UNIVERSITETET
I OSLO

INF1300
Introduksjon til databaser

Dagens tema:

- Data, databaser og databasehåndteringssystemer
- Data versus informasjon
- Beskrivelse av interesseområdet
- Begreper og representasjon av dem
- Elementære setninger

Transiente og persistente data

- Når vi programmerer, legger vi dataene våre i programvariable (eller bare «variable»)
- Når programmet avsluttes, slettes dataene
- Slike data kalles **transiente**
- Data vi vil ta vare på, er dere vant til å skrive til en fil, det vil vanligvis si at dataene bak kulissene lagres på en magnetdisk
- Data som overlever mellom to programkjøringer, kalles **persistente**

Problemer med filer

- Filer er i allminnelighet tilpasset ett bestemt program
- Uansett kan en fil bare brukes av ett program om gangen (programmet må «åpne» filen for å bruke den, og da er filen sperret for alle andre)
- Store programmer trenger ofte mange typer data, noe som gjerne resulterer i mange filer, noe som igjen gjør programmet enda mer komplisert

Databaser og DBMS

- En **database** er en samling persistente data som fysisk bare kan aksesseres fra ett spesialprogram, kalt et databasehåndteringssystem, forkortet **DBMS** (Data Base Management System)
- Vilkårlig mange programmer og brukere kan samtidig være koblet opp mot DBMSet og via det få lest, skrevet og endret data i databasen
- Et slikt program kalles ofte en klient
- Logisk sett kan følgelig vilkårlig mange klienter aksessere dataene i databasen samtidig

Transaksjoner

- En henvendelse fra en klient til DBMSet kalles en **transaksjon**
- En transaksjon som bare leser data, kalles en lesetransaksjon eller en **spørring** (**query**)
- En transaksjon som legger til nye data eller forandrer eksisterende data, kalles en skrivetransaksjon

Litt historie

- Det første DBMS ble laget og presentert av Bachman og Williams i 1964
- DBMSet ble solgt under navnet IDMS
- IDMS var en nettverksdatabase, og var designet for bruk fra et programmeringsspråk (vertsspråk)
- I 1968 kom IBM med IMS som var en hierarkisk database som er en forenkling av nettverksdatabaser
- IMS ble nær enerådende for administrativ data-behandling, og fortsatt er det svært mange store firmaer som har legacy-systemer som bruker IMS

Litt mer historie

- I 1970 presenterte E.F.Codd sin relasjonsmodell
- Dette var en teoretisk beskrivelse av en ny type databaser kalt relasjonsdatabaser
- Relasjonsdatabaser er enkle å beskrive og bruke, men vanskelige å lage DBMS for
- Først i 1977 klarte Oracle å lage et DBMS som fortjener betegnelsen relasjonell
- Relasjonsdatabaser er nå svært mye brukt, og de er hovedtemaet for dette kurset

Data

- Fra programmeringsspråk er vi vant til at vi har forskjellige datatyper som
  - int (heltall)
  - double (desimaltall)
  - char (tegn)
- Heltallsvariable kan ha verdier som 17 og -1024
- Slike verdier kaller vi **data**
- En enkelt verdi heter egentlig et **datum**, men vi bruker som oftest ordet «data» i entall også

