



Bases de Dados

Linguagem SQL – DML (Parte2)

Profa. Elaine Parros Machado de Sousa

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 resultantes de uma consulta
 - **count(*atributo*)** – retorna o nro de valores da coluna *atributo* (não contabiliza NULL)

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*

- ...

Exercícios

Aluno = {Nome, Nusp, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor}

Turma = {Sigla, Numero, NAlunos}

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

1)

2)

3)

4) ...

5) Selecionar a quantidade total de matrículas na disciplina Banco de Dados

5.1) Selecionar a quantidade de alunos distintos que cursam ou já cursaram a disciplina Banco de Dados

SELECT

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

- **ORDER BY** → estabelece a ordenação lógica da tabela de resultados
 - **ASC** (*default*)
 - **DESC**

Exercícios

Aluno = {Nome, Nusp, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor}

Turma = {Sigla, Numero, NAlunos}

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

1) ...

2) ...

3) ...

4) ...

5) ...

6) Selecionar nome e nusp dos alunos, nome e sigla das disciplinas, e nro da turma para todos os alunos matriculados em disciplinas do SCC. Ordenar o resultado por nusp do aluno e por sigla.

Exercícios

Aluno = {Nome, Nusp, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor}

Turma = {Sigla, Numero, NAlunos}

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

7) Selecionar NUSP e idade de todos os alunos que vieram de São Paulo. Ordene o resultado por idade.

7.1) Selecionar a idade média dos alunos que vieram de São Paulo.

SELECT

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


SELECT

- **GROUP BY** → agrupamento de tuplas
 - **para a aplicação de funções agregadas**
 - **HAVING** → condições aplicadas a **grupos** já formados por **GROUP BY**

Exercícios

Aluno = {Nome, Nusp, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor}

Turma = {Sigla, Numero, NAlunos}

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

- 8) Selecionar, para cada aluno, seu nusp e a média das notas das disciplinas em que foi aprovado (nota ≥ 5). Ordenar por nusp de aluno

Consulta 8) Selecionar, para cada aluno, seu nusp e a média das notas das disciplinas em que foi aprovado (nota >= 5). Ordenar por nusp de aluno

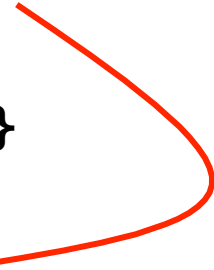
Matricula= {Sigla, Numero, Aluno, Ano, Nota}

{<SCC-125, 1, 11111, 2010, 5.0>,
<SCC-148, 1, 11111, 2010, 7.0>,
<SCC-125, 2, 22222, 2010, 5.0>,
<SCC-148, 1, 22222, 2009, 4.0>}

1º Passo: seleção

```
SELECT ...  
FROM Matricula WHERE Nota >= 5.0
```

{Sigla, Aluno, ... Nota}
{<SCC-125, ... 11111 5.0>,
<SCC-148, 11111 7.0>,
<SCC-125,22222.... 5.0>}



(continuação)

2º Passo: agrupamento e agregação

```
SELECT Aluno, AVG(Nota) as Media  
FROM Matricula WHERE Nota >= 5.0  
GROUP BY Aluno  
ORDER BY Aluno;
```

