Banco de Dados

anthonyferreiralamarca@gmail.com

VIEW

- É uma única tabela derivada de outras tabelas
- Outras tabelas podem ser tabelas de base ou outras views previamente definidas
- É considerada uma tabela virtual, pois não necessariamente existe em forma física
- Limitando as possíveis operações de atualização que podem ser aplicadas às views
- Não oferecem quaisquer limitações sobre consulta

VIEW

- Uma tabela que precisamos consultar com frequência
- Geralmente com junções de outras tabelas e/ou views
- Exemplo
  - Suponhamos que sempre consultamos, no nosso BD ‘Empresa’, o nome dos funcionários e o nome dos projetos que estes trabalham

VIEW

- Sempre teríamos que fazer a junção de três relações (FUNCIONARIO, PROJETO, TRABALHA_EM)
- Poderíamos fazer uma view que seria o resultado desta consulta
- Depois faríamos as consultas nesta view, sem a necessidade de junções
- As tabelas que definem a view
  - Tabelas de definição de view

VIEW

- Estrutura
- CREATE VIEW
- O nome da view
- Lista de nomes de atributos
- E uma consulta para especificar seu conteúdo
- Se não houver funções e nem operações matemáticas, não precisa especificar os novos nomes de atributo
- Serão iguais os das tabelas de definição da view

Exemplo

- CREATE VIEW TRABALHA
  AS SELECT PNOME, UNOME, PROJNOME, HORAS
  FROM FUNCIONARIO, PROJETO, TRABALHA_EM
  WHERE CPF = FCPF AND PNR = PROJNUMERO;
- CREATE VIEW DEP_INFO (DEP_NOME, QTD_FUNC, TOTAL_SAL)
  AS SELECT DNOME, COUNT(*), SUM(SALARIO)
  FROM DEPARTAMENTO, FUNCIONARIO
  WHERE DNR = DNUMERO
  GROUP BY DNOME;

VIEW

- Agora podemos especificar consultas SQL na view
- Recuperar todos os nomes e sobrenomes que trabalham no ProjetoX
- SELECT PNOME,UNOME

  FROM TRABALHA

  WHERE PROJNOME = ‘PRODUTOX’;
- Caso não precise mais da view
- DROP VIEW TRABALHA;

Implementação Eficiente de VIEW

- Materialização de VIEW
- Cria fisicamente uma tabela de view temporária quando ela for consultada pela primeira vez e a mantem com a suposição de que novas consultas a view acontecerão
- Quando uma das tabela de definição for alterada, a view deverá ser atualizada automaticamente
- Técnica denominada como **atualização incremental**

VIEW

- Desta forma, o SGBD pode determinar quais tuplas devem ser inseridas, modificadas ou excluídas em uma tabela view materializada
  - Quando uma atualização de BD é aplicada a uma das tabelas de definição da view
- A VIEW é geralmente mantida como uma tabela materializada
  - Desde que esteja sendo consultada
- Caso a view não seja consultada por um período de tempo, o SGBD pode remove-la fisicamente automaticamente
- Posteriormente será materializada novamente

Atualização de VIEW

- Não é recomendado
- Pode gerar muita ambiguidade nas relações envolvidas
- Alterando de forma errada o que o usuário pediu
- Quando a VIEW está relacionado apenas com uma tabela de definição e tem a sua chave primária
  - Sem problemas

VIEW

- As view definidas usando funções de agrupamento e agregação não são atualizáveis
- Caso queira atualizar a VIEW utilize no final da definição de uma VIEW
  - WITH CHECK OPTION
- O sistema irá verificar a possibilidade da atualização da VIEW
- Planejar estratégias de execução para ela

Exercícios

- Criar uma VIEW que tem o nome do departamento, o nome do gerente e o salário do gerente para cada departamento
- Uma VIEW que tenha o nome do funcionário, nome do supervisor e o salário de cada funcionário que trabalha no departamento ‘administrativo’

Exercícios

- Uma VIEW que tenha o nome do projeto, nome do departamento que o controla, número de funcionário e total de horas trabalhadas por semana em cada projeto
- Uma VIEW que tenha o nome do projeto, nome do departamento que o controla, número de funcionário e total de horas trabalhadas por semana no projeto para cada projeto com mais de três funcionários trabalhando nele

Função

- Operações que são realizadas em determinados SELECT em uma determinada tabela
- Torna-las disponível dentro do banco de dados para outras tabelas
- Toda vez que quiser fazer aquela operação, chame a função que a implemente
- Manutenção apenas na função
- Exemplo
  - Formatações
  - Moeda
  - CEP
  - CPF

Função

- Sintaxe
- CREATE FUNCTION sp_name (parameter)
  [RETURNS type]
  [routine body]
- A lista de parâmetros entre parênteses deve estar sempre presente
- Se não houver parâmetros, uma lista vazia () deve estar presente
- Todos os parâmetros são de entrada (IN)
- Tipo do retorno da função
- Depois o corpo da função com BEGIN e END

Função

- São invocadas diretamente no SELECT
- Seu retorno é colocado no RESULT SET daquele SELECT
- Existe funções em nível do banco de dados e em nível UDF
- Nível UDF (Funções Definidas pelo Usuário)
- Funções definidas pelo usuário que podem ser colocadas no MYSQL. Exemplos.
  - FORMAT DATE
  - TRUNCATE
  - ROUND
  - ETC.....

Função

- Fica disponível em nível de SERVIDOR
- Exemplo nosso aqui é em nível de Banco
- Criar uma função que retorna o nome e o sobrenome dos funcionários concatenados
- CREATE FUNCTION NOME_SOBRENOME (NOME VARCHAR(50), SOBRENOME VARCHAR(50)) RETURNS VARCHAR(100)

```
BEGIN
    RETURN CONCAT(NOME,' ',SOBRENOME);
END;
```

Função

- Dentro das Funções é possível chamar outras funções
  - Concat()
  - Round()
  - Cos()
  - Curtime()
- Visite para demais funções
  - http://ftp.nchu.edu.tw/MySQL/doc/refman/4.1/pt/date-and-time-functions.html
- É possível declarar variáveis dentro das funções
  - Utilize o termo DECLARE

Exercício

- Criar uma função que formata o salário dos funcionários, identificando-o como reais e delimitando os centavos apenas em 2 (duas) casas
- Faça um SELECT e demonstre a função
- Crie uma função para formatar a data em dia, mês e ano
- Faça o SELECT para demonstrar o uso da função

Exercício

- Faça uma função que a partir do CPF do funcionário, traga o nome e seu supervisor mediato
- Faça o SELECT para demonstrar o uso da função