Uitbreidingen op GROUP BY

wim.bertels@ucll.be

Gebruik

Vaak om data te aggregeren (samennemen)
:: typische aggregatie functies zoals SUM, ...

Uitbreiding

Sleutelwoorden: CUBE, ROLLUP, GROUPING SETS

Meer complex voorbeeld

```
SELECT    avg(totaal)
FROM      (SELECT     spelersnr, sum(bedrag) as totaal
           FROM       boetes
           GROUP BY   spelersnr) as totalen
WHERE     spelersnr IN
          (SELECT     spelersnr
           FROM       spelers
           WHERE      plaats = 'Den Haag'
           OR         plaats = 'Rijswijk')
```

En deze?

```
SELECT B1.betalingsnr, B1.bedrag, sum(B2.bedrag)
FROM    boetes as B1, boetes as B2
WHERE B1.betalingsnr >= B2.betalingsnr
GROUP BY B1.betalingsnr, B1.bedrag
ORDER BY B1.betalingsnr
```

ROLLUP

- Verschillende aggregatieniveaus in één instructie
- Als het ware *opgerold*

Vb.

```
SELECT      spelersnr, sum(bedrag)
FROM        boetes
GROUP BY    ROLLUP (spelersnr);
```

ROLLUP uitvoer

```
SELECT      spelersnr, sum(bedrag)
FROM        boetes
GROUP BY    ROLLUP (spelersnr) ;
```

Spelersnr	Sum
6	100
27	175
44	130
8	25
104	50
Null	480

ROLLUP met 2 niveaus

Voorbeeld:

```
SELECT      plaats, spelersnr, sum(bedrag)
FROM        boetes inner join spelers USING (spelersnr)
GROUP BY ROLLUP (plaats, spelersnr);
```

Uitvoer?

ROLLUP met 2 niveaus

- Voorbeeld:

  SELECT plaats, spelersnr, sum(bedrag)

  FROM boetes inner join spelers USING (spelersnr)

  GROUP BY ROLLUP (plaats, spelersnr);

- Per plaats:
  - Per spelersnr
    - De som
  - De som
- Gevolgd door de totale som
- Er wordt als ware van achter naar voor *opgerold*
- De volgorde in de GROUP BY is dus belangrijk voor ROLLUP
- Door welke queries kan je ook bovenstaande informatie verkrijgen?

ROLLUP met 2 niveaus uitvoer

Voorbeeld:

```
SELECT    plaats, spelersnr, sum(bedrag)
FROM      boetes inner join spelers USING (spelersnr)
GROUP BY ROLLUP (plaats, spelersnr);
```

=

```
SELECT    plaats, spelersnr, sum(bedrag)
FROM      boetes inner join spelers USING (spelersnr)
GROUP BY plaats, spelersnr
UNION
SELECT    plaats, null, sum(bedrag)
FROM      boetes inner join spelers USING (spelersnr)
GROUP BY plaats
UNION
SELECT    null, null, sum(bedrag)
FROM      boetes inner join spelers USING (spelersnr);
```

ROLLUP met 2 niveaus uitvoer

Voorbeeld:

```
SELECT    plaats, spelersnr, sum(bedrag)
FROM      boetes inner join spelers USING (spelersnr)
GROUP BY ROLLUP (plaats, spelersnr);
```

Uitvoer?

```
   plaats    | spelersnr |  sum
-------------+-----------+--------
 Den Haag    |         6 | 100.00
 Den Haag    |           | 100.00
 Rijswijk    |         8 |  25.00
 Rijswijk    |        44 | 130.00
 Rijswijk    |           | 155.00
 Zoetermeer  |        27 | 175.00
 Zoetermeer  |       104 |  50.00
 Zoetermeer  |           | 225.00
             |           | 480.00
(9 rows)
```

CUBE

Vergelijkbaar met ROLLUP, maar groepeert voor elke mogelijke combinatie van de meegegeven kolommen.

