

# Técnicas de Otimização

**Geomar André Schreiner**

# Quem sou?

---

- Formação acadêmica
  - BDs NoSQL e NewSQL
- Professor UFFS
- Consultor Banco de Dados (Dante)



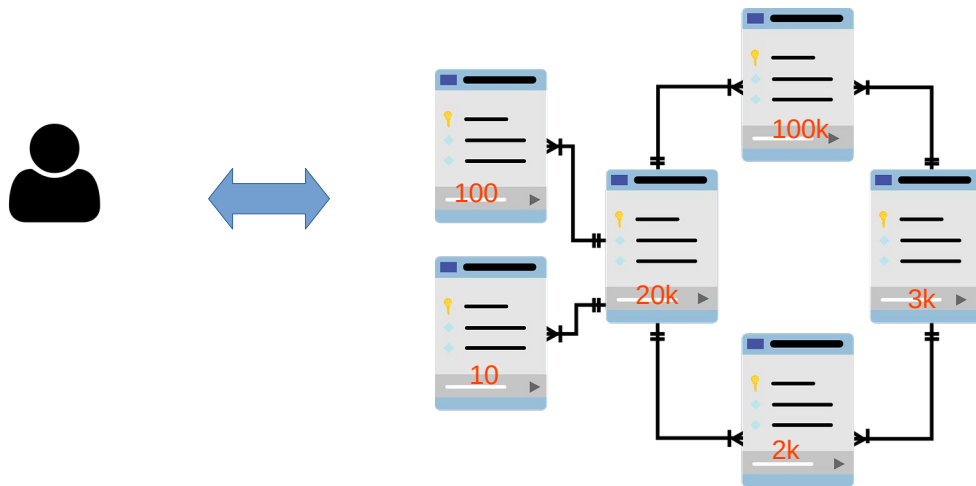
# Objetivo

---

Conversar sobre técnicas gerais (básicas) de otimização em BD (Postgres). Por onde começar, o que ver primeiro e quais os pontos que valem a pena atacar ou ter cuidado.

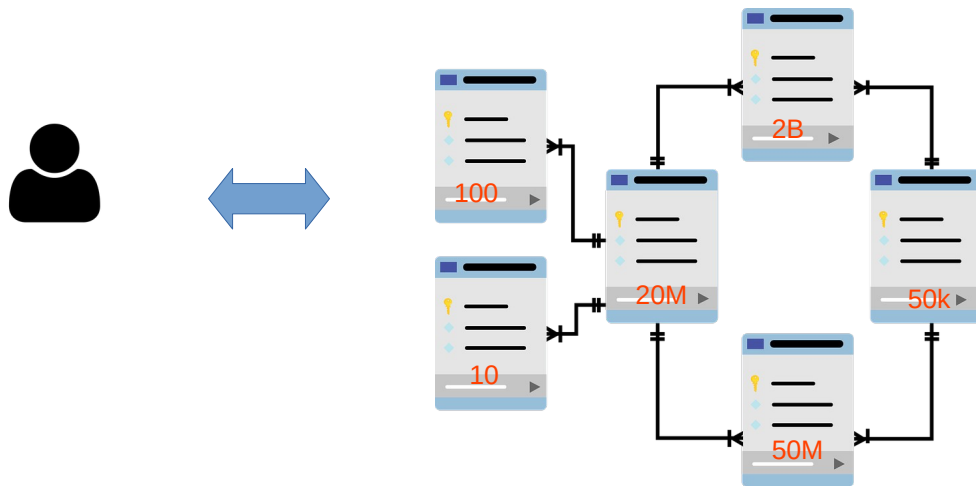
# Postgres é lento?

