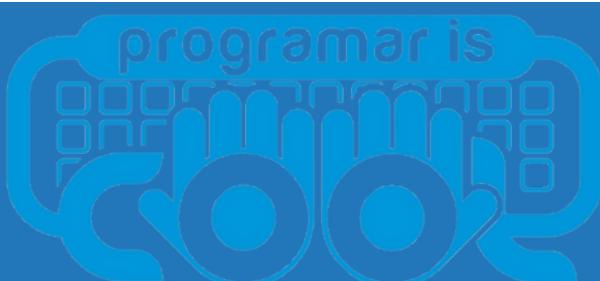
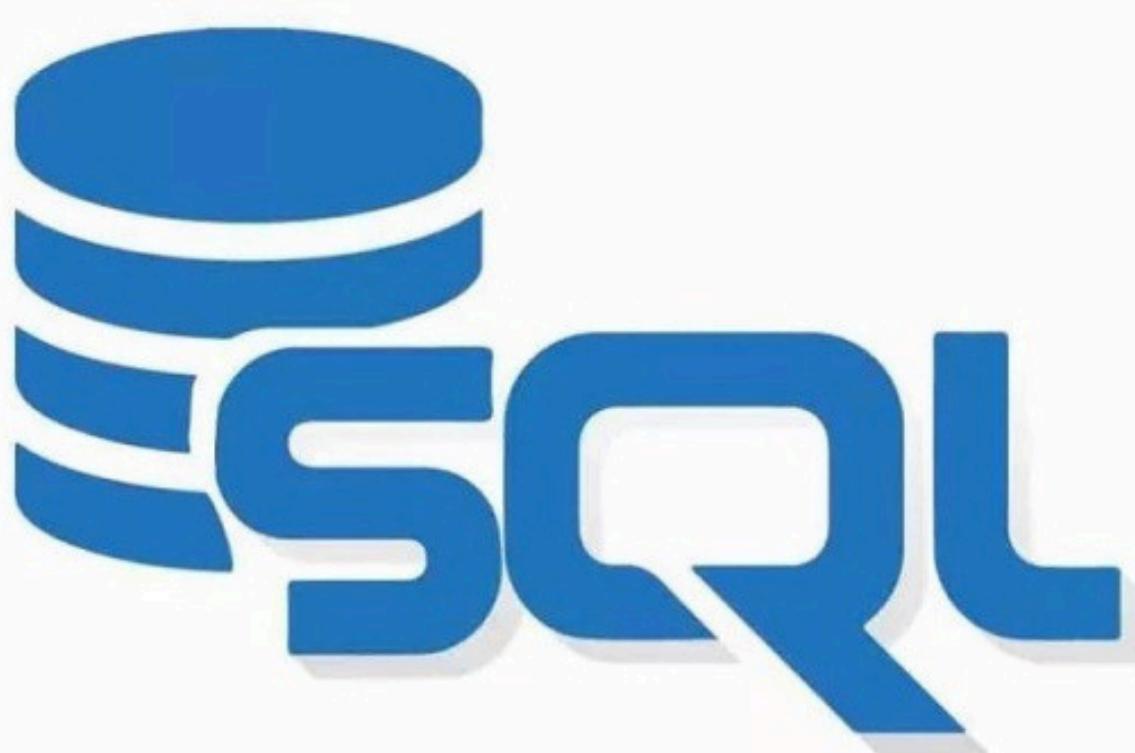
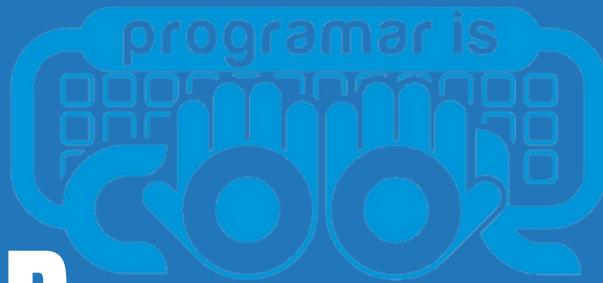


APRENDA 45 FUNÇÕES SQL PARA SE TORNAR UM USUÁRIO AVANÇADO DE BANCO DE DADOS





INTRODUÇÃO

Apresentação do Canal

Seja bem vindo ao **ProgramarIsCool**, o canal do **YouTube** com a missão de despertar nas pessoas a vontade de aprender computação, pois acreditamos que só é possível aprender aquilo que nos interessa.

Abraço!

Marcio Victorino

ProgramarIsCool

<https://www.youtube.com/ProgramarIsCool>

Marcio Victorino





INTRODUÇÃO

SOBRE O AUTOR

Marcio Victorino

É engenheiro da computação formado pelo Instituto Militar de Engenharia (IME), possui Especialização em Sistemas de Informação e Telemática pela Universidade Federal do Rio Grande do Sul (UFRGS), Mestrado em Sistemas e Computação pelo IME e Doutorado em Ciência da Informação pela Universidade de Brasília (UnB).

Tornou-se profissional de TI em 1994 quando concluiu o curso de graduação em Engenharia da Computação. Desde então, participou de vários projetos de TI do Exército Brasileiro (EB), desempenhando diversos papéis, iniciando como desenvolvedor de software até assumir a posição de gerente de projetos.

