

# Databanken

# Introductie

# gegevens vs informatie

**GEGEVENS:**

**10**

**14**

**15**

---

**INFORMATIE:**

Brussel	februari	maart	april
Gemiddelde temperatuur	<b>10</b>	<b>14</b>	<b>15</b>

# samengestelde gegevens

= verzameling van elementaire gegevens

voorbeeld: **adres**

**adres** = (straat, huisnr, postnr, woonplaats)

## berekende gegevens

- worden afgeleid uit andere gegevens
- worden niet opgenomen in een database

voorbeeld: **totaalprijs**

**totaalprijs** = eenheidsprijs x aantal

# gegevens verzameling

bevat een aantal gegevens die logisch bij elkaar horen

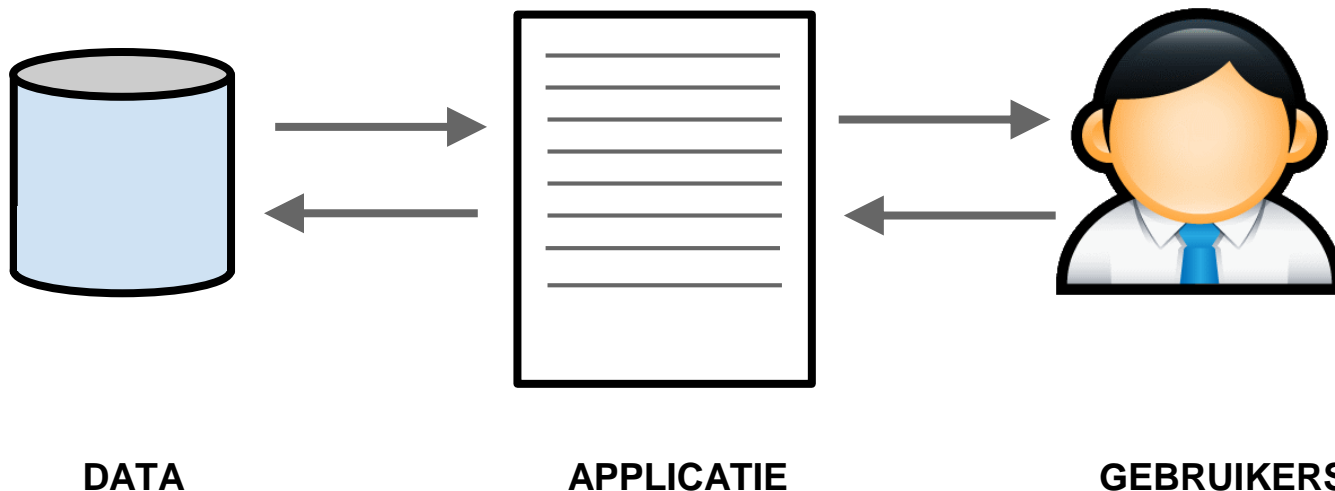
voorbeeld: **Een kasboek**

Een winkelier(ster) houdt zijn/haar kosten en inkomsten bij in **een kasboek**. De kasboek is hier een gegevensverzameling.

