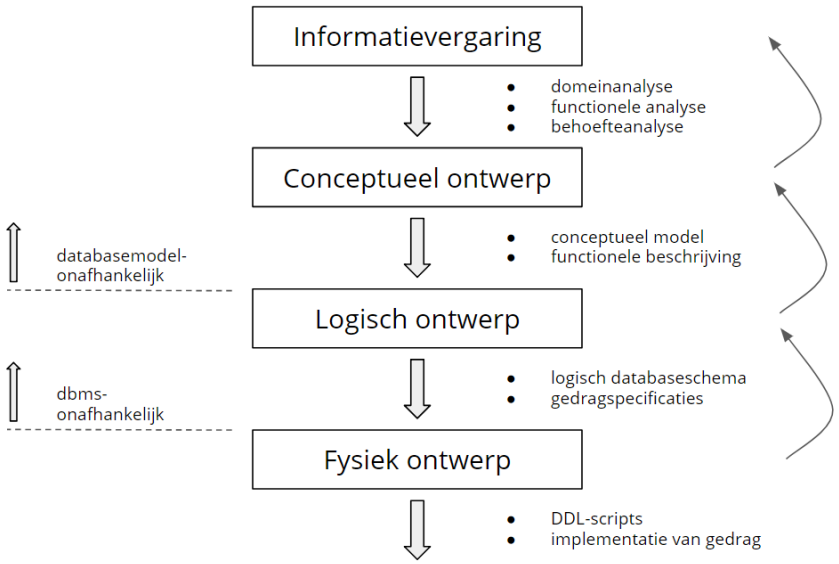
Universiteit Gent - TELIN - DDCM
2023–2024

Relationele gegevensbanken: Ontwerpproject

1 Introductie

Gedurende de theorie- en oefeningenlessen maken jullie stap voor stap kennis met het ontwerpproces van een relationele databank. De theorielessen focussen hierbij vooral op het aanrijken van nieuwe (theoretische) concepten om het ontwerpproces zo correct en volledig mogelijk uit te voeren. Tijdens de oefeningenlessen passen we de leerstof toe op voorbeelden en bekijken we specifieke problemen en oplossingen die we vaak tegenkomen bij het ontwerpen van een databank. Ook werken we tijdens deze oefeningenlessen een specifiek probleem (een projectopgave van vroeger) van begin tot eind, stap voor stap (klassikaal) uit.
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 starten we, in parallel met de lessen, een databankontwerpproject op. Gedurende dit project zijn jullie zélf verantwoordelijk voor het ontwerp, de implementatie en de bevraging van een databank voor de opslag van gegevens met betrekking tot wielerwedstrijden. 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 typische problemen en moeten anticiperen op eventuele moeilijkheden en beperkingen van een databankbeheersysteem. 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.

1.1 Het databankontwerpproces



Figuur 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 de applicaties die deze data gaan gebruiken... Deze fase resulteert in een domeinanalyse, een functionele analyse en een behoeftenanalyse, en wordt toegelicht in Sectie 2.1. Dit is voor jullie ook het vertrekpunt van dit ontwerpproject.
Nadat alle noodzakelijke informatie vergaard is, kunnen we deze informatie gaan 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 bijhorende functionele beschrijving heet het **conceptueel ontwerp**. Meer informatie hierover vinden jullie in Sectie 2.2.
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 conceptueel model naar een databankschema heet het **logisch ontwerp** en deze fase wordt verder toegelicht in Sectie 2.3.
Tot slot kiezen we in de vierde en laatste fase een databankbeheersysteem (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 2.4 en Sectie 3.1.

