SQL for PostgreSQL

황종필

예시 데이터(테이블)

Table : Order

USREID	ORDER_DATE	METHOD	AMOUNT	DISCOUNT
INTEGER	VARCHAR(10)	VARCHAR(10)	INTEGER	NUMERIC
1	2015-08-01	CALL	10,000	-998.7
1	2015-08-03	TOUCH	10,000	
1	2015-08-10	TOUCH	10,000	-950.4
1	2015-08-14	CALL	10,000	-1,000.0
1	2015-08-25	TOUCH	10,000	
2	2015-08-03	TOUCH	5,000	-500.0
2	2015-08-11	TOUCH	5,000	-300.0
2	2015-08-12	TOUCH	5,000	-700.0
2	2015-08-22	TOUCH	5,000	-1,000.0
2	2015-08-28	TOUCH	5,000	-600.0
3	2015-08-07	CALL	10,000	-1,000.0
3	2015-08-19	TOUCH	10,000	-1,000.0
3	2015-08-30	CALL	10,000	-1,000.0
4	2015-08-05	CALL	20,000	-3,000.0
4	2015-08-18	TOUCH	30,000	-5,000.0
5	2015-08-15	CALL	10,000	-1,000.0
5	2015-08-17	CALL	10,000	
5	2015-08-21	CALL	10,000	-1,000.0
5	2015-08-23	CALL	10,000	-1,000.0
5	2015-08-29	CALL	10,000	-1,000.0

Data Script

SELECT

ALL / DISTINCT	중복 표현 / 중복 제거	select [all] order_date from “order” select distinct order_date from “order”
*	전체 열 선택	select * from “order”
ALIAS	컬럼명 재정의	select distinct order_date [as] ord_dt from “order”

Top N Query

LIMIT	출력행 수 제한	select * from “order” order by amount desc limit 5

WHERE 조건절 - 1

- OPERATOR
  - 비교연산자
  - SQL연산자
  - 논리연산자
  - 부정비교연산자
  - 부정SQL연산자
- BUILT-IN FUNCTION(PostgreSQL 기준)
  - 문자형 함수
  - 숫자형 함수
  - 날짜형 함수
  - NULL 관련 함수

WHERE 조건절 - 2

- CAST Expression - 형변환(Data Type Convert)
- CASE Expression - 값변환(Value Change)

OPERATOR(WHERE 조건절) - 1

- 비교 연산자

=	같다.	where order_date = '2015-08-05'
>	보다 크다.	where order_date > '2015-08-05'
>=	보다 크거나 같다.	where order_date >= '2015-08-05'
<	보다 작다.	where order_date < '2015-08-05'
<=	보다 작거나 같다.	where order_date <= '2015-08-05'

OPERATOR(WHERE 조건절) - 2

- SQL 연산자

BETWEEN a AND b	a와 b의 값 사이에 있으면 된다. (a와 b 값이 포함됨)	where order_date between ‘2015-08-05’ and ‘2015-08-20’
IN (list)	리스트에 있는 값 중에서 어느 하나라도 일치 하면 된다.	where order_date in (‘2015-08-05’, ‘2015-08-07’)
LIKE ‘비교문자열’	비교문자열과 형태가 일치하면 된다.(%, _ 사용)	where order_date like ‘2015%2%’ where order_date like ‘2015-08-1_’
IS NULL	NULL 값인 경우 (NULL은 특별 취급)	where discount is null

OPERATOR(WHERE 조건절) - 3

- 논리 연산자

AND	앞에 있는 조건과 뒤에 오는 조건이 참(TRUE)이 되면 결과도 참(TRUE)이 된다. 즉, 앞의 조건과 뒤의 조건을 동시에 만족해야 한다.	where order_date = ‘ 2015-08-12 ’ and userid = 2
OR	앞의 조건이 참(TRUE)이거나 뒤의 조건이 참(TRUE)이 되어야 결과도 참(TRUE)이 된다. 즉, 앞 뒤의 조건 중 하나만 참(TRUE)이면 된다.	where order_date = ‘ 2015-08-12 ’ or userid = 2
NOT	뒤에 오는 조건에 반대되는 결과를 되돌려 준다.	where not order_date = ‘ 2015-08-12 ’ and userid = 2