# database

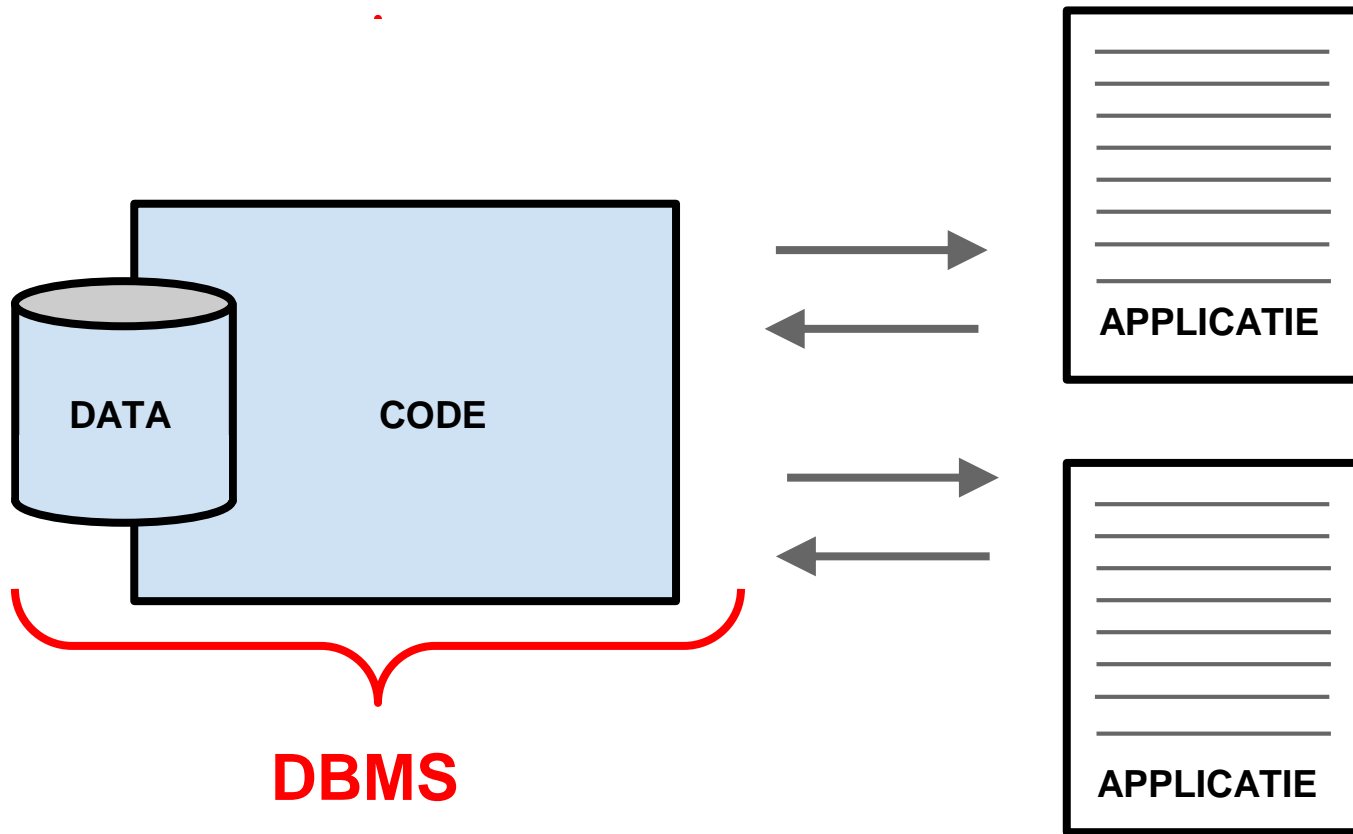
Wordt gebruikt om gegevensverzamelingen te bewaren.

Een **database** is een gecentraliseerde, gestructureerde set gegevens die op de computer bewaard wordt.



# Database Management System (DBMS)

= Data + code die de data aanspreekt en beheert





# Vereisten voor een DBMS

- Simultaan gebruik van gegevens
- Beveiliging van toegang
- Beveiliging van gegevens
- Gedistribueerde gegevens (= gegevens zijn verdeeld over sub-databases maar worden centraal beheerd)

# Soorten DBMS

- Hiërarchische databanken
  - Eén root element
  - Pointers om relatie weer te geven
- Netwerk databanken
  - Meerdere toegangen tot gegevens
- **Relationele databank**
  - Alle gegevens in tabellen
- Objectgeoriënteerde databanken
- XML databanken

# Metadata en data

## METADATA

Cursus: id (int, uniek) en titel (max 20 karakters)

Cursist: id (int, uniek) en naam (max 25 karakters)

Inschrijving: id (int, uniek), cursusid (int gekoppeld aan id in cursus), cursistid (int gekoppeld aan id in cursist)

Iedereen mag alle gegevens lezen.  
Cursusgegevens mogen alleen gewijzigd worden door cursusbeheerders.

