****

**Serviço Nacional de Aprendizagem Industrial**

**SENAI “Gaspar Ricardo Junior”**

Curso

TÉCNICO EM DESENVOLVIMENTO

DE SISTEMAS

SQL Views

Quéren-Hapuque.L.Vieira

Sorocaba

Novembro – 2024



**Serviço Nacional de Aprendizagem Industrial**

**SENAI “Gaspar Ricardo Junior”**

Quéren-Hapuque.L.Vieira

SQL Views

Prof. – Emerson

Sorocaba

Novembro – 2024

**Sumário**

[OBJETIVO 4](#_Toc182379319)

[1. O que são SQL Views e como diferem de tabelas normais? 5](#_Toc182379320)

[1.1. Diferenças entre Views e Tabelas Normais: 5](#_Toc182379321)

[2. Quais as principais vantagens e desvantagens do uso de views em um banco de dados? 6](#_Toc182379322)

[3. Como views podem melhorar a segurança e o desempenho das consultas? 6](#_Toc182379323)

[4. Quando uma view é atualizável e em que casos isso não é possível? 8](#_Toc182379324)

[5. Em que situações específicas é recomendável usar views em um sistema de banco de dados? 9](#_Toc182379325)

[6. Exemplo de view complexa: 9](#_Toc182379326)

[7. Criação de Views no SQL 10](#_Toc182379327)

[7.1. Simples: 10](#_Toc182379328)

[7.2. Filtragem: 11](#_Toc182379329)

[7.3. Agregação: 11](#_Toc182379330)

[7.4. Junção: 11](#_Toc182379331)

[8. CONCLUSÃO 13](#_Toc182379332)

# 

OBJETIVO

O objetivo desta pesquisa é entender o que são as SQL Views,

por que elas são importantes, e como podem ser utilizadas para facilitar o

acesso e a manipulação de dados em bancos de dados relacionais. Além disso,

será explorado o processo de criação de views e exemplos práticos que ilustram

suas aplicações no dia a dia.

# 

# O que são SQL Views e como diferem de tabelas normais?

SQL Views são "visões" virtuais de dados armazenados em um banco de dados. Elas não contêm dados fisicamente, mas representam uma consulta SQL armazenada que, quando acessada, retorna um conjunto de resultados gerado dinamicamente a partir de uma ou mais tabelas. Em outras palavras, uma view é uma espécie de "janela" para acessar dados, permitindo que você visualize e manipule esses dados sem modificar diretamente as tabelas subjacentes.

## Diferenças entre Views e Tabelas Normais:

Armazenamento de Dados: As tabelas armazenam fisicamente os dados, enquanto as views não armazenam dados, mas sim a definição de uma consulta que retorna dados.

Persistência: Tabelas são entidades persistentes no banco de dados, enquanto views são virtuais e suas definições podem ser alteradas ou excluídas sem afetar os dados reais.

Flexibilidade: Views podem combinar dados de múltiplas tabelas, aplicar filtros e operações complexas, enquanto as tabelas são mais estruturadas e focadas em armazenar dados de forma direta..

# Quais as principais vantagens e desvantagens do uso de views em um banco de dados?

Vantagens:

Simplificação de Consultas Complexas: Views podem encapsular consultas complicadas, permitindo que os usuários ou desenvolvedores acessem os dados sem precisar reescrever consultas complexas.

Segurança: Permitem restringir o acesso a determinadas colunas ou linhas, fornecendo uma forma de limitar o que é visível a diferentes usuários. Por exemplo, uma view pode mostrar apenas dados de vendas sem incluir informações confidenciais sobre clientes.

Manutenção: Quando uma consulta frequentemente utilizada precisa ser alterada, isso pode ser feito na definição da view, sem alterar as consultas que dependem dela.

Desvantagens:

Desempenho: Em consultas complexas, especialmente quando as views envolvem junções múltiplas e agregações, o desempenho pode ser prejudicado, já que a consulta é executada toda vez que a view é acessada.