Grupo aluno 11111

<SCC125,...11111, ... 5.0>
<SCC148,...11111,7.0>

Grupo aluno 22222

<SCC125,... 22222, ... 5.0>

Função **AVG** aplicada sobre cada grupo



{Aluno, Media}
{<11111, 6.0>,
<22222, 5.0>}

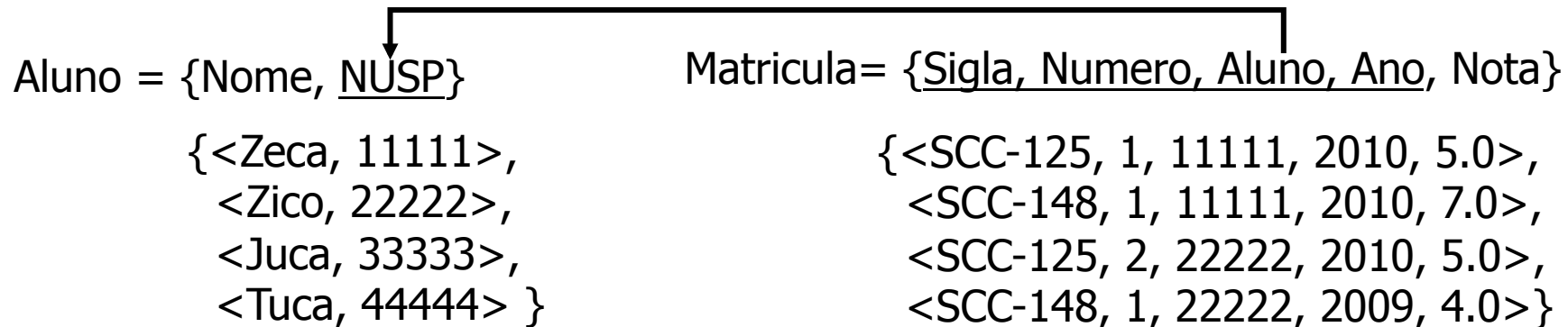
Consulta 8)

```
SELECT Aluno, AVG(Nota) as Media  
FROM Matricula WHERE Nota >= 5.0  
GROUP BY Aluno  
ORDER BY Aluno;
```

E se a consulta for:

Consulta 8.1) Selecionar, para cada aluno, seu nusp, **nome** e a média das notas das disciplinas em que foi aprovado (nota >= 5). Ordenar por **nome** de aluno

Consulta 8.1) Selecionar, para cada aluno, seu nusp, nome e a média das notas das disciplinas em que foi aprovado (nota >= 5). Ordenar por nome de aluno



1º Passo: seleção e junção

```
SELECT ...  
  FROM Aluno A JOIN Matricula M  
           ON M.Aluno = A.NUSP  
 WHERE M.Nota >= 5.0
```

{Nusp, Nome, Sigla, Nota}
{<11111, Zeca, SCC-125, 5.0>, <11111, Zeca, SCC-148, 7.0>, <22222, Zico, SCC-125, 5.0>}

(continuação)

2º Passo: agrupamento e agregação

```
SELECT A.NUSP, A.Nome, AVG(M.Nota) as Media
FROM Aluno A JOIN Matricula M
      ON M.Aluno = A.NUSP
WHERE M.Nota >= 5.0
GROUP BY A.NUSP, A.Nome
ORDER BY A.Nome;
```

{Nusp, Nome, Media}
{<11111, Zeca, 6.0>,
 <22222, Zico, 5.0>}

Grupo aluno 11111

<11111, Zeca ... 5.0>
<11111, Zeca.....7.0>

Sub-grupo Zeca

Grupo aluno 22222

<22222, Zico, ... 5.0>

Sub-grupo Zico

Função **AVG** aplicada sobre cada sub-grupo

Exercícios

Aluno = {Nome, Nusp, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Titulação}

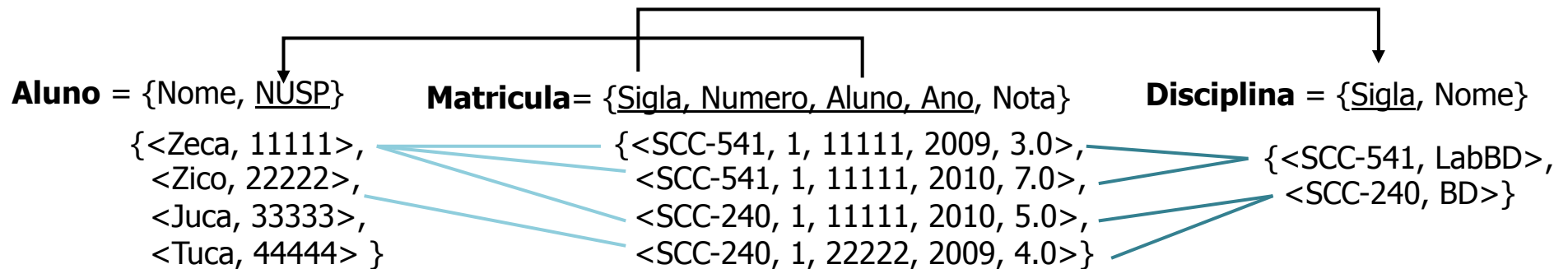
Disciplina = {Sigla, Nome, NCred, Professor}

Turma = {Sigla, Numero, NAlunos}

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

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

Consulta 9) Selecionar nusp e nome dos alunos que fizeram alguma disciplina mais de uma vez. Listar também a sigla e o nome da disciplina, o nro de vezes que o aluno cursou a disciplina e a nota máxima que obteve (considerando todas as vezes que cursou).



