



DISCIPLINA: Banco de Dados 1

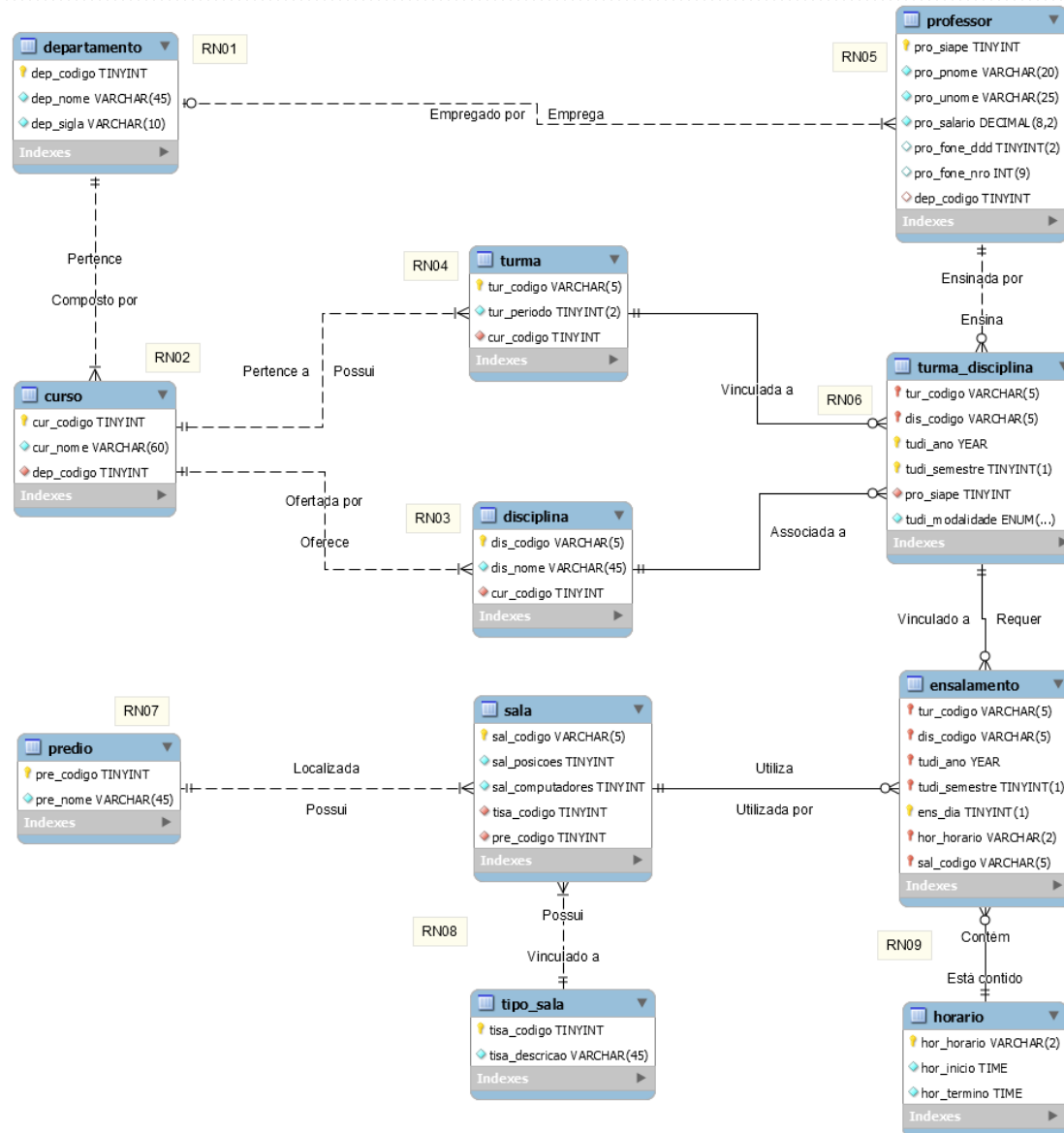
Prof. **GIOVANI** Volnei Meinerz

Aula 18 – SQL (cont.)

Objetivos da Aula

→ Entender a estrutura de operações básicas adicionais de **consulta** em SQL e sua aplicação

Cenário



A operação de renomeação

➤ Vimos que:

- O resultado de uma consulta é uma tabela/relação resultante
- Nela, os nomes dos atributos no cabeçalho são derivados dos nomes contidos na(s) tabela(s) listada(s) na cláusula **FROM**

```
1 • SELECT
2      *
3 FROM
4      reservas.professor;
```

pro_siape	pro_pnome	pro_unome	pro_salario	pro_fone_ddd	pro_fone_nro	dep_codigo
1	Hanka	Ruebbert	60000.00	43	35239000	6
2	Tilo	Gerhold	65000.00	NULL	NULL	2
3	Ekkehart	Schubbert	55000.00	11	754210000	3
4	Gerhard	Huettio	85000.00	14	33448888	2
5	Anoela	Lehmann	75000.00	43	35237777	2
6	Lisa	Reimann	95000.00	NULL	NULL	1
7	Corinna	Enoellmann	130000.00	55	995554500	1
8	Manfred	Schubbert	79000.00	43	998456587	2
9	Lena	Reimann	145000.00	14	997465544	3
10	Giovani	Meinerz	84000.00	55	999838457	1
11	Luiz	Marenco	67000.00	55	996814596	NULL
12	Mano	Lima	81000.00	55	975824684	NULL
13	Cenair	Maicá	89000.00	55	997896341	NULL
NULL	NULL	NULL	NULL	NULL	NULL	NULL

A operação de renomeação (cont.)

