Trilha Prática 2

Queries:

```
--QUERIES BASICAS
--1 Quais são os rfids cadastrados até agora?
SELECT * FROM tbl_rfid;

--2 Quais são as categorias cadastradas até agora?
SELECT * FROM tbl_categoria;

--3 Quais são os clientes cadastrados até agora?
SELECT * FROM tbl_cliente;

--4 Quais são os estabelecimentos cadastrados até agora?
SELECT * FROM tbl_estabelecimento;

--5 Quais são os fornecedores cadastrados até agora?
SELECT * FROM tbl_fornecedor;

--6 Quais são os produtos cadastrados até agora?
SELECT * FROM tbl_produto;

--7 Quais são os funcionários cadastrados até agora?
SELECT * FROM tbl_funcionario;

--8 Quais são os pedidos cadastrados até agora?
SELECT * FROM tbl_pedido;

--9 Quais são os produtos de um pedido específico? (Ex.: Pedido 1)
SELECT * FROM pedido_produto WHERE cp_id_pedido = 1;

--10 Quais são os produtos vendidos por um estabelecimento? (Ex.: Estabelecimento 1)
SELECT * FROM estab_produto WHERE cp_cod_estab = 1;

--11 Quais são os produtos fornecidos por um fornecedor específico? (Ex.: Fornecedor 1)
SELECT * FROM forn_produto WHERE cp_cod_forn = 1;

--12 Quais são os dados cadastrados de um cliente específico? (Ex.: Cliente 1)
SELECT * FROM tbl_cliente WHERE cp_cod_cliente = 1;

--13 Quais são os dados de um produto específico? (Ex.: Produto 1)
SELECT * FROM tbl_produto WHERE cp_id_produto = 1;

--14 Quais pedidos foram feitos por um cliente específico? (Ex.: Cliente 1)
SELECT * FROM tbl_pedido WHERE ce_cod_cliente = 1;
```

```
--15 Quais os produtos de uma categoria específica? (Ex.: Categoria 1)
SELECT * FROM tbl_produto WHERE ce_categoria_principal = 1;

--16 Quais os produtos vendidos em um dia específico? (Ex.: 18/01/2025)
SELECT * FROM estab_produto WHERE data_venda_estab = '2025-01-18';

--17 Quais os estabelecimentos de uma cidade específica? (Ex.: CDD)
SELECT * FROM tbl_estabelecimento WHERE cidade_estab = 'CDD';

--18 Quais são os produtos com preço de venda acima de um valor específico? (Ex.: 2.00)
SELECT * FROM estab_produto WHERE preco_venda_estab > 2.00;

--19 Quais são os funcionários de um estabelecimento específico? (Ex.: Estabelecimento 1)
SELECT * FROM tbl_funcionario WHERE ce_cod_estab = 1;

--20 Qual produto possui um código de barras específico? (Ex.: 000000000001)
SELECT * FROM tbl_produto WHERE cd_ean_prod = '000000000001';

-- QUERIES INTERMEDIÁRIAS --
--1 Quais são os pedidos cadastrados e os nomes dos seus respectivos clientes?
SELECT p.cp_id_pedido, p.data_pedido, p.hora_pedido, p.valor_total, c.nm_cliente
FROM tbl_pedido p
JOIN tbl_cliente c ON p.ce_cod_cliente = c.cp_cod_cliente;

--2 Quais são os produtos cadastrados e os nomes das suas respectivas categorias
principais?
SELECT pr.cp_id_produto, pr.nm_produto, c.nm_categoria
FROM tbl_produto pr
JOIN tbl_categoria c ON pr.ce_categoria_principal = c.cp_cod_categoria;

--3 Quais foram os fornecedores que forneceram produtos e quais produtos foram
fornecidos?
SELECT f.nm_forn, pr.nm_produto
FROM forn_produto fp
JOIN tbl_produto pr ON fp.cp_id_produto = pr.cp_id_produto
JOIN tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn;

--4 Quais estabelecimentos venderam produtos e quais produtos foram vendidos?
SELECT e.nm_estab, pr.nm_produto
FROM estab_produto ep
JOIN tbl_produto pr ON ep.cp_id_produto = pr.cp_id_produto
JOIN tbl_estabelecimento e ON ep.cp_cod_estab = e.cp_cod_estab;

--5 Quais são os pedidos e quais os produtos vendidos nos seus respectivos pedidos?
SELECT p.cp_id_pedido, pr.nm_produto
FROM pedido_produto pp
JOIN tbl_produto pr ON pp.cp_id_produto = pr.cp_id_produto
```

```
JOIN tbl_pedido p ON pp.cp_id_pedido = p.cp_id_pedido;

--6 Quais são os funcionários e seus respectivos estabelecimentos nos quais eles trabalham?
SELECT f.nm_func, e.nm_estab
FROM tbl_funcionario f
JOIN tbl_estabelecimento e ON f.ce_cod_estab = e.cp_cod_estab;

--7 Quais foram os pedidos, seus valores totais e os clientes que realizaram os pedidos?
SELECT p.cp_id_pedido, p.valor_total, c.nm_cliente
FROM tbl_pedido p
JOIN tbl_cliente c ON p.ce_cod_cliente = c.cp_cod_cliente;

--8 Quais são as categorias secundárias de cada produto e qual o nome dos produtos?
SELECT c.nm_categoria, pr.nm_produto
FROM tbl_produto pr
JOIN tbl_categoria c ON pr.ce_categoria_secundaria = c.cp_cod_categoria;

--9 Quais fornecedores venderam quais produtos e quais os preços desses produtos?
SELECT f.nm_forn, pr.nm_produto, fp.preco_venda_forn
FROM forn_produto fp
JOIN tbl_produto pr ON fp.cp_id_produto = pr.cp_id_produto
JOIN tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn;

--10 Quais são os produtos seus estabelecimentos e seus respectivos preços de venda?
SELECT pr.nm_produto, e.nm_estab, ep.preco_venda_estab
FROM estab_produto ep
JOIN tbl_produto pr ON ep.cp_id_produto = pr.cp_id_produto
JOIN tbl_estabelecimento e ON ep.cp_cod_estab = e.cp_cod_estab;

--11 Quais são os produtos, fornecedores, preços de venda e suas datas de vencimento?
SELECT pr.nm_produto, f.nm_forn, fp.preco_venda_forn, fp.data_vencimento
FROM forn_produto fp
JOIN tbl_produto pr ON fp.cp_id_produto = pr.cp_id_produto
JOIN tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn;

--12 Quais são os funcionários, seus estabelecimentos e cidades em que trabalham?
SELECT f.nm_func, e.nm_estab, e.cidade_estab
FROM tbl_funcionario f
JOIN tbl_estabelecimento e ON f.ce_cod_estab = e.cp_cod_estab;

--13 Quais são os fornecedores, seus produtos e os estabelecimentos nos quais são vendidos?
SELECT f.nm_forn, pr.nm_produto, e.nm_estab
FROM forn_produto fp
JOIN tbl_produto pr ON fp.cp_id_produto = pr.cp_id_produto
JOIN tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn
JOIN estab_produto ep ON pr.cp_id_produto = ep.cp_id_produto
```

```
JOIN tbl_estabelecimento e ON ep.cp_cod_estab = e.cp_cod_estab;

--14 Quais são as localizações dos estabelecimentos e dos fornecedores?
SELECT e.nm_estab, e.localizacao_estab, f.nm_forn, f.localizacao_forn
FROM tbl_estabelecimento e
JOIN forn_produto fp ON e.cp_cod_estab = fp.cp_cod_forn
JOIN tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn;

--15 Quais são os pedidos, os clientes que os realizaram, a data e o horário de realização do pedido?
SELECT p.cp_id_pedido, c.nm_cliente, p.data_pedido, p.hora_pedido
FROM tbl_pedido p
JOIN tbl_cliente c ON p.ce_cod_cliente = c.cp_cod_cliente;

-- QUERIES AVANÇADAS --
--1 Quais produtos possuem vendas acima da média e como eles contribuem para o total de vendas de sua categoria principal, ordenados pela contribuição total da categoria?
WITH produto_vendido_estab AS (
	SELECT
	p.cp_id_produto,
	SUM(e.preco_venda_estab) AS total_vendas_estab
	FROM estab_produto e
	JOIN tbl_produto p ON e.cp_id_produto = p.cp_id_produto
	GROUP BY p.cp_id_produto
),
produto_vendido_forn AS (
	SELECT
	f.cp_id_produto,
	SUM(f.preco_venda_forn) AS total_vendas_forn
	FROM forn_produto f
	GROUP BY f.cp_id_produto
),
produto_union_vendas AS (
	SELECT
	cp_id_produto,
	total_vendas_estab AS total_vendas
	FROM produto_vendido_estab
	UNION ALL
	SELECT
	cp_id_produto,
	total_vendas_forn AS total_vendas
	FROM produto_vendido_forn
)
SELECT
	p.cp_id_produto,
	p.nm_produto,
	c.nm_categoria,
```

```
	SUM(u.total_vendas) OVER (PARTITION BY p.ce_categoria_principal) AS
total_vendas_categoria,
	u.total_vendas
FROM produto_union_vendas u
JOIN tbl_produto p ON u.cp_id_produto = p.cp_id_produto
LEFT JOIN tbl_categoria c ON p.ce_categoria_principal = c.cp_cod_categoria
WHERE u.total_vendas > (
	SELECT AVG(total_vendas)
	FROM produto_union_vendas
)
ORDER BY total_vendas_categoria DESC, u.total_vendas DESC;

--2 Quais são os produtos mais vendidos por estabelecimento (considerando fornecedores e
estabelecimentos juntos), categorizados por tipo, e como eles se posicionam em um ranking
local para cada estabelecimento?
WITH vendas_por_estabelecimento AS (
	SELECT
	e.cp_cod_estab,
	e.cp_id_produto,
	SUM(e.preco_venda_estab) AS total_vendas_estab
	FROM estab_produto e
	GROUP BY e.cp_cod_estab, e.cp_id_produto
),
vendas_por_fornecedor AS (
	SELECT
	f.cp_cod_forn AS cp_cod_estab, -- Mapeando fornecedor como "estabelecimento"
	f.cp_id_produto,
	SUM(f.preco_venda_forn) AS total_vendas_forn
	FROM forn_produto f
	GROUP BY f.cp_cod_forn, f.cp_id_produto
),
vendas_combinadas AS (
	SELECT
	cp_cod_estab,
	cp_id_produto,
	total_vendas_estab AS total_vendas
	FROM vendas_por_estabelecimento
	UNION ALL
	SELECT
	cp_cod_estab,
	cp_id_produto,
	total_vendas_forn AS total_vendas
	FROM vendas_por_fornecedor
)
SELECT
	e.nm_estab,
	c.nm_categoria,
	p.nm_produto,
```

```
        v.total_vendas,
        RANK() OVER (PARTITION BY e.nm_estab ORDER BY v.total_vendas DESC) AS
ranking_por_estab
FROM vendas_combinadas v
JOIN tbl_estabelecimento e ON v.cp_cod_estab = e.cp_cod_estab
JOIN tbl_produto p ON v.cp_id_produto = p.cp_id_produto
JOIN tbl_categoria c ON p.ce_categoria_principal = c.cp_cod_categoria
WHERE v.total_vendas > (
        SELECT AVG(total_vendas)
        FROM vendas_combinadas
)
GROUP BY e.nm_estab, c.nm_categoria, p.nm_produto, v.total_vendas
ORDER BY e.nm_estab, ranking_por_estab;

--3 Quais categorias de produtos possuem vendas médias acima da média geral e qual é a
participação relativa de cada categoria no total de vendas do sistema?
WITH vendas_por_produto AS (
        SELECT
        p.cp_id_produto,
        SUM(e.preco_venda_estab) AS total_vendas_estab,
        SUM(f.preco_venda_forn) AS total_vendas_forn
        FROM tbl_produto p
        LEFT JOIN estab_produto e ON p.cp_id_produto = e.cp_id_produto
        LEFT JOIN forn_produto f ON p.cp_id_produto = f.cp_id_produto
        GROUP BY p.cp_id_produto
),
total_vendas_comb AS (
        SELECT
        cp_id_produto,
        COALESCE(total_vendas_estab, 0) + COALESCE(total_vendas_forn, 0) AS
total_vendas
        FROM vendas_por_produto
),
vendas_totais_sistema AS (
        SELECT SUM(total_vendas) AS vendas_totais_sistema
        FROM total_vendas_comb
)
SELECT
        c.nm_categoria,
        COUNT(DISTINCT p.cp_id_produto) AS qtd_produtos,
        AVG(t.total_vendas) AS media_vendas_categoria,
        (SELECT vendas_totais_sistema FROM vendas_totais_sistema) AS
vendas_totais_sistema
FROM total_vendas_comb t
JOIN tbl_produto p ON t.cp_id_produto = p.cp_id_produto
JOIN tbl_categoria c ON p.ce_categoria_principal = c.cp_cod_categoria
WHERE t.total_vendas > (
        SELECT AVG(total_vendas)
```

```
        FROM total_vendas_comb
)
GROUP BY c.nm_categoria
ORDER BY media_vendas_categoria DESC;

--4 Quais cidades possuem os maiores volumes de vendas em cada estado, considerando apenas os produtos com vendas acima da média, e qual é o ranking das cidades dentro de cada estado?
WITH vendas_por_estabelecimento AS (
        SELECT
        e.cp_cod_estab,
        e.cp_id_produto,
        SUM(e.preco_venda_estab) AS total_vendas_estab
        FROM estab_produto e
        GROUP BY e.cp_cod_estab, e.cp_id_produto
),
vendas_por_fornecedor AS (
        SELECT
        f.cp_cod_forn AS cp_cod_estab, -- Mapeando fornecedor como "estabelecimento"
        f.cp_id_produto,
        SUM(f.preco_venda_forn) AS total_vendas_forn
        FROM forn_produto f
        GROUP BY f.cp_cod_forn, f.cp_id_produto
),
vendas_combinadas AS (
        SELECT
        cp_cod_estab,
        cp_id_produto,
        total_vendas_estab AS total_vendas
        FROM vendas_por_estabelecimento
        UNION ALL
        SELECT
        cp_cod_estab,
        cp_id_produto,
        total_vendas_forn AS total_vendas
        FROM vendas_por_fornecedor
),
vendas_por_uf AS (
        SELECT
        e.uf_estab,
        SUM(v.total_vendas) AS total_vendas_uf
        FROM vendas_combinadas v
