


Sistema
FIRJAN |  INFORMA, FORMA, TRANSFORMA.



Banco de Dados

Fabício Curvello Gomes

Sistema
FIRJAN |  INFORMA, FORMA, TRANSFORMA.




Junções e Visualizações de Tabelas



INFORMA, FORMA, TRANSFORMA.

Introdução

Junção de Tabela – Possibilita, por meio de um campo comum entre as tabelas existentes, gerar consultas como se as tabelas relacionadas (múltiplas tabelas) na consulta fossem uma única. Isso permite o cruzamento de dados entre tabelas para extração de informações.

Visualização – É a definição de uma tabela virtual a partir dos dados de uma tabela real. É possível, a partir dos dados de uma única tabela real, ter várias visões para facilitar as operações de consultas, ou seja, uma *visão* é a definição de uma consulta predefinida que sempre é executada a partir de uma tabela real.


3

INFORMA, FORMA, TRANSFORMA.

As Tabelas de um BD

A junção de tabelas é a necessidade de unir duas ou mais tabelas para extrair dados de uma consulta como se fossem consultados de uma única tabela.

Por exemplo, vamos trabalhar com duas tabelas em nosso BD, uma de cliente e outra de venda. A tabela *cliente* possui os dados dos clientes e a tabela *venda* possui as vendas realizadas aos clientes. Com a junção das tabelas *cliente* e *venda* pode-se obter uma consulta que mostre o que cada cliente comprou.

4

As Tabelas de um BD (Cont.)

Vamos exemplificar isso:

Já possuímos a tabela **cliente**. Sua estrutura está ilustrada aqui abaixo:

```
mysql> desc cliente;
```

Field	Type	Null	Key	Default	Extra
matricula	int(11)	NO	PRI	NULL	auto_increment
nome	varchar(60)	NO		NULL	
telefone	char(13)	YES		NULL	
email	varchar(30)	YES		NULL	
nascimento	date	NO		NULL	
cpf	char(14)	YES		NULL	

Visualize todo o conteúdo desta tabela;

Agora insira pelo menos 5 novos clientes, utilizando o comando Insert Into, conforme exemplo abaixo:

```
INSERT INTO cliente VALUES
(NULL, 'Francisco Anysio', '(21)9104-0741',
'fanysio@email.com', '1931-04-12',
'043.569.357-01');
```



5

Sistema
FIRJAN



INFORMA, FORMA, TRANSFORMA.

As Tabelas de um BD (Cont.)

Continuando com nosso exemplo de ensino, agora é a vez de criar a tabela venda:

```
CREATE TABLE venda(BOLETO INT(6) NOT NULL PRIMARY
KEY, VALOR DECIMAL(10,2) NOT NULL, VENCTO DATE NOT
NULL, MATRICULA INT NOT NULL);
```

E agora insira dados sobre registros de vendas, **pelo menos 3 registros para cada um dos 5 novos clientes** da outra tabela (lembre-se de utilizar a mesma "MATRICULA" utilizada na tabela cliente para indicar que esta venda é referente aquele cliente). Segue o exemplo de uma inserção de dados:

```
INSERT INTO venda VALUES(230001, 1300.00,
'2010-07-21', 25);
```

Supondo que **25** seja a matrícula do cliente em questão



Junções de Tabelas

Para determinar o relacionamento entre tabelas por meio de junções, é necessário ter no mínimo duas tabelas que possuam algum campo em comum.

Veja a estrutura padrão do comando, e em seguida, o exemplo baseado nas tabelas que acabamos de criar:

```
SELECT <tabela1.campo>, <tabela2.campo>, <...>
      FROM <tabela1>, <tabela2>, <...>
      WHERE <condição> [operador <condição ...>];
```

tabela1.campo – Campo da 1ª tabela a ser apresentado na extração das informações.

tabela2.campo – Campo de 2ª tabela. (O campo da 2ª tabela será diferente do campo da 1ª, e devem estar indicados na ordem em que se deseja vê-los na extração)

tabela1 – Será a indicação da 1ª tabela.

tabela2 – Será a indicação da 2ª tabela (a ordem de definição das tabelas não influencia na extração das informações)

condição – Será a definição do relacionamento por meio do campo em comum existente entre as tabelas que serão relacionadas (normalmente a relação ocorre entre os campos de chave primária e o campo de chave estrangeira)

operador – Será a utilização opcional de um operador lógico ou auxiliar para ampliar a ação de extração de informações.



7

Sistema
FIRJAN



INFORMA, FORMA, TRANSFORMA.

Junções de Tabelas (Cont.)