Van zodra de databank ontworpen en geïmplementeerd is binnen een dbms kan ze effectief gebruikt worden om data te manipuleren. Hiervoor is een dbms-specifieke datamanipulatietaal voorzien. Een manipulatietaal ondersteunt het **invoeren**, **aanpassen**, **verwijderen** en **opzoeken** van data in een fysieke databank. Meer informatie hieromtrent is terug te vinden in Sectie 3.2.
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. Daarnaast is het ten allen tijde belangrijk om te communiceren met de opdrachtgever (de assistenten) en aanvullende 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 het semester zal er tijd voorzien worden waarin jullie aan het project kunnen werken in de vorm van **lesvrije uren** (ongeveer een lesuur per week). Deze lessen zullen niet noodzakelijk volstaan om het hele project te maken, aangezien ze ook gebruikt dienen te worden ter voorbereiding/verwerking van de overige leerstof van dit vak. Hoewel dit het geval is, raden wij ten zeerste aan om deze lesvrije uren nuttig in te vullen, zodat jullie buiten de vastgelegde lesuren zo weinig mogelijk tijd moeten spenderen aan dit project.

- Dit project wordt gemaakt in **groepen** van 3 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 er op gepaste wijze ingegrepen zal worden wanneer er onregelmatigheden (bv. plagiaat) worden vastgesteld. We verwachten dus, met andere woorden, per groep een originele oplossing. Inschrijven kan via de groepenmodule van dit vak op Ufora tot **vrijdag 13 oktober 2023 om 22u00**. Na deze datum ontvangen jullie de connectieparameters om te connecteren met jullie eigen databank. Ondervinden jullie problemen bij het vormen van groepen of bij het inschrijven, dan mogen jullie hiervoor steeds een e-mail sturen naar de assistenten. Eventueel mogen jullie ervoor kiezen om dit project alleen (of per twee) te maken, maar ook dan is het noodzakelijk om in te schrijven in een groep op Ufora.

- Dit project is **gequoteerd** en zal meetellen voor 40% van jullie niet-periodegebonden evaluatie (NPGE). De andere 60% wordt bepaald door jullie oplossingen op de SQL-oefeningen. Om te slagen voor dit vak dienen jullie minstens 10/20 te ha-

len voor de niet-periodegebonden evaluatie (dus de score op dit project samen met de score op de SQL-oefeningen).

- Het **quoteren** van dit project zal enkel en alleen gebeuren aan de hand van jullie fysieke databank op onze PostgreSQL server en de scripts die jullie gebruiken om de databank te vullen met data en te bevragen. Alles wat jullie als extra materiaal indienen zal niet bekeken worden. Meer informatie in verband met het indienen en het quoteren vinden jullie in Sectie 2.5 en Sectie 3.4.
- Indien jullie niet voldoen aan de verwachte praktische **vereisten** verliezen jullie alle punten voor dit project. Dit is onder andere het geval indien de bestandsnamen van de scripts niet correct zijn, de bestanden niet uitvoerbaar of onleesbaar zijn, er na de deadline wordt ingediend. . . Controleer dus steeds dubbel of alles in orde is, zeker voor het indienen van het project, en dien misschien al eens geruime tijd op voorhand een eerste versie in. Indien we, op welke manier dan ook, onregelmatigheden vaststellen, verliezen jullie alle punten voor dit project en volgen er verdere stappen.
- Indien je geslaagd bent voor de niet-periodengebonden evaluatie (10/20 of meer) kunnen jullie ervoor kiezen om de punten rechtstreeks **over te dragen** naar de tweede zittijd. Is dit niet het geval, dan dienen jullie voor tweede zittijd opnieuw een project en SQL-oefeningen te maken in gewijzigde vorm.
- Het **eerste deel** van het project moeten jullie **indienen** voor **vrijdag 3 november 2023 om 22u00**. Voor dit eerste deel dienen jullie enkel een eenvoudig fysiek ontwerp te voorzien. Het **tweede deel** van het project moeten jullie **indienen** voor **vrijdag 22 december 2023 om 22u00**. Dit deel bestaat uit de fysieke implementatie van de meer geavanceerde beperkingen en de uitwerking van de datamanipulatie-fase, vertrekkende van een modeloplossing die correspondeert met het eerste deel van het project en die door de assistenten zal worden voorzien. Gedetailleerde informatie in verband met (het indienen van) de twee delen van dit project vinden jullie in het vervolg van deze opgave.