        JOIN tbl_estabelecimento e ON v.cp_cod_estab = e.cp_cod_estab
        GROUP BY e.uf_estab
)
SELECT
        e.uf_estab,
        e.cidade_estab,
```

```
        COUNT(DISTINCT v.cp_id_produto) AS qtd_produtos_vendidos,
        SUM(v.total_vendas) AS total_vendas_cidade,
        RANK() OVER (PARTITION BY e.uf_estab ORDER BY SUM(v.total_vendas) DESC)
AS ranking_cidade_uf
FROM vendas_combinadas v
JOIN tbl_estabelecimento e ON v.cp_cod_estab = e.cp_cod_estab
WHERE v.total_vendas > (
        SELECT AVG(total_vendas)
        FROM vendas_combinadas
)
GROUP BY e.uf_estab, e.cidade_estab
ORDER BY e.uf_estab, ranking_cidade_uf;

--5 Quais são as três categorias com maior volume total de vendas, quantos produtos cada
uma possui, e qual é a média de vendas por produto dentro dessas categorias?
WITH vendas_estab AS (
        SELECT
        e.cp_id_produto,
        SUM(e.preco_venda_estab) AS total_vendas_estab
        FROM estab_produto e
        GROUP BY e.cp_id_produto
),
vendas_forn AS (
        SELECT
        f.cp_id_produto,
        SUM(f.preco_venda_forn) AS total_vendas_forn
        FROM forn_produto f
        GROUP BY f.cp_id_produto
),
vendas_comb AS (
        SELECT
        cp_id_produto,
        COALESCE(total_vendas_estab, 0) + COALESCE(total_vendas_forn, 0) AS
total_vendas
        FROM vendas_estab
        FULL OUTER JOIN vendas_forn USING (cp_id_produto)
),
categorias_vendas AS (
        SELECT
        c.nm_categoria,
        COUNT(p.cp_id_produto) AS qtd_produtos,
        SUM(v.total_vendas) AS total_vendas_categoria
        FROM tbl_produto p
        JOIN vendas_comb v ON p.cp_id_produto = v.cp_id_produto
        JOIN tbl_categoria c ON p.ce_categoria_principal = c.cp_cod_categoria
        GROUP BY c.nm_categoria
),
categoria_ranked AS (
```

```
        SELECT
        nm_categoria,
        qtd_produtos,
        total_vendas_categoria,
        RANK() OVER (ORDER BY total_vendas_categoria DESC) AS rank_vendas
        FROM categorias_vendas
)
SELECT
        cr.nm_categoria,
        cr.qtd_produtos,
        cr.total_vendas_categoria,
        cr.rank_vendas,
        AVG(v.total_vendas) AS media_vendas_produtos_categoria
FROM categoria_ranked cr
JOIN tbl_produto p ON p.ce_categoria_principal = (SELECT cp_cod_categoria FROM
tbl_categoria WHERE nm_categoria = cr.nm_categoria LIMIT 1)
JOIN vendas_comb v ON p.cp_id_produto = v.cp_id_produto
WHERE cr.rank_vendas <= 3
GROUP BY cr.nm_categoria, cr.qtd_produtos, cr.total_vendas_categoria, cr.rank_vendas
ORDER BY cr.rank_vendas;

--6 Quais são os cinco produtos mais vendidos em cada categoria, qual foi o total de vendas
de cada um, quantos fornecedores ou estabelecimentos contribuíram para as vendas, e
como eles se comparam com a média geral de vendas?
WITH vendas_por_estabelecimento AS (
        SELECT
        p.cp_id_produto,
        e.cp_cod_estab,
        SUM(e.preco_venda_estab) AS total_vendas_estab
        FROM estab_produto e
        JOIN tbl_produto p ON e.cp_id_produto = p.cp_id_produto
        GROUP BY p.cp_id_produto, e.cp_cod_estab
),
vendas_por_fornecedor AS (
        SELECT
        p.cp_id_produto,
        f.cp_cod_forn,
        SUM(f.preco_venda_forn) AS total_vendas_forn
        FROM forn_produto f
        JOIN tbl_produto p ON f.cp_id_produto = p.cp_id_produto
        GROUP BY p.cp_id_produto, f.cp_cod_forn
),
vendas_combinadas AS (
        SELECT
        cp_id_produto,
        cp_cod_estab AS origem,
        total_vendas_estab AS total_vendas
        FROM vendas_por_estabelecimento
```

```
        UNION ALL
        SELECT
        cp_id_produto,
        cp_cod_forn AS origem,
        total_vendas_forn AS total_vendas
        FROM vendas_por_fornecedor
),
produtos_vendas_totais AS (
        SELECT
        p.cp_id_produto,
        p.nm_produto,
        c.nm_categoria,
        SUM(vc.total_vendas) AS total_vendas_produto,
        COUNT(DISTINCT vc.origem) AS qtd_origens
        FROM vendas_combinadas vc
        JOIN tbl_produto p ON vc.cp_id_produto = p.cp_id_produto
        JOIN tbl_categoria c ON p.ce_categoria_principal = c.cp_cod_categoria
        GROUP BY p.cp_id_produto, p.nm_produto, c.nm_categoria
),
ranking_produtos AS (
        SELECT
        nm_produto,
        nm_categoria,
        total_vendas_produto,
        qtd_origens,
        RANK() OVER (PARTITION BY nm_categoria ORDER BY total_vendas_produto
DESC) AS rank_categoria
        FROM produtos_vendas_totais
)
SELECT
        rp.nm_produto,
        rp.nm_categoria,
        rp.total_vendas_produto,
        rp.qtd_origens,
        rp.rank_categoria,
        AVG(rp.total_vendas_produto) OVER () AS media_vendas_geral
FROM ranking_produtos rp
WHERE rp.rank_categoria <= 5
ORDER BY rp.nm_categoria, rp.rank_categoria;

--7 Quais são os três produtos mais vendidos em cada estado, qual é o volume total de
vendas de cada um, e como esse valor se compara com a média geral das vendas de
produtos em todos os estados?
WITH vendas_estabelecimentos AS (
        SELECT
        e.cp_cod_estab,
        e.cp_id_produto,
        SUM(e.preco_venda_estab) AS total_vendas_estab
```

```
        FROM estab_produto e
        GROUP BY e.cp_cod_estab, e.cp_id_produto
),
vendas_fornecedores AS (
        SELECT
        f.cp_cod_forn,
        f.cp_id_produto,
        SUM(f.preco_venda_forn) AS total_vendas_forn
        FROM forn_produto f
        GROUP BY f.cp_cod_forn, f.cp_id_produto
),
vendas_combinadas AS (
        SELECT
        p.cp_id_produto,
        COALESCE(e.cp_cod_estab, f.cp_cod_forn) AS origem,
        COALESCE(e.total_vendas_estab, 0) + COALESCE(f.total_vendas_forn, 0) AS
total_vendas
        FROM vendas_estabelecimentos e
        FULL OUTER JOIN vendas_fornecedores f USING (cp_id_produto)
        JOIN tbl_produto p ON p.cp_id_produto = COALESCE(e.cp_id_produto,
f.cp_id_produto)
),
vendas_por_estado AS (
        SELECT
        COALESCE(e.uf_estab, f.uf_forn) AS uf,
        vc.cp_id_produto,
        SUM(vc.total_vendas) AS total_vendas_estado
        FROM vendas_combinadas vc
        LEFT JOIN tbl_estabelecimento e ON vc.origem = e.cp_cod_estab
        LEFT JOIN tbl_fornecedor f ON vc.origem = f.cp_cod_forn
        GROUP BY COALESCE(e.uf_estab, f.uf_forn), vc.cp_id_produto
),
ranking_produtos_estado AS (
        SELECT
        vp.uf,
        p.nm_produto,
        SUM(vp.total_vendas_estado) AS vendas_produto_estado,
        RANK() OVER (PARTITION BY vp.uf ORDER BY SUM(vp.total_vendas_estado)
DESC) AS rank_produto_estado
        FROM vendas_por_estado vp
        JOIN tbl_produto p ON vp.cp_id_produto = p.cp_id_produto
        GROUP BY vp.uf, p.nm_produto
)
SELECT
        rpe.uf,
        rpe.nm_produto,
        rpe.vendas_produto_estado,
        rpe.rank_produto_estado,
```

```
	(SELECT AVG(vendas_produto_estado) FROM ranking_produtos_estado) AS
media_vendas_geral
FROM ranking_produtos_estado rpe
WHERE rpe.rank_produto_estado <= 3
ORDER BY rpe.uf, rpe.rank_produto_estado;

--8 Quais são os três principais fornecedores ou estabelecimentos para cada categoria de
produto, com base no volume total de vendas e na quantidade de produtos vendidos, e
como esses valores se comparam à média geral de vendas?
WITH vendas_estabelecimentos AS (
	SELECT
	e.cp_cod_estab AS origem,
	p.ce_categoria_principal AS categoria,
	COUNT(DISTINCT e.cp_id_produto) AS qtd_produtos_estab,
	SUM(e.preco_venda_estab) AS total_vendas_estab
	FROM estab_produto e
	JOIN tbl_produto p ON e.cp_id_produto = p.cp_id_produto
	GROUP BY e.cp_cod_estab, p.ce_categoria_principal
),
vendas_fornecedores AS (
	SELECT
	f.cp_cod_forn AS origem,
	p.ce_categoria_principal AS categoria,
	COUNT(DISTINCT f.cp_id_produto) AS qtd_produtos_forn,
	SUM(f.preco_venda_forn) AS total_vendas_forn
	FROM forn_produto f
	JOIN tbl_produto p ON f.cp_id_produto = p.cp_id_produto
	GROUP BY f.cp_cod_forn, p.ce_categoria_principal
),
vendas_combinadas AS (
	SELECT
	origem,
	categoria,
	COALESCE(qtd_produtos_estab, 0) + COALESCE(qtd_produtos_forn, 0) AS
qtd_produtos,
	COALESCE(total_vendas_estab, 0) + COALESCE(total_vendas_forn, 0) AS
total_vendas
	FROM vendas_estabelecimentos
	FULL OUTER JOIN vendas_fornecedores
	USING (origem, categoria)
),
ranking_categorias AS (
	SELECT
	vc.categoria,
	c.nm_categoria,
	vc.origem,
	vc.qtd_produtos,
	vc.total_vendas,
```

```
        RANK() OVER (PARTITION BY vc.categoria ORDER BY vc.total_vendas DESC) AS
rank_vendas_categoria
        FROM vendas_combinadas vc
        JOIN tbl_categoria c ON vc.categoria = c.cp_cod_categoria
)
SELECT
        r.nm_categoria,
        r.origem,
        r.qtd_produtos,
        r.total_vendas,
        r.rank_vendas_categoria,
        (SELECT AVG(total_vendas) FROM ranking_categorias) AS media_vendas_geral
FROM ranking_categorias r
WHERE r.rank_vendas_categoria <= 3
ORDER BY r.nm_categoria, r.rank_vendas_categoria;

--9 Qual é a diferença entre o preço de venda mais alto de cada produto por fornecedor e o
preço médio de venda do produto, e como isso varia com o número de fornecimentos
realizados por cada fornecedor?
WITH preco_produto_fornecedor AS (
        SELECT
        f.nm_forn,
        p.nm_produto,
        e.preco_venda_estab,
        f.cp_cod_forn,
        p.cp_id_produto,
        ROW_NUMBER() OVER (PARTITION BY p.cp_id_produto ORDER BY
e.preco_venda_estab DESC) AS rank_produto
        FROM estab_produto e
        JOIN tbl_produto p ON e.cp_id_produto = p.cp_id_produto
        JOIN tbl_fornecedor f ON e.cp_cod_estab = f.cp_cod_forn
),
produtos_vendidos_comparacao AS (
        SELECT
        ppf.nm_forn,
        ppf.nm_produto,
        ppf.preco_venda_estab,
        (SELECT AVG(preco_venda_estab)
        FROM estab_produto
        WHERE cp_id_produto = ppf.cp_id_produto) AS preco_medio_produto
        FROM preco_produto_fornecedor ppf
        WHERE ppf.rank_produto = 1
),
vendas_comparadas AS (
        SELECT
        nm_forn,
        nm_produto,
        preco_venda_estab,
```

```
        preco_medio_produto,
        (preco_venda_estab - preco_medio_produto) AS diferenca_preco
        FROM produtos_vendidos_comparacao
)
SELECT
        nm_forn,
        nm_produto,
        preco_venda_estab,
        preco_medio_produto,
        diferenca_preco,
        COUNT(*) OVER (PARTITION BY nm_forn) AS quantidade_fornecimentos
FROM vendas_comparadas
ORDER BY diferenca_preco DESC;

--10 Como posso analisar o total de vendas por pedido, ordenado por cliente, incluindo um ranking de pedidos e filtrando por um cliente específico, enquanto combino informações de produtos vendidos em diferentes estabelecimentos e calculo o valor total dos pedidos durante o ano de 2025?
SELECT
        p.cp_id_pedido,
        p.ce_cod_cliente,
        p.data_pedido,
        p.hora_pedido,
        p.valor_total,
        SUM(ep.preco_venda_estab) AS preco_venda_total,
        ROW_NUMBER() OVER (PARTITION BY p.ce_cod_cliente ORDER BY
p.data_pedido DESC) AS rank_pedido
FROM
        tbl_pedido p
JOIN
        pedido_produto pp ON pp.cp_id_pedido = p.cp_id_pedido
JOIN
        estab_produto ep ON ep.cp_id_produto = pp.cp_id_produto
WHERE
        p.data_pedido BETWEEN '2025-01-01' AND '2025-12-31'
GROUP BY
        p.cp_id_pedido, p.ce_cod_cliente, p.data_pedido, p.hora_pedido, p.valor_total
UNION
SELECT
        p.cp_id_pedido,
        p.ce_cod_cliente,
        p.data_pedido,
        p.hora_pedido,
        p.valor_total,
        0 AS preco_venda_total,
        NULL AS rank_pedido
FROM
        tbl_pedido p
```

```
WHERE
        p.ce_cod_cliente IN (SELECT cp_cod_cliente FROM tbl_cliente WHERE cpf_cliente
= '12345678901')
ORDER BY
        data_pedido DESC;
```

