

FGA0137

Sistemas de Banco de Dados 1

Prof. Maurício Serrano

Material original: Profa. Elaine Parros Machado de Sousa

Prof. Jose Fernando Rodrigues Junior

2021/2

Linguagem SQL

DML

Módulo 4

DML - Introdução

- **Comandos da DML:**

- **INSERT**
- **UPDATE**
- **DELETE**
- **SELECT**

Comandos DML

- **INSERT** – insere uma ou mais tuplas em uma tabela

- Inserção de 1 tupla:

```
INSERT INTO tabela [(atrib1,atrib2,...)]  
VALUES (valor1, valor2,...)
```

- Inserção de múltiplas tuplas:

```
INSERT INTO tabela [(atrib1,atrib2,...)]  
<comando SELECT>
```

Exercício

Aluno = {Nome, NMatr, Idade, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Letra, NAlunos}

Matrícula = {Sigla, Letra, Aluno, Ano, Nota, Frequencia}

- Inserir os seguintes dados:
 - aluna de nome Juliana, nmatr 222, nascida em 10 de abril de 2001, com cidade de origem *default*
 - disciplina FGA0137, Banco de Dados, com 4 créditos.
 - matrícula da Juliana na disciplina FGA0137, turma A
- Criar uma tabela para os alunos menores de idade e alimentar com os alunos menores da tabela Aluno

Exercício

Aluno = {Nome, NMatr, Idade, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Letra, NAlunos}

Matrícula = {Sigla, Letra, Aluno, Ano, Nota, Frequencia}

■ Inserir os seguintes dados:

- aluna de nome Juliana, nmatr 222, nascida em 10 de abril de 2001, com cidade de origem *default*

```
INSERT INTO Aluno (NMatr, Nome, Idade, DataNasc) VALUES (222, 'Juliana',  
20, '10/04/2001');
```

- disciplina FGA0137, Banco de Dados, com 4 créditos

```
INSERT INTO Disciplina VALUES ('FGA0137', 'Banco de Dados', 4, 10,  
'Fundamentos de Bancos de Dados');
```

- matrícula da Juliana na disciplina FGA0137, turma A

```
INSERT INTO Turma VALUES ('FGA0137', 'A', 1);
```

COMMIT;

```
INSERT INTO Matricula (Sigla, Letra, Aluno, Ano) VALUES ('FGA0137',  
'A', 222, 2022);
```

- Criar uma tabela para os alunos menores de idade e alimentar com os alunos menores da tabela Aluno

Exercício

Aluno = {Nome, NMatr, Idade, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Letra, NAlunos}

Matrícula = {Sigla, Letra, Aluno, Ano, Nota, Frequencia}

- Criar uma tabela para os alunos menores de idade e alimentar com os alunos menores da tabela Aluno

```
CREATE TABLE Aluno_Menor(  
  NMatr NUMERIC(7) NOT NULL,  
  Nome VARCHAR(100) NOT NULL,  
  Idade SMALLINT,  
  DataNasc DATE,  
  CidadeOrigem VARCHAR(100) DEFAULT 'Brasilia',  
  
  CONSTRAINT aluno_menor_pk PRIMARY KEY(NMatr),  
  CONSTRAINT aluno_menor_un UNIQUE(Nome),  
  CONSTRAINT aluno_menor_ck CHECK(Idade < 18)  
);  
  
INSERT INTO aluno_menor SELECT * FROM aluno WHERE Idade < 18;
```

Comandos DML

- **UPDATE** – modifica o valor de um atributo em uma ou mais tuplas da tabela

UPDATE *tabela* SET

atributo1 = <valor ou expressão> ,

atributo2 = <valor ou expressão> ,

...

WHERE <condição de localização>

Comandos DML

- **DELETE** – remove uma ou mais tuplas da tabela

```
DELETE FROM tabela1 [FROM tabela2]  
[WHERE <condição de localização>]
```

Exercícios

Aluno = {Nome, NMatr, Idade, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Letra, NAlunos}

Matrícula = {Sigla, Letra, Aluno, Ano, Nota, Frequencia}

■ Atualizar os seguintes dados:

- alterar para 70% a frequência de todos os alunos com nota acima de 5.0 e frequência abaixo 70%

```
UPDATE MATRICULA SET FrequenciaPorc = 70
```

```
WHERE Nota >= 5 AND FrequenciaPorc < 70;
```

- acrescentar um crédito para as disciplinas do departamento de Matemática(MAT)

```
UPDATE Disciplina SET NCred = NCred+1
```

```
WHERE Sigla LIKE 'MAT%';
```

