Instituto Federal

Campus Goiânia

Bacharelado em Sistemas de Informação

Banco de Dados II

Prof. Dory Gonzaga Rodrigues

MySQL®

ORACLE®

PostgreSQL
the world's most advanced open source database

Agenda

- Objetos Avançados

  - VIEWS

    Simples;
    Complexa e
    Atualizável

  - VIEWS MATERIALIZED

  - ANÁLISE “TOP-N”

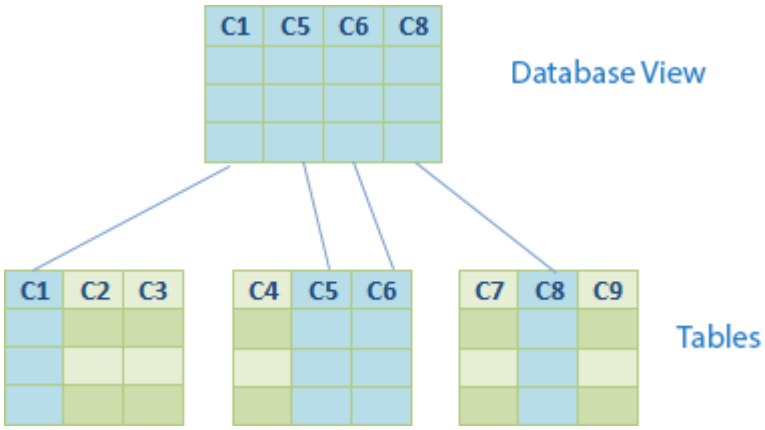
OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados

Um banco de dados é composto por vários objetos.

Uma visão é um objeto do banco de dados. A View é uma “Tabela Virtual" que permite consultas através dela. Compreender o que é uma VIEW de banco de dados e usá-la corretamente é muito muito importante. Nesta aula, vamos discutir sobre o ponto de vista do banco de dados, como uma VIEW é implementada e como usá-la de forma mais eficaz.



OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: VIEWS

- Uma VIEW em um banco de dados representa logicamente subconjuntos de dados de uma ou mais tabelas;

- Uma VIEW é uma tabela lógica baseada em tabela ou outra view.

- Uma VIEW não contém dados próprios mas é como uma janela através da qual os dados das tabelas podem ser vistos ou alterados.

- As tabelas nas quais uma VIEW é baseada são chamadas tabelas-base.

- A VIEW é armazenada como uma instrução SELECT no dicionário de dados.

- Quando ocorre alteração nos dados de uma tabela, a VIEW reflete automaticamente as mudanças também.

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: VIEWS

Por Que Usar Views?

- Para restringir o acesso a dados, ou seja, mostrar apenas o que deve ser mostrado;

- Para facilitar as consultas complexas, ou seja, implementamos as consultas e disponibilizamos os dados já tratados para as aplicações ou usuários finais;

- Para permitir a independência dos dados, busca de dados em diversas tabelas;

- Para apresentar diferentes visões dos mesmos dados, ou seja, a informação será apresentada da forma que interessa a cada usuário ou grupo de Usuário;

- Podemos ter colunas de campos derivados, ou seja, podemos ter colunas de dados calculados. Ex: Total da Nota;

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: VIEWS

Por Que Usar Views?

- Compatibilidade com versões e sistemas legados. Mesmo com alterações nas tabelas do banco de dados (criação/alteração/exclusão) podemos criar visões com o mesmo esquema e tabelas dos sistemas legados;

Desvantagens

- Performance: consulta através de VIEW pode ser lenta, especialmente se a exibição é criada com base em outros visões.

- Dependência entre tabelas: sempre que você alterar a estrutura das tabelas que participam da VIEW, você terá que ajustar na visão também.

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: VIEWS

As VIEW podem ser classificadas da seguinte forma:

Recurso	View Simples	View Complexa
Número de Tabelas	Uma	Duas ou mais
Contém Funções	NÃO	SIM
Contém Dados Agrupados	NÃO	SIM
Uso de comando DML através da VIEW	SIM	DEPENDE

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: CRIANDO UMA VIEW SIMPLES

A sintaxe:

```
CREATE [OR REPLACE] VIEW  nome_da_view AS
      SELECT colunas
       FROM tables
      [WHERE conditions];
```

Opcional:

[OR REPLACE] recria a view se ela já existir!