Sem Indexação e Tunning:

Para esta etapa, foi considerado que o banco de dados no qual está o sql ddl do projeto se chama “teste”, o user se chama “postgres”, a senha do pgAdmin é “a”, o host é “localhost” e a porta é “5432”. Foi criado um script em Python (com a terminação .py), no ambiente Visual Studio Code, que executa todas as 45 queries 50 vezes, utilizando a biblioteca psycopg2 para se conectar ao PostgreSQL. O tempo de execução de cada query é coletado usando time.time() antes e depois da execução da query. Os tempos de execução são armazenados em um dicionário e, posteriormente, salvos em um arquivo CSV usando a biblioteca pandas. O DataFrame criado no script acima é salvo como um arquivo CSV chamado baseline.csv. Cada linha do CSV representa uma execução das queries, e cada coluna representa o tempo de execução de uma query específica. O código é o seguinte:

```python
import psycopg2
import time
import pandas as pd

# Conexão com o banco de dados
conn = psycopg2.connect(
    dbname="teste",
    user="postgres",
    password="a",
    host="localhost",
    port="5432"
)

# Lista de queries
queries = [
"SELECT * FROM tbl_rfid;",
"SELECT * FROM tbl_categoria;",
"SELECT * FROM tbl_cliente;",
"SELECT * FROM tbl_estabelecimento;",
"SELECT * FROM tbl_fornecedor;",
"SELECT * FROM tbl_produto;",
"SELECT * FROM tbl_funcionario;",
"SELECT * FROM tbl_pedido;",
"SELECT * FROM pedido_produto WHERE cp_id_pedido = 1;",
"SELECT * FROM estab_produto WHERE cp_cod_estab = 1;",
"SELECT * FROM forn_produto WHERE cp_cod_forn = 1;",
```

```
"SELECT * FROM tbl_cliente WHERE cp_cod_cliente = 1;",
"SELECT * FROM tbl_produto WHERE cp_id_produto = 1;",
"SELECT * FROM tbl_pedido WHERE ce_cod_cliente = 1;",
"SELECT * FROM tbl_produto WHERE ce_categoria_principal = 1;",
"SELECT * FROM estab_produto WHERE data_venda_estab = '2025-01-18';",
"SELECT * FROM tbl_estabelecimento WHERE cidade_estab = 'CDD';",
"SELECT * FROM estab_produto WHERE preco_venda_estab > 2.00;",
"SELECT * FROM tbl_funcionario WHERE ce_cod_estab = 1;",
"SELECT * FROM tbl_produto WHERE cd_ean_prod = '000000000001';"
"SELECT p.cp_id_pedido, p.data_pedido, p.hora_pedido, p.valor_total,
c.nm_cliente FROM tbl_pedido p JOIN tbl_cliente c ON p.ce_cod_cliente =
c.cp_cod_cliente;",
"SELECT pr.cp_id_produto, pr.nm_produto, c.nm_categoria FROM
tbl_produto pr JOIN tbl_categoria c ON pr.ce_categoria_principal =
c.cp_cod_categoria;",
"SELECT f.nm_forn, pr.nm_produto FROM forn_produto fp JOIN tbl_produto
pr ON fp.cp_id_produto = pr.cp_id_produto JOIN tbl_fornecedor f ON
fp.cp_cod_forn = f.cp_cod_forn;",
"SELECT e.nm_estab, pr.nm_produto FROM estab_produto ep JOIN
tbl_produto pr ON ep.cp_id_produto = pr.cp_id_produto JOIN
tbl_estabelecimento e ON ep.cp_cod_estab = e.cp_cod_estab;",
"SELECT p.cp_id_pedido, pr.nm_produto FROM pedido_produto pp JOIN
tbl_produto pr ON pp.cp_id_produto = pr.cp_id_produto JOIN tbl_pedido p
ON pp.cp_id_pedido = p.cp_id_pedido;",
"SELECT f.nm_func, e.nm_estab FROM tbl_funcionario f JOIN
tbl_estabelecimento e ON f.ce_cod_estab = e.cp_cod_estab;",
"SELECT p.cp_id_pedido, p.valor_total, c.nm_cliente FROM tbl_pedido p
JOIN tbl_cliente c ON p.ce_cod_cliente = c.cp_cod_cliente;",
"SELECT c.nm_categoria, pr.nm_produto FROM tbl_produto pr JOIN
tbl_categoria c ON pr.ce_categoria_secundaria = c.cp_cod_categoria;",
"SELECT f.nm_forn, pr.nm_produto, fp.preco_venda_forn FROM forn_produto
fp JOIN tbl_produto pr ON fp.cp_id_produto = pr.cp_id_produto JOIN
tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn;",
"SELECT pr.nm_produto, e.nm_estab, ep.preco_venda_estab FROM
estab_produto ep JOIN tbl_produto pr ON ep.cp_id_produto =
pr.cp_id_produto JOIN tbl_estabelecimento e ON ep.cp_cod_estab =
e.cp_cod_estab;",
"SELECT pr.nm_produto, f.nm_forn, fp.preco_venda_forn,
fp.data_vencimento FROM forn_produto fp JOIN tbl_produto pr ON
fp.cp_id_produto = pr.cp_id_produto JOIN tbl_fornecedor f ON
fp.cp_cod_forn = f.cp_cod_forn;",
"SELECT f.nm_func, e.nm_estab, e.cidade_estab FROM tbl_funcionario f
JOIN tbl_estabelecimento e ON f.ce_cod_estab = e.cp_cod_estab;",
```

```
"SELECT f.nm_forn, pr.nm_produto, e.nm_estab FROM forn_produto fp JOIN
tbl_produto pr ON fp.cp_id_produto = pr.cp_id_produto JOIN
tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn JOIN estab_produto
ep ON pr.cp_id_produto = ep.cp_id_produto JOIN tbl_estabelecimento e ON
ep.cp_cod_estab = e.cp_cod_estab;",
"SELECT e.nm_estab, e.localizacao_estab, f.nm_forn, f.localizacao_forn
FROM tbl_estabelecimento e JOIN forn_produto fp ON e.cp_cod_estab =
fp.cp_cod_forn JOIN tbl_fornecedor f ON fp.cp_cod_forn =
f.cp_cod_forn;",
"SELECT p.cp_id_pedido, c.nm_cliente, p.data_pedido, p.hora_pedido FROM
tbl_pedido p JOIN tbl_cliente c ON p.ce_cod_cliente =
c.cp_cod_cliente;"
"WITH produto_vendido_estab AS (SELECT p.cp_id_produto,
SUM(e.preco_venda_estab) AS total_vendas_estab FROM estab_produto e
JOIN tbl_produto p ON e.cp_id_produto = p.cp_id_produto GROUP BY
p.cp_id_produto), produto_vendido_forn AS (SELECT f.cp_id_produto,
SUM(f.preco_venda_forn) AS total_vendas_forn FROM forn_produto f GROUP
BY f.cp_id_produto), produto_union_vendas AS (SELECT cp_id_produto,
total_vendas_estab AS total_vendas FROM produto_vendido_estab UNION ALL
SELECT cp_id_produto, total_vendas_forn AS total_vendas FROM
produto_vendido_forn) SELECT p.cp_id_produto, p.nm_produto,
c.nm_categoria, SUM(u.total_vendas) OVER (PARTITION BY
p.ce_categoria_principal) AS total_vendas_categoria, u.total_vendas
FROM produto_union_vendas u JOIN tbl_produto p ON u.cp_id_produto =
p.cp_id_produto LEFT JOIN tbl_categoria c ON p.ce_categoria_principal =
c.cp_cod_categoria WHERE u.total_vendas > (SELECT AVG(total_vendas)
FROM produto_union_vendas) ORDER BY total_vendas_categoria DESC,
u.total_vendas DESC;",
"WITH vendas_por_estabelecimento AS (SELECT e.cp_cod_estab,
e.cp_id_produto, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e GROUP BY e.cp_cod_estab, e.cp_id_produto),
vendas_por_fornecedor AS (SELECT f.cp_cod_forn AS cp_cod_estab,
f.cp_id_produto, SUM(f.preco_venda_forn) AS total_vendas_forn FROM
forn_produto f GROUP BY f.cp_cod_forn, f.cp_id_produto),
vendas_combinadas AS (SELECT cp_cod_estab, cp_id_produto,
total_vendas_estab AS total_vendas FROM vendas_por_estabelecimento
UNION ALL SELECT cp_cod_estab, cp_id_produto, total_vendas_forn AS
total_vendas FROM vendas_por_fornecedor) SELECT e.nm_estab,
c.nm_categoria, p.nm_produto, v.total_vendas, RANK() OVER (PARTITION BY
e.nm_estab ORDER BY v.total_vendas DESC) AS ranking_por_estab FROM
vendas_combinadas v JOIN tbl_estabelecimento e ON v.cp_cod_estab =
e.cp_cod_estab JOIN tbl_produto p ON v.cp_id_produto = p.cp_id_produto
JOIN tbl_categoria c ON p.ce_categoria_principal = c.cp_cod_categoria
```

```
WHERE v.total_vendas > (SELECT AVG(total_vendas) FROM
vendas_combinadas) GROUP BY e.nm_estab, c.nm_categoria, p.nm_produto,
v.total_vendas ORDER BY e.nm_estab, ranking_por_estab;",
"WITH vendas_por_produto AS (SELECT p.cp_id_produto,
SUM(e.preco_venda_estab) AS total_vendas_estab, SUM(f.preco_venda_forn)
AS total_vendas_forn FROM tbl_produto p LEFT JOIN estab_produto e ON
p.cp_id_produto = e.cp_id_produto LEFT JOIN forn_produto f ON
p.cp_id_produto = f.cp_id_produto GROUP BY p.cp_id_produto),
total_vendas_comb AS (SELECT cp_id_produto,
COALESCE(total_vendas_estab, 0) + COALESCE(total_vendas_forn, 0) AS
total_vendas FROM vendas_por_produto), vendas_totais_sistema AS (SELECT
SUM(total_vendas) AS vendas_totais_sistema FROM total_vendas_comb)
SELECT c.nm_categoria, COUNT(DISTINCT p.cp_id_produto) AS qtd_produtos,
AVG(t.total_vendas) AS media_vendas_categoria, (SELECT
vendas_totais_sistema FROM vendas_totais_sistema) AS
vendas_totais_sistema FROM total_vendas_comb t JOIN tbl_produto p ON
t.cp_id_produto = p.cp_id_produto JOIN tbl_categoria c ON
p.ce_categoria_principal = c.cp_cod_categoria WHERE t.total_vendas >
(SELECT AVG(total_vendas) FROM total_vendas_comb) GROUP BY
c.nm_categoria ORDER BY media_vendas_categoria DESC;",
"WITH vendas_por_estabelecimento AS (SELECT e.cp_cod_estab,
e.cp_id_produto, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e GROUP BY e.cp_cod_estab, e.cp_id_produto),
vendas_por_fornecedor AS (SELECT f.cp_cod_forn AS cp_cod_estab,
f.cp_id_produto, SUM(f.preco_venda_forn) AS total_vendas_forn FROM
forn_produto f GROUP BY f.cp_cod_forn, f.cp_id_produto),
vendas_combinadas AS (SELECT cp_cod_estab, cp_id_produto,
total_vendas_estab AS total_vendas FROM vendas_por_estabelecimento
UNION ALL SELECT cp_cod_estab, cp_id_produto, total_vendas_forn AS
total_vendas FROM vendas_por_fornecedor), vendas_por_uf AS (SELECT
e.uf_estab, SUM(v.total_vendas) AS total_vendas_uf FROM
vendas_combinadas v JOIN tbl_estabelecimento e ON v.cp_cod_estab =
e.cp_cod_estab GROUP BY e.uf_estab) SELECT e.uf_estab, e.cidade_estab,
COUNT(DISTINCT v.cp_id_produto) AS qtd_produtos_vendidos,
