

Banco de Dados I

Prof. Diego Buchinger
diego.buchinger@outlook.com
diego.buchinger@udesc.br

Profa. Rebeca Schroeder Freitas
Prof. Fabiano Baldo

DML – *Data Manipulation Language*

Parte II

Consultas Básicas

- Para consultar dados de uma tabela utiliza-se:

```
SELECT lista_de_colunas  
FROM tabela  
[WHERE condição]
```

- *lista_de_colunas* pode ser substituída por asterisco (*) que representa todos os atributos da tabela

- Mapeamento entre SQL e álgebra relacional:

```
SELECT a1, ..., an  
FROM t  
WHERE c
```


$$\pi_{a_1, \dots, a_n}(\sigma_c(t))$$

Consultas Básicas

- Exemplos:

Álgebra Relacional

(Pacientes)

$\sigma_{idade > 18} (\text{pacientes})$

$\pi_{cpf, nome} (t)$

$\pi_{cpf, nome} (\sigma_{idade > 18} (\text{pacientes}))$

SQL

```
SELECT *  
FROM pacientes
```

```
SELECT *  
FROM pacientes  
WHERE idade > 18
```

```
SELECT cpf, nome  
FROM pacientes
```

```
SELECT cpf, nome  
FROM pacientes  
WHERE idade > 18
```

Consultas Básicas

Particularidade na projeção

- Não há eliminação automática de duplicatas
 - Tabela \equiv coleção
 - Para eliminar duplicatas deve-se usar o termo `distinct`

```
SELECT DISTINCT doenca FROM consultas
```

- É possível renomear os campos / colunas (`AS`)
 - Operador ρ (*rho*) na álgebra relacional

```
SELECT codp AS codigo_paciente,  
      codm AS codigo_medico, data  
FROM consultas
```

Consultas Básicas

Particularidade na projeção

- É possível utilizar operadores aritméticos e funções
 - Quantos grupos de 5 leitos podem ser formados em cada ambulatório?

```
SELECT nroa, capacidade/5 AS cap5  
FROM ambulatorios
```

- Qual o salário líquido dos funcionários sabendo que há um desconto de 12,63% sobre o salário base?

```
SELECT nome, ROUND( salario * 0.8737 , 2) AS  
salario_liquido FROM funcionarios
```

Função **ROUND**: parâmetros (valor : numeric , casas : int | numeric)

Consultas Básicas

Particularidade na projeção

- Pode-se usar funções de agregação / agrupamento
 - COUNT: contador de ocorrências [registros ou atributos]

OBS: Conta valores não nulos

```
SELECT COUNT(*) FROM medicos  
WHERE especialidade = 'ortopedia'
```

- MAX / MIN: valor máximo / mínimo de um atributo

```
SELECT MAX(salario) AS maior_salario FROM funcionarios
```

- SUM: soma os valores de um dado atributo
 - ❖ Qual é o gasto total com a folha de pagamento dos funcionários?
- AVG: contabiliza a média dos valores de um dado atributo
 - ❖ Qual é a média de idade dos funcionários de Florianópolis?

Consultas Básicas

Particularidade na projeção

- Pode-se misturar funções de agregação com `distinct`

```
SELECT COUNT (DISTINCT especialidade)  
FROM medicos
```

- Não podem ser combinados outros campos junto com funções de agregação

```
SELECT andar, COUNT (andar)  
FROM ambulatorios
```

- Pode-se realizar *casting* de tipos usando: `campo::tipo`

```
SELECT nome, cpf::text, idade::numeric(3,1)  
FROM pacientes
```


Consultas Básicas

Particularidade na seleção

- Procurar por valores nulos ou não nulos

➤ cláusula `IS [NOT] NULL`

```
SELECT cpf, nome  
FROM medicos WHERE nroa IS NULL
```

- Procurar por intervalos de valores

➤ Cláusula `[NOT] BETWEEN valor1 AND valor2`

```
SELECT * FROM consultas  
WHERE hora BETWEEN '13:00' AND '18:00'
```

