM sinais_vitais WHERE id_sinal = id_sinal_input;
108
109     imc := agente_bdi.calcular_imc(s.peso_kg, s.altura_m);
110     SELECT * INTO p FROM parametros_agente WHERE id = 1;
111
112     f_imc := imc - 25;
113     f_sys := s.pressao_sistolica - 120;
114     f_dia := s.pressao_diastolica - 80;
115
```

Proposta de um Agente BDI



The screenshot shows a PostgreSQL client interface with the following details:

- Connection:** risco_cardio/postgres@PostgreSQL 17
- Toolbar:** Includes icons for connection, refresh, save, filter, and various database operations.
- Query History:** Tab labeled "Query" is selected, showing a history of queries.
- Code Content:** The main area contains the following PL/pgSQL code:

```
112     f_imc := imc - 25;
113     f_sys := s.pressao_sistolica - 120;
114     f_dia := s.pressao_diastolica - 80;
115
116     pontuacao := p.vies + p.peso_imc*f_imc + p.peso_sistolica*f_sys + p.peso_diastolica*f_dia;
117     prob := agente_bdi.sigmoide(pontuacao);
118     cat := agente_bdi.mapear_categoria(prob);
119
120     IF cat='baixo' THEN rec := 'Manter hábitos saudáveis e observação.';
121     ELSIF cat='moderado' THEN rec := 'Recomendar avaliação clínica e monitoramento.';
122     ELSE rec := 'Encaminhar para cardiologia e investigar.'; END IF;
123
124     INSERT INTO avaliacoes(id_sinal, data_avaliacao, imc, pontuacao_risco, categoria_risco, recomendacao)
125     VALUES(id_sinal_input, now(), imc, prob, cat, rec)
126     RETURNING id_avaliacao INTO id_sinal_input;
127
128     RETURN id_sinal_input;
129   END; $$;
```

Proposta de um Agente BDI