- ➔ Quando o **SELECT** contem uma operação aritmética, o próprio texto da operação é utilizado para identificar o atributo computado na tabela resultante

```
9 • SELECT
10     pro_pnome,
11     pro_unome,
12     pro_salario * 1.1
13 FROM
14     professor;
```

pro_pnome	pro_unome	pro_salario * 1.1
Hanka	Ruebbert	66000.000
Tilo	Gerhold	71500.000
Ekkehart	Schubbert	60500.000
Gerhard	Huettig	93500.000
Anoela	Lehmann	82500.000
Lisa	Reimann	104500.000
Corinna	Enoellmann	143000.000
Manfred	Schubbert	86900.000
Lena	Reimann	159500.000
Giovani	Meinerz	92400.000
Luiz	Marenco	73700.000
Mano	Lima	89100.000
Cenair	Maicá	97900.000

A operação de renomeação (cont.)

- ➔ Para facilitar, a SQL permite o uso de um *alias* para renomear temporariamente os atributos de uma tabela resultante, bem como renomear também o nome da própria tabela, por meio da cláusula **AS**

➔ Sintaxe SQL

```
SELECT A1 AS alias [, A2 AS alias, ... , An AS alias]  
FROM T1 AS alias [, T2 AS alias, ... , Tn AS alias]  
[WHERE P]
```

A operação de renomeação (cont.)

- ➔ Reescrevendo a expressão anterior que contem uma operação aritmética

```
1 • SELECT
2     pro_pnome AS Nome,
3     pro_unome AS Sobrenome,
4     pro_salario * 1.1 AS Salário_Reajustado
5 FROM
6     professor;
```

Nome	Sobrenome	Salário_Reajustado
Hanka	Ruebbert	66000.000
Tilo	Gerhold	71500.000
Ekkehart	Schubbert	60500.000
Gerhard	Huettig	93500.000
Angela	Lehmann	82500.000
Lisa	Reimann	104500.000
Corinna	Endellmann	143000.000
Manfred	Schubbert	86900.000
Lena	Reimann	159500.000
Giovani	Meinerz	92400.000
Luiz	Marenco	73700.000
Mano	Lima	89100.000
Cenair	Maicá	97900.000

A operação de renomeação (cont.)

- Se o *alias* contem espaços, o nome deve ser colocado entre aspas simples

```
1 • SELECT
2     pro_pnome AS Nome,
3     pro_unome AS Sobrenome,
4     pro_salario * 1.1 AS 'Salário Reajustado'
5 FROM
6     professor;
```

Nome	Sobrenome	Salário Reajustado
Hanka	Ruebbert	66000.000
Tilo	Gerhold	71500.000
Ekkehart	Schubbert	60500.000
Gerhard	Huettig	93500.000
Anoela	Lehmann	82500.000
Lisa	Reimann	104500.000
Corinna	Endellmann	143000.000
Manfred	Schubbert	86900.000
Lena	Reimann	159500.000
Giovani	Meinerz	92400.000
Luiz	Marengo	73700.000
Mano	Lima	89100.000
Cenair	Maicá	97900.000

A operação de renomeação (cont.)

- ➔ Além do **SELECT**, a cláusula **AS** também pode ser usada com a cláusula **FROM** para renomear tabelas

```
15 -- Sem ALIAS
16 -- Encontrar as turmas vinculadas a disciplinas, ofertadas no segundo semestre de 2017.
17 -- Mostrar o código da turma, o código da disciplina, seu nome, ano e semestre em que foi ofertada
18 • SELECT
19     turma_disciplina.tur_codigo,
20     turma_disciplina.dis_codigo,
21     dis_nome,
22     tudi_ano,
23     tudi_semestre
24 FROM
25     turma,
26     turma_disciplina,
27     disciplina
28 WHERE
29     turma_disciplina.tur_codigo = turma.tur_codigo
30     AND turma_disciplina.dis_codigo = disciplina.dis_codigo
31     AND tudi_ano = 2017
32     AND tudi_semestre = 2;
```

tur_codigo	dis_codigo	dis_nome	tudi_ano	tudi_semestre
C41	EC34D	Banco De Dados 1	2017	2
C51	EC35B	Banco de Dados 2	2017	2
E21	MA35B	Probabilidade Estatística	2017	2
ES21	IF62H	Banco De Dados 1	2017	2
M31	EM33H	Física Experimental	2017	2
N12A	AN32C	Banco De Dados 1	2017	2
N12B	AN32C	Banco De Dados 1	2017	2
N12SP	AN32C	Banco De Dados 1	2017	2

A operação de renomeação (cont.)

- ➔ Além do **SELECT**, a cláusula **AS** também pode ser usada com a cláusula **FROM** para renomear tabelas (cont.)

```
34 -- Com ALIAS
35 -- Encontrar as turmas vinculadas a disciplinas, ofertadas no segundo semestre de 2017.
36 -- Mostrar o código da turma, o código da disciplina, seu nome, ano e semestre em que foi ofertada
37 • SELECT
38     td.tur_codigo AS Turma,
39     td.dis_codigo AS 'Código da Disciplina',
40     dis_nome AS 'Nome da Disciplina',
41     tudi_ano AS Ano,
42     tudi_semestre AS Semestre
43 FROM
44     turma AS t,
45     turma_disciplina AS td,
46     disciplina AS d
47 WHERE
48     td.tur_codigo = t.tur_codigo
49     AND td.dis_codigo = d.dis_codigo
50     AND tudi_ano = 2017
51     AND tudi_semestre = 2;
```

