Van : Maria Boes

Voor : BIS DO DT

Betreft : Rapport Database Ontwerp en Implementatie Biker

Datum : 29 augustus 2018 – versie 1.0

**Inleiding,**

Voor het tweede onderdeel van BIS maak je een databaseontwerp voor de applicatie die je in je eerste beroepsproduct hebt ontworpen. Dit ontwerp werk je vervolgens uit tot een database implementatie. Deze notitie helpt je bij het ordenen en structureren van dit ontwerp en de implementatie. Onderstaande beschrijving geeft een structuur van het rapport weer. Als je wilt afwijken van deze structuur, overleg dan met je docent.

**Structuur van het rapport Biker Database Ontwerp**

1. **Voorblad**

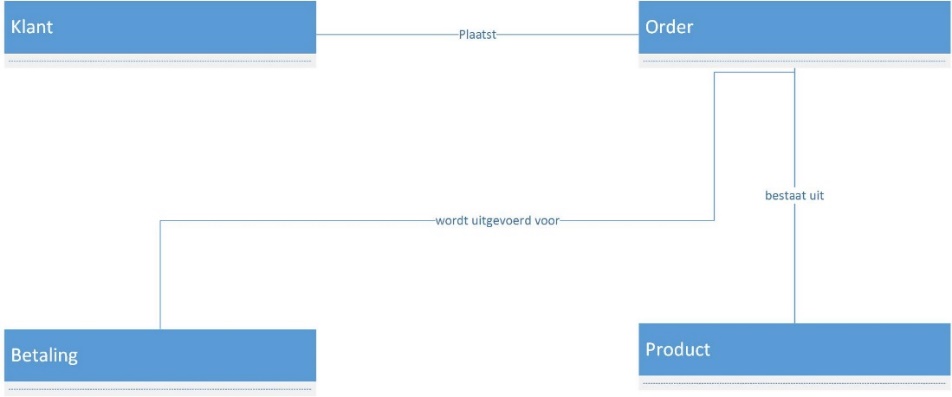
Zorg ervoor dat je document is voorzien van een voorblad dat tenminste de titel, naam/ namen van de auteurs en docenten bevat, een datum en een versienummer. Werk verder met jullie eigen logo en eigen “huisstijl”. Meer richtlijnen voor het maken van een correct rapport vind je in de ICA-controlekaart, zorg dat je rapport hieraan voldoet.

1. **Inhoudsopgave**

De inhoudsopgave is automatisch gegenereerd, zodat deze altijd consistent is en een goede weergave is van hoofdstukken, paragrafen en de bijbehorende paginanummers.

1. **Inleiding**   
   Hier beschrijf je nog even de achtergrond van Biker en de vraagstelling zoals die bij jullie is gekomen, verwijs terug naar het functioneel ontwerp. Geef duidelijk het doel van het rapport; de opbouw van het rapport en de werkwijze zoals die is gehanteerd.
2. **Conceptueel gegevensmodel**In dit hoofdstuk werk je het conceptueel datamodel uit. Kies zelf voor een passende notatiewijze, en laat niet alleen het eindresultaat / het definitieve model zien, maar ook tussenstappen / afwegingen, conform de aanpak van Noun Phrase Identification. Voor de leesbaarheid van het rapport kun je tussenstappen ook opnemen in een bijlage.

Het model kan er als volgt uitzien (maar je mag zelf een tool en notatiewijze kiezen):



1. **Logisch gegevensmodel (ER Model)**

Werk het conceptueel gegevensmodel uit naar een logisch model, waarin je de relaties documenteert conform de Martin notatiewijze (zie de bijlage voor een toelichting op deze standaard).

Gebruik voor dit model een geschikte modelleertool, zoals Astah, of een andere tool waarmee je in staat bent een model volgens de Martin notatiewijze te presenteren. Wanneer je in een tool tegen beperkingen aanloopt, zorg dan dat je consequent aangeeft wat de betekenis van symbolen is.

Zoals we in de les behandelen zorg je dat entiteiten en attributen correct worden weergegeven, relaties zijn voorzien van de juiste cardinaliteit en ook primary identifiers zijn opgenomen. Neem bij attributen op of ze verplicht zijn of niet. Let op de naamgeving van je entiteiten, zelfstandige naamwoorden in enkelvoud!

1. **Fysiek gegevensmodel**Om het gegevensmodel in SQL te kunnen implementeren moet je nadenken over de fysieke implementatie. In dit hoofdstuk leg je de eisen vast waaraan de database moet voldoen. Zorg ervoor dat deze duidelijk en ondubbelzinnig zijn verwoord en dat je keuzes toelicht.

Deze eisen bevatten in ieder geval:

* Primaire en secundaire sleutels
* Datatypes passend bij een SQL server implementatie
* Regels ten aanzien van gewenst gedrag bij update / delete
* Check constraints

Baseer je eisen op de use cases zoals gemaakt in het functioneel ontwerp, en wat je weet over de verschillende entiteiten. Definieer op basis hiervan tenminste 6 zinvolle check constraints.

De notatiewijze die we hanteren ziet er als volgt uit:

(je mag het fysieke model ook uitwerken als DDL-script, of met een PDM (physical domain model), kijk voor een toelichting op de opties die je hebt in de powerpoint van week 9) Welke notatiewijze je ook kiest, zorg in ieder geval dat je keuzes toelicht.

|  |
| --- |
| Klant (Klantnr, KlantNaam, Geslacht)  Klantnr is primaire sleutel, Integer, Not Null,  Klantnaam, Varchar(100), Not Null,  Geslacht, Varchar(1)  Check\_Constraint1 Geslacht is M of V  KlantTelefoon (*Klantnr*, Telefoonnr)  Klantnr en Telefoonnr samen vormen de primaire sleutel  Klantnr is vreemde sleutel die verwijst naar Klantnr in de tabel Klant  Bij verwijderen / update: cascade  *Toelichting keuze: in het functioneel ontwerp staat dat klanten meerdere telefoonnummers kunnen hebben.* |

Onderdeel van je uitwerking van het fysieke model is ook een toelichting hoe je normalisatie hebt toegepast. Je laat zien dat je de technieken van normalisatie beheerst door drie voorbeelden te laten zien waar je redundantie hebt verwijderd uit het gegevensmodel. Kijk naar elementen in de casusbeschrijving die duiden op niet-atomaire gegevens. Maar laat ook een voorbeeld zien waar je al bij het maken van het ER model redundantie hebt voorkomen. Toon van ieder voorbeeld dat je geeft 2 versies, waarin je laat zien hoe de gegevens eerst waren, en wat de tabel(len) na normalisatie waren. (model 1NV/2NV/3NV)

Het resultaat is een combinatie van de gegevens en een toelichtende tekst ten aanzien van de normalisatiestappen, deze kan er als volgt uitzien:

|  |
| --- |
| Voorbeeld verwijderde redundantie om te zorgen dat de tabel Klant in 3NF is: een klant kan meerdere telefoonnummers hebben, de klantgegevens waren voorgesteld als:  Klantnr Klantnaam Telefoonnr1  Telefoonnr2  Mobielnr  Dit was 0NV / herhalende groep telefoonnrs, daarom gemodelleerd als:  Tabel Klant:  Klantnr, Klantnaam (Primary key Klantnr)  Tabel KlantTelefoon  Klantnr, Telefoonnr (Primary key Klantnr, Telefoonnr)  Nu is de uitwerking 3NV. |

*Let op: Geef je overwegingen aan hoe je tot uiteindelijke keuzes bent gekomen.*

1. **Scripts**Neem als bijlage van je rapport de scripts op waarmee je de database kunt implementeren. Zorg ervoor dat deze scripts herhaalbaar zijn, dus dat je in staat bent de database te verwijderen en in zijn geheel opnieuw aan te maken.

Licht in dit hoofdstuk toe welke scripts zijn opgenomen, gebruik logische en betekenisvolle namen voor de scripts. Zorg in ieder geval dat er scripts zijn voor:

* Creatie van de database en tabellen, met attributen met correcte datatypes
* Vullen van iedere tabel met een beperkte voorbeeldpopulatie, voor iedere tabel tenminste 3 records
* Insertscript waarmee alle check constraints worden getest
* Update van de database zodat wordt voldaan aan de eisen uit hoofdstuk 6 en op een logische manier invulling wordt gegeven aan referentiële integriteit via cascading rules.

*Let op: Deze scripts moeten in overeenstemming zijn met de in hoofdstuk 5 en 6 gespecificeerde requirements / het logisch en fysieke datamodel.*

1. **Aandachtspunten/overwegingen voor het vervolg**

Beschrijf in dit hoofdstuk jullie aandachtspunten cq. overwegingen waarmee rekening moet worden gehouden voor het vervolgtraject. Vervolgstappen kun je hier ook in aangeven (onderdelen die nog verder uitgewerkt moeten worden of die extra aandacht vereisen).

Ga specifiek in op het onderwerp beveiliging, door een CRUD-matrix uit te werken. In een CRUD matrix leg je de relatie tussen de usecases die je hebt onderkend, de bijbehorende entiteiten en welke rechten deze usecase nodig heeft voor deze entiteiten.

Gebruik hiervoor de usecases en rollen zoals gespecificeerd in je functioneel ontwerp bij BP1, en maak een logische uitwerking die past bij dit ontwerp en de tabellen uit jullie database ontwerp. Een opzet voor een dergelijke matrix is:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Entiteit**  **Use case** | Klant | Order | Betaling | Product | Service | .. |
| Huren boot | C | C,U |  | R |  |  |
| Betalen borg | R | U | C,U |  |  |  |
| Repareren boot | R |  |  | R | C,R,U,D |  |
| .. |  |  |  |  |  |  |

Lever een korte toelichting per usecase aan, neem dit over uit je Functioneel ontwerp (BP1)

1. **Geraadpleegde bronnen**

Presenteer hier de bronnen die door jullie zijn geraadpleegd conform de APA notatiewijze.

## Bijlage ER modeling - Reference Card ERD

Er zijn veel verschillende varianten / tekentechnieken voor het maken van een ER-diagram, die allemaal een andere achtergrond hebben. Als je meer wilt weten over de achtegronden, en wat je in de praktijk allemaal kunt tegenkomen, kijk dan op: <https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model>

In deze course gebruiken we de Martin (crow’s foot) notation, volgens onderstaande notatiewijze:

