

Databases

Les 9:

# Intro databases, modelleren en SQL DDL



# Onderwerpen les:

- Introductie databases
- DBMS
- Relatieel model
- Modelleren
- ERD: conceptueel, logisch  
en fysiek DB ontwerp
- SQL DDL

# Leerdoelen

- ✓ Aan het einde van de dag kan je achtergronden (kleur, afbeelding en gradients) toepassen.
- ✓ Aan het einde van de dag kan je tekst opmaken (kleur, font, font-size).
- ✓ Aan het einde van de dag kan je een boxshadow toepassen.
- ✓ Aan het einde van de dag weet je wat BEM is.
- ✓ Aan het einde van de dag kan je media toevoegen aan jouw pagina's.

# Introductie Databases

## Wat is een database?

De manier waarop we gegevens / data opslaan is iets dat in de loop van de tijd erg veel is veranderd.



title	release_year	length	replacement_cost
West Lion	2006	159	29.99
Virgin Daisy	2006	179	29.99
Uncut Suicides	2006	172	29.99
Tracy Cider	2006	142	29.99
Song Hedwig	2006	165	29.99
Slacker Liaisons	2006	179	29.99
Sassy Packer	2006	154	29.99
River Outlaw	2006	149	29.99
Right Cranes	2006	153	29.99
Quest Mussolini	2006	177	29.99
Poseidon Forever	2006	159	29.99
Loathing Legally	2006	140	29.99
Lawless Vision	2006	181	29.99
Jingle Sagebrush	2006	124	29.99
Jericho Mulan	2006	171	29.99
Japanese Run	2006	135	29.99
Gilmore Boiled	2006	163	29.99
Floating Garden	2006	145	29.99
Fantasia Park	2006	131	29.99
Extraordinary Conqueror	2006	122	29.99
Everyone Craft	2006	163	29.99
Dirty Ace	2006	147	29.99
Clyde Theory	2006	139	29.99
Clockwork Paradise	2006	143	29.99
Ballroom Mockingbird	2006	173	29.99



# Introductie Databases

## Opslagplaats voor data

Een database is een digitaal opgeslagen archief. De database is een collectie van **persistente** data, dit houdt in dat het wordt opgeslagen in het permanente geheugen van een computersysteem (inhoud gaat niet verloren bij het uitschakelen van het systeem).



# Introductie Databases

## Belang van databases

Steeds meer gegevens worden opgeslagen in een database. Het functioneren van een webshop, overheidsinstantie, de wetenschap en veel meer is ondenkbaar zonder een database.

Customer Relationship Management Systemen (CRM)

Business Intelligence (BI)

Content Management System (CMS)

Artificial Intelligence (AI)

Machine Learning

# Introductie Databases

## Belangrijkste functies database

1. een grote hoeveelheid gegevens opslaan, wijzigen en verwijderen
2. zeker stellen dat gegevens correct en compleet zijn
3. werkzaamheden van meerdere gebruikers tegelijk mogelijk maken
4. gegevens zo snel mogelijk zoeken en aan gebruiker teruggeven
5. structuren aanmaken om gegevens goed op te slaan
6. datazekerheid en dataprotectie garanderen
7. beheer en uitbreiding van de database faciliteren

# Introductie Databases

## DBMS DataBase Management System

Een DBMS is een software systeem waarmee databases worden beheerd. Informatie/data wordt opgeslagen op een manier dat makkelijk kan worden opgehaald, aangepast, opgeschaald en verwijderd.



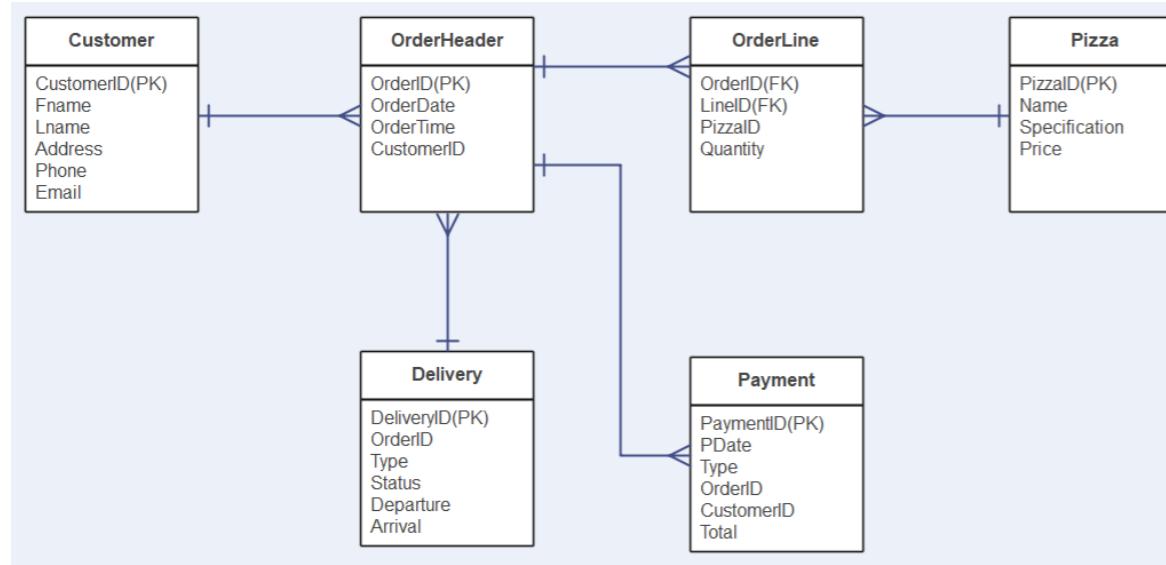
Er zijn verschillende Database management systemen te verkrijgen, met elk zijn eigen voordelen en nadelen. In deze cursus gaan we gebruik maken van PostgreSQL

# Introductie Databases

## Relationele Database

Er is een grote variëteit aan database typen. Waarbij twee types het meest populair zijn:

- Relationele database (SQL): opgebouwd uit rijen en kolommen
- NoSQL database: niet relationele database



Voorbeeld relaties in een relationele database

# Introductie Databases

## Rijen en kolommen

Een relationele database is opgebouwd in rijen en kolommen:

- Een rij / record geeft een groep aan die soortgelijke informatie bevatten over hetzelfde object (bv: een student)
- Een kolom bevat de informatie die voor elk record wordt opgeslagen (bv: voor-naam van alle studenten)

	A	B	C	D	E	
1	Studenten					
2	StudentNr	Voornaam	Achternaam	Email		
3	1232345	Kees	de Man	<a href="mailto:kees@student.nl">kees@student.nl</a>		
4	2632348	Piet	Vries	<a href="mailto:piet@student.nl">piet@student.nl</a>		
5	2432265	Carla	Konings	<a href="mailto:carla@student.nl">carla@student.nl</a>		
6	1343233	John	Doe	<a href="mailto:john@student.nl">john@student.nl</a>		
7	1454321	Suzanne	de Wit	<a href="mailto:suzanne@student.nl">suzanne@student.nl</a>		
8	1897543	Michael	Drenth	<a href="mailto:michael@student.nl">michael@student.nl</a>		
9	1766543	Dwayne	Johnson	<a href="mailto:dwayne@student.nl">dwayne@student.nl</a>		

Een relationele database kun je vergelijken met een excel of Google sheet.