Consultas Básicas

Particularidade na seleção

- Procurar por pertinência em conjunto ou coleção

➤ cláusula **[NOT] IN**

```
SELECT * FROM medicos  
WHERE especialidade IN  
('ortopedia', 'traumatologia')
```

```
SELECT codm, codp, data FROM consultas  
WHERE codm IN (  
    SELECT codm FROM medicos  
    WHERE idade > 40  
)
```

Consultas Básicas

Particularidade na seleção

- Procurar por padrões
 - Cláusula `[NOT] LIKE`
 - Pode receber os seguintes padrões
 - % casa com qualquer cadeia de caracteres
 - _ casa com um único caractere
 - [a-d] casa com qualquer caractere entre as letras apresentadas (SQL-Server)
- ❖ Buscar médicos que nome iniciando por 'M'
- ❖ Buscar médicos com um número exato de décadas de vida

```
SELECT * FROM medicos  
WHERE nome LIKE 'M%'
```

```
SELECT * FROM medicos  
WHERE idade::text LIKE '_0'
```

Consultas Básicas

Particularidade na seleção

- Procurar por padrões

- ❖ Buscar por consultas marcadas para o mês de julho

```
SELECT * FROM consultas  
WHERE data::text LIKE '%/07/%'
```

- ❖ Buscar por pacientes cujo CPF termina com 20000 ou 30000

```
SELECT * FROM pacientes  
WHERE cpf::text LIKE '%20000'  
      OR cpf::text LIKE '%30000'
```

Consultas Básicas

Particularidade na seleção

- Algumas funções básicas (variam entre SGBDs)
 - ❖ Verificar tamanho (# de caracteres) de um campo textual

```
SELECT lenght(doenca), doenca FROM consultas
```

- ❖ Resgatar uma substring – substring(texto, começo, #chars)

```
SELECT substring(data::text from 5 for 2)  
FROM consultas
```

- ❖ Alterar um texto específico em um campo textual
replace(texto, padrão_a_procurar, texto_substituto)

```
SELECT replace(data::text, '/06/', '/07/' )  
FROM consultas
```

União de Tabelas

- SQL implementa a união da álgebra relacional
 - Lembre-se, as tabelas devem ser compatíveis!

Álgebra Relacional

relação-1 U relação-2

SQL

SQL-1 **UNION** SQL-2

❖ Buscar o nome e o CPF dos médicos e funcionários

```
SELECT cpf, nome FROM medicos
UNION
SELECT cpf, nome FROM pacientes
```

Atividade

1. Realize as seguintes consultas no BD:
 - a. Buscar o nome e o CPF dos médicos com menos de 40 anos ou com especialidade diferente de traumatologia
 - b. Buscar todos os dados das consultas marcadas no período da tarde após o dia 19/06/2006
 - c. Buscar o nome e a idade dos pacientes que não residem em Florianópolis
 - d. Buscar a hora das consultas marcadas antes do dia 14/06/2006 e depois do dia 20/06/2006
 - e. Buscar o nome e a idade (em meses) dos pacientes
 - f. Buscar o menor e o maior salário dos funcionários de Florianópolis

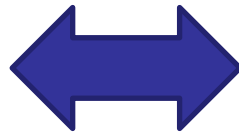
Atividade

- g. Qual o horário da última consulta marcada para o dia 13/06/2006?
- h. Qual a média de idade dos médicos e o total de ambulatorios atendidos por eles?
- i. Buscar o código, o nome e o salário líquido dos funcionários que recebem mais do que \$ 1200. Para quem ganha acima de \$1200, há um desconto de 12% do salário bruto
- j. Buscar o nome dos funcionários que terminam com a letra 'a'
- k. Buscar o nome e idade dos funcionários que possuem o número 6 em seu CPF
- l. Em quais cidades residem os funcionários e médicos da clínica

Consultas Envolvendo Múltiplas Tabelas

- SQL implementa a operação de **PRODUTO CARTESIANO**
 - Relaciona todos os registros de uma tabela com todos os registros de outra tabela
 - Geralmente é necessário vincular as duas tabelas através de uma condição
- Mapeamento entre SQL e álgebra relacional:

```
SELECT a1, ..., an  
FROM t1, ..., tm  
WHERE c
```


$$\pi_{a_1, \dots, a_n} (\sigma_c (t_1 \times \dots \times t_m))$$

Consultas Envolvendo Múltiplas Tabelas

- ❖ Trazer o CPF e nome dos pacientes e suas respectivas datas de consultas agendadas para o período da tarde

```
SELECT cpf, nome, data  
FROM pacientes, consultas  
WHERE hora > '12:00' AND  
       pacientes.codp = consultas.codp
```

