

# SUBQUERIES 2

GECORRELEERDE SUBQUERIES

[wim.bertels@ucll.be](mailto:wim.bertels@ucll.be)

Naamsvermelding-NietCommercieel-GelijkDelen 4.0  
Unported Licentie

# Soorten subqueries

## ■ Scalaire subquery:

- *1 rij, 1 kolom => 1 waarde*

## ■ Rij-subquery:

- *1 rij, meerdere kolommen*

## ■ Kolom-subquery:

- *Meerdere rijen, 1 kolom*

## ■ Tabel-subquery:

- *Meerdere rijen, meerdere kolommen*

# Subquery in WHERE

## Scalaire subquery

■ =

■ >

■ <

■ ...

## Kolom-subquery

■ IN()

■ >= ALL()

■ ...

# Subquery in FROM

- Geef voor alle hemelobjecten die minstens 5 keer bezocht zijn alle reizen die dat hemelobject bezocht hebben. Toon reisnr en objectnaam.

```
SELECT  b.reisnr, vb.objectnaam
FROM    bezoeken AS b INNER JOIN (
        SELECT  objectnaam
        FROM    bezoeken
        GROUP BY objectnaam
        HAVING COUNT(*) >= 5 ) AS v
ON b.objectnaam = vb.objectnaam;
```

# Hoofdqueries en subqueries

- De hoofdquery krijgt enkel de output van de subquery en weet niets over details zoals: gebruikte tabellen, berekeningen, ...
- Alleen de SELECT wordt dus doorgegeven aan de hoofdquery
- De subquery weet alles over de hoofdquery tot in detail en kan alle gegevens van de hoofdquery gebruiken

# Gecorreleerde subqueries

Subquery waarin een kolom wordt gebruikt die tot een tabel behoort uit een ander select-blok.

Dus een gecorreleerde subquery kan niet autonoom uitgevoerd worden.

# Oefening 1

- Geef voor iedere reis  
het bezoek met de  
langste verblijfsduur

# Oplossing 1

- ```
SELECT b.reisnr, b.objectnaam
FROM   bezoeken AS b
WHERE  b.verblijfsduur =
        (SELECT   MAX(verblijfsduur)
         FROM     bezoeken AS allebezoeken
         WHERE    allebezoeken.reisnr = b.reisnr);
```
- Geef voor iedere reis het bezoek met de langste verblijfsduur



# Oefening 2

- Geef de spelers die meer keer bestuurslid zijn geweest dan dat ze wedstrijden hebben gespeeld. Toon spelersnr.

# Oplossing 2

```
■ SELECT spelersnr  
   FROM bestuursleden AS b  
   GROUP BY spelersnr  
   HAVING COUNT(*) >  
           (SELECT COUNT(*)  
            FROM wedstrijden AS w  
            WHERE w.spelersnr = b.spelersnr):
```

- Geef de spelers die meer keer bestuurslid zijn geweest dan dat ze wedstrijden hebben gespeeld.  
Toon spelersnr.

# EXISTS operator (is er iet, of nie?)

- Kijk of er output BESTAAT voor een subquery
- TRUE of FALSE
- Wat er in de SELECT staat maakt niet uit:
  - *Iets = TRUE*
  - *Niets / empty = FALSE*

# EXISTS operator oefening

- Geef alle reizen met een bezoek aan Jupiter. Of:
- = Geef alle reizen waarbij er een bezoek aan Jupiter BESTAAT.
- Toon reisnr en vertrekdatum.

# EXISTS operator oplossing

```
■ SELECT reisnr, vertrekdatum  
   FROM  reizen  
   WHERE EXISTS  
         (SELECT  *, 'iserietofnie'  
          FROM    bezoeken AS b  
          WHERE   b.objectnaam = 'Jupiter'  
          AND     b.reisnr = reizen.reisnr):
```

# EXISTS operator oplossing

```
■ SELECT reisnr, vertrekdatum  
   FROM  reizen  
   WHERE EXISTS  
         (SELECT  reisnr  
          FROM    bezoeken AS b  
          WHERE   b.objectnaam = 'Jupiter'  
          AND     b.reisnr = reizen.reisnr);
```

# NOT EXISTS operator (als er niks is .. dan..)

- Tegenovergestelde van EXISTS

- Geef alle hemelobjecten doe nog nooit bezocht zijn:

- `SELECT h.objectnaam`

`FROM hemelobjecten AS h`

`WHERE NOT EXISTS`

`(SELECT reisnr`

`FROM bezoeken AS b`

`WHERE b.objectnaam = h.objectnaam);`

# En Deze?:

```
■ SELECT objectnaam  
   FROM hemelobjecten  
   WHERE NOT EXISTS (  
       SELECT reisnr  
       FROM bezoeken b  
   );
```

```
■ --..
```



# ANY en ALL operatoren

- Deze operatoren verwachten een rij uitdrukking, om te vergelijken met 1 of meerdere waarden (ALL is de 'voor alle' en ANY is de 'er bestaat' uit de wiskunde)

- $> ALL$        $> ANY$
- $\geq ALL$        $\geq ANY$
- $< \dots$
- $\leq \dots$

# ANY en ALL operator oefening

- Geef de langste reis. Of:
- = Geef de reis waarbij de reisduur groter of gelijk is aan alle reizen.
- Toon reisnr

# ANY en ALL operator oplossing

■ Geef de langste reis:

■ SELECT reisnr

FROM reizen

WHERE reisduur >= ALL

(SELECT reisduur

FROM reizen);

# ANY en ALL operator, andere oplossing?

- Geef de langste reis

- `SELECT reisnr`

- `FROM reizen`

- `WHERE reisduur > ALL`

- `(SELECT reisduur`

- `FROM reizen AS anderen`

- `WHERE anderen.reisnr <> reizen.reisnr);`

- `-- WAT ALS 2 REIZEN DEZELFDE REISDUUR HEBBEN ???`

# ANY en ALL operator oefening 2

- Geef alle reizen, behalve de langste reis. Of:
- = Er is minstens 1 reis met een langere reisduur
- Toon reisnr

# ANY en ALL operator oplossing 2

■ Geef alle reizen, behalve de langste reis:

■ SELECT reisnr

FROM reizen

WHERE reisduur < ANY

(SELECT reisduur

FROM reizen);

# ANY en ALL operator: OPGELET!

■ Geef een lijst van alle planeten die groter zijn dan al hun satellieten

■ `SELECT objectnaam`

`FROM hemelobjecten AS h`

`WHERE satellietvan = 'Zon'`

`AND diameter > ALL`

`(SELECT diameter`

`FROM hemelobjecten AS maan`

`WHERE maan.satellietvan = h.objectnaam);`

■ -- Klopt dit?

```
objectnaam
-----
Mercurius
Venus
Aarde
Mars
Jupiter
Saturnus
Uranus
Neptunus
Pluto
(9 rows)
```

# ANY en ALL operator: OPGELET!

- Geef een lijst van alle planeten die kleiner zijn dan al hun satellieten

```
■ SELECT  objectnaam
FROM      hemelobjecten AS h
WHERE     satellietvan = 'Zon'
AND       diameter < ALL
          (SELECT  diameter
           FROM      hemelobjecten AS maan
           WHERE     maan.satellietvan = h.objectnaam);
```

- -- Daarnet groter, nu kleiner, .. ?

```
objectnaam
-----
Mercurius
Venus
(2 rows)
```



# ANY en ALL operator: OPGELET!

```
■ SELECT diameter  
   FROM   hemelobjecten AS manen  
   WHERE  manen.satellietvan IN ('Mercurius', 'Venus');
```

```
diameter  
-----  
(0 rows)
```

# (ANY en ALL operator) vs NULL

- Vergelijken met NULL: vaak opnieuw NULL (onbekend) tenzij het niet uitmaakt wat deze null waarde ook zou zijn, of het duidelijk is, of ..
- NULL is een geval apart!
- Wanneer de subquery geen output (NULL) heeft dan:
  - *Geeft ALL: waar / TRUE;*
  - *Geeft ANY: onwaar / FALSE;*

# UNIQUE operator

■ Geef de spelers voor wie precies één boete betaald werd.

■ SELECT spelersnr

FROM boetes AS **BT**

WHERE UNIQUE

(SELECT B.spelersnr

FROM boetes AS **B**

WHERE **B.spelersnr = BT.spelersnr**);

■ -- Niet geïmplementeerd in PostgreSQL, kan vervangen worden door  
HAVING

# OVERLAPS operator

- Geef de spelers en hun functie die in het bestuur zaten van 1 januari 1991 tot en met 31 december 1993
- ```
SELECT spelersnr, functie  
FROM bestuursleden  
WHERE (begin_datum, eind_datum)  
OVERLAPS ('1991-01-01', '1993-12-31');
```

# Combinatie oefening 1

- Geef de klanten die op een reis zijn meegegaan waar ook klant met klantnr 126 op meegegaan is.
- Toon klantnr

# Combinatie oefening 1: oplossing

■ Geef de klanten die op een reis zijn meegegaan waar ook klant met klantnr 126 op meegegaan is.

■ SELECT d.klantnr

FROM klanten AS k INNER JOIN deelnames AS **d** USING(klantnr)

WHERE EXISTS

(SELECT \*

FROM deelnames AS **andereDeelnames**

WHERE klantnr = 126

AND **andereDeelnames.reisnr = d.reisnr**)

GROUP BY d.klantnr;

# Combinatie oefening 2

- Geef de planeten die bezocht zijn op een reis waar klantnr 126 niet op meeging.
- Toon alle gegevens van de hemelobjecten

# Extra oefening 2: oplossing

■ Geef de planeten uit ons zonnestelsel die bezocht zijn op een reis waar klantnr 126 niet op meeding.

■ SELECT \*

FROM hemelobjecten AS *h*

WHERE EXISTS

(SELECT \*

FROM bezoeken AS *b*

WHERE NOT EXISTS

(SELECT \*

FROM deelnames AS *d*

WHERE klantnr = 126 AND **d.reisnr = b.reisnr**)

AND *h.objectnaam = b.objectnaam*)

AND satellietvan = 'Zon';



# Uitdaging: Wat doet deze query ?

```
SELECT spelersnr
FROM spelers AS s
WHERE NOT EXISTS
    (SELECT *
     FROM wedstrijden AS w1
     WHERE spelersnr = 57
     AND NOT EXISTS
        (SELECT *
         FROM wedstrijden AS w2
         WHERE w1.teamnr = w2.teamnr
         AND s.spelersnr = w2.spelersnr))
AND spelersnr NOT IN
    (SELECT spelersnr
     FROM wedstrijden
     WHERE teamnr IN
        (SELECT teamnr
         FROM teams
         WHERE teamnr NOT IN
            (SELECT teamnr
             FROM wedstrijden
             WHERE spelersnr = 57))));
```

Wim Bertels (CC)BY-SA-NC

Referenties:

- Slides subqueries deel 1 sql 2012-13, K. Beheydt
- Slides Databanken, H. Martens
- SQL Leerboek, R. Van der lams