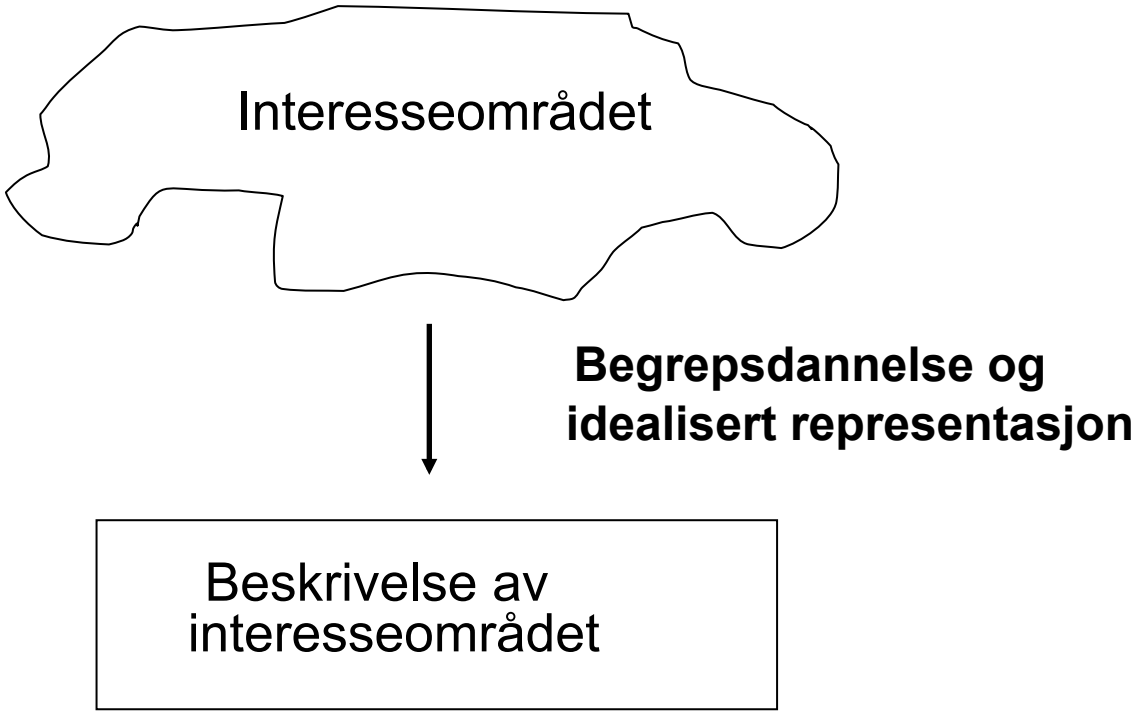
Informasjon

- **Informasjon** består av data pluss regler for hvordan data skal tolkes
- Hvis vi har en (desimaltalls-) variabel *vekt* med verdien 54,2, vil det være naturlig å anta at 54,2 er vekten av ett eller annet
- Men for at 54,2 skal kunne kalles informasjon, er det to spørsmål som må besvares:
  - Hvilken måleenhet er brukt for vekt?
  - Hva er det som veier 54,2 måleenheter?

Interesseområdet (UoD = Universe of Discourse)

- Lovene som styrer virkeligheten, kaller vi **forretningsregler**
- Forretningsregler og naturlover har mange likhetstrekk
- Vi ser effekten av dem, men de kan være vanskelige å finne

Beskrivelse (deskripsjon) av virkeligheten/interesseområdet



Informasjonsmodeller

- En fullstendig beskrivelse av interesseområdet kalles en **informasjonsmodell**
- Informasjonsmodellen uttrykkes gjerne i et (eller flere) *modellspråk*
- Noen aktuelle modellspråk er
  - UML (Unified Modelling Languages)
  - ORM (Object-Role Modelling)
  - ER (Entity Relationship)
- *Vår beskrivelse av UoD skal leses av en datamaskin, så den må være meget nøyaktig og detaljert*

Skranker

- Beskrivelsen av forretningsreglene kalles **skranker**
- *Statiske skranker* beskriver begrensninger på mulige tilstander i interesseområdet
- *Dynamiske skranker* beskriver begrensninger på mulige forandringer i interesseområdet
- Den ferdige ORM-modellen er en beskrivelse av de statiske skrankene i vårt UoD

