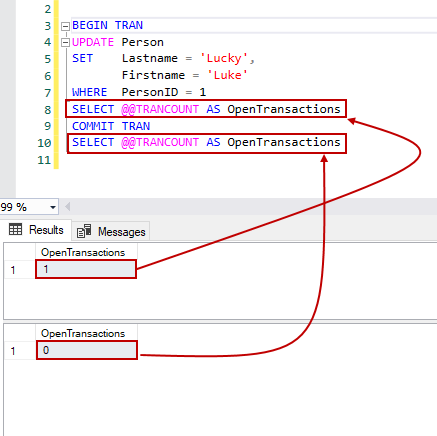
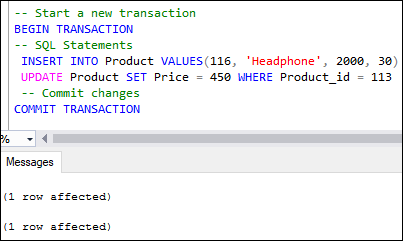
Transactie  
Wat is een transactie? : Een **transactie** bestaat uit een aantal samenhangende acties in een [database](https://nl.wikipedia.org/wiki/Database), waarbij ervoor gezorgd wordt dat deze acties ofwel allemaal plaatsvinden, ofwel geen van alle. Zie hier: <https://nl.wikipedia.org/wiki/Transactie_(dataopslag)>

# **Insert into x values (‘tweet’);**

Hier word er in een tabel x de waarde “tweet” gestoken.

**# insert into metal values (‘H’,90)** Hier word er dus in table metal de values H en 90 samen toegevoegd, dus in de kolom name en value komt H en 90 naast elkaar.

**# commit ;**

Dit is het commando om te “commiten” (Eindigt de transactie, en meer)

**# show transaction\_isolation;**

Dit toont wat die gelezen heeft ofzo idk er saat nu “read committed”

**# DELETE FROM metal where value in (190,193,3,393);**

Hier word dus van de table metal verwijderd wat value heeft gelijk aan 1 van die 4 getallen.

**# drop Table metal;**Dus 1 systeem wilt metal dorppen maar Dit kan je niet doen als een andere persoon bezig is erop, er staat “idle in transaction” leg dit uit met chat gpt!

**# START TRANSACTION ISOLATION LEVEL REPEATALE READ;**Dit heft de docent getypt om een transactie te beginnen.

Dirty read: je leest iets dat nooit gecommit werd

Nonrepeatable read: transactie leest data die eerder gelezen is en vindt de data aangepast is door een transactie die gecommit is sinds de eerste lees operatie.

Phantom read:

Lost update: enkel de veranderingen van de laatste commit van gelijktijdige transacties die dezelfde rijen updaten zullen behouden worden. (dus verliest de eerste update)

Serialization anomaly: Een andere volgorde van de overlappende

Hij doet dus 2 systemen aan in het ene doet hij insert en dan commit, in het andere doet hij ook insert, en dan geeft het een waarschuwing,

Hier wilt hij een oplossing voor.

Een oplossing is een wachtrij maken, goed voorbeeld is een bib waar je 1 boek hebt maar meerdere mensen willen dat gebruiken.  
locking:   
- de rij waar één gbruiken mee werkt wordt gelocked voor de andere gebruikers  
- Als transactie afgelopen is, wordt de blokkade opgeheven

Deadlock:  
- ….

Isolation level: mate van isolatie van gebruikers

Niveaus:  
- serializable: maximum gescheiden  
- repeatable read:   
 - lezen: share blokkades  
 - muteren: exclusive blokkades  
- read committed  
 - lezen: share blokkades  
 - muteren: exclusive blokkades  
- Read uncommitted  
 - lezen: share blokkades  
 - muteren: exclusive blokkades

Een gebruiker is altijd bezig met een transactie tot dat hij of zij typt “commit” of nog iets ma idk wa

Gevolgen als een gebruiker bezig is met een transactie:

Vermijd lang durende transacties

Serializable: concurrency is het laagst, snelheid laagst  
Read Uncommited: Concurrency is hoog, moeten weinig op elkaar wachten, kunnne gegevens lezen die enkele momenten later niet meer bestaan.  
vb; set transactie isolatie level serializable

## Prepared statements

PREPARE statement  
EXECUTE statement  
DEALLOCATE statement

Een prepare statement is bv: SELECT, UPDATE, DELETE, INSERT  
Met parameters gesubstitueerd volgens positie

Lees de noties= pgbouncer.github.io

## Structuur

Nu word leerstof van de cursus van vorig jaar bekeken, zoals select, update, delete, subquerry, tijdelijke tabellen

Default values: bv datum date not null default ‘1999-01-01’

Table beperkingen

Deferrable: fk1 referenced t(pk1) deferrable  
ZIE LINK in powerpoint slide Deferrable week 1! 15/02

Referende actie: iets verwijderen dat een verbinding heeft met een ander ding, lukt niet.  
Cascade: waterval effect. (zoek op cascade in database, why is it a risk?)