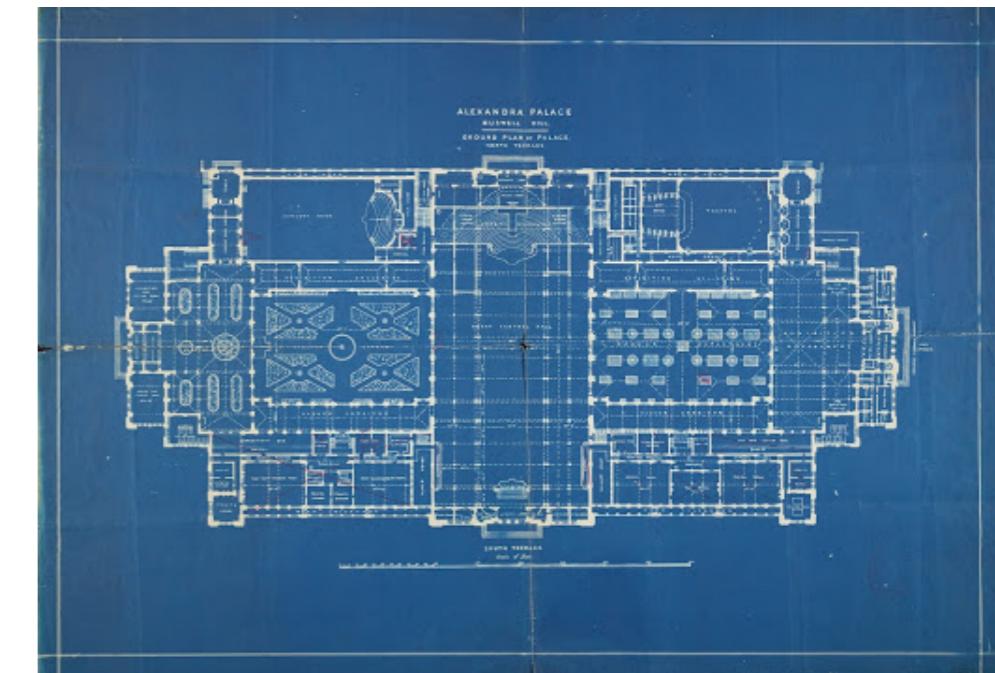
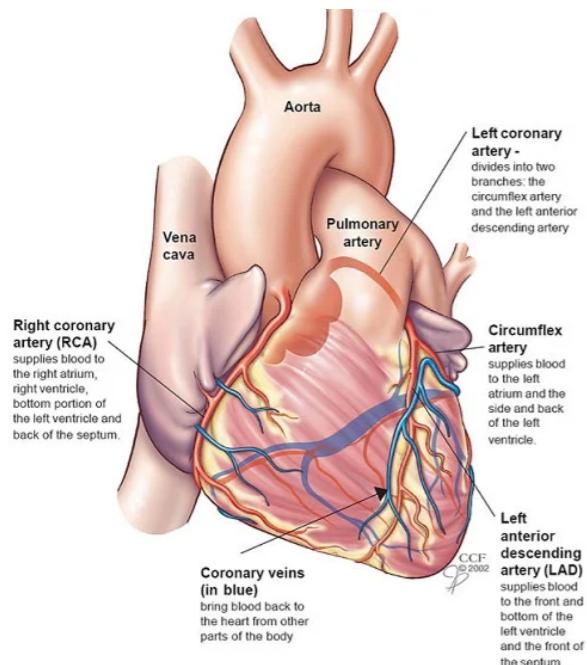
rij/record

kolom

# Modelleren

## Wat is modelleren?

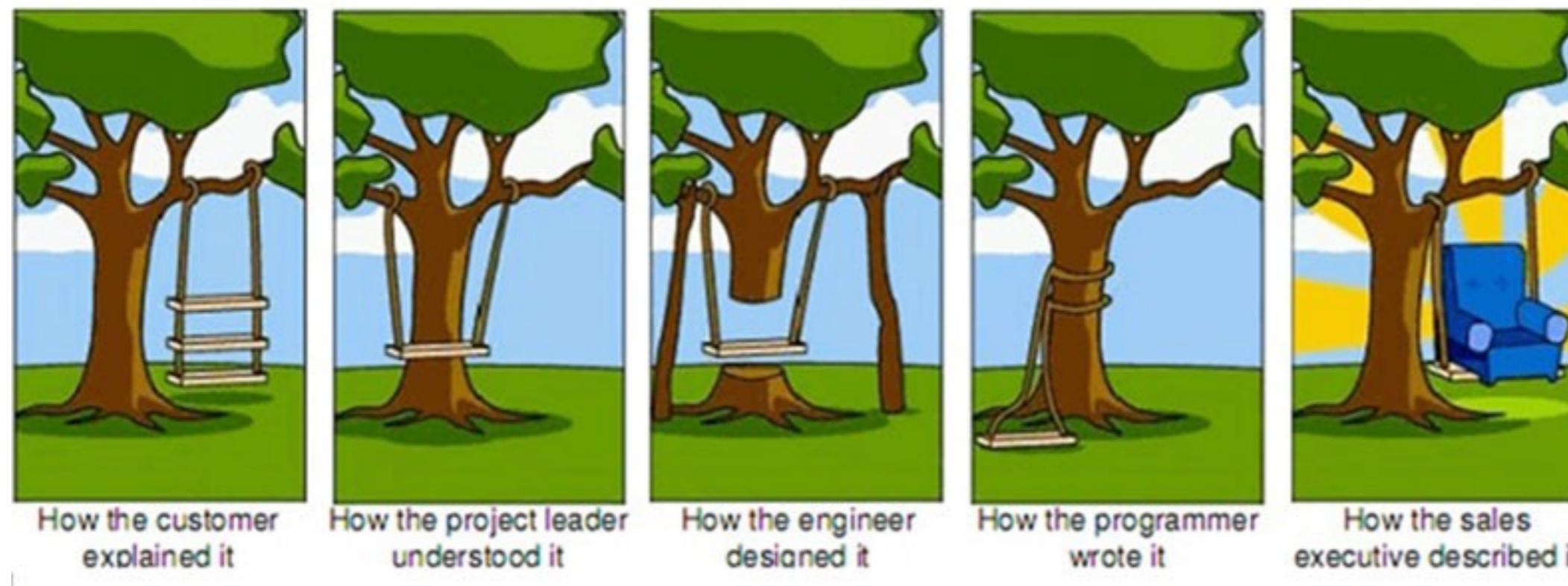
Een model is een schematische weergave van de werkelijkheid. Modellen worden in allerlei vakgebieden gebruikt (Architect & maquettes, bouw & technische tekening/blauwdruk, medici & model hartsstructuur, marketing & sales funnel). In de ICT worden ze voornamelijk gebruikt om de werking van informatieystemen, applicaties, netwerken en apparate te beschrijven.