Atualizações Limitadas: Algumas views, especialmente as que envolvem junções ou agregações, não são atualizáveis. Isso significa que não se pode inserir, atualizar ou excluir dados diretamente através da view.

Manutenção de Views Materializadas: Views materializadas armazenam fisicamente os resultados da consulta, o que pode melhorar o desempenho em algumas situações, mas elas exigem atualização periódica, o que pode adicionar complexidade ao sistema.

# Como views podem melhorar a segurança e o desempenho das consultas?

A estrutura que você propôs para a pesquisa sobre SQL Views está bem organizada e abrange os principais pontos que precisam ser discutidos. Vou ajudar a elaborar respostas e orientações para as perguntas que você propôs, de forma a direcionar a pesquisa de forma mais clara.

1. O que são SQL Views e como diferem de tabelas normais?

SQL Views são "visões" virtuais de dados armazenados em um banco de dados. Elas não contêm dados fisicamente, mas representam uma consulta SQL armazenada que, quando acessada, retorna um conjunto de resultados gerado dinamicamente a partir de uma ou mais tabelas. Em outras palavras, uma view é uma espécie de "janela" para acessar dados, permitindo que você visualize e manipule esses dados sem modificar diretamente as tabelas subjacentes.

Diferenças entre Views e Tabelas Normais:

Armazenamento de Dados: As tabelas armazenam fisicamente os dados, enquanto as views não armazenam dados, mas sim a definição de uma consulta que retorna dados.

Persistência: Tabelas são entidades persistentes no banco de dados, enquanto views são virtuais e suas definições podem ser alteradas ou excluídas sem afetar os dados reais.

Flexibilidade: Views podem combinar dados de múltiplas tabelas, aplicar filtros e operações complexas, enquanto as tabelas são mais estruturadas e focadas em armazenar dados de forma direta.

2. Quais as principais vantagens e desvantagens do uso de views em um banco de dados?

Vantagens:

Simplificação de Consultas Complexas: Views podem encapsular consultas complicadas, permitindo que os usuários ou desenvolvedores acessem os dados sem precisar reescrever consultas complexas.

Segurança: Permitem restringir o acesso a determinadas colunas ou linhas, fornecendo uma forma de limitar o que é visível a diferentes usuários. Por exemplo, uma view pode mostrar apenas dados de vendas sem incluir informações confidenciais sobre clientes.

Manutenção: Quando uma consulta frequentemente utilizada precisa ser alterada, isso pode ser feito na definição da view, sem alterar as consultas que dependem dela.

Desvantagens:

Desempenho: Em consultas complexas, especialmente quando as views envolvem junções múltiplas e agregações, o desempenho pode ser prejudicado, já que a consulta é executada toda vez que a view é acessada.

Atualizações Limitadas: Algumas views, especialmente as que envolvem junções ou agregações, não são atualizáveis. Isso significa que não se pode inserir, atualizar ou excluir dados diretamente através da view.

Manutenção de Views Materializadas: Views materializadas armazenam fisicamente os resultados da consulta, o que pode melhorar o desempenho em algumas situações, mas elas exigem atualização periódica, o que pode adicionar complexidade ao sistema.

3. Como views podem melhorar a segurança e o desempenho das consultas?

Segurança:

Restrição de Acesso: Uma das principais razões para usar views em um sistema de banco de dados é o controle de acesso. Você pode criar views que forneçam acesso apenas a dados específicos de uma tabela, ocultando colunas sensíveis ou linhas de dados (como informações pessoais ou financeiras). Isso garante que os usuários só vejam os dados aos quais têm permissão.

Exemplo: Você pode ter uma tabela de clientes com colunas como nome, endereço, data\_nascimento, e salario. Você pode criar uma view que mostre apenas nome e endereço, restringindo o acesso a dados sensíveis.

Desempenho:

Consultas Simplificadas: Em vez de escrever a mesma consulta SQL complexa várias vezes, uma view pode ser criada para encapsular essa lógica, melhorando a legibilidade e manutenção do código.