Turma	Código da Disciplina	Nome da Disciplina	Ano	Semestre
C41	EC34D	Banco De Dados 1	2017	2
C51	EC35B	Banco de Dados 2	2017	2
E21	MA35B	Probabilidade Estatística	2017	2
ES21	IF62H	Banco De Dados 1	2017	2
M31	EM33H	Física Experimental	2017	2
N12A	AN32C	Banco De Dados 1	2017	2
N12B	AN32C	Banco De Dados 1	2017	2
N12SP	AN32C	Banco De Dados 1	2017	2

Operações de *string*

- Uma *string* é um conjunto de caracteres
- A comparação de *strings* com os operadores relacionais considera sempre a totalidade da *string*

```
1  --
2  -- Operadores relacionais: totalidade da string
3  --
4  • SELECT *
5    FROM professor
6    WHERE pro_pnome = 'Hanka';
```

<

Result Grid | Filter Rows: | Edit: | Export/Import:

	pro_siape	pro_pnome	pro_unome	pro_salario	pro_fone_ddd	pro_fone_nro	dep_codigo
	1	Hanka	Ruebbert	60000.00	43	35239000	6
	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- E se precisarmos comparar partes de uma *string*?

Operações de *string* (cont.)

→ O operador **LIKE**

- Permite encontrar padrões em atributos de *strings*
- Permite a utilização de dois caracteres especiais (curingas)

Caracter	Significado
%	Realiza substituição de zero, um ou mais caracteres, seguintes ou precedentes, de uma <i>string</i>
—	Realiza a substituição de um único caracter

→ LIKE é usado na cláusula WHERE

→ Sintaxe SQL

WHERE expressão **[NOT] LIKE** padrão

Operações de *string* (cont.)

➤ Problema

- Selecione as disciplinas cujo nome inicie com a substring “B”. Mostre código e nome.

```
5 -- Códigos e nomes de disciplinas cujo nome inicie com "B"
6 • SELECT
7     dis_codigo, dis_nome
8 FROM
9     disciplina
10 WHERE
11     dis_nome LIKE 'B%';
```

dis_codigo	dis_nome
AN32C	Banco De Dados 1
EC34D	Banco De Dados 1
EC35B	Banco de Dados 2
ES65B	Banco De Dados 2
IF53C	Banco De Dados 2
IF62H	Banco De Dados 1
NULL	NULL

```
1 -- Todas as disciplinas
2 • SELECT dis_codigo, dis_nome
3 FROM disciplina;
```

dis_codigo	dis_nome
AN32C	Banco De Dados 1
EC34D	Banco De Dados 1
EC35B	Banco de Dados 2
EM33H	Física Experimental
ES31A	Introdução A Engenharia De Software
ES31B	Matemática Discreta
ES31C	Laboratorio De Informática
ES31D	Algoritmos
ES61G	Lógica para Computação
ES63H	Redes de Computadores
ES65B	Banco De Dados 2
IF51A	Laboratorio De Informática
IF51C	Organização De Computadores
IF53C	Banco De Dados 2
IF61A	Introdução A Engenharia De Computação
IF62A	Técnicas De Programação
IF62C	Técnicas De Programação
IF62D	Processo De Produção De Software
IF62H	Banco De Dados 1
IF63C	Requisitos de Software
IF64G	Qualidade de Software
IF67K	Arquitetura de Software
IF68F	Programação Distribuída
MA35B	Probabilidade Estatística
MA62G	Probabilidade Estatística
SO32A	Organização De Computadores
SO36B	Gerência de Configuração
NULL	NULL

Operações de *string* (cont.)

➤ Problema

- Selecione os departamentos que terminem com a substring “ção”. Mostre o nome do departamento.

```
1  -- Nomes de departamentos que terminem com "ção"
2  • SELECT
3      dep_nome
4  FROM
5      departamento
6  WHERE
7      dep_nome LIKE '%ção';
```

	dep_nome
	Departamento Acadêmico de Computação

```
1  • SELECT *
2  FROM departamento;
```

dep_codigo	dep_nome	dep_sigla
1	Departamento Acadêmico de Computação	DACOM
2	Departamento Acadêmico de Elétrica	DAELE
3	Departamento Acadêmico de Matemática	DAMAT
4	Departamento Acadêmico de Eletrônica	DAELN
5	Departamento Acadêmico de Eletrotécnica	DAELT
6	Departamento Acadêmico de Mecânica	DAMEC
7	Departamento Acadêmico de Física	DAFIS
8	Departamento Acadêmico de Gestão e Economia	DAGEE
NULL	NULL	NULL

Operações de *string* (cont.)

➤ Problema

- Selecione os professores cujos nomes contenham a substring “har”. Mostre os nomes dos professores.

```
1  -- Professores cujos nomes contenham a substring "har"
2  • SELECT
3      pro_pnome, pro_unome
4  FROM
5      professor
6  WHERE
7      pro_pnome LIKE '%har%';
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
pro_pnome	pro_unome			
Ekkehart	Schubbert			
Gerhard	Huetitia			

Operações de *string* (cont.)

➤ Problema

- Encontre os professores cujos nomes **não** contenham a substring “har”. Mostre os nomes dos professores.