# Modelleren

## Waarom modelleren?

Modelleren is een goede communicatie tool. Met weinig uitleg kan een model besproken en beoordeeld worden door de beoogde gebruikers.



# Modelleren

## Waarom modelleren in de ICT

### 1. Grip krijgen op onderzoeks domein

Grote/complexe problemen opknippen in deel aspecten.

### 2. Structureren van een aspect

Volgens afgesproken werkwijze wordt een aspect van structuur voorzien. Bij datamodellingen worden bijvoorbeeld entiteittypen, relaties tussen entiteiten en attributen onderscheiden.

### 3. Stepwise refinement

Een model is geschikt om te wijzigen.

### 4. Communicatie

Modellen zijn zo opgezet dat ze een duidelijk overzicht geven van een bepaald aspect van een bepaald vraagstuk. Doordat een model door iedereen makkelijk te lezen is maakt het communiceren hierover een stuk makkelijker.

### 5. Compacte vorm van documentatie

In een model kunnen veel specificaties worden weergegeven die via tekst een uitgebreide omschrijving nodig zullen hebben.

### 6. Eenvoudig om alternatieven aan te geven

Twee modellen met verschillende benaderingen met elkaar vergelijken is makkelijk.

### 7. Eenduidig vastleggen van afspraken

Modellen zorgen voor consistentie en eenduidigheid. Hierdoor is dit uitermate geschikt om afspraken te maken tussen opdrachtgever en opdrachtnemer.

# Modelleren

## Database modelleren

### 1. Informatie verzamelen en modelleren

Bedrijfsprocessen, use cases en activiteitendiagrammen.

### 2. Conceptueel datamodel maken

Begrippen en definities – synoniemen onventariseren, begrippen vaststellen en documenteren, te bewaren gegevens inventariseren, gegevens groeperen en relaties vaststellen, omvang gegevens vaststellen  
→ maken conceptueel datamodel

### 3. Logisch datamodel maken

Tabellen met kolommen en sleutels ontwerpen, Relaties tussen tabellen vaststellen, Redundatie minimaliseren (zelfde data op meerdere plekken)  
→ maken logisch datamodel

### 4. Fysiek datamodel

Datatypes en lengte kolommen vaststellen, Gebruikersrollen definieren, integriteit, consistentie en volledigheid waarborgen.  
→ maken fysiek datamodel en maken DDL script voor het maken van de database

# Modelleren

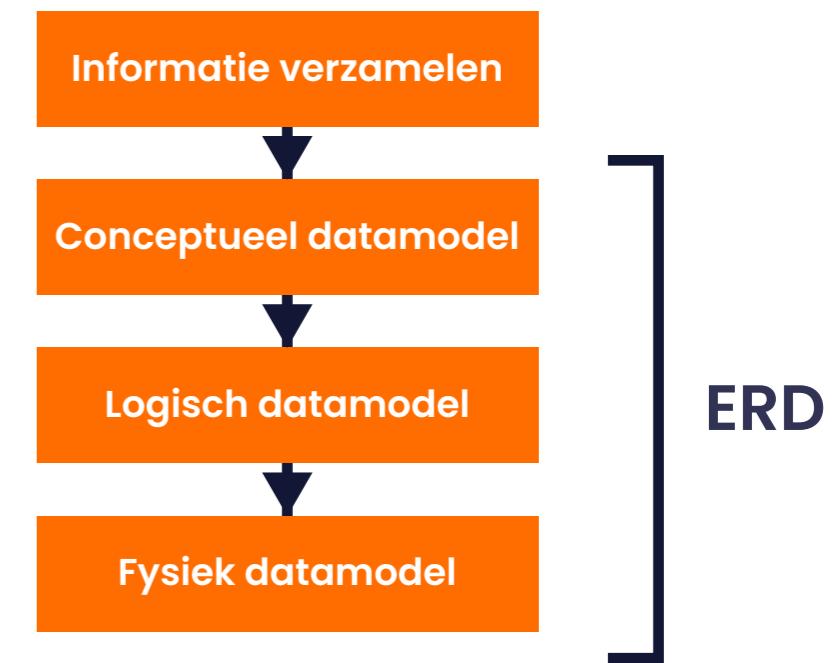
## Entity Relationship Diagram (ERD)

Bij het maken van het conceptueel, logisch en fysiek model gebruiken we een ERD. Een ERD is een communicatiemiddel voor een visuele weergave van een database.

In het conceptuele model gebruiken we entiteiten, attributen en identificaties.

