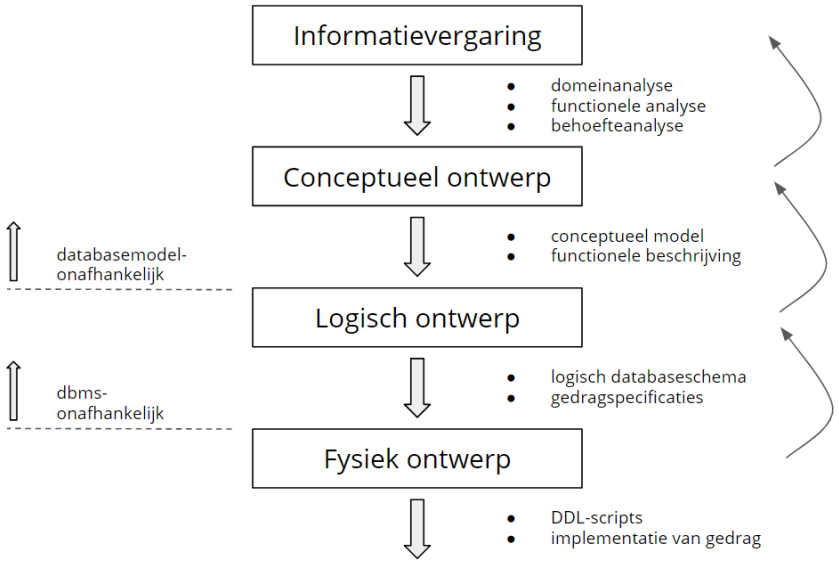
Universiteit Gent - TELIN - DDCM
2019-2020 (zittijd 2)

Databanken: Ontwerpproject

1 Introductie

Gedurende de theorie- en oefeningenlessen maakten jullie stap voor stap kennis met het ontwerpproces van een relationele databank. De theorielessen focusten hierbij vooral op het aanrijken van nieuwe (theoretische) concepten om het ontwerpproces zo correct en volledig mogelijk uit te voeren. Tijdens de oefeningenlessen pasten we de leerstof toe op voorbeelden en bekeken we specifieke problemen en oplossingen die we vaak tegenkomen bij het ontwerpen van een databank. Ook werkten we tijdens de oefeningenlessen een specifiek probleem (de projectopgave van vorig jaar) van begin tot eind, stap voor stap (klassikaal) uit. Al het materiaal (slides, opgaves, voorbeelden, oplossingen) dat hiervoor gebruikt werd is terug te vinden op Ufora.
De theorie- en oefeningenlessen zouden jullie een goede basis moeten geven om zelfstandig een volledige databank te ontwerpen. Om te bewijzen dat jullie hiertoe in staat zijn wordt er opnieuw een volledig databankontwerpproject opgestart. Gedurende dit project zijn jullie zélf verantwoordelijk voor het ontwerp, de implementatie en de bevraging van een databank voor de opslag van wetenschappelijke artikels in verband met COVID-19. De motivatie achter dit overkoepelende project is vooral om jullie in de eerste plaats te laten kennismaken met het ontwerpproces. Daarnaast zullen jullie ook, zoals in het echte bedrijfsleven, creatieve oplossingen moeten bedenken voor problemen en moeten anticiperen op eventuele moeilijkheden en beperkingen van een databanksysteem. Voor we overgaan naar de praktische kant en de probleemstelling laten we jullie nog eens kort kennismaken met het hele ontwerpproces. Een voorstelling van dit proces vinden jullie in Figuur 1.