```
1  -- Professores cujos nomes NÃO contenham a substring "har"
2  • SELECT pro_pnome, pro_unome
3  FROM professor
4  WHERE pro_pnome NOT LIKE '%har%';
5
6  -- ou
7
8  • SELECT pro_pnome, pro_unome
9  FROM professor
10 WHERE NOT (pro_pnome LIKE '%har%');
```

<

Result Grid | Filter Rows: | Export: | Wrap Cell Content:




pro_pnome	pro_unome
Hanka	Ruebbert
Tilo	Gerhold
Angela	Lehmann
Lisa	Reimann
Corinna	Enoellmann
Manfred	Schubbert
Lena	Reimann
Giovani	Meinerz
Luiz	Marengo
Mano	Lima
Cenair	Maicá

Operações de *string* (cont.)

➤ Problema

- Selecione os professores cujo salário tenha "0" e "0" no segundo e terceiro dígitos, respectivamente. Mostre nome e salário.

```
1  -- "0" e "0" no segundo e terceiro dígito, respectivamente
2  • SELECT
3      pro_pnome, pro_unome, pro_salario
4  FROM
5      professor
6  WHERE
7      pro_salario LIKE '_00%';
8
```

< Result Grid |  Filter Rows: | Export:  | Wrap Cell Content: 




	pro_pnome	pro_unome	pro_salario
	Hanka	Ruebbert	60000.00

Operações de *string* (cont.)

➤ Problema

- Selecione os professores cujo penúltimo e antepenúltimo caractere seja "r" e "a", respectivamente. Mostre os nomes.

```
1  -- Professores cujo penúltimo e antepenúltimo
2  -- caractere seja "r" e "a", respectivamente
3  • SELECT
4      pro_pnome, pro_unome
5  FROM
6      professor
7  WHERE
8      pro_pnome LIKE '%ar_';
```

< Result Grid |  Filter Rows: | Export:  | Wrap Cell Content: 







	pro_pnome	pro_unome
	Ekkehart	Schubbert
	Gerhard	Huetitia

Operações de *string* (cont.)

➤ Problema

- Selecione as disciplinas cujo terceiro caracter é igual a "o" e o último igual a "a". Mostre código e nome da disciplina.

```
1  -- Disciplinas cujo terceiro caracter igual "o" e último igual "a"
2  • SELECT
3      dis_codigo, dis_nome
4  FROM
5      disciplina
6  WHERE
7      dis_nome LIKE '__o%a';
```

< Result Grid |  Filter Rows: | Edit:    | Export/Import:   | Wrap Cell Conte

dis_codigo	dis_nome
IF68F	Programação Distribuída
MA35B	Probabilidade Estatística
MA62G	Probabilidade Estatística
NULL	NULL

Especificação de atributo na cláusula SELECT

- O símbolo de asterisco “*” pode ser usado na cláusula **SELECT** para indicar que devem ser mostrados todos os atributos das tabelas listadas na cláusula **FROM**.

```
1 • SELECT
2      *
3 FROM
4      curso c,
5      departamento d
6 WHERE
7      c.dep_codigo = d.dep_codigo;
```

<

Result Grid | | Filter Rows: | Export: | Wrap Cell Content:



	cur_codigo	cur_nome	dep_codigo	dep_codigo	dep_nome	dep_sigla
1		Engenharia de Computação	1	1	Departamento Acadêmico de Computação	DACOM
2		Engenharia Elétrica	2	2	Departamento Acadêmico de Elétrica	DAELE
3		Engenharia Mecânica	6	6	Departamento Acadêmico de Mecânica	DAMEC
4		Engenharia de Software	1	1	Departamento Acadêmico de Computação	DACOM
5		Tecnologia em Análise e Desenvolvimento de Sistemas	1	1	Departamento Acadêmico de Computação	DACOM

Especificação de atributo na cláusula SELECT (cont.)

- Listando os atributos de apenas uma das tabelas listadas na cláusula **FROM**

```
1 • SELECT
2     c.*
3 FROM
4     curso c,
5     departamento d
6 WHERE
7     c.dep_codigo = d.dep_codigo;
```

<

Result Grid |  Filter Rows: | Exports:  | Wrap Cell Conts




	cur_codigo	cur_nome	dep_codigo
	1	Engenharia de Computação	1
	2	Engenharia Elétrica	2
	3	Engenharia Mecânica	6
	4	Engenharia de Software	1
	5	Tecnologia em Análise e Desenvolvimento de Sistemas	1

Especificação de atributo na cláusula SELECT (cont.)

- Listando todos os atributos de uma das tabelas, mais algum outro atributo de outra tabela listada na cláusula **FROM**