➜ Eigenlijk voor elke mogelijke invalshoek

Voorbeeld:

```
SELECT plaats, spelersnr, sum(bedrag)
FROM boetes inner join spelers USING (spelersnr)
GROUP BY CUBE (plaats, spelersnr)
```

Uitvoer:

- Per speler en plaats
- Per plaats
- Per spelers
- Voor alle samen

Speelt de volgorde bij CUBE een rol?

CUBE voorbeeld

Meerdere groeperingen binnen één instructie

Voorbeeld:

```
SELECT      row_number() over () as volgnr,
            geslacht, plaats, count(*)
FROM        spelers
GROUP BY    CUBE (geslacht, plaats)
ORDER BY    geslacht, plaats
```

CUBE Uitvoer

volgnr	geslacht	plaats	count
1	M	Den Haag	7
2	M	Rijswijk	1
3	M	Voorburg	1
4	M		9
5	V	Leiden	1
6	V	Rijswijk	1
7	V	Rotterdam	1
8	V	Zoetermeer	2
9	V		5
11		Den Haag	7
12		Leiden	1
13		Rijswijk	2
14		Rotterdam	1
15		Voorburg	1
16		Zoetermeer	2
10			14

(16 rows)

GROUPING SETS

- Uitgebreide vorm van GROUP BY
- Meer mogelijkheden
  bv.

```
SELECT  geslacht, plaats, count(*)
FROM    spelers
GROUP BY GROUPING SETS ((plaats),(geslacht))
ORDER BY  2, 1
```

- GROUP BY() : alle rijen in één groep
  bv.

```
SELECT  geslacht, plaats, count(*)
FROM    spelers
GROUP BY GROUPING SETS ((geslacht, plaats),(geslacht),())
ORDER BY  1, 2
```

GROUPING SETS vs. ROLLUP

```
ROLLUP(c1,c2,c3)    ==        GROUPING SETS (
                                  (c1, c2, c3),
                                  (c1, c2),
                                  (c1),
                                  ()
                              )
```

GROUPING SETS vs. CUBE

CUBE(c1,c2,c3) == GROUPING SETS (
(c1,c2,c3),
(c1,c2),
(c1,c3),
(c2,c3),
(c1),
(c2),
(c3),
()
)

ALLE combinaties

Combinaties

- Meerdere groeperingen zijn samen mogelijk
- Mogelijkheden :
  - 1 grouping set + 1 simpele : toevoeging
  - 2 of meerder grouping sets : « vermenigvuldiging » van specificaties vergelijkbaar met cartesisch produkt
  - Meerdere grouping sets samen : omzetting
  - Bv. GROUP BY GROUPING SETS (E1,E2),E3

    = GROUP BY GROUPING SETS ((E1,E3),(E2,E3))
- Union !

Grouping sets

- GROUP BY a, CUBE (b, c), GROUPING SETS ((d), (e))

- GROUP BY GROUPING SETS (

  (a, b, c, d), (a, b, c, e),

  (a, b, d), (a, b, e),

  (a, c, d), (a, c, e),

  (a, d), (a, e)

)

GROUP BY DISTINCT

- GROUP BY ROLLUP (a, b), ROLLUP (a, c)

- GROUP BY GROUPING SETS (

  (*a*, b, c),

  (*a*, b),

  (a, b),

  (*a*, c),

  (*a*),

  (a),

  (a, c),

  (a),

  ()

)

GROUP BY DISTINCT

- GROUP BY DISTINCT ROLLUP (a, b), ROLLUP (a, c)

- GROUP BY GROUPING SETS (
  (a, b, c),
  (a, b),
  (a, c),
  (a),
  ()
)
- <> SELECT DISTINCT

Wim Bertels (CC)BY-SA-NC

Referenties:

- Slides Avanced Group By, 2016, P. De Mazière
- https://www.postgresql.org/docs/current/queries-table-expressions.html#QUERIES-GROUPING-SETS