

## Descrição Textual

Uma classe de usuários com a permissão do retail, dos vendedores e da **fabricante intel**, foi designada como administradora de vendas daquela fabricante. Por motivos de segurança, essa classe não pode ter acesso a vendas de produtos de outras fabricantes e só pode ver **vendas com datas de até 90 dias anteriores**, que é a data máxima para devolução dos produtos. Foi proposta uma visão que particione a tabela **Operacao\_venda**, trazendo todos os campos das vendas de fabricante intel no período mencionado a partir da data de acesso a visão.

## Query em SQL Padrão

```
CREATE VIEW vendas_recentes_intel AS

SELECT
    ov.*,
    DATE_DIFF(CURRENT_DATE, ov.data_venda, DAY)
FROM
    Produto p
INNER JOIN Fabricante fab
    ON p.fk_fabricante_id_fab = fab.id_fab
INNER JOIN Operacao_venda ov
    ON p.id_produto = ov.fk_produto_id_produto
WHERE
    nome_fab = 'intel'
    AND
    DATE_DIFF(CURRENT_DATE, ov.data_venda, DAY) < 90
ORDER BY
    data_venda DESC;
```

Crie uma view vendas\_recentes\_intel que seleciona as vendas e a diferença de datas de venda da tabela de produto que cruza com a tabela fabricante e a operação\_venda, na qual o nome da fabricante seja igual a Intel e que diferença de datas sejam de três meses(dia de hoje – 90 dias) e ordene pela data de venda de forma decrescente.

## Materialização

Essa visão é “quente” na escrita, já que muitas vendas são feitas por dia e consequentemente muitas atualizações são necessárias para mantê-la atualizada, cada venda é feita necessariamente no dia mais atual, vulgo o dia atual. Também é fato que a leitura dela não é muito comum, já que a maioria das vendas é inserida com dados corretos no banco e devoluções ou alterações são raras. Dado essas duas características, não é benéfica a materialização dessa visão, uma vez o benefício de lê-la mais rápido não compensa o custo de mantê-la sempre atualizada.

4	EXPLAIN (ANALYZE, COSTS OFF, TIMING ON)
5	select * from vendas_recntes_intel;
Data Output Explain Messages Notifications	
	<div>QUERY PLAN</div> <div>text</div> <div>1 Sort (actual time=0.137..0.140 rows=16 loops=1)</div> <div>2 Sort Key: ov.data_venda DESC</div> <div>3 Sort Method: quicksort Memory: 26kB</div> <div>4 -&gt; Hash Join (actual time=0.058..0.128 rows=16 loops=1)</div> <div>5 Hash Cond: (ov.fk_produto_id_produto = p.id_produto)</div> <div>6 -&gt; Seq Scan on operacao_venda ov (actual time=0.017..0.068 rows=11...</div> <div>7 Filter: (((now())::date - data_venda) &lt; 90)</div> <div>8 Rows Removed by Filter: 86</div> <div>9 -&gt; Hash (actual time=0.035..0.036 rows=2 loops=1)</div> <div>10 Buckets: 1024 Batches: 1 Memory Usage: 9kB</div> <div>11 -&gt; Hash Join (actual time=0.029..0.035 rows=2 loops=1)</div> <div>12 Hash Cond: (p.fk_fabricante_id_fab = fab.id_fab)</div> <div>13 -&gt; Seq Scan on produto p (actual time=0.002..0.004 rows=18 lo...</div> <div>14 -&gt; Hash (actual time=0.022..0.022 rows=1 loops=1)</div> <div>15 Buckets: 1024 Batches: 1 Memory Usage: 9kB</div> <div>16 -&gt; Index Scan using fabricante_nome_fab_key on fabricante ...</div> <div>17 Index Cond: ((nome_fab)::text = 'intel':text)</div> <div>18 Planning Time: 0.242 ms</div> <div>19 Execution Time: 0.176 ms</div>

Figura 2: Visão

1	
2	EXPLAIN (ANALYZE, VERBOSE FALSE, COSTS OFF, TIMING OFF)
3	select * from materialized_vendas_recntes_intel;
Data Output Explain Messages Notifications	
	<div>QUERY PLAN</div> <div>text</div> <div>1 Seq Scan on materialized_vendas_recntes_intel (actual rows=16 loops=1)</div> <div>2 Planning Time: 0.040 ms</div> <div>3 Execution Time: 0.021 ms</div>

Figura 1: Visão Materializada

## Alterações via visão

O **postgresql** não permite atualizações por essa visão devida ao dela ser feita com mais de uma relação, porém há a opção de usar triggers **instead of** para capturar os eventos de atualização e realizá-los de forma adequada, assim evitando a restrição. Embora a visão use mais de uma relação, ela é um mapeamento de 1:1 das colunas da tabela **Operacao\_venda**, então basta replicar a atualização da visão sobre aquela tabela.