- No geral o desempenho das consultas é ótimo, problema é a quantidade de dados



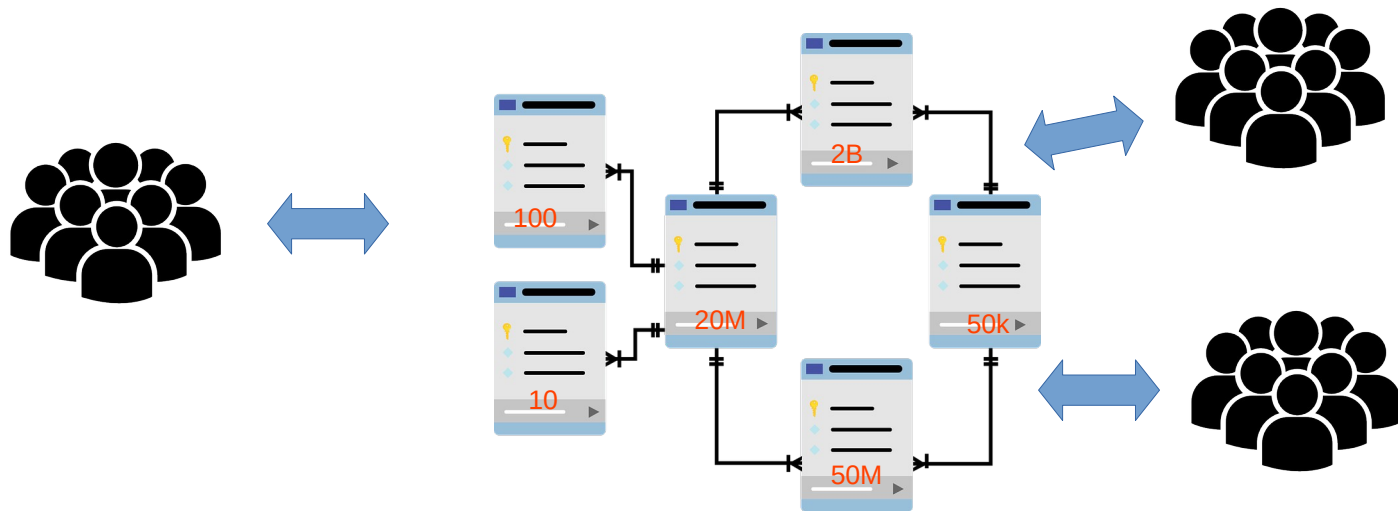
# Postgres é lento?

- No geral o desempenho das consultas é ótimo, problema é a quantidade de dados
  - Vários algoritmos pesados



# Postgres é lento?

- No geral o desempenho das consultas é ótimo, problema é a quantidade de dados e a quantidade de consultas



## Postgres é lento?

---

- No geral o desempenho das consultas é ótimo, problema é a quantidade de dados
- Consultas são escritas e testadas em um ambiente diferente do de produção
  - Com uma escala, geralmente, muito menor

# Otimizar oque?

---

- Otimização é um termo abrangente e pode atacar várias frentes
  - Hardware
  - Tunning de Parâmetros
  - Concorrência e Locks
  - Estrutura do Esquema
  - Consultas



# Otimizar oque?

---

- Otimização é um termo abrangente e pode atacar várias frentes
  - **Hardware**
  - Tunning de Parâmetros
  - Concorrência e Locks
  - Estrutura do Esquema
  - Consultas



# Otimizar oque?

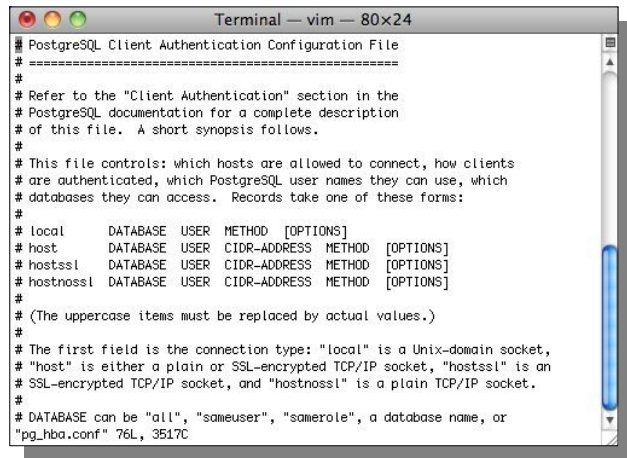
---

- Otimização é um termo abrangente e pode atacar várias frentes
  - **Hardware**
  - Tunning de Parâmetros
  - Concorrência e Locks
  - Estrutura do Esquema
  - Consultas



# Otimizar oque?

- Otimização é um termo abrangente e pode atacar várias frentes
  - Hardware
  - **Tunning de Parâmetros**
  - Concorrência e Locks
  - Estrutura do Esquema
  - Consultas

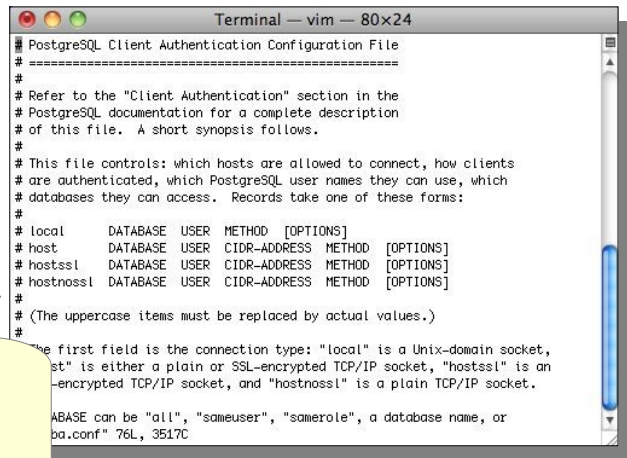


```
Terminal — vim — 80x24
# PostgreSQL Client Authentication Configuration File
# =====
#
# Refer to the "Client Authentication" section in the
# PostgreSQL documentation for a complete description
# of this file. A short synopsis follows.
#
# This file controls: which hosts are allowed to connect, how clients
# are authenticated, which PostgreSQL user names they can use, which
# databases they can access. Records take one of these forms:
#
# local    DATABASE USER METHOD [OPTIONS]
# host     DATABASE USER CIDR-ADDRESS METHOD [OPTIONS]
# hostssl  DATABASE USER CIDR-ADDRESS METHOD [OPTIONS]
# hostnossl DATABASE USER CIDR-ADDRESS METHOD [OPTIONS]
#
# (The uppercase items must be replaced by actual values.)
#
# The first field is the connection type: "local" is a Unix-domain socket,
# "host" is either a plain or SSL-encrypted TCP/IP socket, "hostssl" is an
# SSL-encrypted TCP/IP socket, and "hostnossl" is a plain TCP/IP socket.
#
# DATABASE can be "all", "sameuser", "samerole", a database name, or
# "pg_hba.conf" 76L, 3517C
```

# Otimizar oque?

- Otimização é um termo abrangente e pode atacar várias frentes
- Hardware
- **Tunning de Parâmetros**
- Concorrência e Locks
- Estrutura do Esquema
- Consultas

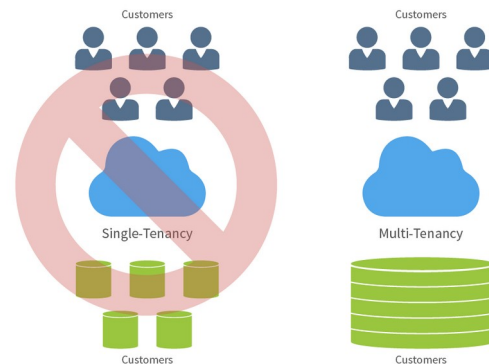
*random\_page\_cost*  
*work\_mem*  
Etc..



```
Terminal — vim — 80x24
# PostgreSQL Client Authentication Configuration File
# =====
#
# Refer to the "Client Authentication" section in the
# PostgreSQL documentation for a complete description
# of this file. A short synopsis follows.
#
# This file controls: which hosts are allowed to connect, how clients
# are authenticated, which PostgreSQL user names they can use, which
# databases they can access. Records take one of these forms:
#
# local   DATABASE USER METHOD [OPTIONS]
# host    DATABASE USER CIDR-ADDRESS METHOD [OPTIONS]
# hostssl DATABASE USER CIDR-ADDRESS METHOD [OPTIONS]
# hostnossl DATABASE USER CIDR-ADDRESS METHOD [OPTIONS]
#
# (The uppercase items must be replaced by actual values.)
#
# The first field is the connection type: "local" is a Unix-domain socket,
# "host" is either a plain or SSL-encrypted TCP/IP socket, "hostssl" is an
# SSL-encrypted TCP/IP socket, and "hostnossl" is a plain TCP/IP socket.
#
# DATABASE can be "all", "sameuser", "samerole", a database name, or
# "replica".
#
# Examples:
# local   all             all                                     trust
# host    all             192.168.1.0/24                       reject
# hostssl all             192.168.1.0/24                       reject
# hostnossl all           192.168.1.0/24                       reject
```

# Otimizar o que?

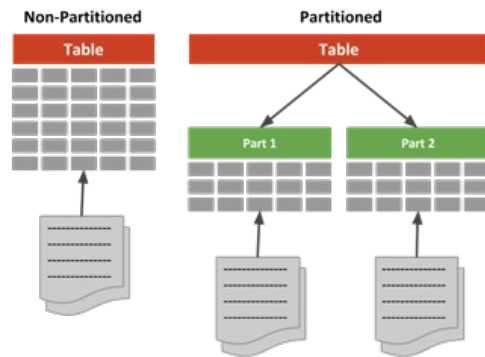
- Otimização é um termo abrangente e pode atacar várias frentes
  - Hardware
  - Tuning de Parâmetros
  - **Concorrência e Locks**
  - Estrutura do Esquema
  - Consultas



Tenancy

# Otimizar oque?

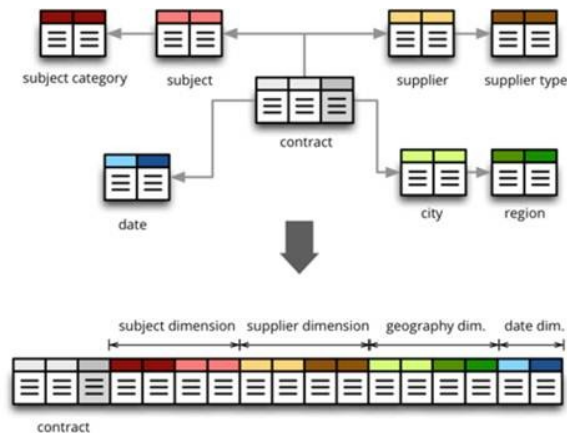
- Otimização é um termo abrangente e pode atacar várias frentes
  - Hardware
  - Tuning de Parâmetros
  - **Concorrência e Locks**
  - Estrutura do Esquema
  - Consultas



Particionamento

# Otimizar o que?

- Otimização é um termo abrangente e pode atacar várias frentes
  - Hardware
  - Tuning de Parâmetros
  - Concorrência e Locks
  - **Estrutura do Esquema**
  - Consultas



Desnormalização

# Otimizar oque?

---

- Otimização é um termo abrangente e pode atacar várias frentes
  - Hardware
  - Tunning de Parâmetros
  - Concorrência e Locks
  - Estrutura do Esquema
  - **Consultas**



*Identificar, analisar e reescrever consultas.*

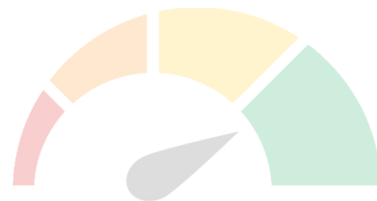


# Otimizar o que?

- Otimização é um termo abrangente e pode atacar várias frentes



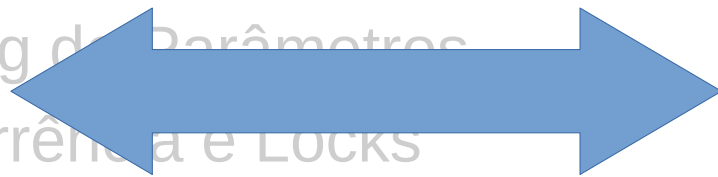
- Software
- Tuning de Parâmetros
- Concorrência e Locks
- Estrutura do Esquema
- Consultas



*Identificar, analisar e reescrever consultas.*

# Otimizar o que?

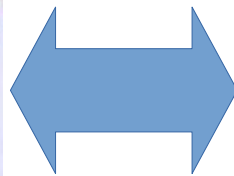
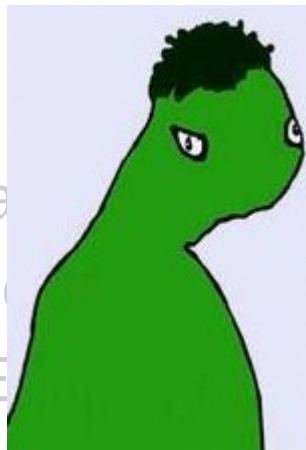
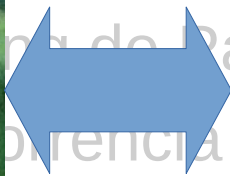
- Otimização é um termo abrangente e pode atacar várias frentes



*Identificação e reescrita de consultas.*

# Otimizar o que?

- Otimização é um termo abrangente e pode atacar várias frentes



Identificação  
reescrever consultas.


# Otimizar Consultas

---

- Identificar consultas lentas pode ser complexo
  - Usar um profiler
    - Software externo que captura as consultas e cria estatísticas


# Otimizar Consultas

---

- Identificar consultas lentas pode ser complexo
  - Usar um profiler
    - Software externo que captura as consultas e cria estatísticas
  - Utilizar os Logs do Postgres
    - log\_statment pode ser utilizado para controle das consultas
    - Análise posterior feita com pgBadger 

# Otimizar Consultas

---

- Identificar consultas lentas pode ser complexo
  - Usar um profiler
    - Software externo que captura as consultas e cria estatísticas
  - Utilizar os Logs do Postgres
    - log\_statment pode ser utilizado para controle das consultas
    - Análise posterior feita com pgBadger 
  - Implementar um profiler com cron e pg\_activity
    - Processo que pega as consultas que estão rodando periodicamente e armazena num DB para análise

# Execução SQL

---

```
1 SELECT
2     DISTINCT
3     a,
4     b
5     --projeção
6 FROM
7     tabela1
8     JOIN tabela2
9     JOIN tabela3
10    ...
11 WHERE
12     a > 20
13     AND b LIKE '%'
14     --predicado
15 GROUP BY
16     X,Y
17 HAVING filtro
18 ORDER BY X,Y
19 LIMIT 1 OFFSET 0
```

# Execução SQL

---

```
1 SELECT
2     DISTINCT
3     a,
4     b
5     --projeção
6 FROM
7     tabela1
8     JOIN tabela2 1
9     JOIN tabela3
10    ...
11 WHERE
12     a > 20
13     AND b LIKE 's' 2
14     --predicado
15 GROUP BY
16     X,Y
17 HAVING filtro
18 ORDER BY X,Y
19 LIMIT 1 OFFSET 0
```



# Execução SQL

---

```
1 SELECT
2     DISTINCT
3     a,
4     b
5     --projeção
6 FROM
7     tabela1
8     JOIN tabela2 1
9     JOIN tabela3
10    ...
11 WHERE
12     a > 20
13     AND b LIKE 's' 2
14     --predicado
15 GROUP BY
16     X,Y 3
17 HAVING filtro
18 ORDER BY X,Y
19 LIMIT 1 OFFSET 0
```

# Execução SQL

---

```
1 SELECT
2     DISTINCT
3     a,
4     b
5     --projeção
6 FROM
7     tabela1
8     JOIN tabela2 1
9     JOIN tabela3
10    ...
11 WHERE
12     a > 20
13     AND b LIKE 's' 2
14     --predicado
15 GROUP BY
16     X,Y 3
17 HAVING filtro
18 ORDER BY X,Y 4
19 LIMIT 1 OFFSET 0
```

# Execução SQL

1	<b>SELECT</b>	
2	<b>DISTINCT</b>	
3	a,	
4	b	<b>5</b>
5	--projeção	
6	<b>FROM</b>	
7	tabela1	
8	<b>JOIN</b> tabela2	<b>1</b>
9	<b>JOIN</b> tabela3	
10	...	
11	<b>WHERE</b>	
12	a > 20	
13	<b>AND</b> b <b>LIKE</b> 's'	<b>2</b>
14	--predicado	
15	<b>GROUP BY</b>	
16	X,Y	<b>3</b>
17	<b>HAVING</b> filtro	
18	<b>ORDER BY</b> X,Y	<b>4</b>
19	<b>LIMIT</b> 1 <b>OFFSET</b> 0	

# Execução SQL

1	<b>SELECT</b>	
2	<b>DISTINCT</b>	<b>6</b>
3	a,	
4	b	<b>5</b>
5	--projeção	
6	<b>FROM</b>	
7	tabela1	
8	<b>JOIN</b> tabela2	<b>1</b>
9	<b>JOIN</b> tabela3	
10	...	
11	<b>WHERE</b>	
12	a > 20	
13	<b>AND</b> b <b>LIKE</b> 's'	<b>2</b>
14	--predicado	
15	<b>GROUP BY</b>	
16	X,Y	<b>3</b>
17	<b>HAVING</b> filtro	
18	<b>ORDER BY</b> X,Y	<b>4</b>
19	<b>LIMIT</b> 1 <b>OFFSET</b> 0	