Em paralelo às atividades de engenharia, desenvolveu a carreira acadêmica por acreditar que a experiência adquirida no desenvolvimento de projetos de TI seria de grande valia para a formação de futuros profissionais da área.

Em 2002, começou a ministrar disciplinas de TI em instituições de ensino superior do Distrito Federal. Em 2014, quando passou para a reserva do EB, ingressou como professor substituto do Departamento de Ciência da Computação (CIC) e, em 2017, tornou-se professor adjunto da Faculdade de Ciência da Informação (FCI) da UnB. Atualmente, é professor da Graduação em Biblioteconomia da FCI e participa de dois programas de Pós-Graduação da UnB nas seguintes linhas de pesquisa: Gestão, Tecnologias e Organização da Informação e do Conhecimento do PPGCInf (FCI) e Ciência de Dados do PPCA (CIC).

ProgramarIsCool

<https://www.youtube.com/ProgramarIsCool>

Marcio Victorino





INTRODUÇÃO

PARA QUEM É ESTE E-BOOK?

Se você já é um usuário de banco de dados relacional e consulta constantemente os dados organizados em tabelas utilizando a linguagem SQL, esse e-book vai lhe ajudar a ser mais produtivo no momento de montar as suas consultas.

Por outro lado, caso você ainda não esteja familiarizado com banco de dados relacional ou com a linguagem SQL, mas tem interesse no assunto, sugiro, antes de ler este e-book, acessar os cursos em formato de playlists disponibilizados gratuitamente no canal do [YouTube ProgramarIsCool](#). Assim, você terá um melhor aproveitamento ao ler este conteúdo.

Bom estudo!

Marcio Victorino

ProgramarIsCool

<https://www.youtube.com/ProgramarIsCool>

Marcio Victorino





INTRODUÇÃO

QUAIS PLAYLISTS DEVO ASSISTIR ANTES DE LER ESTE E-BOOK?

Caso você ainda não esteja familiarizado com banco de dados relacional ou com a linguagem SQL, sugiro acessar as seguintes playlists disponibilizados gratuitamente no canal do **YouTube ProgramarIsCool**:

1) Curso Modelagem de Dados Conceitual:

<https://www.youtube.com/playlist?list=PLdoTFRH60cIASgUnYIQUtqAQsUg1dIKGQ>

2) Curso Projeto Lógico de Banco de Dados:

https://www.youtube.com/playlist?list=PLdoTFRH60cIB_DsBnmg-M4B0MxO7Jz-e4

3) Curso Normalização de Modelos:

https://www.youtube.com/playlist?list=PLdoTFRH60cIB7Eqj9EmydOr_WUNzYNs6U

4) Curso Projeto Físico de Banco de Dados:

https://www.youtube.com/playlist?list=PLdoTFRH60cIB_RIG2O1zr8RjjpkLbeNGfc

5) Curso Fundamentos do SGBDR MySQL:

https://www.youtube.com/playlist?list=PLdoTFRH60cI_DpkdoMp-Yv8hqNwlnrJg_s

6) Curso Linguagem SQL: Comandos DDL:

https://www.youtube.com/playlist?list=PLdoTFRH60cI_CkQW9bjOihIHX_3xQ10X1P

ProgramarIsCool

<https://www.youtube.com/ProgramarIsCool>

Marcio Victorino





INTRODUÇÃO

QUAIS PLAYLISTS DEVO ASSISTIR ANTES DE LER ESTE E-BOOK?

7) Curso Linguagem SQL: Comandos DML:

https://www.youtube.com/playlist?list=PLdoTFRH60cI_DE1N-rlcYsUXuTM0n1bE7E

8) Curso Linguagem SQL: Operadores de Conjuntos:

https://www.youtube.com/playlist?list=PLdoTFRH60cI_C92kuPomiz0sE2HnKR-7rE

9) Curso Linguagem SQL: Funções:

https://www.youtube.com/playlist?list=PLdoTFRH60cI_AWffYv9kqeDGicSc_ItYZE

10) Curso Linguagem SQL: Visões:

https://www.youtube.com/playlist?list=PLdoTFRH60cI_Dn0nN9ur-WGcizhK2Z8C-p

11) Linguagem SQL: Exercícios:

https://www.youtube.com/playlist?list=PLdoTFRH60cI_Agp2Eub3IHuvNAmMfgB_kT

12) Curso Álgebra Relacional:

https://www.youtube.com/playlist?list=PLdoTFRH60cI_DTo_-mZvnuwg_uJeMbu6ZP

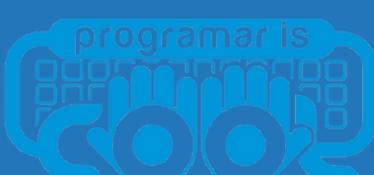
Bom estudo!