## DATA

1	databanken
2	SQL

1	Jan
2	Rita

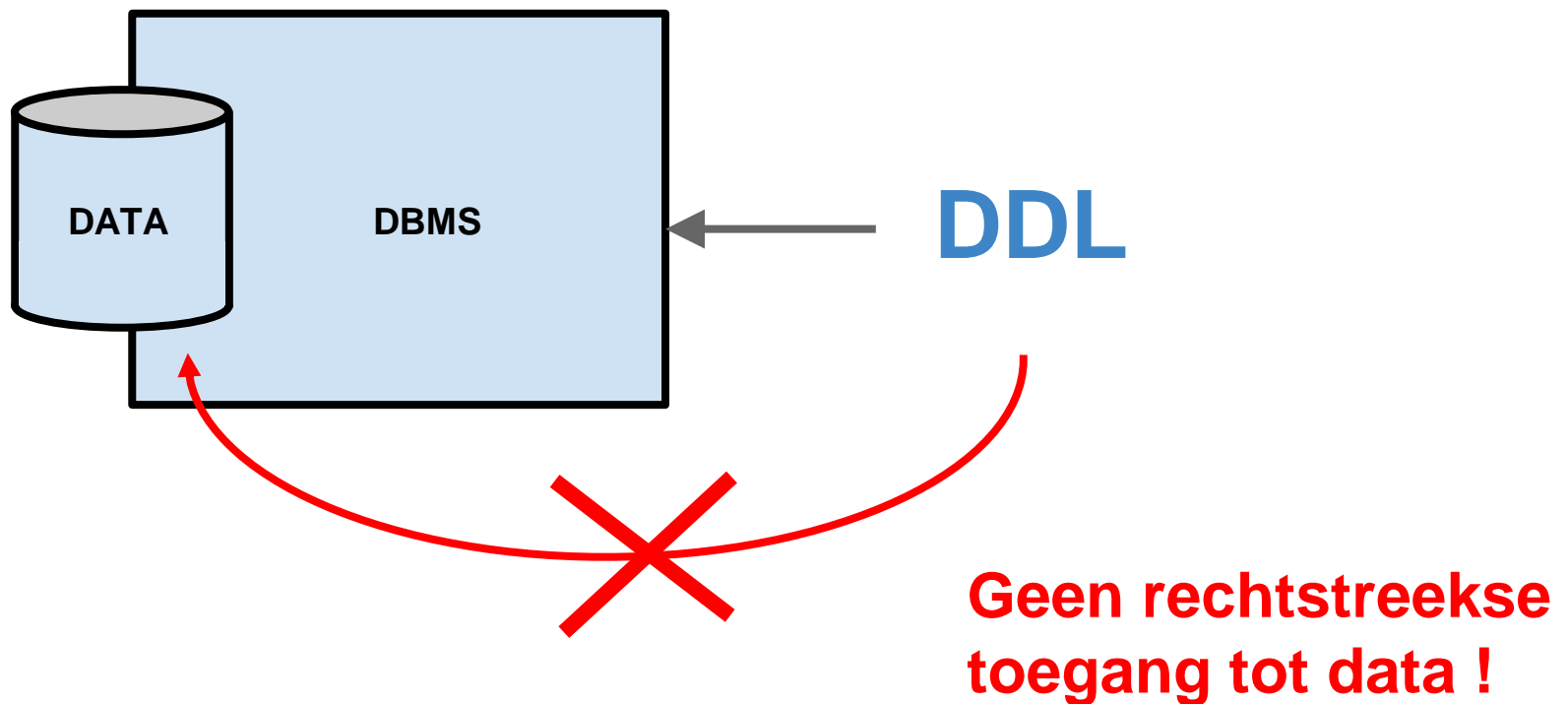
1	1	1
2	2	1
3	1	2

# Data Definition Language (DDL)

- Taal om metadata te definiëren
- Beschrijving van de velden (naam, type, lengte, beperkingen)
- Definitie van recordstructuur (tabellen)
- Relaties tussen de gegevens
- Definiëren van de rechten