Figuur 1: Het databankontwerpproces

1.1 Het databankontwerpproces

Een eerste fase bestaat uit het **vergaren van alle noodzakelijke informatie**. Na deze fase weten we welke data opgeslagen moeten worden in de databank, welke betekenis deze data hebben, wat de functionaliteit is van een applicatie die de data gaat gebruiken,... Deze fase resulteert in een domeinanalyse, een functionele analyse en een behoeftenanalyse. Dit is voor jullie ook het vertrekpunt van deze ontwerpoefening en wordt toegelicht in Sectie 2.
Nadat alle informatie bekend is kunnen we deze informatie abstraheren. Dit resulteert in een abstracte voorstelling van de databank die het conceptueel model wordt genoemd. Het abstraheren van deze informatie in een conceptueel model en een functionele beschrijving heet het **conceptueel ontwerp**. Meer informatie hierover vinden jullie in Sectie 3.
In een derde fase kiezen we een databankmodel en zetten we het conceptueel model om in een logisch databankschema. Deze omzetting gebeurt door middel van een databankmodel-afhankelijk omzettingsalgoritme. Het omzetten van het conceptuele model naar een databankschema heet het **logisch ontwerp** en is verder uitgelegd in Sectie 4.
Tot slot kiezen we in de vierde en laatste fase een databankmanagementsysteem (dbms). In dit dbms kan het logisch ontwerp worden omgezet naar een **fysiek ontwerp** door middel van DDL-instructies, meestal in de vorm van DDL-scripts. Deze fase wordt toegelicht in Sectie 5.

Eens de databank ontworpen en geïmplementeerd is binnen een dbms kan ze effectief gebruikt worden om data te persisteren. Hiervoor is een dbms-specifieke databankmanipulatietaal voorzien. Een manipulatietaal ondersteunt het **invoeren**, **aanpassen**, **verwijderen** en **opzoeken** van data in een fysieke databank. Dit is voor jullie de laatste opdracht en is terug te vinden in Sectie 6.
Belangrijk om op te merken is dat dit ontwerpproces in geen geval sequentieel is en dat mogelijks fasen herzien moeten worden indien er problemen optreden tijdens latere fasen. Het is daarom ten allen tijde belangrijk om te communiceren met de onderzoeksinstantie (de assistenten) en informatie te vragen indien er onduidelijkheden zijn.

1.2 Praktisch

Vooraleer jullie van start kunnen gaan willen we graag nog enkele praktische zaken aanhalen.

- Gedurende de periode van het project zullen er 3 **contactsessies** plaatsvinden waarin jullie problemen, vragen of opmerkingen over het project kunnen doorspelen aan de assistenten. Meer informatie rond deze contactsessies volgt weldra.
- Dit project wordt gemaakt in **groepen** van 2 studenten. Dit betekent niet dat jullie niet met andere groepen mogen overleggen, maar onthoud wel dat het de bedoeling is dat jullie alles zelf kunnen voor het examen en dat wanneer er plagiaat wordt vastgesteld, er op gepaste wijze ingegrepen zal worden. Inschrijven kan via de groepenmodule van dit vak op Ufora **tot 10 juli 2020, 23u59**. Ondervinden jullie problemen bij het vormen van groepen of bij het inschrijven, dan mogen jullie hier steeds voor een mail sturen naar de assistenten.
- Dit project is **gequoteerd** en zal meetellen voor 30% van jullie niet-periodegebonden evaluatie. De andere 70% wordt bepaald door jullie oplossingen van de SQL-oefeningen. Om te slagen voor dit vak dienen jullie minstens 10/20 te halen voor de niet-periodegebonden evaluatie (dus de score op dit project samen met de score op de SQL-oefeningen). 10% van de NPGE kan verdiend worden door een zinvol resultaat (dus geen lege of triviale oplossing) in te dienen dat uitvoerbaar is. 20% van de NPGE wordt bepaald op basis van de kwaliteit van de oplossing.
- Het quoteren van dit project zal enkel en alleen gebeuren aan de hand van de scripts die jullie gebruiken om de databank fysiek te implementeren, te vullen met data en te bevragen. Deze scripts zijn ook het enige materiaal dat jullie moeten **indienen**. Alles wat jullie extra indienen zal niet bekeken worden.

Meer informatie in verband met het indienen en het quoteren vinden jullie in Sectie 7.