Marcio Victorino

ProgramarIsCool

<https://www.youtube.com/ProgramarIsCool>

Marcio Victorino





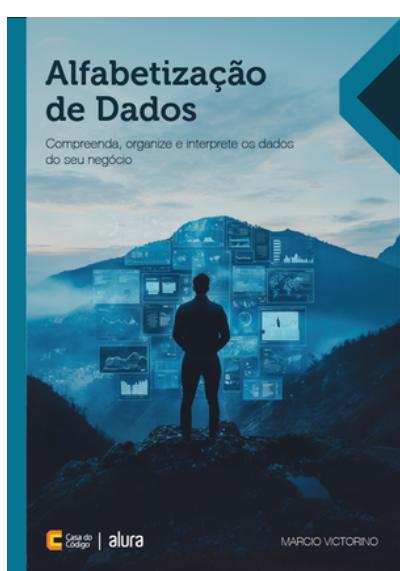
INTRODUÇÃO

Livro Alfabetização de Dados

Gostaria de apresentar o meu livro **Alfabetização de Dados: compreenda, organize e interprete os dados do seu negócio.**

Em um mundo no qual as organizações são ou estão tentando se tornar orientadas a dados (data driven), a diferença entre o bom gestor e o não tão bom gestor é que o primeiro tende a resolver os problemas do negócio tomando decisões baseadas em dados.

Com esse livro, você vai compreender, organizar e interpretar os dados do seu negócio para usá-los como subsídio para a tomada de decisão.



Conheça mais detalhes sobre o livro no link:
<https://programariscool.com.br/>

Marcio Victorino





INTRODUÇÃO

O QUE É SQL?

A SQL (Structured Query Language) ou Linguagem de Consulta Estruturada, é a linguagem declarativa padrão para banco de dados relacional.

Ela foi criada na década de 70 e vem sendo atualizada constantemente, sua última atualização foi realizada em 2016.

Apesar dessa linguagem conter a palavra “Consulta” em sua denominação, ela possui comandos com outros objetivos, como por exemplo, criar estruturas.

Os comandos da linguagem SQL podem ser organizados em várias categorias, as duas mais importantes são:

- **Linguagem de Definição de Dados (DDL)**: usada para criar estruturas em um banco de dados, como por exemplo, tabelas. **Linguagem de Definição de Dados**
- **Manipulação de Dados (DML)**: usada para manipular os dados organizados em tabelas em um banco de dados. As manipulações típicas são a recuperação, inserção, remoção e modificação dos dados.

Neste e-book, abordaremos apenas funções utilizadas em comandos DML da linguagem SQL.

programariscool@gmail.com

Marcio Victorino





INTRODUÇÃO

COMO USAR ESTE E-BOOK?

Este e-book tem o objetivo de ir direto ao assunto, neste caso apresentaremos consultas utilizando as funções mais importantes da linguagem SQL.

As funções foram organizadas em três categorias:

- **Funções de Caracteres:** utilizadas para manipular dados literais (strings).
- **Funções Numéricas:** utilizadas para manipular números.
- **Funções de Datas:** utilizadas para manipular dados que representam datas ou tempo.

Inicialmente apresentaremos uma tabela com dados, descreveremos em poucas palavras o objetivo de cada função e, em seguida, executaremos uma consulta SQL focando a tabela e apresentaremos o resultado para que o leitor entenda facilmente o uso da função.

Todas as consultas apresentadas neste e-book foram executadas no SGBD MySQL. Apesar da SQL ser uma linguagem padronizada por organismos internacionais (American National Standards Institute - ANSI), algumas funções podem apresentar diferenças em SGBDs específicos. Então, pode acontecer de uma função apresentada não funcionar da mesma forma em outro SGBD.

Esperamos que seu aprendizado seja útil e divertido.

Boa leitura!

programariscool@gmail.com

Marcio Victorino



FUNÇÕES DE CARACTERES

1 - LOWER

Tabela Departamento

Codigo	Nome	UF
10	RH	RJ
20	TI	RJ
30	Logística	DF
40	Financeiro	DF
50	Venda	RS
60	Pesquisa	RJ

Função LOWER(coluna): retorna todos os caracteres da resposta à consulta em letras minúsculas.

Ex: `SELECT LOWER(Nome)`
`FROM Departamento;`

Resultado

LOWER(Nome)
rh
ti
logística
financeiro
venda
pesquisa

programariscool@gmail.com

Marcio Victorino



2 - UPPER

Tabela Departamento

Codigo	Nome	UF
10	RH	RJ
20	TI	RJ
30	Logística	DF
40	Financeiro	DF
50	Venda	RS
60	Pesquisa	RJ

Função UPPER(coluna): retorna todos os caracteres da resposta à consulta em letras maiúsculas.

Ex: `SELECT UPPER(Nome)`
`FROM Departamento;`

Resultado

UPPER(Nome)
RH
TI
LOGÍSTICA
FINANCEIRO
VENDA
PESQUISA

programariscool@gmail.com

Marcio Victorino



3 - CONCAT

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario
123	Igor	Pereira	100
159	Denise	Moreno	440
369	Marcelo	Neiva	900
456	Ana	Oliveira	200
789	Clara	Silva	300
963	Deyse	Silva	330

Função CONCAT(coluna1, coluna2): retorna duas ou mais colunas concatenadas.