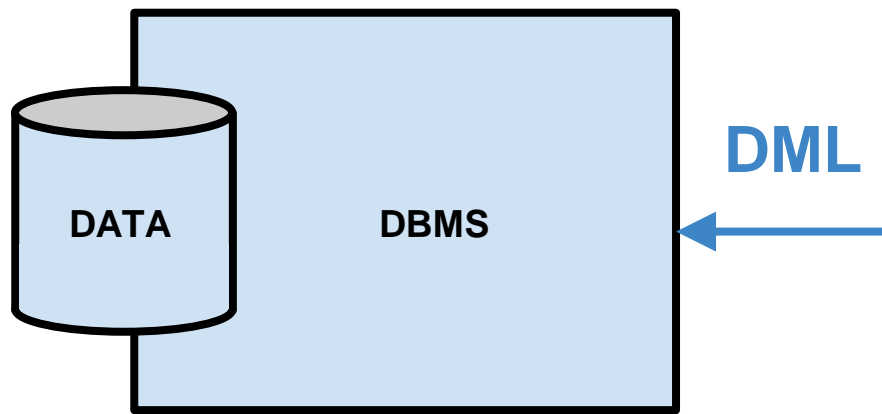
# Data Definition Language

De **Data Definition Language** dient om aan het DBMS de opdracht geven metadata aan te maken, aan te passen of te verwijderen.



# Data Manipulation Language

De **Data Manipulation Language** is ofwel een onderdeel van de programmeertaal ofwel een aparte (query) taal.



- Voeg een cursist toe.
- Verander de titel van de cursus met als id is 1.
- Verwijder de inschrijving voor cursist 1 voor de cursus 2.
- Vraag een lijst op van alle cursisten die deelnemen aan een cursus.

# Modelleren

# datamodel

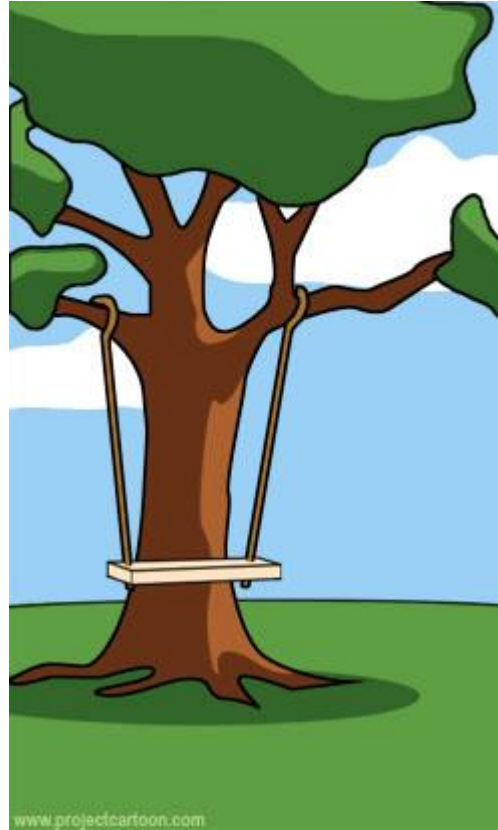
Met een **datamodel** (of gegevensmodel) wordt beschreven hoe de gegevens in een informatiesysteem gestructureerd zijn.