- Indien jullie een fout maken tegen de verwachte **vereisten** verliezen jullie alle punten voor dit project. Dit is onder andere het geval indien de bestandsnamen niet correct zijn, de bestanden niet uitvoerbaar of onleesbaar zijn, er na de deadline wordt ingediend,... Indien we, op welke manier dan ook, plagiaat of bedrog vaststellen, verliezen jullie alle punten voor dit project en volgen er verdere stappen.
- Aangezien dit een herkansing is, worden de punten van de niet-periodegebonden evaluatie in geen geval overgedragen naar volgend academiejaar. Wel hoeven jullie dit project enkel te maken indien jullie score voor de NPGE (eerste zit) lager was dan 10/20. Indien jullie totaalscore lager was dan 10/20, maar de score voor de NPGE wel minstens 10/20 bedroeg, mogen jullie kiezen om de NPGE toch opnieuw te doen. Let wel, enkel het laatste resultaat zal behouden blijven.
- De **deadline** voor dit project is 14 augustus 2020 om 23u59.

Indien jullie vragen hebben kunnen jullie ons steeds contacteren door te mailen naar toon.boeckling@ugent.be, yoram.timmerman@ugent.be en joachim.peeters@ugent.be (dus graag naar alle drie de assistenten!). Jullie zijn ook welkom om vragen te stellen tijdens de online feedbacksessies. Meer info hierover vinden jullie op Ufora.

Veel succes!

2 Informatievergaring

Vooraleer we van start kunnen gaan met het ontwerpen van een databank moeten we alle informatie die belangrijk is voor dit ontwerp verzamelen en onderzoeken. Hiervoor is het belangrijk om de algemene probleemstelling te kennen en te weten welke data de opdrachtgever wil kunnen opslaan in de databank. Ook is het belangrijk om te weten wat de betekenis en context van de verschillende data is. Dit resulteert in een **domeinanalyse**. Daarnaast moeten we ook de informatiestromen (herkomst van informatie, aanmaak van nieuwe informatie,...) analyseren. Deze **functionele analyse** kan resulteren in data en functionaliteiten die oorspronkelijk niet bekend waren. Tot slot voeren we een **behoefteanalyse** uit waarbij kennis wordt vergaard over de gewenste functionaliteiten, de applicaties die de data gaan gebruiken en de manier waarop deze applicaties de data gaan gebruiken.
In dit project zullen we een databank ontwerpen voor de opslag van wetenschappelijke publicaties rond COVID-19. Aangezien jullie niet rechtstreeks met de opdrachtgever contact kunnen opnemen hebben wij reeds voor jullie deze fase uitgewerkt.

Alle wensen van de onderzoeksinstantie zijn in één beschrijving hieronder samengevat. Indien er onduidelijkheden of problemen zijn over bepaalde zaken kunnen jullie ons steeds om hulp vragen. Onthoud wel dat dit een oefening is op het zelf genereren van creatieve oplossingen en ideeën en dat eigenlijk alle informatie die jullie nodig hebben gegeven is in de beschrijving. Indien jullie het noodzakelijk achten mogen jullie steeds extra's toevoegen aan deze beschrijving.

2.1 Beschrijving

Een instantie die instaat voor onderzoek naar COVID-19 vraagt jouw hulp om een databank te ontwikkelen voor de opslag van wetenschappelijke artikels met betrekking tot de ziekte. Voor elk artikel willen ze de titel, het jaar en het tijdschrift van publicatie bijhouden. Het is echter mogelijk dat het jaar en het tijdschrift niet bekend zijn of niet bestaan omdat een artikel bijvoorbeeld nog niet is gepubliceerd. Enkel artikels waarvan de publicatiedatum niet gekend is, of die gepubliceerd zijn in of na het jaar 1900 moeten worden bijgehouden aangezien enkel deze relevant zijn voor onderzoek. Voor de eenvoud mogen jullie veronderstellen dat de titel een artikel uniek identificeert.
Artikels kunnen worden opgedeeld in twee categorieën. Enerzijds bestaan er artikels waarvan de (volledige) inhoud bekend is en anderzijds bestaan er artikels zonder inhoud die enkel als referentie gebruikt worden in een artikel uit de eerste categorie. Artikels die geen inhoud hebben, moeten minstens door één artikel mét inhoud worden gerefereerd. Artikels zonder inhoud die hier niet aan voldoen (en die dus geen enkele keer worden gerefereerd door een ander artikel), mogen wel toegevoegd worden, maar moeten eenvoudig opgespoord kunnen worden door een (databank)gebruiker.
Indien de inhoud van een artikel bekend is wil de onderzoeksinstantie natuurlijk deze inhoud volledig bijhouden. Elke inhoud bestaat ten eerste uit een geordende verzameling van secties die een titel en een volgnummer hebben, beiden uniek binnen een artikel. Daarnaast bevat elke sectie tekstuele inhoud, indien die bekend is. Ten tweede is het vaak zo dat artikels tabellen en figuren bevatten. Zowel tabellen als figuren werken met een aparte unieke nummering binnen een artikel en hebben optioneel een bijschrift dat extra uitleg verschaft bij de tabel of de figuur. Voor alle artikels (zowel met als zonder inhoud) moet er worden bijgehouden welke artikels uit de eerste categorie naar dit artikel refereren. Het refereren naar een artikel kan enkel indien het gerefereerde artikel gepubliceerd is voor (of in hetzelfde jaar van) het refererende artikel (als deze informatie bekend is) en het refererende artikel inhoud ter beschikking heeft.
Tot slot is het belangrijk om de auteurs van een artikel op te slaan. Een auteur wordt uniek geïdentificeerd door zijn voor- en achternaam (die allebei mogelijks ontbreken of onbekend zijn). Daarnaast worden er voor een auteur de namen van de affiliaties waar deze auteur gewerkt heeft opgeslagen. Deze namen worden uniek