Ex: `SELECT CONCAT(Primeiro_Nome, Familia_Nome)
FROM Empregado;`

Resultado

CONCAT(Primeiro_Nome, Familia_Nome)
IgorPereira
DeniseMoreno
MarceloNeiva
AnaOliveira
ClaraSilva
DeyseSilva

programariscool@gmail.com

Marcio Victorino



3 - CONCAT

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario
123	Igor	Pereira	100
159	Denise	Moreno	440
369	Marcelo	Neiva	900
456	Ana	Oliveira	200
789	Clara	Silva	300
963	Deyse	Silva	330

Melhoraremos a consulta anterior, colocando um espaço entre os literais e mudaremos o nome da coluna resultante usando um alias por meio do uso do termo “as”.

Ex: **SELECT CONCAT(Primeiro_Nome, ' ', Familia_Nome) AS Nome_Completo
FROM Empregado;**

Resultado

Nome_Completo
Igor Pereira
Denise Moreno
Marcelo Neiva
Ana Oliveira
Clara Silva
Deyse Silva

4 - SUBSTR

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario
123	Igor	Pereira	100
159	Denise	Moreno	440
369	Marcelo	Neiva	900
456	Ana	Oliveira	200
789	Clara	Silva	300
963	Deyse	Silva	330

Função SUBSTR(coluna, numero1, numero2): retorna um subconjunto de caracteres de uma coluna. O primeiro parâmetro numérico (numero1) indica a posição inicial do subconjunto de caracteres a ser recuperado e o segundo parâmetro numérico (numero2) indica a quantidade de caracteres. Cabe ressaltar que o primeiro caractere de um literal representa a posição 1.

Ex: SELECT SUBSTR(Familia_Nome, 2, 4)

FROM Empregado;

Resultado

SUBSTR(Familia_Nome, 2, 4)
erei
oren
eiva
live
ilva
ilva

Obs: Na consulta acima, o número 2 representa a posição inicial do subconjunto de caracteres a ser recuperado e o número 4 indica a quantidade de caracteres.

programariscool@gmail.com

Marcio Victorino



5 - LENGTH

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario
123	Igor	Pereira	100
159	Denise	Moreno	440
369	Marcelo	Neiva	900
456	Ana	Oliveira	200
789	Clara	Silva	300
963	Deyse	Silva	330

Função LENGTH(coluna): retorna o número de caracteres de uma coluna de uma tabela.

Ex: `SELECT LENGTH(Familia_Nome)`
`FROM Empregado;`

Resultado

LENGTH(Familia_Nome)
7
6
5
8
5
5

programariscool@gmail.com

Marcio Victorino



6 - REPLACE

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario
123	Igor	Pereira	100
159	Denise	Moreno	440
369	Marcelo	Neiva	900
456	Ana	Oliveira	200
789	Clara	Silva	300
963	Deyse	Silva	330

Função REPLACE(coluna, cadeia1, cadeia2): substitui um conjunto de caracteres de uma coluna quando parte do conteúdo dessa coluna coincidir com a cadeia1. Nesse caso, os caracteres da cadeia1 serão substituídos pelos caracteres da cadeia2.

Ex: `SELECT REPLACE(Primeiro_Nome, 'Dey', 'Day')`
FROM Empregado;

Resultado

REPLACE(Primeiro_Nome, 'Dey', 'Day')
Igor
Denise
Marcelo
Ana
Clara
Dayse

Obs: Na consulta acima, o foco da substituição realizada pela função REPLACE foi a coluna Primeiro_Nome da tabela Empregado. Então, a função percorreu todos as ocorrências do campo Primeiro_Nome e quando encontrou a cadeia de caracteres 'Dey', substituiu por 'Day'.

programariscool@gmail.com

Marcio Victorino



7 - COALESCE

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario	Sexo
123	Igor	Pereira	100	M
159	Denise	Moreno	440	F
369	Marcelo	Neiva	900	NULL
456	Ana	Oliveira	200	NULL
789	Clara	Silva	300	F
963	Deyse	Silva	330	F

Função COALESCE(coluna, valor): substitui um valor nulo (NULL) de uma coluna por um valor determinado.

Ex: `SELECT COALESCE(Sexo, 'I')`
FROM Empregado;

Resultado

COALESCE(Sexo, 'I')
M
F
I
I
F
F

Obs: Na consulta acima, o foco da substituição realizada pela função COALESCE foi a coluna Sexo da tabela Empregado. Então, a função percorreu todos as ocorrências do campo Sexo e quando encontrou 'NULL', substituiu por 'I'. Cabe ressaltar que o valor a ser usado para substituir o 'NULL' é determinado pelo programador da consulta. A função COALESCE também pode ser utilizada para dados numéricos.

programariscool@gmail.com

Marcio Victorino



8 - LPAD

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario	Sexo
123	Igor	Pereira	100	M
159	Denise	Moreno	440	F
369	Marcelo	Neiva	900	NULL
456	Ana	Oliveira	200	NULL
789	Clara	Silva	300	F
963	Deyse	Silva	330	F

Função LPAD(coluna, comprimento, ‘char’): adiciona o caractere ‘char’ à esquerda (left) da coluna de uma tabela até que tenha a quantidade de caracteres igual a comprimento.