# Execução SQL

1	<b>SELECT</b>	
2	<b>DISTINCT</b>	<b>6</b>
3	a,	
4	b	<b>5</b>
5	--projeção	
6	<b>FROM</b>	
7	tabela1	
8	<b>JOIN</b> tabela2	<b>1</b>
9	<b>JOIN</b> tabela3	
10	...	
11	<b>WHERE</b>	
12	a > 20	
13	<b>AND</b> b <b>LIKE</b> 's'	<b>2</b>
14	--predicado	
15	<b>GROUP BY</b>	
16	X,Y	<b>3</b>
17	<b>HAVING</b> filtro	
18	<b>ORDER BY</b> X,Y	<b>4</b>
19	<b>LIMIT</b> 1 <b>OFFSET</b> 0	<b>7</b>

# Otimizar Consultas

---

- Premissa de otimização
  - Devemos sempre preconizar a redução dos dados que serão utilizados nas junções (JOIN).

# Otimizar Consultas

---

- Premissa de otimização
  - Devemos sempre preconizar a redução dos dados que serão utilizados nas junções (JOIN).



# Otimizar Consultas

---

- Premissa de otimização
  - Devemos sempre preconizar a redução dos dados que serão utilizados nas junções (JOIN).
    - Alterar ordem de junções
    - CTEs
    - Agrupar informações previamente
    - E outros...
- Indexação
- Particionamento



# Dicas de Otimização

---

- DISTINCT

```
1 SELECT
2     DISTINCT
3     c.nome,
4     h.tipo
5 FROM clientes c
6     JOIN habilitacoes h ON h.codh = c.codh
```

# Dicas de Otimização

- DISTINCT

```
1 SELECT  
2     DISTINCT  
3     c.nome,  
4     h.tipo  
5 FROM clientes c  
6     JOIN habilitacoes h ON
```



# Dicas de Otimização

- DISTINCT

```
1 SELECT
2     DISTINCT
3     c.nome,
4     h.tipo
5 FROM cliente
6 JOIN habilitação ON c.codh
```

Neste contexto está errado, mas quando necessário deve ser utilizado!

# Dicas de Otimização

---

- JOIN

```
1= SELECT  
2     c.nome,  
3     h.tipo  
4 FROM clientes c, habitacoes h  
5 WHERE h.codh = c.codh
```

# Dicas de Otimização

- JOIN

```
1 SELECT  
2     c.nome,  
3     h.tipo  
4 FROM clientes c, habilit  
5 WHERE h.codh = c.codh
```



# Dicas de Otimização

- JOIN

```
1 SELECT
```

```
2     c.nome,
```

```
3     h.tipo
```

```
4 FROM clientes c
```

```
5 WHERE h.codh = c.codh
```

```
1 SELECT
```

```
2     c.nome,
```

```
3     h.tipo
```

```
4 FROM clientes c
```

```
5     JOIN habitacoes h ON h.codh = c.codh
```

## Dicas de Otimização

---

- Preciso contar quantos clientes existem na minha base de dados

```
SELECT count(*) FROM clientes c ;  
SELECT count(cpf) FROM clientes c ;
```

## Dicas de Otimização

---

- Apresentar as habilitações e quantos clientes a possuem

```
1 SELECT
2     tipo,
3     count(c.cpf)
4 FROM clientes c
5     JOIN habilitacoes h ON h.codh = c.codh
6 GROUP BY tipo
```



# Dicas de Otimização

- Apresentar as habilitações e quantos clientes a possuem

```
1 SELECT
2     tipo,
3     count(c.cpf)
4 FROM clientes c
5     JOIN habilitacoes h ON h.codh = c.codh
6 GROUP BY tipo
```

*Join de 400k Clientes  
com 5 habilitações,  
agrupa os dados*

## Dicas de Otimização

- Apresentar as habilitações e quantos clientes a possuem

```
1 SELECT
2     tipo,
3     count(c.cpf)
4 FROM clientes c
5     JOIN habilitacoes
6 GROUP BY tipo
```

```
1 WITH cte_simples AS (
2     SELECT
3         c.codh,
4         count(cpf) AS qt
5     FROM clientes c
6     GROUP BY c.codh
7 )
8 SELECT
9     tipo,
10    qt
11 FROM cte_simples c
12     JOIN habilitacoes h ON h.codh = c.codh
```

# Dicas de Otimização

- Apresentar as habilitações e quantos clientes a possuem

```
1 SELECT
2     tipo,
3     count(c.cpf)
4 FROM clientes c
5     JOIN habilitacoes
6 GROUP BY tipo
```

