SQL for PostgreSQL

황종필

Table Re-create

- Customers
- Orders

Table Script

Windows Functions

- WINDOW_FUNCTION([arguments])
  OVER (PARTITION BY … [ORDER BY …])

1. 그룹 내 순서 획득
   - RANK, DENSE_RANK, ROW_NUMBER
2. 그룹 내 특정위치 컬럼 획득
   - FIRST_VALUE, LAST_VALUE, LAG, LEAD, NTH_VALUE
3. 그룹 내 집계정보 획득
   - SUM, MAX, MIN, AVG, COUNT
4. 기타
   - CUME_DIST, NTILE

Windows Functions - 1(그룹 내 순서 획득)

```
select userid
     , join_date
     , rank()         over (order by join_date)         as rank
     , dense_rank()   over (order by join_date)         as dense_rank
     , row_number()   over (order by join_date)         as row_number
     , percent_rank() over (order by join_date)         as percent_rank1
     , percent_rank() over (order by join_date, userid) as percent_rank2
  from customer
 order by join_date
        , userid
```

userid	join_date	rank	dense_rank	row_number	percent_rank1	percent_rank2
1	2015-08-01	1	1	1	0	0
3	2015-08-01	1	1	2	0	0.2
2	2015-08-02	3	2	3	0.4	0.4
4	2015-08-03	4	3	4	0.6	0.6
5	2015-08-07	5	4	5	0.8	0.8
6	2015-08-22	6	5	6	1	1

Windows Functions - 2(그룹 내 특정위치 컬럼 획득)

```
select userid
     , order_date
     , method
     , first_value(method)      over (partition by userid                          ) as first_value
     , first_value(method)      over (partition by userid order by order_date      ) as first_value_order
     , last_value (method)      over (partition by userid                          ) as last_value
     , last_value (method)      over (partition by userid order by order_date      ) as last_value_order
     , lag (order_date, 1)      over (partition by userid order by order_date      ) as lag_value_asc
     , lag (order_date, 1)      over (partition by userid order by order_date desc) as lag_value_desc
     , lead(order_date, 1)      over (partition by userid order by order_date      ) as lead_value_asc
     , lead(order_date, 1)      over (partition by userid order by order_date desc) as lead_value_desc
     , nth_value(order_date, 2) over (partition by userid order by order_date      ) as nth_value_asc
     , nth_value(order_date, 2) over (partition by userid order by order_date desc) as nth_value_desc
  from orders
 where userid in (1, 3)
 order by userid
        , order_date
```

userid	order_date	method	first_value	first_value_order	last_value	last_value_order	lag_value_asc	lag_value_desc	lead_value_asc	lead_value_desc	nth_value_asc	nth_value_desc
1	2015-08-01	CALL	CALL	CALL	TOUCH	CALL		2015-08-03	2015-08-03			2015-08-14
1	2015-08-03	TOUCH	CALL	CALL	TOUCH	TOUCH	2015-08-01	2015-08-10	2015-08-10	2015-08-01	2015-08-03	2015-08-14
1	2015-08-10	TOUCH	CALL	CALL	TOUCH	TOUCH	2015-08-03	2015-08-14	2015-08-14	2015-08-03	2015-08-03	2015-08-14
1	2015-08-14	CALL	CALL	CALL	TOUCH	CALL	2015-08-10	2015-08-25	2015-08-25	2015-08-10	2015-08-03	2015-08-14
1	2015-08-25	TOUCH	CALL	CALL	TOUCH	TOUCH	2015-08-14			2015-08-14	2015-08-03	
3	2015-08-07	CALL	CALL	CALL	CALL	CALL		2015-08-19	2015-08-19			2015-08-19
3	2015-08-19	TOUCH	CALL	CALL	CALL	TOUCH	2015-08-07	2015-08-30	2015-08-30	2015-08-07	2015-08-19	2015-08-19
3	2015-08-30	CALL	CALL	CALL	CALL	CALL	2015-08-19			2015-08-19	2015-08-19	

Windows Functions - 3(그룹 내 집계정보 획득)

```
select userid
     , amount
     , sum  (amount) over (partition by userid                    ) as sum_over
     , sum  (amount) over (partition by userid order by order_date) as sum_over_order
     , max  (amount) over (partition by userid                    ) as max_over
     , max  (amount) over (partition by userid order by order_date) as max_over_order
     , min  (amount) over (partition by userid                    ) as min_over
     , min  (amount) over (partition by userid order by order_date) as min_over_order
     , avg  (amount) over (partition by userid                    ) as avg_over
     , avg  (amount) over (partition by userid order by order_date) as avg_over_order
     , count(amount) over (partition by userid                    ) as count_over
     , count(amount) over (partition by userid order by order_date) as count_over_order
  from orders
 where userid in (3, 4, 5)
```

userid	amount	sum_over	sum_over_order	max_over	max_over_order	min_over	min_over_order	avg_over	avg_over_order	count_over	count_over_order
3	10000	30000	10000	10000	10000	10000	10000	10000	10000	3	1
3	10000	30000	20000	10000	10000	10000	10000	10000	10000	3	2
3	10000	30000	30000	10000	10000	10000	10000	10000	10000	3	3
4	20000	50000	20000	30000	20000	20000	20000	25000	20000	2	1
4	30000	50000	50000	30000	30000	20000	20000	25000	25000	2	2
5	10000	75000	10000	30000	10000	5000	10000	15000	10000	5	1
5	20000	75000	30000	30000	20000	5000	10000	15000	15000	5	2
5	5000	75000	35000	30000	20000	5000	5000	15000	11667	5	3
5	10000	75000	45000	30000	20000	5000	5000	15000	11250	5	4
5	30000	75000	75000	30000	30000	5000	5000	15000	15000	5	5

Windows Functions - 4(기타)

```
select userid
     , join_date
     , cume_dist() over (order by join_date)         as cum_dist1
     , cume_dist() over (order by join_date, userid) as cum_dist2
     , ntile(3)    over (order by join_date)         as ntile3
     , ntile(4)    over (order by join_date)         as ntile4
  from customer
 order by join_date
        , userid
```

userid	join_date	cum_dist1	cum_dist2	ntile3	ntile4
1	2015-08-01	0.333333333	0.166666667	1	1
3	2015-08-01	0.333333333	0.333333333	1	1
2	2015-08-02	0.5	0.5	2	2
4	2015-08-03	0.666666667	0.666666667	2	2
5	2015-08-07	0.833333333	0.833333333	3	3
6	2015-08-22	1	1	3	4