Ex: `SELECT Primeiro_Nome, LPAD(Primeiro_Nome, 10, '-')
FROM Empregado;`

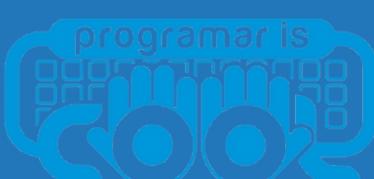
Resultado

Primeiro_Nome	LPAD(Primeiro_Nome, 10, '-')
Igor	-----Igor
Denise	----Denise
Marcelo	---Marcelo
Ana	-----Ana
Clara	----Clara
Deyse	----Deyse

Obs: Na consulta acima, todas as ocorrências resultantes da função LPAD ocupam 10 caracteres. A função recuperou o Primeiro_Nome e completou as 10 posições colocando o caractere ‘-’ à esquerda. Cabe ressaltar que qualquer outro caractere poderia ter sido utilizado.

programariscool@gmail.com

Marcio Victorino



9 - RPAD

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario	Sexo
123	Igor	Pereira	100	M
159	Denise	Moreno	440	F
369	Marcelo	Neiva	900	NULL
456	Ana	Oliveira	200	NULL
789	Clara	Silva	300	F
963	Deyse	Silva	330	F

Função RPAD(coluna, comprimento, ‘char’): adiciona o caractere ‘char’ à direita (right) da coluna de uma tabela até que tenha a quantidade de caracteres igual a comprimento.

Ex: `SELECT Primeiro_Nome, RPAD(Primeiro_Nome, 10, '-')
FROM Empregado;`

Resultado

Primeiro_Nome	RPAD(Primeiro_Nome, 10, '-')
Igor	Igor-----
Denise	Denise----
Marcelo	Marcelo---
Ana	Ana-----
Clara	Clara----
Deyse	Deyse-----

Obs: Na consulta acima, todas as ocorrências resultantes da função RPAD ocupam 10 caracteres. A função recuperou o Primeiro_Nome e completou as 10 posições colocando o caractere ‘-’ à direita. Cabe ressaltar que qualquer outro caractere poderia ter sido utilizado.

programariscool@gmail.com

Marcio Victorino



10 - LTRIM

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario	Sexo
123	Igor	Pereira	100	M
159	Denise	Moreno	440	F
369	Marcelo	Neiva	900	NULL
456	Ana	Oliveira	200	NULL
789	Clara	Silva	300	F
963	Deyse	Silva	330	F

Função LTRIM(coluna): remove caracteres em branco que estejam no início (à esquerda - left) do conteúdo da coluna de uma tabela.

Ex: `SELECT LTRIM(Primeiro_Nome)`
FROM Empregado;

Resultado

LTRIM(Primeiro_Nome)
Igor
Denise
Marcelo
Ana
Clara
Deyse

Obs: Na consulta acima, todas as ocorrências do caractere “em branco” (ou “espaço”) que estavam à esquerda dos dados da coluna Primeiro_Nome foram removidas. Esses caracteres estavam antes dos nomes “Denise”, “Ana” e “Clara”. Cabe ressaltar que também existe a função RTRIM que remove as ocorrências do caractere “em branco” à direita do conteúdo da coluna de uma tabela.

programariscool@gmail.com

Marcio Victorino



FUNÇÕES NUMÉRICAS

11 - AVG

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario
123	Igor	Pereira	200.20
159	Denise	Moreno	400.40
369	Marcelo	Neiva	500.50
456	Ana	Oliveira	600.60
789	Clara	Silva	700.70
963	Deyse	Silva	800.80

Função AVG(coluna): retorna a média do conteúdo das colunas de uma tabela.

Ex: **SELECT AVG(Salario)**
FROM Empregado;

Resultado
AVG(Salario)
533.866667

Obs: Na consulta acima, foi calculada a média dos salários dos empregados.

programariscool@gmail.com

Marcio Victorino



12 - COUNT

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario
123	Igor	Pereira	200.20
159	Denise	Moreno	400.40
369	Marcelo	Neiva	500.50
456	Ana	Oliveira	600.60
789	Clara	Silva	700.70
963	Deyse	Silva	800.80

Função COUNT(coluna): retorna a quantidade de ocorrências diferentes de NULL nas colunas de uma tabela.

Ex: `SELECT COUNT(Salario)`
FROM Empregado;

Resultado
COUNT(Salario)
6

Obs: Na consulta acima, foi contada a quantidade de salários existentes. Caso algum salário estivesse sem valor, ou seja, fosse NULL, ele não seria contado.

13 - SUM

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario
123	Igor	Pereira	200.20
159	Denise	Moreno	400.40
369	Marcelo	Neiva	500.50
456	Ana	Oliveira	600.60
789	Clara	Silva	700.70
963	Deyse	Silva	800.80

Função SUM(coluna): retorna a soma do conteúdo das colunas de uma tabela.

