Banco de Dados I

Aula11

SQL - JOIN

Prof. MSc. Adalto Selau Sparremberger


assparremberger@senacrs.com.br

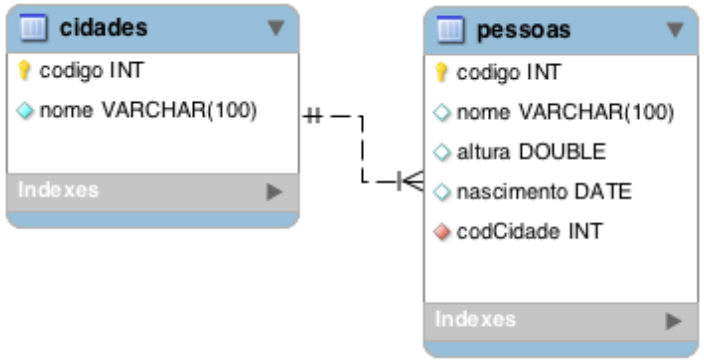

@adaltoss


/assparremberger

Fecomércio RS | Senac

Recaptulando...

```
CREATE TABLE cidades  (
    codigo INT NOT NULL AUTO_INCREMENT ,
    nome VARCHAR(100) ,
    PRIMARY KEY ( codigo )
);
```




```
CREATE TABLE pessoas (
    codigo INT NOT NULL PRIMARY KEY AUTO_INCREMENT ,
    nome VARCHAR(100) NOT NULL ,
    altura DOUBLE ,
    nascimento DATE DEFAULT `1970-12-25` ,
    codCidade INT ,
    FOREIGN KEY ( codCidade ) REFERENCES cidades( codigo )
);
```

Mas antes... Um pouco mais de:

SELECT

DISTINCT

- A instrução SELECT DISTINCT é usada para retornar apenas valores distintos (diferentes).
- Dentro de uma tabela, uma coluna geralmente contém muitos valores duplicados; e, às vezes, você deseja listar apenas os diferentes valores (distintos).
- Ex: Retornar os valores de altura das pessoas sem repetir valores, retornando apenas um registro de cada valor diferente.

```
SELECT DISTINCT altura FROM pessoa
```

ORDER BY

- A palavra-chave ORDER BY é usada para classificar o conjunto de resultados em ordem crescente ou decrescente.
- A palavra-chave ORDER BY classifica os registros em ordem crescente por padrão. Para classificar os registros em ordem decrescente, use a palavra-chave DESC.
- É utilizado para ordenar por uma determinada coluna, linhas de uma tabela resultante de uma consulta:
- Ex: ordenar as pessoas por ordem de nome:
  - SELECT * FROM pessoa ORDER BY nome
- Ex: ordenar as pessoas por ordem decrescente de nome:
  - SELECT * FROM pessoa ORDER BY nome DESC

TOP / LIMIT

- A cláusula SELECT TOP é usada para especificar o número de registros a serem retornados.
- A cláusula SELECT TOP é útil em tabelas grandes com milhares de registros. O retorno de um grande número de registros pode afetar o desempenho.
- Ex: Retornar apenas os 10 primeiros nomes de pessoas cadastradas:
  - MySQL: SELECT nome FROM pessoa LIMIT 10
  - SQL Server: SELECT TOP 10 nome FROM pessoa
  - Oracle: SELECT nome FROM pessoa ROWNUM <= 10

MIN(), MAX(), AVG(), COUNT(), SUM()

- A função MIN() retorna o menor valor da coluna selecionada.
- A função MAX() retorna o maior valor da coluna selecionada.
- A função COUNT() retorna o número de linhas que correspondem a um critério especificado.
- A função AVG() retorna o valor médio de uma coluna numérica.
- A função SUM() retorna a soma total de uma coluna numérica.

IN

- O operador IN permite especificar vários valores em uma cláusula WHERE.
- O operador IN é uma abreviação de várias condições OR.
- EX: retornar os nomes das pessoas que o valor da altura seja 1.75, 1.8 ou 1.85, somente.
  - SELECT nome FROM pessoa WHERE altura IN (1.75,1.8 , 1.85)

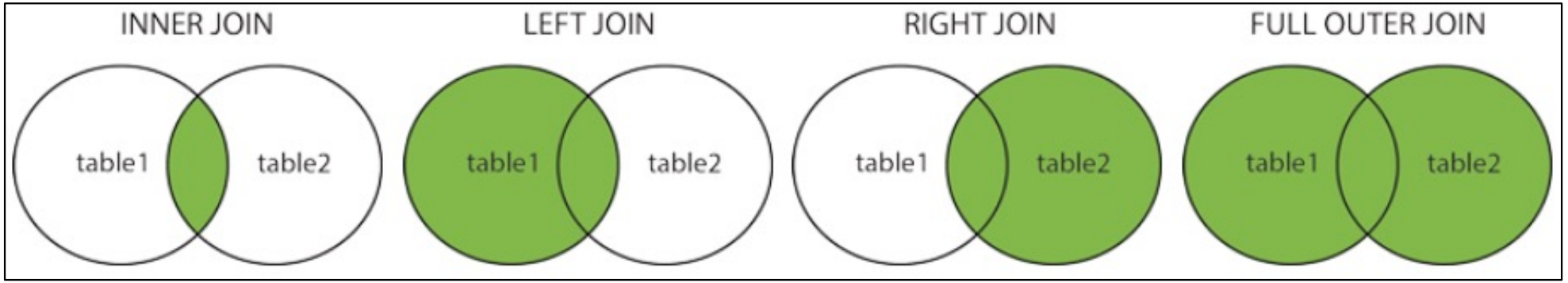
JOIN

- Uma cláusula JOIN é usada para combinar linhas de duas ou mais tabelas, com base em uma coluna relacionada entre elas.
- Ex: Retornar nome das pessoas e o nome da respectiva cidade desta pessoa

```
SELECT pessoa.nome, cidade.nome
FROM pessoa
INNER JOIN cidade
ON pessoa.codCidade = cidade.codigo
```

Tipos de JOINs

- **(INNER) JOIN** : Retorna os registros que possuem valores correspondentes nas duas tabelas;
- **LEFT (OUTER) JOIN**: Retorna todos os registros da tabela esquerda e os registros correspondentes da tabela direita (que se relacionam com a tabela direita);
- **RIGHT (OUTER) JOIN**: Retorna todos os registros da tabela da direita e os registros correspondentes da tabela da esquerda (que se relacionam com a tabela direita);
- **FULL (OUTER) JOIN**: Retorna todos os registros quando houver uma correspondência na tabela esquerda ou direita;



Fonte: https://www.w3schools.com/sql/

Exercícios

```
CREATE TABLE estados (
   codigo INT NOT NULL PRIMARY KEY AUTO_INCREMENT ,
   nome VARCHAR(100) NOT NULL
);
CREATE TABLE cidades (
   codigo INT NOT NULL PRIMARY KEY AUTO_INCREMENT ,
   nome VARCHAR(100) NOT NULL,
   codEstado INT DEFAULT 1 ,
   FOREIGN KEY (codEstado) REFERENCES estados(codigo)
);
CREATE TABLE pessoas (
   codigo INT NOT NULL PRIMARY KEY AUTO_INCREMENT ,
   nome VARCHAR(100) NOT NULL ,
   altura DOUBLE ,
   nascimento DATE DEFAULT '1970-12-25' ,
   codCidade INT ,
   FOREIGN KEY (codCidade) REFERENCES cidades(codigo)
);
CREATE TABLE pedidos (
   codigo INT NOT NULL PRIMARY KEY AUTO_INCREMENT ,
   endereco VARCHAR(100) NOT NULL ,
   horario DATETIME  ,
   codCliente INT ,
   FOREIGN KEY (codCliente) REFERENCES pessoas(codigo)
);
```

Retornar o horário do pedido, o endereço, nome do cliente, nome da Cidade do cliente e o nome do estado

- # BIBLIOGRAFIA

- HEUSER, Carlos Alberto. Projeto de Bancos de Dados: Projeto de banco de dados: Volume 4 da Série Livros didáticos informática UFRGS. Bookman Editora, 2009.
- RAMAKRISHNAN, R.; GEHRKE, J. Sistema de Gerenciamento de Banco de dados. Terceira Edição. 2008. Mc Graw Hill.
- ORACLE. MySQL 5.7 Reference Manual. Disponível em: https://dev.mysql.com/doc/refman/5.7/en/
- SQL Tutorial. MySQL, SQL Server, MS Access, Oracle, Sybase, Informix, Postgres, and other database systems. Disponível em: https://www.w3schools.com/sql/
- Documentação do SQL Server. Disponível em: https://docs.microsoft.com/pt-br/sql/sql-server/sql-server-technical-documentation?view=sql-server-2017