```
risco_cardio/postgres@PostgreSQL 17
Query History
Query 131
131 -- Trigger
132 ✓ CREATE OR REPLACE FUNCTION agente_bdi.trigger_sinais_insert()
133 RETURNS TRIGGER LANGUAGE plpgsql AS $$ 
134 BEGIN
135     PERFORM agente_bdi.avaliar_sinal(NEW.id_sinal);
136     RETURN NEW;
137 END; $$;
138
139 DROP TRIGGER IF EXISTS trg_sinais_insert ON sinais_vitais;
140 ✓ CREATE TRIGGER trg_sinais_insert AFTER INSERT ON sinais_vitais
141 FOR EACH ROW EXECUTE FUNCTION agente_bdi.trigger_sinais_insert();
142
```

Esse trigger serve para acionar automaticamente o agente BDI sempre que um novo sinal vital é inserido na tabela **sinais_vitais**.

Ele assegura a **autonomia** do agente. O agente fica em constante **monitoramento**.

Reparou no comando **PERFORM**?

Proposta de um Agente BDI

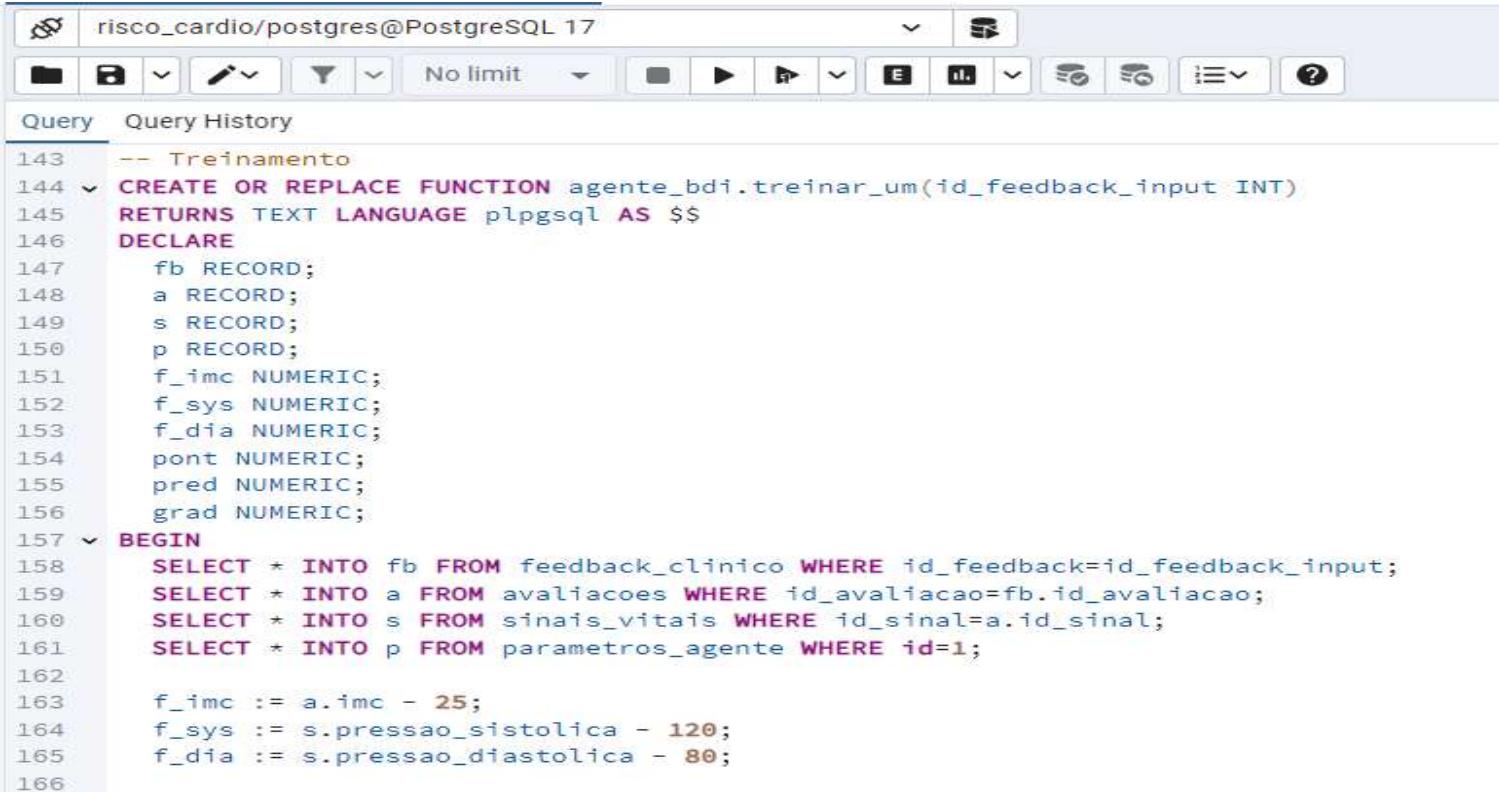
```
134 BEGIN  
135   PERFORM agente_bdi.avaliar_sinal(NEW.id_sinal);
```

No PostgreSQL (PL/pgSQL), o comando **PERFORM** é usado para executar uma função ou consulta cujo resultado você não quer capturar — ou seja, quando você não precisa do retorno, apenas deseja que ela seja executada.

Use **PERFORM** quando você quer:

- ✓ Executar uma função que tem efeitos colaterais
(ex.: grava log, dispara alerta, atualiza tabelas, insere dados).
- ✓ Chamar uma função que retorna algo, mas cujo retorno não

Proposta de um Agente BDI



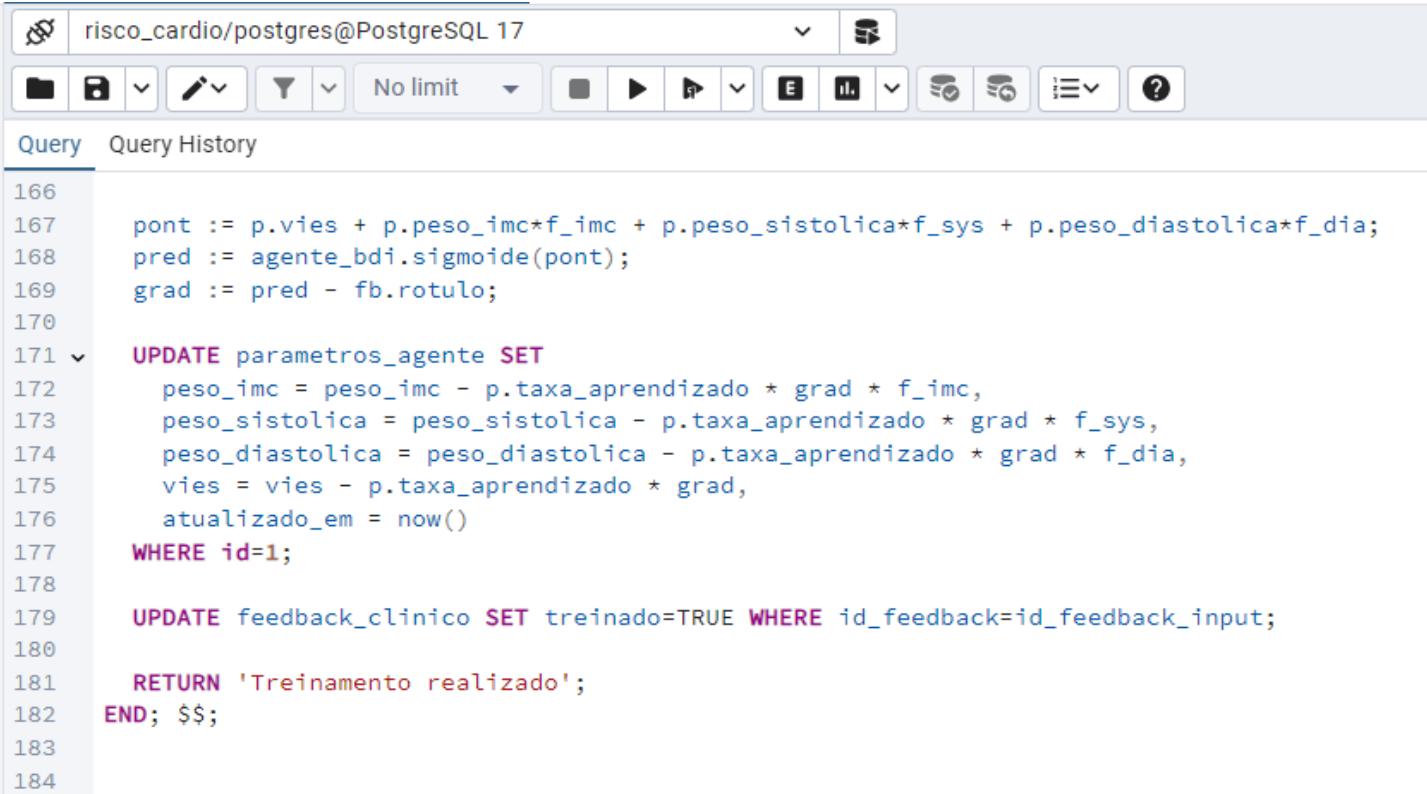
The screenshot shows a pgAdmin 4 interface with a query editor window. The title bar indicates the connection is to 'risco_cardio/postgres@PostgreSQL 17'. The toolbar has various icons for database management. The left sidebar shows 'Query History' and 'Query' tabs, with 'Query' selected. The main area contains a multi-line SQL script:

```
-- Treinamento
CREATE OR REPLACE FUNCTION agente_bdi.treinar_um(id_feedback_input INT)
RETURNS TEXT LANGUAGE plpgsql AS $$
DECLARE
    fb RECORD;
    a RECORD;
    s RECORD;
    p RECORD;
    f_imc NUMERIC;
    f_sys NUMERIC;
    f_dia NUMERIC;
    pont NUMERIC;
    pred NUMERIC;
    grad NUMERIC;
BEGIN
    SELECT * INTO fb FROM feedback_clinico WHERE id_feedback=id_feedback_input;
    SELECT * INTO a FROM avaliacoes WHERE id_avaliacao=fb.id_avaliacao;
    SELECT * INTO s FROM sinais_vitais WHERE id_sinal=a.id_sinal;
    SELECT * INTO p FROM parametros_agente WHERE id=1;

    f_imc := a.imc - 25;
    f_sys := s.pressao_sistolica - 120;
    f_dia := s.pressao_diastolica - 80;

```

Proposta de um Agente BDI



The screenshot shows a PostgreSQL client interface with the following details:

- Connection:** risco_cardio/postgres@PostgreSQL 17
- Toolbar:** Includes icons for file, copy, paste, search, and various database operations.
- Menu Bar:** Shows "No limit" and other standard menu items.
- Query Tab:** Active tab, showing the query history.
- Code Area:** Displays the following PostgreSQL code:

```
166
167     pont := p.vies + p.peso_imc*f_imc + p.peso_sistolica*f_sys + p.peso_diastolica*f_dia;
168     pred := agente_bdi.sigmoide(pont);
169     grad := pred - fb.rotulo;
170
171     UPDATE parametros_agente SET
172         peso_imc = peso_imc - p.taxa_aprendizado * grad * f_imc,
173         peso_sistolica = peso_sistolica - p.taxa_aprendizado * grad * f_sys,
174         peso_diastolica = peso_diastolica - p.taxa_aprendizado * grad * f_dia,
175         vies = vies - p.taxa_aprendizado * grad,
176         atualizado_em = now()
177     WHERE id=1;
178
179     UPDATE feedback_clinico SET treinado=TRUE WHERE id_feedback=id_feedback_input;
180
181     RETURN 'Treinamento realizado';
182 END; $$;
```

Proposta de um Agente BDI



The screenshot shows a pgAdmin III interface with a query editor window. The title bar indicates the connection is to 'risco_cardio/postgres@PostgreSQL 17'. The toolbar has various icons for database management. The main area is a 'Query' tab with the following SQL script:

```
184
185 -- ===== 3. Conjunto de testes sintéticos =====
186 -- Pacientes
187 ✓ INSERT INTO pacientes(nome, data_nascimento) VALUES
188     ('João Teste', '1980-01-01'),
189     ('Maria Teste', '1975-05-10');
190
191 -- Sinais vitais de treinamento (antes do feedback)
192 ✓ INSERT INTO sinais_vitais(id_paciente, peso_kg, altura_m, pressao_sistolica, pressao_diastolica) VALUES
193     (1, 95, 1.70, 160, 100), -- risco alto
194     (1, 82, 1.70, 130, 85), -- moderado
195     (2, 60, 1.60, 115, 75), -- baixo
196     (2, 70, 1.60, 145, 95); -- alto
197
198 -- Suponha que o médico informe que os casos 1 e 4 tiveram eventos (1), os demais não (0)
199 ✓ INSERT INTO feedback_clinico(id_avaliacao, rotulo, comentario)
200 VALUES
201     (1,1,'Evento real ocorreu'),
202     (2,0,'Sem evento'),
203     (3,0,'Sem evento'),
204     (4,1,'Evento real ocorreu');
205
```

Proposta de um Agente BDI

The screenshot shows a PostgreSQL database interface. The SQL query entered is:

```
205  
206   SELECT * FROM parametros_agente;  
207
```

The results are displayed in a table:

	id [PK] integer	peso_imc numeric	peso_sistolica numeric	peso_diastolica numeric	vies numeric	taxa_aprendizado numeric	atualizado_em timestamp without time zone
1	1	0.5	0.02	0.01	-1.0	0.05	2025-11-19 14:25:02.542217

O aprendizado do agente pode ser atestado pelas mudanças nos pesos associados a cada variável de entrada (imc, sistolica e diastolica).

Portanto, precisamos ver como os pesos mudam após o treinamento.

Na figura acima, temos o valor dos pesos antes do treinamento.

O índice de massa corporal (imc) corresponde ao resultado do peso do paciente dividido pelo quadrado da altura do mesmo paciente.

O agente recebe os valores do peso (95 kg), da altura (1,70 m), da pressão arterial sistólica (160) e da pressão arterial diastólica (100). Portanto, sua pressão arterial é de 16 por 10 (valor elevado) e seu imc = 32.87197.

Proposta de um Agente BDI

```
207  
208 -- Treinar em lote  
209 SELECT agente_bdi.treinar_um(id_feedback) FROM feedback_clinico;  
210
```

Data Output Messages Notifications

treinar_um

	treinar_um
1	Treinamento realizado
2	Treinamento realizado
3	Treinamento realizado
4	Treinamento realizado
5	Treinamento realizado
6	Treinamento realizado
7	Treinamento realizado
8	Treinamento realizado

O **treinamento** **do**
Agente **foi**
realizado.

Proposta de um Agente BDI

```
205  
206 SELECT * FROM parametros_agente;  
207
```

Data Output Messages Notifications

SQL

	id [PK] integer	peso_imc numeric	peso_sistolica numeric	peso_diastolica numeric	vies numeric	taxa_aprendizado numeric	atualizado_em timestamp without time zone
1	1	0.5	0.02	0.01	-1.0	0.05	2025-11-19 14:25:02.542217

```
205  
206 SELECT * FROM parametros_agente;  
207
```

Data Output Messages Notifications

SQL

	id [PK] integer	peso_imc numeric	peso_sistolica numeric	peso_diastolica numeric	vies numeric	taxa_aprendizado numeric	atualizado_em timestamp without time zone
1	1	0.374441313	0.564488521260	0.49603257779926	-1.07669700	0.05	2025-11-19 15:19:45.029426

Aqui podemos visualizar o aprendizado do agente:

O **peso_imc**, por exemplo, saiu de **0.5** para **0.374441313** (valor aproximado).

Proposta de um Agente BDI

Uma nova ocorrência é inserida após a fase de treinamento estar encerrada.

