

INF1300

Introduksjon til databaser

Dagens tema:

- Relasjonsmodellen
- Funksjonelle avhengigheter og nøkler
- Realisering: Fra ORM til relasjoner

Relasjonsmodellen

- **Datamodell**
Mengde av begreper for å beskrive strukturen til en database
- **Relasjonsmodellen**
Databasen kan betraktes som en samling av tabeller

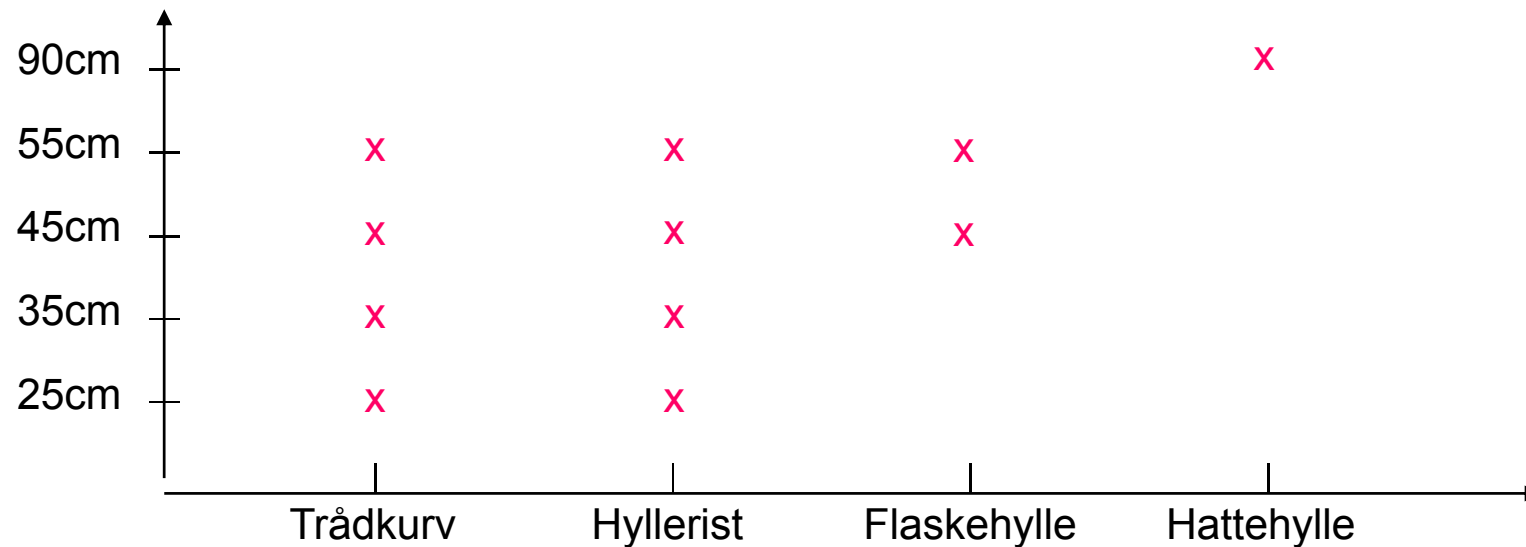
Relasjoner og relasjonsdatabaser

Personale

Ans#	Navn	Fdato	Pers#	Avd
10	Gro	290264	39201	nil
9	Berit	131172	35697	Knøttene
8	Bjørn	150571	34322	Knøttene
12	Liv	031079	39201	nil

