Banco de Dados

José Antônio da Cunha

CEFET – RN

Manipulando Dados

A normalização dos dados de um sistema gera várias tabelas. Sendo assim, muitas vezes é necessário ler dados de mais de uma tabela ao mesmo tempo para formar uma informação. Então, a seguir será mostrado como manipular dados de mais de uma tabela.

Manipulando Dados

Unindo Dados de Duas ou mais Tabelas - Union

Union[ALL] – é uma cláusula do comando Select que une (de acordo com certas regras) os dados de duas ou mais tabelas e exibe o resultado dessa união, não apresentando as linhas duplicadas.

Manipulando Dados

Regras para utilizar Union e Union All

- O número de colunas a ser exibido tem que ser o mesmo em todos os comandos select.
- Os datatype tem que ser o mesmo em todas as colunas correspondentes.
- O título das colunas será apresentado de acordo com os títulos do primeiro select.
- Cada select pode ter sua cláusula where.
- A cláusula Order By deve ser colocada no final do último select.

Manipulando Dados

```
CREATE TABLE Cliente
(
        Cod_Cli int identity not null Prmary key,
        Nome_Cli        varchar(50)  not null,
        Sexo_Cli        char(01)      not null Check(Sexo_Cli  IN ('F','M'))
                        DEFAULT 'F',
        Renda_Cli       decimal (10,2) Check(Renda_Cli >=0) DEFAULT 0,
        RG_Cli          char(12)      not null Unique
)
```

Manipulando Dados

```
CREATE TABLE Funcionario
(
        Cod_Func        int not null Primary key,
        Nome_Func       varchar(50)  not null,
        Sexo_Func       char(01)    not null Check(Sexo_Func  IN ('F', 'M'))
                        DEFAULT 'F',
        Sal_Func        decimal(10,2)   not null Check(Sal_Func >= 0 ),
        Data_CadFunc    smalldatetime not null Default Getdate()
)
```

Manipulando Dados

Cod_Cli	Nome_Cli	Sexo_Cli	Renda_Cli	RG_Cli
1	João Carlos	M	5000.00	123456
2	Daniel	M	6000.00	145678
3	Helena	F	6000.00	564333
4	Roberta	F	5000.00	564322
5	Renata	F	3000.00	654321
6	Giovanna	F	3500.00	564322

Tabela Cliente

Manipulando Dados

Cod_Func	Nome_Func	Sexo_Func	Sal_Func	Data_CadFunc
1	Salvador	M	7000.00	03/01/1995
2	Antônio	M	6000.00	06/05/1998
3	Antonieta	F	6500.00	07/08/1998
4	Eduardo	M	4000.00	06/08/2000
5	Leny	F	3000.00	06/08/2001

Tabela Funcionario

Manipulando Dados

Com base nas tabelas criadas anteriormente, escreva uma consulta que exiba o tipo se é ‘cliente’ ou ‘funcionário’, o nome, o sexo, a data de cadastro e o RG.

Manipulando Dados

```
SELECT Tipo = 'Funcionário',
       Nome = Nome_Func,
       Sexo   = Sexo_Func,
       Salário = Sal_Func,
       [Data de Cadastro] = Data_CadFunc,
       RG = null
FROM Funcionario
UNION [ALL]
SELECT 'Cliente',
```

Manipulando Dados

```
        Nome_Cli,
        Sexo_Cli,
        Renda_Cli,
        Null,
        RG_Cli
FROM Cliente
ORDER BY tipo
```

Manipulando Dados

Observe que as tabelas não têm as mesma estrutura, ou seja, têm colunas que existem na primeira tabela e não existem na segunda e vice-versa. Mas se você precisar exibir o valor de uma coluna que existe em uma tabela, mas não existe na outra, no select da tabela que não tem essa coluna coloque o valor null ou um outro valor qualquer, desde que o valor tenha o mesmo datatype da coluna que ocupa a respectiva posição no select.

Manipulando Dados

Associando dados de de duas ou mais Tabelas

Associar uma ou mais tabelas é o mesmo que ligá-las por meio de um ou mais colunas que elas tenham em comum, com o objetivo de obter dados relacionados entre essas tabelas em questão.

- **INNER JOIN** ou **JOIN** – apenas dados Relacionados
  - Natural_Join
  - Equi-Join
  - Self Join

Manipulando Dados

Associando dados de de duas ou mais Tabelas

- **OUTER JOIN** – Dados Relacionados e não Relacionados
  - Left Outer Join ou Left Join
  - Right Outer Join ou Right Join
  - Full Outer Join ou Full Join
- **CROSS JOIN** – Produto Cartesiano

Manipulando Dados

Associando dados de duas ou mais Tabelas

Suponha que você tenha uma tabela Pai e uma tabela Filho com a estrutura e os dados apresentados em seguida:

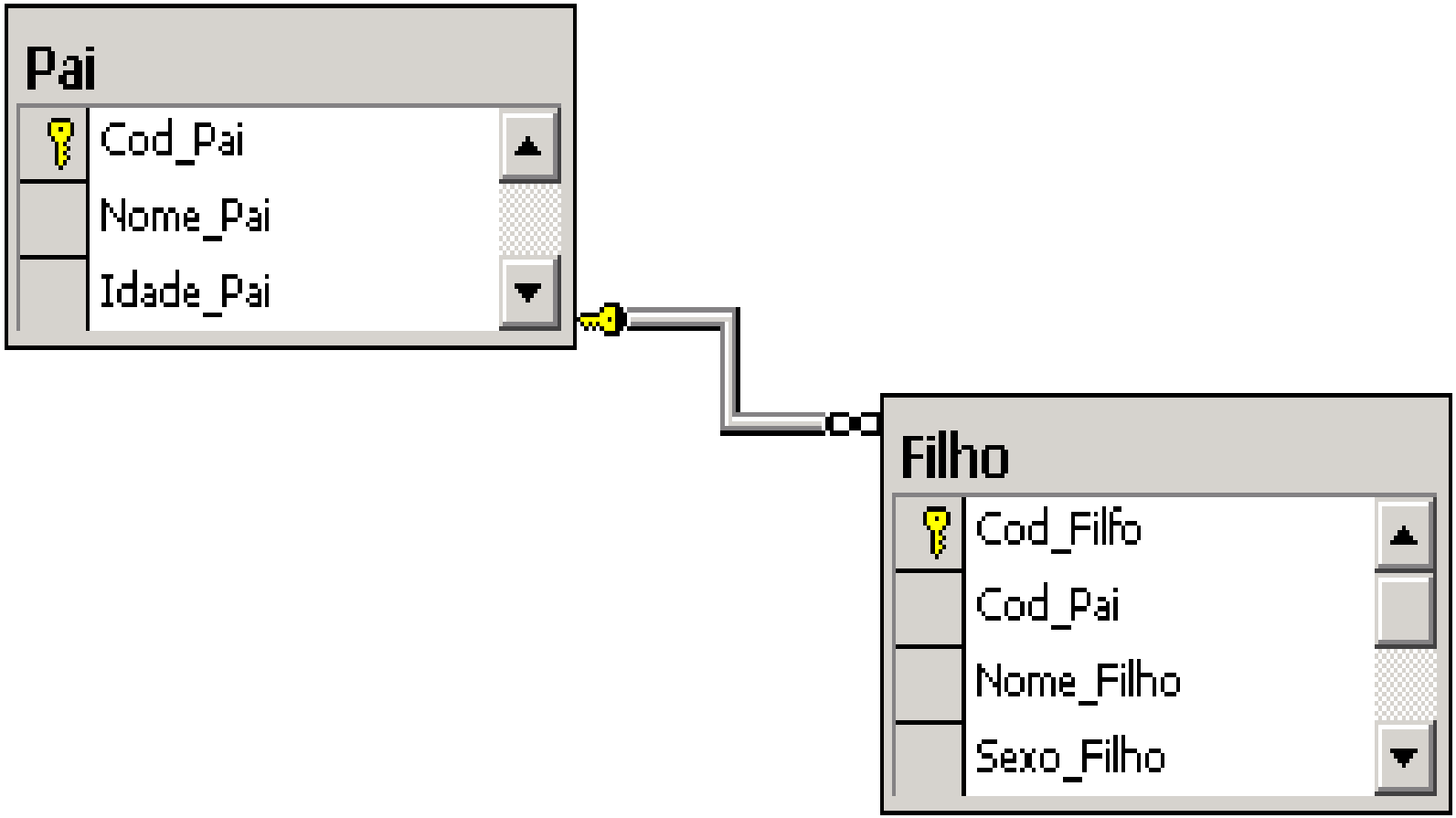
```
CREATE TABLE Pai
(
        Cod_Pai int identity not null Primary key,
        Nome_Pai  char(30)  not null,
        Idade_Pai  tinyint      not null
)
```

Manipulando Dados

Associando dados de de duas ou mais Tabelas

```
CREATE TABLE Filho
(
    Cod_Filho int identity not null Primary key,
    Cod_Pai int            not null References Pai(Cod_pai),
    Nome_Filho  char(30)  not null,
    Sexo_Filho  tinyint      not null Check(Sexo_Filho IN ('F', 'M'))
)
```

Manipulando Dados



Manipulando Dados

Cod_Pai	**Nome_Pai**	**Idade_Pai**
1	Daniel de Souza Leão Sobrinho	62
2	João Carlos da Silva	38
3	Fernando de Oliveira	36

Cod_Filho	Cod_Pai	Nome_Filho	Sexo_Filho
1	1	Renata de Oliveira Leão	F
2	1	Fernando de Oliveira Leão	M
3	1	Roberta de Oliveira Leão	F
4	1	Jairo de Oliveira Leão	M
5	2	Giovanna da Silva	F
6	3	Lucas Ribeiro Oliveira	M
7	3	Helder Ribeiro Oliveira	M

Manipulando Dados

INNER JOIN – Cláusula INNER JOIN tem como objetivo selecionar de duas ou mais tabelas apenas os dados relacionados entre todas elas. Um INNER JOIN pode ser:

- Um Natural Join ou
- Um Eqüi Join

Manipulando Dados

Ex.: INNER JOIN – Natural Join

Suponha que você precise obter o nome de todos os pais e o nome e o sexo de todos os filhos que cada pai possui.

```
SELECT Pai.Nome_Pai          as [Nome do Pai],
       Filho.Nome_Filho      as [Nome do Filho],
       Filho.Sexo_Filho      as Sexo
FROM Pai INNER JOIN Filho
ON (Pai.Cod_Pai = Filho.Cod_Pai)
```

Manipulando Dados

Ex.: INNER JOIN – Equi Join

Suponha que você deseje fazer uma consulta, que retorne todas as linhas relacionadas das tabelas Pai e Filho e apresente todas as colunas das tabelas envolvidas.

```
SELECT *

FROM Pai INNER JOIN Filho

ON Pai.Cod_Pai = Filho.Cod_Pai
```

Manipulando Dados

Se você selecionar no seu comando com INNER JOIN apenas colunas que não se repetem, esse INNER JOIN será um **Natural Join**:

Manipulando Dados

Self Join: é um Join de uma tabela consigo mesma. Este tipo de join apresenta dados de um auto-relacionamento.

```
CREATE TABLE Emprgado (
        Cod_Emp int not null,
        Nome_Emp     varchar(50) not null,
        Cod_Chefe int null,
        Constraint pk_Emp Primary key (Cod_Emp),
        Constraint fk_Emp Foreign Key (Cod_Chefe)
        References (Cod_Emp))
```

Manipulando Dados

Manipulando Dados

A tabela Empregado armazena dados dos empregados e o seu chefe, sendo que um chefe só pode ser chefe se antes for um empregado.

Se você observar, a coluna Cod_Chefe aceita valores nulos, para que você possa inserir empregados que não tenhas chefe algum, ou que sejam seu próprio chefe.

Manipulando Dados

Cod_Emp	Nome_Emp	Cod_Chefe
1	João	Null
2	Matheus	1
3	Lucas	1
4	Pedro	2
5	Thiago	2
6	José	2
7	Tânia	1
8	Joana	3
9	Rosana	3
10	Maria	4

Manipulando Dados

Ex.: Escreva uma consulta que obtenha o nome de todos os empregados e de seus chefes.

```
SELECT C.Nome_Emp AS Chefe,
            E.Nome_Emp AS Empregado
FROM Empregado C, Empregado E
WHERE C.Cod_Emp = E.Cod_Chefe
ORDER BY 1
```

Manipulando Dados

Left Join: a cláusula LEFT JOIN é utilizada para mostrar todos os dados da tabela escrita do lado esquerdo do join.

Ex.: Insira um novo registro na tabela Pai (8, ‘Jairo Cabral’, 45) e faça a seguinte consulta:

Escreva uma consulta que exiba os dados de todos os pais relacionando esses dados com os respectivos filhos, mostrando também os dados dos pais que ainda estão sem filhos registrados.

Manipulando Dados

```
SELECT Pai.Nome_Pai as [Nome do Pai],
            Filho.Nome_Filho as [Nome do Filho],
            Filho.Sexo_Filho   as Sexo
FROM Pai Left Outer Join Filho
ON Pai.Cod_Pai = Filho.Cod_Pai
```

Se você precisasse obter como resposta apenas os dados dos pais para os quais você ainda não inseriu nenhum filho, bastaria acrescentar a cláusula WHERE como segue:

WHERE Filho.Nome_Filho IS NULL

Manipulando Dados

Right Join: a cláusula Right Join retorna todos os registros da tabela do lado direito do join.

Ex.: suponha que por algum motivo foi necessário inserir alguns filhos na tabela Filho, para um pai que não existe na tabela Pai. Então, para fazer isso, você "desligou" a constraint de chave estrangeira inseriu os dados inconsistentes e "religou" a chave estrangeira, dizendo para o sistema não checar dados já existentes.

Insira três novos filho na tabela Filho, cujo Cod_Pai seja 1000.

Manipulando Dados

Se precisasse obter como resposta os dados dos filhos e seus pais, mostrando inclusive os filhos órfãos, você escreveria o seguinte comando:

```
SELECT Pai.Nome_Pai          as [Nome do Pai],
       Filho.Nome_Filho      as [Nome do Filho],
       Filho.Sexo_Filho      as Sexo
FROM Pai RIGHT OUTER JOIN Filho
ON (Pai.Cod_Pai = Filho.Cod_Pai)
```

Manipulando Dados

Full Join: a cláusula Full Join retorna todos os dados relacionados e não relacionados, o seja, registros que não tenham correspondência à direita, serão retornados e, também os que não tenham correspondência à esquerda, serão retornados, e vice-versa.

```
SELECT Pai.Nome_Pai        as [Nome do Pai],
       Filho.Nome_Filho    as [Nome do Filho],
       Filho.Sexo_Filho    as Sexo
FROM Pai FULL OUTER JOIN Filho
ON (Pai.Cod_Pai = Filho.Cod_Pai)
```

Manipulando Dados

Cross Join: é o produto cartesiano. O CROSS JOIN relaciona todas as linhas da tabela A com todas as linhas da tabela B.

Cod_Mat	Nome_Mat
1	Caderno
2	Lápis
3	Borracha
4	Caneta

Material

Num_Ped	Data_Ped
1	24/06/2002
2	06/08/2002

Pedido

```
Select Pedido.Num_Ped,
        material.Nome_Mat
FROM Pedido CROSS JOIN Material
```

Manipulando Dados

Cod_Pai	Nome_Pai
1	Daniel
2	João Carlos
3	Fernando
4	Jairo

Tabela Pai

Manipulando Dados

Cod_Filho	Cod_Pai	Nome_Filho	Sexo_Filho
1	1	Renata	F
2	1	Fernando	M
3	1	Roberta	F
4	1	Jairo	M
5	2	Giovanna	F
6	3	Lucas	M
7	3	Helder	M

Tabela Filho

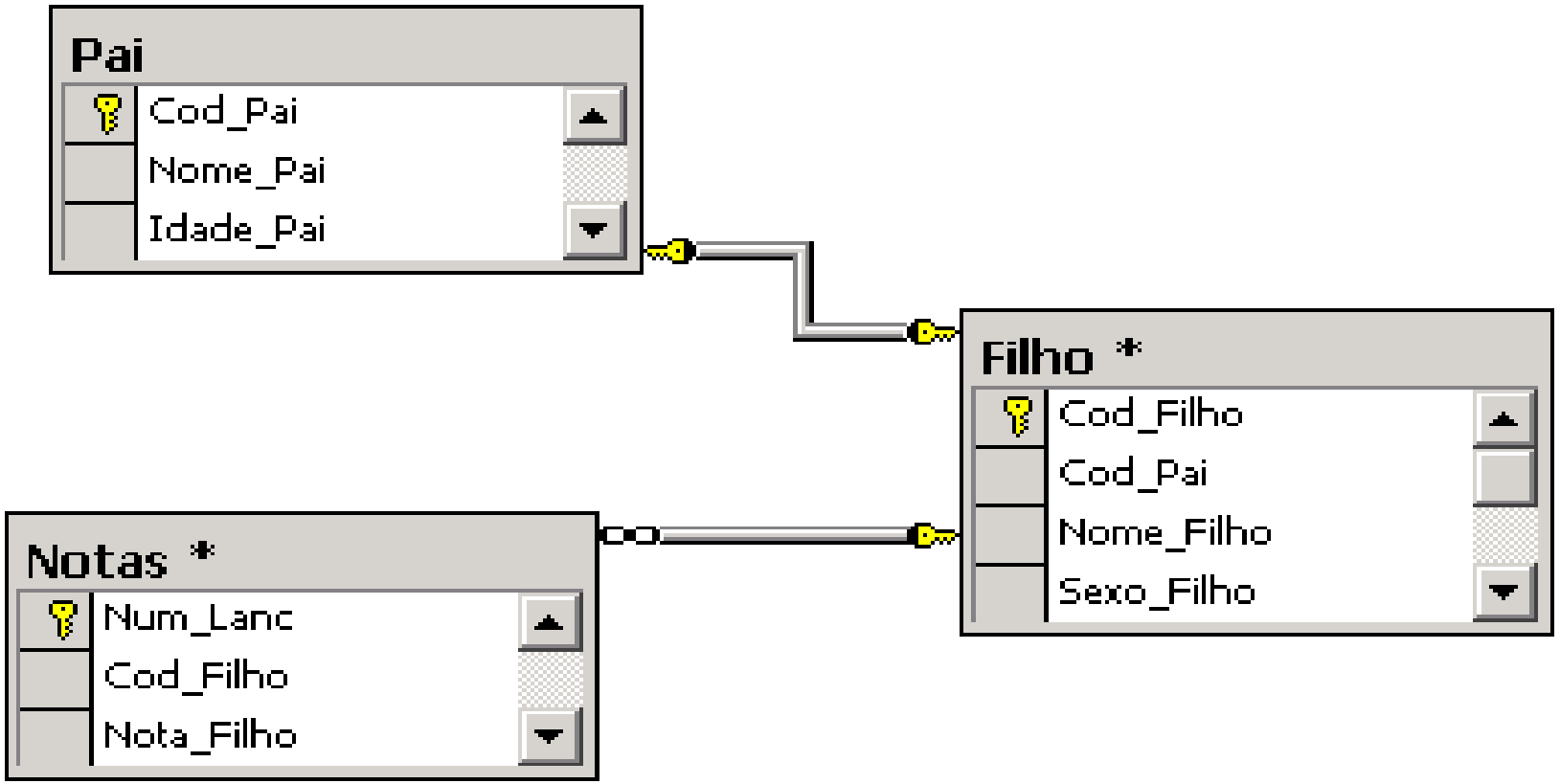
Manipulando Dados

Num_Lanc	Cod_Filho	Nota_Filho
1	1	10,00
2	1	9,00
3	1	10,00
4	2	8,00
5	2	7,00
6	3	5,00
7	3	10,00

Tabela Notas

Manipulando Dados

Joins com mais de duas tabelas



Manipulando Dados

Ex.: Faça uma consulta que retorne todos os filhos e seus respectivos pais e suas notas.

```
SELECT Distinct Pai.Nome_Pai as [Nome do Pai],
      Filho.Nome_Filho as [Nome do Filho],
      Notas.Nota_Filho  as [Nota do Filho]
FROM Pai INNER JOIN Filho
    ON (Pai.Cod_Pai = Filho.Cod_Pai)
    INNER JOIN Notas
    ON (Filho.Cod_Filho = Notas.Cod_Filho)
```