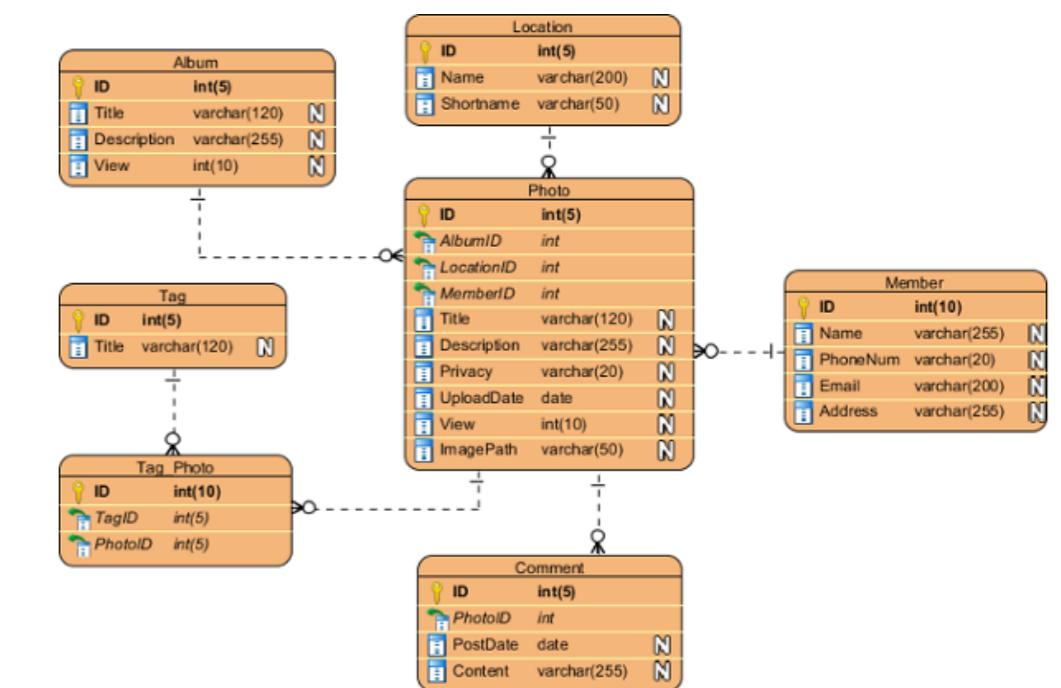
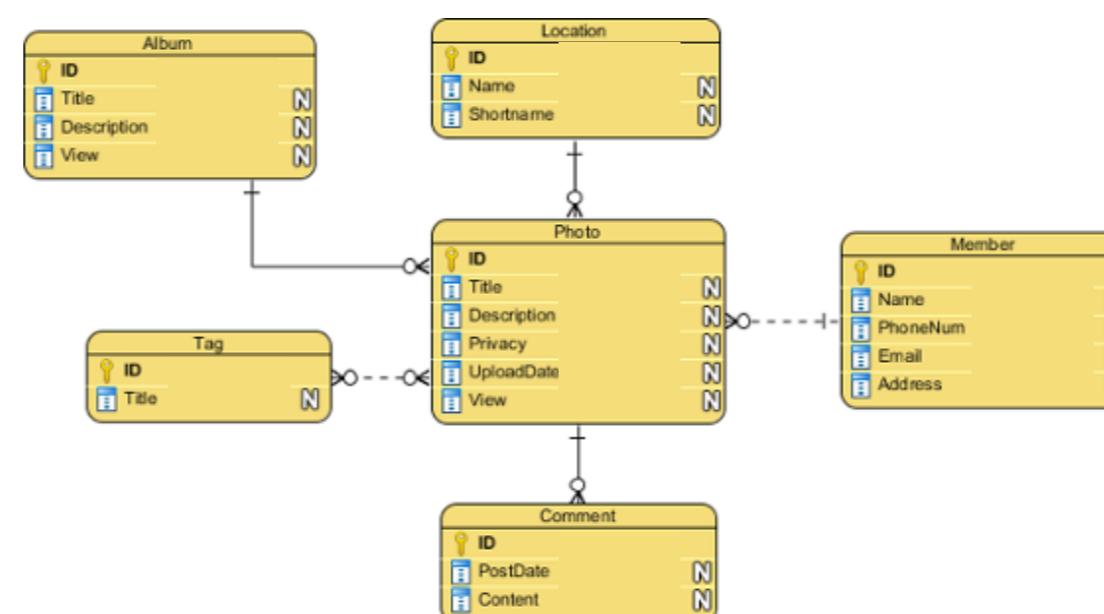
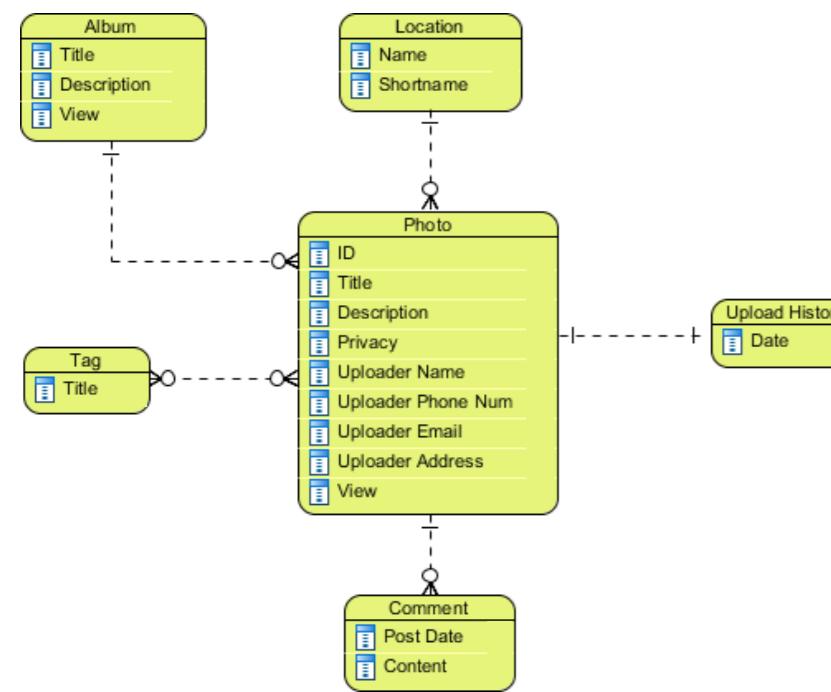
In het logisch model gebruikwen we tabellen, kolommen, constraints en primaire en vreemde sleutels sleutels.

In het fysieke model gebruiken we datatypes.



# Modelleren

## Entity Relationship Diagram (ERD)



1 Conceptueel datamodel

2 Logisch datamodel

3 Fysiek datamodel

# Modelleren

## Conceptueel datamodel

### Belangrijke begrippen

**Entiteit**

Een entiteit kan worden gezien als een object, een abstract iets. Bijvoorbeeld: Werknemer, Student, Aankoop, Auto.

**Attribuut**

Eigenschap van een entiteit. Bijvoorbeeld voornaam, salaris, adres.

**Relatie**

Relatie geeft het verband aan tussen twee entiteiten. Een Klant doet een aankoop. Een werknemer werkt bij een Bedrijf.

# Modelleren

## Visual Paradigm

Er zijn meerdere programma's beschikbaar voor het maken van ICT modellen. Populaire programma's zijn: Lucidchart, Microsoft Visio, Bizagi Modeler, ER studio en Visual Paradigm.

Deze laatste gaan wij gebruiken. Download het programma en gebruik de 30 dagen gratis proefversie.

| <https://www.visual-paradigm.com/download/>

# Modelleren

## Visual Paradigm

### Maken ERD

- 1: new project
- 2: diagram navigator – database modelling
- 3: right-click ERD – new ERD
- 4: drag and drop om te bouwen

# Modelleren

## Conceptueel datamodel

### Stappen maken conceptueel model

1. Entiteiten vinden
2. Attributen vinden
3. Relaties vinden
4. Model tekenen
5. Kardinaliteit van relaties bepalen
6. Identificaties bepalen

# Modelleren

## Conceptueel datamodel

### Voorbeeld casus

Schaakbond RedHotPawn wil automatiseren. Van iedere speler wordt geregistreerd: NAW-gegevens (naam, adres, woonplaats) en geboortedatum. Van de club worden ook de NAW-gegevens en het logo geregistreerd.

Elke speler en elke club heeft een eigen ID-nummer. Er wordt vastgelegd welke speler bij welke club speelt met begindatum en einddatum.

# Modelleren

## Conceptueel datamodel

### Stap 1 entiteiten vinden

Bij het zoeken naar entiteiten kun je uit een geschreven tekst zelfstandig naamwoorden op-schrijven, deze komen vaak in aanmerking als entiteit of attribuut.

Schaakbond RedHotPawn wil automatiseren. Van iedere **speler** wordt geregistreerd: **NAW-gegevens** (naam, adres, woonplaats), **geboortedatum** en **KNSB rating**. Van de **club** worden ook de **NAW-gegevens** en het **logo** geregistreerd.

Elke speler en elke club heeft een eigen **ID-nummer**. Er wordt vastgelegd welke speler bij welke club speelt met **begindatum** en **einddatum**.