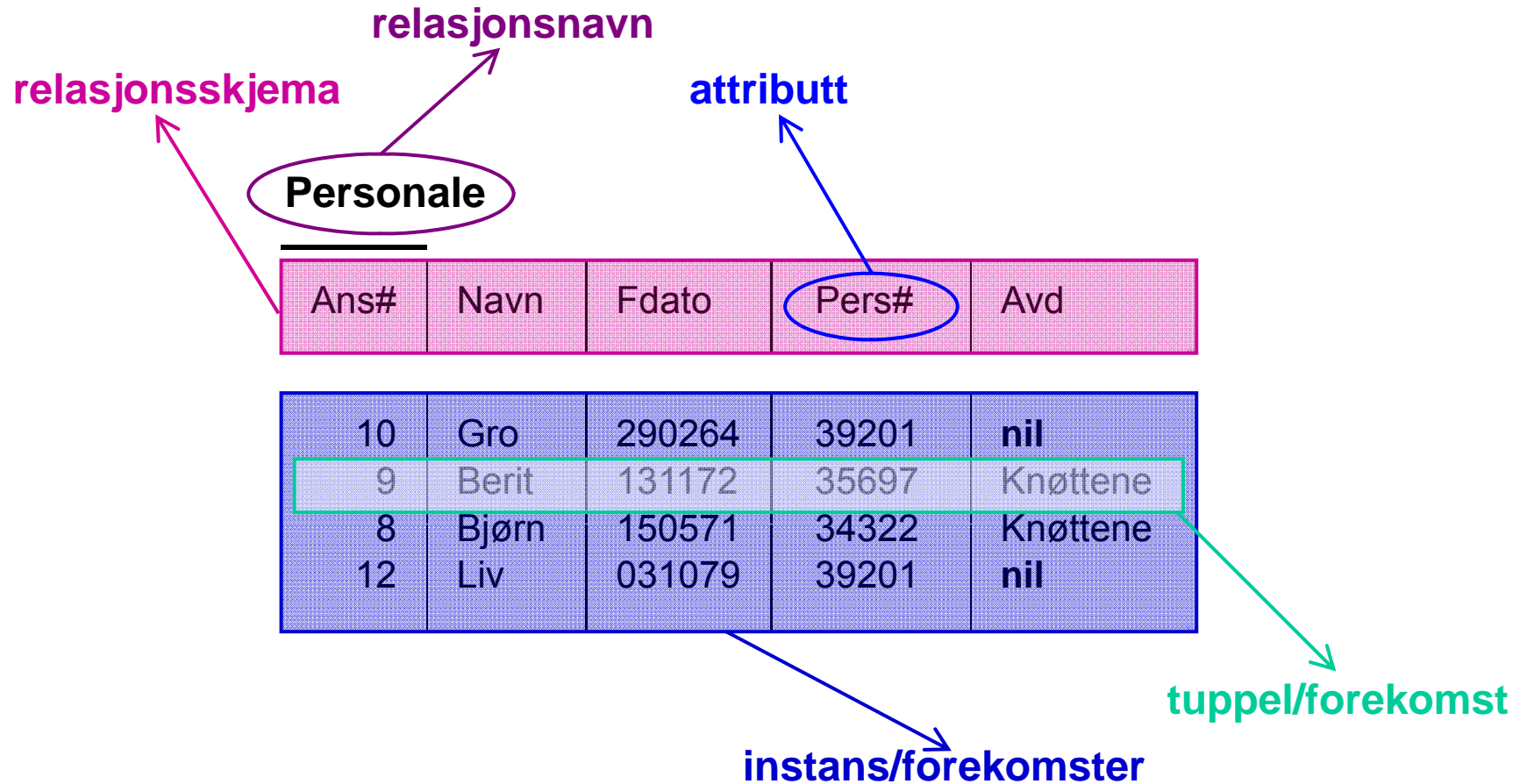
- **Relasjon:** Et matematisk begrep som kan tolkes som en tabell med verdier
- **Relasjonsdatabase:** En samling relasjoner
- **nil** indikerer at ingen verdi ligger lagret i denne posisjonen

Matematisk relasjon



- En **matematisk relasjon** er en mengde av **ordnede** tupper.
I diagrammet over utgjør **x**-ene en matematisk relasjon. I dette tilfellet består den av 2-tupper og kalles en **binærrelasjon**. Den kan også skrives f.eks. slik:
{<trådkurv, 25cm>, <trådkurv, 35cm>, ..., <Hattehylle, 90cm>}

Relasjoner – terminologi



Relasjoner – terminologi

dom(Fdato) =
{sekssifrede tall
med begrensninger
på hvilke tall
som er lovlige}

dom(Avd) =
{Knøttene,
Rosa pantern,
Tommeliten,
Trollungene}

Personale

Ans#	Navn	Fdato	Pers#	Avd
10	Gro	290264	39201	nil
9	Berit	131172	35697	Knøttene
8	Bjørn	150571	34322	Knøttene
12	Liv	031079	39201	nil

Formelle definisjoner

- **Domene**: En mengde *atomære* verdier.
(At elementene i et domene er atomære, betyr at elementene ikke selv kan være mengder.)
- **Attributt**: Et navn på en rolle spilt av et domene («*kolonnenavn*»)
Hvis A er et attributt, skriver vi $\text{dom}(A) = D$ for å uttrykke at A er en rolle spilt av domenet D .
- **Relasjonsskjema** $R(A_1, A_2, \dots, A_n)$: En navngitt mengde attributter $R = \{A_1, A_2, \dots, A_n\}$ der R er **relasjonsnavnet**. n kalles relasjonens *grad* eller *aritet*.