No exemplo que criamos anteriormente, as tabelas **cliente** e **venda** possuem em comum o campo de código do cliente **MATRICULA**. (na verdade, mesmo se os nomes dos campos fossem diferentes, mas a estrutura e tipos fossem iguais, já seria possível trabalhar com junções de tabelas)

Imagine a necessidade de obter uma consulta dos boletos (tabela **venda**) existentes em cobrança e o nome (tabela **cliente**) dos clientes que estão com estes boletos. Execute a seguinte instrução:

```
SELECT venda.BOLETO, cliente.NOME FROM cliente,
venda WHERE cliente.MATRICULA = venda.MATRICULA;
```



8

Junções de Tabelas (Cont.)

Agora teste os comandos abaixo e analise as diferenças nos resultados:

```
SELECT cliente.NOME, venda.BOLETO, venda.VALOR FROM
cliente, venda WHERE cliente.MATRICULA =
venda.MATRICULA ORDER BY cliente.NOME;
```

```
SELECT cliente.NOME, venda.BOLETO, venda.VALOR FROM
cliente, venda WHERE cliente.MATRICULA =
venda.MATRICULA AND cliente.NOME LIKE 'FRANC%';
```

```
SELECT cliente.NOME, venda.VENCTO FROM cliente, venda
WHERE cliente.MATRICULA = venda.MATRICULA AND
venda.VENCTO LIKE '2012-07%' ORDER BY venda.VENCTO;
```

```
SELECT cliente.NOME, venda.VENCTO FROM cliente, venda
WHERE cliente.MATRICULA = venda.MATRICULA AND
venda.VENCTO LIKE '____-07%' ORDER BY venda.VENCTO;
```



9

Agrupamento de Junções

As consultas relacionadas de mais de uma tabela por junções também podem ser realizadas com GROUP BY.

O agrupamento de informações entre tabelas relacionadas vai depender da forma como a relação é projetada.

Outra informação a ser obtida é a quantidade de boletos em carteira de cada cliente. Para solucionar essa necessidade, deve-se utilizar com a WHERE a cláusula GROUP BY.

Vamos exemplificar isso. Digite o comando abaixo:



```
SELECT cliente.NOME, COUNT(*) FROM cliente, venda
WHERE cliente.MATRICULA = venda.MATRICULA GROUP BY
cliente.NOME;
```

A cláusula GROUP BY faz com que as informações sejam agrupadas pelo nome do cliente.

Será apresentado inclusive a quantidade de títulos que cada cliente possui em carteira.




10



  INFORMA, FORMA, TRANSFORMA.

Agrupamento de Junções (Cont.)

Abaixo aparece uma consulta que mostra o código de cada cliente, seu nome, a quantidade de boletos em carteira por cliente e o total que cada cliente deve. Confira:

```
SELECT cliente.MATRICULA, cliente.NOME, COUNT(*),  
SUM(venda.VALOR) FROM cliente, venda WHERE  
cliente.MATRICULA = venda.MATRICULA GROUP BY  
cliente.NOME;
```

 11

  INFORMA, FORMA, TRANSFORMA.

Apelidos


É possível “apelidar” um campo (coluna) de uma tabela no momento de exibição da saída de um comando `SELECT`.



Isto pode ser útil quando o nome da coluna não é simples de se entender pelo usuário final, ou quando simplesmente é mais conveniente exibir-se outro nome para a referida coluna.

Veja um exemplo:

```
SELECT cliente.MATRICULA, cliente.NOME, COUNT(*)  
AS TITULOS, SUM(venda.VALOR) AS TOTAL FROM  
cliente, venda WHERE cliente.MATRICULA =  
venda.MATRICULA GROUP BY cliente.NOME;
```

Observe o resultado apresentado no *MySQL Command Line Client*.

 12


INFORMA, FORMA, TRANSFORMA.



Exemplos de Agrupamentos e Apelidos

Seguem mais alguns exemplos de comandos com agrupamentos e apelidos. Digite-os e interprete a saída apresentada no *MySQL Command Line Client*:

```
SELECT cliente.NOME AS CLIENTE, COUNT(*) AS  
VENCIDOS FROM cliente, venda WHERE  
cliente.MATRICULA = venda.MATRICULA AND  
VENCTO <= '2010-07-21' GROUP BY cliente.NOME  
ORDER BY cliente.NOME;
```

```
SELECT cliente.NOME, venda.VALOR, venda.VALOR *  
0.10 AS JUROS, venda.VALOR * 1.10 AS TOTAL FROM  
cliente, venda WHERE cliente.MATRICULA =  
venda.MATRICULA AND VENCTO <= '2008-12-31' ORDER  
BY cliente.NOME;
```

13

INFORMA, FORMA, TRANSFORMA.


Visualização de Dados



Visualizar dados é um recurso que possibilita, a partir de uma tabela real existente, criar tabelas virtuais.

