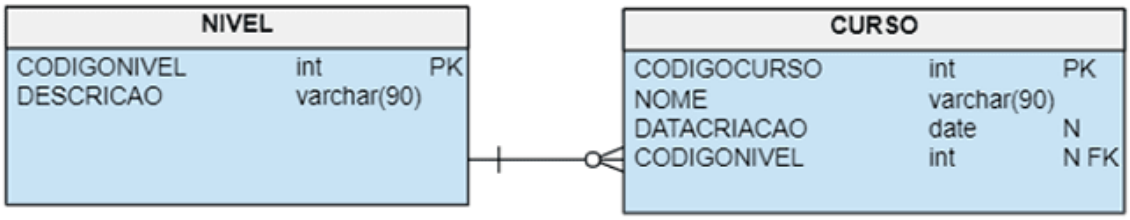
Operar consultas envolvendo junções interior e exterior

OPERAÇÃO DE JUNÇÃO DE TABELAS

A junção de tabelas é uma das operações mais importantes criadas pelo modelo relacional de banco dados. Na definição matemática, o resultado da junção de duas tabelas é um subconjunto do produto cartesiano entre essas tabelas. A especificação desse subconjunto é feita por uma condição de junção entre colunas das duas tabelas.

Veremos que existem dois tipos de junção: interna, denominada INNER JOIN; e externa, denominada OUTER JOIN que, por sua vez, pode ser LEFT, FULL ou RIGHT OUTER JOIN.




```
CREATE TABLE NIVEL (
CODIGONIVEL int NOT NULL,
DESCRICAO varchar(90) NOT NULL,
CONSTRAINT CHAVEPNIVEL PRIMARY KEY (CODIGONIVEL)
);

CREATE TABLE CURSO (
CODIGOCURSO int NOT NULL,
NOME varchar(90) NOT NULL UNIQUE,
DATACRIACAO date NULL,
CODIGONIVEL int NULL,
CONSTRAINT CHAVEPCURSO PRIMARY KEY (CODIGOCURSO)
);

ALTER TABLE CURSO ADD FOREIGN KEY(CODIGONIVEL) REFERENCES NIVEL;

INSERT INTO NIVEL (CODIGONIVEL, DESCRICAO) VALUES (1,'Graduação');
INSERT INTO NIVEL (CODIGONIVEL, DESCRICAO) VALUES
(2,'Especialização');
```

```
INSERT INTO NIVEL (CODIGONIVEL, DESCRICAO) VALUES (3,'Mestrado');

INSERT INTO NIVEL (CODIGONIVEL, DESCRICAO) VALUES (4,'Doutorado');

INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO,CODIGONIVEL) VALUES
(1,'Sistemas de Informação','19/06/1999',1);

INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO,CODIGONIVEL) VALUES
(2,'Medicina','10/05/1990',1);

INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO,CODIGONIVEL) VALUES
(3,'Nutrição','19/02/2012',NULL);

INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO,CODIGONIVEL) VALUES
(4,'Pedagogia','19/06/1999',1);

INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO,CODIGONIVEL) VALUES
(5,'Saúde da Família','10/09/1999',3);

INSERT INTO CURSO (CODIGOCURSO,NOME,DATACRIACAO,CODIGONIVEL) VALUES
(6,'Computação Aplicada','10/09/1999',NULL);


SELECT * FROM NIVEL;

SELECT * FROM CURSO;
```

OPERAÇÃO DE PRODUTO CARTESIANO

Em termos estruturais, a tabela NIVEL possui duas colunas e quatro registros. De forma semelhante, a tabela CURSO possui quatro colunas e seis registros. Você pode chegar a essa conclusão ao analisar o script anterior.

Vamos, agora, executar a seguinte consulta SQL:

SELECT * FROM CURSO, NIVEL;

Junção interna

Estamos interessados em obter informações úteis para o nosso usuário. Perceba que, por exemplo, o curso de Sistemas de Informação está classificado como um curso pertencente ao nível de graduação.

Como chegamos a essa conclusão? Avalie o conteúdo das linhas 14 e 18 do script que insere informações nas tabelas NIVEL e CURSO.

Note que, se examinarmos cada linha como um fato, vamos perceber que somente a primeira corresponde à realidade cadastrada no banco de dados. Não por coincidência, perceba que os valores das colunas CODIGONIVEL são os mesmos. Nas três últimas, diferentes.

ra recuperar as linhas que correspondem à realidade cadastrada no banco de dados, você pode executar o comando a seguir:

```
SELECT *
FROM CURSO,NIVEL
WHERE NIVEL.CODIGONIVEL=CURSO.CODIGONIVEL;
```

Perceba que foram recuperadas as linhas que de fato relacionam cursos aos seus respectivos níveis. No entanto, a forma mais usada para retornar os mesmos resultados é com o auxílio da cláusula de junção interna, o qual possui sintaxe básica conforme a seguir:

SELECT *
FROM TABELA1 [INNER] JOIN TABELA2 ON (CONDIÇÃOJUNÇÃO) [USING (COLUNA_DE_JUNÇÃO)]

Veja, a seguir, o código SQL correspondente ao nosso exemplo:

```
SELECT *
FROM CURSO INNER JOIN NIVEL ON(NIVEL.CODIGONIVEL=CURSO.CODIGONIVEL);
```

Note também que é possível declarar a cláusula USING especificando a coluna alvo da junção. No caso do nosso exemplo, a coluna CODIGONIVEL. A consulta pode ser reescrita conforme a seguir:

```
SELECT *
FROM CURSO INNER JOIN NIVEL USING(CODIGONIVEL);
```

Se desejarmos exibir o código e o nome do curso, além do código e o nome do nível, podemos, então executar o código a seguir:

```
SELECT CURSO.CODIGOCURSO, CURSO.NOME
FROM CURSO INNER JOIN NIVEL USING(CODIGONIVEL);
```

```
SELECT CURSO.CODIGOCURSO,CURSO.NOME,NIVEL.CODIGONIVEL,NIVEL.DESCRICAO
FROM CURSO INNER JOIN NIVEL ON(NIVEL.CODIGONIVEL=CURSO.CODIGONIVEL);
```

Perceba que, no comando SELECT, usamos uma referência mais completa: NOMETABELA.NOMECOLUNA. Essa referência só é obrigatória para a coluna CODIGONIVEL, uma vez que é necessário especificar de qual tabela o SGBD irá buscar os valores. No entanto, em termos de organização de código, é interessante usar esse tipo de referência para cada coluna.

Finalmente, observe que poderíamos também, no contexto da consulta, renomear as tabelas envolvidas, deixando o código mais elegante e legível, conforme a seguir:

```
SELECT C.CODIGOCURSO,C.NOME,N.CODIGONIVEL,N.DESCRICAO
FROM CURSO C INNER JOIN NIVEL N ON (N.CODIGONIVEL=C.CODIGONIVEL);
```

Junção externa

O resultado da consulta anterior exibe somente os cursos para os quais há registro de informação sobre o nível associado a eles. E se quiséssemos incluir na listagem todos os registros da tabela CURSO?

Para incluir no resultado da consulta todas as ocorrências da tabela CURSO, podemos usar a cláusula LEFT JOIN (junção à esquerda). Nesse tipo de junção, o resultado contém todos os registros da tabela declarada à esquerda da cláusula JOIN, mesmo que não haja registros correspondentes na tabela da direita. Em especial, quando não há correspondência, os resultados são retornados com o valor NULL.

```
SELECT C.CODIGOCURSO,C.NOME,N.CODIGONIVEL,N.DESCRICAO

FROM CURSO C LEFT JOIN NIVEL N ON (N.CODIGONIVEL=C.CODIGONIVEL);
```

Perceba também que, de forma semelhante, poderíamos ter interesse em exibir todos os registros da tabela à direita da cláusula JOIN. Em nosso exemplo, a tabela NIVEL. A cláusula RIGHT JOIN (junção à direita) é usada para essa finalidade. Nesse tipo de junção, o resultado contém todos os registros da tabela declarada à direita da cláusula JOIN, mesmo que não haja registros correspondentes na tabela da esquerda. Em especial, quando não há correspondência, os resultados são retornados com o valor NULL.

```
SELECT C.CODIGOCURSO,C.NOME,N.CODIGONIVEL,N.DESCRICAO

FROM CURSO C RIGHT JOIN NIVEL N ON (N.CODIGONIVEL=C.CODIGONIVEL);
```

Note que todos os registros da tabela NIVEL aparecem no resultado. As quatro primeiras linhas do resultado da consulta correspondem aos registros que efetivamente estão relacionados à tabela CURSO. As duas últimas linhas são registros que não estão relacionados a qualquer curso existente no banco de dados.

Perceba que o mesmo resultado pode ser obtido se usarmos junção à esquerda e junção à direita, alternando a posição das tabelas envolvidas. Com isso, queremos dizer que TABELA1 LEFT JOIN TABELA2 é equivalente a TABELA2 RIGHT JOIN TABELA1.

```
SELECT C.CODIGOCURSO,C.NOME,N.CODIGONIVEL,N.DESCRICAO
FROM CURSO C LEFT JOIN NIVEL N ON (N.CODIGONIVEL=C.CODIGONIVEL);

SELECT C.CODIGOCURSO,C.NOME,N.CODIGONIVEL,N.DESCRICAO
FROM NIVEL N LEFT JOIN CURSO C ON (N.CODIGONIVEL=C.CODIGONIVEL);
```

Outro tipo de junção externa, denominada **FULL OUTER JOIN** (junção completa), apresenta todos os registros das tabelas à esquerda e à direita, mesmo os registros não relacionados. Em outras palavras, a tabela resultante exibirá todos os registros de ambas as tabelas, além de valores NULL no caso dos registros sem correspondência.

```
SELECT C.CODIGOCURSO,C.NOME,N.CODIGONIVEL,N.DESCRICAO
FROM CURSO C FULL OUTER JOIN NIVEL N ON
(N.CODIGONIVEL=C.CODIGONIVEL);
```