Formelle definisjoner

- **Instans** av et relasjonsskjema $R(A_1, A_2, \dots, A_n)$:
En mengde $\{t_1, t_2, \dots, t_m\}$ («*rader*») der hver t_k er et n -tupple av verdier fra domenenene til A_1, A_2, \dots, A_n .
(Noen av verdiene kan være **nil**, f.eks. fordi verdien for et attributt ikke er lagt inn ennå, fordi verdien er ukjent eller fordi den ikke er relevant.)
- Dersom t er et tupple i en instans av $R(A_1, A_2, \dots, A_n)$ og $t = \langle v_1, v_2, \dots, v_n \rangle$, så er f.eks. $t[A_2] = \langle v_2 \rangle$ og $t[A_3, A_1, A_5] = \langle v_3, v_1, v_5 \rangle$.

Formelle definisjoner

- **Relasjon**: Et relasjonsskjema med en tilhørende instans.

Relasjonsskjemaet kalles relasjonens **intensjon**.

Instansen kalles relasjonens **ekstensjon**.

Merk:

- Tuplens rekkefølge i en instans er vilkårlig
- Verdienes rekkefølge i et tuppel er i utgangspunktet ikke vilkårlig (dette er mest for at notasjonen skal bli enklere)
- I en instans kan det ikke finnes to like tupler
- Et domene kan være endelig eller uendelig
- To attributter i et relasjonsskjema kan ha samme domene, men ikke samme navn

Nøkler og nøkkelattributter

Personale

Ans#	Navn	Fdato	Pers#	Avd
10	Gro	290264	39201	nil
9	Berit	131172	35697	Knøttene
8	Bjørn	150571	34322	Knøttene
12	Liv	031079	39201	nil

- Vi ønsker ikke at to ansatte skal kunne ha samme Ans#
- To personer kan aldri ha samme fødselsnummer = Fdato + Pers#

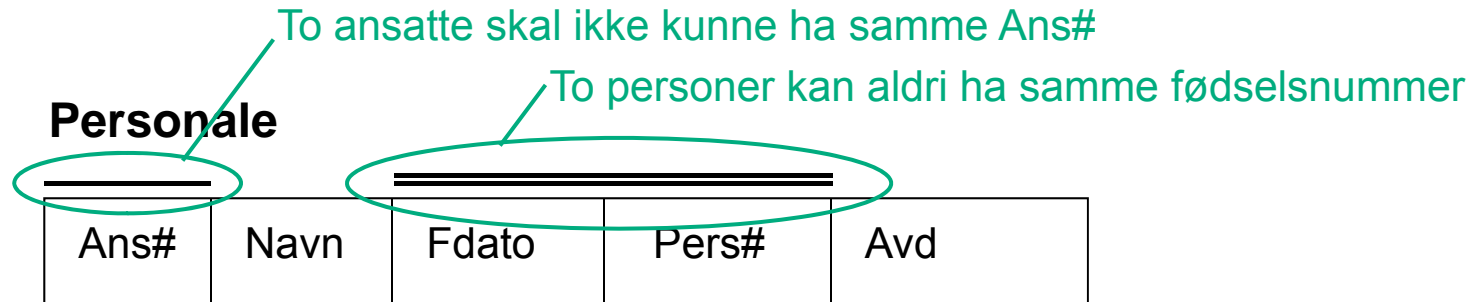
Nøkler og nøkkelattributter

- **Supernøkkel:** En kombinasjon (delmengde) X av attributtene $\{A_1, A_2, \dots, A_n\}$ som er slik at hvis t og u er to tupler hvor $t \neq u$, så er $t[X] \neq u[X]$.
Merk: Relasjonsskjemaet er alltid selv en supernøkkel
- **Kandidatnøkkel:** En minimal supernøkkel
Dvs: Fjerning av et hvilket som helst attributt fører til at de gjenværende attributtene ikke lenger utgjør en supernøkkel.
- **Supernøkler benyttes til å uttrykke integritetsregler**

Nøkler og nøkkelattributter

- **Primærnøkkel:** En utvalgt blant kandidatnøklerne. Alle relasjoner skal ha nøyaktig én primærnøkkel.
- **Nøkkelattributt:** Attributt som er med i (minst) en kandidatnøkkel.