# Waarom een datamodel?



Hoe de klant het  
uitlegde

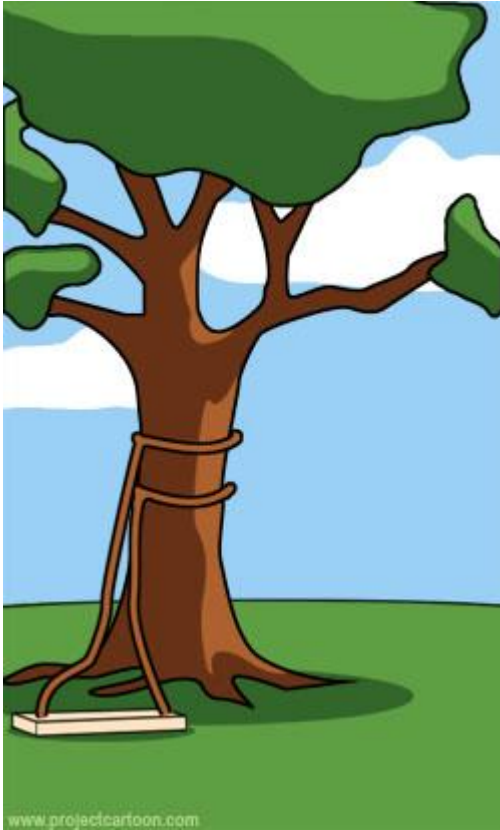


Hoe de projectleider  
het begreep



Hoe de analist  
het ontwierp

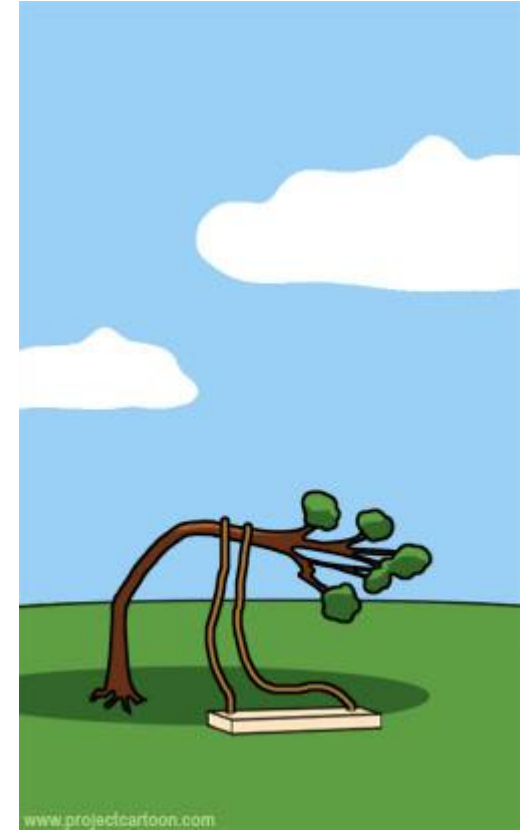
# Waarom een datamodel?



Hoe de  
programmeur het  
schreef



Wat de klant  
werkelijk nodig had



Het rampenplan

# Hoe modelleren?

1. Door gebruik te maken van een ERM of Entity Relationship Model.  
→ Visuele weergave van de entiteiten en relaties
2. Door de gegevens te normaliseren volgens de regels van Codd.  
→ Normaalvormen

# Relationele databases

# Relationele databases

- Voor een gebruiker (beheerder) bestaat een relationele databank uit een **verzameling van tabellen** (en niets anders dan tabellen)
- In een relationele databank worden **entiteiten** bijgehouden (*cursist, cursus*) en **relaties tussen die entiteiten** (*cursist Jan is ingeschreven voor cursus databanken*)
- Zowel entiteiten als relaties worden bijgehouden in tabellen

# Relationele databanken

**Cursustabel**

id	titel
1	databanken
2	SQL

**Cursisttabel**

id	naam
1	Jan
2	Rita

**Inschrijvingtabel**

id	cursistid	cursusid
1	1	1
2	2	1
3	1	2

Cursist Jan is ingeschreven voor de cursussen “databanken” en “SQL”. Cursiste Rita is ingeschreven voor de cursus “databanken”

# Tabellen

- **Projectie**  
= kolommen beperken
- **Selectie**  
= rijen selecteren
- **Join**  
= samenvoegen van verschillende tabellen

# Tabellen

- Tabellen bestaan uit rijen (**records**) en kolommen (**velden**)
- Elke kolom in een tabel heeft een **unieke naam** (in die tabel)
- Een kolom bevat een bepaald **gegevenstype**: int, float, tekst,...
- Records in een relationele databank hebben **geen volgorde**