```
1 • SELECT
2     c.*, dep_sigla
3 FROM
4     curso c,
5     departamento d
6 WHERE
7     c.dep_codigo = d.dep_codigo;
```

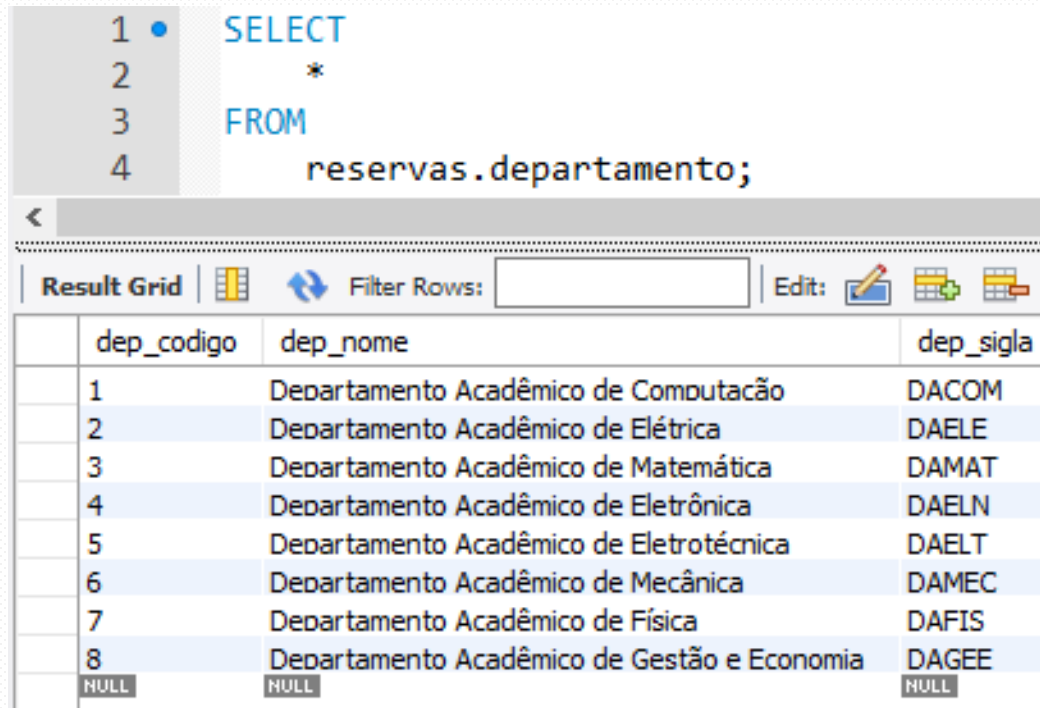
<

Result Grid |  Filter Rows: | Export:  Wrap Cell Content: 

	cur_codigo	cur_nome	dep_codigo	dep_sigla
	1	Engenharia de Computação	1	DACOM
	2	Engenharia Elétrica	2	DAELE
	3	Engenharia Mecânica	6	DAMEC
	4	Engenharia de Software	1	DACOM
	5	Tecnologia em Análise e Desenvolvimento de Sistemas	1	DACOM

Ordem de Classificação

- ➔ Os resultados gerados por um **SELECT** podem ser ordenados
- ➔ Quando nenhum ordenamento é especificado, o SGBD ordena o resultado pelo atributo que compõe a chave primária



The screenshot shows a database query interface. At the top, a query editor displays the following SQL statement:

```
1 • SELECT
2      *
3 FROM
4      reservas.departamento;
```

Below the query editor, there is a toolbar with icons for 'Result Grid', 'Filter Rows', and 'Edit'. The 'Result Grid' is active, showing a table with the following data:

	dep_codigo	dep_nome	dep_sigla
	1	Departamento Acadêmico de Computação	DACOM
	2	Departamento Acadêmico de Elétrica	DAELE
	3	Departamento Acadêmico de Matemática	DAMAT
	4	Departamento Acadêmico de Eletrônica	DAELN
	5	Departamento Acadêmico de Eletrotécnica	DAELT
	6	Departamento Acadêmico de Mecânica	DAMEC
	7	Departamento Acadêmico de Física	DAFIS
	8	Departamento Acadêmico de Gestão e Economia	DAGEE
	NULL	NULL	NULL

Ordem de Classificação (cont.)

- Quando o ordenamento é necessário, a SQL permite utilizar a cláusula **ORDER BY**
- Tipos de ordenação
 - **ASC** – indica que a ordenação será ascendente
 - **DESC** – indica que a ordenação será descendente
- Sintaxe SQL

```
SELECT A1 [, A2, ... , An]  
FROM T1 [, T2, ... , Tn]  
[WHERE P]  
[ORDER BY A1 [ASC | DESC], A2 [ASC | DESC], ... ]
```

- Quando o tipo de ordenação for omitido, a ordem-padrão será ascendente

Ordem de Classificação (cont.)

➤ Problema

- Ordenar departamentos ascendentemente. Mostre todas as colunas.

```
1  -- Ordenar departamentos ascendentemente
2  ● SELECT *
3  FROM departamento
4  ORDER BY dep_nome ASC;
5  -- ou
6  ● SELECT *
7  FROM departamento
8  ORDER BY dep_nome;
```

	dep_codigo	dep_nome	dep_sigla
	1	Departamento Acadêmico de Computação	DACOM
	2	Departamento Acadêmico de Elétrica	DAELE
	4	Departamento Acadêmico de Eletrônica	DAELN
	5	Departamento Acadêmico de Eletrotécnica	DAELT
	7	Departamento Acadêmico de Física	DAFIS
	8	Departamento Acadêmico de Gestão e Economia	DAGEE
	3	Departamento Acadêmico de Matemática	DAMAT
	6	Departamento Acadêmico de Mecânica	DAMEC
	NULL	NULL	NULL




Ordem de Classificação (cont.)

➤ Problema

- Ordenar professores por salário, do maior para o menor. Mostre nome salário.

```
1 -- Ordenar professores por salário, do maior para o menor.  
2 • SELECT pro_pnome, pro_unome, pro_salario  
3 FROM professor  
4 ORDER BY pro_salario DESC;
```

<

Result Grid |  Filter Rows: | Export:  | Wrap Cell Content: 

	pro_pnome	pro_unome	pro_salario
	Lena	Reimann	145000.00
	Corinna	Endellmann	130000.00
	Lisa	Reimann	95000.00
	Cenair	Maicá	89000.00
	Gerhard	Huettig	85000.00
	Giovani	Meinerz	84000.00
	Mano	Lima	81000.00
	Manfred	Schubbert	79000.00
	Anoela	Lehmann	75000.00
	Luiz	Marenco	67000.00
	Tilo	Gerhold	65000.00
	Hanka	Ruebbert	60000.00
	Ekkehart	Schubbert	55000.00

Ordem de Classificação (cont.)

➤ Problema

- Ordenar professores por região de telefonia (DDD), do menor para o maior. Mostre siape, nome e telefone.