Nøkler og nøkkelattributter



- Primærnøkkelen blir gjerne markert med én strek
- Andre kandidatnøkler er i dette tilfellet markert med to streker
- Merk likheten mellom kandidatnøkler og entydighetsskranker i ORM: Begge angir at forekomster under skranken bare kan forekomme én gang

Funksjonelle avhengigheter

Personale

Ans#	Navn	Fdato	Pers#	Avd
------	------	-------	-------	-----

- Det at en person har høyst ett Ans#, gjør at hvis vi vet hvilken person det er snakk om (dvs. vi kjenner personens Ans#), så vet vi også navnet, fødselsnummeret og avdelingen til personen.
- Primærnøkkelen definerer altså en funksjon fra forekomstene av Ans# til forekomstene av Navn, Fdato, Pers# og Avd.
 - Det samme gjelder andre kandidatnøkler: Hvis vi kjenner forekomstene for attributtene Fdato og Pers#, så har vi bare én mulig verdi for hver av Ans#, Navn og Avd.
- Vi sier at Navn, Fdato, Pers#, Avd er **funksjonelt avhengig** av Ans#, eller at vi har en **FD** (Functional Dependency) fra Ans# til Navn, Fdato, Pers#, Avd.
- Den vanlige notasjonen for en FD er: **Ans# → Navn, Fdato, Pers#, Avd**

Fremmednøkler

Barn

Løpe#	Navn	Fdato	Avd	TilknPers
2	Lisa	180502	Rosa Pantern	nil
5	Trym	030205	Knøttene	9
4	Anne	301102	Tommeliten	nil
7	Anne	151204	Knøttene	8

- Vi vil at TilknPers skal referere til forekomster i **Personale-**tabellen

Fremmednøkler

- **Fremmednøkkel:** Ett eller flere attributter som peker ut/refererer et tuppel i en annen relasjon.

Personale

Ans#	Navn	Fdato	Pers#	Avd
10	Gro	290264	39201	nil
9	Berit	131172	35697	Knøttene
8	Bjørn	150571	34322	Knøttene
12	Liv	031079	39201	nil

Barn

Løpe#	Navn	Fdato	Avd	TilknPers
2	Lisa	180502	Rosa Pantern	nil
5	Trym	030205	Knøttene	9
4	Anne	301102	Tommeliten	nil
7	Anne	151204	Knøttene	8

Fremmednøkler

- Fremmednøkkelen må ha samme antall attributter som primærnøkkelen i den relasjonen den peker ut, og attributtene må ha parvis samme domener.
Noen databasesystemer tillater også fremmednøkler til kandidatnøkler som ikke er primærnøkler.
- Korresponderende attributter behøver ikke å ha samme navn.
- Det er lov å ha fremmednøkler til «seg selv»
- Fremmednøkler benyttes til å uttrykke integritetsregler

Påkrevde integritetsregler i relasjonsdatabaser

- **Entitetsintegritet:**
Alle relasjonsskjemaer skal ha en og bare en primærnøkkel.
Ingen av attributtene i primærnøkkelen får være **nil**.
- **Referanseintegritet:**
Hvis fremmednøkkelen ikke er **nil**, så skal det finnes et tuppel i den refererte relasjonen hvor primærnøkkelen har samme verdi som fremmednøkkelen (dvs. at det refererte tuppelet skal eksistere).

I tillegg kan databasen ha andre integritetsregler, for eksempel kandidatnøkler utover primærnøkklene.