Ex: `SELECT SUM(Salario)
FROM Empregado;`

Resultado
<code>SUM(Salario)</code>
3203.20

Obs: Na consulta acima, foi calculada a soma total de todos os salários dos empregados.

14 - MAX

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario
123	Igor	Pereira	200.20
159	Denise	Moreno	400.40
369	Marcelo	Neiva	500.50
456	Ana	Oliveira	600.60
789	Clara	Silva	700.70
963	Deyse	Silva	800.80

Função MAX(coluna): retorna o valor máximo do conteúdo das colunas de uma tabela.

Ex: `SELECT MAX(Salario)
FROM Empregado;`

Resultado
MAX(Salario)
800.80

Obs: A consulta acima retornou o maior de todos os salários dos empregados.

programariscool@gmail.com

Marcio Victorino



15 - MIN

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Salario
123	Igor	Pereira	200.20
159	Denise	Moreno	400.40
369	Marcelo	Neiva	500.50
456	Ana	Oliveira	600.60
789	Clara	Silva	700.70
963	Deyse	Silva	800.80

Função MIN(coluna): retorna o valor mínimo do conteúdo das colunas de uma tabela.

Ex: `SELECT MIN(Salario)`
FROM Empregado;

Resultado
<code>MIN(Salario)</code>
200.20

Obs: A consulta acima retornou o menor de todos os salários dos empregados.

programariscool@gmail.com

Marcio Victorino



16 - ABS

Tabela Valores

Valor1	Valor2	Valor3
1	4	100.568
2	1	200.453
3	0	300.635
4	-3	400.542
5	-8	500.287

Função ABS(coluna): retorna o valor absoluto (positivo) da coluna de uma tabela, ou seja, altera valores negativos para valores positivos.

Ex: **SELECT ABS(Valor2)**
FROM Valores;

Resultado

ABS(Valor2)
4
1
0
3
8

programariscool@gmail.com

Marcio Victorino



17 - CEIL

Tabela Valores

Valor1	Valor2	Valor3
1	4	100.568
2	1	200.453
3	0	300.635
4	-3	400.542
5	-8	500.287

Função CEIL(coluna): retorna o valor inteiro imediatamente superior ou igual ao valor da coluna de uma tabela.

Ex: **SELECT CEIL(Valor3)**
FROM Valores;

Resultado

CEIL(Valor3)
101
201
301
401
501

programariscool@gmail.com

Marcio Victorino



18 - FLOOR

Tabela Valores

Valor1	Valor2	Valor3
1	4	100.568
2	1	200.453
3	0	300.635
4	-3	400.542
5	-8	500.287

Função FLOOR(coluna): retorna o valor inteiro imediatamente inferior ou igual ao valor da coluna de uma tabela.

Ex: **SELECT FLOOR(Valor3)**
FROM Valores;

Resultado

FLOOR(Valor3)
100
200
300
400
500

programariscool@gmail.com

Marcio Victorino



19 - MOD

Tabela Valores

Valor1	Valor2	Valor3
1	4	100.568
2	1	200.453
3	0	300.635
4	-3	400.542
5	-8	500.287

Função MOD(coluna, n): retorna o resto resultante da divisão do valor da coluna de uma tabela por "n".

Ex: `SELECT MOD(Valor1, 2)`
`FROM Valores;`

Resultado

MOD(Valor1, 2)
1
0
1
0
1

Obs: A consulta acima retornou o resto da divisão da coluna Valor1 pelo número 2. Essa consulta é utilizada para encontrar os valores da coluna Valor1 que são pares, pois se o resto da divisão por 2 for 0, o número é par.

programariscool@gmail.com

Marcio Victorino



20 - POWER

Tabela Valores

Valor1	Valor2	Valor3
1	4	100.568
2	1	200.453
3	0	300.635
4	-3	400.542
5	-8	500.287

Função POWER(coluna, expoente): retorna o valor da coluna de uma tabela elevado ao número “expoente”.

Ex: `SELECT POWER(Valor1, 2)`
`FROM Valores;`

Resultado

POWER(Valor1, 2)
1
4
9
16
25

21 - SQRT

Tabela Valores

Valor1	Valor2	Valor3
1	4	100.568
2	1	200.453
3	0	300.635
4	-3	400.542
5	-8	500.287

Função SQRT(coluna): retorna a raiz quadrada do valor da coluna de uma tabela.

Ex: `SELECT SQRT(Valor1)`
`FROM Valores;`

Resultado

SQRT(Valor1)
1
1.4142135623730951
1.7320508075688772
2
2.23606797749979

programariscool@gmail.com

Marcio Victorino



22 - SIGN

Tabela Valores

Valor1	Valor2	Valor3
1	4	100.568
2	1	200.453
3	0	300.635
4	-3	400.542
5	-8	500.287

Função SIGN(coluna): retorna -1 se o valor da coluna de uma tabela for negativo, 1 se o valor for positivo e 0 se o valor for zero.

Ex: **SELECT SIGN(Valor2)**
FROM Valores;

Resultado

SIGN(Valor2)
1
1
0
-1
-1

programariscool@gmail.com

Marcio Victorino



23 - ROUND

Tabela Valores

Valor1	Valor2	Valor3
1	4	100.568
2	1	200.453
3	0	300.635
4	-3	400.542
5	-8	500.287

Função ROUND(coluna, n): retorna o valor de uma coluna arredondado para “n” casas decimais. Isso significa que se o algarismo na posição n+1 for igual ou superior a 5, o algarismo na posição n é acrescido de 1.

Ex: `SELECT ROUND(Valor3, 2)`
`FROM Valores;`

Resultado
<code>ROUND(Valor3, 2)</code>
100.57
200.45
300.64
400.54
500.29

24 - TRUNCATE

Tabela Valores

Valor1	Valor2	Valor3
1	4	100.568
2	1	200.453
3	0	300.635
4	-3	400.542
5	-8	500.287

Função TRUNCATE(coluna, n): retorna o valor de uma coluna truncado para “n” casas decimais. Diferentemente da função ROUND, a função TRUNCATE não faz arredondamento, ela simplesmente elimina algarismos.

Ex: `SELECT TRUNCATE(Valor3, 2)`
`FROM Valores;`

Resultado

TRUNCATE(Valor3, 2)
100.56
200.45
300.63
400.54
500.28

programariscool@gmail.com

Marcio Victorino



25 - LOG

Tabela Valores

Valor1	Valor2	Valor3
1	4	100.568
2	1	200.453
3	0	300.635
4	-3	400.542
5	-8	500.287

Função LOG(coluna, n): retorna o logaritmo do valor de uma coluna considerando como base o número “n”.

Ex: `SELECT LOG(Valor1, 2)`
`FROM Valores;`

Resultado

LOG(Valor1, 2)
NULL
1
0.6309297535714574
0.5
0.43067655807339306

Obs: Repare que quando o Valor1 da tabela Valores é 1 (primeiro registro), a consulta retorna “NULL”. Isto acontece porque $\text{LOG}(1, 2)$ não existe. Por outro lado, o $\text{LOG}(2, 2)$ retornou 1, pois 2 elevado a 1 resulta em 2.

programariscool@gmail.com

Marcio Victorino



FUNÇÕES DE DATAS

26 - DATE FORMAT

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função DATE_FORMAT(coluna, '%d/%m/%Y'): formata uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: SELECT DATE_FORMAT(Data_Admissao, '%d/%m/%Y')
FROM Empregado;

Resultado

DATE_FORMAT(Data_Admissao, '%d/%m/%Y')
01/01/2019
18/02/2016
01/01/2012
28/04/2008
12/07/2010
01/11/2014

Obs: A consulta acima transforma a data do formato americano para o europeu.

programariscool@gmail.com

Marcio Victorino



27 - EXTRACT

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função EXTRACT(YEAR FROM coluna): retorna o ano, mês ou dia de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT EXTRACT(YEAR FROM Data_Admissao)
FROM Empregado;`

Resultado

EXTRACT(YEAR FROM Data_Admissao)
2019
2016
2012
2008
2010
2014

Obs: Pode-se utilizar a sintaxe “MONTH FROM” ou “DAY FROM” para retornar o mês e o dia, respectivamente.

programariscool@gmail.com

Marcio Victorino



28 - DAY

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função DAY(coluna): retorna o dia de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT DAY(Data_Admissao)`
FROM Empregado;

Resultado

DAY(data_admissao)
1
18
1
28
12
1

programariscool@gmail.com

Marcio Victorino



29 - DAYNAME

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função DAYNAME(coluna): retorna o nome do dia de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT DAYNAME(Data_Admissao)`
FROM Empregado;

Resultado

DAYNAME(Data_Admissao)
Tuesday
Thursday
Sunday
Monday
Monday
Saturday

programariscool@gmail.com

Marcio Victorino



30 - MONTH

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função MONTH(coluna): retorna o mês de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT MONTH(Data_Admissao)`
`FROM Empregado;`

Resultado

MONTH(data_admissao)
1
2
1
4
7
11

programariscool@gmail.com

Marcio Victorino



31 - MONTHNAME

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função MONTHNAME(coluna): retorna o nome do mês de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT MONTHNAME(Data_Admissao)`
`FROM Empregado;`

Resultado

MONTHNAME(Data_Admissao)
January
February
January
April
July
November

programariscool@gmail.com

Marcio Victorino



32 - YEAR

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função YEAR(coluna): retorna o ano de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT YEAR(Data_Admissao)`
FROM Empregado;

Resultado

YEAR(data_admissao)
2019
2016
2012
2008
2010
2014

programariscool@gmail.com

Marcio Victorino



33 - DAYOFYEAR

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função DAYOFYEAR(coluna): retorna o dia do ano de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT DAYOFYEAR(Data_Admissao)`
FROM Empregado;

Resultado

DAYOFYEAR(Data_Admissao)
1
49
1
119
193
305

Obs: A consulta acima retornou o dia do ano das datas de admissão. Por exemplo, a primeira linha informa que a data consultada equivale ao primeiro dia do ano.

programariscool@gmail.com

Marcio Victorino



34 - DAYOFMONTH

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função DAYOFMONTH(coluna): retorna o dia do mês de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT DAYOFMONTH(Data_Admissao)`
FROM Empregado;

Resultado

DAYOFMONTH(Data_Admissao)
1
18
1
28
12
1

Obs: A consulta acima retornou o dia do mês das datas de admissão. Por exemplo, a primeira linha informa que a data consultada equivale ao primeiro dia do mês.

programariscool@gmail.com

Marcio Victorino



35 - DAYOFWEEK

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função DAYOFWEEK(coluna): retorna o dia da semana de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT DAYOFWEEK(Data_Admissao)`
FROM Empregado;

Resultado

DAYOFWEEK(Data_Admissao)
3
5
1
2
2
7

Obs: A consulta acima retornou o dia da semana das datas de admissão. Por exemplo, a primeira linha informa que a data consultada equivale ao terceiro dia da semana.

programariscool@gmail.com

Marcio Victorino



36 - DATEDIFF

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função DATEDIFF(col1, col2): retorna o intervalo entre duas datas armazenadas nas colunas de uma tabela. Essa função também pode ser usada considerando diretamente duas datas, sem necessariamente consultar tabelas.

Ex: SELECT DATEDIFF(Data_Demissao, Data_Admissao)

FROM Empregado;

Resultado

DATEDIFF(Data_Demissao, Data_Admissao)
364
980
875
2518
3630
1209

Obs: Na consulta acima, foi calculado o número de dias que cada empregado trabalhou na empresa.

programariscool@gmail.com

Marcio Victorino



37 - DATE_ADD

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função DATE_ADD(coluna, numero_dias): adiciona dias a uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT DATE_ADD(Data_Demissao, interval 30 day)`
FROM Empregado;

Resultado

DATE_ADD(Data_Demissao, interval 30 day)
2020-01-30
2018-11-24
2014-06-24
2015-04-20
2020-07-19
2018-03-24

Obs: Na consulta acima, foram acrescentados 30 dias à data de demissão dos empregados da empresa. Essa função também pode utilizar intervalos de meses (month) ou anos (year).

programariscool@gmail.com

Marcio Victorino



38 - DATE_SUB

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função DATE_SUB(coluna, numero_dias): subtrai dias de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT DATE_SUB(Data_Demissao, interval 15 day)`
FROM Empregado;

Resultado

DATE_Sub(Data_Demissao, interval 15 day)
2019-12-16
2018-10-10
2014-05-10
2015-03-06
2020-06-04
2018-02-07

Obs: Na consulta acima, foram subtraídos 15 dias da data de demissão dos empregados da empresa. Essa função também pode utilizar intervalos de meses (month) ou anos (year).

programariscool@gmail.com

Marcio Victorino



39 - LAST_DAY

Tabela Empregado

Matricula	Primeiro_Nome	Familia_Nome	Data_Admissao	Data_Demissao
123	Igor	Pereira	2019-01-01	2019-12-31
159	Denise	Moreno	2016-02-18	2018-10-25
369	Marcelo	Neiva	2012-01-01	2014-05-25
456	Ana	Oliveira	2008-04-28	2015-03-21
789	Clara	Silva	2010-07-12	2020-06-19
963	Deyse	Silva	2014-11-01	2018-02-22

Função LAST_DAY(coluna): retorna o último dia do mês de uma data armazenada na coluna de uma tabela. Essa função também pode ser usada considerando diretamente uma data, sem necessariamente consultar tabelas.

Ex: `SELECT LAST_DAY(Data_Admissao)`
FROM Empregado;

Resultado

LAST_DAY(Data_Admissao)
2019-01-31
2016-02-29
2012-01-31
2008-04-30
2010-07-31
2014-11-30

programariscool@gmail.com

Marcio Victorino



40 - CURDATE

Função CURDATE(): retorna data atual. Repare que essa função não usa colunas de tabelas, ela simplesmente apresenta a data do sistema.

Ex: `SELECT CURDATE();`

Resultado

CURDATE()
2020-08-21

41 - NOW

Função NOW(): retorna data/hora atual. Repare que essa função não usa colunas de tabelas, ela simplesmente apresenta a data/hora do sistema.

Ex: `SELECT NOW();`

Resultado

NOW()
2020-08-21 18:08:30

programariscool@gmail.com

Marcio Victorino



42 - SYSDATE

Função SYSDATE(): retorna data/hora atual. Repare que essa função não usa colunas de tabelas, ela simplesmente apresenta a data/hora do sistema.

Ex: **SELECT SYSDATE();**

Resultado

SYSDATE()
2020-08-21 18:10:58

43 - CURTIME

Função CURTIME(): retorna o horário atual. Repare que essa função não usa colunas de tabelas, ela simplesmente apresenta o horário atual do sistema.

Ex: **SELECT CURTIME();**

Resultado

CURTIME()
18:14:43

programariscool@gmail.com

Marcio Victorino



44 - TO_DATE

Função TO_DATE(): é uma função para conversão de tipos que converte do formato cadeia de caracteres para data.

45 - TO_CHAR

Função TO_CHAR(): é uma função para conversão de tipos que converte do formato data para cadeia de caracteres.

programariscool@gmail.com

Marcio Victorino