SUM(v.total_vendas) AS total_vendas_cidade, RANK() OVER (PARTITION BY
e.uf_estab ORDER BY SUM(v.total_vendas) DESC) AS ranking_cidade_uf FROM
vendas_combinadas v JOIN tbl_estabelecimento e ON v.cp_cod_estab =
e.cp_cod_estab WHERE v.total_vendas > (SELECT AVG(total_vendas) FROM
vendas_combinadas) GROUP BY e.uf_estab, e.cidade_estab ORDER BY
e.uf_estab, ranking_cidade_uf;",
"WITH vendas_estab AS (SELECT e.cp_id_produto, SUM(e.preco_venda_estab)
AS total_vendas_estab FROM estab_produto e GROUP BY e.cp_id_produto),
vendas_forn AS (SELECT f.cp_id_produto, SUM(f.preco_venda_forn) AS
```

```
total_vendas_forn FROM forn_produto f GROUP BY f.cp_id_produto),
vendas_comb AS (SELECT cp_id_produto, COALESCE(total_vendas_estab, 0) +
COALESCE(total_vendas_forn, 0) AS total_vendas FROM vendas_estab FULL
OUTER JOIN vendas_forn USING (cp_id_produto)), categorias_vendas AS
(SELECT c.nm_categoria, COUNT(p.cp_id_produto) AS qtd_produtos,
SUM(v.total_vendas) AS total_vendas_categoria FROM tbl_produto p JOIN
vendas_comb v ON p.cp_id_produto = v.cp_id_produto JOIN tbl_categoria c
ON p.ce_categoria_principal = c.cp_cod_categoria GROUP BY
c.nm_categoria), categoria_ranked AS (SELECT nm_categoria,
qtd_produtos, total_vendas_categoria, RANK() OVER (ORDER BY
total_vendas_categoria DESC) AS rank_vendas FROM categorias_vendas)
SELECT cr.nm_categoria, cr.qtd_produtos, cr.total_vendas_categoria,
cr.rank_vendas, AVG(v.total_vendas) AS media_vendas_produtos_categoria
FROM categoria_ranked cr JOIN tbl_produto p ON p.ce_categoria_principal
= (SELECT cp_cod_categoria FROM tbl_categoria WHERE nm_categoria =
cr.nm_categoria LIMIT 1) JOIN vendas_comb v ON p.cp_id_produto =
v.cp_id_produto WHERE cr.rank_vendas <= 3 GROUP BY cr.nm_categoria,
cr.qtd_produtos, cr.total_vendas_categoria, cr.rank_vendas ORDER BY
cr.rank_vendas;",
"WITH vendas_por_estabelecimento AS (SELECT p.cp_id_produto,
e.cp_cod_estab, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e JOIN tbl_produto p ON e.cp_id_produto = p.cp_id_produto
GROUP BY p.cp_id_produto, e.cp_cod_estab), vendas_por_fornecedor AS
(SELECT p.cp_id_produto, f.cp_cod_forn, SUM(f.preco_venda_forn) AS
total_vendas_forn FROM forn_produto f JOIN tbl_produto p ON
f.cp_id_produto = p.cp_id_produto GROUP BY p.cp_id_produto,
f.cp_cod_forn), vendas_combinadas AS (SELECT cp_id_produto,
cp_cod_estab AS origem, total_vendas_estab AS total_vendas FROM
vendas_por_estabelecimento UNION ALL SELECT cp_id_produto, cp_cod_forn
AS origem, total_vendas_forn AS total_vendas FROM
vendas_por_fornecedor), produtos_vendas_totais AS (SELECT
p.cp_id_produto, p.nm_produto, c.nm_categoria, SUM(vc.total_vendas) AS
total_vendas_produto, COUNT(DISTINCT vc.origem) AS qtd_origens FROM
vendas_combinadas vc JOIN tbl_produto p ON vc.cp_id_produto =
p.cp_id_produto JOIN tbl_categoria c ON p.ce_categoria_principal =
c.cp_cod_categoria GROUP BY p.cp_id_produto, p.nm_produto,
c.nm_categoria), ranking_produtos AS (SELECT nm_produto, nm_categoria,
total_vendas_produto, qtd_origens, RANK() OVER (PARTITION BY
nm_categoria ORDER BY total_vendas_produto DESC) AS rank_categoria FROM
produtos_vendas_totais) SELECT rp.nm_produto, rp.nm_categoria,
rp.total_vendas_produto, rp.qtd_origens, rp.rank_categoria,
AVG(rp.total_vendas_produto) OVER () AS media_vendas_geral FROM
```

```
ranking_produtos rp WHERE rp.rank_categoria <= 5 ORDER BY
rp.nm_categoria, rp.rank_categoria;",
"WITH vendas_estabelecimentos AS (SELECT e.cp_cod_estab,
e.cp_id_produto, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e GROUP BY e.cp_cod_estab, e.cp_id_produto),
vendas_fornecedores AS (SELECT f.cp_cod_forn, f.cp_id_produto,
SUM(f.preco_venda_forn) AS total_vendas_forn FROM forn_produto f GROUP
BY f.cp_cod_forn, f.cp_id_produto), vendas_combinadas AS (SELECT
p.cp_id_produto, COALESCE(e.cp_cod_estab, f.cp_cod_forn) AS origem,
COALESCE(e.total_vendas_estab, 0) + COALESCE(f.total_vendas_forn, 0) AS
total_vendas FROM vendas_estabelecimentos e FULL OUTER JOIN
vendas_fornecedores f USING (cp_id_produto) JOIN tbl_produto p ON
p.cp_id_produto = COALESCE(e.cp_id_produto, f.cp_id_produto)),
vendas_por_estado AS (SELECT COALESCE(e.uf_estab, f.uf_forn) AS uf,
vc.cp_id_produto, SUM(vc.total_vendas) AS total_vendas_estado FROM
vendas_combinadas vc LEFT JOIN tbl_estabelecimento e ON vc.origem =
e.cp_cod_estab LEFT JOIN tbl_fornecedor f ON vc.origem = f.cp_cod_forn
GROUP BY COALESCE(e.uf_estab, f.uf_forn), vc.cp_id_produto),
ranking_produtos_estado AS (SELECT vp.uf, p.nm_produto,
SUM(vp.total_vendas_estado) AS vendas_produto_estado, RANK() OVER
(PARTITION BY vp.uf ORDER BY SUM(vp.total_vendas_estado) DESC) AS
rank_produto_estado FROM vendas_por_estado vp JOIN tbl_produto p ON
vp.cp_id_produto = p.cp_id_produto GROUP BY vp.uf, p.nm_produto) SELECT
rpe.uf, rpe.nm_produto, rpe.vendas_produto_estado,
rpe.rank_produto_estado, (SELECT AVG(vendas_produto_estado) FROM
ranking_produtos_estado) AS media_vendas_geral FROM
ranking_produtos_estado rpe WHERE rpe.rank_produto_estado <= 3 ORDER BY
rpe.uf, rpe.rank_produto_estado;",
"WITH vendas_estabelecimentos AS (SELECT e.cp_cod_estab AS origem,
p.ce_categoria_principal AS categoria, COUNT(DISTINCT e.cp_id_produto)
AS qtd_produtos_estab, SUM(e.preco_venda_estab) AS total_vendas_estab
FROM estab_produto e JOIN tbl_produto p ON e.cp_id_produto =
p.cp_id_produto GROUP BY e.cp_cod_estab, p.ce_categoria_principal),
vendas_fornecedores AS (SELECT f.cp_cod_forn AS origem,
p.ce_categoria_principal AS categoria, COUNT(DISTINCT f.cp_id_produto)
AS qtd_produtos_forn, SUM(f.preco_venda_forn) AS total_vendas_forn FROM
forn_produto f JOIN tbl_produto p ON f.cp_id_produto = p.cp_id_produto
GROUP BY f.cp_cod_forn, p.ce_categoria_principal), vendas_combinadas AS
(SELECT origem, categoria, COALESCE(qtd_produtos_estab, 0) +
COALESCE(qtd_produtos_forn, 0) AS qtd_produtos,
COALESCE(total_vendas_estab, 0) + COALESCE(total_vendas_forn, 0) AS
total_vendas FROM vendas_estabelecimentos FULL OUTER JOIN
vendas_fornecedores USING (origem, categoria)), ranking_categorias AS
```

```
(SELECT vc.categoria, c.nm_categoria, vc.origem, vc.qtd_produtos,
vc.total_vendas, RANK() OVER (PARTITION BY vc.categoria ORDER BY
vc.total_vendas DESC) AS rank_vendas_categoria FROM vendas_combinadas
vc JOIN tbl_categoria c ON vc.categoria = c.cp_cod_categoria) SELECT
r.nm_categoria, r.origem, r.qtd_produtos, r.total_vendas,
r.rank_vendas_categoria, (SELECT AVG(total_vendas) FROM
ranking_categorias) AS media_vendas_geral FROM ranking_categorias r
WHERE r.rank_vendas_categoria <= 3 ORDER BY r.nm_categoria,
r.rank_vendas_categoria;",
"WITH preco_produto_fornecedor AS (SELECT f.nm_forn, p.nm_produto,
e.preco_venda_estab, f.cp_cod_forn, p.cp_id_produto, ROW_NUMBER() OVER
(PARTITION BY p.cp_id_produto ORDER BY e.preco_venda_estab DESC) AS
rank_produto FROM estab_produto e JOIN tbl_produto p ON e.cp_id_produto
= p.cp_id_produto JOIN tbl_fornecedor f ON e.cp_cod_estab =
f.cp_cod_forn), produtos_vendidos_comparacao AS (SELECT ppf.nm_forn,
ppf.nm_produto, ppf.preco_venda_estab, (SELECT AVG(preco_venda_estab)
FROM estab_produto WHERE cp_id_produto = ppf.cp_id_produto) AS
preco_medio_produto FROM preco_produto_fornecedor ppf WHERE
ppf.rank_produto = 1), vendas_comparadas AS (SELECT nm_forn,
nm_produto, preco_venda_estab, preco_medio_produto, (preco_venda_estab
- preco_medio_produto) AS diferenca_preco FROM
produtos_vendidos_comparacao) SELECT nm_forn, nm_produto,
preco_venda_estab, preco_medio_produto, diferenca_preco, COUNT(*) OVER
(PARTITION BY nm_forn) AS quantidade_fornecimentos FROM
vendas_comparadas ORDER BY diferenca_preco DESC;",
"SELECT p.cp_id_pedido, p.ce_cod_cliente, p.data_pedido, p.hora_pedido,
p.valor_total, SUM(ep.preco_venda_estab) AS preco_venda_total,
ROW_NUMBER() OVER (PARTITION BY p.ce_cod_cliente ORDER BY p.data_pedido
DESC) AS rank_pedido FROM tbl_pedido p JOIN pedido_produto pp ON
pp.cp_id_pedido = p.cp_id_pedido JOIN estab_produto ep ON
ep.cp_id_produto = pp.cp_id_produto WHERE p.data_pedido BETWEEN
'2025-01-01' AND '2025-12-31' GROUP BY p.cp_id_pedido,
p.ce_cod_cliente, p.data_pedido, p.hora_pedido, p.valor_total UNION
SELECT p.cp_id_pedido, p.ce_cod_cliente, p.data_pedido, p.hora_pedido,
p.valor_total, 0 AS preco_venda_total, NULL AS rank_pedido FROM
tbl_pedido p WHERE p.ce_cod_cliente IN (SELECT cp_cod_cliente FROM
tbl_cliente WHERE cpf_cliente = '12345678901') ORDER BY data_pedido
DESC;"
]

# Dicionário para armazenar os tempos de execução
execution_times = {query: [] for query in queries}
```

```
# Executar as queries 50 vezes
for _ in range(50):
    for query in queries:
        start_time = time.time()
        with conn.cursor() as cursor:
            cursor.execute(query)
        end_time = time.time()
        execution_times[query].append(end_time - start_time)

# Fechar a conexão
conn.close()

# Criar um DataFrame com os tempos de execução
df = pd.DataFrame(execution_times)
df.to_csv("baseline.csv", index=False)
```