Catalogus (idk)  
Example: select \* from information\_schema.tables;

Schema = namespace (zoals in uw files op uw computer)  
Standaard is een schema “public”

Rechten: revoke priveleg on   
objecttype objectname  
to role  
(“with GRANT option” wilt zeggen dat die role diezelfe privileges kan doorgeven aan andere roles)

Localisatie  
Je hebt speciale tekens in andere talen die wij niet hebben, maar hoe werkt dat in een database?  
Met encoding: UTF-8, LATIN1, …  
Dit beinvloed alles, zoals sorteren volgorde, tijd, nummers etc

Datatypes: zie cursus  
Nummer, geld, character, binair, tijd, boolean,…

Char(n): vaste lengte  
Plaats word gereserveerd  
Varchar(n): variabele lengte 🡨 het is het beste!  
Flexibele lengte met maximum, trager?   
tekst: onbeperkte variablele lengte  
Meest flexibel, traagst?

Strings vs identifiers  
‘dit een string’  
“dit is een identifier”  
bv ‘hond’ vs “54 mijngekketabelnaam”  
sql standard uppercase > in de praktijk: sqlcode is case insensitive tenzij tussen quates.

IDS (handig voor boekhouding, maar je kan het opnieuw gebruiken niet zoals in java dus..)  
Identity datatype (geen echte sleutel, betekenis? Willekeur, exploits?)   
Identity kolom (een primary key is by defualt de identity)  
Identity maakt een sequens, dit is een teller dat optelt zoals in dns add record!

EINDE LES 1 15 feb!

I love you babybooboo <3 you're gonna do great!!! much love!

START LES 2 22 feb!

Pooling  
Pooling heeft als doen om verbindingen te bundelen in 1.

Een Connection pool is een cache van de database connecties, die onderhouden word door de database zodat ze hergebruikt kunnen worden voor request in de toekomst.  
AKA het is een memory van een al gemaakte connectie om sneller te verbinden next time.

Afbeelding met tekst, diagram, lijn, Lettertype

Automatisch gegenereerde beschrijving

Wat als je zonder pooling verbind?   
Als je zonder pooling verbindt is de connectie heel “duur”. Elke connectie heeft een opstart kost en vraagt geheugen om te kunnen voldoen aan alle mogelijke eisen die van de connectie worden verwacht.   
Dus bv je hebt 10 database verbindingen

Maar met pooling heb je dus dat die 10 verbindingen worden gebundeld / gedeeld  
Dus bv je hebt 10 verbindingen met pooling server -> De pooling server verbind dan met de database in 1 verbinding!

Je kan op verschillende manieren gaan bundelen, vergelijk conceptueel een pooling server met een dynamisch wachtrij systeem bv

● Sessie: – Elke pooling connectie heeft zijn eigen databank connectie. – Deze databank connectie komt terug ter beschikking zo gauw de pooling connectie wordt afgesloten. – Alsof je een normale verbinding hebt, alleen is ze sneller ter beschikking

● Transactie: – Elke transactie verloopt door dezelfde databank connectie. – Deze databank connectie komt terug ter beschikking zo gauw de transactie beëindigd wordt. – Je kan geen veronderstellingen maken over zaken die buiten je transactie gebeuren!

● Instructie (statement): – Deze databank connectie komt terug ter beschikking na elke instructie. – Het is zinloos expliciet transacties te gebruiken! – Je kan hiermee wel het hoogst aantal verbindingen met de laagste kost bereiken voor applicaties

## Hiërarchische model

Soorten database modelen

Opstellen van een Hiërarchische model

## Idk de titel les 29 feb RDBMS

**ACID**  
Atomicity: CF boolean  
Consistency: CF state  
Isolation: CD Concurrency  
Durability: CD commit, levensduur en select (duurzaamheid, bv als je server uitvalt en terug aangaat, dat het nog werkt enzo)

**OLTP**: databank die beschikbaar is en transacties verwerkt

**OLAP**: Software technology you can use to analyze business data from different points of view: Source: <https://aws.amazon.com/what-is/olap/#:~:text=Online%20analytical%20processing%20(OLAP)%20is,smart%20meters%2C%20and%20internal%20systems>.

Als je een produKt wilt dan moet je de volgende vragen stellen:  
Softwarevereisten  
SQL standaard

### Impliciete JOIN

Search\_path  
SHOW search\_path  
SET search\_path public, myschema

Expliciete FULL OUTER JOIN  
SELECT spelers.spelersn,r naam, bedragFROM spelers FULL OUTER JOIN boetes USING (spelersnr);  
zie slide voor meer info

TIP: ALTIJD LEZEN DE DOKUMENTATIE

<https://nl.wikipedia.org/wiki/Cartesisch_product>

SELECT \*  
FROM teams CROSS JOIN boetes; (dit toont aan dat je de boedling had om een cartesisch product zou hebben)

Natural Join  
Natuuurlijke join, lexicografisch  
SELECT ¨\*  
FROM teams NATURAL INNER JOIN boetes  
WHERE divisie = ‘ere’

Equi/theta JOIN

Equi join = verglijking met =   
Theta Join: vergelijking met een andere vergelijkingsoperator

## TEMPORARY TABLE

## LATERAL

<https://www.geeksforgeeks.org/lateral-keyword-in-sql/>

SELECT \*  
FROM nummers, ALTERAL (SELECT generate\_series(1,max\_num)) AS max\_lijst;

Vraag: Hoe oud is wim bertels?