Relasjonsdatabaser - definisjoner

- **Relasjonsdatabaseskjema**: Samling av relasjonsskjemaer + integritetsregler
- **Relasjonsdatabaseinstans**: Samling av relasjonsinstanser
- **Relasjonsdatabase** =
Relasjonsdatabaseskjema +
relasjonsdatabaseinstans

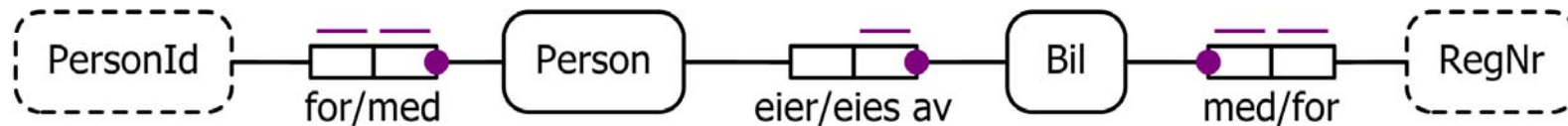
Merk:

- I forskjellige relasjonsskjemaer kan attributtnavn gjenbrukes
- Notasjon for å skille mellom attributter med like navn: $R.A_i$

Realiseringsalgoritmen

Fra ORM-diagram
til relasjonsdatabaseskjema

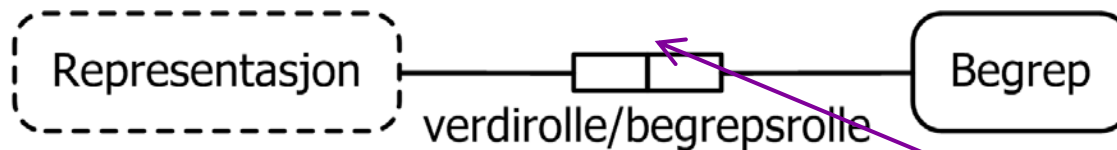
Underliggende idé (forenklet)



- For hvert begrep: lag en tabell
 - For hver faktatype: lag en tabell
 - Perfekte broer brukes til å bestemme hvordan begrepene skal representeres
 - Entydighetspiler brukes til å bestemme primærnøkler
 - For å få en “penere” database: slå sammen tabeller med samme primærnøkkel
- Person(), Bil()
 - eier/eies_av(,)
 - Person(PersonId)
Bil(RegNr)
eier/eies_av(PersonId, RegNr)
 - Person(PersonId)
Bil(RegNr)
eier/eies_av(PersonId, RegNr)
 - Person(PersonId)
Bil(RegNr, PersonId)

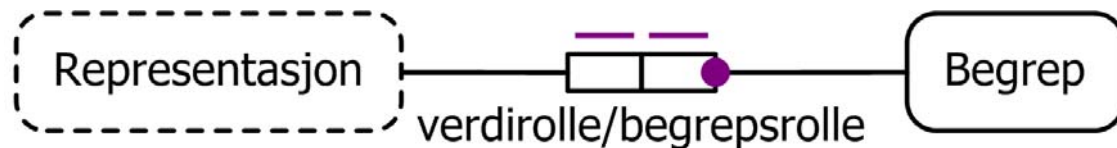
Setningstyper (repetisjon)

- Bro

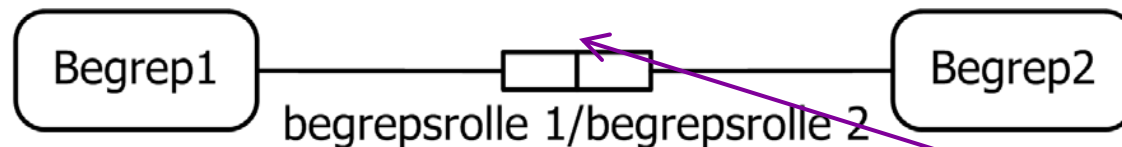


Minst én entydighetskranke

- Perfekt bro



- Faktatype (vi viser bare en **binær** faktatype her)