- ❖ Trazer os nomes dos médicos que tem a mesma especialidade do médico 'João'
[neste caso é necessário renomear uma tabela utilizando AS]

OBS: Quando as colunas são iguais é necessário usar o nome da tabela como prefixo. Assim, é comum renomear as tabelas com nomes mais enxutos

Consultas Envolvendo Múltiplas Tabelas

- SQL implementa as operações de **JUNÇÃO**

```
SELECT a1, ..., an  
FROM t1 [INNER] JOIN t2  
      ON condição_junção  
WHERE c
```

- ❖ Trazer o CPF e nome dos pacientes e suas respectivas datas de consultas agendadas para o período da tarde

```
SELECT p.cpf, p.nome, c.data  
FROM pacientes AS p JOIN consultas AS c  
      ON p.codp = c.codp  
WHERE hora > '12:00'
```

Consultas Envolvendo Múltiplas Tabelas

- **JUNÇÃO NATURAL**

- Junção utilizando as colunas que possuem mesmo nome entre as tabelas relacionadas. Assim, a condição não é declarada

```
SELECT a1, ..., an  
FROM t1 NATURAL JOIN t2  
WHERE c
```

- ❖ Trazer o CPF e nome dos médicos e suas respectivas datas de consultas agendadas para o período da manhã

```
SELECT m.cpf, m.nome, c.data  
FROM medicos AS m NATURAL JOIN consultas AS c  
WHERE hora <= '12:00'
```

Consultas Envolvendo Múltiplas Tabelas

- **JUNÇÕES EXTERNAS**

- Junções que mantêm os elementos (linhas) não relacionados de uma ou mais tabelas no resultado

```
SELECT a1, ..., an  
FROM t1 LEFT|RIGHT|FULL [OUTER] JOIN t2  
      ON condição_junção  
WHERE c
```

- ❖ Listar todos os pacientes e, caso existam, as datas e horários de suas consultas.

```
SELECT p.cpf, p.nome, c.data  
FROM pacientes AS p LEFT JOIN consultas AS c  
      ON p.codp = c.cod
```

Consultas Envolvendo Múltiplas Tabelas

- **JUNÇÕES EXTERNAS**

- ❖ Listar todos os médicos e funcionários (nome e cidade), vinculando aqueles que moram em uma mesma cidade

```
SELECT f.nome, f.cidade, m.nome, m.cidade  
FROM funcionarios AS f FULL JOIN medicos AS m  
      ON f.cidade = m.cidade
```

Consultas Aninhadas

- Já vimos que é possível realizar consultas alinhadas usando a cláusula [NOT] IN

```
SELECT codm, codp, data FROM consultas
WHERE codm IN (
    SELECT codm FROM medicos
    WHERE idade > 40
)
```

- É possível realizar operações de diferença e interseção

$\pi_{CPF}(Funcionarios) - \pi_{CPF}(Pacientes)$

```
SELECT cpf FROM funcionarios
WHERE cpf NOT IN (
    SELECT cpf FROM pacientes
)
```

$\pi_{CPF}(Medicos) \cap \pi_{CPF}(Pacientes)$

```
SELECT cpf FROM medicos
WHERE cpf IN (
    SELECT cpf
    FROM pacientes
)
```

Consultas Aninhadas

- Pode-se fazer uso também de **Subconsultas unitárias**
 - Cardinalidade da subconsulta = 1
 - Neste caso não é necessário utilizar cláusula de subconsulta

❖ Buscar o nome e CPF dos médicos que possuem a mesma especialidade do que o médico de CPF 10000100000

```
SELECT nome, cpf FROM medicos
WHERE cpf != 10000100000 AND
      especialidade = (
        SELECT especialidade FROM medicos
        WHERE cpf = 10000100000
      )
```