OPERATOR(WHERE 조건절) - 4

- 부정 비교 연산자

!=	같지 않다.	where order_date != ‘ 2015-08-12 ’ and userid = 2
^=	같지 않다.(동작X)	where order_date ^= ‘ 2015-08-12 ’ and userid = 2
<>	같지 않다.(ISO 표준)	where order_date <> ‘ 2015-08-12 ’ and userid = 2
NOT 컬럼명 =	~와 같지 않다.	where not order_date = ‘ 2015-08-12 ’ and userid = 2
NOT 컬럼명 >	~보다 크지 않다.	where not order_date > ‘ 2015-08-12 ’ and userid = 2

OPERATOR(WHERE 조건절) - 5

- 부정 SQL 연산자

NOT BETWEEN a AND b	a와 b의 값 사이에 있지 않다. (a, b 값을 포함하지 않는다.)	where order_date not between ‘2015-08-05’ and ‘2015-08-20’
NOT IN (list)	list 값과 일치하지 않는다.	where order_date not in (‘2015-08-15’, ‘2015-08-23’) and userid = 5
IS NOT NULL	NULL 값을 갖지 않는다.	where order_date is not null

OPERATOR(WHERE 조건절) - 6

- 연산자의 우선순위

연산 우선순위	설명
1	괄호 ()
2	NOT 연산자
3	비교 연산자, SQL 비교 연산자
4	AND
5	OR

Build-In Function(WHERE 조건절) - 1

- 문자형 함수

concat(substring, substring) [or ||]	두 개의 문자열을 병합
concat_ws(sep text, substing1, substing2, ...)	여러 개의 문자열 사이에 구분자를 추가하여 병합
substring(string [from int] [for int])	문자열의 n번째(from)자리부터 m개(for)의 부분문자열을 반환
position(substring in string)	문자열에 특정부분문자열의 위치를 반환
replace(string, from text, to text)	부분문자열(from)을 다른 부분문자열(to)로 치환
legnth(string)	문자열의 길이 반환
trim(string)	문자열의 앞뒤 공백을 제거
upper(string)	문자열을 대문자로 치환
lower(string)	문자열을 소문자로 치환

Build-In Function(WHERE 조건절) - 2

- 문자형 함수

left(str text, n int)	좌측에서 n번째까지의 문자열 반환
right (str text, n int)	우측에서 n번째까지의 문자열 반환
lpad(string text, length int [, fill text])	기존문자열에 특정문자열(fill)을 좌측에 추가하여 특정 길이(length)의 문자열 반환
rpad(string text, length int [, fill text])	기존문자열에 특정문자열(fill)을 우측에 추가하여 특정 길이(length)의 문자열 반환

Build-In Function(WHERE 조건절) - 3

```
select userid
     , order_date
     , concat('date : ', order_date)                 as concat1
     , 'date : ' || order_date                       as concat2
     , concat_ws(' / ', 'start', order_date, 'end') as concat2
     , substring(order_date from 1 for 4)            as substring1
     , substring(order_date from 6 for 2)            as substring2
     , position('-' in order_date)                   as position
     , replace(order_date, '-', '/')                 as replace
     , length(order_date)                            as length
     , trim('  date   ')                             as trim
     , upper('date')                                 as upper
     , lower('DATE')                                 as lower
     , left(order_date, 4)                           as left
     , right(order_date, 2)                          as right
     , lpad(cast(userid as varchar), 10, '0')        as lapd
     , rpad(cast(userid as varchar), 10, '0')        as rpad
  from "order"
 limit 5
```

userid integer	order_date character varying(10)	concat1 text	concat2 text	concat2 text	substring1 text	substring2 text	position integer	replace text	length integer	trim text	upper text	lower text	left text	right text	lapd text	rpad text
1	2015-08-01	date : 2015-08-01	date : 2015-08-01	start / 2015-08-01 / end	2015	08	5	2015/08/01	10	date	DATE	date	2015	01	0000000001	1000000000
1	2015-08-03	date : 2015-08-03	date : 2015-08-03	start / 2015-08-03 / end	2015	08	5	2015/08/03	10	date	DATE	date	2015	03	0000000001	1000000000
1	2015-08-10	date : 2015-08-10	date : 2015-08-10	start / 2015-08-10 / end	2015	08	5	2015/08/10	10	date	DATE	date	2015	10	0000000001	1000000000
1	2015-08-14	date : 2015-08-14	date : 2015-08-14	start / 2015-08-14 / end	2015	08	5	2015/08/14	10	date	DATE	date	2015	14	0000000001	1000000000
1	2015-08-25	date : 2015-08-25	date : 2015-08-25	start / 2015-08-25 / end	2015	08	5	2015/08/25	10	date	DATE	date	2015	25	0000000001	1000000000