# De primaire sleutel (Primary Key of PK)

Elke rij (record) in een tabel moet aangesproken worden via een sleutel

Moet **uniek** zijn voor elke record

- Uniek veld aanwezig? → **primaire sleutel**
- Combinatie van velden uniek? → **gecombineerde primaire sleutel**
- Andere gevallen: een **extra uniek veld** toevoegen. (dikwijls met autonummering)

# Index

- Op de primary key wordt automatisch een index geplaatst.
- Vergelijkbaar met een inhoudstafel waarbij elke lijn een verwijzing bevat naar de rest van de uitleg.
- Wordt gebruikt om snel gegevens op te zoeken.

# De Foreign Key (FK)

= een attribuut in de tabel die verwijst naar een andere tabel waar deze attribuut tevens voorkomt en de primary key vormt van die andere tabel.

**Voorbeeld:** Een klantnr in tabel facturen verwijst naar het klantnr in de tabel klanten. Het klantnr van de tabel klanten is de PK van de tabel.

# Entity Relationship Model

# Het Entity Relationship Model (ERM)

- = een gedetailleerde, logische beschrijving van alle data gebruikt in een bepaald project.
- Wordt voorgesteld door een Entity Relationship Diagram (ERD)
- Bestaat uit entiteiten, attributen en relaties

# Entiteiten

= elke persoon, plaats, gebeurtenis, concept, enz... waarover de gebruiker gegevens wil bijhouden.

# Attributen

Elke entiteit heeft één of meerdere kenmerken of attributen.

# Relaties

Relaties geven het verband weer tussen entiteiten.

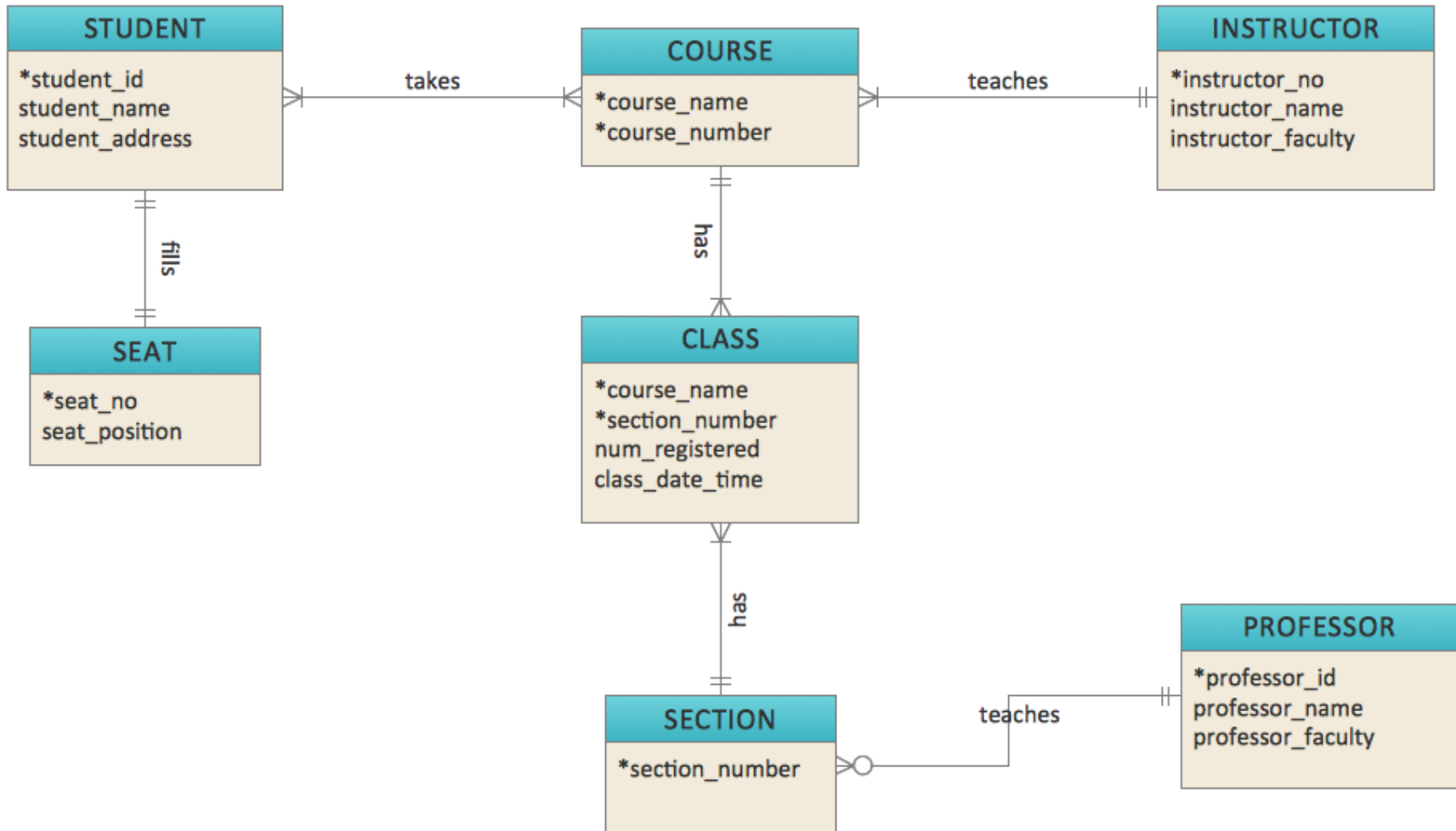
- **1:1 relaties (één op één relaties)** Elke instantie van een bepaalde entiteit komt maximaal met één instantie van de andere entiteit overeen.
- **1:N relaties (één op veel relaties)** Voor elke entiteit zijn er meerdere instanties van de andere entiteit mogelijk
- **N:N relaties (veel op veel relaties)** Voor elke entiteit zijn er meerdere instanties van de andere entiteit en vice versa