**Entiteiten:** speler, club

**Attributen:** NAW-gegevens speler, NAW-gegevens club, geboortedatum, KNSB rating, logo, begin-en einddatum

# Modelleren

## Conceptueel datamodel

### Stap 2 attributen vinden bij entiteiten

**Speler:** ID-nummer, naam, adres, woonplaats, geboortedatum, KNSB rating

**Club:** ID-nummer, naam, adres, plaats, logo

**Wat gebeurt er met begin en einddatum?**

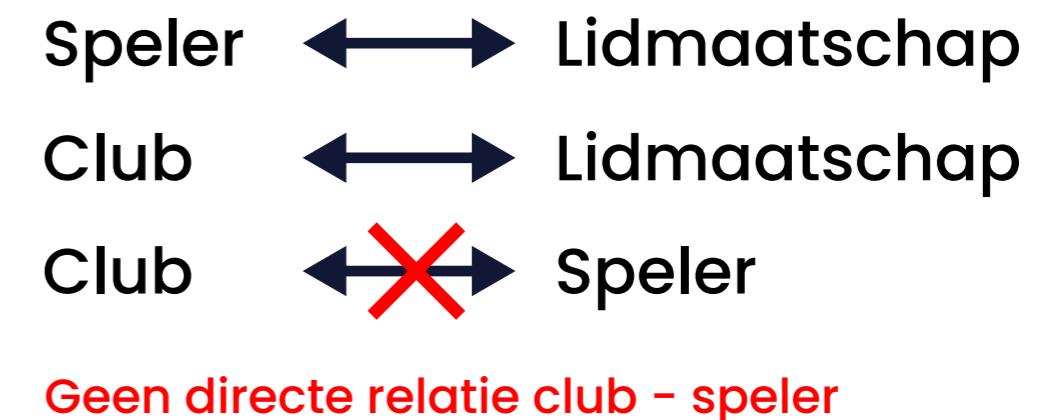
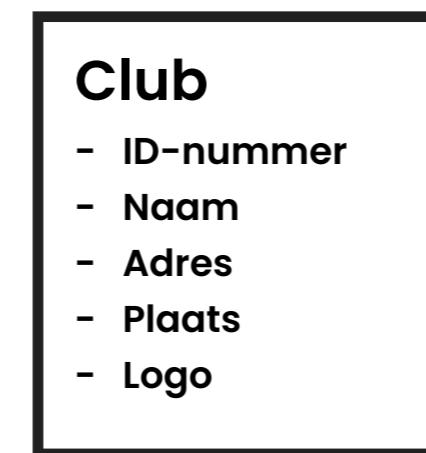
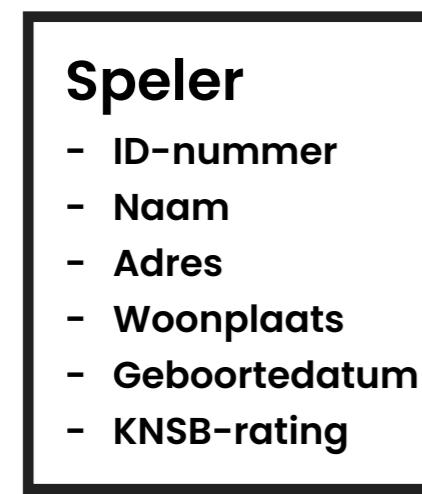
Begin en einddatum zegt niet echt iets over speler en club, maar meer over de relatie tussen beiden. Daarom maken we een nieuwe entiteit: lidmaatschap.

**Lidmaatschap:** Begindatum, Einddatum

# Modelleren

## Conceptueel datamodel

### Stap 3 Relaties tussen entiteiten



Schaakbond RedHotPawn wil automatiseren. Van iedere speler wordt geregistreerd: NAW-gegevens (naam, adres, woonplaats), geboortedatum en KNSB rating. Van de club worden ook de NAW-gegevens en het logo geregistreerd.

Elke speler en elke club heeft een eigen ID-nummer. Er wordt **vastgelegd welke speler bij welke club speelt** met begindatum en einddatum.

# Modelleren

## Conceptueel datamodel

### Stap 4 model tekenen

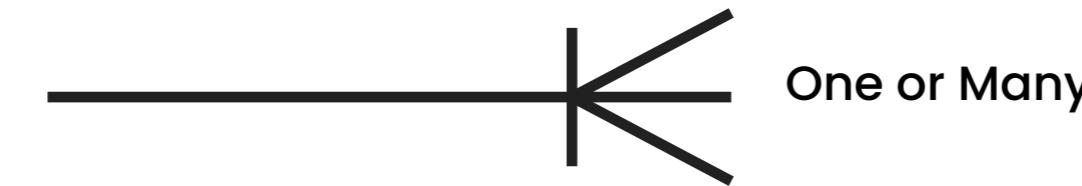
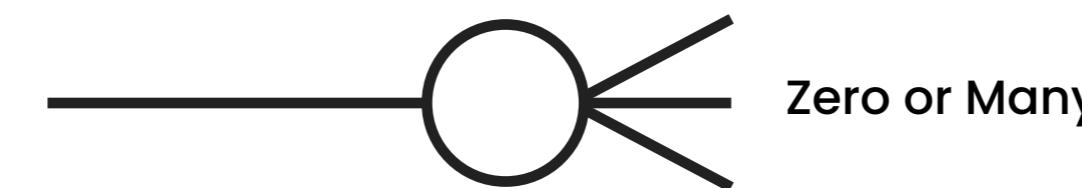
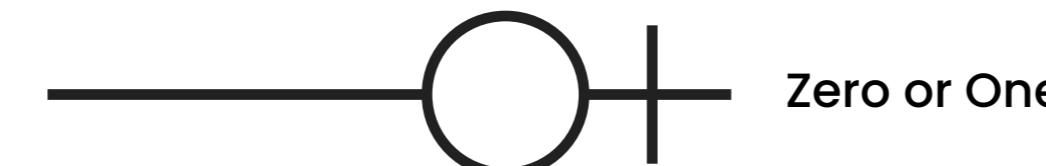


# Modelleren

## Conceptueel datamodel

### Stap 5 kardinaliteit toevoegen

Kardinaliteit geeft het aantal keer dat een relatie voor kan/mag voorkomen. Dit kan zijn:  
0 op veel, 1 op veel, 1 op 1 et cetera.

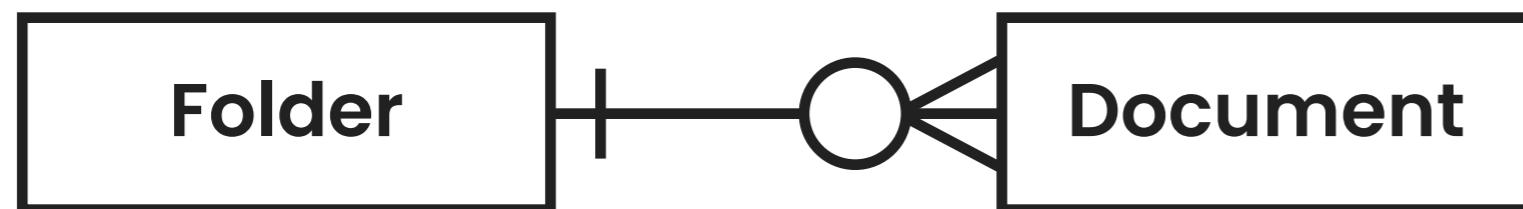


# Modelleren

## Conceptueel datamodel

### Stap 5 kardinaliteit toevoegen

Kardinaliteit werkt 2 kanten op < >. School < > studentent. Een student is ingeschreven bij 1 school. Een school heeft meerdere ingeschreven studenten.