gevormd door de naam van een instituut (vb. Universiteit Gent) en de naam van een onderzoekslabo (vb. DDCM). Het is mogelijk dat zowel de naam van het instituut als de naam van het onderzoekslabo ontbreken of onbekend zijn. Ook is het niet noodzakelijk dat voor iedere auteur een affiliatie wordt opgeslagen, aangezien een auteur mogelijks onafhankelijk van een affiliatie een artikel kan schrijven.

2.2 Bijkomende info

Om de informatievergaring af te sluiten, willen we nog een aantal zaken duidelijk maken in verband met de operaties die uigevoerd kunnen worden op de data. Dit is een hulp bij het implementeren van de beperkingen later in dit project. Ten eerste moet het ten allen tijde mogelijk zijn data toe te voegen. Daarnaast is het ook mogelijk om alle data te verwijderen met de restrictie dat dit enkel en alleen kan als ze niet meer gebruikt worden elders in de databank. Een voorbeeld is dat een affiliatie enkel verwijderd kan worden indien er geen enkele auteur meer in de databank is opgeslagen die tot deze affiliatie behoort. Tot slot is het in geen geval mogelijk om data die reeds in de databank zit aan te passen. Hier moet je dus geen rekening mee houden.

3 Conceptueel ontwerp

In dit deel focussen we ons op het abstraheren en modelleren van de informatie beschreven in 2.1. Voor deze stap is het belangrijk dat jullie alle informatie reeds uitgebreid hebben geanalyseerd en de probleemstelling goed hebben begrepen. Nu moeten jullie deze informatie omzetten naar een abstracte voorstelling van de databank. Hiervoor gebruiken we het conceptuele **Enhanced Entity-Relationship-diagram** (EER-diagram). Een nadeel van dit diagram is dat het geen ondersteuning biedt voor het weergeven van de functionaliteit en het gedrag van de abstracte concepten. Daarom is het belangrijk om alles wat niet voorgesteld kan worden door het diagram neer te schrijven in een **functionele beschrijving**.
Aangezien jullie het conceptuele ontwerp reeds uitgebreid hebben bestudeerd in vorige lessen behandelen we dit hier niet meer in detail. Voor meer informatie kunnen jullie steeds de slides van de oefeningenlessen ‘Conceptueel ontwerp’ of Hoofdstuk 3 in het boek ‘Principes van databases’ raadplegen.

3.1 Opdrachten

1. Zet de beschrijving gegeven in 2.1 om naar een EER-diagram. Let op de juiste notaties en zorg ervoor dat je diagram zo volledig mogelijk is.

2. Alles wat niet weergegeven kan worden in het EER-diagram moet gespecificeerd worden in een aparte functionele beschrijving.

Let bij het conceptuele ontwerp goed op volgende zaken.