# Relaties: grafische voorstelling

1-op-1	1-op-veel	veel-op-veel
		
Met één instantie van de ene entiteit komt maximaal één instantie van de andere entiteit overeen.	Met één instantie van de ene entiteit komen één of meerdere instanties van de andere entiteit overeen	Met één of meerdere instant van de ene entiteit komen één of meerdere instanties van de andere entiteit overeen.
		

# Voorbeeld: ERD



# Normalisatie

# Databank ontwerp

Een goed ontwerp beantwoordt aan de drie normaalvormen:

**1ste normaalvorm:** velden mogen alleen enkelvoudige attributen bevatten

**2de normaalvorm:** in een tabel met een samengestelde primaire sleutel moet elk attribuut afhankelijk zijn van de volledige primaire sleutel

**3de normaalvorm:** attributen mogen alleen afhankelijk zijn van de primaire sleutel, niet van de andere attributen

# Voorbeeld normaliseren: De bestelbon



**PIZZERIA**

**LUIGI**

**Order nr:** 12034  
**Datum:** 13/01/2014

Klantnr: 00259

Alberto Pinacolada  
Steenbrugstraat 135  
1930 Zaventem

artikelnr	omschrijving	aantal	prijs	totaal
10015	Pizza Polo	1	€7,00	€7,00
10280	Pizza Margeritte	3	€6,50	€19,50
10147	Pizza Mozzarella en tomaat	2	€6,50	€13,00

**TOTAAL**

€49,50

# Voorbereiding: de nulde normaalvorm

## ORDERS

ordernr

orderdatum

klantnr

klantnaam

adres

postcode

plaats

artnr

artomschrijving

aantal

prijs

regeltotaal

eindtotaal

Eerst moeten we de informatiebehoefte gaan inventariseren.

Alle gegevens op de bestelbon onder elkaar zetten. Bovenaan plaatsen we in hoofdletters de naam van deze lijst.

