

# **BANCO DE DADOS**

**Linguagem SQL**

**Consultas com funções - SELECT**

# SQL (STRUCTURED QUERY LANGUAGE)

Esquema para o modelo Relacional do contexto didático:  
**EMPRESA.**

FUNCIONARIO (ident, nome, sobrenome, endereco, dtnasc, salario, sexo, supident, dnumero)

DEPENDENTE (fident, nome, dt\_nasc, sexo, relacionamento)

DEPARTAMENTO (numero, nome, gident, dtinicio)

LOCALIZACOES (dnumero, localizacao)

PROJETO (numero, nome, localizacao, dnumero)

TRABALHA\_EM (pnumero, fident, horas)

Algumas simplificações nos nomes de variáveis foram realizadas para facilitar o uso delas nas consultas. A chave primária, em cada relação, está sublinhada.

# FUNÇÕES

- A linguagem SQL implementa uma variedade de funções que podem ser utilizadas de diversas formas.
- Importante observar que algumas funções são implementadas por apenas alguns produtos, e podem também ter nomes diferentes em produtos diferentes

**Apresentaremos a seguir algumas funções do MySQL (que podem ou não ser implementadas em outros produtos)**

# FUNÇÕES NUMÉRICAS

## TRUNCATE (X,D)

```
1  mysql> SELECT TRUNCATE(1.223,1);
2      -> 1.2
3  mysql> SELECT TRUNCATE(1.999,1);
4      -> 1.9
5  mysql> SELECT TRUNCATE(1.999,0);
6      -> 1
7  mysql> SELECT TRUNCATE(-1.999,1);
8      -> -1.9
9  mysql> SELECT TRUNCATE(122,-2);
10     -> 100
11  mysql> SELECT TRUNCATE(10.28*100,0);
12     -> 1028
```

## ROUND (X), ROUND (X,D)

```
1  mysql> SELECT ROUND(-1.23);
2      -> -1
3  mysql> SELECT ROUND(-1.58);
4      -> -2
5  mysql> SELECT ROUND(1.58);
6      -> 2
7  mysql> SELECT ROUND(1.298, 1);
8      -> 1.3
9  mysql> SELECT ROUND(1.298, 0);
10     -> 1
11  mysql> SELECT ROUND(23.298, -1);
12     -> 20
```

# FUNÇÕES STRING

CONCAT (str1,str2...), CONCAT\_WS (separador, str1, str2...)

```
1  mysql> SELECT CONCAT('My', 'S', 'QL');  
2      -> 'MySQL'  
3  mysql> SELECT CONCAT('My', NULL, 'QL');  
4      -> NULL  
5  mysql> SELECT CONCAT(14.3);  
6      -> '14.3'
```

```
1  mysql> SELECT CONCAT_WS(',', 'First name', 'Second name', 'Last Name');  
2      -> 'First name,Second name,Last Name'  
3  mysql> SELECT CONCAT_WS(',', 'First name', NULL, 'Last Name');  
4      -> 'First name,Last Name'
```

LOWER (str), UPPER (str)

```
1  mysql> SELECT LOWER('QUADRATICALLY');  
2      -> 'quadratically'
```

```
1  mysql> SELECT UPPER('Hej');  
2      -> 'HEJ'
```

# FUNÇÕES STRING

SUBSTRING(str,pos),  
SUBSTRING(str FROM pos),  
SUBSTRING(str,pos,qtde),  
SUBSTRING(str FROM pos FOR qtde)

```
1  mysql> SELECT SUBSTRING('Quadratically',5);  
2      -> 'ratically'  
3  mysql> SELECT SUBSTRING('foobarbar' FROM 4);  
4      -> 'barbar'  
5  mysql> SELECT SUBSTRING('Quadratically',5,6);  
6      -> 'ratica'  
7  mysql> SELECT SUBSTRING('Sakila', -3);  
8      -> 'ila'  
9  mysql> SELECT SUBSTRING('Sakila', -5, 3);  
10     -> 'aki'  
11  mysql> SELECT SUBSTRING('Sakila' FROM -4 FOR 2);  
12     -> 'ki'
```

