INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

Programação com acesso a BD

- ***Aula 11: Linguagem SQL: Consultas em Tabelas***

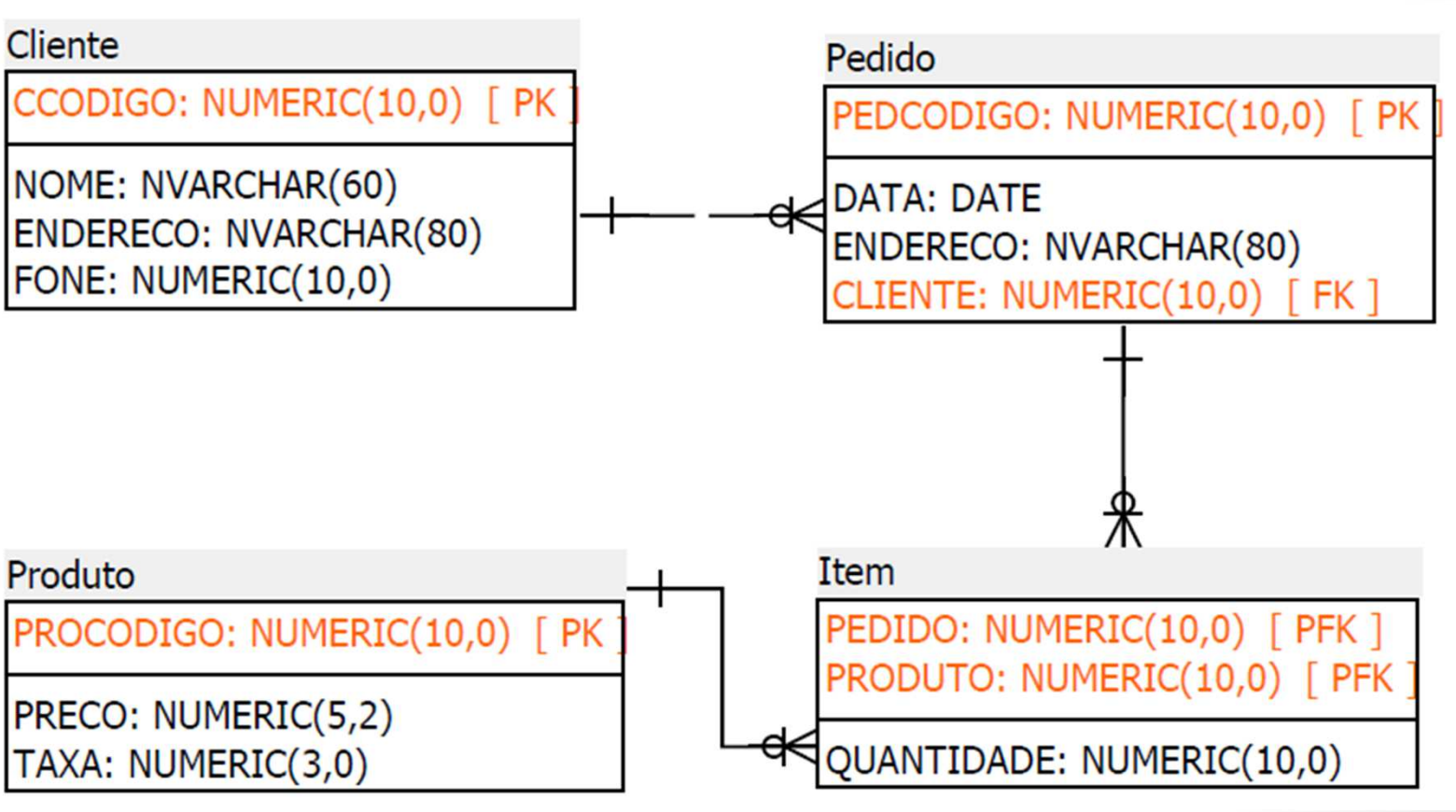
Prof.: Clayton Maciel Costa
clayton.maciel@ifrn.edu.br

Agenda

- Estrutura Básica de consultas SQL:
  - A cláusula SELECT;
  - A cláusula FROM;
  - A cláusula WHERE.
- Operação com cadeias de caracteres LIKE;
- Operação de renomeação AS;
- Operação ORDER BY;
- Funções Básicas de Agregação.