Com indexação:

-- PLANO DE INDEXAÇÃO --
-- Esses índices ajudarão a reduzir o tempo de execução das queries mais utilizadas
CREATE INDEX idx_cp_id_pedido ON pedido_produto (cp_id_pedido);
CREATE INDEX idx_data_venda_estab ON estab_produto (data_venda_estab);
CREATE INDEX idx_preco_venda_estab ON estab_produto (preco_venda_estab);
CREATE INDEX idx_cp_cod_cliente ON tbl_cliente (cp_cod_cliente);
CREATE INDEX idx_cd_ean_prod ON tbl_produto (cd_ean_prod);
CREATE INDEX idx_ce_cod_cliente ON tbl_pedido (ce_cod_cliente);

-- Esses índices ajudarão a reduzir o tempo de execução das queries de consulta com múltiplos JOINs
CREATE INDEX idx_tbl_produto_ce_categoria_secundaria ON tbl_produto (ce_categoria_secundaria);
CREATE INDEX idx_tbl_categoria_cp_cod_categoria ON tbl_categoria (cp_cod_categoria);
CREATE INDEX idx_tbl_fornecedor_cp_cod_forn ON tbl_fornecedor (cp_cod_forn);
CREATE INDEX idx_forn_produto_data_vencimento ON forn_produto (data_vencimento);
CREATE INDEX idx_tbl_estabelecimento_localizacao_estab ON tbl_estabelecimento (localizacao_estab);
CREATE INDEX idx_forn_produto_composto ON forn_produto (cp_cod_forn, cp_id_produto);
CREATE INDEX idx_estab_produto_composto ON estab_produto (cp_cod_estab, cp_id_produto);

-- Esses índices ajudarão a reduzir o tempo de execução das queries que envolvem múltiplas agregações e ordenações.
CREATE INDEX idx_estab_localizacao ON tbl_estabelecimento (cp_cod_estab, uf_estab, cidade_estab);

CREATE INDEX idx_produto_vendas_total ON estab_produto (cp_id_produto, preco_venda_estab);
CREATE INDEX idx_produto_fornecedor_vendas ON forn_produto (cp_id_produto, preco_venda_forn);
CREATE INDEX idx_estab_vendas_estado ON estab_produto (cp_cod_estab, preco_venda_estab);
CREATE INDEX idx_forn_vendas_estado ON forn_produto (cp_cod_forn, preco_venda_forn);