1º Passo: junção

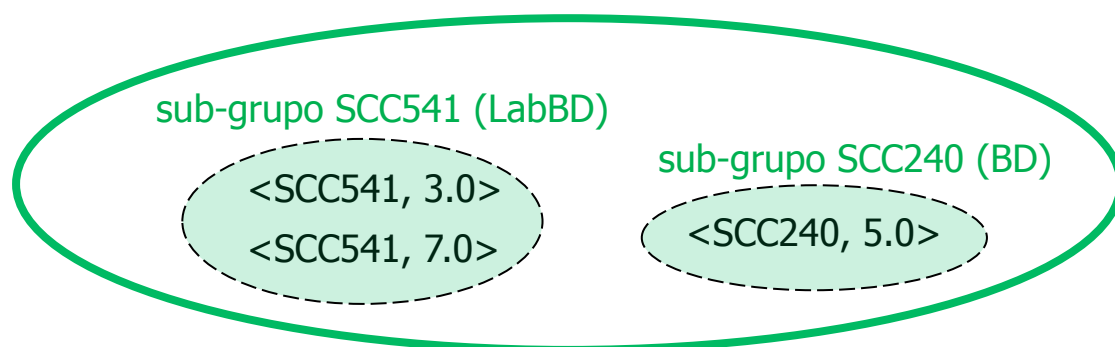
```
select ....
  from Aluno A join Matricula M
           on A.NUSP = M.Aluno
  join Disciplina D
  on D.Sigla = M.Sigla
```

(continuação)

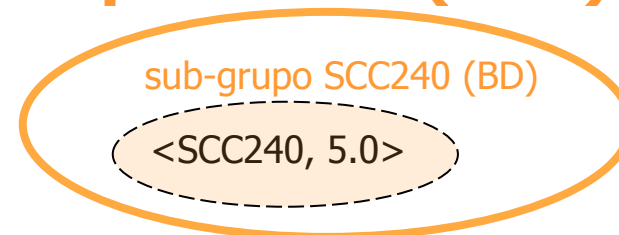
2º Passo: agrupamento e agregação

```
select .....  
  from Aluno A join Matricula M  
           on A.NUSP = M.Aluno  
  join Disciplina D  
           on D.Sigla = M.Sigla  
 group by A.NUSP, A.Nome, D.Sigla, D.Nome
```

Grupo 11111 (Zeca)



Grupo 22222 (Zico)

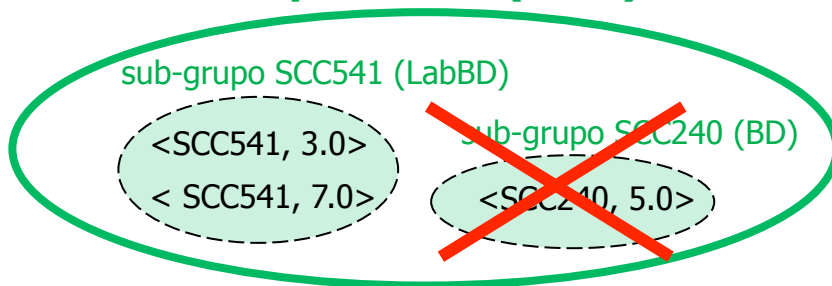


(continuação)

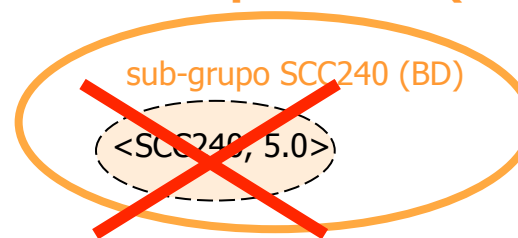
3º Passo: condição having

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

Grupo 11111 (Zeca)



Grupo 22222 (Zico)



Funções **COUNT** e **MAX** aplicadas sobre cada sub-grupo (no último nível)

{A.NUSP, A.Nome, D.Sigla, D.Nome, count, max}
{<11111, Zeca, SCC541, LabBD, 2, 7.0>}

Exercícios

Aluno = {Nome, Nusp, DataNasc, CidadeOrigem}

Professor = {Nome, NFunc, Titulação}

Disciplina = {Sigla, Nome, NCred, Professor}

Turma = {Sigla, Numero, NAlunos}

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

- 10) Selecionar a quantidade de alunos por turma por ano;
- 11) Selecionar NUSP dos alunos que **não cursaram** nenhuma disciplina nos anos de 2018 e 2019.
 - 11.1) Remover os alunos que não cursaram nenhuma disciplina nos anos de 2018 e 2019.
- 12) Selecionar nome e NUSP dos alunos mais novos (menor idade).