Een folderbevat 0 of meerdere documenten

Een document zit in 1 folder

**Bedenk een aantal relaties uit de echte wereld:**

- 1 op 1
- 1 op veel
- veel op veel

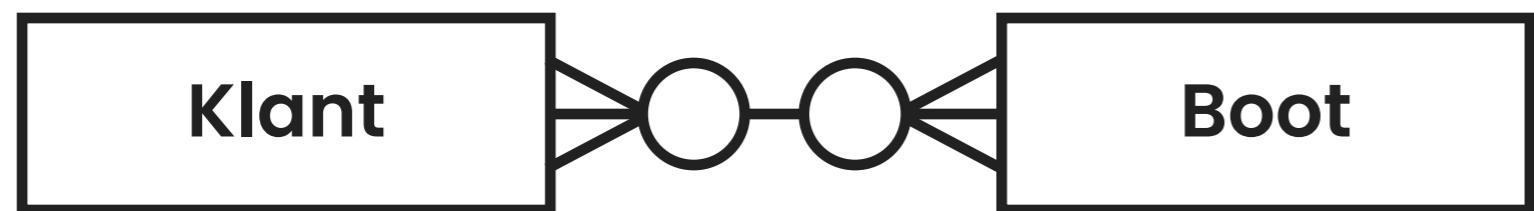
# Modelleren

## Conceptueel datamodel

### Stap 5 kardinaliteit toevoegen

Een kardinaliteit meet niet een bepaald moment maar werkt in de loop van de tijd. Een klant huurt een boot. De klant huurt misschien maar 1 boot tegelijk, maar kan in de loop van de tijd wel meerdere boten hebben gehuurd.

Heb je ergens een 1 op 1 relatie. Ga goed na of dit wel juist is.



# Modelleren

## Conceptueel datamodel

### Stap 5 kardinaliteit toevoegen

Speler	$\longleftrightarrow$	Lidmaatschap	Een speler heeft 0 of meer lidmaatschappen Een lidmaatschap hoort bij 1 speler
Club	$\longleftrightarrow$	Lidmaatschap	Een lidmaatschap hoort bij 1 club Een club heeft 0 of meerdere lidmaatschappen

# Modelleren

## Conceptueel datamodel

### Stap 5 kardinaliteit toevoegen



# Modelleren

## Conceptueel datamodel

### Stap 6 Identificaties toevoegen

Is er een attribuut die bij elke record van een entiteit uniek zal blijven (bv: studentnummer, bsn-nummer, ISDN-nummer, barcode etcetera) dan kun je deze gebruiken als unieke identifier in het conceptueel model.



# Modelleren

## Logisch datamodel

### Belangrijke begrippen

**Tabel**

Entiteiten noemen we tabellen in een relationele database

**Kolom**

Attributen noemen we kolommen in een relationele database

**Primaire sleutel**

Een record van een bepaalde tabel heeft een unieke sleutel.  
Dit noemen we de primary key en dit kan de identifier zijn van  
het conceptueel model.

In visual paradigm laten staan op conceptueel

# Modelleren

## Logisch datamodel

### Stappen maken logisch model

1. Converteer de conceptuele entiteiten met attributen naar relationele tabellen met kolommen
2. Identificeer de primaire sleutel van ieder tabel
3. Vertalen relaties (1 op 1 juist? many to many) naar koppel tabellen met foreign keys)
4. Identificeer de primaire sleutel van iedere childtabel of kopeltabel
5. Voeg constraints toe (not Null of Unique)

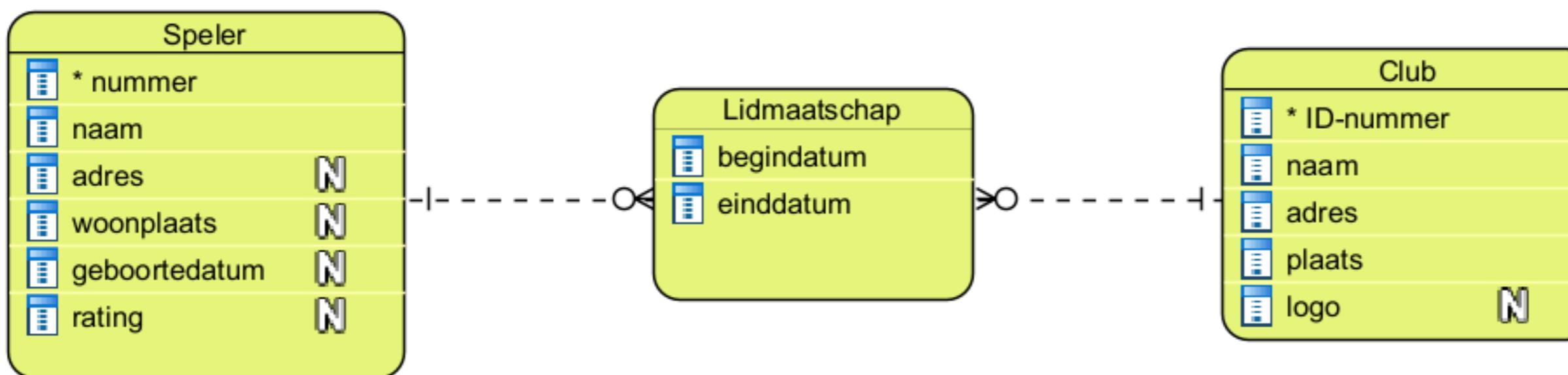
# Modelleren

## Logisch datamodel

### Stap 1 conversie entiteiten naar relationele tabellen

Een entiteit wordt een tabel en een attribuut wordt een kolom van een tabel.

- Voor elk attribuut wordt bepaald of deze optioneel of verplicht is. (Nullable = mag leeg zijn)
- Tabelnaam moet uniek zijn binnen de database
- Kolomnaam moet uniek zijn binnen de tabel
- Namen mogen geen spaties bevatten



# Modelleren

## Logisch datamodel

### Stap 2 Identificatie van primaire sleutels

- 1 Een primaire sleutel is bij elke kolom verplicht (attribuut of combinatie van attributen), deze attributen kunnen dus niet leeg zijn.
- 2 Een primaire sleutel moet uniek zijn
- 3 Hij is zo klein mogelijk, omvat zo min mogelijk attributen en een simpel datatype (integer of string)
- 4 Wordt gebruikt voor relaties met tabellen

**Enkelvoudige  
sleutel**

Primaire sleutel omvat 1 kolom

**Samengestelde /  
brede sleutel**

Meer dan 1 kolom (combinatie)

# Modelleren

## Logisch datamodel

### Stap 2 Identificatie van primaire sleutels

#### Natuurlijke sleutel

Bestaat uit 1 of meer attributen die al worden gebruikt.  
Waarschijnlijk al ontstaan in het conceptueel model.