De aldus bekomen lijst wordt ook wel de nulde normaalvorm genoemd.

# 1ste normaalvorm

## ORDERS

ordernr  
orderdatum  
klantnr  
klantnaam  
adres  
postcode  
plaats  
artnr  
artomschrijving  
aantal  
prijs

~~regeltotaal  
eindtotaal~~

**Stap 1:** Verwijder alle  
procesgegevens

Ter herinnering:

**1ste**

**normaalvorm:**

velden mogen alleen  
enkelvoudige  
attributen bevatten

# 1ste normaalvorm

## ORDERS

ordernr

orderdatum

klantnr

klantnaam

adres

postcode

plaats

artnr

artomschrijving

aantal

prijs

**Stap 2: Geef de sleutel aan**

(Onderlijn het uniek veld dat het order kan identificeren)



# 1ste normaalvorm

## ORDERS

ordernr

orderdatum

klantnr

klantnaam

adres

postcode

plaats

RG artnr

RG artomschrijving

RG aantal

RG prijs

**Stap 3:** Geef de deilverzameling aan die een herhaald aantal keren voorkomt t.o.v. de primaire sleutel.

Deilverzameling

= RG

= Repeterende Groep

# 1ste normaalvorm

## ORDERS

ordernr  
orderdatum  
klantnr  
klantnaam  
adres  
postcode  
plaats

## BESTELDE\_ARTIKELEN

ordernr  
artnr  
artomschrijving  
aantal  
prijs



Stap 4: Stop de repeterende groepen in een nieuwe lijst en voeg er de sleutel van de oorspronkelijke groep bij

# 1ste normaalvorm

## ORDERS

ordernr  
orderdatum  
klantnr  
klantnaam  
adres  
postcode  
plaats

## BESTELDE\_ARTIKELEN

ordernr  
artnr  
artomschrijving  
aantal  
prijs

Stap 5: Bepaal de sleutel van de nieuwe tabel

Ter herinnering:

**1ste normaalvorm:**  
velden mogen alleen  
enkelvoudige attributen  
bevatten

# 2de Normalvorm

## ORDERS

ordernr

orderdatum

klantnr

klantnaam

adres

postcode

plaats

In een tabel met een samengestelde primaire sleutel moet elk attribuut afhankelijk zijn van de volledige primaire sleutel

## BESTELDE\_ARTIKELEN

ordernr

artnr

aantal

## ARTIKELEN

artnr

artomschrijving

prijs

# 3de Normalvorm

## ORDERS

ordernr

orderdatum

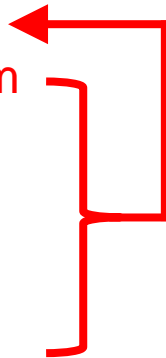
**klantnr**

klantnaam

adres

postcode

plaats



Attributen mogen alleen afhankelijk zijn van de primaire sleutel, niet van de andere attributen

## BESTELDE\_ARTIKELEN

ordernr

artnr

aantal

## ARTIKELEN

artnr

artomschrijving

prijs

# 3de Normalvorm

## ORDERS

ordernr  
orderdatum  
klantr

Attributen mogen alleen afhankelijk zijn van de primaire sleutel, niet van de andere attributen

## KLANTEN

klantr  
klientnaam  
adres  
postcode  
plaats

## BESTELDE\_ARTIKELEN

ordernr  
artnr  
aantal

## ARTIKELEN

artnr  
artomschrijving  
prijs

# Oefening

We houden klanten bij in een databank. Een klant heeft een klantnummer, een naam, een adres, een postnummer en een gemeente. Voor elke klant willen we ook een aantal telefoonnummers bijhouden. (aantal ligt niet vast). Ontwerp de database.

# Oplossing

## **tabel klant**

klnr int

naam varchar(50)

postnr char(4)

adres varchar(100)

## **tabel postnummer**

postnr char(4)

gemeente varchar(50)

## **tabel telefoon**

klnr int

telefoon char(12)