LEFT (str1,tamanho)

```
1  mysql> SELECT LEFT('foobarbar', 5);  
2      -> 'fooba'
```

REPLACE (str, origem, destino)

```
1  mysql> SELECT REPLACE('www.mysql.com', 'w', 'ww');  
2      -> 'wwwwww.mysql.com'
```



# FUNÇÕES DE DATA

NOW ()

```
1 mysql> SELECT NOW();  
2         -> '2007-12-15 23:50:26'  
3 mysql> SELECT NOW() + 0;  
4         -> 20071215235026.000000
```

MONTH(data), MONTHNAME (data)

```
1 mysql> SELECT MONTH('2008-02-03');  
2         -> 2
```

```
1 mysql> SELECT MONTHNAME('2008-02-03');  
2         -> 'February'
```

DAYNAME (data), DAYOFMONTH (data)

DAYOFWEEK (data), DAYOFYEAR(data)

```
1 mysql> SELECT DAYNAME('2007-02-03');  
2         -> 'Saturday'
```

```
1 mysql> SELECT DAYOFWEEK('2007-02-03');  
2         -> 7
```

```
1 mysql> SELECT DAYOFMONTH('2007-02-03');  
2         -> 3
```

```
1 mysql> SELECT DAYOFYEAR('2007-02-03');  
2         -> 34
```

# FUNÇÕES DE DATA

DATADIFF(data1, data2) => data1 – data2

```
1  mysql> SELECT DATEDIFF('2007-12-31 23:59:59', '2007-12-30');  
2      -> 1  
3  mysql> SELECT DATEDIFF('2010-11-30 23:59:59', '2010-12-31');  
4      -> -31
```

TO\_DAYS(data) => Dias desde o ano zero

```
1  mysql> SELECT TO_DAYS('2008-10-07'), TO_DAYS('08-10-07');  
2      -> 733687, 733687
```

SUBDATE(data,dias) e variação

```
1  mysql> SELECT DATE_SUB('2008-01-02', INTERVAL 31 DAY);  
2      -> '2007-12-02'  
3  mysql> SELECT SUBDATE('2008-01-02', INTERVAL 31 DAY);  
4      -> '2007-12-02'
```



# EXEMPLO

Selecionar nome, sobrenome e data de nascimento de todos os funcionários que fazem aniversário no mês de “Agosto”.

```
SELECT nome, sobrenome, dtnasc  
FROM funcionario  
WHERE month(dtnasc)=8
```

# FUNCIONARIO

ident	nome	sobrenome	endereco	dtnasc	salario	sexo	supident	dnumero
1163	Claudia	Morais	Rua A – 1 --SP	12/08/1974	5.558,00	F	NULL	4
1164	Jorge	Vila Verde	Rua M – 25 -- SP	29/01/1986	1.550,00	M	1163	1
1165	Moacir	Junqueira	Rua F – 57 -- SP	08/11/1981	1.550,00	M	1164	1
1166	Patrícia	Sorte	Rua F –22 -- SP	22/06/1979	2.200,00	F	1163	4
1167	Caio	Brotas	Rua I –100 -- SP	15/08/1977	3.400,00	M	1163	3
1170	Antonio	Monde	Rua X - 5 -- SP	23/03/1985	2.300,00	M	1167	NULL

nome	sobrenome	dtnasc
Claudia	Morais	12/08/1974
Caio	Brotas	15/08/1977

# AGREGAÇÕES

- As funções de agregação são funções SQL que permitem executar uma operação aritmética nos valores de uma coluna em todos os registros de uma tabela.
- As funções são:
  - **AVG** (calcula a média dos valores de um campo),
  - **COUNT** (retorna a quantidade de registros existentes),
  - **SUM** (calcula a soma dos valores de um campo),
  - **MAX** (retorna o maior valor existente para um campo),
  - **MIN** (retorna o menor valor existente para um campo).