■ Remover os seguintes dados

- matrícula dos alunos da turma A de FGA0241

```
DELETE FROM MATRICULA WHERE Sigla = 'FGA0241' AND Letra = 'A';
```

- disciplinas com número de créditos superior a 6

```
DELETE FROM MATRICULA WHERE Ncred > 6;
```

Comandos DML

- **SELECT** – comando de consulta
 - retorno \Rightarrow tabela resultado (**multiconjunto**)

```
SELECT [DISTINCT|ALL] <lista de atributos>
FROM <lista de tabelas>
[WHERE <condições>]
[GROUP BY atributo]
[HAVING <condições>]
[ORDER BY atributo [ASC|DESC]]
```

SELECT

- **SELECT** → **O QUE** se deseja na tabela resultado
 - **ALL** – resultado pode conter tuplas duplicadas (*default*)
 - **DISTINCT** – resultado contém somente tuplas distintas
 - *<lista de atributos>* OU
 - ***** (para todos os atributos)
- **FROM** → **DE ONDE** retirar os dados necessários
- **WHERE** → **CONDIÇÕES** de consulta
 - condições de seleção
 - condições de junção

SELECT

- Cláusula **FROM** com mais de uma tabela
 - **Junção** (Join)
 - **WHERE** \Rightarrow condição de junção

```
SELECT [DISTINCT|ALL] <atributos>
FROM tabela1, tabela2
WHERE tabela1.atributo1 =
      tabela2.atributo3
```

SELECT

- Funções Agregadas
 - entrada \Rightarrow conjunto de valores
 - saída \Rightarrow valor
 - Exemplos:
 - `AVG(atributo)` \rightarrow calcula a média da coluna *atributo*
 - `COUNT()`
 - `count(*)` – retorna o número de tuplas de uma consulta
 - `count(atributo)` – retorna o nro de valores da coluna *atributo*

SELECT

- Funções Agregadas

- Exemplos

- **MAX**(*atributo*) → recupera o valor máximo da coluna *atributo*
 - **MIN**(*atributo*) → recupera o valor mínimo da coluna *atributo*
 - **SUM**(*atributo*) → obtém a soma de valores da coluna *atributo*
 - ...

SELECT

- **GROUP BY** → agrupamento de tuplas
 - para a aplicação de funções agregadas
- **HAVING** → condições aplicadas **a grupos já formados** por **GROUP BY**
- **ORDER BY** → estabelece a ordenação lógica da tabela de resultados
 - **ASC** (*default*)
 - **DESC**

SELECT

- **GROUP BY**, ou agrupamento, assume a presença de valores repetidos → portanto, apesar de aceito, não faz sentido a realização de agrupamentos sobre atributos chave
- **HAVING** estabelece a condição para a seleção de registros
- **ORDER BY** / estabelece a ordenação lógica da tabela de resultados
 - **ASC** (*default*)
 - **DESC**

SELECT

```
SELECT lista_de_atributos_de_agregação,  
       lista_de_operações_de_agregação
```

```
FROM R
```

```
GROUP BY lista_de_atributos_de_agregação
```

```
HAVING condição
```

equivale a

```
SELECT *
```

```
FROM (SELECT lista_de_atributos_de_agregação,  
            lista_de_operações_de_agregação
```

```
      FROM R
```

```
      GROUP BY lista_de_atributos_de_agregação)
```

```
WHERE condição
```

Exemplo:

Aluno = {Nome, NMatr}

{<Zeca, 11111>,
<Zico, 22222>,
<Juca, 33333>,
<Tuca, 44444> }

Matricula= {Sigla, Letra, Aluno, Ano, Nota}

{<FGA0060 A, 11111, 2010, 3.0>,
<FGA0060, A, 11111, 2011, 7.0>,
<FGA0137, A, 11111, 2009, 5.0>,
<FGA0137, A, 22222, 2009, 4.0>}

Disciplina = {Sigla, Nome}

{<FGA0137, BD1>,
<FGA0060, BD2>}

- Selecionar os nomes dos alunos que fizeram uma mesma disciplina mais de uma vez. Listar também o nome da disciplina, o nro de vezes que cursou e a nota máxima que o aluno obteve (considerando todas as vezes que cursou).

