Visões (Views)

André L. Maravilha

Conceitos iniciais

- Uma ***view*** (ou visão) é uma **tabela virtual** derivada de uma ou mais tabelas.
  - Uma *view* não existe de forma física, ou seja, seus registros não são realmente armazenados na base de dados.

- Pode-se interpretar uma *view* como uma tabela que precise ser consultada com frequência, embora não exista fisicamente na base de dados.

- Quando uma *view* é utilizada, os dados da mesma são armazenados em uma tabela temporária.

Conceitos iniciais

- Algumas vantagens de se utilizar *views*:
  - Economia de tempo com reescrita de instruções grandes.
  - Acesso mais rápido, já que a *view* é executada uma vez e seus dados ficam armazenados em uma tabela temporária.
  - Mascara a complexidade do banco de dados.
    - Podemos criar várias *views* de junções complexas e trabalharmos com elas como uma tabela simples.
    - Quando ocorre a atualização das tabelas que originaram uma *view*, a mesma apresentará os dados atualizados quando for utilizada.

Views:
Criação de uma view

Sintaxe:

```
CREATE VIEW <nome da view> AS
  <instruções SQL>;
```

onde,

<nome da view>
É o nome que irá identificar a tabela virtual, ou seja, a *view*.

Views:
Criação de uma view

Sintaxe:

```
CREATE VIEW <nome da view> AS
  <instruções SQL>;
```

onde,

<instruções SQL>
É uma instrução SELECT. O resultado dessa instrução SELECT irá compor a tabela virtual, ou seja, a *view.*

Views:
Exclusão de uma view

Para excluir uma view, basta utilizar:

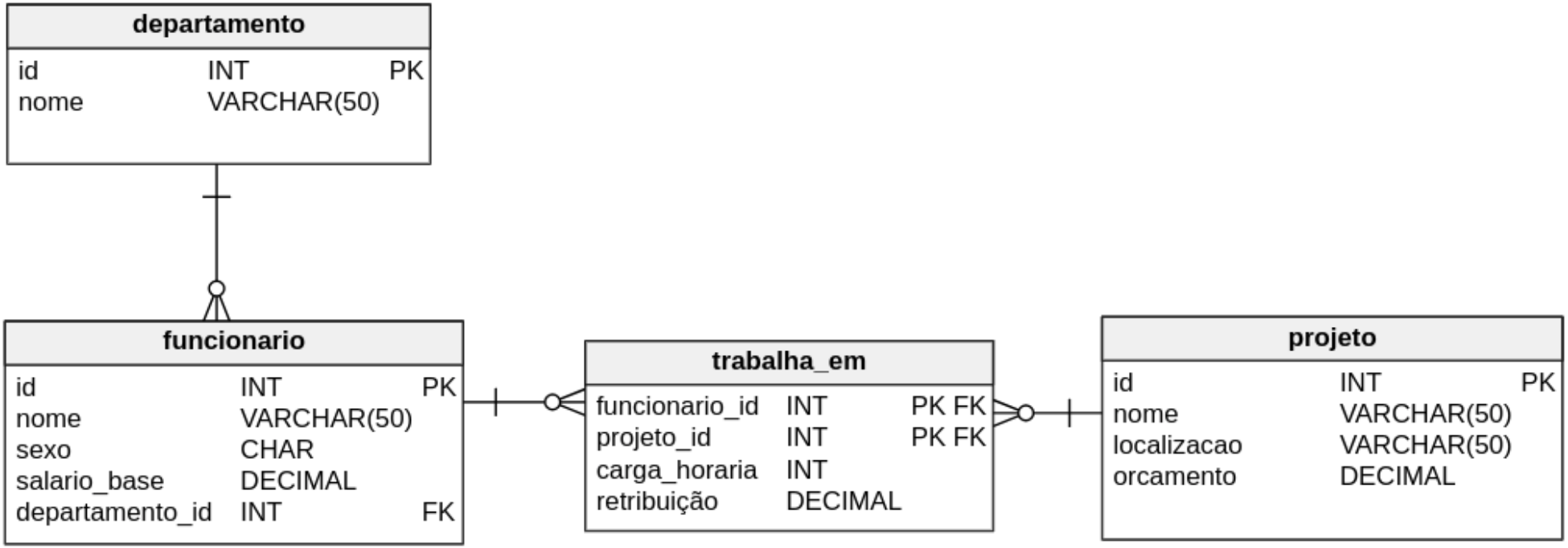
```
DROP VIEW <nome da view>;
```

Exemplo:

```
DROP VIEW minha_view;
```

Views:
Exemplos de criação e uso de views

- Considere a base de dados “empresa” apresentada abaixo:
  - A empresa possui vários departamentos. Cada funcionário está vinculado a um departamento e pode trabalhar em vários projetos. Além do salário base, o funcionário também recebe uma retribuição para cada projeto que trabalha.



Views:
Exemplos de criação e uso de views

- Suponhamos que os projetos são classificados de acordo com seu orçamento e que essa classificação é acessada frequentemente.
  - Seria interessante a criação de uma *view* para a consulta que faz a classificação dos projetos.

Criação da *view*:

```
CREATE VIEW vw_classificacao_projetos AS
    SELECT id, nome, orcamento, IF(orcamento <= 100000, "Pequeno", "Grande") AS classe FROM projeto;
```

Podemos utilizar a *view* como utilizamos uma tabela qualquer:

```
SELECT nome FROM vw_classificacao_projetos WHERE classe = "Grande";

SELECT classe, sum(orcamento) FROM vw_classificacao_projetos GROUP BY classe;
```

Views:
Exemplos de criação e uso de views

- Suponhamos que constantemente precisemos realizar um *join* entre as tabelas de funcionários e projetos.
  - Seria interessante a criação de uma *view* apresentar o resultado desse *join* e utilizarmos como uma tabela sempre que precisarmos.

Criação da *view*:

```
CREATE VIEW vw_funcionario_projeto AS
    SELECT funcionario.id AS f_id, funcionario.nome AS f_nome, projeto.id AS p_id, projeto.nome AS p_nome
    FROM funcionario
    JOIN trabalha_em ON funcionario.id = trabalha_em.id_funcionario
    JOIN projeto ON trabalha_em.id_projeto = projeto.id;
```

Podemos utilizar a *view*:

```
SELECT p_nome AS nome_projeto FROM vw_funcionario_projeto WHERE f_nome = "Alice";
```

Processamento das views

- A maneira como os SGBDs fazem o processamento das *views* pode variar de acordo com o tipo do SGBD.
- O MySQL, quando encontra uma *view* que não tenha funções agregadas, cláusulas DISTINCT, GROUP BY, HAVING, LIMIT, UNIION e subqueries, faz o processamento das consultas na *view* substituindo as consultas na *view* pelas consultas feitas nas tabelas base da *view*.

Por exemplo, considere a seguinte *view*:

```
CREATE VIEW vw_teste(codigo, nome) AS
    SELECT id, nome FROM funcionario WHERE salario_base > 5000;
```

A consulta

```
SELECT * FROM vw_teste;
```

será processada da seguinte forma:

- O * será convertido em “id, nome”, correspondente a “codigo, nome”.
- “vw_teste” será convertido em “funcionario”.
- Será adicionada a cláusula “salario_base > 5000”.

Processamento das views

Assim, a consulta que será de fato realizada na base de dados é:

```
SELECT id, nome FROM funcionario WHERE salario_base > 5000;
```

Um outro exemplo...

```
SELECT codigo, nome FROM vw_teste WHERE codigo < 9999;
```

será convertido em:

```
SELECT id, nome FROM funcionario WHERE id < 9999 AND salario_base > 5000;
```

Uma observação importante:
Para views que utilizam qualquer um dos recursos enumerados anteriormente, o MySQL cria uma tabela temporária com os dados da *view* e, então, as consultas são executadas diretamente na tabela temporária.