Build-In Function(WHERE 조건절) - 4

- 숫자형 함수

abs(x)	절대값
ceil(numeric), ceiling(numeric)	소수점 올림
floor(numeric)	소수점 내림
round(v numeric, s int)	v를 소수점 s번째에서 반올림
trunc(v numeric, s int)	v를 소수점 s번째에서 잘라버림
mod(y, x)	y에서 x를 나눈 나머지
exp(n numeric)	e(자연상수)의 n승
power(a numeric, b numeric)	a의 b승
sign(numeric)	양수는 1, 0은 0, 음수는 -1로 치환
sqrt(numeric)	제곱근

Build-In Function(WHERE 조건절) - 5

```
select amount
     , discount
     , abs(discount)      as abs
     , ceil(discount)     as ceil
     , ceiling(discount)  as ceiling
     , floor(discount)    as floor
     , round(discount, 0) as round
     , trunc(discount, 0) as trunc
     , mod(9, 4)          as mod
     , exp(1)             as exp
     , power(3, 2)        as power
     , rk
     , sign(rk)           as sign
     , sqrt(amount)       as sqrt
  from (select ord.*
             , row_number() over (order by userid, order_date) - 3 as rk
          from "order" ord
         limit 5) ord
```

amount integer	discount numeric	abs numeric	ceil numeric	ceiling numeric	floor numeric	round numeric	trunc numeric	mod integer	exp double precision	power double precision	rk bigint	sign double precision	sqrt double precision
10000	-998.7	998.7	-998	-998	-999	-999	-998	1	2.71828182845905	9	-2	-1	100
10000								1	2.71828182845905	9	-1	-1	100
10000	-950.4	950.4	-950	-950	-951	-950	-950	1	2.71828182845905	9	0	0	100
10000	-1000	1000	-1000	-1000	-1000	-1000	-1000	1	2.71828182845905	9	1	1	100
10000								1	2.71828182845905	9	2	1	100

Build-In Function(WHERE 조건절) - 6

- 날짜형 함수

current_date	현재 일자
current_time	현재 시분초
current_timestamp	현재 일시
date_part(text, timestamp[or interval])	일시에서 원하는 부분 추출(년, 월, 일, 시, …)
date_trunc(text, timestamp[or interval])	일시에서 지정한 부분 이하 절사
extract(field from timestamp)	일시에서 원하는 부분 추출(년, 월, 일, 시, …)
now()	현재 일시

Build-In Function(WHERE 조건절) - 7

```
select order_dttm
     , current_date                    as current_date
     , current_time                    as current_time
     , current_timestamp               as current_timestamp
     , date_part('month', order_dttm)  as date_part
     , date_trunc('month', order_dttm) as date_trunc
     , extract(month from order_dttm)  as extract
     , now()                           as now
  from (select ord.*
             , to_date(order_date, 'YYYY-MM-DD') as order_dttm
          from "order" ord
         limit 5) ord
```

order_dttm date	current_date date	current_time time with time zone	current_timestamp timestamp with time zone	date_part double precision	date_trunc timestamp with time zone	extract double precision	now timestamp with time zone
2015-08-01	2015-09-01	12:35:30.951+09	2015-09-01 12:35:30.951+09	8	2015-08-01 00:00:00+09	8	2015-09-01 12:35:30.951+09
2015-08-03	2015-09-01	12:35:30.951+09	2015-09-01 12:35:30.951+09	8	2015-08-01 00:00:00+09	8	2015-09-01 12:35:30.951+09
2015-08-10	2015-09-01	12:35:30.951+09	2015-09-01 12:35:30.951+09	8	2015-08-01 00:00:00+09	8	2015-09-01 12:35:30.951+09
2015-08-14	2015-09-01	12:35:30.951+09	2015-09-01 12:35:30.951+09	8	2015-08-01 00:00:00+09	8	2015-09-01 12:35:30.951+09
2015-08-25	2015-09-01	12:35:30.951+09	2015-09-01 12:35:30.951+09	8	2015-08-01 00:00:00+09	8	2015-09-01 12:35:30.951+09

Build-In Function(WHERE 조건절) - 8

- NULL 관련 함수

coalesce(x, y)	x가 null이면 y로 치환.
nullif(x, y)	x와 y가 같으면 null을 같지 않다면 x를 반환.

- 기타 함수

greatest(x, y, z, …)	나열된 항목 중 가장 큰 값 반환.
least(x, y, z, …)	나열된 항목 중 가장 작은 값 반환.