Consultas Aninhadas

- Existem ainda as cláusulas **ANY** , **ALL** e **EXISTS** (Cálculo Relacional)
 - **ANY**: testa se uma dada condição é verdadeira para pelo menos um valor da consulta aninhada
 - ❖ Buscar o nome e a idade dos médicos que são mais velhos do que pelo menos um funcionário

```
SELECT nome, idade FROM medicos
WHERE idade > ANY (
    SELECT idade FROM funcionarios
)
```

Consultas Aninhadas

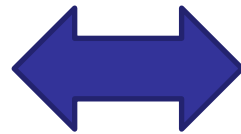
- Existem ainda as cláusulas **ANY** , **ALL** e **EXISTS** (Cálculo Relacional)
 - **ALL**: testa se uma dada condição é verdadeira para todos os valores de uma consulta aninhada
 - ❖ Buscar o nome e a idade dos médicos que são mais velhos do que todos os funcionários de Florianópolis

```
SELECT nome, idade FROM medicos
WHERE idade > ALL (
    SELECT idade FROM funcionarios
    WHERE cidade = 'Florianopolis'
)
```

Consultas Aninhadas

- Existem ainda as cláusulas **ANY** , **ALL** e **EXISTS** (Cálculo Relacional)
 - **EXISTS**: testa se um predicado é verdadeiro ou falso; a subconsulta é executada para cada linha da consulta externa
- Mapeamento entre SQL e álgebra relacional:

```
SELECT a1, ..., an
FROM t1
WHERE EXISTS
  (SELECT * FROM t2
   WHERE d > 5 AND
    t2.x = t1.c)
```


$$\{t_1.a_1, \dots, t_1.a_n \mid t_1 \in T_1 \wedge \\ \exists t_2 \in T_2 \\ (t_2.d > 5 \wedge \\ t_2.x = t_1.c) \}$$

Consultas Aninhadas

- Existem ainda as cláusulas **ANY** , **ALL** e **EXISTS** (Cálculo Relacional)
 - **EXISTS**: testa se um predicado é verdadeiro ou falso; a subconsulta é executada para cada linha da consulta externa

❖ Buscar o nome dos médicos que possuem uma consulta para o dia 06 de novembro de 2013

$\{ m.nome \mid m \in medicos \wedge$
 $\exists c \in consultas$
 $(c.data = '2013/11/06'$
 $\wedge c.codm = m.codm) \}$

```
SELECT nome FROM medicos AS m
WHERE EXISTS (
    SELECT * FROM consultas
    WHERE data = '2013/11/06'
    AND codm = m.codm )
```

Consultas Aninhadas

- Existem ainda as cláusulas **ANY** , **ALL** e **EXISTS** (Cálculo Relacional)
 - **EXISTS**: testa se um predicado é verdadeiro ou falso; a subconsulta é executada para cada linha da consulta externa
- ❖ Buscar o nome dos funcionários de Florianópolis que nunca se consultaram como pacientes na clínica

$\{ f.nome \mid f \in funcionarios \wedge$
 $f.cidade = 'Florianopolis'$
 $\neg \exists p \in pacientes$
 $(p.cpf = f.cpf) \}$

?

Consultas Aninhadas

- Subconsulta na cláusula **FROM / JOIN**
 - Consulta externa é feita sobre um subconjunto resposta
 - Útil para otimização filtrando linhas e colunas antecipadamente

❖ Buscar os dados dos médicos e a hora das consultas que estão agendadas para o dia 06/11/2013

```
SELECT medicos.*, c.hora
FROM medicos JOIN
    (SELECT codm, hora FROM consultas
     WHERE data = '2013/11/06')
AS c ON medicos.codm = c.codm
```

Consultas Aninhadas

- Subconsulta na cláusula **FROM** / **JOIN**
 - Consulta externa é feita sobre um subconjunto resposta
 - Útil para otimização filtrando linhas e colunas antecipadamente

❖ Buscar os números e andares dos ambulatórios em que médicos de Florianópolis dão atendimento

```
SELECT amb.* FROM
  (SELECT nroa, andar FROM ambulatorios)
AS amb JOIN
  (SELECT nroa FROM medicos
   WHERE cidade = 'Florianopolis')
AS mflo ON amb.nroa = mflo.nroa
```

Atividade

2. Realize as seguintes consultas usando produto cartesiano ou junção (quando possível use junção natural):
 - a. Buscar nome e CPF dos médicos que são pacientes do hospital
 - b. Buscar os nomes dos funcionários e médicos que residem numa mesma cidade; mostrar também qual é a cidade
 - c. Buscar código e nome dos pacientes com consulta marcada para horários após às 14 horas
 - d. Buscar o número e andar dos ambulatórios utilizados por médicos ortopedistas
 - e. Buscar nome e CPF dos pacientes que têm consultas marcadas entre os dias 14 e 16 de junho de 2006.
 - f. Buscar o nome e a idade dos médicos que têm consulta com a paciente Ana

Atividade

- g. Buscar o código e nome dos médicos que atendem no mesmo ambulatório do médico Pedro
- h. Buscar o nome, cpf e idade dos pacientes que tem consultas marcadas com ortopedistas para antes do dia 16/06/2006
- i. Nome e salário dos funcionários que moram na mesma cidade do funcionário Carlos e possuem salário superior ao dele
- j. Buscar os dados de todos os ambulatórios e, para aqueles onde médicos dão atendimento, exibir também os seus nomes.
- k. Para cada consulta listar o nome do médico, o nome do paciente, a data, o horário e o ambulatório utilizado.

Atividade

3. Realize as seguintes consultas usando subconsultas com IN, ANY, ALL e/ou EXISTS, ou subconsultas na cláusula FROM:
 - a. Buscar nome e CPF dos médicos que são pacientes do hospital
 - b. Buscar código e nome dos pacientes com consulta marcada para horários após às 14 horas.
 - c. Buscar o número e andar dos ambulatórios onde nenhum médico dá atendimento.
 - d. Buscar o número e o andar de todos os ambulatórios, exceto o de menor capacidade.
 - e. Buscar nome e CPF dos médicos que não atendem em ambulatórios com capacidade superior à capacidade dos ambulatórios do segundo andar.
 - f. Buscar nome e CPF dos médicos que têm consultas marcadas com todos os pacientes.

Ordenar Tuplas Resultantes

- Resultados podem ser ordenadas pela cláusula **ORDER BY**

```
SELECT lista_atributos
FROM lista_tabelas
[WHERE condições]
[ORDER BY nome_atrib_1 [ASC|DESC]
{[, nome_atrib_n [ASC|DESC]]}]
```

- Mapeamento entre SQL e álgebra relacional:

$\tau_{idade\ asc, nome\ desc}(\pi_{nome, idade}(Funcionarios))$

```
SELECT nome, idade FROM pacientes
ORDER BY idade ASC, nome DESC
```

Limitar Tuplas Resultantes

- Resultados podem ser limitados pela cláusula **LIMIT**

```
SELECT lista_atributos  
FROM lista_tabelas  
[WHERE condições]  
[ORDER BY regras]  
[LIMIT v1 [,v2]]
```

- Se apenas v_1 é utilizado ele representa o número de tuplas
- Se v_1 e v_2 forem utilizados, v_1 representa quantos registros iniciais devem ser pulados e v_2 representa o número de tuplas

```
SELECT lista_atributos  
FROM lista_tabelas  
[WHERE condições]  
[ORDER BY regras]  
[LIMIT qtd OFFSET ini]
```

- No **PostgreSQL** deve-se utilizar **LIMIT** para indicar o número de tuplas e **OFFSET** para indicar quantos registros iniciais devem ser pulados.

Limitar Tuplas Resultantes

- Resultados podem ser limitados pela cláusula **LIMIT**

- ❖ Retornar o nome e a idade dos 3 pacientes mais velhos

```
SELECT nome, idade FROM pacientes  
ORDER BY idade DESC LIMIT 3
```

- ❖ Retornar o nome, a idade e o salario dos funcionários que recebem o segundo e terceiro maior salário

```
SELECT nome, salario  
FROM funcionarios  
ORDER BY salario DESC  
LIMIT 1, 2
```

```
SELECT nome, salario  
FROM funcionarios  
ORDER BY salario DESC  
LIMIT 2 OFFSET 1
```