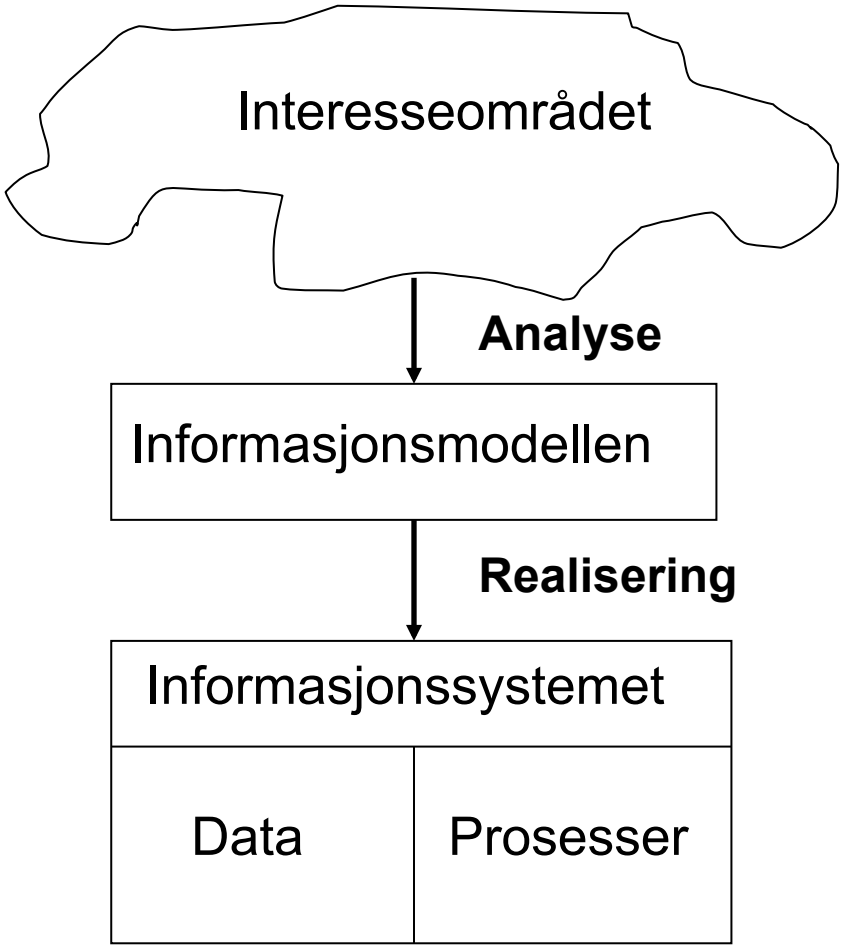
Det begrepsmessige skjema

- Informasjonsmodellen brukt som regelverk (*preskripsjon*) for hvordan informasjonssystemet skal oppføre seg, kalles **det begrepsmessige skjema**
- Det begrepsmessige skjema uttrykkes i et språk som passer for den databaseteknologien vi skal bruke, f.eks.
  - SQL (Structured Query Language) for relasjonsdatabaser
  - ODL (Object Definition Language) for objektdatabaser

