Data Management

Notities 25/04/2024

Venster functies:

Over():

* Why?
  + 2 verschillende dimenties
    - Data samenvoegen (veel data, weinig data)
    - Aggregatie
      * …
        + Probleem je kan niet aggregeren maar niet samenvoegen

Oplossing: window functions/vender functies

Je krijgt hetzelfde resultaat als aggregeren en samenvoegen, voegt de rijen gewoon niet samen

* Vb van de oplossing:
  + Nummerings functies:
    - Er is nergens een volgorde vastgelegd, dus ze zijn niet vastbepaald
  + Row\_number():
    - Zijn ze nu vastbepaald?
      * Nee: de select gebeurt eerst, de order by zorgt er dus niet voor dat de rijnummers uniek bepaald zijn (kan per ongeluk uniek zijn, maar dat is helemaal niet zeker)
  + Over():
    - Nu zijn ze wel uniek bepaald
* Venster functie:
  + Denken aan oude ramen (die je naar boven moet schuiven):
    - Wat is het bereik?:
      * Helemaal open: alles
      * Dicht: je ziet niets
* Rank():
  + Telt op de hoeveelste plaats het staat
    - Vb1: 7 allemaal uit Den Haag (krijgen allemaal 1), de volgende is dan 8 en niet 2
    - Vb2: wel 2 (why? Vragen aan Leen haar pa)
* Partitioneren:
  + Vergelijkbaar met Group By, maar fijner/specifieker
* MERK OP:
  + In tegenstelling tot aggregatiefuncties:
    - Rijen gaan niet samenvoegen
* Partition by:
  + Heeft de rijnummer een betekenis of is het willekeurig toegepast
    - Het is willekeurig
      * WANT de ORDER BY 2 komt na de SELECT
* Partition by order by:
  + Rijnummer wel een betekenis???
* Order by:
  + Venster wordt steeds dichter geschoven (klopt dit??? Zie opname +-13u42)
  + Wat is de som:
    - De huidige + de vorige
      * Bij vb:
        + Als de rijen dezelfde waarde hebben (vb 1980) wordt er maar 1 daarvan bij opgeteld
* SELECT-instructie
  + Window-component:
    - Je geeft eigenlijk mee wat je anders in de OVER() mee zou geven
* OVER() vs GROUP BY:
  + Venster =/= groepering
    - Per groep heb je maar 1 waarde
    - Per venster heb je meerdere waardes
  + Twee manieren om per speler zijn totaal aantal boetes te tonen:
    - Venster:
      * OVER(PARTITION BY spelersnr)
    - Groepering:
      * SUM + GROUP BY

Ze geven niet dezelfde uitvoer

Venster: er w =/= samengeveogd, dus als 1 speler 2 boetes heeft, worden die beiden getoond

Groepering: er w samengevoegd, dus als 1 speler 2 boetes heeft krijg je de som

* Range:
  + Zie opname +-13u48
* Gelijke orde:
  + Range/bereik = het standaard gedrag
  + Rows =/= het standaard gedrag
  + Twee rijen zijn equivalent bij ORDER BY
  + Bij het voorbeeld met OVER(Order by… rows…):
    - Telt elke rij apart op
* Range parameters:
  + Zie ppt + opname +-13u50
* Range vs rows:
  + Rows: alles w apart opgeteld
  + Range (= default (vanaf het begin tot waaraan je bent (identieke dingen worden samen toegevoegd))): alles met vb dezelfde spelersnr w als een blok gezien en worden alle boetes in een keer opgeteld
* Gebruik:
  + Soms wil je over rijen heen bereiken: kan je lag voor gebruiken
    - Lag is een vertraging
      * Vraag om uitleg bij het voorbeeld, waarom zou je dit doen?
* Opgave en uitvoer
  + Fout: de volgorde is verkeerd, de grootste boetesom = 175 dus die zou rang 1 moeten zijn
    - Oplossing:
      * 1st: from
        + Welke tabellen raadplegen: teams en boetes (left outer join want de mensen zonder boete moeten eruit gehaald worden)
      * 2de: is de where component nodig?
        + Nee: er is geen filter nodig
      * 3de: is er een group by nodig:
        + Ja: groeperen op teamnummer
      * 4de: having nodig?
        + Nee
      * 5de: select nodig?
        + Ja: we moeten met de over() de rank proberen te bepalen

XML:

XML:

* Hiërarchische structuren (boomstructuur)

Eenvoudige XML Syntax:

* Case sensitive
* …

Doel:

* Kan verschillende doelen hebben
  + Data
  + Html
  + Om zowel hiërarchische datastructuren te omschrijven als deze te bevatten
* Vb: je hebt 4 nodes en elke node moet met de 3 andere nodes kunnen praten
  + Hoeveel mogelijke combinaties:
    - 4 \* 3 = 12
  + Je voegt er een node aan toe
    - 5 \* 4 = 20
* Vb: met XML ofzo (zie paint afbeelding die ik heb gemaakt):
  + Mediator patroon (er zit iets in het midden dat alles met elkaar kan doen praten (dus er moet maar telkens 1 verbinding leggen naar dat in het midden))
    - Dus: 4 nodes
      * 4 verbindingen
    - Voeg er 1 node aan toe
      * 5 verbindingen

Het stijgt dus veel minder snel

* Functies voor XML:
  + Zie ppt + opname +-14u08

JSON moeten we niet kennen voor het examen

Replicatie:

Vb: data repliceren naar andere nodes

Replicatie:

* CAP theorema:
  + Hoge beschikbaarheid
* Fysische applicatie (klopt dit? Zie opname +-14u27)
* Logische applicatie (klopr dit?)

CAP:

* Consistentie:
  + … zie ppt + opname +-14u28
* Beschikbaarheid
  + …
* Partitie tolerant
  + …
* CAP theorema zegt dat je maar aan 2 van de 3 kan voldoen (klopt dit? Zie opname +-14u28)

CAP voorbeeld 1:

* We hebben een aantal nodes (3 nodes)
  + Zie paint documenten

…:

* Zie ppt + opname +-14u32

…:

* …

Logische:

* Logische replicatie = fijner
* Typisch gebruik:
  + Zie ppt

Overzicht van planeet pg:

* … (zie opname +-15u38)
* Heilige graal multi…
  + Positief: heel goed op vlak van schaal
  + Negatief/problemen: je kan op meerdere plaatsen dezelfde data aanpassen (gaat voor errors zorgen!)
    - Zie opname voor meer info en of het correct is (zie opname +-14u39)
* Schaalbaarheid is niet altijd nodig
  + Vb: tijdens corona moesten de gemeentes bijhouden wie er allemaal wanneer vaccinaties moest krijgen, dit was op een systeem van 4-8gb ram en 2 cores (had niet meer nodig)
    - Niet alles is vb Google (Google moet wel schaalbaarheid hebben)
* Zie opname voor een voorbeeld van iets (zie opname +-14u42)

Pacelc:

* Pac -> cap (de pac is de cap)
* Elc -> else latency consistency (je moet kiezen tussen latency en consistency)

Conclusie:

* De perfecte set-up bestaat eigenlijk niet