```
1 -- Ordenar professores por DDD, do menor para o maior
2 • SELECT pro_siape, pro_pnome, pro_unome, pro_fone_ddd, pro_fone_nro
3 FROM professor
4 ORDER BY pro_fone_ddd;
```

pro_siape	pro_pnome	pro_unome	pro_fone_ddd	pro_fone_nro
2	Tilo	Gerhold	NULL	NULL
6	Lisa	Reimann	NULL	NULL
3	Ekkehart	Schubbert	11	754210000
4	Gerhard	Huettig	14	33448888
9	Lena	Reimann	14	997465544
1	Hanka	Ruebbert	43	35239000
5	Anoela	Lehmann	43	35237777
8	Manfred	Schubbert	43	998456587
7	Corinna	Enoellmann	55	995554500
10	Giovani	Meinerz	55	999838457
11	Luiz	Marengo	55	996014596
12	Mano	Lima	55	975824684
13	Cenair	Maicá	55	997896341
NULL	NULL	NULL	NULL	NULL

- Por vezes podem existir atributos com valores **repetidos**
- Pode ser necessária a ordenação do resultado, recorrendo a mais de um atributo

Ordem de Classificação (cont.)

➤ Problema

- Ordenar professores por DDD (de forma ascendente) e, no seu interior, pelo nome. Mostre siape, nome e telefone.

```
6 -- Ordenar professores por DDD (de forma ascendente) e,  
7 -- no interior de DDD, por nome  
8 • SELECT pro_siape, pro_pnome, pro_unome, pro_fone_ddd, pro_fone_nro  
9 FROM professor  
10 ORDER BY pro_fone_ddd ASC, pro_pnome ASC;
```

pro_siape	pro_pnome	pro_unome	pro_fone_ddd	pro_fone_nro
6	Lisa	Reimann	NULL	NULL
2	Tilo	Gerhold	NULL	NULL
3	Ekkehart	Schubbert	11	754210000
4	Gerhard	Huettig	14	33448888
9	Lena	Reimann	14	997465544
5	Angela	Lehmann	43	35237777
1	Hanka	Ruebbert	43	35239000
8	Manfred	Schubbert	43	998456587
13	Cenair	Maicá	55	997896341
7	Corinna	Engelmann	55	995554500
10	Giovani	Meinerz	55	999838457
11	Luiz	Marenco	55	996814596
12	Mano	Lima	55	975824684
NULL	NULL	NULL	NULL	NULL

Operadores da Cláusula WHERE

→ Operador de comparação **BETWEEN**

- Permite comparar uma expressão com intervalos de valores
- Sintaxe SQL

```
SELECT A1 [, A2, ... , An]  
FROM T1 [, T2, ... , Tn]  
[WHERE Expressão [NOT] BETWEEN Valor1 AND Valor2]
```

→ Onde

- **Expressão**, pode ser um atributo ou um cálculo
- **Valor₁** e **Valor₂**, representam o intervalo em relação ao qual a **Expressão** será comparada
- Tabela resultante conterá os registros cuja expressão comparada estiver compreendida entre os valores especificados



Operadores da Cláusula WHERE (cont.)

➤ Problema

- Selecionar os professores cujos salários estejam compreendidos entre R\$65000,00 e R\$80000,00. Mostre nome e salário.

```
1  -- Professores cujos salários estejam compreendidos
2  -- entre R$65000,00 e R$80000,00
3  SELECT
4      pro_pnome, pro_unome, pro_salario
5  FROM
6      professor
7  WHERE
8      pro_salario BETWEEN 65000 AND 80000;
```

<

Result Grid |  Filter Rows: | Export:  Wrap Cell Conte

	pro_pnome	pro_unome	pro_salario
	Tilo	Gerhold	65000.00
	Anaëla	Lehmann	75000.00
	Manfred	Schubbert	79000.00
	Luiz	Marengo	67000.00

Operadores da Cláusula WHERE (cont.)

➤ Problema

- Selecionar os professores cujos salários não estejam compreendidos entre R\$65000,00 e R\$80000,00. Mostre nome e salário.

```
1 • SELECT pro_pnome, pro_unome, pro_salario
2   FROM professor
3  WHERE pro_salario NOT BETWEEN 65000 AND 80000;
4   -- ou
5 • SELECT pro_pnome, pro_unome, pro_salario
6   FROM professor
7  WHERE NOT (pro_salario BETWEEN 65000 AND 80000);
```

pro_pnome	pro_unome	pro_salario
Hanka	Ruebbert	60000.00
Ekkehart	Schubbert	55000.00
Gerhard	Huettig	85000.00
Lisa	Reimann	95000.00
Corinna	Engellmann	130000.00
Lena	Reimann	145000.00
Giovani	Meinerz	84000.00
Mano	Lima	81000.00
Cenair	Maicá	89000.00

Tratamento de Valores Nulos

→ Operador **IS [NOT] NULL**

→ Utilizado para verificar se o valor de um atributo é (ou não) nulo

→ Sintaxe SQL

```
SELECT A1 [, A2, ... , An]  
FROM T1 [, T2, ... , Tn]  
[WHERE A1 IS [NOT] NULL
```





Tratamento Valores Nulos (cont.)

➤ Problema

- Encontre os professores sem telefone cadastrado. Mostre seus nomes.