- Neem de opgave verschillende keren goed door en probeer alle informatie die je leest te modelleren. Hou steeds de beschrijving van het probleem bij de hand zodat je niks vergeet in de omzetting. Wees dus volledig!
- Vraag je steeds af of je conceptueel model strookt met je eigen interpretatie en met de beschrijving. De bedoeling is dat je model aantoont dat je weet waarover het probleem gaat en dat je de beschrijving juist interpreteert.
- Zorg voor het consistent zijn van notaties in het diagram en de functionele beschrijving.
- Hoe leesbaarder en eenvoudiger het conceptuele ontwerp is, hoe makkelijker de volgende fasen zullen zijn. Besteed genoeg tijd aan dit ontwerp en controleer dubbel of alles klopt met de beschrijving. Indien je vragen, opmerkingen of problemen hebt, mag je steeds de assistenten contacteren.

4 Logisch ontwerp

Wanneer het conceptuele model en de functionele beschrijving ontworpen zijn kan er een databankmodel gekozen worden. Daarbij moet er steeds een afweging worden gemaakt tussen de voor- en nadelen van de databankmodellen in de context van van het gestelde probleem. Voor de eenvoudigheid en om uiteindelijke inconsistenties in de dataopslag te vermijden kiezen we hier voor het relationele databankmodel.
Het logisch databankontwerp bestaat uit de omzetting van het EER-diagram naar een **relationeel databankschema** door middel van een omzettingsalgoritme. Ook wordt er geredeneerd over eenvoudige relatie-beperkingen (uniciteit, optionaliteit, domeinrestricties,. . .) en functionele beperkingen. Voor meer informatie verwijzen we naar de oefeningenlessen ‘Logisch ontwerp’, en naar Hoofdstuk 4 en 5 in het boek ‘Principes van databases’.

4.1 Opdrachten

1. Zet het EER-diagram dat jullie hebben ontworpen in 3.1 om naar een relationeel databankschema.
2. Benadruk expliciet in je relationeel databankschema de verschillende eenvoudige beperkingen (sleutels, uniciteit, optionaliteit, domeinrestricties,. . .).
3. Redeneer over de (extra) functionele beperkingen van je relationeel databankschema op basis van de vereisten beschreven in 2.2.

Let bij de omzetting goed op volgende zaken.

- Doe de omzetting van je conceptueel ontwerp naar een relationeel databankschema zoals gezien in de oefeningenlessen. Vergeet niks om te zetten vanuit het EER-diagram. Hou hierbij de beschrijving gegeven in 2.1 en 2.2 goed in de gaten.
- Schrijf nauwgezet uit hoe en waarom je bepaalde beslissingen neemt voor de omzetting. Zo is het makkelijker om later aanpassingen te maken en te weten wat je hebt gedaan.
- Zorg ervoor dat de benamingen en verwoording van je schema in lijn liggen met je EER-diagram en functionele beschrijving uit de vorige fase.
- Probeer zoveel mogelijk informatie in je schema kwijt. Vermeld alle zaken die je denkt niet te kunnen omzetten.
- Lees na de omzetting je relationeel databankschema na en redeneer over de data die later in dit project in de relaties zal worden opgeslagen.
- Voor vragen, opmerkingen of problemen mag je steeds de assistenten contacteren.

5 Fysiek ontwerp

De logische ontwerpsfase resulteert in een relationeel databankschema en in een verzameling beperkingen. Eens dit alles voorhanden is kunnen we beginnen met het fysiek databankontwerp. Daarvoor moet eerst een relationeel databankbeheersysteem (dbms) gekozen worden en nadien moeten alle elementen uit de vorige fase geïmplementeerd worden. Voor dit probleem kiezen we voor het open-source dbms PostgreSQL.
De implementatie van een relationele databank gebeurt door middel van de gestandaardiseerde Structured Query Language (SQL). Voor het fysiek databankontwerp worden instructies gebruikt die gecategoriseerd worden onder de datadefinitietaal (DDL) van SQL. Meestal worden SQL- (DDL-) scripts gebruikt voor het gemakkelijk opslaan en uitvoeren van deze instructies. Na deze fase heb je een **werkende databank** bestaande uit tabellen aangevuld met beperkingen zoals primaire sleutels, vreemde sleutels, `CHECK`-beperkingen, triggers,... en ben je klaar om ze te vullen met data.
Vooraleer jullie beginnen met dit fysiek ontwerp is het aangeraden om de workshops ‘Introductie PostgreSQL & fysiek ontwerp’ en (later) ‘Triggers, functies & views’ te maken. Hierin maken jullie kennis met het beheren, opzetten en implementeren van een fysieke relationele databank in PostgreSQL. Voor meer informatie kunnen jullie ook steeds Hoofdstuk 6 van het boek ‘Principes van databases’ raadplegen.

