

Informatiesystemen 1

Thom Wiggers
s4119444

10 oktober 2012

Inhoudsopgave

1 Huishoudelijke mededelingen	1
2 Taak 1 - ORC vs. SQL	1
2.1 Inleiding	1
2.2 Systeem	2
2.2.1 ORM	2
2.2.2 SQL Tabellen	3
2.3 Eenvoudige gegevens uit het systeem halen	5
2.3.1 SQL	5
2.3.2 ORC	5
2.4 Conditioneel gegevens uit het systeem halen	5
2.4.1 SQL	5
2.4.2 ORC	6
2.5 Complexe informatie uit het systeem halen	6
2.5.1 SQL	6
2.5.2 ORC	7

1 Huishoudelijke mededelingen

Dit project maak ik individueel.

2 Taak 1 - ORC vs. SQL

2.1 Inleiding

ORM is een methode om door middel van modellen systemen te ontwikkelen waarbij gepoogd wordt zo min mogelijk fouten toe te laten en zo veel mogelijk redundantie in de opgeslagen gegevens te voorkomen.

ORM bestaat voornamelijk uit een verzameling afspraken over taalgebruik en notaties. Hierdoor zou een goed model ook voor niet-domeinexperts leesbaar en begrijpbaar moeten zijn.

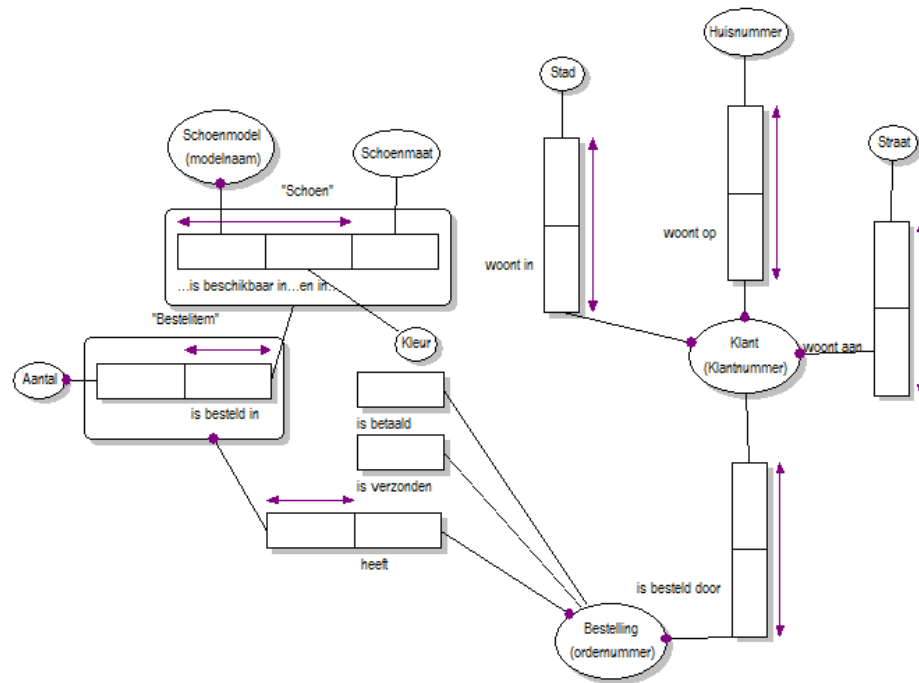
Hoewel ORM modellen vooral worden omgezet naar klassieke relationele (SQL)-databases, is het ook mogelijk om direct 'vragen' te stellen aan een dataset in ORM. Deze querytaal staat bekend als Object-Role Calculus (ORC).

Ik ga hier proberen ORC te vergelijken met de SQL taal, door middel van het vergelijken van enkele verschillende queries, zoekvragen, waarbij ik ook in ga op de fundamentele verschillen tussen de twee verschillende systemen. Hiervoor zal ik een voorbeeldsysteem beschrijven, zowel uitgevoerd in ORM als draaiende op de populaire relationele database PostgreSQL.

2.2 Systeem

Ik ga hier een voorbeeldsysteem beschrijven van een webwinkel waar men schoenen verkoopt. In deze webwinkel houdt men bestellingen bij, en profielen van klanten. Bestellingen kunnen bestaan uit een of meerdere paren schoenen, in verschillende maten en aantallen.

2.2.1 ORM



Figuur 1: ORM Model van het voorbeeldsysteem

— Check syntax — Todo: Fix diagram: Naam, constraint Schoen

Schoen: Schoenmodel (modelnaam) is beschikbaar
 in Schoenmaat en is beschikbaar in Kleur.
 Bestelitem: Schoen is besteld in aantal.
 Bestelling (ordernummer) heeft Bestelitem.
 Bestelling (ordernummer) is besteld door
 Klant (klantnummer).
 Bestelling is betaald.
 Bestelling is verzonden.
 Klant (klantnummer) woont aan
 Straat (straatnaam).
 Klant (klantnummer) woont op
 Huisnummer (nr).
 Klant (klantnummer) woont in Stad (naam).
 Klant (klantnummer) heet Naam.

