Data Management

Notities 15/02/2024

Examen:

* Schriftelijk (gesloten boek): 10 punten
* PC (met examenbeperking): 10 punten
* Optioneel Opdracht (waardoor we papieren cursus mogen gebruiken bij PC examen)
  + Opdrachten + notities bijhouden
* Kunnen ook vragen stellen over Database Foundations (zeker nog eens herhalen van de data vakken die ik heb gehad!)
* Extra aandachtspunten examen (als hier fouten tegen worden gemaakt krijg je daar geen punten voor):
  + Niet kunnen normaliseren, technisch (PK, FK, UN) of conceptueel (SDG)
  + Fouten tegen de conceptuele verwerking van een select-instructie (dus in welke volgorde)
  + Vergeten te joinen wnr dit moet
  + Niet kunnen werken met transacties
  + …
  + …
  + …

Studie belasting:

* 6SP dus 150-180u
  + Zelf studeren: 62-92u
    - Ongeveer 6u per week zelf studeren (zeker doen!!!)
  + Voor het examen: 48u studeren (+4u examen)
  + Contacturen: 40u

Vragen:

* Stel tijdens de les
* Mag wel een mail sturen (antwoord op inhoudelijke vragen krijg je tijdens de les)
* Als de lessen gedaan zijn, kunnen we geen vragen meer stellen

**Een inleiding tot CLA:**

CAA:

* Contributer assignment agreement
* Werk is enkel eigendom van het bedrijf

CLA:

* Contributer license agreement
* Eigendom zit bij zowel bedrijf als van de maker
* Is wat we gebruiken tijdens dit vak

**Transacties en …:**

* Semafoor = verkeerslicht (dus is een manier om te beschrijven wie er groen licht heeft, wie er rood licht heeft, om aan de databank te geraken
  + DUS: transacties gebruiken
    - Oftewel gebeurd een transactie volledig, oftewel niet (was een naam voor, zie opname +-13u51)

Transacties:

* Vanaf begin tot een commit of rollback
* Wnr zinvol?
  + Als een bepaald gegeven uit meerdere tabellen geschrapt moeten worden
  + Als gebruiker zich vergist …
* Transacties kan je zien zoals bij de bank, oftewel schrijf je iets over oftewel doe je dit niet

Savepoints:

* Maken een deel van een actuele transactie ongedaan
* Tussentijdse momentopnames in een transactie

Wanneer twee mensen gebruik maken van dezelfde database op hetzelfde moment:

* Als een gebruiker een transactie doet, wordt de transactie nog niet getoond (ook al is het dezelfde gebruiker maar gwn op twee toestellen/browsers) totdat de transactie is gecommit

Problemen multi-user gebruik:

* Dirty read (uncommited read):
  + Een gebruiker leest een gegeven dat nooit gecommit werd (buiten de sessie die de transactie heeft uitgevoerd)
  + Ppt:
    - Bij transactie twee zie je data die er nooit had mogen zijn/die je nooit had mogen lezen
* Nonrepeatable (or nonreproductible) read:
  + …
  + Een transactie leest data die eerder gelezen is en vindt dat de data aangepast is door een transactie die gecommit is sinds de eerste lees operatie
  + Ppt:
    - Transactie 2 voert een read uit, transactie 1 veranderd hun vb tweet ‘tweet’ naar ‘tweety’, transactie 2 voert nog een uit na de commit van transactie 1 dus de ‘tweet’ read is nonrepeatable (klopt dit?)
  + Vorm van constructie die we kunnen vermijden
* Phantom read
  + Een transacte herleest data die eerder gelezen is en vindt dat er data bijgekomen of verwijderd is door een andere transactie die gecommit is sinds de eerste lees operatie
* Lost read/update (welk van de twee? Zie ppt):
  + Enkel de veranderingen van de laatste commit van gelijktijdige transacties die dezelfde rijen updaten zullen behouden worden
    - Dus als er op hetzelfde moment door 1 transactie 10% w toegevoegd en bij de andere transactie 20% w toegevoegd, komt er geen 30% maar maar 20% (als dit de laatste commit is geweest, als 10% de laatste commit is geweest is het 10% dat erbij komt)
* Serialization anomaly:
  + Een andere (interne) volgorde van de overlappende transacties zorgt voor een ander resultaat
    - …
  + …

Maximaal scheiden: (isolation level)

* “Start TRANSACTION ISOLATION LEVEL SERIALIZABLE;”
  + Gooit een error als er twee transacties op hetzelfde moment gebeuren en transactie 1 heeft al gecommit wanneer transactie 2 wilt committen
  + Serializable kijkt naar (zie opname +-14u15)

Oplossing:

* Transacties serieel verwerken
* Oplossing indien honderden gebruikers tegelijk willen werken
  + Transacties parallel verwerken (met een wachtrij werken)
    - Dus scheduling en met een lock table werken

Lock table:

* Share: tabel is gelockt, maar anderen mogen het wel lezen
* Exclusive: tabel is gelockt en niemand anders mag het lezen

Deadlocks:

* Locks zijn met elkaar verbonden
* Twee op elkaar aan het wachten = deadlock (klopt dit? Zie ppt)
* Vb:
  + Een iemand wilt de tabel droppen, maar moet wachten op een andere persoon die ook in een transaction zit. Die tweede persoon is gwn ‘idle in transaction’ dus de drop table transactie kan niet uitgevoerd worden. De beheerder/… moet dan de IP van degene die vb naar het toilet is gegaan en is vergeten om uit de transaction te gaan uit de transaction smijten