Banco de Dados Exemplo

- Considere o esquema de banco de dados:



Estrutura Básica de consultas SQL

- SQL é baseada em operações de conjuntos e de álgebra relacional com algumas modificações e extensões;
- Uma consulta SQL básica tem a forma:

```
Select A1, A2, ..., An
From r1, r2, ..., rm
Where P
```

  - A_i's representam atributos;
  - R_i's representam relações;
  - P é um predicado.

- O resultado de uma consulta SQL é sempre uma relação.

A cláusula SELECT

- A cláusula `select` corresponde à operação de projeção;
- É utilizada para listar os atributos pretendidos no resultado da consulta;
  - Exemplos:
    - Listar os nomes de todos os clientes

```
Select nome
From Cliente
```

    - Um asterisco na cláusula `select` denota "todos os atributos"

```
Select *
From Cliente
```

- As maiúsculas e minúsculas não são distinguidas na linguagem SQL.

A cláusula SELECT

- O SQL permite duplicações nos resultados de consultas;
- Para forçar a eliminação de duplicações, deve-se inserir a palavra-chave `distinct` após `select`.
  - Apresentar os endereços de todos clientes, sem repetições.

```
Select distinct endereco
From Cliente
```

- A palavra-chave `all` indica que os duplicados não devem ser removidos.

```
Select all endereco
From Cliente
```

A cláusula SELECT

- A cláusula `select` pode conter expressões aritméticas envolvendo as operações, +, –, *, e /, com argumentos constantes ou atributos;

- Por exemplo, a consulta abaixo devolve uma relação idêntica à relação Produto, exceto que o atributo taxa é multiplicado por 100.

```
Select procodigo, preço, taxa*100
From Produto
```

- Geralmente, os banco de dados definem uma biblioteca de funções que podem ser utilizadas com a cláusula SELECT;

A cláusula FROM

- A cláusula `from` corresponde à operação de produto cartesiano;
- Indica as relações a consultar na avaliação da expressão;
  - Por exemplo: Encontrar o produto cartesiano Cliente x Pedido.

```
Select *
From Cliente, Pedido
```

A cláusula FROM

- **IMPORTANTE**: Na grande maioria das consultas SQL, quando mais de uma tabela está na cláusula `from`, é necessário a criação de junções na cláusula WHERE.
- Por exemplo: Encontrar todos os clientes que tem pedido.

```
Select *
From Cliente, Pedido
Where Cliente.ccodigo = pedido.cliente
```

Junção

- **Uma junção corresponde a um relacionamento entre duas ou mais tabelas.**

A cláusula WHERE

- A cláusula `where` corresponde ao predicado de seleção;
- É formada por um predicado envolvendo atributos de relações que aparecem na cláusula `from`.
  - Encontrar telefone dos clientes que se chamam 'João da Silva'.

```
Select telefone
From Cliente
Where nome='João da Silva'
```

- Os resultados de comparações podem ser combinados por intermédio dos conectivos lógicos **and**, **or**, e **not**;
- Por exemplo: Encontrar os códigos dos produtos do pedido número 203, cujas quantidades são superiores a dois itens.

```
Select produto
From Itens
Where pedido=203 and quantidade>2
```

A cláusula WHERE

- A linguagem SQL possui um operador de comparação `between` para especificar condições em que um valor deve estar contido num intervalo de valores (incluindo os seus extremos).
  - Por exemplo: apresentar os códigos dos produtos cujos preços estão entre R$5,00 e R$30,00

```
Select procodigo
From Produto
Where preço between 5 and 30
```

- Para negar a condição pode-se colocar o conectivo `not` antes de `between`.
  - Apresentar os códigos dos produtos cujas taxas NÃO estão entre 20% e 35%

```
Select procodigo
From Produto
Where taxa not between 0,2 and 0,35
```

Operações com cadeia de caracteres

- O SQL inclui um mecanismo de concordância de padrões para comparações envolvendo cadeias de caracteres;
- Os padrões são descritos recorrendo a dois caracteres especiais:
  - percentagem(%): concorda com qualquer subcadeia.
  - sublinhado (_): concorda com qualquer carácter.
- Por exemplo: listar todos os clientes que começam com a letra “A”