```
1 -- Professores sem telefone cadastrado
2 SELECT
3     pro_pnome, pro_unome
4 FROM
5     professor
6 WHERE
7     (pro_fone_ddd and pro_fone_nro) IS NULL;
```

<

Result Grid |   Filter Rows: | Export:  | Wrap



	pro_pnome	pro_unome
	Tilo	Gerhold
	Lisa	Reimann

Tratamento Valores Nulos (cont.)

➤ Problema

- Encontre os professores com telefone cadastrado. Mostre seus nomes.

```
1  -- Professores sem telefone cadastrado
2  • SELECT
3      pro_pnome, pro_unome
4  FROM
5      professor
6  WHERE
7      (pro_fone_ddd AND pro_fone_nro) IS NOT NULL;
```

< Result Grid |  Filter Rows: | Export:  | Wrap Cell C

	pro_pnome	pro_unome
	Hanka	Ruebbert
	Ekkehart	Schubbert
	Gerhard	Huettia
	Anoela	Lehmann
	Corinna	Engellmann
	Manfred	Schubbert
	Lena	Reimann
	Giovani	Meinerz
	Luiz	Marenco
	Mano	Lima
	Cenair	Maicá

Tratamento Valores Nulos (cont.)

→ Problema

- Reescrever consulta envolvendo ordenação, agora, sem nulos

Ordem de Classificação (cont.)

→ Problema

- Ordenar professores por DDD (de forma ascendente) e, no seu interior, pelo nome. Mostre siape, nome e telefone.

```
6 -- Ordenar professores por DDD (de forma ascendente) e,  
7 -- no interior de DDD, por nome  
8 • SELECT pro_siape, pro_pnome, pro_unome, pro_fone_ddd, pro_fone_nro  
9 FROM professor  
10 ORDER BY pro_fone_ddd ASC, pro_pnome ASC;
```

pro_siape	pro_pnome	pro_unome	pro_fone_ddd	pro_fone_nro
6	Lisa	Reimann	NULL	NULL
2	Tilo	Gerhold	NULL	NULL
3	Ekkehart	Schubbert	11	754210000
4	Gerhard	Huettio	14	33448888
9	Lena	Reimann	14	997465544
5	Anoela	Lehmann	43	35237777
1	Hanka	Ruebbert	43	35239000
8	Manfred	Schubbert	43	998456587
13	Cenair	Maicá	55	997896341
7	Corinna	Endelmann	55	995554500
10	Giovani	Meinerz	55	999838457
11	Luiz	Marengo	55	996814596
12	Mano	Lima	55	975824684
NULL	NULL	NULL	NULL	NULL

28/41

Banco de Dados 1 – Prof. Giovani

Tratamento Valores Nulos (cont.)

➤ Problema

➤ Reescrever consulta envolvendo ordenação, agora, sem nulos

```
3 • SELECT pro_siape, pro_pnome, pro_unome, pro_fone_ddd, pro_fone_nro
4 FROM professor
5 WHERE (pro_fone_ddd AND pro_fone_nro) IS NOT NULL
6 ORDER BY pro_fone_ddd ASC, pro_pnome ASC;
```

pro_siape	pro_pnome	pro_unome	pro_fone_ddd	pro_fone_nro
3	Ekkehart	Schubbert	11	754210000
4	Gerhard	Huettig	14	33448888
9	Lena	Reimann	14	997465544
5	Angela	Lehmann	43	35237777
1	Hanka	Ruebbert	43	35239000
8	Manfred	Schubbert	43	998456587
13	Cenair	Maicá	55	997896341
7	Corinna	Endellmann	55	995554500
10	Giovani	Meinerz	55	999838457
11	Luiz	Marengo	55	996814596
12	Mano	Lima	55	975824684
NULL	NULL	NULL	NULL	NULL

Teste de Membros de Conjunto

→ Operador [NOT] IN

- Utilizado para verificar se o valor de um atributo coincide com qualquer valor de uma lista (está ou não, contido)

→ Sintaxe SQL

```
SELECT A1 [, A2, ... , An]  
FROM T1 [, T2, ... , Tn]  
[WHERE Expressão [NOT] IN (Valor1, Valor2, ... , Valorn)]
```

→ Onde

- **Expressão**, valor de atributo a ser testado
- **Valor₁, Valor₂, ..., Valor_n**, conjunto de valores em relação aos quais a **Expressão** será testada

Teste de Membros de Conjunto (cont.)

➔ Problema

- ➔ Encontrar as salas do tipo “Teórica”, “Pesquisa” e “Monitoria”.
Mostre código da sala, tipo da sala e nome do prédio.

```
1  -- Mostra as salas, com respectivo tipo (descrição)
2  -- e prédio onde estão situadas
3  • SELECT sal_codigo, tisa_descricao, pre_nome
4  FROM sala, tipo_sala, predio
5  WHERE sala.tisa_codigo = tipo_sala.tisa_codigo
6        AND sala.pre_codigo = predio.pre_codigo;
```

sal_codigo	tisa_descricao	pre_nome
I201	Laboratório	Bloco I
I202	Laboratório	Bloco I
I205	Laboratório	Bloco I
K005	Pesquisa	Bloco K
K008	Laboratório	Bloco K
K009	Laboratório	Bloco K
P003	Laboratório	Bloco P
P005	Laboratório	Bloco P
P101	Teórica	Bloco P
P105	Laboratório	Bloco P
P205	Laboratório	Bloco P
A040	Laboratório	Bloco A
A137	Monitoria	Bloco A
A140	Teórica	Bloco A
A146	Teórica	Bloco A