Alteração de dados através de views

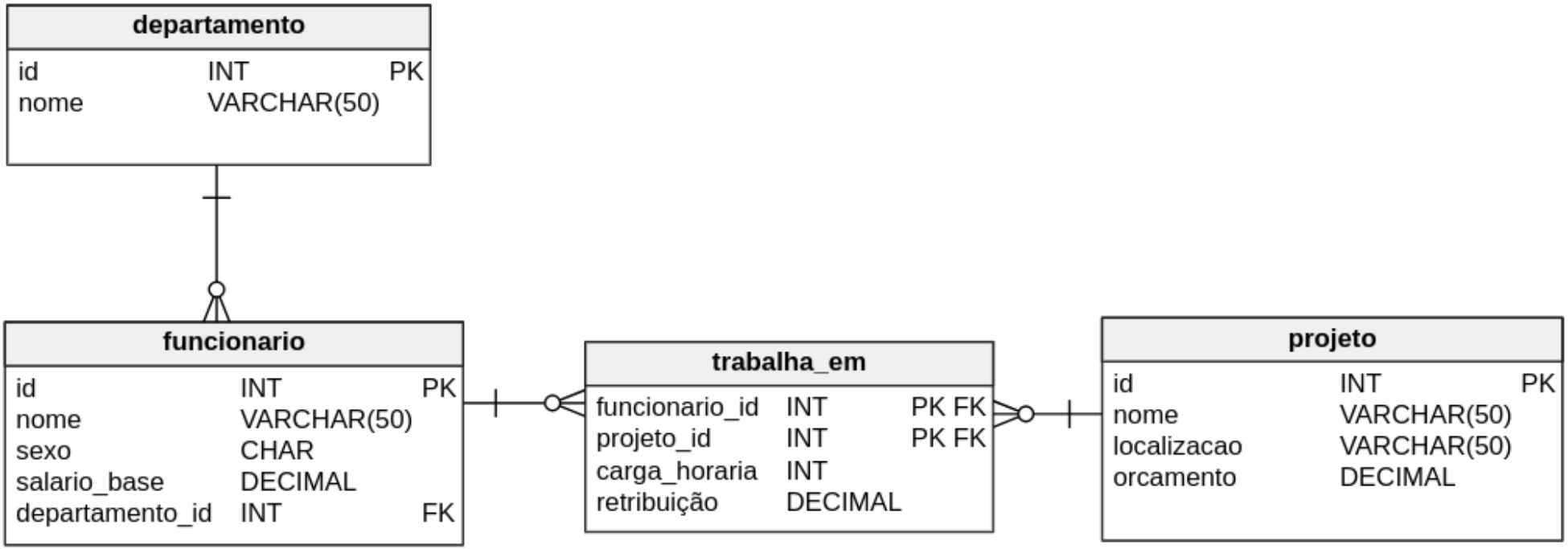
- Os diferentes SGBDs tratam de maneiras específicas as questões das atualizações e inserções em *views*.

- O MySQL assume que uma *view* é atualizável (ou seja, pode ser utilizada com comandos INSERT, UPDATE ou DELETE) em alguns casos.
  - *Views* que podem ser utilizadas com comandos UPDATE ou DELETE não podem possuir funções de agregação, colunas derivadas, cláusulas DISTINCT, GROUP BY, HAVING, subqueries e, além disso, não podem ser *views* que não utilizam tabelas.
  - *Views* que podem ser utilizadas com o comando INSERT não devem possuir nenhuma das características acima e, além disso, não pode ter alguma coluna da tabela base que é NOT NULL e não tem valor *default*.

Exercícios

Considere a base de dados “empresa” apresentada abaixo para realização dos exercícios a seguir.

- A empresa possui vários departamentos. Cada funcionário está vinculado a um departamento e pode trabalhar em vários projetos. Além do salário base, o funcionário também recebe uma retribuição para cada projeto que trabalha.



Exercícios

1) Crie uma *view* que contenha as seguintes colunas:

- O id do funcionário.
- O nome do funcionário.
- O id do projeto em que o funcionário trabalha
- O nome do projeto que o funcionário trabalha.
- A carga horária que o funcionário se dedica ao projeto.
- A retribuição do funcionário por trabalhar no projeto.

2) Crie uma *view* que contenha as seguintes colunas:

- O id do funcionário.
- O nome do funcionário.
- O salário base do funcionário.
- A soma das retribuições do funcionário pelos projetos que ele trabalha.

3) Realize algumas consultas utilizando as *views* criadas acima.

4) Realize algumas alterações nas tabelas originais e verifique se os dados das *views* também foram atualizados.