.

```
Select *
From Cliente
Where nome like ‘A%’
```

- Por exemplo: listar todos clientes que terminem com nome “Silva”

```
Select *
From Cliente
Where nome like ‘%Silva’
```

Operações com cadeia de caracteres

- Por exemplo, listar todos os clientes cujo endereço inclua “Rua Nélio Rodrigues” .

```
Select nome
From Cliente
Where endereço like ‘% Rua Nélio Rodrigues %’
```

- Encontrar uma cadeia de caracteres que de fato contenha o símbolo de porcentagem
  - Ex.: 10%

```
like ‘10\%’
```

Operações com cadeia de caracteres

- Outros exemplos:
- **LIKE 'A%'** - Todas as palavras que iniciem com a letra A;
- **LIKE '%A'** - Todas que terminem com a letra A;
- **LIKE '%A%'** - Todas que tenham a letra A em qualquer posição;
- **LIKE 'A_'** - String de dois caracteres que tenham a primeira letra A e o segundo caractere seja qualquer outro;
- **LIKE '_A'** - String de dois caracteres cujo primeiro caractere seja qualquer um e a última letra seja A;
- **LIKE '_A_'** - String de três caracteres cuja segunda letra seja A, independentemente do primeiro ou do último caractere;
- **LIKE '%A_'** - Todos que tenham a letra A na penúltima posição e a última seja qualquer outro caractere;
- **LIKE '_A%'** - Todos que tenham a letra A na segunda posição e o primeiro caractere seja qualquer um.

Operação de renomeação AS

- A linguagem SQL permite a renomeação de relações e atributos recorrendo à cláusula `as` :

  old_name `as` *new_name*

- Listar os nome e códigos dos pedidos de cada cliente, renomeando a coluna pedcodigo para codPedido

```
Select c.nome, p.pedcodigo as codPedido
From Cliente as c, Pedido as p
Where c.ccodigo = p.pedcodigo
```

- Caso se pretenda utilizar um nome com espaços, esse nome deverá ser colocado entre **aspas**.

```
Select c.nome, p.pedcodigo as “codigo do Pedido”
From Cliente as c, Pedido as p
Where c.ccodigo = p.pedcodigo
```

Operação de ordenação ORDER BY

- Listar em ordem alfabética os nomes de todos os clientes que possuem algum pedido:

```
Select distinct nome
From Cliente as c, Pedido as p
Where c.ccodigo = p.cliente
Order by nome;
```

- Pode-se especificar `desc` para ordenação descrescente ou `asc` para ordenação ascendente, para cada atributo; por padrão, assume-se ordem ascendente.

```
Order by nome desc;
```

- Pode-se ter mais do que uma chave de ordenação, separando-as com vírgulas.

Funções de Agregação

- Estas funções aplicam-se a multiconjuntos de valores de uma coluna de uma relação, devolvendo um único valor como resultado.
- Funções Básicas:
  - **avg:** calcula valor médio;
  - **min:** calcula valor mínimo;
  - **max:** calcula valor máximo;
  - **sum:** calcula a soma dos valores;
  - **count:** calcula o número de valores.

Funções de Agregação

- Determinar o preço médio dos produtos comprados no dia 16/03/2008.

```
Select AVG(preço)
From Pedido as p, Itens as i,
Produto as pr
Where p.data='16/03/2008' and
      p.pedcodigo=i.produto and
      i.produto=pr.procodigo
```

- Calcular a quantidade de clientes.

```
Select COUNT(ccodigo)
From Cliente
```

- Encontrar o número de clientes com nomes diferentes.

```
Select COUNT(distinct nome)
From Cliente
```

Exercício de Fixação

- Faça as seguintes consultas SQL:
  - Selecione os pedidos dos clientes da cidade de ‘baltimore’;
  - Selecione os clientes que fizeram pedidos em ’10/10/2008’
  - Selecione os produtos que começam com a letra ‘C’.
  - Selecione os clientes que seus telefones apresentam a cadeia ‘11’ em qualquer parte.
  - Selecione o cliente que mais fez pedidos.

FIM