Na mesmo diretório em que foi feito o primeiro script e a baseline, execute outro script com o seguinte código, para obter a tabela com os tempos de execução do banco já indexado e a tabela de comparação com a baseline (speedup).

```
import psycopg2
import time
import pandas as pd

# Conexão com o banco de dados
conn = psycopg2.connect(
    dbname="teste",
    user="postgres",
    password="a",
    host="localhost",
    port="5432"
)

# Lista de queries
queries = [
"SELECT * FROM tbl_rfid;",
"SELECT * FROM tbl_categoria;",
"SELECT * FROM tbl_cliente;",
"SELECT * FROM tbl_estabelecimento;",
"SELECT * FROM tbl_fornecedor;",
"SELECT * FROM tbl_produto;",
"SELECT * FROM tbl_funcionario;",
"SELECT * FROM tbl_pedido;",
"SELECT * FROM pedido_produto WHERE cp_id_pedido = 1;",
"SELECT * FROM estab_produto WHERE cp_cod_estab = 1;",
"SELECT * FROM forn_produto WHERE cp_cod_forn = 1;",
"SELECT * FROM tbl_cliente WHERE cp_cod_cliente = 1;",
"SELECT * FROM tbl_produto WHERE cp_id_produto = 1;",
"SELECT * FROM tbl_pedido WHERE ce_cod_cliente = 1;",
"SELECT * FROM tbl_produto WHERE ce_categoria_principal = 1;",
"SELECT * FROM estab_produto WHERE data_venda_estab = '2025-01-18';",
```

```
"SELECT * FROM tbl_estabelecimento WHERE cidade_estab = 'CDD';",
"SELECT * FROM estab_produto WHERE preco_venda_estab > 2.00;",
"SELECT * FROM tbl_funcionario WHERE ce_cod_estab = 1;",
"SELECT * FROM tbl_produto WHERE cd_ean_prod = '000000000001';"
"SELECT p.cp_id_pedido, p.data_pedido, p.hora_pedido, p.valor_total,
c.nm_cliente FROM tbl_pedido p JOIN tbl_cliente c ON p.ce_cod_cliente =
c.cp_cod_cliente;",
"SELECT pr.cp_id_produto, pr.nm_produto, c.nm_categoria FROM
tbl_produto pr JOIN tbl_categoria c ON pr.ce_categoria_principal =
c.cp_cod_categoria;",
"SELECT f.nm_forn, pr.nm_produto FROM forn_produto fp JOIN tbl_produto
pr ON fp.cp_id_produto = pr.cp_id_produto JOIN tbl_fornecedor f ON
fp.cp_cod_forn = f.cp_cod_forn;",
"SELECT e.nm_estab, pr.nm_produto FROM estab_produto ep JOIN
tbl_produto pr ON ep.cp_id_produto = pr.cp_id_produto JOIN
tbl_estabelecimento e ON ep.cp_cod_estab = e.cp_cod_estab;",
"SELECT p.cp_id_pedido, pr.nm_produto FROM pedido_produto pp JOIN
tbl_produto pr ON pp.cp_id_produto = pr.cp_id_produto JOIN tbl_pedido p
ON pp.cp_id_pedido = p.cp_id_pedido;",
"SELECT f.nm_func, e.nm_estab FROM tbl_funcionario f JOIN
tbl_estabelecimento e ON f.ce_cod_estab = e.cp_cod_estab;",
"SELECT p.cp_id_pedido, p.valor_total, c.nm_cliente FROM tbl_pedido p
JOIN tbl_cliente c ON p.ce_cod_cliente = c.cp_cod_cliente;",
"SELECT c.nm_categoria, pr.nm_produto FROM tbl_produto pr JOIN
tbl_categoria c ON pr.ce_categoria_secundaria = c.cp_cod_categoria;",
"SELECT f.nm_forn, pr.nm_produto, fp.preco_venda_forn FROM forn_produto
fp JOIN tbl_produto pr ON fp.cp_id_produto = pr.cp_id_produto JOIN
tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn;",
"SELECT pr.nm_produto, e.nm_estab, ep.preco_venda_estab FROM
estab_produto ep JOIN tbl_produto pr ON ep.cp_id_produto =
pr.cp_id_produto JOIN tbl_estabelecimento e ON ep.cp_cod_estab =
e.cp_cod_estab;",
"SELECT pr.nm_produto, f.nm_forn, fp.preco_venda_forn,
fp.data_vencimento FROM forn_produto fp JOIN tbl_produto pr ON
fp.cp_id_produto = pr.cp_id_produto JOIN tbl_fornecedor f ON
fp.cp_cod_forn = f.cp_cod_forn;",
"SELECT f.nm_func, e.nm_estab, e.cidade_estab FROM tbl_funcionario f
JOIN tbl_estabelecimento e ON f.ce_cod_estab = e.cp_cod_estab;",
"SELECT f.nm_forn, pr.nm_produto, e.nm_estab FROM forn_produto fp JOIN
tbl_produto pr ON fp.cp_id_produto = pr.cp_id_produto JOIN
tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn JOIN estab_produto
ep ON pr.cp_id_produto = ep.cp_id_produto JOIN tbl_estabelecimento e ON
ep.cp_cod_estab = e.cp_cod_estab;",
```

```
"SELECT e.nm_estab, e.localizacao_estab, f.nm_forn, f.localizacao_forn
FROM tbl_estabelecimento e JOIN forn_produto fp ON e.cp_cod_estab =
fp.cp_cod_forn JOIN tbl_fornecedor f ON fp.cp_cod_forn =
f.cp_cod_forn;",
"SELECT p.cp_id_pedido, c.nm_cliente, p.data_pedido, p.hora_pedido FROM
tbl_pedido p JOIN tbl_cliente c ON p.ce_cod_cliente =
c.cp_cod_cliente;"
"WITH produto_vendido_estab AS (SELECT p.cp_id_produto,
SUM(e.preco_venda_estab) AS total_vendas_estab FROM estab_produto e
JOIN tbl_produto p ON e.cp_id_produto = p.cp_id_produto GROUP BY
p.cp_id_produto), produto_vendido_forn AS (SELECT f.cp_id_produto,
SUM(f.preco_venda_forn) AS total_vendas_forn FROM forn_produto f GROUP
BY f.cp_id_produto), produto_union_vendas AS (SELECT cp_id_produto,
total_vendas_estab AS total_vendas FROM produto_vendido_estab UNION ALL
SELECT cp_id_produto, total_vendas_forn AS total_vendas FROM
produto_vendido_forn) SELECT p.cp_id_produto, p.nm_produto,
c.nm_categoria, SUM(u.total_vendas) OVER (PARTITION BY
p.ce_categoria_principal) AS total_vendas_categoria, u.total_vendas
FROM produto_union_vendas u JOIN tbl_produto p ON u.cp_id_produto =
p.cp_id_produto LEFT JOIN tbl_categoria c ON p.ce_categoria_principal =
c.cp_cod_categoria WHERE u.total_vendas > (SELECT AVG(total_vendas)
FROM produto_union_vendas) ORDER BY total_vendas_categoria DESC,
u.total_vendas DESC;",
"WITH vendas_por_estabelecimento AS (SELECT e.cp_cod_estab,
e.cp_id_produto, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e GROUP BY e.cp_cod_estab, e.cp_id_produto),
vendas_por_fornecedor AS (SELECT f.cp_cod_forn AS cp_cod_estab,
f.cp_id_produto, SUM(f.preco_venda_forn) AS total_vendas_forn FROM
forn_produto f GROUP BY f.cp_cod_forn, f.cp_id_produto),
vendas_combinadas AS (SELECT cp_cod_estab, cp_id_produto,
total_vendas_estab AS total_vendas FROM vendas_por_estabelecimento
UNION ALL SELECT cp_cod_estab, cp_id_produto, total_vendas_forn AS
total_vendas FROM vendas_por_fornecedor) SELECT e.nm_estab,
c.nm_categoria, p.nm_produto, v.total_vendas, RANK() OVER (PARTITION BY
e.nm_estab ORDER BY v.total_vendas DESC) AS ranking_por_estab FROM
vendas_combinadas v JOIN tbl_estabelecimento e ON v.cp_cod_estab =
e.cp_cod_estab JOIN tbl_produto p ON v.cp_id_produto = p.cp_id_produto
JOIN tbl_categoria c ON p.ce_categoria_principal = c.cp_cod_categoria
WHERE v.total_vendas > (SELECT AVG(total_vendas) FROM
vendas_combinadas) GROUP BY e.nm_estab, c.nm_categoria, p.nm_produto,
v.total_vendas ORDER BY e.nm_estab, ranking_por_estab;",
"WITH vendas_por_produto AS (SELECT p.cp_id_produto,
SUM(e.preco_venda_estab) AS total_vendas_estab, SUM(f.preco_venda_forn)
```

```
AS total_vendas_forn FROM tbl_produto p LEFT JOIN estab_produto e ON
p.cp_id_produto = e.cp_id_produto LEFT JOIN forn_produto f ON
p.cp_id_produto = f.cp_id_produto GROUP BY p.cp_id_produto),
total_vendas_comb AS (SELECT cp_id_produto,
COALESCE(total_vendas_estab, 0) + COALESCE(total_vendas_forn, 0) AS
total_vendas FROM vendas_por_produto), vendas_totais_sistema AS (SELECT
SUM(total_vendas) AS vendas_totais_sistema FROM total_vendas_comb)
SELECT c.nm_categoria, COUNT(DISTINCT p.cp_id_produto) AS qtd_produtos,
AVG(t.total_vendas) AS media_vendas_categoria, (SELECT
vendas_totais_sistema FROM vendas_totais_sistema) AS
vendas_totais_sistema FROM total_vendas_comb t JOIN tbl_produto p ON
t.cp_id_produto = p.cp_id_produto JOIN tbl_categoria c ON
p.ce_categoria_principal = c.cp_cod_categoria WHERE t.total_vendas >
(SELECT AVG(total_vendas) FROM total_vendas_comb) GROUP BY
c.nm_categoria ORDER BY media_vendas_categoria DESC;",
"WITH vendas_por_estabelecimento AS (SELECT e.cp_cod_estab,
e.cp_id_produto, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e GROUP BY e.cp_cod_estab, e.cp_id_produto),
vendas_por_fornecedor AS (SELECT f.cp_cod_forn AS cp_cod_estab,
f.cp_id_produto, SUM(f.preco_venda_forn) AS total_vendas_forn FROM
forn_produto f GROUP BY f.cp_cod_forn, f.cp_id_produto),
vendas_combinadas AS (SELECT cp_cod_estab, cp_id_produto,
total_vendas_estab AS total_vendas FROM vendas_por_estabelecimento
UNION ALL SELECT cp_cod_estab, cp_id_produto, total_vendas_forn AS
total_vendas FROM vendas_por_fornecedor), vendas_por_uf AS (SELECT
e.uf_estab, SUM(v.total_vendas) AS total_vendas_uf FROM
vendas_combinadas v JOIN tbl_estabelecimento e ON v.cp_cod_estab =
e.cp_cod_estab GROUP BY e.uf_estab) SELECT e.uf_estab, e.cidade_estab,
COUNT(DISTINCT v.cp_id_produto) AS qtd_produtos_vendidos,
SUM(v.total_vendas) AS total_vendas_cidade, RANK() OVER (PARTITION BY
e.uf_estab ORDER BY SUM(v.total_vendas) DESC) AS ranking_cidade_uf FROM
vendas_combinadas v JOIN tbl_estabelecimento e ON v.cp_cod_estab =
e.cp_cod_estab WHERE v.total_vendas > (SELECT AVG(total_vendas) FROM
vendas_combinadas) GROUP BY e.uf_estab, e.cidade_estab ORDER BY
e.uf_estab, ranking_cidade_uf;",
"WITH vendas_estab AS (SELECT e.cp_id_produto, SUM(e.preco_venda_estab)
AS total_vendas_estab FROM estab_produto e GROUP BY e.cp_id_produto),
vendas_forn AS (SELECT f.cp_id_produto, SUM(f.preco_venda_forn) AS
total_vendas_forn FROM forn_produto f GROUP BY f.cp_id_produto),
vendas_comb AS (SELECT cp_id_produto, COALESCE(total_vendas_estab, 0) +
COALESCE(total_vendas_forn, 0) AS total_vendas FROM vendas_estab FULL
OUTER JOIN vendas_forn USING (cp_id_produto)), categorias_vendas AS
(SELECT c.nm_categoria, COUNT(p.cp_id_produto) AS qtd_produtos,
```

```
SUM(v.total_vendas) AS total_vendas_categoria FROM tbl_produto p JOIN
vendas_comb v ON p.cp_id_produto = v.cp_id_produto JOIN tbl_categoria c
ON p.ce_categoria_principal = c.cp_cod_categoria GROUP BY
c.nm_categoria), categoria_ranked AS (SELECT nm_categoria,
qtd_produtos, total_vendas_categoria, RANK() OVER (ORDER BY
total_vendas_categoria DESC) AS rank_vendas FROM categorias_vendas)
SELECT cr.nm_categoria, cr.qtd_produtos, cr.total_vendas_categoria,
cr.rank_vendas, AVG(v.total_vendas) AS media_vendas_produtos_categoria
FROM categoria_ranked cr JOIN tbl_produto p ON p.ce_categoria_principal
= (SELECT cp_cod_categoria FROM tbl_categoria WHERE nm_categoria =
cr.nm_categoria LIMIT 1) JOIN vendas_comb v ON p.cp_id_produto =
v.cp_id_produto WHERE cr.rank_vendas <= 3 GROUP BY cr.nm_categoria,
cr.qtd_produtos, cr.total_vendas_categoria, cr.rank_vendas ORDER BY
cr.rank_vendas;",
"WITH vendas_por_estabelecimento AS (SELECT p.cp_id_produto,
e.cp_cod_estab, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e JOIN tbl_produto p ON e.cp_id_produto = p.cp_id_produto
GROUP BY p.cp_id_produto, e.cp_cod_estab), vendas_por_fornecedor AS
(SELECT p.cp_id_produto, f.cp_cod_forn, SUM(f.preco_venda_forn) AS
total_vendas_forn FROM forn_produto f JOIN tbl_produto p ON
f.cp_id_produto = p.cp_id_produto GROUP BY p.cp_id_produto,
f.cp_cod_forn), vendas_combinadas AS (SELECT cp_id_produto,
cp_cod_estab AS origem, total_vendas_estab AS total_vendas FROM
vendas_por_estabelecimento UNION ALL SELECT cp_id_produto, cp_cod_forn
AS origem, total_vendas_forn AS total_vendas FROM
vendas_por_fornecedor), produtos_vendas_totais AS (SELECT
p.cp_id_produto, p.nm_produto, c.nm_categoria, SUM(vc.total_vendas) AS
total_vendas_produto, COUNT(DISTINCT vc.origem) AS qtd_origens FROM
vendas_combinadas vc JOIN tbl_produto p ON vc.cp_id_produto =
p.cp_id_produto JOIN tbl_categoria c ON p.ce_categoria_principal =
c.cp_cod_categoria GROUP BY p.cp_id_produto, p.nm_produto,
c.nm_categoria), ranking_produtos AS (SELECT nm_produto, nm_categoria,
total_vendas_produto, qtd_origens, RANK() OVER (PARTITION BY
nm_categoria ORDER BY total_vendas_produto DESC) AS rank_categoria FROM
produtos_vendas_totais) SELECT rp.nm_produto, rp.nm_categoria,
rp.total_vendas_produto, rp.qtd_origens, rp.rank_categoria,
AVG(rp.total_vendas_produto) OVER () AS media_vendas_geral FROM
ranking_produtos rp WHERE rp.rank_categoria <= 5 ORDER BY
rp.nm_categoria, rp.rank_categoria;",
"WITH vendas_estabelecimentos AS (SELECT e.cp_cod_estab,
e.cp_id_produto, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e GROUP BY e.cp_cod_estab, e.cp_id_produto),
vendas_fornecedores AS (SELECT f.cp_cod_forn, f.cp_id_produto,
```

```
SUM(f.preco_venda_forn) AS total_vendas_forn FROM forn_produto f GROUP
BY f.cp_cod_forn, f.cp_id_produto), vendas_combinadas AS (SELECT
p.cp_id_produto, COALESCE(e.cp_cod_estab, f.cp_cod_forn) AS origem,
COALESCE(e.total_vendas_estab, 0) + COALESCE(f.total_vendas_forn, 0) AS
total_vendas FROM vendas_estabelecimentos e FULL OUTER JOIN
vendas_fornecedores f USING (cp_id_produto) JOIN tbl_produto p ON
p.cp_id_produto = COALESCE(e.cp_id_produto, f.cp_id_produto)),
vendas_por_estado AS (SELECT COALESCE(e.uf_estab, f.uf_forn) AS uf,
vc.cp_id_produto, SUM(vc.total_vendas) AS total_vendas_estado FROM
vendas_combinadas vc LEFT JOIN tbl_estabelecimento e ON vc.origem =
e.cp_cod_estab LEFT JOIN tbl_fornecedor f ON vc.origem = f.cp_cod_forn
GROUP BY COALESCE(e.uf_estab, f.uf_forn), vc.cp_id_produto),
ranking_produtos_estado AS (SELECT vp.uf, p.nm_produto,
SUM(vp.total_vendas_estado) AS vendas_produto_estado, RANK() OVER
(PARTITION BY vp.uf ORDER BY SUM(vp.total_vendas_estado) DESC) AS
rank_produto_estado FROM vendas_por_estado vp JOIN tbl_produto p ON
vp.cp_id_produto = p.cp_id_produto GROUP BY vp.uf, p.nm_produto) SELECT
rpe.uf, rpe.nm_produto, rpe.vendas_produto_estado,
rpe.rank_produto_estado, (SELECT AVG(vendas_produto_estado) FROM
ranking_produtos_estado) AS media_vendas_geral FROM
ranking_produtos_estado rpe WHERE rpe.rank_produto_estado <= 3 ORDER BY
rpe.uf, rpe.rank_produto_estado;",
"WITH vendas_estabelecimentos AS (SELECT e.cp_cod_estab AS origem,
p.ce_categoria_principal AS categoria, COUNT(DISTINCT e.cp_id_produto)
AS qtd_produtos_estab, SUM(e.preco_venda_estab) AS total_vendas_estab
FROM estab_produto e JOIN tbl_produto p ON e.cp_id_produto =
p.cp_id_produto GROUP BY e.cp_cod_estab, p.ce_categoria_principal),
vendas_fornecedores AS (SELECT f.cp_cod_forn AS origem,
p.ce_categoria_principal AS categoria, COUNT(DISTINCT f.cp_id_produto)
AS qtd_produtos_forn, SUM(f.preco_venda_forn) AS total_vendas_forn FROM
forn_produto f JOIN tbl_produto p ON f.cp_id_produto = p.cp_id_produto
GROUP BY f.cp_cod_forn, p.ce_categoria_principal), vendas_combinadas AS
(SELECT origem, categoria, COALESCE(qtd_produtos_estab, 0) +
COALESCE(qtd_produtos_forn, 0) AS qtd_produtos,
COALESCE(total_vendas_estab, 0) + COALESCE(total_vendas_forn, 0) AS
total_vendas FROM vendas_estabelecimentos FULL OUTER JOIN
vendas_fornecedores USING (origem, categoria)), ranking_categorias AS
(SELECT vc.categoria, c.nm_categoria, vc.origem, vc.qtd_produtos,
vc.total_vendas, RANK() OVER (PARTITION BY vc.categoria ORDER BY
vc.total_vendas DESC) AS rank_vendas_categoria FROM vendas_combinadas
vc JOIN tbl_categoria c ON vc.categoria = c.cp_cod_categoria) SELECT
r.nm_categoria, r.origem, r.qtd_produtos, r.total_vendas,
r.rank_vendas_categoria, (SELECT AVG(total_vendas) FROM
```

```
ranking_categorias) AS media_vendas_geral FROM ranking_categorias r
WHERE r.rank_vendas_categoria <= 3 ORDER BY r.nm_categoria,
r.rank_vendas_categoria;",
"WITH preco_produto_fornecedor AS (SELECT f.nm_forn, p.nm_produto,
e.preco_venda_estab, f.cp_cod_forn, p.cp_id_produto, ROW_NUMBER() OVER
(PARTITION BY p.cp_id_produto ORDER BY e.preco_venda_estab DESC) AS
rank_produto FROM estab_produto e JOIN tbl_produto p ON e.cp_id_produto
= p.cp_id_produto JOIN tbl_fornecedor f ON e.cp_cod_estab =
f.cp_cod_forn), produtos_vendidos_comparacao AS (SELECT ppf.nm_forn,
ppf.nm_produto, ppf.preco_venda_estab, (SELECT AVG(preco_venda_estab)
FROM estab_produto WHERE cp_id_produto = ppf.cp_id_produto) AS
preco_medio_produto FROM preco_produto_fornecedor ppf WHERE
ppf.rank_produto = 1), vendas_comparadas AS (SELECT nm_forn,
nm_produto, preco_venda_estab, preco_medio_produto, (preco_venda_estab
- preco_medio_produto) AS diferenca_preco FROM
produtos_vendidos_comparacao) SELECT nm_forn, nm_produto,
preco_venda_estab, preco_medio_produto, diferenca_preco, COUNT(*) OVER
(PARTITION BY nm_forn) AS quantidade_fornecimentos FROM
vendas_comparadas ORDER BY diferenca_preco DESC;",
"SELECT p.cp_id_pedido, p.ce_cod_cliente, p.data_pedido, p.hora_pedido,
p.valor_total, SUM(ep.preco_venda_estab) AS preco_venda_total,
ROW_NUMBER() OVER (PARTITION BY p.ce_cod_cliente ORDER BY p.data_pedido
DESC) AS rank_pedido FROM tbl_pedido p JOIN pedido_produto pp ON
pp.cp_id_pedido = p.cp_id_pedido JOIN estab_produto ep ON
ep.cp_id_produto = pp.cp_id_produto WHERE p.data_pedido BETWEEN
'2025-01-01' AND '2025-12-31' GROUP BY p.cp_id_pedido,
p.ce_cod_cliente, p.data_pedido, p.hora_pedido, p.valor_total UNION
SELECT p.cp_id_pedido, p.ce_cod_cliente, p.data_pedido, p.hora_pedido,
p.valor_total, 0 AS preco_venda_total, NULL AS rank_pedido FROM
tbl_pedido p WHERE p.ce_cod_cliente IN (SELECT cp_cod_cliente FROM
tbl_cliente WHERE cpf_cliente = '12345678901') ORDER BY data_pedido
DESC;"
]

# Dicionário para armazenar os tempos de execução
execution_times_indexed = {query: [] for query in queries}

# Executar as queries 50 vezes
for _ in range(50):
    for query in queries:
        start_time = time.time()
        with conn.cursor() as cursor:
            cursor.execute(query)
```

```
        end_time = time.time()
        execution_times_indexed[query].append(end_time - start_time)

# Fechar a conexão
conn.close()

# Criar um DataFrame com os tempos de execução após a indexação
df_indexed = pd.DataFrame(execution_times_indexed)
df_indexed.to_csv("indexed.csv", index=False)

# Carregar a baseline
df_baseline = pd.read_csv("baseline.csv")

# Calcular o speedup
speedup = df_baseline.mean() / df_indexed.mean()

# Criar um DataFrame com os resultados
df_speedup = pd.DataFrame({
    'Query': queries,
    'Baseline Time': df_baseline.mean(),
    'Indexed Time': df_indexed.mean(),
    'Speedup': speedup
})

# Salvar a planilha de melhoria de desempenho
df_speedup.to_csv("performance_improvement.csv", index=False)
```