As tabelas virtuais (obtidas do recurso de visualização) se parecem com as reais, mas não são as tabelas reais em si. São apenas uma composição em forma de consulta predefinida a partir de uma tabela real.

As tabelas reais possuem os dados cadastrados e por esta razão ocupam espaço em disco, já as virtuais possuem apenas as referências de acesso à consulta das tabelas reais, e por assim dizer, não ocupam espaço em disco.

As visualizações agilizam as operações de consulta, uma vez que concentram em cada tabela virtual os campos que realmente interessam.

14

  INFORMA, FORMA, TRANSFORMA.


Visualização de Dados (Cont.)

O recurso de visualização de dados é útil quando há necessidade de fazer determinadas consultas com frequência, ou seja, é uma forma eficiente de deixar definidas as consultas que serão usadas como relatórios e que ocorrem sempre com grande frequência.



As visões podem ocorrer a partir de uma única tabela, de múltiplas tabelas ou mesmo a partir de outras visões já construídas. Para usar esse recurso, deve-se utilizar basicamente a seguinte instrução:

```
CREATE VIEW <tabela virtual> AS <consulta>;
```

tabela virtual – É a definição do nome da visualização a ser criada.
consulta – É a consulta (*select*) que irá gerar a tabela virtual.



15

  INFORMA, FORMA, TRANSFORMA.


Visualização de Dados (Cont.)

O exemplo seguinte cria uma tabela virtual (uma visão) de nome *visao1* a partir dos dados da tabela *funcionario*, baseando-se nas colunas *NOME*, *DEPTO* e *SALARIO*.



```
CREATE VIEW visao1 AS SELECT NOME, DEPTO, SALARIO  
FROM funcionario;
```

```
SHOW TABLES;
```

```
SELECT * FROM visao1;
```



16



INFORMA, FORMA, TRANSFORMA.


Visualização de Dados (Cont.)

As tabelas virtuais com o recurso de definição de visões podem ser usadas também com junções. Imagine ter uma visão definida para apresentar os títulos em atraso anteriores à data de 01/05/2012.

```
CREATE VIEW visao2 AS SELECT cliente.NOME AS
CLIENTE, COUNT(*) AS VENCIDOS FROM cliente, venda
WHERE cliente.MATRICULA = venda.MATRICULA AND
VENCTO <= '2012-05-01' GROUP BY cliente.NOME ORDER
BY cliente.NOME;

SHOW TABLES;

SELECT * FROM visao2;
```


17

Visualização de Dados (Cont.)

As visões permitem que sejam definidas, a partir de uma visão, outras visões. É aconselhável não criar muitas subdivisões de uma visão existente. Para exemplificar, será criada primeiramente a visão denominada **visao3**, e posteriormente a **visao4** a partir da **visao3**.


Passo 1 – Criar a visão3:



```
CREATE VIEW visao3 AS SELECT cliente.MATRICULA,
cliente.NOME, cliente.TELEFONE, cliente.CPF,
cliente.NASCIMENTO, venda.BOLETO, venda.VALOR,
venda.VENCTO FROM cliente, venda WHERE
cliente.MATRICULA = venda.MATRICULA ORDER BY
cliente.NOME;
```

Passo 2 – Criar a visão4 a partir da visao3:

```
CREATE VIEW visao4 AS SELECT MATRICULA, NOME, BOLETO,
VALOR, VENCTO FROM visao3;
```

Passo 3 – Visualize as tabelas virtuais visao3 e visao4, seus respectivos conteúdos, e analise comparando estruturas e conteúdos com os comandos acima.


18


INFORMA, FORMA, TRANSFORMA.



Visualização de Dados (Cont.)

Uma operação que pode ser realizada é a criação de uma tabela real (física) a partir de uma visão definida.

O exemplo abaixo ilustra isso:

```
CREATE TABLE dadoscli AS SELECT * FROM visao4;
```

19

INFORMA, FORMA, TRANSFORMA.


Visualização de Dados (Cont.)

Para visualizar todas as tabelas e visões existentes em um BD, nada muda. Basta utilizar o já conhecido comando:

```
SHOW TABLES;
```

Para apagar uma tabela virtual (visão), o comando é semelhante ao que já conhecemos, porém, troca-se o termo TABLE por VIEW:

```
DROP VIEW <tabela virtual>;
```


20

Sistema FIRJAN |  INFORMA, FORMA, TRANSFORMA.


Dúvidas?



21

Sistema FIRJAN |  INFORMA, FORMA, TRANSFORMA.

Bibliografia



MySQL 5.1 Interativo (3ª Edição)
José Augusto N.G. Manzano
Ed. Érica

22