Manipulando Dados

Ex.: Faça uma consulta que retorne todos os filhos e seus respectivos pais e suas notas. Inclusive os filhos sem notas.

```
SELECT Distinct Pai.Nome_Pai as [Nome do Pai],
      Filho.Nome_Filho as [Nome do Filho],
      Notas.Nota_Filho  as [Nota do Filho]
FROM Pai INNER JOIN Filho
    ON (Pai.Cod_Pai = Filho.Cod_Pai)
    LEFT JOIN Notas
    ON (Filho.Cod_Filho = Notas.Cod_Filho)
```

Manipulando Dados

Ex.: Faça uma consulta que retorne todos os filhos e seus respectivos pais e suas notas, mostre também os pais com filhos e filhos sem notas e os pais sem filhos.

```
SELECT Distinct Pai.Nome_Pai as [Nome do Pai],
      Filho.Nome_Filho as [Nome do Filho],
      Notas.Nota_Filho  as [Nota do Filho]
FROM Pai LEFT JOIN Filho
    ON (Pai.Cod_Pai = Filho.Cod_Pai)
    LEFT JOIN Notas
    ON (Filho.Cod_Filho = Notas.Cod_Filho)
```

Manipulando Dados

Ex.: Faça uma consulta que retorne pai com filho e filho com nota, e filho sem pai e sem nota.

```
SELECT Distinct Pai.Nome_Pai as [Nome do Pai],
      Filho.Nome_Filho as [Nome do Filho],
      Notas.Nota_Filho  as [Nota do Filho]
FROM Pai FULL JOIN Filho
    ON (Pai.Cod_Pai = Filho.Cod_Pai)
    FULL JOIN Notas
    ON (Filho.Cod_Filho = Notas.Cod_Filho)
```

Manipulando Dados

Update com Join – muitas vezes será necessário alterar os dados de uma tabela com base na existência ou não de dados relacionados em outra tabela. Por exemplo, você poderia desejar aumentar a idade para mais um ano de todos os pais que possuem filhos.

```
Update Pai Set Idade_Pai = Idade_Pai + 1

FROM Pai INNER JOIN Filho

ON (Pai.Cod_Pai = Filho.Cod_Pai)
```

```
Update Pai Set Idade_Pai = Idade_Pai + 1

FROM Pai, Filho

WHERE (Pai.Cod_Pai = Filho.Cod_Pai)
```

Manipulando Dados

Delete com Join – suponha que você precisasse excluir os filhos que não receberam um respectivo pai na tabela Pai.

```
DELETE Filho

FROM Pai RIGHT JOIN Filho

ON (Pai.Cod_Pai = Filho.Cod_Pai)

WHERE Pai.Cod_Pai IS NULL
```

Manipulando Dados

Join com chave estrangeira composta – suponha que você tenha duas tabelas como as apresentadas a seguir. A primeira tem uma chave primária composta de duas colunas e a segunda, que se relaciona com a primeira, tem uma chave estrangeira composta das mesmas duas colunas:

Manipulando Dados

```
CREATE TABLE Tab_A
(
        ColunaA1        datatype Not Null,
        ColunaA2        datatype Not Null,
        ColunaA3        datatype Not Null,
        Constraint pk_TabA Primary Key (ColunaA1, ColunaA2)
)
```

Manipulando Dados

```
CREATE TABLE Tab_B
(
        ColunaB1        datatype Not Null Primary Key,
        ColunaA1        datatype Not Null,
        ColunaA2        datatype Not Null,
        Constraint FK_TabB Foreign Key (ColunaA1, ColunaA2)
        References Tab_A (ColunaA1, ColunaA2)
)
```

Manipulando Dados

Para obter dados relacionados entre as duas tabelas, o join deve ser escrito desta forma:

```
SELECT Tab_A.ColunaA1, Tab_A.ColunaA2, Tab_A.ColunaA3,
       Tab_B.ColunaB1
FROM Tab_A INNER JOIN Tab_B
ON (Tab_A.ColunaA1 = Tab_B.ColunaA1) AND
   (Tab_A.ColunaA2 = Tab_B.ColunaA2)
```