Aggregate window functions

POSTGRESQL SUMMARY STATS AND WINDOW FUNCTIONS

Michel Semaan
Data Scientist

Source table

Query

```
SELECT
  Year, COUNT(*) AS Medals
FROM Summer_Medals
WHERE
  Country = 'BRA'
  AND Medal = 'Gold'
  AND Year >= 1992
GROUP BY Year
ORDER BY Year ASC;
```

Result

```
| Year | Medals |
|------|--------|
| 1992 | 13     |
| 1996 | 5      |
| 2004 | 18     |
| 2008 | 14     |
| 2012 | 14     |
```

Aggregate functions

MAX Query

```
WITH Brazil_Medals AS (...)

SELECT MAX(Medals) AS Max_Medals
FROM Brazil_Medals;
```

MAX Result

```
18
```

SUM Query

```
WITH Brazil_Medals AS (...)

SELECT SUM(Medals) AS Total_Medals
FROM Brazil_Medals;
```

SUM Result

```
64
```

MAX Window function

Query

```
WITH Brazil_Medals AS (...)

SELECT
  Year, Medals,
  MAX(Medals)
    OVER (ORDER BY Year ASC) AS Max_Medals
FROM Brazil_Medals;
```

Result

```
| Year | Medals | Max_Medals |
|------|--------|------------|
| 1992 | 13     | 13         |
| 1996 | 5      | 13         |
| 2004 | 18     | 18         |
| 2008 | 14     | 18         |
| 2012 | 14     | 18         |
```

SUM Window function

Query

```
WITH Brazil_Medals AS (...)

SELECT
  Year, Medals,
  SUM(Medals) OVER (ORDER BY Year ASC) AS Medals_RT
FROM Brazil_Medals;
```

Result

```
| Year | Medals | Medals_RT |
|------|--------|-----------|
| 1992 | 13     | 13        |
| 1996 | 5      | 18        |
| 2004 | 18     | 36        |
| 2008 | 14     | 50        |
| 2012 | 14     | 64        |
```

Partitioning with aggregate window functions

Query

```
WITH Medals AS (...)
SELECT Year, Country, Medals,
  SUM(Meals) OVER (...)
FROM Medals;
```

Result

Year	Country	Medals	Medals_RT
2004	BRA	18	18
2008	BRA	14	32
2012	BRA	14	46
2004	CUB	31	77
2008	CUB	2	79
2012	CUB	5	84

Query

```
WITH Medals AS (...)
SELECT Year, Country, Medals,
  SUM(Meals) OVER (PARTITION BY Country ...)
FROM Medals;
```

Result

Year	Country	Medals	Medals_RT
2004	BRA	18	18
2008	BRA	14	32
2012	BRA	14	46
2004	CUB	31	31
2008	CUB	2	33
2012	CUB	5	38

Let's practice!

POSTGRESQL SUMMARY STATS AND WINDOW FUNCTIONS

Frames

POSTGRESQL SUMMARY STATS AND WINDOW FUNCTIONS

Michel Semaan
Data Scientist

Motivation

`LAST_VALUE`

```
LAST_VALUE(City) OVER (
 ORDER BY Year ASC
 RANGE BETWEEN
   UNBOUNDED PRECEDING AND
   UNBOUNDED FOLLOWING
) AS Last_City
```

- Frame: `RANGE BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING`
- Without the frame, `LAST_VALUE` would return the row's value in the City column
- By default, a frame starts at the beginning of a table or partition and ends at the current row

ROWS BETWEEN

- `ROWS BETWEEN [START] AND [FINISH]`
  - `n PRECEDING` : `n` rows before the current row
  - `CURRENT ROW` : the current row
  - `n FOLLOWING` : `n` rows after the current row

Examples

- `ROWS BETWEEN 3 PRECEDING AND CURRENT ROW`
- `ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING`
- `ROWS BETWEEN 5 PRECEDING AND 1 PRECEDING`

Source table

Query

```
SELECT
  Year, COUNT(*) AS Medals
FROM Summer_Medals
WHERE
  Country = 'RUS'
  AND Medal = 'Gold'
GROUP BY Year
ORDER BY Year ASC;
```

Result

```
| Year | Medals |
|------|--------|
| 1996 | 36     |
| 2000 | 66     |
| 2004 | 47     |
| 2008 | 43     |
| 2012 | 47     |
```

MAX without a frame

Query

```
WITH Russia_Medals AS (...)

SELECT
  Year, Medals,
  MAX(Medals)
    OVER (ORDER BY Year ASC) AS Max_Medals
FROM Russia_Medals
ORDER BY Year ASC;
```