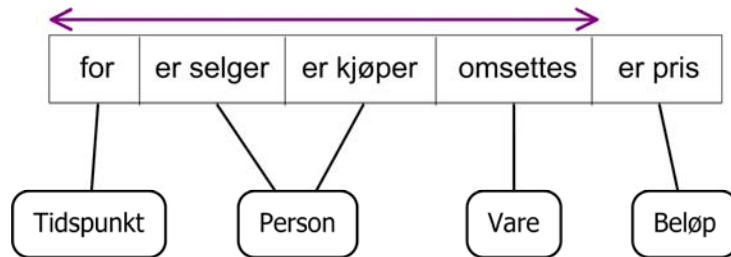


Minst én entydighetskranke

Forutsetninger/forberedelser

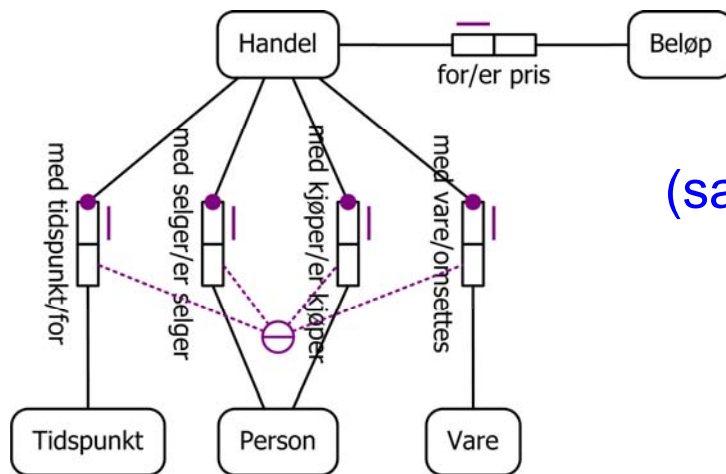
- A. Alle **lange piler** må gjøres til gjenstand for begrepsdannelse (og gis et navn)
- B. ORM-diagrammet må være **refererbart**
- C. Diagrammet må ikke inneholde **synonyme broer**: alle broer må ha en entydig begrepsrolle

A. Begrepsdannelse av «lange» piler

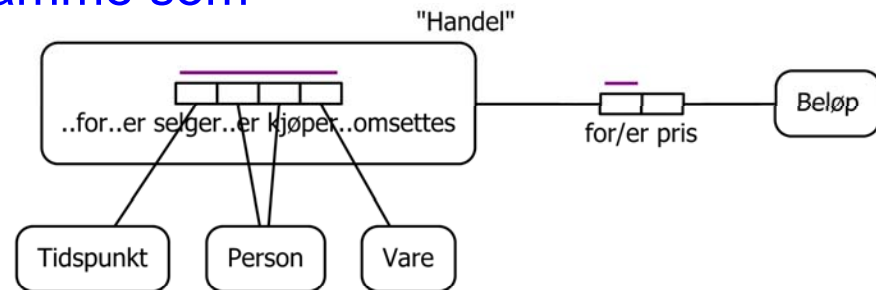


En lang pil er en ekstern entydighet i forkledning

erstattes av



(samme som

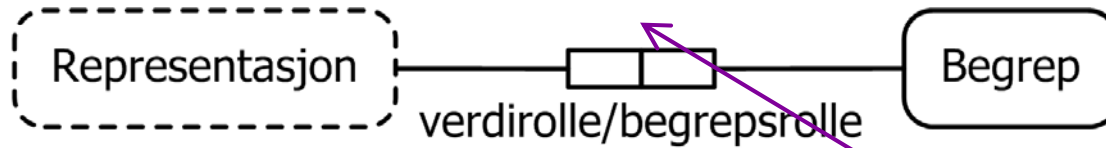


B. Refererbare ORM-diagrammer

- Intuitivt er et ORM-diagram refererbart hvis alle begreper kan representeres entydig (via perfekte broer)
- Vi kommer tilbake til hva som skal til for å gjøre et ORM-diagram refererbart under realiseringsalgoritmen

C. Eliminasjon av synonyme broer

Bro:



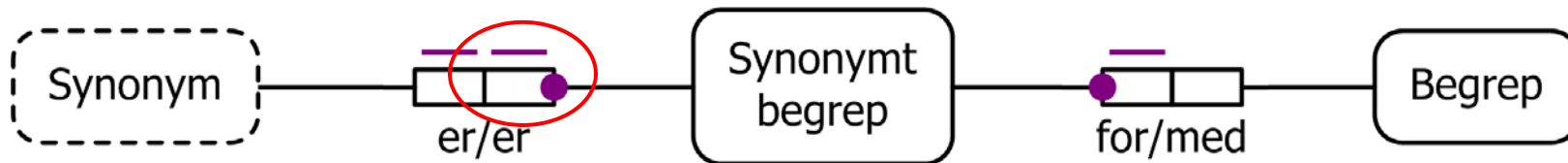
Etter begrepsdannelsene:
alltid én eller to **korte**
entydighetsskranke