Views Materializadas: Em cenários de leitura intensiva, uma view materializada pode ser utilizada para armazenar fisicamente os resultados de uma consulta, melhorando o desempenho das consultas subsequentes, já que não será necessário reexecutar a consulta original toda vez que a view for acessada.

# Quando uma view é atualizável e em que casos isso não é possível?

Uma **view é atualizável** quando os dados subjacentes da view podem ser modificados diretamente através dela, ou seja, inserções, atualizações e exclusões podem ser feitas via comandos SQL como INSERT, UPDATE e DELETE na view.

**Condições para uma view ser atualizável**:

* A view deve se basear em **uma única tabela**.
* A view não deve ter **funções de agregação** (como SUM, COUNT, etc.).
* A view não pode ter **junções complexas**.
* A view não pode conter **subconsultas** ou **cláusulas** GROUP BY.

**Exemplo de uma view não atualizável**:

* Uma view que envolva uma **junção entre várias tabelas**, ou que utilize **funções agregadas**, como SUM, AVG, não será atualizável, pois as operações de agregação ou junção não sabem como refletir as mudanças de volta para as tabelas originais.

# Em que situações específicas é recomendável usar views em um sistema de banco de dados?

Acesso a Dados Complexos: Quando há necessidade de apresentar dados de várias tabelas de forma simplificada (como em relatórios que exigem junções complexas ou agregações), as views são uma maneira prática de centralizar essa lógica.

Segurança: Se for necessário restringir o acesso a dados sensíveis (como informações de clientes ou transações financeiras), você pode criar views que ocultem certas colunas ou linhas da tabela original.

Reutilização de Consultas: Quando uma consulta SQL é usada frequentemente, uma view pode ser criada para evitar a duplicação de código e melhorar a manutenção.

Desempenho: Em alguns casos, especialmente quando se trabalha com grandes volumes de dados, views materializadas podem ser usadas para armazenar resultados de consultas pesadas, evitando a execução repetida da mesma consulta e melhorando o desempenho.

# Exemplo de view complexa:

CREATE VIEW relatorio\_vendas\_completo AS

SELECT p.nome AS produto,

r.nome AS regiao,

COUNT(v.id) AS total\_vendas,

SUM(v.valor\_venda) AS total\_vendas\_valor

FROM vendas v

JOIN produtos p ON v.produto\_id = p.produto\_id

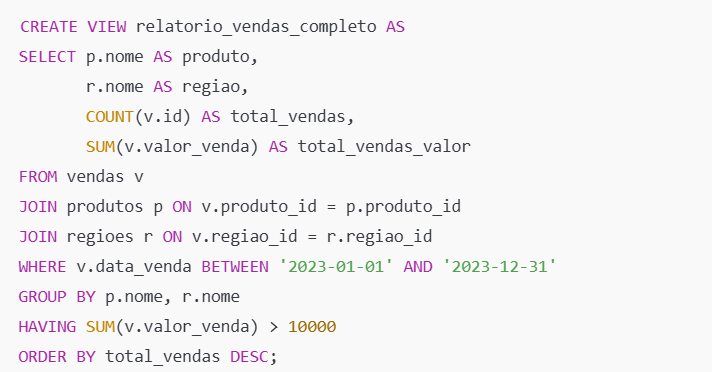
JOIN regioes r ON v.regiao\_id = r.regiao\_id

WHERE v.data\_venda BETWEEN '2023-01-01' AND '2023-12-31'

GROUP BY p.nome, r.nome

HAVING SUM(v.valor\_venda) > 10000

ORDER BY total\_vendas DESC;



Neste exemplo, a view relatorio\_vendas\_completo traz um relatório de vendas agrupado por produto e região para o ano de 2023. Ela inclui:

Junções entre as tabelas vendas, produtos e regioes.

Funções de agregação como COUNT (para o número de vendas) e SUM (para o valor total das vendas).

Um filtro de intervalo de datas (vendas de 2023).

Uma cláusula HAVING para filtrar apenas as regiões e produtos com vendas acima de 10.000 reais.