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: CRIANDO UMA VIEW SIMPLES

Exemplo: Crie uma visão que projete o nome do CD e o valor de venda dos CDs da gravadora.

```
CREATE OR REPLACE VIEW vCD_Venda AS
    SELECT Nome_CD, Preco_Venda
     FROM CD;
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: CRIANDO UMA VIEW SIMPLES

Existem algumas regras no uso da instrução SELECT na criação de uma VIEW SIMPLES:

- O SELECT pode conter subconsulta na cláusula WHERE, mas não na cláusula FROM.

- O SELECT não pode fazer referência a variáveis, incluindo variável local, variável de usuário ou variável de sessão.

- O SELECT não pode se referir aos parâmetros de Prepared Statements.

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: CRIANDO UMA VIEW COM SUBCONSULTA

Exercício: Crie uma visão que projete o código da gravadora, o nome do CD e o valor de venda do CD que estão acima da média de preço de venda.

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: CRIANDO UMA VIEW COM SUBCONSULTA

Exercício: Crie uma visão que projete o código da gravadora, o nome do CD e o valor de venda do CD que estão acima da média de preço de venda.

```
CREATE OR REPLACE VIEW v_CDsAcimaDaMedia AS
    SELECT idGravadora, Nome_CD, Preco_Venda
     FROM CD
    WHERE Preco_Venda > (SELECT AVG(Preco_Venda)
                           FROM cd
                        );
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: CRIANDO UMA VIEW COMPLEXA

Exercício: Crie uma visão que projete o nome da gravadora e a quantidade de CDs gravados por gravadora, mesmo que exista gravadoras sem CDs cadastrados.

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: CRIANDO UMA VIEW COMPLEXA

Exercício: Crie uma visão que projete o nome da gravadora e a quantidade de CDs gravados por gravadora, mesmo que exista gravadoras sem CDs cadastrados.

```
CREATE OR REPLACE VIEW v_GravadoraQtdeCDs AS
        SELECT NomeGravadora, COUNT(idCD) AS Qtde_CDs
         FROM GRAVADORA left join CD using(idGravadora)
    GROUP BY idGravadora;
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: CRIANDO UMA VIEW COMPLEXA

Exercício: Crie uma visão que mostre as Gravadoras ordenadas pelo nome, com seus respectivos CDs com a quantidade de músicas e o tempo total das músicas.

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: CRIANDO UMA VIEW COMPLEXA

Exercício: Crie uma visão que mostre as Gravadoras ordenadas pelo nome, com seus respectivos CDs com a quantidade de músicas e o tempo total das músicas.

```
CREATE OR REPLACE VIEW vGRAV_CD_QTDE_MUSICA_TEMPO AS
      SELECT      nomeGravadora AS NomeG,
                  nome_CD       AS NCd,
                  COUNT(idCD)   AS QTDE_Musica,
                  SUM(duracao)  AS TempoTotal
        FROM      Gravadora
   LEFT JOIN      CD using(idgravadora)
   LEFT JOIN      Faixa using(idcd)
   LEFT JOIN      Musica using(idmusica)
   GROUP BY       NomeG, NCd
   ORDER BY       NomeG ASC, NCd ASC;
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: SELECIONANDO DADOS DE UMA VIEW

Podemos utilizar o comando SELECT para realizar consultas em uma VIEW.

```
SELECT *
 FROM vCD_Gravadora;

SELECT *
 FROM v_CDsAcimaDaMediaPreco

SELECT *
 FROM v_GravadoraQtdeCDs

SELECT nomeg, ncd, MAX(qtde_musica) AS QTDE
 FROM vGRAV_CD_QTDE_MUSICA_TEMPO
GROUP BY nomeg, ncd
ORDER BY qtde DESC LIMIT 1;
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: UMA VIEW ATUALIZÁVEL

As VIEWs não são apenas de leitura (consulta), mas também atualizáveis. No entanto, a fim de criar um VIEW atualizável, a instrução SELECT que define a visão tem que seguir algumas regras:

- O SELECT não deve usar GROUP BY ou HAVING cláusula;

- O SELECT não deve usar a cláusula DISTINCT na lista das colunas;

- O SELECT não deve referir-se na cláusula FROM a VIEW somente leitura;

- O SELECT não deve conter expressões do tipo (agregados, funções, colunas calculadas.

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: UMA VIEW ATUALIZÁVEL

Exemplo: Crie uma view atualizável da tabela Gravadora, contendo as colunas idGravadora, NomeGravadora, Endereco e Telefone.

Exemplo: Atualize, através da VIEW da Gravadora, o telefone para '+55 62 3723 5555' da gravadora com código igual a “1”.

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: UMA VIEW ATUALIZÁVEL

Exemplo: Crie uma view atualizável da tabela Gravadora, contendo as colunas idGravadora, NomeGravadora, Endereco e Telefone.

```
CREATE OR REPLACE VIEW v_atu_Gravadora AS
    SELECT idGravadora, NomeGravadora, Endereco, Telefone
     FROM GRAVADORA;
```

Exemplo: Atualize, através da VIEW da Gravadora, o telefone para '+55 62 3723 5555' da gravadora com código igual a “1”.