Agrupa 400k Clientes  
depois join com 5  
habilitações

```
1 WITH cte_simples AS (
2     SELECT
3         c.codh,
4         count(cpf) AS qt
5     FROM clientes c
6     GROUP BY c.codh
7 )
8 SELECT
9     tipo,
10    qt
11 FROM cte_simples c
12     JOIN habilitacoes h ON h.codh = c.codh
```

## Dicas de Otimização

- Apresente os Veículos locados por clientes com habilitação do tipo 'X'

```
1 SELECT
2     v.matricula,
3     v.nome,
4     v.modelo
5 FROM veiculos v
6     JOIN locacoes l ON v.matricula = l.matricula
7     JOIN clientes c ON c.cpf = l.cpf
8     JOIN habilitacoes h ON h.codh = c.codh
9 WHERE h.tipo = 'X'
```

## Dicas de Otimização

- Apresente os Veículos locados por clientes com habilitação do tipo 'X'

```
1 SELECT
2     v.matricula,
3
4     (
5         SELECT
6             v.matricula,
7             v.nome,
8             v.modelo
9         FROM veiculos v
10            JOIN locacoes l ON v.matricula = l.matricula
11            WHERE EXISTS (SELECT 1 FROM clientes c
12                          JOIN habilitacoes h ON h.codh = c.codh
13                          WHERE c.cpf = l.cpf AND h.tipo = 'X')
14     )
```

## Dicas de Otimização

---

- Quando usar uma subconsulta em IN tente fazer uma CTE materializada
- Abuse das funções Window Function
- Sempre que alterar um operador leve em consideração a complexidade que gera na consulta e o possível ganho
  - Teste tudo
- UNION ALL é preferível ao UNION (quando possível)
- Evite OR o otimizador não consegue assumir muita coisa

# Dicas de Otimização

---

- Indexação
  - Índices são uma parte importante na otimização, muitas vezes é preferível criar um índice e não alterar consultas
  - Para criar bons índices é necessário conhecer bem o BD, para poder tirar o máximo de desempenho sem causar prejuízos

# Dicas de Otimização

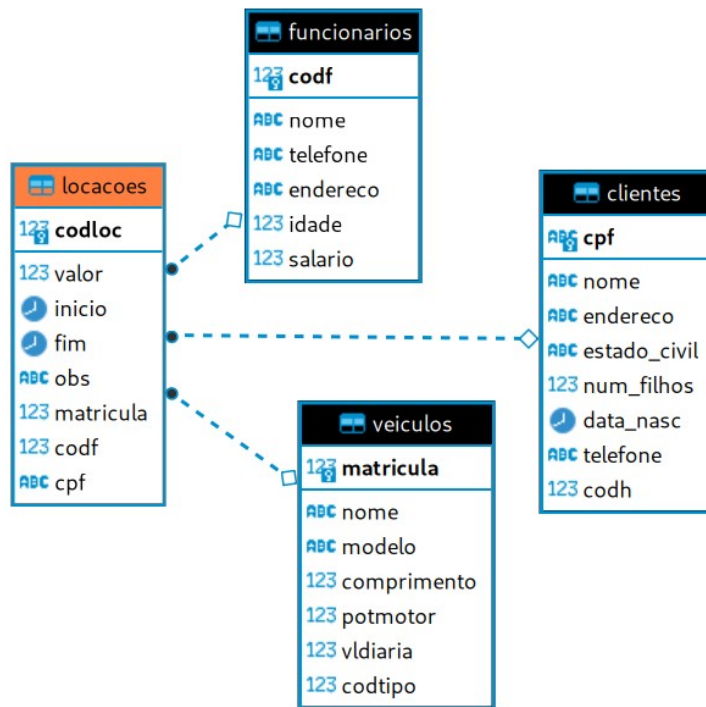
---

- Indexação
  - Evitar criar muitos índices
  - Eliminar sempre que possível índices não utilizados ou duplicados
  - Considerar mais consultas na criação de um índice



# Dicas de Otimização

- Indexação



# Dicas de Otimização

---

- Indexação

```
1 SELECT
2     c.nome cliente,
3     inicio,
4     v.nome barco
5 FROM clientes c
6     JOIN locacoes l ON c.cpf = l.cpf
7     JOIN veiculos v ON v.matricula = l.matricula
8 WHERE inicio = now()
```

