cin.ufpe.br

Centro de Informática

U · F · P · E

Universidade Federal de Pernambuco

Fundamentos de SQL

"Structured Query Language"

Aula2

Apresentado por:

Robson do Nascimento Fidalgo

rdnf@cin.ufpe.br

DML - Consultando Dados em Tabelas

- **Ligação entre tabelas (Join)**
  - Diz-se que tabelas estão relacionadas se tiverem campos comuns (ch. primária e ch. estrangeira)
  - O efeito do JOIN é a criação de uma tabela temporária em que cada par de linhas, que satisfação a condição de ligação, são ligados para formar uma única linha.
  - A ligação é sempre estabelecida à frente da cláusula WHERE usando o operador relacional da igualdade (=).
    - FROM estabelece o produto cartesiano entre as tabelas listadas
    - WHERE filtra as linhas úteis segundo a condição especificada

DML - Consultando Dados em Tabelas

- **Ligação entre tabelas (Join) - cont**
  - Pode-se misturar as cláusulas de comparação, vistas anteriormente, juntamente com operadores lógicos (And, Or e Not) para formar condições de comparação mais complexas
    - Para se ligar várias tabelas, usa-se o operador lógico AND.
  - É preciso ter muito cuidado com os JOINS, pois exigem alto custo de execução (implicam diretamente na performance).

DML - Consultando Dados em Tabelas

- **Ligação entre tabelas (Join) - exemplos**

--Projetar todos os Artigos publicados (Título) e seus Eventos (Sigla) nomeando as tabelas

```
Select A.Titulo, E.Sigla
From Artigo A, Evento E
Where A.CodEvento = E.CodEvento;
```

--Projetar todos os Pesquisadores (Nome) e seus Artigos (Título) nos idiomas Português ou Inglês

```
Select P.Nome, A.Titulo
From Pesquisador P, Escreve E, Artigo A
Where P.CodPesquisador = E.CodPesquisador And
A.CodArtigo = E.CodArtigo  And
A.Idioma In ('Português','Inglês');
```

DML - Consultando Dados em Tabelas

- **Ligação entre tabelas (Join) – Junções Externas**
  - Junção Interna = Inner Join
    - Esta junção retorna todos os pares com correspondentes de linhas nas duas tabelas
    - É a mesma coisa que Junção natural
    - EX: Os 2 vistos até agora
  - Junção Externa = Outer Join
    - Esta junção retorna todos as linhas obtidas pela operação INNER JOIN, mais as linhas da tabela da esquerda (**Left Outer Join**), da direita (**Right Outer Join**) ou de ambas (**Full Outer Join**) que não atendam à condição de junção natural
    - EX: Retornar todos os Títulos dos Artigos (publicados ou não) e a Sigla dos Eventos (somente dos artigos aceitos para publicação)

DML - Consultando Dados em Tabelas

- **Ligação entre tabelas (Join) - Sintaxes**

Tradicional

```
SELECT ...
FROM Tabela_x TX, Tabela_y TY
WHERE TX.PK = TY.FK [AND outras condições]
```

Inner/Outer Join

```
SELECT ...
FROM Tabela_x TX  {INNER JOIN | LEFT OUTER JOIN |
   RIGHT OUTER JOIN | FULL OUTER JOIN} Tabela_y TY
   ON X.PK = Y.FK
[WHERE outras condições]
```

DML - Consultando Dados em Tabelas

- **Ligação entre tabelas (Join) – Junções Externas**
  - Junção Interna = Inner Join
    - EX:

--Projetar todos os Artigos publicados (Título) e seus Eventos (Sigla)

Select A.Titulo, E.Sigla

From Artigo A Inner Join Evento E

On A.CodEvento = E.CodEvento;

--Projetar todos os Pesquisadores (Nome) e seus Artigos (Título) nos idiomas Português ou Inglês

Select P.Nome, A.Titulo

From **(Pesquisador P Inner Join Escreve E on P.CodPesquisador = E.CodPesquisador)** Inner Join Artigo A on A.CodArtigo = E.CodArtigo

Where A.Idioma In ('Português','Inglês');