5.1 Opdrachten

1. Maak een persoonlijke PostgreSQL databank aan.
2. Implementeer jullie relationeel databankschema.
3. Implementeer alle vereisten en beperkingen zoals beschreven in 2.1 en 2.2.

Let bij het fysiek ontwerp goed op volgende zaken.

- Zorg ervoor dat je een zo volledig mogelijke omzetting doet. Dit maakt het makkelijker om in de volgende fase je data correct in te voeren. Herbekijk tijdens het implementeren steeds de probleemstelling.
- Zet al je DDL-statements om de databank te implementeren samen in 1 groot SQL-script. Dit script kan in de volgende fase ook aangevuld worden met statements om de data in de databank te importeren. Door in het begin van het script alle tabellen, beperkingen, triggers,... te verwijderen kan je dit script steeds opnieuw uitvoeren wanneer je iets hebt aangepast. Dit zorgt er ook voor dat je steeds van een lege (en dus propere) toestand opnieuw kan starten zonder dat er wijzigingen verloren gaan.
- Indien je een foutmelding krijgt bij het uitvoeren van een statement, probeer dan te achterhalen wat dit betekent (door bv. de foutmelding op te zoeken) en denk goed na waarom dit zou kunnen gebeuren.
- Voor vragen, opmerkingen of problemen mag je steeds de assistenten contacteren.

5.2 Indienen

Wanneer jullie fysiek ontwerp volledig is afgerond, kunnen jullie op basis hiervan een .sql-script genereren. Dit doe je door op de commandolijn (te openen via de Windows Start-knop, vervolgens ‘cmd’ te typen en het programma dat verschijnt aan te klikken) volgend commando uit te voeren.

```
pg_dump
    --dbname nameofdb
    --username postgres
    --schema-only
    --file db_fysiek_groepX.sql
```

Hierbij dien je uiteraard ‘nameofdb’ te vervangen door de naam van de databank die je hebt aangemaakt, en ‘X’ door jouw groepsnummer. Dit bestand bevat alle statements die jullie gebruikt hebben voor de fysieke implementatie van de databank, met uitzondering van het statement voor het aanmaken van een databank. Hieronder verstaan we dus de definitie van

- tabellen,
- functies en triggers,
- aanvullende beperkingen (zoals sleutels, uniciteit, domeinrestricties,. . .),
- views.

Dit .sql-script is onderdeel van de oplossing van jullie project dat jullie uiteindelijk zullen indienen. Geef dit bestand de naam `db_fysiek_groepX.sql`, waarbij je 'X' vervangt door jouw groepsnummer. Let op, het is **zeer** belangrijk dat dit script correct en volledig uitvoert! Als dit niet het geval is, kunnen wij jullie project helaas niet verbeteren en quoteren. Test dit dus grondig!

6 Datamanipulatie

Het fysiek ontwerp resulteert in een werkende databank die klaar is voor gebruik. In deze databank zorgen alle gedefinieerde elementen ervoor dat de gewenste functionaliteit effectief aanwezig is. Wanneer alles geïmplementeerd en getest is kunnen we overgaan naar de laatste fase van het ontwerpproces: de datamanipulatie. In deze fase is het de bedoeling om de ontworpen databank te gaan gebruiken. Dit gebeurt door de databank te vullen met data (zie 6.1). Daarna is het mogelijk om deze data te gaan analyseren (zie 6.2).
Het toevoegen, aanpassen en verwijderen van data in een databank door middel van SQL-queries is aan bod gekomen in de workshop 'Data importeren' en 'Datamanipulatie'. In de oefeningenlessen SQL hebben jullie daarnaast gefocust op het bevragen en analyseren van data door middel van (complexe) SQL-queries. Voor meer informatie kunnen jullie ook steeds Hoofdstuk 6 van het boek 'Principes van databases' raadplegen.