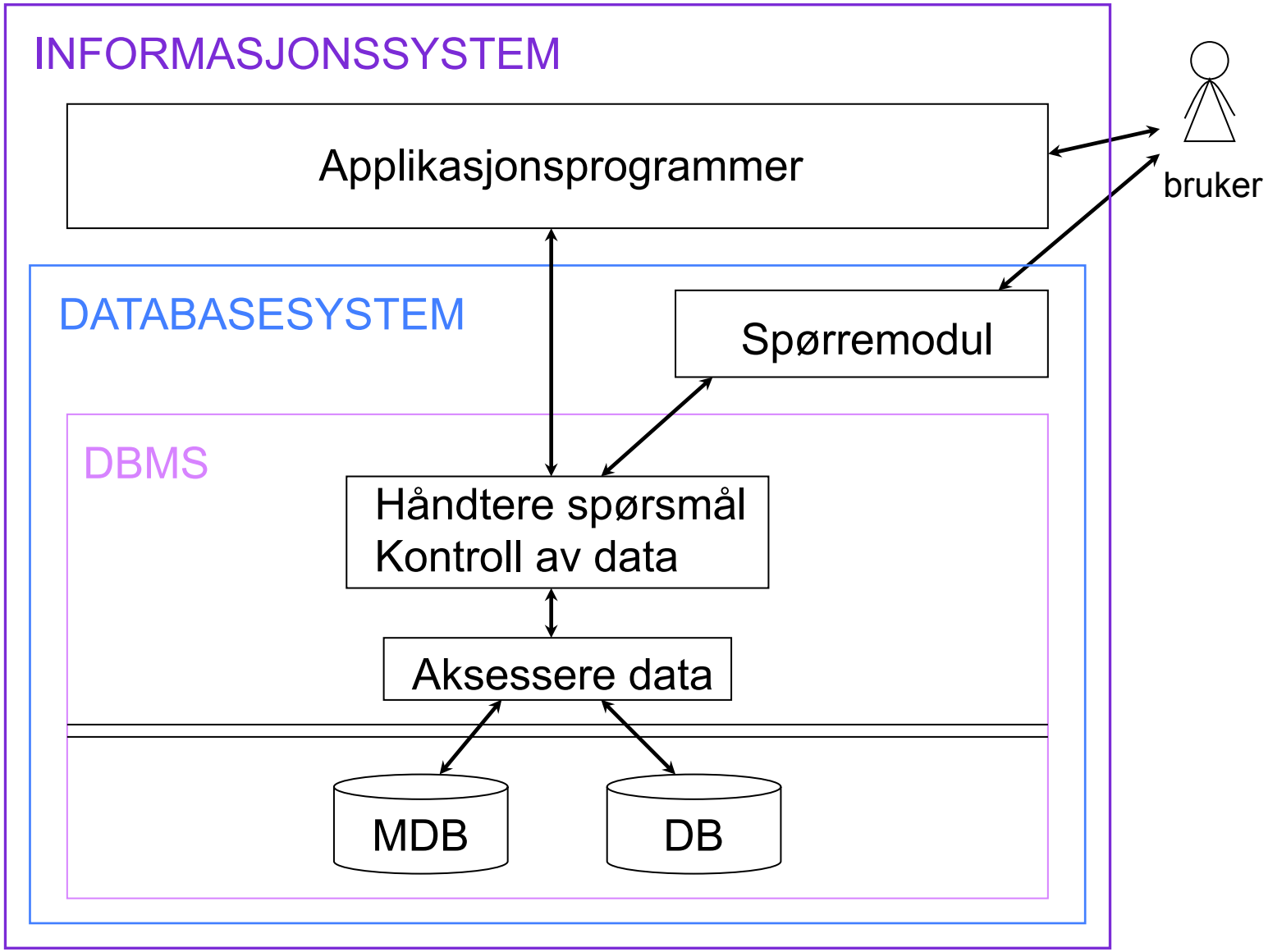
Integritetsregler

- I det begrepsmessige skjemaet kaller vi skrankene for **integritetsregler**
- Integritetsreglene bestemmer
  - hva som er lovlig å *lagre* i informasjons-systemet (lovlige tilstander i databasen)
  - hva som er lovlige *forandringer* (lovlige transisjoner/transaksjoner)
- Det er umulig for en bruker av informasjons-systemet å gjøre noe som strider mot integritetsreglene

Informasjonssystemer



Informasjonssystemer vs. databaser



DBMS – Database Management System

- Spesialisert programvare som understøtter
  - Persistens
  - Transaksjonshåndtering
  - Programmeringsgrensesnitt (API)

ORM – Object Role Modelling

- **Modelleringsspråk**
- **Modelleringsmetode**

  Tre viktige prinsipper:

  1. Ogdens trekant: Sammenhengen mellom virkelighet og modell
  2. Naturlig språk: Modellen må kunne uttrykkes slik at den kan forstås av informerte brukere
  3. 100%-prinsippet: Vi kan lage en nøyaktig nok modell av virkeligheten

Begreper

- Helt fra menneskene fikk språket, har vi satt navn på grupper av tilsvarende ting
- Platon, i sin idé-lære, var den første som beskrev dette systematisk (vitenskapelig)
- Selv om det ikke er noen kyr her inne, vet dere alle hva jeg mener med ordet «ku»
- Og selv de som aldri har sett (et bilde av) en klapperslange, forstår ordet «klapperslange»
- *Ku* og *klapperslange* er to eksempler på begreper
- To mer abstrakte eksempler er *lån* og *flyavgang*