1990 is op pensioen,  
1970 bestaat niet meer.

Womp womp womp

## Set\_operatoren

??

## EXCEPT

<https://www.sqlshack.com/understanding-the-sql-except-statement-with-examples/>

SELECT splersnr  
FROM boetes  
EXCEPT ALL  
SELECT spelersnr  
FROM ..

## UNION

<https://www.w3schools.com/sql/sql_union.asp>

## VIEWS

CREATE VIEW wspelers AS   
SELECT spelersnr, plaats  
FROM spelers  
WHERE plaats = ‘Leuven’  
<https://www.w3schools.com/sql/sql_view.asp>

Views zijn heel special, kijk een video erover.

## SECURITY

ZIE GROTE LIJST OP DIA   
Het eerste wat je doet met een onbekende table, is Revoke

## INDEX

Doel: beïnvloeden van de verwerkingstijd.

### B-tree (default)

Binaire zoek boom (node kan 2+leaves hebben)  
Best voor <, >, <=, >= en =

<https://www.youtube.com/watch?v=bOFlJ0oUjU4>

### planner / Optimizer

…

Optimizer zoekt voor jou de beste strategie, verwachte verwerkingstijd, aantal rijen, indexen, interne statistieken.

### CREATE INDEX

<https://www.w3schools.com/sql/sql_create_index.asp>

### Delete/Update

Als je iets aanpast in de tabel, dan klopt de indexering niet meer eh, dan ga je iets hebben zoals 20,21,142,22,23, Dit fix je met REINDEX.

### REINDEX

<https://www.postgresql.org/docs/current/sql-reindex.html>

Video over indexering in postgresql

<https://www.youtube.com/watch?v=-qNSXK7s7_w>

### Welke kolommen om op te indexeren?

Richtlijnen voor keuzze van kolommen:   
zie slide.

### Speciale indexvormen

* Multi-tableindex
* Virtuele kolomindex
* Selectieve index
* Hash-index
* Bitmapindex

### GIST & SP-GIST

### Gin & Brin

Je doet beter niet overall indexen op want dat gebruikt ook veel geheugen.

Samenvatting: Zie slide

### INDEX catalog\_table

## Algememe optimalisatie richtlijnen

Aptimiser  
Indexen  
Inzicht  
maar soms niet optimaal

### Vermijd de OR-operator

### Onnodig gebruik van UNION

### Vermijd de NOT-operator

### Isoleer kolommen in conditites

### Gebruik de BETWEEN-operator

### Bepaalde vormen van LIKE-operator

### Redundante condities bij joins

### Vermijd de HAVING-component

### SELECT-component : compact houden

### Vermijd DISTINCT

### ALL-optie bij set operatoren

### Vermijd datatype-conversies

### Volgorde tabellen

### Vermijd ANY- en ALL-operatoren

## Detection and benchmarking

## OLAP

BEKIJK DE SLIDES THUIS, niet goed kunnen opletten (afgeleid door pokémon)

## FILTER

## Object-Oriented databasemodel

ORDBMS/ODMS

OOPL

ODL

Basicly basis java ofzo?

## Object relational database manage systems

<https://en.wikipedia.org/wiki/Object–relational_database#:~:text=An%20object–relational%20database%20(ORD,and%20in%20the%20query%20language>.  
Object related databasemodelen

Ja kijk gwn naar de slides en zoek op op youtube, dan gaat het duidelijk zijn ofzo

Language plpgsql

AS => ::

Standaard neemt hij alles op zoals primary keys en hun indexes, je kan geen forein keys meenemen.

Sleutels & indexes

## Non-relationeel

Verteld alles dat hij de slides staat van “enkele voorbeelden” bv wat mulitvalue databanken zijn.   
Dus als je meer hierover wilt weten, kijk de opname waarin hij opsomt wat alles is.   
21/03 om 13:39

## BASE VS ACID

ACID = atomicity, consistency, isolation, and durability  
ACID = atomiciteit, consistentie, isolatie en duurzaamheid,

Soft-state= wilt zeggen dat het niet altijd hetzelfde is. De inhoud kan veranderen zonder een andere invoer. Als persoon A een querry uitvoert, en persoon B kan dezeflde querry uitvoeren maar een andere output krijgen dan persoon A.

## FDW (Foreign Data Wrapper)

Je hebt een foreign table, en een datalink.

## Create Extensions

Van postgres naar postgres of van postgres naar een bestand