Ordenação pelo total de vendas (total\_vendas), do maior para o menor.

# Criação de Views no SQL

## Simples:

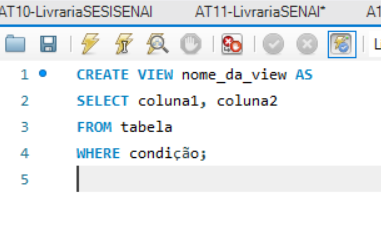
No SQL, uma view é uma "visão" ou uma tabela virtual criada a partir do resultado de uma consulta SELECT. Ela pode ser utilizada para simplificar consultas complexas, ocultar dados sensíveis, ou até mesmo melhorar a reutilização de códigos SQL.

CREATE VIEW nome\_da\_view AS

SELECT coluna1, coluna2, ...

FROM tabela

WHERE condição;



nome\_da\_view: o nome que você deseja atribuir à view.

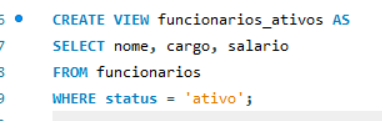
SELECT: a consulta SELECT que define quais dados serão apresentados na view.

FROM tabela: a tabela (ou tabelas) da qual os dados serão extraídos.

WHERE condição: uma condição opcional para filtrar as linhas que serão exibidas na view.

## Filtragem:

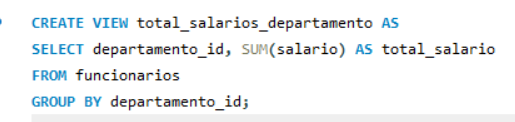
Uma view de filtragem é usada para selecionar colunas e linhas específicas de uma tabela. Exemplo:



Neste exemplo, a view funcionarios\_ativos exibe apenas os funcionários ativos (onde o campo status é igual a 'ativo'), e inclui as colunas nome, cargo e salario.

## Agregação:

As views de agregação são usadas para aplicar funções de agregação, como SUM, AVG, COUNT, etc., para resumir ou calcular dados.

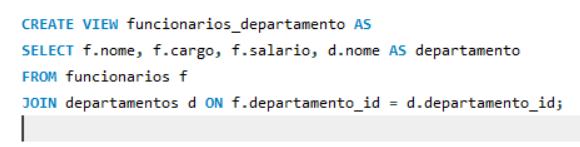


Aqui, a view total\_salarios\_departamento mostra a soma dos salários por cada departamento\_id, com o resultado rotulado como total\_salario.

## Junção:

Uma view de junção é usada para combinar dados de duas ou mais tabelas. Isso é útil quando você precisa de dados provenientes de diferentes tabelas, mas não quer fazer junções repetidas em várias consultas.

Exemplo de view combinando dados de duas tabelas (supondo que exista uma tabela departamentos):



Aqui, a view funcionarios\_departamento traz o nome do funcionário, cargo, salário e o nome do departamento associado, utilizando uma junção entre as tabelas funcionarios e departamentos.

# CONCLUSÃO

A pesquisa sobre SQL Views deve ser focada em destacar como as views, tanto simples quanto complexas, podem simplificar a consulta de dados e melhorar o desempenho e segurança em sistemas de banco de dados. Além disso, é importante analisar as vantagens e desvantagens do uso de views em diferentes cenários, especialmente em termos de manutenção e performance.

Sugestões de práticas recomendadas:

Criação de views para simplificação de acesso: Use views sempre que for necessário simplificar consultas complexas ou combinar dados de várias tabelas.

Manutenção regular de views materializadas: Caso utilize views materializadas, planeje uma estratégia de atualização periódica para garantir que os dados estejam sempre consistentes.

Controle de acesso: Utilize views para restringir o acesso a dados sensíveis e garantir a segurança no sistema.

Com esses pontos, sua pesquisa pode abordar os aspectos técnicos e práticos das SQL Views de forma aprofundada e útil para profissionais que trabalham com bancos de dados relacionais.