Com tunning:

Com o auxílio do site https://pgtune.leopard.in.ua/?dbVersion=17&osType=windows&dbType=desktop&cpuNum=&totalMemory=12&totalMemoryUnit=GB&connectionNum=&hdType=ssd, foi elaborado o seguinte plano de tunning para o banco de dados, editado manualmente no arquivo postgresql.conf:

DB Version: 17
OS Type: windows
DB Type: desktop
Total Memory (RAM): 12 GB
Data Storage: ssd

max_connections = 20
shared_buffers = 768MB
effective_cache_size = 3GB
maintenance_work_mem = 768MB

checkpoint_completion_target = 0.9
wal_buffers = 16MB
default_statistics_target = 100
random_page_cost = 1.1
work_mem = 16MB
huge_pages = off
min_wal_size = 100MB
max_wal_size = 2GB
wal_level = minimal
max_wal_senders = 0

No mesmo diretório das outras etapas, cria-se um terceiro script para comparar o desempenho da baseline com o código indexado e após o tunning:

```
import psycopg2
import time
import pandas as pd

# Conexão com o banco de dados
conn = psycopg2.connect(
    dbname="teste",
    user="postgres",
    password="a",
    host="localhost",
    port="5432"
)

# Lista de queries
queries = [
"SELECT * FROM tbl_rfid;",
"SELECT * FROM tbl_categoria;",
"SELECT * FROM tbl_cliente;",
"SELECT * FROM tbl_estabelecimento;",
"SELECT * FROM tbl_fornecedor;",
"SELECT * FROM tbl_produto;",
"SELECT * FROM tbl_funcionario;",
"SELECT * FROM tbl_pedido;",
"SELECT * FROM pedido_produto WHERE cp_id_pedido = 1;",
"SELECT * FROM estab_produto WHERE cp_cod_estab = 1;",
"SELECT * FROM forn_produto WHERE cp_cod_forn = 1;",
"SELECT * FROM tbl_cliente WHERE cp_cod_cliente = 1;",
"SELECT * FROM tbl_produto WHERE cp_id_produto = 1;",
"SELECT * FROM tbl_pedido WHERE ce_cod_cliente = 1;",
"SELECT * FROM tbl_produto WHERE ce_categoria_principal = 1;",
"SELECT * FROM estab_produto WHERE data_venda_estab = '2025-01-18';",
```

```
"SELECT * FROM tbl_estabelecimento WHERE cidade_estab = 'CDD';",
"SELECT * FROM estab_produto WHERE preco_venda_estab > 2.00;",
"SELECT * FROM tbl_funcionario WHERE ce_cod_estab = 1;",
"SELECT * FROM tbl_produto WHERE cd_ean_prod = '000000000001';"
"SELECT p.cp_id_pedido, p.data_pedido, p.hora_pedido, p.valor_total,
c.nm_cliente FROM tbl_pedido p JOIN tbl_cliente c ON p.ce_cod_cliente =
c.cp_cod_cliente;",
"SELECT pr.cp_id_produto, pr.nm_produto, c.nm_categoria FROM
tbl_produto pr JOIN tbl_categoria c ON pr.ce_categoria_principal =
c.cp_cod_categoria;",
"SELECT f.nm_forn, pr.nm_produto FROM forn_produto fp JOIN tbl_produto
pr ON fp.cp_id_produto = pr.cp_id_produto JOIN tbl_fornecedor f ON
fp.cp_cod_forn = f.cp_cod_forn;",
"SELECT e.nm_estab, pr.nm_produto FROM estab_produto ep JOIN
tbl_produto pr ON ep.cp_id_produto = pr.cp_id_produto JOIN
tbl_estabelecimento e ON ep.cp_cod_estab = e.cp_cod_estab;",
"SELECT p.cp_id_pedido, pr.nm_produto FROM pedido_produto pp JOIN
tbl_produto pr ON pp.cp_id_produto = pr.cp_id_produto JOIN tbl_pedido p
ON pp.cp_id_pedido = p.cp_id_pedido;",
"SELECT f.nm_func, e.nm_estab FROM tbl_funcionario f JOIN
tbl_estabelecimento e ON f.ce_cod_estab = e.cp_cod_estab;",
"SELECT p.cp_id_pedido, p.valor_total, c.nm_cliente FROM tbl_pedido p
JOIN tbl_cliente c ON p.ce_cod_cliente = c.cp_cod_cliente;",
"SELECT c.nm_categoria, pr.nm_produto FROM tbl_produto pr JOIN
tbl_categoria c ON pr.ce_categoria_secundaria = c.cp_cod_categoria;",
"SELECT f.nm_forn, pr.nm_produto, fp.preco_venda_forn FROM forn_produto
fp JOIN tbl_produto pr ON fp.cp_id_produto = pr.cp_id_produto JOIN
tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn;",
"SELECT pr.nm_produto, e.nm_estab, ep.preco_venda_estab FROM
estab_produto ep JOIN tbl_produto pr ON ep.cp_id_produto =
pr.cp_id_produto JOIN tbl_estabelecimento e ON ep.cp_cod_estab =
e.cp_cod_estab;",
"SELECT pr.nm_produto, f.nm_forn, fp.preco_venda_forn,
fp.data_vencimento FROM forn_produto fp JOIN tbl_produto pr ON
fp.cp_id_produto = pr.cp_id_produto JOIN tbl_fornecedor f ON
fp.cp_cod_forn = f.cp_cod_forn;",
"SELECT f.nm_func, e.nm_estab, e.cidade_estab FROM tbl_funcionario f
JOIN tbl_estabelecimento e ON f.ce_cod_estab = e.cp_cod_estab;",
"SELECT f.nm_forn, pr.nm_produto, e.nm_estab FROM forn_produto fp JOIN
tbl_produto pr ON fp.cp_id_produto = pr.cp_id_produto JOIN
tbl_fornecedor f ON fp.cp_cod_forn = f.cp_cod_forn JOIN estab_produto
ep ON pr.cp_id_produto = ep.cp_id_produto JOIN tbl_estabelecimento e ON
ep.cp_cod_estab = e.cp_cod_estab;",
```

```
"SELECT e.nm_estab, e.localizacao_estab, f.nm_forn, f.localizacao_forn
FROM tbl_estabelecimento e JOIN forn_produto fp ON e.cp_cod_estab =
fp.cp_cod_forn JOIN tbl_fornecedor f ON fp.cp_cod_forn =
f.cp_cod_forn;",
"SELECT p.cp_id_pedido, c.nm_cliente, p.data_pedido, p.hora_pedido FROM
tbl_pedido p JOIN tbl_cliente c ON p.ce_cod_cliente =
c.cp_cod_cliente;"
"WITH produto_vendido_estab AS (SELECT p.cp_id_produto,
SUM(e.preco_venda_estab) AS total_vendas_estab FROM estab_produto e
JOIN tbl_produto p ON e.cp_id_produto = p.cp_id_produto GROUP BY
p.cp_id_produto), produto_vendido_forn AS (SELECT f.cp_id_produto,
SUM(f.preco_venda_forn) AS total_vendas_forn FROM forn_produto f GROUP
BY f.cp_id_produto), produto_union_vendas AS (SELECT cp_id_produto,
total_vendas_estab AS total_vendas FROM produto_vendido_estab UNION ALL
SELECT cp_id_produto, total_vendas_forn AS total_vendas FROM
produto_vendido_forn) SELECT p.cp_id_produto, p.nm_produto,
c.nm_categoria, SUM(u.total_vendas) OVER (PARTITION BY
p.ce_categoria_principal) AS total_vendas_categoria, u.total_vendas
FROM produto_union_vendas u JOIN tbl_produto p ON u.cp_id_produto =
p.cp_id_produto LEFT JOIN tbl_categoria c ON p.ce_categoria_principal =
c.cp_cod_categoria WHERE u.total_vendas > (SELECT AVG(total_vendas)
FROM produto_union_vendas) ORDER BY total_vendas_categoria DESC,
u.total_vendas DESC;",
"WITH vendas_por_estabelecimento AS (SELECT e.cp_cod_estab,
e.cp_id_produto, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e GROUP BY e.cp_cod_estab, e.cp_id_produto),
vendas_por_fornecedor AS (SELECT f.cp_cod_forn AS cp_cod_estab,
f.cp_id_produto, SUM(f.preco_venda_forn) AS total_vendas_forn FROM
forn_produto f GROUP BY f.cp_cod_forn, f.cp_id_produto),
vendas_combinadas AS (SELECT cp_cod_estab, cp_id_produto,
total_vendas_estab AS total_vendas FROM vendas_por_estabelecimento
UNION ALL SELECT cp_cod_estab, cp_id_produto, total_vendas_forn AS
total_vendas FROM vendas_por_fornecedor) SELECT e.nm_estab,
c.nm_categoria, p.nm_produto, v.total_vendas, RANK() OVER (PARTITION BY
e.nm_estab ORDER BY v.total_vendas DESC) AS ranking_por_estab FROM
vendas_combinadas v JOIN tbl_estabelecimento e ON v.cp_cod_estab =
e.cp_cod_estab JOIN tbl_produto p ON v.cp_id_produto = p.cp_id_produto
JOIN tbl_categoria c ON p.ce_categoria_principal = c.cp_cod_categoria
WHERE v.total_vendas > (SELECT AVG(total_vendas) FROM
vendas_combinadas) GROUP BY e.nm_estab, c.nm_categoria, p.nm_produto,
v.total_vendas ORDER BY e.nm_estab, ranking_por_estab;",
"WITH vendas_por_produto AS (SELECT p.cp_id_produto,
SUM(e.preco_venda_estab) AS total_vendas_estab, SUM(f.preco_venda_forn)
```

```
AS total_vendas_forn FROM tbl_produto p LEFT JOIN estab_produto e ON
p.cp_id_produto = e.cp_id_produto LEFT JOIN forn_produto f ON
p.cp_id_produto = f.cp_id_produto GROUP BY p.cp_id_produto),
total_vendas_comb AS (SELECT cp_id_produto,
COALESCE(total_vendas_estab, 0) + COALESCE(total_vendas_forn, 0) AS
total_vendas FROM vendas_por_produto), vendas_totais_sistema AS (SELECT
SUM(total_vendas) AS vendas_totais_sistema FROM total_vendas_comb)
SELECT c.nm_categoria, COUNT(DISTINCT p.cp_id_produto) AS qtd_produtos,
AVG(t.total_vendas) AS media_vendas_categoria, (SELECT
vendas_totais_sistema FROM vendas_totais_sistema) AS
vendas_totais_sistema FROM total_vendas_comb t JOIN tbl_produto p ON
t.cp_id_produto = p.cp_id_produto JOIN tbl_categoria c ON
p.ce_categoria_principal = c.cp_cod_categoria WHERE t.total_vendas >
(SELECT AVG(total_vendas) FROM total_vendas_comb) GROUP BY
c.nm_categoria ORDER BY media_vendas_categoria DESC;",
"WITH vendas_por_estabelecimento AS (SELECT e.cp_cod_estab,
e.cp_id_produto, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e GROUP BY e.cp_cod_estab, e.cp_id_produto),
vendas_por_fornecedor AS (SELECT f.cp_cod_forn AS cp_cod_estab,
f.cp_id_produto, SUM(f.preco_venda_forn) AS total_vendas_forn FROM
forn_produto f GROUP BY f.cp_cod_forn, f.cp_id_produto),
vendas_combinadas AS (SELECT cp_cod_estab, cp_id_produto,
total_vendas_estab AS total_vendas FROM vendas_por_estabelecimento
UNION ALL SELECT cp_cod_estab, cp_id_produto, total_vendas_forn AS
total_vendas FROM vendas_por_fornecedor), vendas_por_uf AS (SELECT
e.uf_estab, SUM(v.total_vendas) AS total_vendas_uf FROM
vendas_combinadas v JOIN tbl_estabelecimento e ON v.cp_cod_estab =
e.cp_cod_estab GROUP BY e.uf_estab) SELECT e.uf_estab, e.cidade_estab,
COUNT(DISTINCT v.cp_id_produto) AS qtd_produtos_vendidos,
SUM(v.total_vendas) AS total_vendas_cidade, RANK() OVER (PARTITION BY
e.uf_estab ORDER BY SUM(v.total_vendas) DESC) AS ranking_cidade_uf FROM
vendas_combinadas v JOIN tbl_estabelecimento e ON v.cp_cod_estab =
e.cp_cod_estab WHERE v.total_vendas > (SELECT AVG(total_vendas) FROM
