DOCUMENTAÇÃO – AULA INVERTIDA G3

SENAI . SP

TÉCNICO EM DESENVOLVIMENTO

Funções Nativas

SENAI
SÃO PAULO

Sumário

1. Funções Nativas

As funções nativas são basicamente as funções padrão do SQL- server.

2. Funções de Agregação

As funções de agregação permitem executar uma operação aritmética nos valores de uma coluna em todos os registros de uma tabela. Ela processa um conjunto de valores contidos em uma única coluna de uma tabela e retorna um único valor como resultado.

AVG

Retorna o valor da média (a soma de todos os valores divido pela quantidade) de um conjunto de valores.

SUM

Soma todos os valores de um conjunto de valores, retornando um valor total no final.

COUNT

Função de contagem, conta quantos elementos possuem em um conjunto de valores (quantas linhas possuem em uma tabela, por exemplo).

MAX

Retorna o maior valor de um conjunto de valores.

MIN

Retorna o menor valor de um conjunto de valores.

3. Funções Analíticas

Uma função analítica calcula valores em um grupo de linhas e retorna um único resultado para cada linha. Isso é diferente de uma função de agregação, que retorna um único resultado para um grupo de linhas.

CUME_DIST

Vamos criar uma tabela nomeada scores com alguns dados de amostra para a demonstração:

```
CREATE TABLE scores (name VARCHAR(20) PRIMARY KEY, score INT NOT NULL);

INSERT INTO scores(name, score)

VALUES      ('Smith',81),
            ('Jones',55),
            ('Williams',55),
            ('Taylor',62),
            ('Brown',62),
            ('Davies',84),
            ('Evans',87),
            ('Wilson',72),
            ('Thomas',72),
            ('Johnson',100);
```

A declaração a seguir encontra a distribuição cumulativa da pontuação no conjunto de resultados:

```
SELECT name, score,

 ROW_NUMBER() OVER (ORDER BY score) row_num, CUME_DIST() OVER (ORDER BY score) cume_dist_val FROM scores;
```

name	score	row_num	cume_dist_val
Jones	55	1	0.2
Williams	55	2	0.2
Taylor	62	3	0.4
Brown	62	4	0.4
Thomas	72	5	0.6
Wilson	72	6	0.6
Smith	81	7	0.7
Davies	84	8	0.8
Evans	87	9	0.9
Johnson	100	10	1

Neste exemplo, a pontuação é classificada em ordem crescente de 55 a 100. Observe que a ROW_NUMBER()função foi adicionada para referência.

Então, como a CUME_DIST()função executa o cálculo?

Para a primeira linha, a função encontra o número de linhas no conjunto de resultados, que têm valor menor ou igual a 55. O resultado é 2. Então a CUME_DIST()função divide 2 pelo número total de linhas que é 10: 2/10. o resultado é 0,2 ou 20%. A mesma lógica é aplicada à segunda linha.

LAG

A LAG() função é uma função de janela que permite que você olhe para trás várias linhas e acesse os dados desta linha a partir da linha atual.

productline	order_year	order_value	prev_year_order_value
Classic Cars	2003	1374832	NULL
Classic Cars	2004	1763137	1374832
Classic Cars	2005	715954	1763137
Motorcycles	2003	348909	NULL
Motorcycles	2004	527244	348909
Motorcycles	2005	245273	527244
Planes	2003	309784	NULL
Planes	2004	471971	309784
Planes	2005	172882	471971
Ships	2003	222182	NULL
Ships	2004	337326	222182
Ships	2005	104490	337326

Primeiro, usamos uma expressão de tabela comum para obter o valor do pedido de cada produto em cada ano.

Em seguida, dividimos os produtos usando as linhas de produtos em partições, classificamos cada partição por ano do pedido e aplicamos a LAG()função a cada partição classificada para obter o valor do pedido do ano anterior de cada produto.

LAST-VALUE

A LAST_VALUE() é uma função de janela que permite selecionar a última linha em um conjunto ordenado de linhas.

CREATE TABLE overtime (employee_name VARCHAR(50) NOT NULL, department VARCHAR(50) NOT NULL, hours INT NOT NULL, PRIMARY KEY (employee_name , department));

INSERT INTO overtime(employee_name, department, hours)

VALUES ('Diane Murphy','Accounting',37),

('Mary Patterson','Accounting',74),

('Jeff Firrelli','Accounting',40),

('William Patterson','Finance',58),

('Gerard Bondur','Finance',47),

('Anthony Bow','Finance',66),

('Leslie Jennings','IT',90),

('Leslie Thompson','IT',88),

('Julie Firrelli','Sales',81),

('Steve Patterson','Sales',29),

('Foon Yue Tseng','Sales',65),

('George Vanauf','Marketing',89),

('Loui Bondur','Marketing',49),

('Gerard Hernandez','Marketing',66),

('Pamela Castillo','SCM',96),

('Larry Bott','SCM',100),

('Barry Jones','SCM',65);

A seguinte declaração obtém o nome do funcionário, horas extras e o funcionário que tem o maior número de horas extras:

SELECT employee_name, hours, LAST_VALUE(employee_name) OVER (ORDER BY hours RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) highest_overtime_employee FROM overtime;

employee_name	hours	highest_overtime_employee
Steve Patterson	29	Larry Bott
Diane Murphy	37	Larry Bott
Jeff Firrelli	40	Larry Bott
Gerard Bondur	47	Larry Bott
Loui Bondur	49	Larry Bott
William Patterson	58	Larry Bott
Foon Yue Tseng	65	Larry Bott
Barry Jones	65	Larry Bott
Anthony Bow	66	Larry Bott
Gerard Hernandez	66	Larry Bott
Mary Patterson	74	Larry Bott
Julie Firrelli	81	Larry Bott
Leslie Thompson	88	Larry Bott
George Vanauf	89	Larry Bott
Leslie Jennings	90	Larry Bott
Pamela Castillo	96	Larry Bott
Larry Bott	100	Larry Bott

Neste exemplo, a ORDER BY cláusula especificou a ordem lógica das linhas no conjunto de resultados por horas de baixo para alto.

SELECT employee_name, department, hours, LAST_VALUE(employee_name) OVER (PARTITION BY department ORDER BY hours RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) most_overtime_employee FROM overtime;

employee_name	department	hours	most_overtime_employee
Diane Murphy	Accounting	37	Mary Patterson
Jeff Firrelli	Accounting	40	Mary Patterson
Mary Patterson	Accounting	74	Mary Patterson
Gerard Bondur	Finance	47	Anthony Bow
William Patterson	Finance	58	Anthony Bow
Anthony Bow	Finance	66	Anthony Bow
Leslie Thompson	IT	88	Leslie Jennings
Leslie Jennings	IT	90	Leslie Jennings
Loui Bondur	Marketing	49	George Vanauf
Gerard Hernandez	Marketing	66	George Vanauf
George Vanauf	Marketing	89	George Vanauf
Steve Patterson	Sales	29	Julie Firrelli
Foon Yue Tseng	Sales	65	Julie Firrelli
Julie Firrelli	Sales	81	Julie Firrelli
Barry Jones	SCM	65	Larry Bott
Pamela Castillo	SCM	96	Larry Bott
Larry Bott	SCM	100	Larry Bott

Neste exemplo, primeiro, a PARTITION BY cláusula dividiu os funcionários por departamentos. Em seguida, a ORDER BY cláusula ordena os funcionários em cada departamento por horas extras de baixo para alto.

A especificação do quadro, neste caso, é a partição inteira. Como resultado, a LAST_VALUE() selecionou a última linha de cada partição que era o funcionário com maior número de horas extras.

4. Funções de Classificação

RANK

A função de RANK retorna à classificação de cada linha na partição de um conjunto de resultados. A classificação de uma linha é 1 mais o número de classificações que vêm antes da linha em questão. Por exemplo os produtos são classificados em uma ordem em determinado contexto que compara sua quantidade, mas como o produto 494 e 495 têm a mesma quantidade eles dividem a mesma classificação.

```
ProductID   Name                  LocationID   Quantity Rank
----------- --------------------- ------------ -------- ----
494         Paint - Silver        3            49       1
495         Paint - Blue          3            49       1
493         Paint - Red           3            41       3
496         Paint - Yellow        3            30       4
492         Paint - Black         3            17       5
495         Paint - Blue          4            35       1
496         Paint - Yellow        4            25       2
493         Paint - Red           4            24       3
492         Paint - Black         4            14       4
494         Paint - Silver        4            12       5
 (10 row(s) affected)
```

ROW_NUMBER

A função de ROW_NUMBER numera a saída de um conjunto de resultados. Basicamente retornando o número sequencial de uma linha em uma partição de um conjunto de resultados, começando em 1 na primeira linha de cada partição. Por exemplo a tabela a seguir calcula um número de linha para os vendedores com base em sua classificação de vendas.

```
Row FirstName    LastName              SalesYTD
--- ------------ --------------------- ------------
1   Linda        Mitchell              4251368.54
2   Jae          Pak                   4116871.22
3   Michael      Blythe                3763178.17
4   Jillian      Carson                3189418.36
5   Ranjit       Varkey Chudukatil     3121616.32
6   José         Saraiva               2604540.71
7   Shu          Ito                   2458535.61
8   Tsvi         Reiter                2315185.61
9   Rachel       Valdez                1827066.71
10  Tete         Mensa-Annan           1576562.19
11  David        Campbell              1573012.93
12  Garrett      Vargas                1453719.46
13  Lynn         Tsoflias              1421810.92
14  Pamela       Ansman-Wolfe          1352577.13
```

DANSE_RANK

A função DENSE_RANK retorna à posição de cada linha de uma partição do conjunto de resultados, sem nenhum intervalo nos valores de classificação. a Classificação de uma linha é 1 mais o número de valores de classificação distintos que vêm antes da linha em questão. Por exemplo os produtos são classificados em um inventário de acordo com determinado contexto, porém como os produtos 494 e 495 têm a mesma quantidade ambos são classificados como 1.

```
ProductID   Name                               LocationID Quantity Rank
----------- ---------------------------------- ---------- -------- -----
494         Paint - Silver                     3          49       1
495         Paint - Blue                       3          49       1
493         Paint - Red                        3          41       2
496         Paint - Yellow                     3          30       3
492         Paint - Black                      3          17       4
495         Paint - Blue                       4          35       1
496         Paint - Yellow                     4          25       2
493         Paint - Red                        4          24       3
492         Paint - Black                      4          14       4
494         Paint - Silver                     4          12       5

(10 row(s) affected)
```

5. Funções Escalares

LTRIM

A função de LTRIM é uma função de texto. Ela retira automaticamente os espaços do texto que ficam à esquerda**.**

LEFT

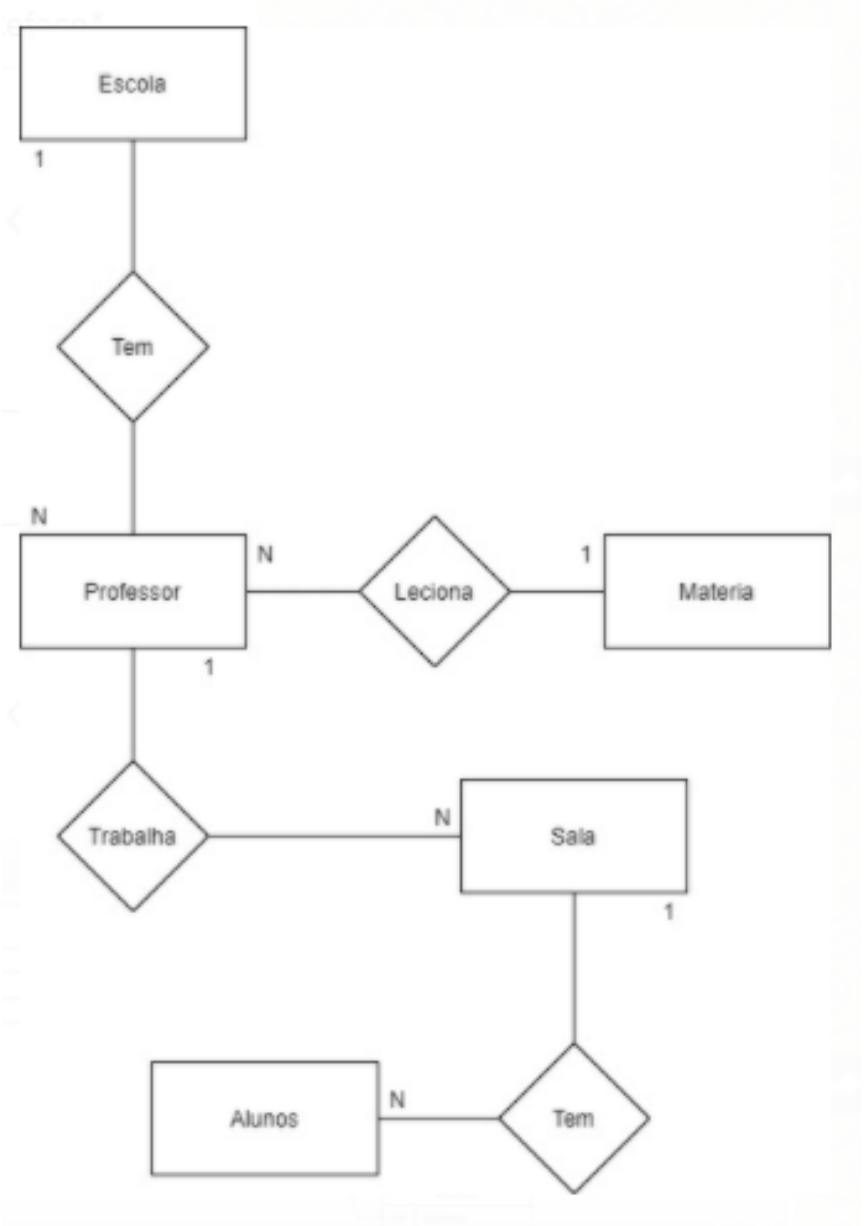
O LEFT extrai caracteres à esquerda de um campo ou um texto. Além de extrair caracteres, essa função te dá a margem de decidir quantos caracteres deseja excluir.

KAHOOT

Abaixo está disponibilizado o link do nosso **kahoot**:

https://create.kahoot.it/share/funcoes-nativas/b1eafbd4-dc29-445f-9fc2-af30cd0a9efe

MODELO CONCEITUAL



MODELO FÍSICO

MODELO LÓGICO