2.2.2 SQL Tabellen

Het transformeren van het ORM model naar SQL tabellen is redelijk eenvoudig, maar om dubbel voorkomende gegevens te voorkomen heb ik op verschillende plaatsen een identificatienummer aan tabellen toegevoegd als vervangende primary key.

Hier wordt al een duidelijk nadeel van SQL-gebaseerde databases zichtbaar: het is niet mogelijk om gegevens te objectiveren.

Zie hiervoor de tabellen 1, 2, 3, 4 en 5.

id	Schoenmodel	Kleur	Schoenmaat
1	Model a	Zwart	43
2	Model a	Zwart	42
3	Model a	Wit	43
...

Tabel 1: Schoen tabel

id	schoenid	Aantal
1	2	1
2	2	2
3	3	1
...

Tabel 2: Bestelitem Tabel

Bestelnummer	Bestelddoor	Verzonden	Betaald
1	1	Y	Y
2	1	N	Y
3	2	Y	N
...

Tabel 3: Bestellingen

Bestelitem	Bestelling
1	1
2	1
3	2
...	...

Tabel 4: BestelitemBestelling: Bestelde items horende bij bestellingen

Klantnummer	Naam	Straat	Huisnummer	Stad
1	John Doe	Heyendaalseweg	91	Nijmegen
2	Steve Foo	Asselsestraat	34	Apeldoorn
3	Jane Bar	Kanaalstraat	33	Amsterdam

Tabel 5: Klant tabel

2.3 Eenvoudige gegevens uit het systeem halen

2.3.1 SQL

Eenvoudige gegevens uit het informatiesysteem halen is eenvoudig in SQL. Een **SELECT** statement is erg eenvoudig voor elkaar te krijgen. Bijvoorbeeld het selecteren van alle verschillende schoenen die in de winkel te koop worden aangeboden:

```
SELECT Schoenmodel, Kleur, Schoenmaat  
FROM Schoen;
```

2.3.2 ORC

In ORC is het ook eenvoudig om dezelfde gegevens op te halen:

```
Schoenmodel, Kleur, Schoenmaat FROM Schoen
```

Het verschil tussen ORC en SQL is hier niet zo groot. SQL heeft **SELECT**, maar verder zou men bijna copy/paste kunnen doen.

2.4 Conditioneel gegevens uit het systeem halen

Men wil niet altijd alle gegevens uit een informatiesysteem hebben. Daarom is het in SQL en in ORC mogelijk om condities op te geven waaraan de op te vragen informatie moet voldoen.

2.4.1 SQL

Stel, ik wil alle verschillende modellen van zwarte schoenen hebben in maat 43.

```

SELECT Schoenmodel
FROM Schoen
WHERE Schoenmaat = '43'
      AND Kleur='Zwart';

```

2.4.2 ORC

In ORC gaat het zo:

```

Schoenmodel FROM Schoen in Kleur 'Zwart'
      AND in Schoenmaat '43'

```

Ook hier is er een grote overeenkomst tussen SQL en ORC. SQL heeft hier het **WHERE** keyword om onderscheid te maken tussen de condities en de tabel, maar verder is het grotendeels hetzelfde.

2.5 Complexe informatie uit het systeem halen

Stel, ik wil weten uit welke steden de mensen komen die zwarte schoenen besteld hebben.

2.5.1 SQL

```

SELECT stad
FROM Klant
WHERE Klant.Klantnummer IN
      (SELECT Bestelddoor
      FROM Bestellingen
      JOIN BestelitemBestelling
      ON Bestelling = Bestelnummer
      WHERE Bestelitem IN
      (SELECT id
      FROM Bestelitem

```

```

WHERE Schoenid IN
  (SELECT id
   FROM Schoen
   WHERE Kleur = 'Zwart')
)
)

```

Dit is duidelijk een erg complexe query, omdat men vele tabellen door moet omdat de koppeling tussen de informatie vrij zwak is in Relationale Databases.

2.5.2 ORC

```

Stad woonplaats van Klant
  die besteld heeft Bestelling
    die Bestelitem heeft
      met Schoen in kleur Zwart

```

De ORC query is hier duidelijk een stuk eenvoudiger. Dit komt doordat in een ORM-schema de relaties tussen verschillende gegevens een stuk duidelijker is edefineerd. Waar SQL steeds in verschillende lijsten moet zoeken die allemaal apart gemaakt dienen te worden, is het hier iets wat impliciet gebeurt.

2.6 Conclusie

Waar ORC en SQL vaak erg op elkaar lijken, is ORC soms in staat om veel duidelijker queries op te stellen, vooral wanneer het om complexe informatie gaat. Relationale Databases zijn minder sterk in het weergeven van de verbanden tussen informatie, en er moeten vaak extra

gegevenstypen geïntroduceerd worden (identificatienummers bijvoorbeeld) om dataduplicatie te voorkomen.