Consultas Aninhadas (Nested Queries)

Consultas Aninhadas (Nested Queries)

- **Não correlacionadas** – consultas independentes
 - ex: selecionar nome e nusp dos alunos mais velhos (com a maior idade)

```
select nome, nusp from aluno
where DataNasc IN
    (select min(DataNasc)
     from aluno);
```

Consultas Aninhadas

- **Correlacionadas** – condição na cláusula WHERE da consulta interna usa algum atributo de tabela declarada na consulta externa
 - consulta interna é executada uma vez para cada tupla avaliada na consulta externa

EXEMPLO:

Aluno = {Nome, Nusp, DataNasc}

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

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

- Selecionar **nusp**, **sigla da disciplina** e **nota** dos alunos que obtiveram **nota maior que a média** dos alunos que cursaram a **mesma disciplina no mesmo ano**.

```
select M.aluno, M.sigla, M.ano, M.nota
from Matricula M
where M.nota > (select avg(M1.nota)
               from Matricula M1
               where M1.sigla = M.sigla and
                     M1.ano = M.ano) ;
```


EXEMPLO:

Aluno = {Nome, Nusp, DataNasc}

Disciplina = {Sigla, Nome, NCred, Professor}

Matrícula = {Sigla, Numero, Aluno, Ano, Nota}

- Selecionar nome e nusp dos alunos matriculados, em 2019, em **todas** as disciplinas ministradas pelo professor com nfunc 123.

```
select nome, nusp from aluno A where
    NOT EXISTS ( (select sigla from disciplina
                    where professor = 123)
                MINUS
                (select sigla from matricula
                    where aluno = A.nusp and
                      ano = 2019)
    )
```

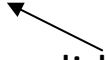


Planos de Consulta – Otimizador Oracle

Exemplo:

Candidato = {codigo, ...}

TelefoneCandidato = {candidato, telefone}



```
-- 3 maneiras de fazer a mesma consulta:  
-- selecionar os códigos de todos os candidatos que  
-- possuem telefone cadastrado
```

```
-- teste 1 (sem junção)
```

```
select candidato from TelefoneCandidato;
```

```
-- teste 2 (com junção)
```

```
select codigo from Candidato join  
    TelefoneCandidato on codigo = candidato;
```

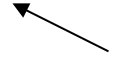
```
-- teste 3 (consulta correlacionada)
```

```
select C.codigo from Candidato C  
    where exists  
        (select * from TelefoneCandidato T  
            where t.candidato = C.codigo);
```

Exemplo: planos de consulta – otimizador ORACLE

Candidato = {codigo, ...}

TelefoneCandidato = {candidato, telefone}



-- Plano de consulta: teste 1 (sem junção)

SELECT STATEMENT

INDEX FULL SCAN PK_TELEFONECANDIDATO

```
select candidato from
  TelefoneCandidato;
```

-- Plano de consulta: teste 2 (com junção)

SELECT STATEMENT

INDEX FULL SCAN PK_TELEFONECANDIDATO

```
select codigo
from Candidato join
  TelefoneCandidato on
  codigo = candidato;
```

-- Plano de consulta: teste 3 (consulta correlacionada)

SELECT STATEMENT

NESTED LOOPS

SORT UNIQUE

INDEX FULL SCAN PK_TELEFONECANDIDATO

INDEX UNIQUE SCAN PK_CANDIDATO

```
select C.codigo
from Candidato C
where exists
  (select * from
    TelefoneCandidato T
   where t.candidato
        = C.codigo);
```

Leitura recomendada

- R. Elmasri, S. Navathe: *Sistemas de Banco de Dados*
 - 4ª Edição
 - Capítulos 8 e 9
 - 6ª Edição
 - Capítulos 4 e 5
- A. Silberschatz, H. F. Korth, s. Sudarshan: *Sistema de Banco de Dados*
 - Capítulo 4
- Oracle - Manuais em *list of books* no site
 - ***SQL Language Reference***
 - ***Performance Tuning Guide***