Result

Year	Medals	Max_Medals
1996	36	36
2000	66	66
2004	47	66
2008	43	66
2012	47	66

MAX with a frame

Query

```
WITH Russia_Medals AS (...)

SELECT
  Year, Medals,
  MAX(Medals)
    OVER (ORDER BY Year ASC) AS Max_Medals,
  MAX(Medals)
    OVER (ORDER BY Year ASC
          ROWS BETWEEN
          1 PRECEDING AND CURRENT ROW)
    AS Max_Medals_Last
FROM Russia_Medals
ORDER BY Year ASC;
```

Result

```
| Year | Medals | Max_Medals | Max_Medals_Last |
|------|--------|------------|-----------------|
| 1996 | 36     | 36         | 36              |
| 2000 | 66     | 66         | 66              |
| 2004 | 47     | 66         | 66              |
| 2008 | 43     | 66         | 47              |
| 2012 | 47     | 66         | 47              |
```

Current and following rows

Query

```
WITH Russia_Medals AS (...)

SELECT
  Year, Medals,
  MAX(Medals)
    OVER (ORDER BY Year ASC
          ROWS BETWEEN
          CURRENT ROW AND 1 FOLLOWING)
    AS Max_Medals_Next
FROM Russia_Medals
ORDER BY Year ASC;
```

Result

Year	Medals	Max_Medals_Next
1996	36	66
2000	66	66
2004	47	47
2008	43	47
2012	47	47

Let's practice!

POSTGRESQL SUMMARY STATS AND WINDOW FUNCTIONS

Moving averages and totals

POSTGRESQL SUMMARY STATS AND WINDOW FUNCTIONS

Michel Semaan

Moving averages

Overview

- Moving average (MA): Average of last `n` periods
  - **Example**: 10-day MA of units sold in sales is the average of the last 10 days' sold units
  - Used to indicate momentum/trends
  - Also useful in eliminating seasonality
- Moving total: Sum of last `n` periods
  - **Example**: Sum of the last 3 Olympic games' medals
  - Used to indicate performance; if the sum is going down, overall performance is going down

Source table

Query

```
SELECT
  Year, COUNT(*) AS Medals
FROM Summer_Medals
WHERE
  Country = 'USA'
  AND Medal = 'Gold'
  AND Year >= 1980
GROUP BY Year
ORDER BY Year ASC;
```

Result

```
| Year | Medals |
|------|--------|
| 1984 | 168    |
| 1988 | 77     |
| 1992 | 89     |
| 1996 | 160    |
| 2000 | 130    |
| 2004 | 116    |
| 2008 | 125    |
| 2012 | 147    |
```

Moving average

Query

```
WITH US_Medals AS (...)

SELECT
  Year, Medals,
  AVG(Medals) OVER
    (ORDER BY Year ASC
     ROWS BETWEEN
     2 PRECEDING AND CURRENT ROW) AS Medals_MA
FROM US_Medals
ORDER BY Year ASC;
```

Result

Year	Medals	Medals_MA
1984	168	168.00
1988	77	122.50
1992	89	111.33
1996	160	108.67
2000	130	126.33
2004	116	135.33
2008	125	123.67
2012	147	129.33

Moving total

Query

```
WITH US_Medals AS (...)

SELECT
  Year, Medals,
  SUM(Medals) OVER
    (ORDER BY Year ASC
     ROWS BETWEEN
     2 PRECEDING AND CURRENT ROW) AS Medals_MT
FROM US_Medals
ORDER BY Year ASC;
```

Result

Year	Medals	Medals_MT
1984	168	168
1988	77	245
1992	89	334
1996	160	326
2000	130	379
2004	116	406
2008	125	371
2012	147	388

ROWS vs RANGE

- `RANGE BETWEEN [START] AND [FINISH]`
  - Functions much the same as `ROWS BETWEEN`
  - `RANGE` treats duplicates in `OVER` 's `ORDER BY` subclause as a single entity

Table

```
| Year | Medals | Rows_RT | Range_RT |
|------|--------|---------|----------|
| 1992 | 10     | 10      | 10       |
| 1996 | 50     | 60      | 110      |
| 2000 | 50     | 110     | 110      |
| 2004 | 60     | 170     | 230      |
| 2008 | 60     | 230     | 230      |
| 2012 | 70     | 300     | 300      |
```

- `ROWS BETWEEN` is almost always used over `RANGE BETWEEN`

Let's practice!

POSTGRESQL SUMMARY STATS AND WINDOW FUNCTIONS