6.1 Importeren

Een eerste stap in deze fase bestaat eruit om data te importeren in jouw aangemaakte databank. De onderzoeksinstantie heeft voor jullie drie .csv bestanden voorzien waarin alle data die ze in de databank willen hebben reeds zijn opgeslagen. Deze bestanden hebben volgende naam en inhoud.

- `artikels.csv`: Dit bestand bevat alle data met betrekking tot alle artikels en de auteurs hiervan.
- `inhoud.csv`: Dit bestand bevat alle data met betrekking tot de inhoud van artikels.
- `referenties.csv`: Dit bestand bevat alle data met betrekking tot referenties van artikels.

Let op: alle drie de .csv-bestanden beginnen met een header op de eerste rij die niet geïmporteerd moet worden als data in de databank. Gebruik daarnaast ook beide bestanden enkel voor de beschreven doeleinden, m.a.w. gebruik `referenties.csv` niet voor het importeren van bv. titels van artikels.

6.1.1 Opdrachten

Om jullie op weg te helpen hebben wij reeds een .sql-script voorzien voor de aanmaak van 3 super-relaties (`artikel_super`, `inhoud_super` en `referenties_super`) waarin je de data uit de .csv-bestanden makkelijk kan importeren. Dit kan je vinden in het bestand met naam "super_relaties.sql". Jullie taak is nu om zo goed mogelijk de eigenlijke data niet-redundant te gaan verspreiden over jullie tabellen ontworpen in Sectie 5. Met 'niet-redundant' wordt bedoeld dat er in geen enkele tabel rijen dubbel mogen voorkomen. Daarnaast is het ook belangrijk dat het proberen importeren van foutieve data (bv. refereren naar een artikel dat later gepubliceerd is dan het refererende artikel) opgevangen en afgehandeld wordt door het databanksysteem. Het is dus belangrijk dat alle voorwaarden en beperkingen die in de informatievergaringsstap werden geïdentificeerd, effectief worden geïmplementeerd in je databank. Test dit grondig, want ook hierop zullen wij een groot deel van de punten geven!

Let bij het importeren van data goed op volgende zaken.

- Voor het toevoegen van data is er een .sql-script gegeven voor de aanmaak van 3 super-relaties, zoals aangehaald hierboven. Het is noodzakelijk voor het correct quoteren van het project dat alle data uit `artikels.csv`, `inhoud.csv` en `referenties.csv` respectievelijk in de `artikel_super`, `inhoud_super` en `referenties_super` relatie worden geïmporteerd. Vanuit deze super-relaties kunnen ze dan verspreid worden naar jouw eigen relaties.

- Alle data in de .csv bestanden zijn geformateerd als tekstuele data. Het is dus aan jou om de juiste conversie te doen naar de datatypes van de attributen in jouw relaties.

- Aangezien er veel ontbrekende `NULL`-waarden in de data zitten, is het belangrijk om deze correct te behandelen. Het correct vergelijken van `NULL`-waarden kan met de vergelijkingsoperator `IS DISTINCT FROM`. In het geval de twee vergeleken waarden beiden NULL zijn geeft deze operator 'True' terug, in het geval slechts één van beiden NULL is, geeft deze operator 'False' terug. In het geval beide waarden verschillend zijn van NULL, vertoont deze operator hetzelfde gedrag als de standaard =-operator. Jammer genoeg verloopt de evaluatie van IS `DISTINCT FROM` bijzonder traag. Een snellere manier om de waarden (inclusief `NULL`) in kolommen `A` en `B` te testen op ge-

lijkheid is met behulp van de standaardoperatoren, bv. door uitvoering van `A = B OR (A IS NULL AND B IS NULL)`.

- Indien je een foutmelding krijgt bij het uitvoeren van een statement, probeer dan te achterhalen wat dit betekent (door bv. de foutmelding op te zoeken) en denk goed na waarom dit zou kunnen gebeuren.
- Voor vragen, opmerkingen of problemen mag je steeds de assistenten contacteren.