Build-In Function(WHERE 조건절) - 9

```
select amount
     , discount
     , amount + discount                as discount_amonut
     , amount + coalesce(discount, 0) as coalesce
     , nullif(discount, discount)     as nullif
     , greatest(100, 1, 3, 5, -1)     as greatest
     , least(100, 1, 3, 5, -1)        as least
  from "order"
 limit 5
```

amount integer	discount numeric	discount_amonut numeric	coalesce numeric	nullif numeric	greatest integer	least integer
10000	-998.7	9001.3	9001.3		100	-1
10000			10000		100	-1
10000	-950.4	9049.6	9049.6		100	-1
10000	-1000	9000	9000		100	-1
10000			10000		100	-1

Build-In Function(WHERE 조건절) - 10

- 변환형 함수

to_char(x, text)	숫자형, 일시형 데이터를 문자형 데이터로 변환
to_date(x, text)	문자형 데이터를 일자형 데이터로 변환
to_timestamp(x, text)	문자형 데이터를 일시형 데이터로 변환
to_number(x, text)	문자형 데이터를 숫자형 데이터로 변환
cast(x as y)	x를 y[숫자,문자,일시]형 데이터로 변환
x::y	x를 y[숫자,문자,일시]형 데이터로 변환

Build-In Function(WHERE 조건절) - 11

```
select order_date::timestamp                          as order_dttm
     , to_char(order_date::timestamp, 'YYYY-MM-DD')   as to_char
     , to_date(order_date, 'YYYY-MM-DD')              as to_date
     , to_timestamp(order_date, 'YYYY-MM-DD')         as to_timestamp
     , to_number('19,999.9', '99G999.9')              as to_number
     , cast(current_date as varchar)                  as cast1
     , cast(order_date as timestamp)                  as cast2
     , cast('1999' as integer)                        as cast3
     , cast('1999.9' as numeric)                      as cast4
     , '1999.9'::numeric                              as cast5
  from "order"
 limit 5
```

order_dttm timestamp without time zone	to_char text	to_date date	to_timestamp timestamp with time zone	to_number numeric	cast1 character varying	cast2 timestamp without time zone	cast3 integer	cast4 numeric	cast5 numeric
2015-08-01 00:00:00	2015-08-01	2015-08-01	2015-08-01 00:00:00+09	19999.9	2015-09-01	2015-08-01 00:00:00	1999	1999.9	1999.9
2015-08-03 00:00:00	2015-08-03	2015-08-03	2015-08-03 00:00:00+09	19999.9	2015-09-01	2015-08-03 00:00:00	1999	1999.9	1999.9
2015-08-10 00:00:00	2015-08-10	2015-08-10	2015-08-10 00:00:00+09	19999.9	2015-09-01	2015-08-10 00:00:00	1999	1999.9	1999.9
2015-08-14 00:00:00	2015-08-14	2015-08-14	2015-08-14 00:00:00+09	19999.9	2015-09-01	2015-08-14 00:00:00	1999	1999.9	1999.9
2015-08-25 00:00:00	2015-08-25	2015-08-25	2015-08-25 00:00:00+09	19999.9	2015-09-01	2015-08-25 00:00:00	1999	1999.9	1999.9

CASE Expression(WHERE 조건절) - 1

case simple_case_expression 조건 else 표현절 end	Simple_case_expression 조건이 맞으면 simple_case_expression 조건 내의 then 절을 수행하고, 조건이 맞지 않으면 else 절을 수행한다.
Case searched_case_expression 조건 else 표현절 end	Searched_case_expression 조건이 맞으면 searched_case_expression 조건내의 then 절을 수행하고, 조건이 맞지 않으면 else 절을 수행한다.

CASE Expression(WHERE 조건절)

```
select userid
     , orders
     , gmv
     , case orders
            when 1 then 'A'
            when 2 then 'B'
            when 3 then 'C'
            else        'D'
       end as simple_case
     , case when orders >= 4 and gmv >= 40000 then 'A'
            when orders >= 3 and gmv >= 30000 then 'B'
            when orders >= 2 and gmv >= 10000 then 'C'
            else                                   'D'
       end as searched_case
  from (select userid
             , count(userid) as orders
             , sum(amount)   as gmv
          from "order"
         group by userid) ord
```

userid integer	orders bigint	gmv bigint	simple_case text	searched_case text
4	2	50000	B	C
1	5	50000	D	A
5	5	50000	D	A
3	3	30000	C	B
2	5	25000	D	C