#### Surrogaatsleutel / technische sleutel

Een kunstmatige kolom (integer) die aan de tabel  
wordt toegevoegd om als primaire sleutel te fungeren.

# Modelleren

## Logisch datamodel

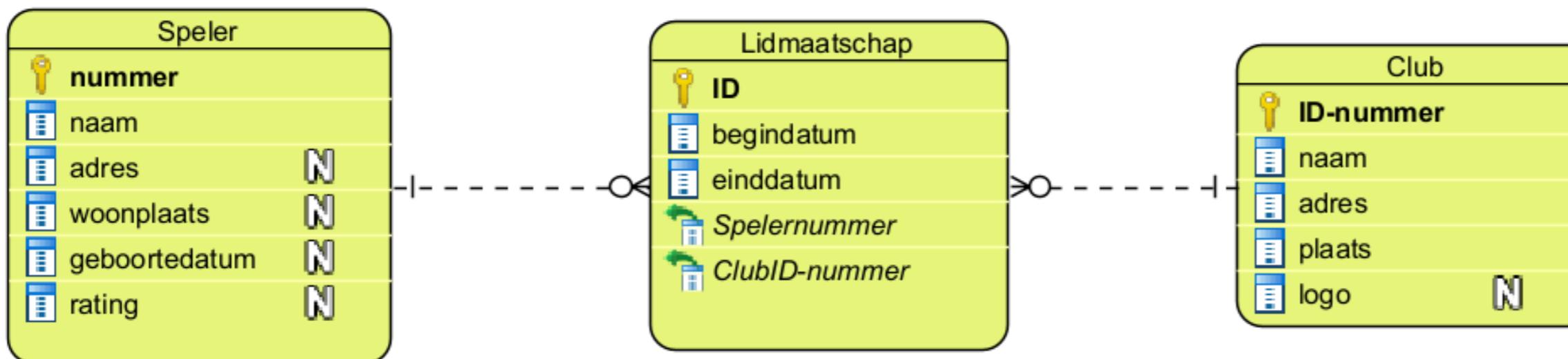
### Stap 2 Identificatie van primaire sleutels

#### Natuurlijke sleutel

Bestaat uit 1 of meer attributen die al worden gebruikt.  
Waarschijnlijk al ontstaan in het conceptueel model.

#### Surrogaatsleutel / technische sleutel

Een kunstmatige kolom (integer) die aan de tabel wordt toegevoegd om als primaire sleutel te fungeren.



# Modelleren

## Logisch datamodel

### Stap 3 Vertalen relaties

- 1 Controleer 1 op 1 relaties. Is dit nodig? Kan het 1 tabel worden? Blijft het een 1 op 1 relatie? Aan welke kant moet de foreign key?
- 2 Many to many relaties zijn niet mogelijk in een database. Hier is een koppeltabel nodig die de relaties gaat bijhouden. Een boek wordt door veel leners geleend. Een lener leent 0 of meerdere boeken. In de tabel boek kunnen we niet angeven door welke leners het boek allemaal geleend is (geen lijstjes mogelijk). Ook in lener kunnen we niet angeven welke boeken hij allemaal geleend heeft.

#### Uitlening

ID	LenerID	BoekID	Datum
0001	2312	AB1231	12-01-2022
0002	2312	AX2235	12-01-2022
0003	6532	AX2235	18-01-2022
0004	4544	AB1231	22-01-2022

# Modelleren

## Logisch datamodel

### Stap 4 Identificeren sleutels van child tabellen en koppeltabellen

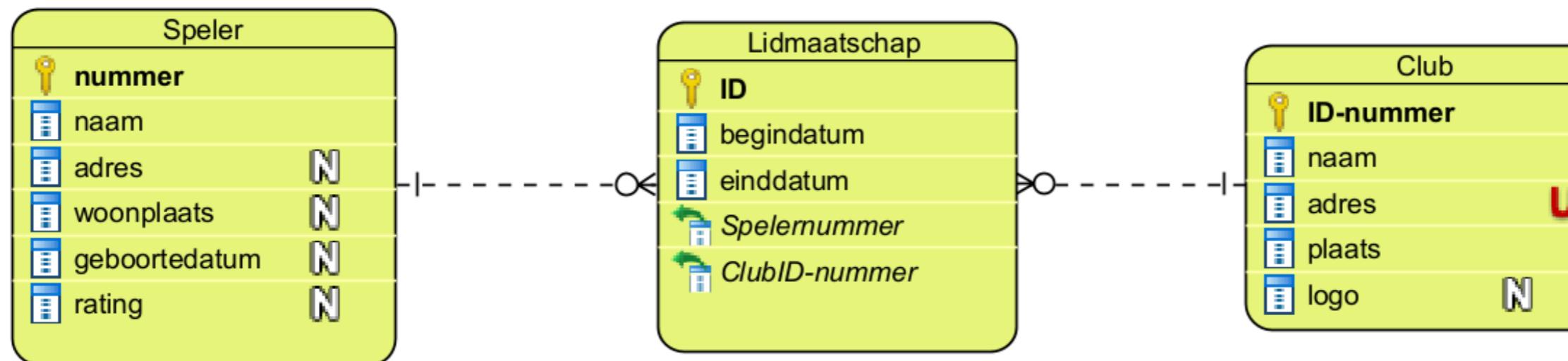
Het kan zijn dat in de voorgaande stappen nieuwe tabellen zijn toegevoegd. Ook deze tabellen moeten een primaire sleutel hebben. Nu is het de vraag waar komen nu de vreemde sleutels? De childtabel krijgt de vreemde sleutel richting de primary key van de parenttabel. Een opleiding heeft meerdere studenten. Een student heeft 1 opleiding. Student is een childtabel van de parent opleiding.

# Modelleren

## Logisch datamodel

### Stap 5 Constraints

- Primary key: Als er 1 sleutelkolom is mag de waarde slechts één keer voorkomen. Als de sleutel uit meerdere kolommen bestaat, mag de combinatie van waarden slechts één keer voorkomen.
- Foreign key: De foreign key moet overeenkomen met de waarde van de primary key in de record waar een relatie mee is.
- Not Null / Nullabl: In dit model gebruiken we Nullable dit geeft aan dat een kolom leeg mag zijn. Not Null in de database is het tegenovergestelde.
- Unique: Geeft aan dat de waarde in de kolom uniek moet zijn.