```
9  • SELECT sal_codigo, tisa_descricao, pre_nome
10 FROM sala, tipo_sala, predio
11 WHERE sala.tisa_codigo = tipo_sala.tisa_codigo
12        AND sala.pre_codigo = predio.pre_codigo
13        AND tisa_descricao IN ('Teórica', 'Pesquisa', 'Monitoria');
```

sal_codigo	tisa_descricao	pre_nome
P101	Teórica	Bloco P
A140	Teórica	Bloco A
A146	Teórica	Bloco A
K005	Pesquisa	Bloco K
A137	Monitoria	Bloco A

Teste de Membros de Conjunto (cont.)

→ Problema

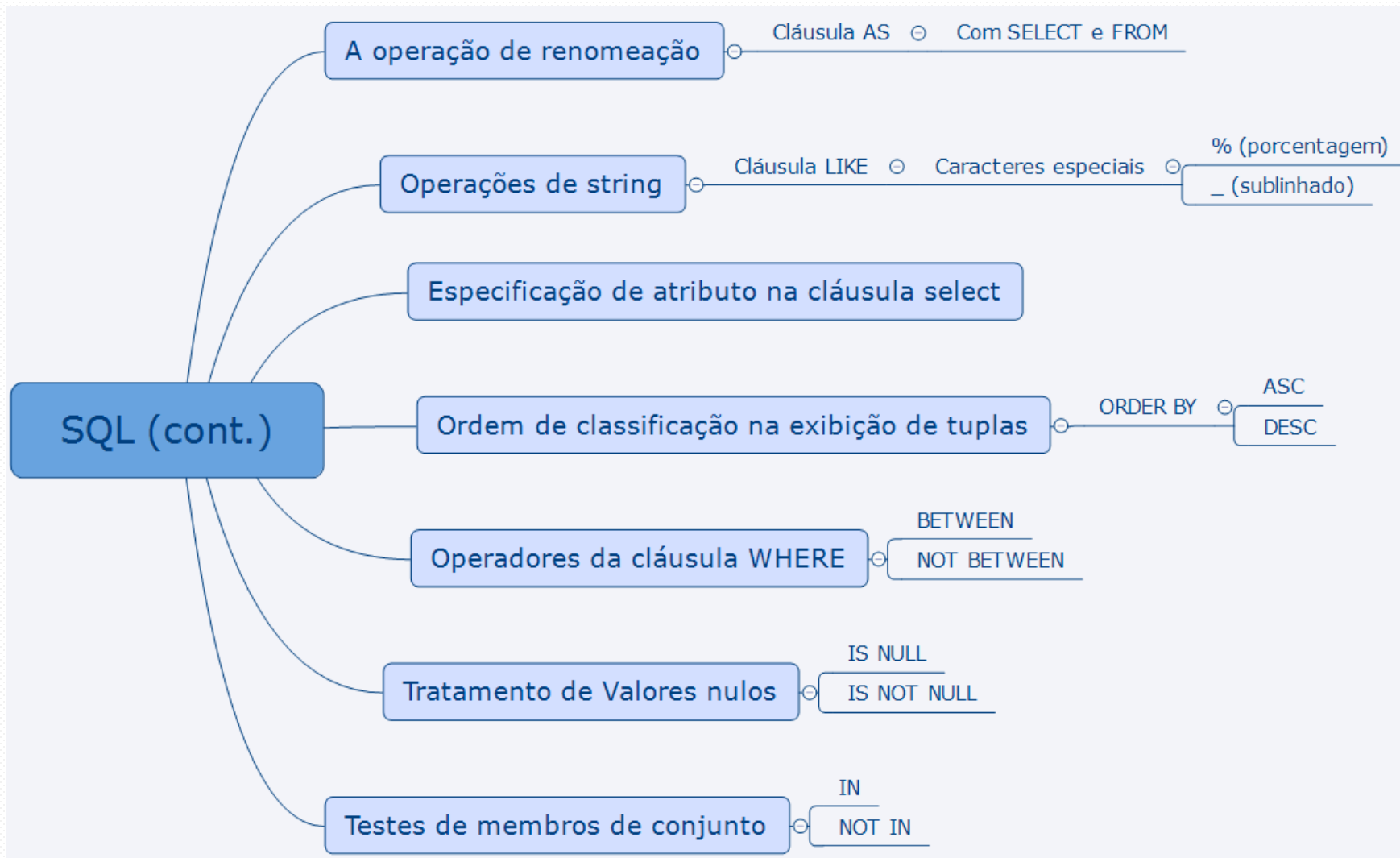
- Encontrar as salas diferentes do tipo “Teórica”, “Pesquisa” e “Monitoria”. Mostre código da sala, tipo da sala e nome do prédio.

```
3 • SELECT sal_codigo, tisa_descricao, pre_nome
4 FROM sala, tipo_sala, predio
5 WHERE sala.tisa_codigo = tipo_sala.tisa_codigo
6       AND sala.pre_codigo = predio.pre_codigo
7       AND tisa_descricao NOT IN ('Teórica', 'Pesquisa', 'Monitoria');
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

sal_codigo	tisa_descricao	pre_nome
I201	Laboratório	Bloco I
I202	Laboratório	Bloco I
I205	Laboratório	Bloco I
K008	Laboratório	Bloco K
K009	Laboratório	Bloco K
P003	Laboratório	Bloco P
P005	Laboratório	Bloco P
P105	Laboratório	Bloco P
P205	Laboratório	Bloco P
A040	Laboratório	Bloco A

Resumo da Aula





DISCIPLINA: Banco de Dados 1

Prof. **GIOVANI** Volnei Meinerz

Aula 18 – SQL (cont.)