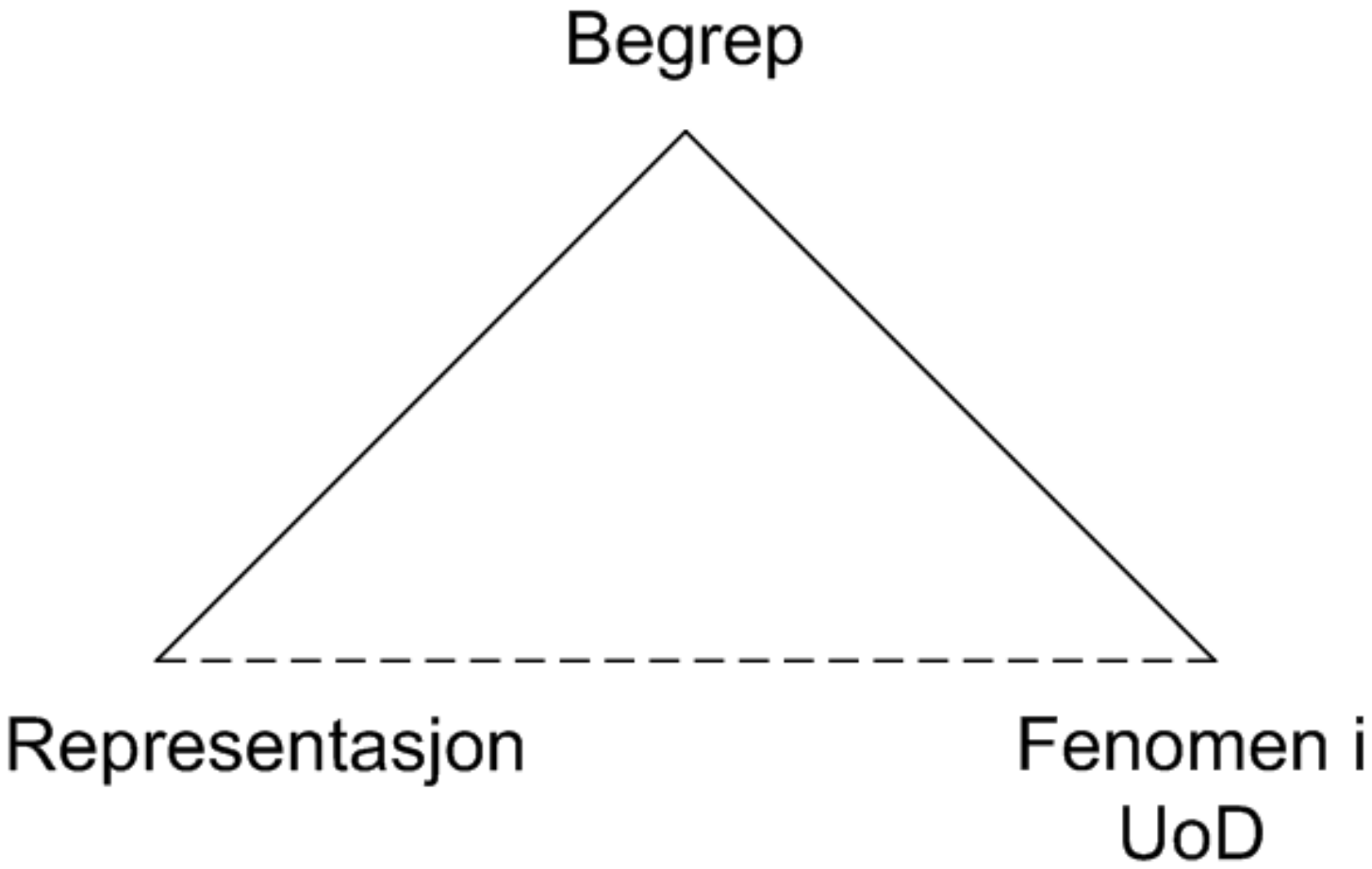
Representasjon

- For å lage et begrepsmessig skjema for UoD må vi velge hvilke begreper skjemaet skal inneholde
- I tillegg må vi for hvert begrep bestemme oss for hvordan vi skal lagre informasjon om forekomster av dette begrepet
- Vi kan ikke lagre kyr i databasen, så vi (eller bonden som skal registrere melkeproduksjon) må finne en måte å lagre informasjon som identifiserer hver enkelt ku
- En slik identifikasjonsmåte kalles en **representasjon** eller **identifikator** for begrepet

Ontologi

- Vitenskapen om sammenhengen mellom virkelighet (UoD), begreper og representasjon kalles **ontologi**
- Ontologi er et meget aktivt forskningsfelt
- Ett eksempel er oljeindustrien som prøver å bli enige om felles begreper og representasjoner for å beskrive alle installasjoner og alt vedlikehold i Nordsjøen
- Hovedpoenget med ontologi betegnes gjerne som Ogdens trekant

Ogdens trekant



Elementære setninger

- En setning som ikke kan deles opp uten å miste meningsinnhold, kalles **elementær**
- Eksempel:
  - Per liker is og brus

  Denne setningen er ikke elementær fordi den kan erstattes av de to elementære setningene
  - Per liker brus
  - Per liker is

Øvelse

Hvilke av disse setningene er elementære?

- Dag er født 27. august 1996
- Hans og Grete er søsken
- DK41050 er en hvit Toyota Yaris
- Anne fikk B i INF1010
- Liv ble ansatt som rådgiver 1.8.2003

Setninger og Ogdens trekant - 1

- Se på setningen «Hanne tar INF1000»
- Det er mye informasjon som er underforstått:
  - Hanne er navn på en student
  - INF1000 er en emnekode
- Det vi mener er at

  studenten med navn Hanne

  tar emnet med emnekode INF1000

Setninger og Ogdens trekant - 2

- La oss se nærmere på setningen:
  «Studenten med navn Hanne tar emnet med emnekode INF1000»
- «student» og «emne» er begreper
- «navn» og «emnekode» er deres representasjoner
- «Hanne» og «INF1000» er forekomster (data)

Setninger og faktatyper

- De to setningene under har samme meningsinnhold:
  - «Studenten med navn Hanne tar emnet med emnekode INF1000»
  - «Emnet med emnekode INF1000 har deltaker studenten med navn Hanne»
- Vi kan forme liknende **fakta** ved å bytte ut forekomstene:
  - «Studenten med navn Henrik tar emnet med emnekode INF1100» (eller: «Emnet med emnekode INF1100 har deltaker studenten med navn Henrik»)
- I ORM tegner vi denne **faktatypen** slik: