Bases de Dados

Módulo 17: Funções de Janela

Prof. André Bruno de Oliveira

28/05/24 08:20

Funções de janela

Definição:

- Uma função de janela (*window function*) executa um cálculo sobre um conjunto de linhas que, de alguma forma, estão relacionadas à linha corrente. Este conjunto de linhas é chamado da janela.

Funções de janela

Definição:

- Essas funções diferenciam-se de outras funções da SQL pelas seguintes características:
- Presença da cláusula OVER. Basicamente, se uma função tem a cláusula OVER ela vai operar como uma função de janela, incluindo as já conhecidas funções COUNT, SUM, AVG, MAX e MIN – isto é, OVER muda o comportamento dessas funções). A cláusula OVER é usada para a definição da janela (assim como GROUP BY é usada para definir grupos).

Funções de janela

Definição:

- Elas permitem com que você crie consultas capazes de misturar linhas individuais (não agrupadas) de uma tabela com resultados agregados – ao contrário da agregação com GROUP BY que agrupa diversas linhas em uma só no resultado final.

Funções de janela

Definição:

- A seguir será realizada uma breve introdução às funções de janela através de exemplos com as bases de dados “PesqCompras.db” e “TopicosAdicionais.db”. Porém o tema é bastante abrangente e, para detalhes completos, você poderá consultar o manual do SQLite https://www.sqlite.org/windowfunctions.html ou tutoriais disponibilizados na Internet. Nos exemplos apresentados, as funções de janela aparecerão na cláusula SELECT.

Computando Agregações com OVER e PARTITION BY

- Exemplo 1: OVER + PARTITION BY: Lista o id da família, o nome do produto comprado e o total de produtos que a família comprou na base de dados "PesqCompras.db".

```
SELECT
c.id_familia,
p.nom_prod,
COUNT(*) OVER (PARTITION BY c.id_familia) AS tot_produtos_comprados
FROM Compras c INNER JOIN Produto p ON (p.id = c.id_produto);
```

	id_familia	nom_prod	tot_produtos_comprados
1	010001	Kits comida congelada	54
2	010001	Beterraba	54
3	010001	Camarao rosa	54
4	010001	Carne bovina - visceras : Figado	54
5	010001	Carnes Salgadas: Costela	54

Computando Agregações com OVER e PARTITION BY

- **Exemplo 1:**
- Compreendendo o que foi feito:
- O COUNT(*) PARTITION BY (BY c.id_familia) é responsável pelo cálculo do total de produtos comprados por cada família.
- O OVER está trabalhando junto com o PARTITION BY entre parênteses.
- O OVER é utilizado para "mudar o comportamento" do COUNT(*), fazendo com que o GROUP BY não seja mais exigido.

```
COUNT(*) OVER (PARTITION BY c.id_familia) AS tot_produtos_comprados
```

2 Computando Agregações com OVER e PARTITION BY

- **Exemplo 1:**
- Por sua vez, é o PARTITION BY que fará o papel similar ao do GROUP BY. Neste exemplo, ele serve para determinar que o "id_familia" define as partições dos resultados. Uma partição consiste em todas as linhas que possuam o mesmo valor para todas as colunas indicadas no PARTITION BY.

```
SELECT
c.id_familia,
p.nom_prod,
COUNT(*) OVER (PARTITION BY c.id_familia) AS tot_produtos_comprados
FROM Compras c INNER JOIN Produto p ON (p.id = c.id_produto);
```

2 Computando Agregações com OVER e PARTITION BY

- **Exemplo 1:**
- Os valores dos campos id_família e nom_prod são exibidos linha a linha, pois não fazem parte do agrupamento e o resultado da função de grupo (COUNT) retorna o total de ocorrências de cada id_família obtido pela junção entre a tabela Compras e Produto.

```
SELECT
c.id_familia,
p.nom_prod,
COUNT(*) OVER (PARTITION BY c.id_familia) AS tot_produtos_comprados
FROM Compras c INNER JOIN Produto p ON (p.id = c.id_produto);
```

	id_familia	nom_prod	tot_produtos_comprados
1	010001	Kits comida congelada	54
2	010001	Beterraba	54
3	010001	Camarao rosa	54
4	010001	Carne bovina - visceras : Figado	54
5	010001	Carnes Salgadas: Costela	54

Computando Agregações com OVER e PARTITION BY

- **Exemplo 1.1:**
- Experimente incluir a cláusula DISTINCT SELECT e retirar o nome do produto. O resultado mostra o total de compras de cada família.

```
SELECT DISTINCT
c.id_familia,
COUNT(*) OVER (PARTITION BY c.id_familia) AS tot_produtos_comprados
FROM Compras c INNER JOIN Produto p ON (p.id = c.id_produto);
```

	id_familia	tot_produtos_comprados
1	010001	54
2	010002	34
3	010003	56
4	010004	53

Computando Rankings

- A SQL também disponibiliza funções de janela para computar rankings. São elas: CUME_DIST(), DENSE_RANK(), NTILE(), RANK(), ROW_NUMBER() e PERCENT_RANK().
- Essas funções geram resultados tendo por base um conjunto de linhas relacionada à linha corrente. Elas normalmente baseiam-se no valor ordenado (ORDER BY) de um campo. Vejamos alguns exemplos, desta vez com a tabela T_FILME, da base de dados "TopicosAdicionais.db"

Computando Rankings

- **Exemplo 2:** DENSE_RANK(): computa o ranking de uma linha em um conjunto ordenado de linhas sem deixar "gaps" no caso de empate.

```
SELECT
titulo,
avaliacao,
DENSE_RANK () OVER (ORDER BY avaliacao DESC) avaliacao_ranking
FROM t_filme;
```

	titulo	avaliacao	avaliacao_ranking
1	A Vida é Bela	4.7	1
2	Intocáveis	4.7	1
3	Orgulho e Preconceito	4.6	2
4	O Auto da Compadecida	4.6	2
5	Que Horas Ela Volta?	4.5	3
6	As Pontes de Madison	4.5	3

3 Computando Rankings

- **Exemplo 2:**
  - Nesse exemplo, DENSE_RANK () OVER (ORDER BY avaliacao DESC) serve para produzir o ranking do filme de acordo com a nota da avaliação considerando a ordem descendente. Ou seja: o filme mais bem avaliado receberá o valor de ranking 1, o segundo o valor 2, etc.

DENSE_RANK () OVER (ORDER BY avaliacao DESC)

	titulo	avaliacao	avaliacao_ranking
1	A Vida é Bela	4.7	1
2	Intocáveis	4.7	1
3	Orgulho e Preconceito	4.6	2
4	O Auto da Compadecida	4.6	2
5	Que Horas Ela Volta?	4.5	3
6	As Pontes de Madison	4.5	3

Computando Rankings

- **Exemplo 3:**
  - RANK(): é parecido ao DENSE_RANK(), mas deixa "gaps". RANK não retorna um valor consecutivo no caso de haver o mesmo valor. DENSE_RANK() retorna um valor consecutivo sempre.

	titulo	avaliacao	avaliacao_ranking
1	A Vida é Bela	4.7	1
2	Intocáveis	4.7	1
3	Orgulho e Preconceito	4.6	3
4	O Auto da Compadecida	4.6	3
5	Que Horas Ela Volta?	4.5	5
6	As Pontes de Madison	4.5	5

Computando Rankings

- **Exemplo 4:**
  - **ROW_NUMBER()**: Serve para associar um inteiro sequencial a cada linha.

```
SELECT titulo, ano,
ROW_NUMBER() OVER (ORDER BY ano) numero, pais
FROM t_filme
```

	titulo	ano	numero	pais
1	... E o Vento Levou	1939	1	US
2	Cidadão Kane	1941	2	US
3	Casablanca	1942	3	US
4	Jane Eyre	1943	4	US
5	Os Incompreendidos	1959	5	FR
6	Julieta dos Espíritos	1965	6	IT

Computando Rankings

- **Exemplo 4:**
  - **ROW_NUMBER()**: É útil em situações que seja necessário retornar uma instância de uma tabela que ocupa uma determinada posição. Por exemplo, desejamos retornar o ranking dos filmes que estão entre a décima linha e a decima quinta linha.

```
SELECT * FROM  (SELECT
titulo,
avaliacao,
DENSE_RANK () OVER (ORDER BY avaliacao DESC) avaliacao_ranking,
ROW_NUMBER() OVER (ORDER BY ano) numero
FROM t_filme
 ) WHERE numero between 10 and 15;
```

Computando Rankings

- **Exemplo 5:**
- RANK() + PARTITION BY + ORDER BY: ao incluir o PARTITION BY, torna-se possível gerar um ranking por janela. No exemplo a seguir, geramos o ranking do filme por país, de acordo com a sua avaliação.

```
SELECT
titulo,
avaliacao,
pais,
RANK () OVER (PARTITION BY pais ORDER BY avaliacao DESC) AS rank_pais
FROM t_filme;
```

Computando Rankings

- **Exemplo 4:**

```
SELECT * FROM  (SELECT
titulo,
avaliacao,
DENSE_RANK () OVER (ORDER BY avaliacao DESC) avaliacao_ranking,
ROW_NUMBER() OVER (ORDER BY ano) numero
FROM t_filme
 ) WHERE numero between 10 and 15;
```

	titulo	avaliacao	avaliacao_ranking	numero
1	Hair	4.1	7	14
2	Ensina-me a Viver	3.9	9	10
3	Monty Python em Busca do Cálice Sagrado	3.9	9	12
4	O Discreto Charme da Burguesia	3.5	13	11
5	Bye Bye Brasil	3.5	13	15
6	Se Segura, Malandro!	3.1	17	13

Computando Rankings

- **Exemplo 5:**

```
SELECT
titulo,
avaliacao,
pais,
RANK () OVER (PARTITION BY pais ORDER BY avaliacao DESC) AS rank_pais
FROM t_filme;
```

	titulo	avaliacao	pais	rank_pais
1	Um Conto Chinês	4.3	AR	1
2	O Filho da Noiva	4.2	AR	2
3	O Cidadão Ilustre	4	AR	3
4	Nove Rainhas	4	AR	3
5	O Mesmo Amor, a Mesma Chuva	3.5	AR	5
6	Dois Dias, uma Noite	3.7	BE	1
7	O Auto da Compadecida	4.6	BR	1

Computando Rankings

- **Exemplo 5:**
- Veja que nesse exemplo, os rankings são produzidos por país – eles são reiniciado cada vez que ocorre uma troca de país.

```
SELECT titulo, avaliacao,pais,
RANK () OVER (PARTITION BY pais ORDER BY avaliacao DESC) AS rank_pais
FROM t_filme;
```

	titulo	avaliacao	pais	rank_pais
1	Um Conto Chinês	4.3	AR	1
2	O Filho da Noiva	4.2	AR	2
3	O Cidadão Ilustre	4	AR	3
4	Nove Rainhas	4	AR	3
5	O Mesmo Amor, a Mesma Chuva	3.5	AR	5
6	Dois Dias, uma Noite	3.7	BE	1
7	O Auto da Compadecida	4.6	BR	1
8	Que Horas Ela Volta?	4.5	BR	2
9	Cidade de Deus	4.5	BR	2
10	Cássia Eller	4.4	BR	4

Outras Funções de Janela

- As funções FIRST_VALUE(), LAST_VALUE(), LAG(), LEAD(), NTH_VALUE() e NTILE() servem para gerar estatísticas básicas e computar outros tipos de resultados. Vamos conhecê-las.

Outras Funções de Janela

- **Exemplo 6:**
- LAG(): fornece acesso aos dados de uma linha anterior à linha corrente.

```
SELECT
titulo,
ano,
pais,
LAG ( ano, 1, NULL ) OVER (ORDER BY ano ) AS anoAnterior
FROM t_filme ;
```

	titulo	ano	pais	anoAnterior
1	... E o Vento Levou	1939	US	*NULL*
2	Cidadão Kane	1941	US	1939
3	Casablanca	1942	US	1941
4	Jane Eyre	1943	US	1942
5	Os Incompreendidos	1959	FR	1943
6	Julieta dos Espíritos	1965	IT	1959
7	Fahrenheit 451	1966	FR	1965

Neste exemplo, LAG (ano, 1, NULL) OVER (ORDER BY ano) AS anoAnterior, serve para recuperar o valor do ano que aparece 1 linha antes da linha corrente, de acordo com a ordenação por ano. O valor do segundo parâmetro, NULL, é usado para colocar o anoAnterior do filme mais antigo (“... E o Vento Levou”) como NULL, já que não existe nenhuma linha anterior à deste filme.

Outras Funções de Janela

- **Exemplo 6.1:**
- A Função LAG pode ser útil em situações que seja preciso fazer um avaliação comparative entre linhas diferentes. Nesta query é feita uma avaliação entre o interval de tempo de um filme e o filme seguinte em cada país.

```
SELECT pais, titulo, ano,
IIF((LAG ( ano, 1, null ) OVER (ORDER BY pais asc, ANO desc)- ano)>=0,
   (LAG ( ano, 1, null ) OVER (ORDER BY pais asc, ANO desc)- ano),NULL) AS Intervalo
FROM t_filme ;
```

	pais	titulo	ano	Intervalo
1	AR	O Cidadão Ilustre	2017	*NULL*
2	AR	Um Conto Chinês	2011	6
3	AR	O Filho da Noiva	2001	10
4	AR	Nove Rainhas	2000	1
5	AR	O Mesmo Amor, a Mesma Chuva	1999	1
6	BE	Dois Dias, uma Noite	2015	*NULL*
7	BR	Bacurau	2019	*NULL*
8	BR	Que Horas Ela Volta?	2015	4
9	BR	Nise - O Coração da Loucura	2015	0
10	BR	Elis	2015	0
11	BR	Cássia Eller	2014	1

Outras Funções de Janela

- **Exemplo 6.1:**
- Este exemplo instroduzui a função IIF na query. A função IIF (X, Y, Z) retorna o valor Y se X for verdade e Z caso falso.
- Os bancos de dados costumam incluir funções neste estilo para ajudar na montage dos resultados.

```
SELECT pais, titulo, ano,
IIF((LAG ( ano, 1, null ) OVER (ORDER BY pais asc, ANO desc)- ano)>=0,
   (LAG ( ano, 1, null ) OVER (ORDER BY pais asc, ANO desc)- ano),NULL) AS Intervalo
FROM t_filme ;
```

Outras Funções de Janela

- **Exemplo 7:**
- LAG() + PARTITION BY: como sempre, podemos usar o PARTITION BY para determinar uma partição. No exemplo abaixo, usamos "pais" como partição para "resetar" o valor do campo gerado pela função LAG() a cada mudança de país.

```
SELECT titulo, ano, pais,
LAG ( ano, 1, NULL ) OVER (PARTITION BY pais ORDER BY ano ) AS anoAnterior
FROM t_filme ;
```

	titulo	ano	pais	anoAnterior
1	O Mesmo Amor, a Mesma Chuva	1999	AR	*NULL*
2	Nove Rainhas	2000	AR	1999
3	O Filho da Noiva	2001	AR	2000
4	Um Conto Chinês	2011	AR	2001
5	O Cidadão Ilustre	2017	AR	2011
6	Dois Dias, uma Noite	2015	BE	*NULL*
7	Macunaíma	1969	BR	*NULL*
8	Se Segura, Malandro!	1978	BR	1969
9	Bye Bye Brasil	1979	BR	1978
10	Eles Não Usam Black-Tie	1981	BR	1979

Outras Funções de Janela

- **Exemplo 8:**
- **FIRST_VALUE()** e LAST_VALUE(): retornam, respectivamente, o primeiro e último valor de uma determinada janela especificada.
- SELECT titulo, ano, pais,
- FIRST_VALUE(titulo) OVER (PARTITION BY pais ORDER BY ano) AS primeiro_filme
- FROM t_filme;

	titulo	ano	pais	primeiro_filme
1	O Mesmo Amor, a Mesma Chuva	1999	AR	O Mesmo Amor, a Mesma Chuva
2	Nove Rainhas	2000	AR	O Mesmo Amor, a Mesma Chuva
3	O Filho da Noiva	2001	AR	O Mesmo Amor, a Mesma Chuva
4	Um Conto Chinês	2011	AR	O Mesmo Amor, a Mesma Chuva
5	O Cidadão Ilustre	2017	AR	O Mesmo Amor, a Mesma Chuva
6	Dois Dias, uma Noite	2015	BE	Dois Dias, uma Noite
7	Macunaíma	1969	BR	Macunaíma
8	Se Segura, Malandro!	1978	BR	Macunaíma
9	Bye Bye Brasil	1979	BR	Macunaíma

Outras Funções de Janela

- **Exemplo 8: FIRST_VALUE()**
- Neste exemplo, as janelas (partições) são formadas pelos países e a ordenação é feita pelo ano. Logo, o primeiro valor da janela (FIRST_VALUE) será sempre o filme mais antigo de cada país.
- FIRST_VALUE(titulo) OVER (PARTITION BY pais ORDER BY ano) AS primeiro_filme

	titulo	ano	pais	primeiro_filme
1	O Mesmo Amor, a Mesma Chuva	1999	AR	O Mesmo Amor, a Mesma Chuva
2	Nove Rainhas	2000	AR	O Mesmo Amor, a Mesma Chuva
3	O Filho da Noiva	2001	AR	O Mesmo Amor, a Mesma Chuva
4	Um Conto Chinês	2011	AR	O Mesmo Amor, a Mesma Chuva
5	O Cidadão Ilustre	2017	AR	O Mesmo Amor, a Mesma Chuva
6	Dois Dias, uma Noite	2015	BE	Dois Dias, uma Noite
7	Macunaíma	1969	BR	Macunaíma
8	Se Segura, Malandro!	1978	BR	Macunaíma
9	Bye Bye Brasil	1979	BR	Macunaíma

Obrigado