1º Passo: junção

```
select ....  
  from Aluno A join Matricula M  
           on A.NMatr = M.Aluno  
  join Disciplina D  
  on D.Sigla = M.Sigla
```

Exemplo: (continuação)

2º Passo: agrupamento e agregação

```
select A.Nome, D.Nome, count(*), max(M.Nota)
  from Aluno A join Matricula M
           on A.NMatr = M.Aluno
  join Disciplina D
    on D.Sigla = M.Sigla
 group by A.Nome, D.Nome
```

Grupo Zeca

sub-grupo BD2

<BD2, 3.0>

<BD2, 7.0>

sub-grupo BD1

<BD1, 5.0>

Grupo Zico

sub-grupo BD1

<BD1, 4.0>

Funções **COUNT** e **MAX** aplicadas sobre cada sub-grupo

Exemplo: (continuação)

3º Passo: condição having

```
select A.Nome, D.Nome, count(*), max(M.Nota)
  from Aluno A join Matricula M
           on A.NMatr = M.Aluno
  join Disciplina D
  on D.Sigla = M.Sigla
  group by A.Nome, D.Nome
  having count(*) > 1
```

Grupo Zeca

sub-grupo BD2

<BD2, 3.0>

<BD2, 7.0>

~~sub-grupo BD1~~

~~<BD1, 5.0>~~

Grupo Zico

~~sub-grupo BD1~~

~~<BD1, 4.0>~~



{A.Nome, D.Nome, count, max}
{< Zeca, BD2, 2, 7.0>}

SELECT

- Quando **nenhum** atributo de agregação é passado, nenhum atributo é considerado, isto é, todas as tuplas são IGUAIS formando um único grupo – toda a relação; por exemplo:

SELECT COUNT(*)

FROM Aluno → total de tuplas do único grupo

Ou

SELECT AVG(Nota)

FROM Matricula → média das notas do único grupo

Aluno = {Nome, NMatr}

Matricula= {Sigla, Letra, Aluno, Ano, Nota}

SELECT

- Quando **todos** os atributos da relação (ou pelo menos todos os que definem sua chave) são passados como atributos de agregação, todas as tuplas são DIFERENTES formando um número de grupos igual ao número de tuplas – isto é, a agregação produz o mesmo resultado que um SELECT sem agregação; por exemplo:

SELECT NMatr, Max(Idade)

FROM Aluno

GROUP BY NMatr

Equivale a

SELECT NMatr, Idade

FROM Aluno

SELECT Sigla, Numero, Aluno, Ano, AVG(Nota)

FROM Matricula

GROUP BY Sigla, Numero, Aluno, Ano

Equivale a

SELECT Sigla, Numero, Aluno, Ano, Nota

FROM Matricula

Aluno = {Nome, NMatr}

Matricula = {Sigla, Letra, Aluno, Ano, Nota}

SELECT

Quando ~~todos~~ os atributos da relação (ou pelo menos todos os que de
agrega Obviamente, isto é um erro que não se
deve cometer.
número de grupos igual ao número de tuplas – isto é, a
agregação produz o mesmo resultado que um SELECT sem
agregação; por exemplo:

SELECT NMatr, Max(Idade)

FROM Aluno

GROUP BY NMatr

Equivale a

SELECT NMatr, Idade

FROM Aluno

SELECT Sigla, Numero, Aluno, Ano, AVG(Nota)

FROM Matricula

GROUP BY Sigla, Numero, Aluno, Ano

Equivale a

SELECT Sigla, Numero, Aluno, Ano, Nota

FROM Matricula

Exercícios

Aluno = {Nome, NMatr, Idade, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Letra, NAlunos}

Matrícula = {Sigla, Letra, Aluno, Ano, Nota, Frequencia}

- 1) Selecionar nome, nmatr e datanasc de todos os alunos que são de Brasília.
- 2) Selecionar nmatr dos alunos que cursam a disciplina FGA0137 ou a FGA0060.
- 3) Selecionar nome e nmatr de todos os alunos matriculados em disciplinas da FGA
- 4) Selecionar nome e nmatr dos alunos, nome e sigla das disciplinas, e número de alunos da turma de todos os alunos matriculados em disciplinas da FGA

Exercícios

Aluno = {Nome, NMatr, Idade, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Idade, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor, Livro}

Turma = {Sigla, Letra, NAlunos}

Matrícula = {Sigla, Letra, Aluno, Ano, Nota, Frequencia}

- 5) Selecionar, para cada aluno, seu nome e a média das notas das disciplinas que cursou. Ordenar por nome de aluno.
- 6) Selecionar, para cada aluno, seu nome e a média das notas das disciplinas em que foi aprovado (nota ≥ 5). Ordenar por nome de aluno.
- 7) Selecionar os nomes dos alunos que fizeram uma mesma disciplina mais de uma vez. Listar também a sigla da disciplina, o nro de vezes que cursou e a nota máxima que o aluno obteve (considerando todas as vezes que cursou).

Leitura recomendada

- R. Elmasri, S. Navathe: *Fundamentals of Database Systems* – 4th Edition
 - Capítulo 8
- A. Silberschatz, H. F. Korth, s. Sudarshan: *Sistema de Banco de Dados*
 - Capítulo 4