Indien jullie vragen, opmerkingen of problemen hebben kunnen jullie ons steeds contacteren door

- te mailen naar ann.vanoverberge@ugent.be, toon.boeckling@ugent.be, maxime.deforche@ugent.be en rihem.nasfi@ugent.be. Om de kans op een snel antwoord te verhogen, kan je best mailen naar meerdere assistenten;
- na afspraak langs te komen in ons kantoor (Sint-Pietersnieuwstraat 41, Technicum T3, verdieping 1, `https://soleway.ugent.be/routes/4453`) of op campus Schoonmmeersen (Valentin Vaerwyckweg 1). Zorg wel steeds dat je ons contacteert vooraleer je langskomt zodat we kunnen garanderen dat er zeker iemand aanwezig is;

- een (online) meeting met een van de assistenten aan te vragen;
- tijdens de oefeningenlessen vragen te stellen.

1.3 Overzicht deadlines

Hieronder volgt nog een overzicht van alle deadlines waarmee jullie rekening moeten houden. Let op, als een deadline gemist wordt, kan je geen punten meer krijgen voor het corresponderende deel van het project.

- **Deadline groepsinschrijving**: vrijdag 13 oktober 2023 om 22u00
- **Deadline indienen deel 1**: vrijdag 3 november 2023 om 22u00
- **Deadline indienen deel 2**: vrijdag 22 december 2023 om 22u00

1.4 Voorbeeldproject

Als extra hulp geven we jullie ook toegang tot een volledig uitgewerkte voorbeeldoplossing van een vroeger project. Deze uitwerking kan je als leidraad gebruiken om te zien wat er in elke fase van het project verwacht wordt en wat jullie voor elk deel moeten indienen. Het voorbeeld kunnen jullie terugvinden op Github: `https://github.com/DDCM-UGent/project_voorbeeld`.

Veel succes!

2 Basisontwerp (deel 1)

2.1 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 bestuderen. Concreet is het noodzakelijk 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 van de verschillende data is. Dit resulteert in een **domeinanalyse**. Daarnaast moeten we 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.
Aangezien jullie niet rechtstreeks met de opdrachtgever contact kunnen opnemen hebben wij reeds voor jullie deze fase uitgewerkt. Alle wensen van de opdrachtgever zijn in één beschrijving hieronder samengevat. Indien er onduidelijkheden of problemen zijn over bepaalde zaken kunnen jullie ons steeds om hulp vragen, maar onthoud wel dat dit een oefening is op het zelf genereren van creatieve oplossingen. Indien jullie het noodzakelijk achten mogen jullie steeds extra's toevoegen aan deze beschrijving.

2.1.1 Beschrijving

In dit project is het aan jullie om een databank te ontwerpen voor de opslag van data met betrekking tot wielrenners, wielerteams en wielerwedstrijden.
Eerst en vooral is het belangrijk dat data met betrekking tot wielrenners opgeslagen kunnen worden. Voor de eenvoudigheid mag je veronderstellen dat elke wielrenner een unieke naam heeft. Ook wordt voor iedere wielrenner de landcode van het land dat correspondeert met zijn/haar nationaliteit (bestaande uit exact 2 karakters, bv. 'BE'), de geboortedatum en (optioneel) de geboorteplaats, het gewicht en de lengte opgeslagen. Iedere wielrenner is huidig lid van exact 1 wielerteam. Elk wielerteam kan uniek geïdentificeerd worden door zowel een naam alsook door een verkorte naam (bestaande uit exact 3 karakters). Daarnaast hoort bij elk wielerteam een code die correspondeert met het land van herkomst (bestaande uit exact 2 karakters, bv. 'BE') en een status (de mogelijkheden hiervoor zijn beperkt tot 'World Tour' en 'Pro Tour').
Naast wielrenners en wielerteams moet er ook data met betrekking tot wielerwedstrijden opgeslagen kunnen worden. Elke wielerwedstrijd heeft een unieke naam en een landcode die correspondeert met het land waarin deze wedstrijd plaatsvindt (bestaande uit exact 2 karakters, bv. 'BE'). Er wordt een onderscheid gemaakt tussen

eendagswedstrijden (bv. ‘Ronde van Vlaanderen’) en rittenkoersen (of, met andere woorden, meerdaagse wedstrijden, bv. ‘Tour de France’). Het verschil tussen eendagswedstrijden en rittenkoersen is dat eendagswedstrijden uit exact 1 rit bestaan en rittenkoersen mogelijks uit meerdere ritten bestaan. Bij elke rit hoort een unieke identifier, een datum waarop de rit werd afgewerkt, een type rit (de mogelijkheden hiervoor zijn beperkt tot ‘Road Race’, ‘Individual Time Trial’ en ‘Team Time Trial’) en een beschrijving die aangeeft hoe de overwinning werd behaald (bv. ‘Sprint of large group’). Bij ritten die gereden worden in rittenkoersen hoort nog een positief volgnummer dat per rittenkoers slechts eenmalig kan voorkomen. Elke rit volgt een bepaalde route, die elk een unieke identifier, een vertrekplaats (bv. ‘Antwerpen’), een aankomstplaats (bv. ‘Oudenaarde’), een afstand, een aantal hoogtemeters en een moeilijkheidsniveau (een nummer tussen 1 tot 5) hebben. Let hier op dat de mogelijkheid bestaat dat meerdere ritten (eventueel zelfs ritten in verschillende wedstrijden) dezelfde route volgen.
Tot slot moet er nog data met betrekking tot de uitslag van deelnemende wielrenners in ritten worden bijgehouden. Een wielrenner kan natuurlijk op eenzelfde datum slechts aan 1 rit deelnemen. Voor iedere wielrenner die deelneemt aan een rit moet zijn/haar positie in het ritklassement (een positief nummer), de tijd waarin hij/zij de rit heeft afgewerkt, de bonustijd die hij/zij onderweg heeft verzameld en een status (de mogelijkheden hiervoor zijn beperkt tot ‘Did Finish’, ‘Did Not Finish’, ‘Did Not Start’, ‘Over Time Limit’ en ‘Disqualified’) te worden bijgehouden. Indien de uitslagstatus van een renner in een rit in een rittenkoers niet gelijk is aan ‘Did Finish’, kan hij/zij niet meer deelnemen aan ritten in dezelfde rittenkoers met een hoger volgnummer. Ook heeft de uitslagstatus van een renner in een rit invloed op de optionaliteit van de andere attributen, zoals aangegeven hieronder.

- Als de uitslagstatus van een renner in een rit ‘Did Finish’ is, is de plaats, tijd en bonustijd van deze renner in deze rit gekend.
- Als de uitslagstatus van een renner in een rit niet gelijk is aan ‘Did Finish’, is de plaats van deze renner in deze rit niet gekend.
- Als de uitslagstatus van een renner in een rit ‘Did Not Finish’ of ‘Did Not Start’ is, is de tijd van deze renner in deze rit niet gekend.
- Als de uitslagstatus van een renner in een rit ‘Did Not Start’ is, is de bonustijd van deze renner in deze rit niet gekend.

2.1.2 Bijkomende info

Om de informatievergaring af te sluiten, willen we nog een aantal zaken duidelijk maken in verband met de operaties die uitgevoerd 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 om data (in willekeurige volgorde) 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. Zo kan, bijvoorbeeld, een wielrenner enkel verwijderd worden indien deze niet meer in relatie staat met andere entiteiten (wedstrijden, teams...). Tot slot is het in geen geval mogelijk om data die reeds in de databank zitten aan te passen. Met aanpassingen moet je dus in geen geval rekening houden.

2.2 Conceptueel ontwerp

In dit deel focussen we ons op het abstraheren en modelleren van de informatie beschreven in Sectie 2.1.1. 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 conceptueel **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. Daarnaast is het ook zo dat sommige beperkingen die op deze abstracte concepten inwerken niet weergegeven kunnen worden in het diagram. Daarom is het belangrijk om alles wat niet voorgesteld kan worden in het diagram neer te schrijven in een **functionele beschrijving**.
Aangezien jullie het conceptueel ontwerp reeds uitgebreid hebben bestudeerd tijdens de lessen behandelen we dit hier niet meer in detail. Voor meer informatie kunnen jullie steeds de slides en video's van de oefeningenlessen 'Conceptueel ontwerp' en Hoofdstuk 3 in het boek 'Principes van databases' raadplegen.

2.2.1 Opdrachten

1. Zet de beschrijving gegeven in Sectie 2.1.1 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 conceptueel 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 niets vergeet in de omzetting. Denk dus bv. ook aan de mogelijke domeinrestricties voor de attributen in je model. 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.

- Zorg ervoor dat notaties in het diagram en de functionele beschrijving consistent zijn.
- Hoe leesbaarder en eenvoudiger het conceptueel 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.
- Het verkregen conceptueel ontwerp hoeven jullie niet in te dienen.

2.3 Logisch ontwerp

Wanneer het EER-diagram 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 verschillende databankmodellen in het kader 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 beperkingen (uniciteit, optionaliteit, domeinrestricties. . .) en het gedrag bij het toevoegen van data. Voor meer informatie verwijzen we naar de slides en video's van de oefeningenlessen 'Logisch ontwerp', en naar Hoofdstuk 4 en 5 in het boek 'Principes van databases'.

2.3.1 Opdrachten

1. Zet het EER-diagram dat jullie hebben ontworpen in Sectie 2.2.1 om naar een relationeel databankschema.
2. Benadruk expliciet in je relationeel databankschema de verschillende eenvoudige beperkingen (sleutels, uniciteit, optionaliteit, domeinrestricties. . .).
3. Onderzoek het gedrag bij het toevoegen van data en neem dit op als tekst in je logisch ontwerp.

Let bij het logisch ontwerp goed op volgende zaken.

- Doe de omzetting van je conceptueel ontwerp naar een relationeel databankschema zoals gezien in de oefeningenlessen. Let er op dat je niets vergeet om te zetten vanuit het EER-diagram en de functionele beschrijving. Hou hierbij de beschrijving gegeven in Sectie 2.1.1 en Sectie 2.1.2 goed in de gaten.

- Schrijf nauwgezet uit hoe en waarom je bepaalde beslissingen neemt tijdens 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 zelf te modelleren. 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 tabellen zullen worden opgeslagen.
- Voor vragen, opmerkingen of problemen mag je steeds de assistenten contacteren.
- Het verkregen logisch ontwerp hoeven jullie niet in te dienen.

2.4 Fysiek ontwerp I – Basiscomponenten

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 componenten 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 om deze instructies gemakkelijk op te slaan en uit te voeren.
In dit eerste deel van het project moeten de structuur van de databank (tabellen, attributen. . .) en alle basiscomponenten (datatypes, primaire sleutels, vreemde sleutels, `NOT NULL`-beperkingen, `UNIQUE`-beperkingen. . .) geïmplementeerd worden. De implementatie van de meer geavanceerde componenten (`CHECK`-beperkingen, triggers, functies, views) zal worden behandeld in het tweede deel van het project (zie Sectie 3.1).
Vooraleer jullie beginnen met dit fysiek ontwerp is het aangeraden om de workshop ‘Fysiek ontwerp 1’ 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.

2.4.1 Opdrachten

1. Connecteer met jullie persoonlijke databank die correspondeert met het eerste deel van dit project (met naam `e761028_deel1_groepX`). Deze databank vinden jullie terug op onze server (met naam `ddcmstud.ugent.be` en poort 8081). Connecteren met deze databank kan via de verkregen connectieparameters.

2. Implementeer alle vereiste basiscomponenten van jullie relationeel databankschema (alles behalve `CHECK`-beperkingen, triggers, functies en views) zoals beschreven in Sectie 2.1.1 en Sectie 2.1.2.

Let bij het fysiek ontwerp (deel 1) goed op volgende zaken.

- Zorg ervoor dat je een zo volledig mogelijke omzetting doet. Dit maakt het makkelijker om in een latere 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. Door in het begin van het script alle tabellen, beperkingen... 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.

- De code van het fysiek ontwerp moet niet ingediend worden. Zorg er wel voor dat jullie fysieke basisdatabank voor de deadline geïmplementeerd is op onze server.

2.5 Indienen

Deadline: vrijdag 3 november 2023 om 22u00

We zullen dit deel van het project manueel beoordelen op basis van jullie fysieke databank (met naam `e761028_deel1_groepX`) die zich bevindt op onze server. Voor het eerste deel hoeven jullie dus het conceptueel ontwerp, het logisch ontwerp en de code voor het fysiek ontwerp niet in te dienen. Zorg er wel voor dat de databank volledig afgewerkt is tegen de deadline en dus alle nodige basiscomponenten (op `CHECK`-beperkingen, triggers, functies en views na) bevat. Aanpassingen aan jullie databank zullen vanaf de deadline niet meer mogelijk zijn.

3 Geavanceerd ontwerp en datamanipulatie (deel 2)

3.1 Fysiek ontwerp II – Geavanceerde componenten

Voor het tweede deel van het project krijgen jullie een persoonlijke, (deels) geïmplementeerde versie van de databank als vertrekpunt. Deze databank (met naam `e761028_deel2_groepX`) zal door de assistenten beschikbaar gesteld worden kort na de deadline van het eerste deel van het project en zal beschikken over alle basiscomponenten[1]. Connecteren met deze databank kan opnieuw via de verkregen connectieparameters.
In deze fase van het ontwerpproces is het de bedoeling om de meer geavanceerde componenten (`CHECK`-beperkingen, triggers, functies en views) te implementeren in de gegeven databank. De implementatie van deze componenten gebeurt wederom door middel van de gestandaardiseerde Structured Query Language (SQL). Na deze fase heb je een **volledige en werkende databank** bestaande uit tabellen en beperkingen, zoals primaire sleutels, vreemde sleutels, `CHECK`-beperkingen, triggers..., en ben je klaar om ze te vullen met data.
Vooraleer jullie beginnen met deze fase is het aangeraden om de workshops 'Fysiek ontwerp 1' en 'Fysiek ontwerp 2' 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.

3.1.1 Opdrachten

1. Connecteer met jullie persoonlijke databank die correspondeert met het eerste deel van dit project (met naam `e761028_deel1_groepX`). Deze databank vinden jullie terug op onze server (met naam `ddcmstud.ugent.be` en poort 8081). Connecteren met deze databank kan via de verkregen connectieparameters.

2. Bekijk nauwgezet de componenten die reeds geïmplementeerd zijn in deze databank en zorg ervoor dat je alles begrijpt alvorens je de databank verder uitbreidt.

3. Implementeer alle vereiste, geavanceerde componenten van jullie relationeel databankschema (`CHECK`-beperkingen, triggers, functies en views) zoals beschreven in Sectie 2.1.1 en Sectie 2.1.2.

Let bij het fysiek ontwerp (deel 2) 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 tij-

[1] Je kan deze databank dus zien als een modeloplossing voor het eerste deel van het project.

dens het implementeren steeds de probleemstelling zodat je zeker de vereiste functionaliteit voorziet.

- 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 functies, 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.
- Test zeker eens of het mogelijk is om alle gegeven data zonder problemen in te laden vooraleer je de meer geavanceerde beperkingen (bv. triggers) implementeert.
- Door zelf te proberen om foutieve data (die niet voldoen aan bepaalde beperkingen) toe te voegen aan de databank, kan je testen of de door jou geïmplementeerde beperkingen ook effectief werken.
- 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.

3.1.2 Indienen

We zullen dit deel van het project deels manueel, deels automatisch beoordelen op basis van jullie fysieke databank (met naam `e761028_deel2_groepX`) die zich bevindt op onze server. We zullen, bijvoorbeeld, door middel van een script, proberen om foutieve data toe te voegen aan jullie databank en kijken of dit wordt tegengehouden door het dbms. Zorg er dus voor dat de databank volledig afgewerkt is tegen de deadline en alle vereiste componenten (inclusief de nodige `CHECK`-beperkingen, triggers, functies en views) bevat.

3.2 Data importeren

Het fysiek ontwerp resulteert in een werkende databank die klaar is voor gebruik. In deze databank zorgen alle gedefinieerde componenten 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.

Als eerste stap dienen jullie de meegeleverde data te importeren in jullie aangemaakte databank. De opdrachtgever heeft voor jullie twee .csv bestanden voorzien waarin alle data die ze in de databank willen hebben reeds zijn opgeslagen. In het eerste bestand (met naam `renners.csv`) stelt iedere rij een renner voor en daarnaast bevat de rij alle data die corresponderen met het huidige team van deze renner. In het tweede bestand (met naam `uitslagen.csv`) stelt iedere rij een resultaat van een renner in een rit voor en daarnaast bevat de rij alle data die corresponderen met deze renner en deze rit. Let op, beide .csv-bestanden beginnen met een header op de eerste rij die niet geïmporteerd moet worden als data in de databank. Gebruik de bestanden ook enkel voor de beschreven doeleinden (bv. het bestand met naam `uitslagen.csv` bevat, zoals aangegeven, niet noodzakelijk alle data met betrekking tot alle renners).
Het toevoegen, aanpassen en verwijderen van data in een databank door middel van SQL-queries is aan bod gekomen in de workshops 'Data importeren' en 'Datamanipulatie'. Voor meer informatie kunnen jullie ook steeds Hoofdstuk 6 van het boek 'Principes van databases' raadplegen.

3.2.1 Opdrachten

1. Om jullie op weg te helpen hebben wij reeds een .sql-script voorzien voor de aanmaak van twee supertabellen (`super_renners` en `super_uitslagen`) waarin je de data uit de .csv-bestanden makkelijk kan importeren. Maak deze supertabellen aan in jullie persoonlijke databank door uitvoering van dit script.
2. Importeer alle data vanuit de twee .csv bestanden in de supertabellen.
3. Verspreid de data vanuit de supertabellen over de eigenlijke tabellen in jullie databank door middel van `INSERT`-statements. Zorg ervoor dat de data correct, volledig en niet-redundant in de tabellen opgeslagen worden. Met 'niet-redundant' wordt bedoeld dat er in geen enkele tabel rijen dubbel mogen voorkomen.

Let bij het importeren van data goed op volgende zaken.

- Voor de aanmaak van de supertabellen is er reeds een .sql-script (met naam `supertables.sql`) voorzien op Ufora, zoals aangehaald hierboven. Om het project correct te kunnen verbeteren is het noodzakelijk dat alle data uit de twee .csv bestanden eerst in deze supertabellen worden geïmporteerd. Vanuit deze supertabellen kunnen ze dan verspreid worden over de eigenlijke tabellen door middel van `INSERT`-statements.
- 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 tabellen. Let ook op dat je de header van de .csv bestanden niet importeert als data in de supertabellen.

- Alle data die opgeslagen zijn in de .csv bestanden voldoen normaal gezien aan alle beperkingen die beschreven staan in Sectie 2.1.1.
- 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.

3.2.2 Indienen

Zorg er op het einde van de datamanipulatie-fase voor dat jullie fysieke databank op onze server correct en volledig gevuld is met de gegeven data. Ook is het noodzakelijk dat jullie 1 .sql-script indienen met naam `importeren_groepX.sql` waarbij je 'X' vervangt door je groepsnummer. Dit bestand bevat alle `INSERT`-statements voor de verspreiding van data vanuit de twee gegeven supertabellen naar de eigenlijke tabellen[2]. Let op, het is niet mogelijk om 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 zeer belangrijk dat dit script correct, volledig en zonder onderbrekingen 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. De supertabellen mogen voor het indienen verwijderd worden.

3.3 Analyse

Nu alle data in de databank zitten is het mogelijk om te testen of dit correct is gebeurd door middel van analyse-queries. In de oefeningenlessen SQL hebben jullie het bevragen en analyseren van data door middel van (complexe) SQL-queries uitgebreid ingeoefend. Voor meer informatie kunnen jullie ook steeds Hoofdstuk 6 van het boek 'Principes van databases' raadplegen.

3.3.1 Opdrachten

Aangezien jullie ondertussen experts zijn in het schrijven van SQL-queries is het jullie taak om een aantal tellingen te doen van de data in de databank. De vragen waarop jullie als antwoord een SQL-query moeten schrijven staan hieronder opgelijst. Zorg ervoor dat elke query exact 1 getal teruggegeven.

1. Hoeveel unieke wielrenners zijn er die minstens 1.8 meter groot zijn?

[2] Je mag er dus vanuit gaan dat de data reeds aanwezig zijn in de supertabellen. Concreet betekent dit dat de definitie van de supertabellen en het importeren van de data uit de .csv-bestanden in de supertabellen niet moet worden opgenomen in het script.

2. Hoeveel unieke landen zijn er waarin er teams gevestigd zijn met als status ‘World Tour’?
3. Hoeveel unieke wedstrijden zijn er die plaatsvinden in België (landcode ‘BE’) of in Nederland (landcode ‘NL’)?
4. Hoeveel unieke routes zijn er met een moeilijkheidsgraad die strikt lager is dan 3 en met minstens 1000 hoogtemeters?
5. Hoeveel unieke rittenkoersen zijn er waarvan alle ritten (strikt) plaatsvonden voor 1 augustus 2022?
6. Hoeveel unieke wielrenners reden ooit een rit waarin ze niet gefinisht zijn (waar, met andere woorden, de status van hun uitslag gelijk is aan ‘Did Not Finish’)?
7. Hoeveel unieke wielrenners zijn er die minstens 3 ritten gewonnen hebben?

3.3.2 Indienen

Na het analyseren van de data dienen jullie opnieuw 1 .sql-script in met naam `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 script moet 1 query bevatten die exact 1 getal teruggeeft en verplicht eindigt met een ‘;’. Let op, net zoals bij het vorige script 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.

3.4 Indienen

Deadline: vrijdag 22 december 2023 om 22u00

Om het correct quoteren van dit project mogelijk te maken is het noodzakelijk om

1. Een aangevulde implementatie en met data gevulde databank (met naam `e761028_deel2_groepX`) te hebben op onze server (zie Sectie 3.1.2 en Sectie 3.2.2), en
2. 2 .sql-scripts in te dienen, zijnde
   - `importeren_groepX.sql` met alle statements voor de verspreiding van de data vanuit de supertabellen naar jullie tabellen (zie Sectie 3.2.2), en
   - `queries_groepX.sql` met alle SQL-queries die antwoord geven op de analyse-vragen (zie Sectie 3.3.2).

Hierin moeten jullie vanzelfsprekend ‘X’ vervangen door jullie groepsnummer. Deze .sql-bestanden dienen jullie te bundelen in een .zip-bestand met naam `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 40% van jullie niet-periodegebonden evaluatie meetellen. Tijdens het verbeteren zullen wij rekening houden met de volledigheid en de correctheid van jullie oplossing. Door uitvoering van een automatisch verbeterscript (met manuele controle) testen we of alle data aanwezig zijn in jullie databank (op basis van de tellingen) en of er voldaan is aan alle vereisten en beperkingen.