Isolation level:

* = mate van isolatie van gebruikers
* Niveaus:
  + Serializable: max gescheiden
  + Repeatable read:
    - Read moet gerepeat kunnen worden
  + Read commited (… (zie ppt)):
  + Read uncommited (ergste!) (dirty read)
    - Heel laxe beveiliging
* De locks van een andere transactie worden wel altijd gerespecteerd, welk niveau ze ook gebruiken (zelfs als het andere niveaus zijn)

Gevolgen:

* Vermijd langdurende transacties
  + Vb: iemand die koffie is gaan halen en die dus alle andere transacties ook ophoudt (dus ‘idle in transaction’)
  + Transactie is voor jou een kleine eenheid die allemaal samen moeten gebeuren (DUS: alles dat samen moet gebeuren = een transactie)
    - Lange transactie proberen te vermijden want die kunnen andere transacties ook ophouden
* Serializable
  + Concurrency is het laagst (dus duurt het langst)
  + …
* …
* …

**Prepared statements:**

* = objecten die aan de server zitten
* Kan gebruikt worden om performantie te optimaliseren
* ‘leven’ zolang de sessie/transactie bezig is

Prepared statements maken:

* Prepare … (zie opname +-14u51)
* …
* …

Prepared statement kan onderliggend zijn:

* Zie ppt (+ opname +-14u53)

Opzoeken van prepared statements:

* …

…:

* …

Embedded …:

* …

**Creatie en Ontwerp:**

Insert:

* In een insert kan je een select uitvoeren
  + Gebruiken beiden een tabel
* (zie opname +-14u55)
* Delete kan ook subqueries hebben
  + Vb ppt: degene die al langer dan gemiddeld in de club zitten worden verwijderd
* Tijdelijke tabellen (temporary? Zie opname +-14u56)
  + Op uw database bestaat een ‘stofzuiger’ die om de zoveel tijd tijdelijke tabellen verwijderd
* Kopie tabel
  + …
* Default waarden
  + Now = welk uur het is op dat moment

Beperkingen:

* Op tabel
* Op kolom

Deferrable:

* Vb: je hebt studenten met fietsen, een student kan meerdere fietsen hebben
  + Kan je een fiets toevoegen aan Jan als Jan niet bestaat in de studenten tabel? NEE, dit komt door (zie opname +-15u00 voor de term)
    - Je kan dit well deferren naar het einde van de transacties (dus het configureren van die constraint naar het einde van de transactie) en dus zeggen dat je Jan wel gaat toevoegen, maar nu nog niet
  + Kan je Jan verwijderen als er al fietsen zijn toegekend aan Jan? NEE!

Referende actie:

* Restrict = Kan het niet uitstellen tot het einde van de transactie(? Klopt dit? Zie opname +-15u02)
* …
* …
* …

Catalogus:

* Meta en organisatie data over de databank in de databank zelf (zie opname +-15u04)

**Schema’s en lokalen:**

Wat?:

* Schema = namespace (schema heeft een unieke naam)
  + Schema 1 kan tabel d hebben, schema 2 kan ook tabel d hebben, maar het zijn wel twee verschillende schema’s
* …

Rechten:

* Moet ervoor zorgen dat mensen aan het schema kunnen
* Grant = toekennen van rechten
* Revoke = wegnemen van rechten
* Privileges doorgeven:
  + Het is niet omdat je recht hebt om een schema te bekijken dat je dit ook aan andere mensen kunt doorgeven
    - Je kan het enkel doorgeven als je het gegrant hebt gekregen (je kan ook enkel de rechten doorgeven die je zelf hebt) (klopt dit? Zie opname +-15u06)

Localisatie:

* Tekens uit andere talen die niet beschikbaar zijn (vb Turkse tekens) in de database door localisatie
  + DUS: localisatie is heel belangrijk!
* Je kan opzoeken wat de instellingen zijn in je databank
* In Amerika w de datum vb ook anders geschreven dus daar moet je ook rekening mee houden

Impact locale:

* Je kan iets niet opzoeken omdat je die tekens niet hebt
  + Daarmee dat er wordt aangeraden om altijd UTF-8 te gebruiken, zodat dit niet gebeurt en dat er een standaard is
* Afhankelijk van uw locale hangt de volgorde af van hoe iets wordt bekeken (zie opname +-15u11 voor meer info)

**Datatypes & Functies:**

* Datatypes zijn goed omdat als alles gwn een string zou mogen zijn, kan iedereen daar gwn wat dan ook invullen

Algemene verschillen voor teken- en datatypes:

* Char: vaste lengte
* Varchar: variabele lengte
  + Varchar is het snelste en gebruikt het minste ruimte (dus wij moeten varchar gebruiken)

Strings vs Identifiers:

* ‘string’
  + ‘’ = string literal (is letterlijk dat, heeft geen betekenis)
* “identifier”
  + “” = identifier (zoekt naar een object met die naam (klopt dit? Zie opname +-15u15)
    - Als je wilt dat er een verschil is gebruik je “”

**IDS:**

Voor- en nadelen van een ID gebruiken:

* ID heeft vaak geen betekenis
  + Kan wel een betekenis krijgen, maar het blijft eigenlijk random afhangend van de volgorde (vb: facturen)
* Geen slechte sleutel
* Willekeurig (op basis van de volgorde)
* Exploits (zie opname +-15u18 (iets over webontwikkeling?))

IDENTITY:

* = standaard voor IDs
* Als mensen geen rechten hebben tot sequences, dan kunnen ze ook geen dingen toevoegen(/aanpassen)