# GROUP BY - HAVING

- As funções de agregação são bastante utilizadas com os recursos de agrupamento, possibilitando que os valores agregados sejam consolidados por grupos.
- As funções de agregação podem utilizar a cláusula **GROUP BY** para serem executadas.
- A cláusula **HAVING** também pode ser utilizada para aplicar filtros pós agregações.
- Note que o **WHERE** filtra os dados antes da agregação, o **HAVING** filtra após a agregação.

ident	nome	sobrenome	endereco	dtnasc	salario	sexo	supident	dnumero
1163	Claudia	Morais	Rua A – 1 --SP	12/08/1974	5.558,00	F	NULL	4
1164	Jorge	Vila Verde	Rua M – 25 -- SP	29/01/1986	1.550,00	M	1163	1
1165	Moacir	Junqueira	Rua F – 57 -- SP	08/11/1981	1.550,00	M	1164	1
1166	Patrícia	Sorte	Rua F –22 -- SP	22/06/1979	2.200,00	F	1163	4
1167	Caio	Brotas	Rua I –100 -- SP	15/08/1977	3.400,00	M	1163	3
1170	Antonio	Monde	Rua X - 5 -- SP	23/03/1985	2.300,00	M	1167	NULL

- TABELA FUCIONARIO, que será utilizada para todos os exemplos que serão apresentados a seguir.



# AVG() - MÉDIA

```
SELECT AVG(salario) FROM funcionario
```

	AVG(salario)
1	2759.6667

```
SELECT AVG(salario) FROM funcionario  
WHERE dnumero=4
```

	AVG(salario)
1	3879.0000



# SUM() - SOMA

```
SELECT SUM(salario) FROM funcionario
```





	SQL	Result
1	SELECT SUM(salario) FROM funcionario	16558

```
SELECT SUM(salario) FROM funcionario  
WHERE dnumero=4
```



	SQL	Result
1	SELECT SUM(salario) FROM funcionario WHERE dnumero=4	7758

# MAX() E MIN() – MÁXIMO E MÍNIMO

```
SELECT MAX(salario), MIN(salario), MAX(dtnasc), MIN(dtnasc)
FROM funcionario
```




	 `MAX(salario)`	 `MIN(salario)`	 `MAX(dtnasc)`	 `MIN(dtnasc)`
1	5558	1550	1986-01-29	1974-08-12

```
SELECT MAX(salario), MIN(salario)
FROM funcionario WHERE dnumero=4
```



	 `MAX(salario)`	 `MIN(salario)`
1	5558	2200

# COUNT() – CONTAR

```
SELECT COUNT(*), COUNT(salario), COUNT(supident)
FROM funcionario
```

	 `COUNT(*)` ÷	 `COUNT(salario)` ÷	 `COUNT(supident)` ÷
1	6	6	5

```
SELECT COUNT(*), COUNT(supident)
FROM funcionario WHERE dnumero=1
```

	 `COUNT(*)` ÷	 `COUNT(supident)` ÷
1	2	2

# GROUP BY

```
SELECT dnumero, AVG(salario), SUM(salario)
FROM funcionario GROUP BY (dnumero)
```

	dnumero	`AVG(salario)`	`SUM(salario)`
1	1	1550.0000	3100
2	4	3879.0000	7758
3	3	3400.0000	3400
4	<null>	2300.0000	2300

```
SELECT dnumero, COUNT(*), MAX(salario)
FROM funcionario WHERE dnumero>1
GROUP BY (dnumero)
```

	dnumero	`COUNT(*)`	`MAX(salario)`
1	4	2	5558
2	3	1	3400

# HAVING

SELECT dnumero, **AVG(salario), SUM(salario)** FROM funcionario  
GROUP BY (dnumero) **HAVING SUM(salario)<5000**

	dnumero	AVG(salario)	SUM(salario)
1	1	1550.0000	3100
2	3	3400.0000	3400
3	<null>	2300.0000	2300

SELECT dnumero, **COUNT(\*), MAX(salario)**  
FROM funcionario **WHERE dnumero>1**  
GROUP BY (dnumero) **HAVING MAX(SALARIO)>5000**

	dnumero	COUNT(*)	MAX(salario)
1	4	2	5558

# **BANCO DE DADOS**

**Linguagem SQL**

**Consultas com funções - SELECT**