cin.ufpe.br

Centro de Informática

U · F · P · E

Universidade Federal de Pernambuco

Fundamentos de SQL

"Structured Query Language"

Aula1

Apresentado por:

Robson do Nascimento Fidalgo

rdnf@cin.ufpe.br

SQL - Introdução

- SQL- Structured Query Language
  - Apesar do QUERY, não é apenas de consulta (inclusão, alteração,...)
- É fundamentada na álgebra relacional, inclui comandos para:
  - Definição, Consulta e Atualização de Dados e da Bases de Dados
- Histórico:
  - Definição da1ª versão em 1974 – IBM – chamada SEQUEL
  - 1975 implementado o 1º protótipo
  - Revisada e ampliada entre 1976/77.
    - Teve seu nome alterado para SQL por razões Jurídicas
  - Publicada como padrão para SGBDR em 1986 pela ANSI
  - Versões posteriores a SQL1 (1986) → SQL2 (1992) e SQL3 (1999)

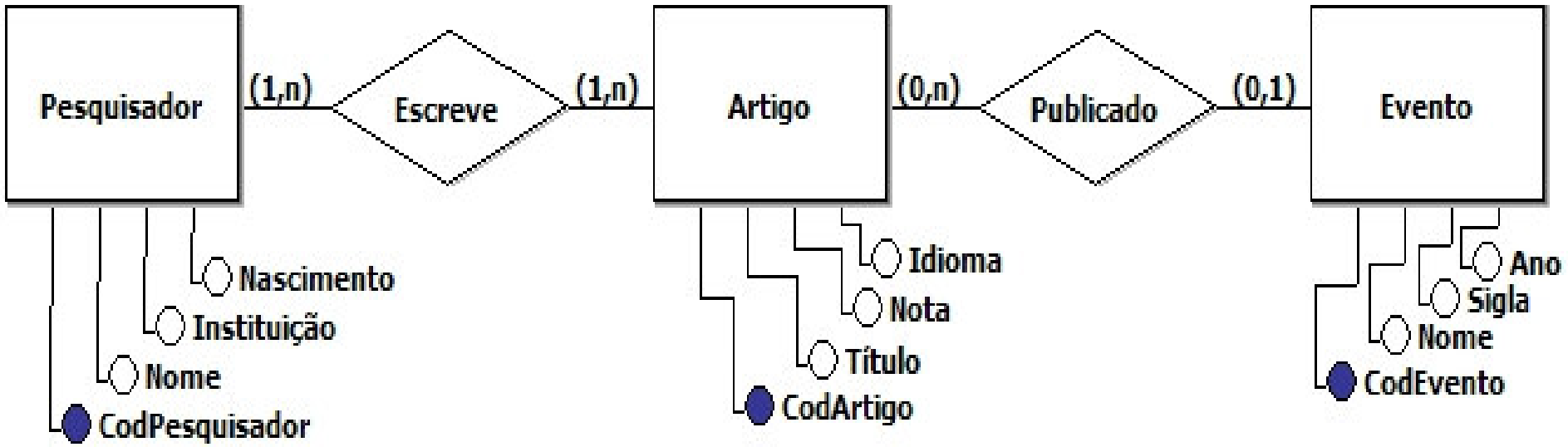
SQL - Propriedades

- A linguagem é composta de duas partes principais:
  - DML- linguagem de manipulação de dados
    - Compreende comandos para inserir, consultar, remover e modificar dados.
  - DDL - Linguagem de definição de dados
    - Fornece comandos para criação, remoção e atualização de tabelas, índices e visões do banco de dados.

SQL - Atenção !

- Cada implementação de SQL possui algumas adaptações para resolver certas particularidades, portanto, qualquer comando pode ser usado de forma diferente em um determinado SGBDR.
  - Recomenda-se a leitura do manual do fabricante para maiores informações sobre o uso da linguagem SQL em SGBDR comerciais.
- O SQL usado neste curso será o baseado no ORACLE 10G, mas nenhuma característica específica deste SGBD será abordada. Assim, espera-se que os exemplos, com poucas modificações, executem em qualquer SGBDR.

Exemplo Modelos ER & Relacional



Pesquisador (<u>CodPesquisador</u>, Nome, Instituição, Nascimento)

Escreve (<u>CodPesquisador,CodArtigo</u>)

CodPesquisador referencia Pesquisador, CodArtigo referencia Artigo

Artigo(<u>CodArtigo</u>, Título, Nota, Idioma, CodEvento)

CodEvento referencia Evento

Evento (<u>CodEvento</u>, Nome, Sigla, Ano)

SQL - Tipos de Dados Genéricos

- Cada SGBD tem um conjunto próprio de tipos de dados.
- Os Principais tipos de dados do Oracle são:
  - Char(n) : cadeia de tamanho fixo. Padrão é 1 e o máximo é 2000
  - Varchar2(n): cadeia de tamanho variável com o max de n (max = 4000)
  - Number(p,e): numérico
    - **p** é precisão (quantidade de dígitos → max 38)
    - **e** é escala (número de casas decimais → menor ou igual a **p**)
  - Date: data e hora, inclui século, ano, mês, dia, hora, minuto e segundo.

DDL - Criando Tabelas

- **CREATE TABLE** - Cria uma nova tabela com seus campos e define as restrições de campo.

CREATE TABLE ***Tabela*** **(**

Coluna1 Tipo **[(*Tamanho*)] [NOT NULL] [DEFAULT] [...],**

[,*Coluna2 Tipo* [(T*amanho*)] [NOT NULL] [DEFAULT] [...],

[UNIQUE (*Candidata1*[, *Candidata2*[, ...]])]

[CHECK (*condição*)]

[PRIMARY KEY (*Primária1*[, *Primária2* [, ...]])]

[FOREIGN KEY (*Estrangeira1*[, *Estrangeira2* [, ...]]) REFERENCES *TabelaExterna* [(*ColunaExterna1* [, *ColunaExterna2* [, ...]])]

[CONSTRAINT *Nome* PRIMARY KEY (*Primária1*[, *Primária2* [, ...]])]

[CONSTRAINT *Nome* FOREIGN KEY (*Estrangeira1*[, *Estrangeira2* [, ...]]) REFERENCES *TabelaExterna* [(*ColunaExterna1* [, *ColunaExterna2* [, ...]])]);

DDL - Criando Tabelas

- Exemplo:

```
-- Cria tabela Pesquisador
Create Table Pesquisador(
CodPesquisador Varchar2(4),
Nome Varchar2(80) NOT NULL,
Instituicao Varchar(40) NOT NULL,
Nascimento Date,
Primary Key (CodPesquisador),
Unique (Nome, Nascimento));
```

DDL - Criando Tabelas

- Exemplo:

```
-- Cria tabela Evento
Create Table Evento(
CodEvento Varchar2(4),
Nome Varchar2(80) NOT NULL,
Sigla Varchar2(10) NOT NULL,
Ano Number(4),
Primary Key (CodEvento));
```

DDL - Criando Tabelas

- Exemplo:

```
-- Cria tabela Artigo
Create Table Artigo(
CodArtigo Varchar2(4),
Titulo Varchar2(80) NOT NULL,
Nota Number(4,2) NOT NULL,
Idioma Varchar2(15),
CodEvento Varchar2(4),
Primary Key (CodArtigo),
Foreign Key (CodEvento) References Evento,
Check (Nota <= 10));
```

DDL - Criando Tabelas

- Exemplo:

```
-- Cria tabela Escreve
Create Table Escreve(
CodPesquisador Varchar2(4),
CodArtigo Varchar2(4),
Constraint Escreve_pk primary key
(CodPesquisador, CodArtigo),
Constraint EscrevePesquisador_Fk Foreign Key
(CodPesquisador) References Pesquisador,
Constraint EscreveArtigo_Fk Foreign Key
(CodArtigo) References Artigo);
```

DDL - Alterando Tabelas

- **ALTER TABLE** - permite inserir/eliminar/modificar colunas nas tabelas já existentes

ALTER TABLE *Tabela*
{ADD { coluna | restrição }|
DROP { coluna | restrição } |
MODIFY coluna}

DDL - Alterando Tabelas

- **Exemplo:**

-- Adicionar o campo E-MAIL na tabela Pesquisador

```
Alter Table Pesquisador
Add  Email Varchar2(40);
```

-- Modificar o campo E-MAIL na tabela Pesquisador

```
Alter Table Pesquisador
Modify Email Char(50);
```

-- ELIMINAR o campo E-MAIL na tabela Pesquisador

```
Alter Table Pesquisador
Drop Column Email;
```

DDL - Criando Índices

- **<u>CREATE INDEX</u>** - Cria um novo índice em uma tabela existente.

  CREATE [UNIQUE] INDEX *Indice* ON
  ***Tabela* (*Coluna1* [, *Coluna2*...])**

  - Os índices são estruturas que permitem agilizar a busca e ordenação de dados em tabelas

- Exemplo

```
-- Criar índice para o campo Nome da tabela Evento
CREATE UNIQUE INDEX NomeEventoIDX
ON  Evento (Nome);
```

```
-- Criar índice para o campo CodEvento da tabela Artigo
CREATE INDEX EventoArtigoIDX
ON  Artigo (CodEvento);
```

DDL - Excluindo Tabelas e Índices

- **DROP** - Exclui uma tabela ou um índice já existente
  - CUIDADO, pois os dados também são excluídos

DROP {TABLE *tabela* | INDEX *índice*}

Exemplo:

--Excluir a tabela Artigo
DROP TABLE Artigo;

-- Excluir o índice* NomeEventoIDX *da tabela Evento
DROP INDEX NomeEventoIDX**;**

DML - Inserindo Dados em Tabelas

- **INSERT INTO** - Adiciona uma linha ou várias linhas na tabela.

  INSERT INTO *Tabela* [(*Coluna1*[, *Coluna2* [, ...]])]
  {VALUES (*Valor1*[, *Valor2*[, ...]) | SELECT *Cláusula* }

  - OBS:
    - Os dados são inseridos pela ordem especificada.
    - valores para campos CHAR, VARCHAR2 ou DATE são inseridos entre aspas.
    - Se omitir a lista de colunas, serão selecionadas todas as colunas da tabela, pela sua ordem de criação

DML - Inserindo Dados em Tabelas

- **Exemplos**

--Insere um registro usando apenas alguns campos da tabela Artigo

```
Insert Into Artigo(CodArtigo, Titulo, Nota)
Values ('1111', 'Normalização Morreu? ', 10.00);
```

--Insere um registro completo na tabela Artigo

```
Insert Into Artigo
Values ('2222', 'Desafios em Banco de Dados', 10.00, 'Português', '1010');
```

DML - Inserindo Dados em Tabelas

- **Exemplos**

--Criar a tabela PesquisadorVeterano e em seguida inserir nesta tabela os Pesquisadores com nascimento < 01/01/1960

```
Create Table PesquisadorVeterano(
CodPesquisador Varchar2(4),
Nome Varchar2(80) NOT NULL,
Instituicao Varchar(40) NOT NULL,
Nascimento Date,
Primary Key (CodPesquisador),
Unique (Nome, Nascimento));

Insert Into PesquisadorVeterano
SELECT CodPesquisador, Nome, Instituicao, Nascimento
FROM Pesquisador
WHERE Nascimento < '01/01/1960';
```

Mais a frente, o comando SELECT será visto com detalhes

DML - Atualizando Dados em Tabelas

- **UPDATE** - Com base nos critérios especificados, altera valores de campos de uma tabela.

UPDATE *Tabela*
SET *Coluna = Valor*
[WHERE *Condição*];

- A cláusula WHERE especifica quais dados da coluna serão alterados. Sem esta, todos os dados serão atualizados com o valor especificado.
- Mais a frente, WHERE será visto com detalhes.

Exemplo:

--Alterar o Nome e Ano da tabela Evento com CodEvento = '1111'

```
Update Evento
SET Nome = 'Novo Nome',
Ano = 2000
WHERE CodEvento = '1111';
```

-- Tirar 1 ponto de todos os artigos

```
Update Artigo
SET Nota = Nota - 1;
```

DML - Excluindo Dados em Tabelas

- **DELETE** - Exclui as linhas de uma ou mais tabelas que satisfaçam a condição.

```
DELETE
FROM Tabela
[WHERE Condição]
```

A cláusula WHERE especifica quais linhas da tabela serão excluídas. Sem esta, todas as linhas da tabela serão excluídas.

Exemplo:

```
--Excluir os registros da tabela Artigos, onde CodArtigo = '1111'
DELETE
FROM Artigo
WHERE CodArtigo = '1111';

--Excluir todos os registros da tabela Artigo
DELETE
FROM Artigo;
```

Não é a mesma coisa que DROP TABLE.

DML - Consultando Dados em Tabelas

- Estrutura Básica

SELECT ➡ PROJEÇÃO

FROM ➡ TABELA OU PRODUTO CARTESIANO DELAS

WHERE ➡ SELEÇÃO

$$\Pi_{Coluna1[,Coluna2\,[,...]\,]}\,(\sigma_{Condição}(Tabela1\,[X\,Tabela2\,[X\,...\,]\,]))$$

SELECT *Coluna1*[,*Coluna2* [, ...]]
FROM *Tabela1*,[*Tabela2* [, ...]]
WHERE *Condição*

DML - Consultando Dados em Tabelas

- Estrutura Genérica

SELECT [DISTINCT | ALL] { * | [*Tabela*.]*Coluna1* [AS *Alias1*]

[[*Tabela*.]*Coluna2* [AS *Alias2*] [, ...]]}

FROM *Tabela1* [, *Tabela2* [, ...]]

[WHERE {*Condição Simples* | *Condição de Sub-consulta*}]

[ORDER BY *Coluna1* [ASC | DESC] [,*Coluna2* [ASC | DESC] [, ...]]]

[GROUP BY *Coluna1* [,*Coluna2* [, ...]] [HAVING *Condição*]]

[{UNION | IMTERSECT | EXCEPT} SELECT ...]

DML - Consultando Dados em Tabelas

- **SELECT/FROM**
  - Projeta os dados da(s) tabela(s), de acordo com os critérios especificados.
  - A projeção do resultado é em uma estrutura tipo tabela
- Basta informar o que se quer, sem se preocupar como fazer isto (SQL Não é procedural).

DML - Consultando Dados em Tabelas

- **SELECT/FROM**

- Na cláusula SELECT, pode-se utilizar operadores aritméticos e funções de agregações, para projetar cálculos.

Oper. Aritméticos	
+	Adição
-	subtração
*	Multiplicação
/	Divisão

Funções de Agregação	
AVG	Média
MIN	Mínimo
MAX	Máximo
COUNT	Contar
SUM	Somar

DML - Consultando Dados em Tabelas

- **Exemplos:**

--Projetar todas as informações dos Pesquisadores

```
Select CodPesquisador, Nome, Instituicao, Nascimento
From Pesquisador ;
        OU
Select *
From Pesquisador ;
```

O * projeta todas as colunas de todas as tabelas especificadas na cláusula **FROM**

--Projetar a média das notas dos Artigos

```
Select Avg (Nota)
From Artigo;
```

--Projetar todos os Artigos(títulos) e suas notas com -1 ponto

```
Select Titulo, Nota – 1
From Artigo;
```

--Projetar a quantidade de Pesquisadores cadastrados

```
Select Count (*) as Quantidade, 'Ud' as Unidade
From Pesquisador;
```

DML - Consultando Dados em Tabelas

- Em SQL a eliminação de linhas duplicadas não é feita automaticamente, devendo a mesma ser especificada explicitamente.
  - ALL é o padrão quando não especificado DISTINCT.

- **Exemplos:**

--Projetar todos os Títulos Dos Artigos repetidamente

```
Select All Titulo
From Artigo;
```

OU

```
Select Titulo
From Artigo;
```

--Projetar todos os Títulos dos Artigos sem duplicatas

```
Select Distinct Titulo
From Artigo;
```

DML - Consultando Dados em Tabelas

- Uma coluna pode ser especificada pelo nome da sua tabela (*Tabela.Coluna*), bem como, ser renomeada durante a consulta (*Coluna AS ColunaRenomeada*)

```
--Projetar todos os Nomes e Instituições da tabela Pesquisador
NOTE:  mesmo especificando Tabela.Coluna, FROM é obrigatório
Select Pesquisador.Nome, Pesquisador.Instituicao
From Pesquisador;

--Projetar todos os títulos dos Artigos e suas Notas com –1 ponto
Select Titulo, Nota -1 as Reducao
From Artigo;
```

DML - Consultando Dados em Tabelas

- **WHERE -** Especifica quais linhas da(s) tabela(s) listada(s) na cláusula FROM são afetadas pela condição.
  - Se esta não for especificada, a consulta retornará todas as linhas da tabela.
- Operadores Utilizados

Lógicos	
AND	E
OR	Ou
NOT	Não

Relacionais			
<> ou !=	Diferente	=	Igual a
>	Maior que	>=	Maior ou igual a
<	Menor que	<=	Menor ou igual a

- A condição de WHERE pode ser de três tipos:
  - Comparação
  - Ligação entre tabelas (Join)
  - Sub-Consulta (Sub-Queries)

DML - Consultando Dados em Tabelas

- **Comparação**
  - **Operador Relacional**

```
--Projetar Artigos com Nota acima de 8
Select *
From Artigo
Where Nota > 8;
```

  - **Operador [Not] Between**
    - A condição é verdadeira quando o resultado está compreendido entre o Limite Inicial e o Limite Final (inclusive).

```
--Projetar Artigos com nota de 7 a 9
Select *
From Artigo
Where Nota Between 7 And 9;
```

DML - Consultando Dados em Tabelas

- **Comparação**
  - **Operador [Not] In**
    - A condição é verdadeira se o resultado é igual a um dos valores entre parênteses

```
--Projetar os Eventos que ocorreram em 2000, 2002 ou 2004
Select *
From  Evento
Where Ano In (2000, 2002, 2004);
```

DML - Consultando Dados em Tabelas

- **Comparação**
  - **Operador [Not] Like**
    - A condição é satisfeita quando o resultado é igual ao valor da cadeia de caracteres.
    - Caracteres especiais para construção da cadeia de caracteres:

      "%" → Usado para representar zero ou mais caracteres.
      "?" → Usado para representar um caractere.

```
--Projetar todos os Pesquisadores cujo nome
tenha 10 caracteres e inicie com R
Select *
From Pesquisador
Where Nome Like 'R?????????';

--Projetar todos os Artigos que tenham
Banco de Dados no seu título
Select *
From Artigo
Where Titulo Like '%Banco de Dados%';
```

DML - Consultando Dados em Tabelas

- **Comparação**
  - **Operador Is [Not] Null**
    - A condição é satisfeita quando o valor da coluna for NULL

```
--Projetar todos os Artigos que estão sem nota definida
Select *
From Artigo
Where Nota Is Null;
```

- **Pode-se misturas os vários tipos de comparação**

```
--Projetar todos os Artigos que iniciam com R, estão com nota definido e
   têm idioma = 'Português'
Select *
From Artigo
Where Titulo Like 'R%' And
Nota Is Not Null And
Idioma = 'Português';
```

cin.ufpe.br

Centro de Informática

U · F · P · E

Universidade Federal de Pernambuco