DML - Consultando Dados em Tabelas

- **Ligação entre tabelas (Join) – Junções Externas**
  - Junção Externa = Outer Join
  - EX Left Outer Join

-- Retornar todos os Títulos dos Artigos (publicados ou não) e a Sigla dos Eventos (somente dos artigos aceitos para publicação)

```
Select A.Titulo, E.Sigla
From Artigo A Left Outer Join Evento E
On A.CodEvento = E.CodEvento;
```

-- Retornar todas as Siglas dos Eventos (com ou sem artigos publicados) e o Título dos Artigos que foram aceitos para publicação nos respectivos eventos

```
Select A.Titulo, E.Sigla
From Artigo A Right Outer Join Evento E
On A.CodEvento = E.CodEvento;
```

-- Retornar todos os Títulos dos Artigos (publicados ou não) e a sigla de todos os Eventos (com ou sem artigos publicados)

```
Select A.Titulo, E.Sigla
From Artigo A Full Outer Join Evento E
On A.CodEvento = E.CodEvento;
```

DML - Consultando Dados em Tabelas

- Sub-Consulta (Sub-Queries)
  - As sub-consultas podem retornar um valor simples, ou um conjunto de valores
- Sub-consultas que retornam um valor simples
  - Usadas para fazer comparação elemento-elemento
  - WHERE expressão {= | <> | > | >= | < | <=} (Sub-Consulta)

--Projetar os Artigos (título) com nota acima da média

```
Select Titulo
From Artigo
Where Nota >
      (Select Avg (Nota)
      From Artigo) ;
```

A sub-consulta deve retornar uma única tupla. Para isso pode-se usar as funções de agregação (AVG,MIN,MAX,...)

DML - Consultando Dados em Tabelas

- Sub-consultas que retornam um conjunto de valores
  - Usadas para fazer comparação elemento-conjunto
  - Podem ser definidas através das cláusulas IN, ANY e ALL
  - WHERE expressão [NOT] IN (Sub-Consulta)
    - Estabelece uma relação de pertinência (∈) entre elementos e conjuntos (tabelas). Sua avaliação retorna um valor booleano.

```
--Projetar Pesquisadores (Nome) que possuem Artigos sem Nota
Select Nome
From Pesquisador
Where CodPesquisador NOT IN
(Select CodPesquisador
From Escreve E Inner Join Artigo A on E.CodArtigo = A.CodArtigo
Where Nota Is not Null);
```

DML - Consultando Dados em Tabelas

- Sub-consultas que retornam um conjunto de valores
  - WHERE expressão { = | <> | > | >= | < | <= } ANY (Sub-consulta)
    - Verifica se a condição é verdadeira para pelo menos um dos valores retornados pela sub-consulta
    - Permite outras formas de comparação elemento-conjunto.
      - = ANY → tem mesmo efeito que IN

--Projetar Pesquisadores (nome) que possuem Artigo sem Nota

```
Select Nome
From Pesquisador
Where CodPesquisador = ANY
      (Select CodPesquisador
      From Escreve E Inner Join Artigo A
      on E.CodArtigo = A.CodArtigo
      Where Nota Is Null);
```

--Projetar pesquisadores (nome), exceto o do mais idoso

```
Select Nome
From Pesquisador
Where Nascimento > ANY
      (Select Nascimento
       From Pesquisador)
```

DML - Consultando Dados em Tabelas

- Sub-consultas que retornam um conjunto de valores
  - WHERE expressão { = | <> | > | >= | < | <= } ALL (Sub-consulta)
    - Verifica se a condição é verdadeira para todos os valores retornados pela sub-consulta.
    - É o oposto de ANY.
      - <> ALL → tem mesmo efeito que NOT IN

--Projetar os Artigos(título) com Nota maior que todos os artigos do Evento 'International XPTO Conference'

```
SELECT Titulo
FROM Artigo
WHERE Nota > ALL
      (SELECT Nota
      FROM Artigo A, Evento E
      WHERE A.CodEvento = E.CodEvento
      AND E.Nome = 'International XPTO Conference');
```

DML - Consultando Dados em Tabelas

- **ORDER BY -** Ordenar os resultados pelos valores de uma ou mais colunas.
  - As linhas são ordenadas pela primeira coluna após ORDER BY. Quando as linhas de uma coluna possuem valores iguais, estas serão classificadas pelo valor da segunda coluna após ORDER BY e assim por diante**.**
  - **Tipos de Ordenação:**
    - ASC→ Ascendente
    - DESC → Decrescente

- **Exemplo:**

  --Projetar o nome e o nascimento dos pesquisadores em ordem decrescente do nascimento. Para datas iguais, considerar a ordem alfabética do nome do Pesquisador

  SELECT Nome, Nascimento

  FROM Pesquisador

  ORDER BY Nascimento DESC, Nome ASC;

DML - Consultando Dados em Tabelas

- **GROUP BY -** agrupa os resultados por valores idênticos.
  - Utiliza-da com as funções de agregação
  - OBS: Os campos do GROUP BY devem aparecer no SELECT!
- **Exemplo:**

```
--Projetar a média das Notas dos Artigos por Evento
Select E.Sigla, Avg (A.Nota) As MediaNota
From Artigo A, Evento E
Where A.CodEvento = E. CodEvento
Group By E.Sigla ;
```

DML - Consultando Dados em Tabelas

- **HAVING -** Utilizada para filtrar o resultado dos grupos
  - Só é atendida depois do Agrupamento !
  - Só existe se associada à cláusula **GROUP BY** (mas o oposto não)
  - Vem depois do GROUP BY e antes do ORDER BY
  - A condição só pode envolver os campos/funções do SELECT
- **Exemplos:**

```
--Projetar os Eventos (Sigla) cuja as média das Notas dos Artigos é > 8
Select E.Sigla, Avg (A.Nota)
From Artigo A, Evento E
Where A.CodEvento = E. CodEvento
Group By E.Sigla
Having Avg (A.Nota)> 8;
```

```
---Projetar os Pesquisadores (nome) que publicaram mais de 3 Artigos
Select P.Nome, Count(Distinct A.Titulo)
From Pesquisador P, Artigo A, Escreve E
Where P.CodPesquisador = E.CodPesquisador And E.CodArtigo = A.CodArtigo
Group By AUTOR.Nome
Having Count(DISTINCT LIVRO.Titulo) > 3;
```

DML - Consultando Dados em Tabelas

- **Operações sobre Conjuntos**
  - Aplicáveis apenas em tabelas compatíveis
  - **UNION (∪) -** Faz a união, eliminando linhas repetidas.
    - Acrescenta-se ALL para manter as linhas repetidas
  - **INTERSECT (∩) -** Retorna apenas as linhas que pertencem às duas tabelas
  - **EXCEPT** (–) **-** Retorna apenas as linhas que pertencem à primeira tabela, com exceção das que aparecem na segunda.

DML - Consultando Dados em Tabelas

- Considere:
  - MEDICO (CodMedico, Nome, CRM) e
  - PACIENTE (CodPaciente, Nome);
  - DEPOSITANTE (CPF, Nome, Agencia, Conta) e
  - DEVEDOR (CPF, Nome, Agencia, Conta).

- **UNION**

```
--Projetar o nome de todas
as pessoas cadastradas
no hospital
(Select Nome
From Medico)
Union
(Select Nome
From Paciente);
```

```
-- Projetar todos os clientes da agência
A1 com empréstimo ou depósito
(Select *
From Depositante
Where Agencia = 'A1')
Union All
(Select *
From Devedor
Where Agencia = 'A1');
```

DML - Consultando Dados em Tabelas

- Considere:
  - MEDICO (CodMedico, Nome, CRM) e
  - PACIENTE (CodPaciente, Nome);
  - DEPOSITANTE (CPF, Nome, Agencia, Conta) e
  - DEVEDOR (CPF, Nome, Agencia, Conta).

- **INTERSECT**

--Projetar o nome de todas as pessoas que são médicos e pacientes ao mesmo tempo

```
(Select Nome
From Medico)
Intersect
(Select Nome
From Paciente);
```

--Projetar todos os clientes da agência A1 com empréstimo e depósito

```
(Select *
From Depositante
Where Agencia = 'A1')
Intersect
(Select *
From Devedor
Where Agencia = 'A1');
```

DML - Consultando Dados em Tabelas

- Considere:
  - MEDICO (CodMedico, Nome, CRM) e
  - PACIENTE (CodPaciente, Nome);
  - DEPOSITANTE (CPF, Nome, Agencia, Conta) e
  - DEVEDOR (CPF, Nome, Agencia, Conta).

- **EXCEPT**

```
--Projetar o nome de todas
as pessoas que são médicos
e não são pacientes
(Select Nome
From Medico)
Except
(Select Nome
From Paciente);
```

```
/*Projetar todos os clientes da agência
A1 com conta e sem empréstimo */
(Select *
From Depositante
Where Agencia = 'a1')
Except
(Select *
From Devedor
Where Agencia = 'a1');
```

DDL - Manipulando Visões

- VISÕES: São tabelas virtuais que não ocupam espaço físico
- Após definida uma visão, qualquer operação de consulta pode ser aplicada sobre ela.
  - Consultas sobre visões são escritas da mesma forma como para uma tabela qualquer
- Operações realizadas sobre uma visão refletem as tabelas físicas das quais elas derivam.
- São ótimas para substituir consultas freqüentemente usadas

DDL - Manipulando Visões

- Permite personalizar tabelas de acordo com o perfil do usuário
- Pode-se fazer uma visão de uma outra visão
- A Remoção de uma visão é em cascata
- SINTAXE:
  - **Criar Visão**
    - **CREATE VIEW *Visão* [*Colunas*] AS *ExpressãoConsulta;***
  - **Remover Visão**
    - **DROP VIEW *Visão;***

DDL - Manipulando Visões

- Exemplos:

--Criar visão dos clientes que tem conta ou empréstimo no banco

```
Create View TodosClientes as
    (Select *
    From Depositante)
Union
    (Select *
    From Devedor);
```

--**Remover a visão TodosClientes**

```
Drop View TodosClientes;
```

--Criar um visão dos Artigos do Evento '1111' com –1 ponto de punição

```
Create View Punicao (codArtigo, titulo, NovaNota) as
(Select codArtigo, titulo, Nota - 1
From Artigo2 A, Evento E
Where A.CodEvento = E.CodEvento and
E.CodEvento= '1111');
```

--**Remover a visão Punicao**

```
Drop View Punicao;
```

DDL - Autorizações de Acesso

- SGBDR podem ser acessados por diversos usuários.
- Cada usuário tem um determinado perfil em relação aos dados das tabelas ou visões.
  - Alguns usuários só podem consultar, outros atualizar e consultar, outros só inserir,outros podem tudo (DBA), etc.
- Objetivo: proteger os dados do uso indevido de qualquer usuário
  - Os privilégios garantem ***segurança*** **e** ***integridade dos dados***, bem como a responsabilidade de cada usuário sobre seus dados específicos.
- Comandos: **GRANT** e **REVOKE**

DDL - Autorizações de Acesso

- **GRANT**
  - Atribui privilégios de utilização das tabelas ou visões de uma BD

GRANT *Privilégios* **ON** *Tabelas*/Visões **TO** *LoginUsuários*

  - Privilégios podem ser:
    - Select: pode executar uma consulta sobre a tabela
    - Insert**:** pode executar uma inserção sobre a tabela
    - Delete: pode apagar registros da tabela
    - Update: pode modificar registros da tabela
    - All Privileges: pode executar qualquer operação sobre a tabela
  - Exemplos**:**
    - Grant Select On Artigo, Pesquisador To Paulo, Pedro
    - Grant Select, Insert, Update On TodosClientes To Ana
    - Grant All Privileges On Depositante To Public
      - PUBLIC = Todos os usuários da base de dados

DDL - Autorizações de Acesso

- **REVOKE**
  - Revoga os privilégios de acesso aos usuários.

  REVOKE *Privilégios* **ON** *Tabelas/Visões* **FROM** *LoginUsuários*

  - Exemplos:
    - Revoke Select On Artigo, Pesquisador From Paulo, Pedro
    - Revoke Select, Insert, Update On TodosClientes From Ana
    - Revoke All Privileges On Depositante From Public

cin.ufpe.br

Centro de Informática

U · F · P · E

Universidade Federal de Pernambuco