빅쿼리를 활용한 빅데이터 분석 (25)

서진호

자료 다운로드: https://www.github.com/synabreu/BigQuery

제 25 강 목표

1. 윈도우 함수는 무엇인가?
2. 윈도우 함수 종류
3. 집계 분석 함수
4. 탐색 함수
5. 번호 매기기 함수

윈도우 함수(WINDOW FUNCTION)

- 윈도우 함수, 분석 윈도우 함수, 분석 함수라고도 부름.
- 행 그룹에 대해 하나의 결과를 반환하는 집계 함수와는 달리 행 그룹의 값을 계산하고 각 행마다 하나의 결과를 반환함.
- 윈도우 함수는 OVER 절을 포함하고, OVER 절은 평가 중인 행을 중심으로 행의 기간을 정의함.
- 각 행에 대한 윈도우 함수 결과는 선택된 행 윈도우를 입력으로 사용하여 집계 방식으로 계산함.
- 윈도우 함수를 사용하면 이동 평균, 항목의 순위, 누적 합계를 계산하고, 그 외 기타 분석한다.

윈도우 함수 구문

```
function_name ( [ argument_list ] ) OVER over_clause

over_clause:
  { named_window | ( [ window_specification ] ) }

window_specification:
  [ named_window ]
  [ PARTITION BY partition_expression [, ...] ]
  [ ORDER BY expression [ { ASC | DESC } ] [, ...] ]
  [ window_frame_clause ]

window_frame_clause:
  { rows_range } { frame_start | frame_between }

rows_range:
  { ROWS | RANGE }
```

OVER절 정의

```
function_name ( [ argument_list ] ) OVER over_clause

over_clause:
  { named_window | ( [ window_specification ] ) }
```

윈도우 사양 정의

```
window_specification:
   [ named_window ]
   [ PARTITION BY partition_expression [, ...] ]
   [ ORDER BY expression [ { ASC | DESC } ] [, ...] ]
   [ window_frame_clause ]
```

윈도우 범위 설정

```
RANGE BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW
```

```
SELECT book, LAST_VALUE(item)
  OVER (ORDER BY year)
FROM Library
```

```
SELECT book, LAST_VALUE(item)
  OVER (
    ORDER BY year
    RANGE BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW)
FROM Library
```

실습1 – 윈도우 함수 사용 예제

총 합계 계산

- (**banana**, **apple**, **leek**, **cabbage**, **lettuce**, **kale**) = 54 total purchases
- (**banana**, **apple**, **leek**, **cabbage**, **lettuce**, **kale**) = 54 total purchases
- (**banana**, **apple**, **leek**, **cabbage**, **lettuce**, **kale**) = 54 total purchases
- (**banana**, **apple**, **leek**, **cabbage**, **lettuce**, **kale**) = 54 total purchases
- (**banana**, **apple**, **leek**, **cabbage**, **lettuce**, **kale**) = 54 total purchases
- (**banana**, **apple**, **leek**, **cabbage**, **lettuce**, **kale**) = 54 total purchases

소계 계산

- fruit
  - (**banana**, **apple**) = 10 total purchases
  - (**banana**, **apple**) = 10 total purchases
- vegetable
  - (**leek**, **cabbage**, **lettuce**, **kale**) = 44 total purchases
  - (**leek**, **cabbage**, **lettuce**, **kale**) = 44 total purchases
  - (**leek**, **cabbage**, **lettuce**, **kale**) = 44 total purchases
  - (**leek**, **cabbage**, **lettuce**, **kale**) = 44 total purchases

누적 합계 계산

- fruit
  - (**banana**, apple) = 2 total purchases
  - (**banana**, **apple**) = 10 total purchases
- vegetable
  - (**leek**, cabbage, lettuce, kale) = 2 total purchases
  - (**leek**, **cabbage**, lettuce, kale) = 11 total purchases
  - (**leek**, **cabbage**, **lettuce**, kale) = 21 total purchases
  - (**leek**, **cabbage**, **lettuce**, **kale**) = 44 total purchases

누적 합계 계산 (2)

- (banana, leek, apple, cabbage, lettuce, kale) = NULL
- (banana, leek, apple, cabbage, lettuce, kale) = NULL
- (**banana**, leek, apple, cabbage, lettuce, kale) = 2
- (**banana**, **leek**, apple, cabbage, lettuce, kale) = 4
- (**banana**, **leek**, **apple**, cabbage, lettuce, kale) = 12
- (**banana**, **leek**, **apple**, **cabbage**, lettuce, kale) = 21

이동 평균 계산

- (**banana**, **leek**, apple, cabbage, lettuce, kale) = 2 average purchases
- (**banana**, **leek**, **apple**, cabbage, lettuce, kale) = 4 average purchases
- (banana, **leek**, **apple**, **cabbage**, lettuce, kale) = 6.3333 average purchases
- (banana, leek, **apple**, **cabbage**, **lettuce**, kale) = 9 average purchases
- (banana, leek, apple, **cabbage**, **lettuce**, **kale**) = 14 average purchases
- (banana, leek, apple, cabbage, **lettuce**, **kale**) = 16.5 average purchases

범위 내의 항목 수 계산

- (**goose**, **dog**, **ox**, **goat**, duck, cat) = 4 animals between population range 0-2.
- (**goose**, **dog**, **ox**, **goat**, **duck**, cat) = 5 animals between population range 1-3.
- (**goose**, **dog**, **ox**, **goat**, **duck**, cat) = 5 animals between population range 1-3.
- (**goose**, **dog**, **ox**, **goat**, **duck**, cat) = 5 animals between population range 1-3.
- (goose, **dog**, **ox**, **goat**, **duck**, cat) = 4 animals between population range 2-4.
- (goose, dog, ox, goat, duck, **cat**) = 1 animal between population range 22-24.

각 카테고리에서 가장 인기있는 항목 가져오기

- fruit
  - (**banana**, **apple**) = apple is most popular
  - (**banana**, **apple**) = apple is most popular
- vegetable
  - (**leek**, **cabbage**, **lettuce**, **kale**) = kale is most popular
  - (**leek**, **cabbage**, **lettuce**, **kale**) = kale is most popular
  - (**leek**, **cabbage**, **lettuce**, **kale**) = kale is most popular
  - (**leek**, **cabbage**, **lettuce**, **kale**) = kale is most popular

범위의 마지막 값 가져오기

- fruit
  - (**banana**, **apple**) = apple is most popular
  - (**banana**, **apple**) = apple is most popular
- vegetable
  - (**leek**, **cabbage**, lettuce, kale) = cabbage is most popular
  - (**leek**, **cabbage**, **lettuce**, kale) = lettuce is most popular
  - (leek, **cabbage**, **lettuce**, **kale**) = kale is most popular
  - (leek, cabbage, **lettuce**, **kale**) = kale is most popular

순위 계산

- department 1
  - (**Jacob**, **Anthony**, **Andrew**) = Assign rank 1 to Jacob
  - (**Jacob**, **Anthony**, **Andrew**) = Assign rank 2 to Anthony
  - (**Jacob**, **Anthony**, **Andrew**) = Assign rank 3 to Andrew
- department 2
  - (**Isabella**, **Daniel**, **Jose**) = Assign rank 1 to Isabella
  - (**Isabella**, **Daniel**, **Jose**) = Assign rank 2 to Daniel
  - (**Isabella**, **Daniel**, **Jose**) = Assign rank 3 to Jose

실습2 – 명명된 윈도우 사용 예제

명명된 윈도우 참조 (1)

- 윈도우 사양에서 명명된 윈도우를 사용하기 위한 규칙은 화면에서 보듯이 적용됩니다.
- 첫째, 명명된 윈도우의 사양은 윈도우 사양 절에 정의된 새 사양으로 확장할 수 있습니다.
- 둘째, 중복된 정의를 가질 수 없습니다. 명명된 윈도우와 윈도우 사양 절에 ORDER BY 절이 있으면 오류가 발생합니다.
- 셋째, 명명된 윈도우와 PARTITION BY는 윈도우 사양에 함께 표시될 수 없습니다.
- 넷째, PARTITION BY가 필요한 경우 명명된 윈도우에 추가하시면 됩니다.
- 다섯째, ORDER BY 절, 외부 쿼리 또는 하위 쿼리에서 명명된 윈도우를 참조할 수 없습니다.

명명된 윈도우 참조 (2)

```
--this works:
SELECT item, purchases, LAST_VALUE(item)
  OVER (item_window ROWS BETWEEN 2 PRECEDING AND 2 FOLLOWING) AS most_popular
FROM Produce
WINDOW item_window AS (ORDER BY purchases)

--this does not work:
SELECT item, purchases, LAST_VALUE(item)
  OVER (item_window ORDER BY purchases) AS most_popular
FROM Produce
WINDOW item_window AS (ROWS BETWEEN 2 PRECEDING AND 2 FOLLOWING)
```