Agrupar Tuplas

- Tuplas podem ser agrupados pela cláusula **GROUP BY**
 - Agrupa as tuplas que possuem mesmo valor nas colunas especificadas para o agrupamento
 - Apenas os atributos de agrupamento podem aparecer no resultado final da consulta
 - Geralmente é utilizada alguma **função de agregação** (ex: contagem, somatório) sobre o resultado da consulta

```
SELECT lista_atributos
FROM lista_tabelas
[WHERE condições]
[GROUP BY lista_atributos_agrupamento
 [HAVING condição_para_agrupamento]]
```

Agrupar Tuplas

- Tuplas podem ser agrupados pela cláusula **GROUP BY**
 - Funções de agregação: COUNT, SUM, AVG, MIN, MAX

- ❖ Listar quantos médicos existem por especialidade

```
SELECT especialidade, COUNT(*)  
FROM medicos  
GROUP BY especialidade
```

- ❖ Qual é a média de salários pagos aos funcionários por sua cidade de origem?

```
SELECT cidade, AVG(salario)  
FROM funcionarios  
GROUP BY cidade
```

Agrupar Tuplas

- Tuplas podem ser agrupados pela cláusula **GROUP BY**
 - Opcionalmente pode-se utilizar a cláusula **HAVING** para aplicar condições sobre os grupos que são formados
 - As condições só podem ser definidas sobre atributos do agrupamento ou sobre funções de agregação
- ❖ Listar as cidades que são origem de pelo menos mais do que um paciente e informar quantos pacientes são dessas cidades

```
SELECT cidade, COUNT(*)  
FROM pacientes  
GROUP BY cidade  
HAVING COUNT(*) > 1
```


Atualização com Consultas

- Comandos de atualização (INSERT, UPDATE e DELETE) podem incluir comandos de consulta

❖ Ex1: a médica Maria pediu para cancelar todas as suas consultas após as 17:00

```
DELETE FROM consultas
WHERE hora > '17:00'
AND codm IN
  (SELECT codm FROM medicos
   WHERE nome = 'Maria')
```

Atualização com Consultas

- Comandos de atualização (INSERT, UPDATE e DELETE) podem incluir comandos de consulta
- ❖ Ex2: a direção da clínica determinou que deve haver sempre dois médicos por ambulatório, caso contrário o médico não deve ter um ambulatório definido/fixo

```
UPDATE medicos  
SET nroa = NULL  
WHERE NOT EXISTS  
  (SELECT * FROM medicos AS m  
   WHERE m.codm != medicos.codm  
   AND m.nroa = medicos.nroa)
```

Atualização com Consultas

- Comandos de atualização (INSERT, UPDATE e DELETE) podem incluir comandos de consulta
 - ❖ Ex3: os leitos do ambulatório 4 foram transferidos para o ambulatório de número 2.

?

Atividade

4. Realize as seguintes consultas:
 - a. Mostre os dados de todos os funcionários ordenados pelo salário (decrescente) e pela idade (crescente). Buscar apenas os três primeiros funcionários nesta ordem.
 - b. Mostre o nome dos médicos, o número e andar do ambulatório onde eles atendem, ordenado pelo número do ambulatório
 - c. Mostre o nome do médico e o nome dos pacientes com consulta marcada, ordenado pela data e pela hora. Buscar apenas as tuplas 3 a 5, nesta ordem.
 - d. Mostre as idades dos médicos e o total de médicos com a mesma idade
 - e. Mostre as datas e o total de consultas em cada data, para horários após às 12 horas

Atividade

- f. Mostrar os andares onde existem ambulatórios e a média de capacidade por andar
- g. Mostrar os andares onde existem ambulatórios e a média de capacidade no andar seja maior ou igual a 40
- h. Mostrar o nome dos médicos que possuem mais de uma consulta marcada
- i. Passar todas as consultas da paciente Ana para às 19:00
- j. Excluir os pacientes que não possuem consultas marcadas
- k. Passar todas as consultas do médico Pedro marcadas para o período da manhã para o dia 21/11/2006, no mesmo horário
- l. O ambulatório 4 foi transferido para o mesmo andar do ambulatório 1 e sua capacidade é agora o dobro da capacidade do ambulatório de maior capacidade da clínica