vendas_combinadas) GROUP BY e.uf_estab, e.cidade_estab ORDER BY
e.uf_estab, ranking_cidade_uf;",
"WITH vendas_estab AS (SELECT e.cp_id_produto, SUM(e.preco_venda_estab)
AS total_vendas_estab FROM estab_produto e GROUP BY e.cp_id_produto),
vendas_forn AS (SELECT f.cp_id_produto, SUM(f.preco_venda_forn) AS
total_vendas_forn FROM forn_produto f GROUP BY f.cp_id_produto),
vendas_comb AS (SELECT cp_id_produto, COALESCE(total_vendas_estab, 0) +
COALESCE(total_vendas_forn, 0) AS total_vendas FROM vendas_estab FULL
OUTER JOIN vendas_forn USING (cp_id_produto)), categorias_vendas AS
(SELECT c.nm_categoria, COUNT(p.cp_id_produto) AS qtd_produtos,
```

```
SUM(v.total_vendas) AS total_vendas_categoria FROM tbl_produto p JOIN
vendas_comb v ON p.cp_id_produto = v.cp_id_produto JOIN tbl_categoria c
ON p.ce_categoria_principal = c.cp_cod_categoria GROUP BY
c.nm_categoria), categoria_ranked AS (SELECT nm_categoria,
qtd_produtos, total_vendas_categoria, RANK() OVER (ORDER BY
total_vendas_categoria DESC) AS rank_vendas FROM categorias_vendas)
SELECT cr.nm_categoria, cr.qtd_produtos, cr.total_vendas_categoria,
cr.rank_vendas, AVG(v.total_vendas) AS media_vendas_produtos_categoria
FROM categoria_ranked cr JOIN tbl_produto p ON p.ce_categoria_principal
= (SELECT cp_cod_categoria FROM tbl_categoria WHERE nm_categoria =
cr.nm_categoria LIMIT 1) JOIN vendas_comb v ON p.cp_id_produto =
v.cp_id_produto WHERE cr.rank_vendas <= 3 GROUP BY cr.nm_categoria,
cr.qtd_produtos, cr.total_vendas_categoria, cr.rank_vendas ORDER BY
cr.rank_vendas;",
"WITH vendas_por_estabelecimento AS (SELECT p.cp_id_produto,
e.cp_cod_estab, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e JOIN tbl_produto p ON e.cp_id_produto = p.cp_id_produto
GROUP BY p.cp_id_produto, e.cp_cod_estab), vendas_por_fornecedor AS
(SELECT p.cp_id_produto, f.cp_cod_forn, SUM(f.preco_venda_forn) AS
total_vendas_forn FROM forn_produto f JOIN tbl_produto p ON
f.cp_id_produto = p.cp_id_produto GROUP BY p.cp_id_produto,
f.cp_cod_forn), vendas_combinadas AS (SELECT cp_id_produto,
cp_cod_estab AS origem, total_vendas_estab AS total_vendas FROM
vendas_por_estabelecimento UNION ALL SELECT cp_id_produto, cp_cod_forn
AS origem, total_vendas_forn AS total_vendas FROM
vendas_por_fornecedor), produtos_vendas_totais AS (SELECT
p.cp_id_produto, p.nm_produto, c.nm_categoria, SUM(vc.total_vendas) AS
total_vendas_produto, COUNT(DISTINCT vc.origem) AS qtd_origens FROM
vendas_combinadas vc JOIN tbl_produto p ON vc.cp_id_produto =
p.cp_id_produto JOIN tbl_categoria c ON p.ce_categoria_principal =
c.cp_cod_categoria GROUP BY p.cp_id_produto, p.nm_produto,
c.nm_categoria), ranking_produtos AS (SELECT nm_produto, nm_categoria,
total_vendas_produto, qtd_origens, RANK() OVER (PARTITION BY
nm_categoria ORDER BY total_vendas_produto DESC) AS rank_categoria FROM
produtos_vendas_totais) SELECT rp.nm_produto, rp.nm_categoria,
rp.total_vendas_produto, rp.qtd_origens, rp.rank_categoria,
AVG(rp.total_vendas_produto) OVER () AS media_vendas_geral FROM
ranking_produtos rp WHERE rp.rank_categoria <= 5 ORDER BY
rp.nm_categoria, rp.rank_categoria;",
"WITH vendas_estabelecimentos AS (SELECT e.cp_cod_estab,
e.cp_id_produto, SUM(e.preco_venda_estab) AS total_vendas_estab FROM
estab_produto e GROUP BY e.cp_cod_estab, e.cp_id_produto),
vendas_fornecedores AS (SELECT f.cp_cod_forn, f.cp_id_produto,
```

```
SUM(f.preco_venda_forn) AS total_vendas_forn FROM forn_produto f GROUP
BY f.cp_cod_forn, f.cp_id_produto), vendas_combinadas AS (SELECT
p.cp_id_produto, COALESCE(e.cp_cod_estab, f.cp_cod_forn) AS origem,
COALESCE(e.total_vendas_estab, 0) + COALESCE(f.total_vendas_forn, 0) AS
total_vendas FROM vendas_estabelecimentos e FULL OUTER JOIN
vendas_fornecedores f USING (cp_id_produto) JOIN tbl_produto p ON
p.cp_id_produto = COALESCE(e.cp_id_produto, f.cp_id_produto)),
vendas_por_estado AS (SELECT COALESCE(e.uf_estab, f.uf_forn) AS uf,
vc.cp_id_produto, SUM(vc.total_vendas) AS total_vendas_estado FROM
vendas_combinadas vc LEFT JOIN tbl_estabelecimento e ON vc.origem =
e.cp_cod_estab LEFT JOIN tbl_fornecedor f ON vc.origem = f.cp_cod_forn
GROUP BY COALESCE(e.uf_estab, f.uf_forn), vc.cp_id_produto),
ranking_produtos_estado AS (SELECT vp.uf, p.nm_produto,
SUM(vp.total_vendas_estado) AS vendas_produto_estado, RANK() OVER
(PARTITION BY vp.uf ORDER BY SUM(vp.total_vendas_estado) DESC) AS
rank_produto_estado FROM vendas_por_estado vp JOIN tbl_produto p ON
vp.cp_id_produto = p.cp_id_produto GROUP BY vp.uf, p.nm_produto) SELECT
rpe.uf, rpe.nm_produto, rpe.vendas_produto_estado,
rpe.rank_produto_estado, (SELECT AVG(vendas_produto_estado) FROM
ranking_produtos_estado) AS media_vendas_geral FROM
ranking_produtos_estado rpe WHERE rpe.rank_produto_estado <= 3 ORDER BY
rpe.uf, rpe.rank_produto_estado;",
"WITH vendas_estabelecimentos AS (SELECT e.cp_cod_estab AS origem,
p.ce_categoria_principal AS categoria, COUNT(DISTINCT e.cp_id_produto)
AS qtd_produtos_estab, SUM(e.preco_venda_estab) AS total_vendas_estab
FROM estab_produto e JOIN tbl_produto p ON e.cp_id_produto =
p.cp_id_produto GROUP BY e.cp_cod_estab, p.ce_categoria_principal),
vendas_fornecedores AS (SELECT f.cp_cod_forn AS origem,
p.ce_categoria_principal AS categoria, COUNT(DISTINCT f.cp_id_produto)
AS qtd_produtos_forn, SUM(f.preco_venda_forn) AS total_vendas_forn FROM
forn_produto f JOIN tbl_produto p ON f.cp_id_produto = p.cp_id_produto
GROUP BY f.cp_cod_forn, p.ce_categoria_principal), vendas_combinadas AS
(SELECT origem, categoria, COALESCE(qtd_produtos_estab, 0) +
COALESCE(qtd_produtos_forn, 0) AS qtd_produtos,
COALESCE(total_vendas_estab, 0) + COALESCE(total_vendas_forn, 0) AS
total_vendas FROM vendas_estabelecimentos FULL OUTER JOIN
vendas_fornecedores USING (origem, categoria)), ranking_categorias AS
(SELECT vc.categoria, c.nm_categoria, vc.origem, vc.qtd_produtos,
vc.total_vendas, RANK() OVER (PARTITION BY vc.categoria ORDER BY
vc.total_vendas DESC) AS rank_vendas_categoria FROM vendas_combinadas
vc JOIN tbl_categoria c ON vc.categoria = c.cp_cod_categoria) SELECT
r.nm_categoria, r.origem, r.qtd_produtos, r.total_vendas,
r.rank_vendas_categoria, (SELECT AVG(total_vendas) FROM
```

```
ranking_categorias) AS media_vendas_geral FROM ranking_categorias r
WHERE r.rank_vendas_categoria <= 3 ORDER BY r.nm_categoria,
r.rank_vendas_categoria;",
"WITH preco_produto_fornecedor AS (SELECT f.nm_forn, p.nm_produto,
e.preco_venda_estab, f.cp_cod_forn, p.cp_id_produto, ROW_NUMBER() OVER
(PARTITION BY p.cp_id_produto ORDER BY e.preco_venda_estab DESC) AS
rank_produto FROM estab_produto e JOIN tbl_produto p ON e.cp_id_produto
= p.cp_id_produto JOIN tbl_fornecedor f ON e.cp_cod_estab =
f.cp_cod_forn), produtos_vendidos_comparacao AS (SELECT ppf.nm_forn,
ppf.nm_produto, ppf.preco_venda_estab, (SELECT AVG(preco_venda_estab)
FROM estab_produto WHERE cp_id_produto = ppf.cp_id_produto) AS
preco_medio_produto FROM preco_produto_fornecedor ppf WHERE
ppf.rank_produto = 1), vendas_comparadas AS (SELECT nm_forn,
nm_produto, preco_venda_estab, preco_medio_produto, (preco_venda_estab
- preco_medio_produto) AS diferenca_preco FROM
produtos_vendidos_comparacao) SELECT nm_forn, nm_produto,
preco_venda_estab, preco_medio_produto, diferenca_preco, COUNT(*) OVER
(PARTITION BY nm_forn) AS quantidade_fornecimentos FROM
vendas_comparadas ORDER BY diferenca_preco DESC;",
"SELECT p.cp_id_pedido, p.ce_cod_cliente, p.data_pedido, p.hora_pedido,
p.valor_total, SUM(ep.preco_venda_estab) AS preco_venda_total,
ROW_NUMBER() OVER (PARTITION BY p.ce_cod_cliente ORDER BY p.data_pedido
DESC) AS rank_pedido FROM tbl_pedido p JOIN pedido_produto pp ON
pp.cp_id_pedido = p.cp_id_pedido JOIN estab_produto ep ON
ep.cp_id_produto = pp.cp_id_produto WHERE p.data_pedido BETWEEN
'2025-01-01' AND '2025-12-31' GROUP BY p.cp_id_pedido,
p.ce_cod_cliente, p.data_pedido, p.hora_pedido, p.valor_total UNION
SELECT p.cp_id_pedido, p.ce_cod_cliente, p.data_pedido, p.hora_pedido,
p.valor_total, 0 AS preco_venda_total, NULL AS rank_pedido FROM
tbl_pedido p WHERE p.ce_cod_cliente IN (SELECT cp_cod_cliente FROM
tbl_cliente WHERE cpf_cliente = '12345678901') ORDER BY data_pedido
DESC;"
]

# Dicionário para armazenar os tempos de execução
execution_times_tuned = {query: [] for query in queries}

# Executar as queries 50 vezes
for _ in range(50):
    for query in queries:
        start_time = time.time()
        with conn.cursor() as cursor:
            cursor.execute(query)
```

```
        end_time = time.time()
        execution_times_tuned[query].append(end_time - start_time)

# Fechar a conexão
conn.close()

# Criar um DataFrame com os tempos de execução após o tuning
df_tuned = pd.DataFrame(execution_times_tuned)
df_tuned.to_csv("tuned.csv", index=False)

# Carregar a baseline
df_baseline = pd.read_csv("baseline.csv")

# Calcular o speedup
speedup = df_baseline.mean() / df_tuned.mean()

# Criar um DataFrame com os resultados
df_speedup = pd.DataFrame({
    'Query': queries,
    'Baseline Time': df_baseline.mean(),
    'Tuned Time': df_tuned.mean(),
    'Speedup': speedup
})

# Salvar a planilha de melhoria de desempenho
df_speedup.to_csv("performance_improvement_tuning.csv", index=False)
```