# Dicas de Otimização

---

- Indexação

```
1 SELECT
2     c.nome cliente,
3     inicio,
4     v.nome barco
5 FROM clientes c
6     JOIN locacoes l ON c.cpf = l.cpf
7     JOIN veiculos v ON v.matricula = l.matricula
8 WHERE inicio = now()
```

```
1 CREATE INDEX idx_data ON locacoes
2 (inicio);
3 CREATE INDEX idx_cpf ON locacoes
4 (cpf);
5 CREATE INDEX idx_data ON locacoes
6 (matricula);
```

# Dicas de Otimização

- Indexação

```
1 SELECT
2     c.nome cliente,
3     inicio,
4     v.nome barco
5 FROM clientes c
6     JOIN locacoes l ON c.cpf = l.cpf
7     JOIN veiculos v ON v.matricula = l.matricula
8 WHERE inicio = now()
```

```
1 CREATE INDEX idx_data ON locacoes
2   (inicio);
3 CREATE INDEX idx_cpf ON locacoes
4   (cpf);
5 CREATE INDEX idx_data ON locacoes
6   (matricula);
```

```
1 CREATE INDEX idx_data ON locacoes
2   (inicio, cpf, matricula);
```

## Dicas de Otimização

---

- Indexação
  - Utilize o INCLUDE do índice pois ele pode eliminar um look up
  - Quando são utilizadas funções na busca (upper e afins) o BD não usa um índice simples
  - Like “%nome%” não usa índice tradicional
    - GIN

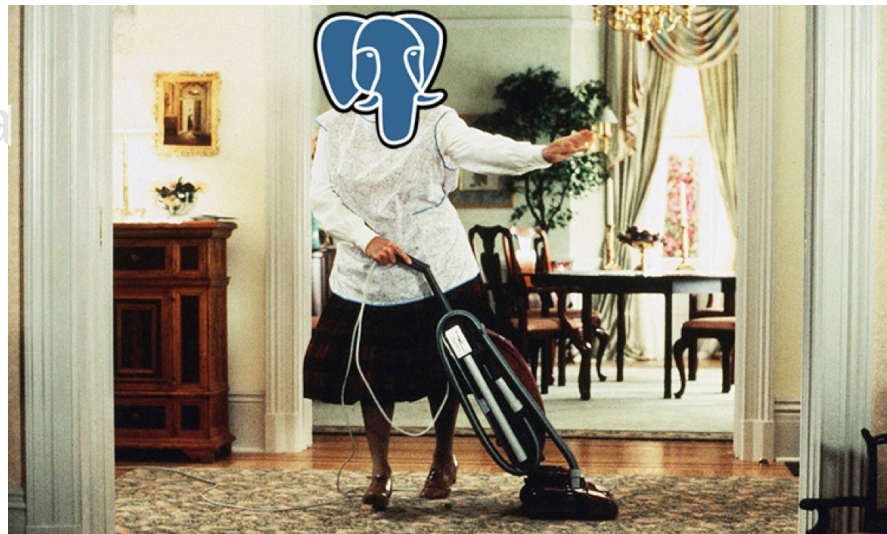
## Dicas de Otimização

---

- Indexação
  - Utilize o INCLUDE do índice pois ele pode eliminar um look up
  - Like “%nome%” não usa índice tradicional
    - GIN
  - Utilizando índice lembre do auto-vacuum

# Dicas de Otimização

- Indexação
  - Utilize o INCLUDE do índice pois ele pode eliminar um loock up
  - Like “%nome%” não usa
  - GIN
  - Utilizando índice lembre

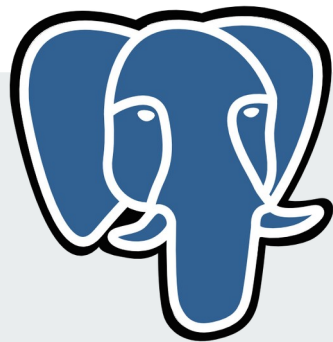


# Finalizando!

---

- Muitas coisas ainda poderiam ser vistas
  - Operadores mais divertidos em SQL
  - Índices mais complexos (com funções e Where)
  - View Materializada e Stored Procedures
  - Particionamento
- Ter sempre em mente a premissa de otimização, e testar muitas vezes a consulta alterada
- Geralmente o problema do BD é consulta mal escrita





# Dúvidas?

**Geomar A Schreiner**  
**[schreiner@dantedb.tech](mailto:schreiner@dantedb.tech)**