3	select * from vendas_recntes_intel order by id_venda asc									
Data Output		Explain	Messages	Notifications						
	<div><div><div><div></div><div>Id_venda</div><div>integer</div></div><div><div></div><div></div><div></div></div></div></div>	<div><div><div><div></div><div>valor_venda</div><div>double precision</div></div><div><div></div><div></div><div></div></div></div></div>	<div><div><div><div></div><div>quantidade</div><div>integer</div></div><div><div></div><div></div><div></div></div></div></div>	<div><div><div><div></div><div>data_venda</div><div>date</div></div><div><div></div><div></div><div></div></div></div></div>	<div><div><div><div></div><div>fk_produto_id_produto</div><div>integer</div></div><div><div></div><div></div><div></div></div></div></div>	<div><div><div><div></div><div>fk_vendedor_id_vendedor</div><div>integer</div></div><div><div></div><div></div><div></div></div></div></div>	<div><div><div><div></div><div>fk_cliente_id_cliente</div><div>integer</div></div><div><div></div><div></div><div></div></div></div></div>	<div><div><div><div></div><div>dias_distancia</div><div>integer</div></div><div><div></div><div></div><div></div></div></div></div>		
1	2157	2500	5	2021-06-14	1	3	6	10		
2	2158	4000	8	2021-04-04	1	3	6	81		

```
5 delete from vendas_recetes_intel where id_venda = 2157
```

Data Output Explain Messages Notifications

ERROR: cannot delete from view "vendas\_recetes\_intel"  
DETAIL: Views that do not select from a single table or view are not automatically updatable.  
HINT: To enable deleting from the view, provide an INSTEAD OF DELETE trigger or an unconditional ON DELETE DO INSTEAD rule.  
SQL state: 55000

## Teste

Criando o trigger de delete somente.

```
1 CREATE OR REPLACE FUNCTION delete_vendas_recetes_intel()
2 RETURNS TRIGGER language plpgsql
3 as
4 $$
5 BEGIN
6
7     delete from Operacao_venda
8     where
9         (OLD.id_venda = id_venda OR (OLD.id_venda is null))
10        AND (OLD.valor_venda = valor_venda OR (OLD.valor_venda is null))
11        AND (OLD.quantidade = quantidade OR (OLD.quantidade is null))
12        AND (OLD.data_venda = data_venda OR (OLD.data_venda is null))
13        AND (OLD.fk_produto_id_produto = fk_produto_id_produto OR (OLD.fk_produto_id_produto is null))
14        AND (OLD.fk_vendedor_id_vendedor = fk_vendedor_id_vendedor OR (OLD.fk_vendedor_id_vendedor is null))
15        AND (OLD.fk_cliente_id_cliente = fk_cliente_id_cliente OR (OLD.fk_cliente_id_cliente is null))
16        AND (OLD.dias_distancia = dias_distancia OR (OLD.dias_distancia is null));
17
18     return NULL;
19 END;
20 $$;
21 create trigger trigger_delete_vendas_recetes_intel
22 instead of delete on vendas_recetes_intel
23 for each row execute procedure delete_vendas_recetes_intel();
```

Tentando novamente.

```
2 delete from vendas_recetes_intel where id_venda = 2157;
3 select * from vendas_recetes_intel order by id_venda asc;
```

Data Output Explain Messages Notifications

	id_venda integer	valor_venda double precision	quantidade integer	data_venda date	fk_produto_id_produto integer	fk_vendedor_id_vendedor integer	fk_cliente_id_cliente integer	dias_distancia integer
1	2158	4000	8	2021-04-04	1	3	6	81
2	2159	4500	9	2021-05-21	1	4	6	34