```
SELECT * FROM Gravadora;

UPDATE v_atu_Gravadora
    SET Telefone  =  '+55 62 3723 5555'
  WHERE idGravadora  =  1;

SELECT * FROM Gravadora;
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: VIEWS MATERIALIZED

O que é uma View Materialized?

- As visões materializadas possuem as mesmas regras utilizadas nas visões não materializadas. A diferença é que o resultado (os dados) da consulta da visão são gravados/persistidos em um formulário semelhante a estrutura de uma tabela. As principais diferenças ;

Principal Diferença

- Uma visão materializada armazena os dados contidos no momento em que foi criada. Caso ocorram atualizações nos dados das tabelas base, a visão materializa não é atualizada imediatamente. Para que os novos dados possam ser gerados para a visão materializada, utilizamos o comando:

REFRESH MATERIALIZED VIEW <nome_da_view>;

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: VIEWS MATERIALIZED

Quando utilizar a View Materialized?

- As visões materializadas devem ser utilizadas quando os dados mais recentes não sejam tão importantes. Normalmente quando se trata de dados estatísticos.

Exemplo: uma view materializada de uma tabela de vendas para que sejam realizadas consultas estatísticas ou dados históricos das vendas.

Sintaxe:

```
CREATE MATERIALIZED VIEW nome AS
    SELECT  … ;
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: VIEWS MATERIALIZED

Exemplo:

```
CREATE MATERIALIZED VIEW  v_GravadoraQtdeCDs AS
        SELECT NomeGravadora, COUNT(idCD) AS Qtde_CDs
         FROM GRAVADORA left join CD using(idGravadora)
    GROUP BY idGravadora;
```

Lembre-se: caso tenhamos novos dados ou atualizações realizadas na tabela base (Gravadora) após a criação da visão materializada, devemos utilizar o comando abaixo para atualizar a visão:

```
REFRESH MATERIALIZED VIEW v_GravadoraQtdeCDs;
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: APAGANDO UMA VIEW

A sintaxe:

```
DROP VIEW [IF EXISTS] nome_da_view ;
DROP MATERIALIZED VIEW [IF EXISTS] nome_da_view ;
```

Exemplos:

```
DROP VIEW vCD_Gravadora;

DROP VIEW v_CDsAcimaDaMediaPreco;

DROP VIEW v_GravadoraQtdeCds;

DROP VIEW  IF EXISTS vGrav_CD_Qtde_Musica_Tempo;

DROP MATERIALIZED VIEW IF EXISTS v_gravadoraqtdecds;
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: ANÁLISE "Top-N"

As consultas do tipo **"Top-N"** são aquelas que pedem os “n” maiores ou “n” menores valores de uma coluna.

Exemplo:

– Quais são os dez produtos mais vendidos?
– Quais são os dez produtos menos vendidos?

Tanto o conjunto dos maiores quanto dos menores valores são considerados consultas Top-N.

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: ANÁLISE "Top-N"

As consultas Top-N usam uma estrutura de consultas aninhadas consistentes com os elementos descritos abaixo:

- Uma consulta ou view com a cláusula ORDER BY para assegurar que a classificação esteja na ordem desejada.

- O operador LIMIT para limitar o número de linhas no conjunto final de resultados.

Exemplo:

```
SELECT *
 FROM AUTOR
ORDER BY Nome_Autor ASC LIMIT 4;
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: ANÁLISE "Top-N"

Exemplo: Quais são as dez músicas MAIS LONGAS ?

Exemplo: Quais são as dez músicas MAIS CURTAS ?

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: ANÁLISE "Top-N"

Exemplo: Quais são as dez músicas MAIS LONGAS ?

```
SELECT *
 FROM MUSICA
ORDER BY DURACAO DESC
 LIMIT 10;
```

Exemplo: Quais são as dez músicas MAIS CURTAS ?

```
SELECT *
 FROM MUSICA
ORDER BY DURACAO ASC
 LIMIT 10;
```

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: ANÁLISE "Top-N"

Exemplo: Quais são as DEZ músicas MAIS LONGAS ordenadas de forma CRESCENTE pelo Nome da Musica?

OBJETOS AVANÇADOS

SQL – AVANÇADA

- Objetos Avançados: ANÁLISE "Top-N"

Exemplo: Quais são as DEZ músicas MAIS LONGAS ordenadas de forma CRESCENTE pelo Nome da Musica?

```
SELECT m.nomeMusica, m.duracao
 FROM (     SELECT *
              FROM MUSICA
          ORDER BY DURACAO DESC
              LIMIT 10
        ) M
ORDER BY nomeMusica ASC;
```