# Modelleren

## Fysiek datamodel

### Keuze DBMS

Bij een fysiek datamodel bepaal je welke datatypes de kolommen krijgen. Hiervoor is het belangrijk om te weten welke datatypen het gekozen DBMS ondersteunt. Wij kiezen voor PostgreSQL. Dit merk kent onder andere:

**INTEGER**

**TIMESTAMP (datum en tijd)**

**NUMERIC(n)**

**CHAR(n)**

**BOOLEAN**

**VARCHAR(n)**

**DATE**

**BYETA (of BLOB voor plaatjes etc)**

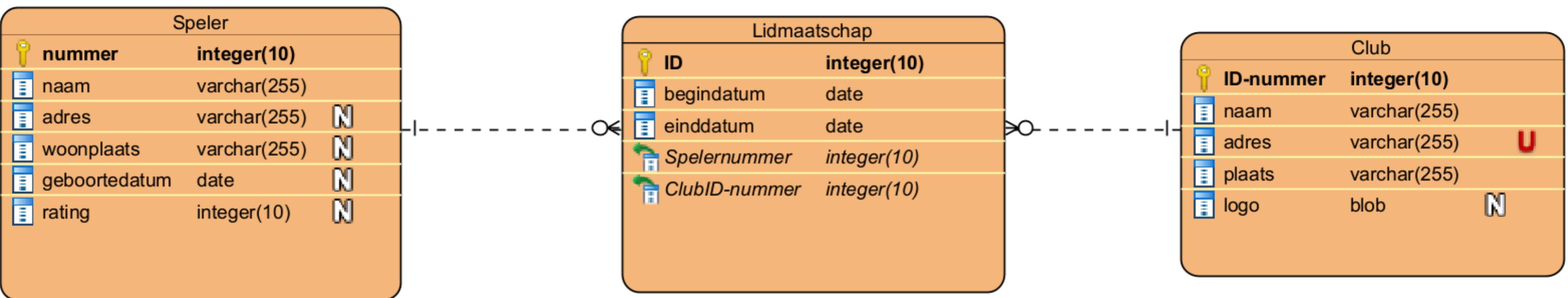
**TIME**

# Modelleren

## Fysiek datamodel

ERD aanpassen naar fysiek model

In visual paradigm tabellen op fysiek zetten



# Modelleren / SQL

## Maken SQL DDL script

### Wat is een DDL script?

Data Definition Language (DDL) is een SQL syntax voor het aamaken van tabellen en meer.

### Maken database

Een database aanmaken kan op vele manieren. Wij gaan het programma pgadmin gebruiken voor het maken van een PostgreSQL database.

| <https://www.pgadmin.org/download/>

PGADMIN sloom?

| <https://stackoverflow.com/questions/62186945/why-pgadmin-4-is-too-slow>

# Modelleren / SQL

## Maken SQL DDL script

### Maken database

In PGADMIN:

- right click op PostgreSQL (14)
- Create -> Database
- Voer naam in -> Save
- Right click nieuwe database -> Query Tool

### Maken database met SQL DDL

```
CREATE DATABASE Databasenaam;
```

```
DROP DATABASE Databasenaam;
```

# Modelleren / SQL

## Maken SQL DDL script

Maken tabellen

```
CREATE TABLE speler (
    nummer INTEGER,
    adres VARCHAR(255),
    woonplaats VARCHAR(255),
    geboortedatum DATE,
    rating INTEGER
);
```

Maak zelf Club aan

# Modelleren / SQL

## Maken SQL DDL script

### Tabellen aanpassen

Tabel hernoemen

```
ALTER TABLE speler RENAME TO lid;
```

Kolom toevoegen

```
ALTER TABLE speler ADD COLUMN postcode CHAR(6);
```

Kolom hernoemen

```
ALTER TABLE speler RENAME COLUMN woonplaats TO plaats;
```

# Modelleren / SQL

## Maken SQL DDL script

### Tabellen aanpassen

Kolom verwijderen

```
ALTER TABLE speler DROP COLUMN postcode;
```

# Modelleren / SQL

## Maken SQL DDL script

### Constraints gebruiken

Bij aanmaken tabel

```
CREATE TABLE klant(  
    nummer INTEGER,  
    adres VARCHAR(255) NOT NULL,  
    woonplaats VARCHAR(255) NOT NULL,  
    email VARCHAR(255) UNIQUE,  
);
```

# Modelleren / SQL

## Maken SQL DDL script

### Constraints gebruiken

Achteraf

```
ALTER TABLE klant  
ADD CONSTRAINT email_unique UNIQUE (email);
```



Naam constraint

# Modelleren / SQL

## Maken SQL DDL script

### Constraints gebruiken – Primaire sleutel

Enkelvoudig bij aanmaken

```
CREATE TABLE speler (
    nummer INTEGER PRIMARY KEY,
    ..etc
);
```

Enkelvoudig achteraf

```
ALTER TABLE klant
ADD PRIMARY KEY (nummer);
```

# Modelleren / SQL

## Maken SQL DDL script

### Constraints gebruiken – Primaire sleutel

Samengesteld bij aanmaken

```
CREATE TABLE orderregel (
    orderid INTEGER,
    klantid INTEGER,
    ...etc,
    PRIMARY KEY (orderid, klantid)
);
```

Samengesteld achteraf

```
ALTER TABLE orderregel
ADD PRIMARY KEY (orderid, klantid);
```

# Modelleren / SQL

## Maken SQL DDL script

Constraints gebruiken – Primaire sleutel – Surrogaatsleutel

Samengesteld bij aanmaken

```
CREATE TABLE lidmaatschap(  
    id SERIAL NOT NULL,  
    ... etc,  
    PRIMARY KEY (id)  
);
```

**Nieuwsgierig?**  
Zelf uitzoeken CREATE SEQUENCE

# Modelleren / SQL

## Maken SQL DDL script

### Constraints gebruiken – Vreemde sleutel

Bij aanmaken

```
CREATE TABLE lidmaatschap(  
    ... etc,  
    spelernr INTEGER REFERENCES speler(spelernr)  
);
```

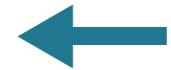
# Modelleren / SQL

## Maken SQL DDL script

### Constraints gebruiken – Vreemde sleutel

Achteraf

```
ALTER TABLE lidmaatschap(  
    ADD CONSTRAINT lidmaatschap_spelernr_fk  
        FOREIGN KEY (spelernr) REFERENCES speler(spelernr);
```



Naam constraint

Achteraf is beter omdat de primariy key en table waar je naartoe verwijst misschien nog niet bestaat.

# Modelleren / SQL

## Opdracht casus

### Pokerclub de Rode Vrouw

Een pokerclub wil hun leden en toernooien bijhouden in een database. De pokerclub wil van zijn spelers de NAW gegevens opslaan. Van elk toernooi willen ze bijhouden welke leden deelnemen, wat de prijzenpot is en wie er heeft gewonnen (voor gemak winner takes all). Van elk lid wordt ook bijgehouden de wanneer hij/zij lid is geworden en wanneer lidmaatschap is gestopt.

- Maak een conceptueel, logisch en fysiek model aan de hand van de stappen.
- Maak een DDL script