6.1.2 Indienen

Na het importeren van de data dienen jullie opnieuw 1 .sql-script in met de naam `db_importeren_groepX.sql` waarbij je 'X' vervangt door je groepsnummer. Dit bestand bevat alle `INSERT`-statements voor de verspreiding[1] van data vanuit de 3 gegeven super-relaties naar jullie tabellen. Let op, het is niet mogelijk om, zoals bij het fysiek ontwerp, dit script automatisch te genereren via bv. `pg_dump`. Je dient dus de `INSERT`-statements die je doorheen het project hebt gebruikt, manueel in hetzelfde bestand te plaatsen! Het is opnieuw **zeer** belangrijk dat dit script correct en volledig uitvoert! Als dit niet het geval is, kunnen wij jullie project helaas niet verbeteren en quoteren. Test dus grondig dat je script effectief doet wat het zou moeten doen.

6.2 Analyse

Nu alle data in de databank zitten is het mogelijk om te testen of dit correct is gebeurd door middel van analyses.

6.2.1 Opdrachten

Aangezien jullie ondertussen experts zijn in het schrijven van SQL-queries is het jullie taak om een aantal tellingen te doen van data in de databank. De vragen waarop jullie als antwoord een SQL-query moeten schrijven staan hieronder opgelijst.

1. Geef het totale aantal verschillende artikels in de databank die inhoud bevatten.
2. Geef het totale aantal verschillende auteurs waarvan de achternaam begint met de hoofdletter 'B'.
3. Geef het totale aantal verschillende figuren in de databank.

[1] Je mag er dus vanuit gaan dat de data reeds aanwezig zijn in de super-relaties. Het importeren van de .csv-bestanden naar de super-relaties moet niet worden opgenomen in het script.

4. Geef het totale aantal verschillende affiliaties in de databank waarvoor het instituut of het onderzoekslabo onbekend of ontbrekend is.

6.2.2 Indienen

Na het analyseren van de data dienen jullie opnieuw 1 .sql-script in met de naam `db_queries_groepX.sql` waarbij je 'X' vervangt door je groepsnummer. Dit bestand bevat alle SQL-queries die op bovenstaande vragen een antwoord geven. Je dient dit bestand opnieuw (zoals bij het importeren van de data) zelf aan te maken, door de query's er manueel in te plakken. Elke regel in jullie oplossingenbestand moet 1 query bevatten die verplicht eindigt met een ';'. Let op, net zoals bij de vorige scripts is het opnieuw **zeer** belangrijk dat dit script correct en volledig uitvoert! Als dit niet het geval is, kunnen wij jullie project helaas niet verbeteren en quoteren.

7 Indienen

Om het correct quoteren van dit project mogelijk te maken is het noodzakelijk 3 .sql-scripts in te dienen. Dit zijn

- `db_fysiek_groepX.sql` met alle statements die jullie gebruikt hebben voor de fysieke implementatie van de databank,
- `db_importeren_groepX.sql` met alle statements voor de verspreiding van data vanuit de 3 gegeven super-relaties naar jullie tabellen, en
- `db_queries_groepX.sql` met alle SQL-queries die antwoord geven op bovenstaande analyse-vragen.

Hierin moeten jullie vanzelfsprekend 'X' vervangen door jullie groepsnummer. Deze .sql-bestanden dienen jullie te bundelen in een .zip-bestand met de naam `db_project_groepX.zip` waarbij je 'X' opnieuw vervangt door je groepsnummer. Dit .zip-bestand dienen jullie **voor** de deadline in via de 'Opdrachten'-module op Ufora.
Zoals reeds vermeld zal dit project voor 30% van jullie niet-periodegebonden evaluatie meetellen. 10% van de NPGE kan verdiend worden door een **zinvol** resultaat (dus geen lege of triviale oplossing) in te dienen en te zorgen dat alle scripts **uitvoerbaar** zijn. 20% van de NPGE wordt bepaald op basis van de **kwaliteit** van de oplossing. De kwaliteit zal automatisch beoordeeld worden door uitvoering van een verbeterscript dat zal testen of voldaan is aan alle vereisten en beperkingen en of alle data aanwezig zijn in jullie databank.