```
211
212 -- Em produção:
213 INSERT INTO sinais_vitais(id_paciente, peso_kg, altura_m, pressao_sistolica, pressao_diastolica) VALUES
214 (4, 109, 1.77, 140, 90);
215
```

Data Output [Messages](#) Notifications

INSERT 0 1

Query returned successfully in 65 msec.

Proposta de um Agente BDI

```
207
208  -- Treinar em lote
209  SELECT agente_bdi.treinar_um(id_feedback) FROM feedback_clinico;
210
211
212  -- Em produção:
213  INSERT INTO sinais_vitais(id_paciente, peso_kg, altura_m, pressao_sistolica, pressao_diastolica) VALUES
214  (4, 109, 1.77, 140, 90);
215
216  SELECT * FROM SINAIS_VITAIS;
```

Data Output Messages Notifications

SQL

	id_sinal [PK] integer	id_paciente integer	data_medicao timestamp without time zone	peso_kg numeric	altura_m numeric	pressao_sistolica integer	pressao_diastolica integer
1		1	2025-11-19 15:01:55.545313	95	1.70	160	100
2		2	2025-11-19 15:01:55.545313	82	1.70	130	85
3		3	2025-11-19 15:01:55.545313	60	1.60	115	75
4		4	2025-11-19 15:01:55.545313	70	1.60	145	95
5		5	2025-11-19 15:03:03.292763	95	1.70	160	100
6		6	2025-11-19 15:03:03.292763	82	1.70	130	85
7		7	2025-11-19 15:03:03.292763	60	1.60	115	75
8		8	2025-11-19 15:03:03.292763	70	1.60	145	95
9		10	2025-11-19 15:31:52.480205	109	1.77	140	90

Proposta de um Agente BDI

- A trigger de nosso agente executa aquele **PERFORM** que destacamos anteriormente.
- Esse **PERFORM** executa
agente_bdi.avaliar_sinal().

```
130
131  -- Trigger
132 ✓ CREATE OR REPLACE FUNCTION agente_bdi.trigger_sinais_insert()
133   RETURNS TRIGGER LANGUAGE plpgsql AS $$ 
134 BEGIN
135   PERFORM agente_bdi.avaliar_sinal(NEW.id_sinal);
136   RETURN NEW;
137 END; $$;
138
139 DROP TRIGGER IF EXISTS trg_sinais_insert ON sinais_vitais;
140 ✓ CREATE TRIGGER trg_sinais_insert AFTER INSERT ON sinais_vitais
141   FOR EACH ROW EXECUTE FUNCTION agente_bdi.trigger_sinais_insert();
142
```

Proposta de um Agente BDI

Então, ao inserir o sinal, o agente:

- calcula IMC
- calcula risco
- classifica
- grava na tabela avaliacoes

Para ver a avaliação gerada:

Proposta de um Agente BDI

```
-- Treinar em lote
209 SELECT agente_bdi.treinar_um(id_feedback) FROM feedback_clinico;
210
211
212 -- Em produção:
213 INSERT INTO sinais_vitais(id_paciente, peso_kg, altura_m, pressao_sistolica, pressao_diastolica) VALUES
214 (4, 109, 1.77, 140, 90);
215
216 SELECT * FROM SINAIS_VITAIS;
217
218
219 select * from avaliacoes;
220
```

Data Output Messages Notifications

Showing rows: 1 to 9 

	id_avaliacao [PK]	id_sinal	data_avaliacao	imc	pontuacao_risco	categoria_risco	recomendacao
1	1	1	2025-11-19 15:01:55.545313	32.8719723183391003	0.98084754547474646401	alto	Encaminhar para cardiologia e investigar.
2	2	2	2025-11-19 15:01:55.545313	28.3737024221453287	0.71846317814578268241	alto	Encaminhar para cardiologia e investigar.
3	3	3	2025-11-19 15:01:55.545313	23.4375000000000000	0.12661228857349682788	baixo	Manter hábitos saudáveis e observação.
4	4	4	2025-11-19 15:01:55.545313	27.3437500000000000	0.69463420554944507886	alto	Encaminhar para cardiologia e investigar.
5	5	5	2025-11-19 15:03:03.292763	32.8719723183391003	0.98084754547474646401	alto	Encaminhar para cardiologia e investigar.
6	6	6	2025-11-19 15:03:03.292763	28.3737024221453287	0.71846317814578268241	alto	Encaminhar para cardiologia e investigar.
7	7	7	2025-11-19 15:03:03.292763	23.4375000000000000	0.12661228857349682788	baixo	Manter hábitos saudáveis e observação.
8	8	8	2025-11-19 15:03:03.292763	27.3437500000000000	0.69463420554944507886	alto	Encaminhar para cardiologia e investigar.
9	9	10	2025-11-19 15:31:52.480205	34.7920457084490408	0.99999993424700157922	alto	Encaminhar para cardiologia e investigar.

Proposta de um Agente BDI

- Basta observar a última linha (com id_avaliacao = 9 ou id_sinal = 10).
 - O risco desse paciente é alto.