Synonym
bro:



entydighetsskranke
mangler

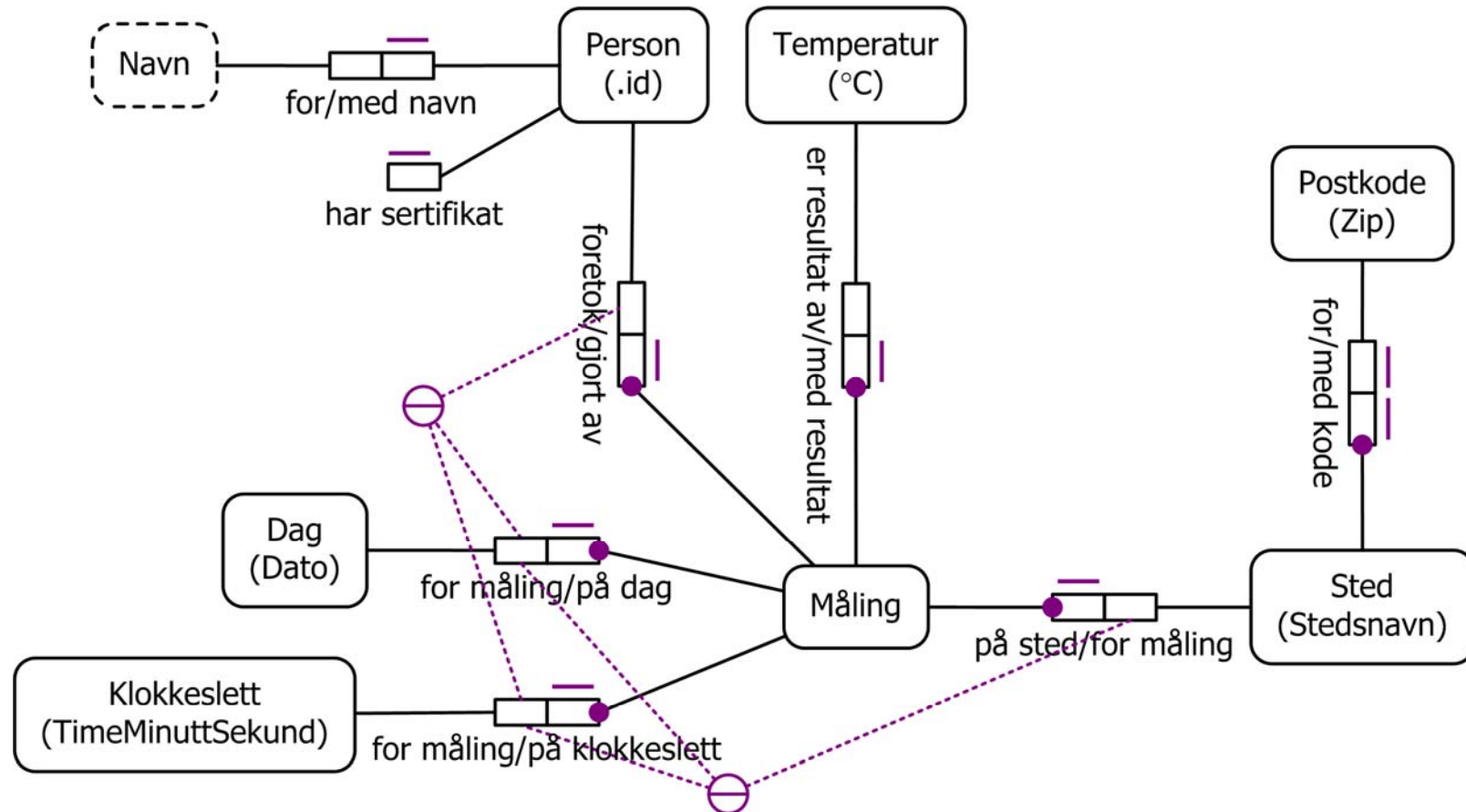
erstattes av



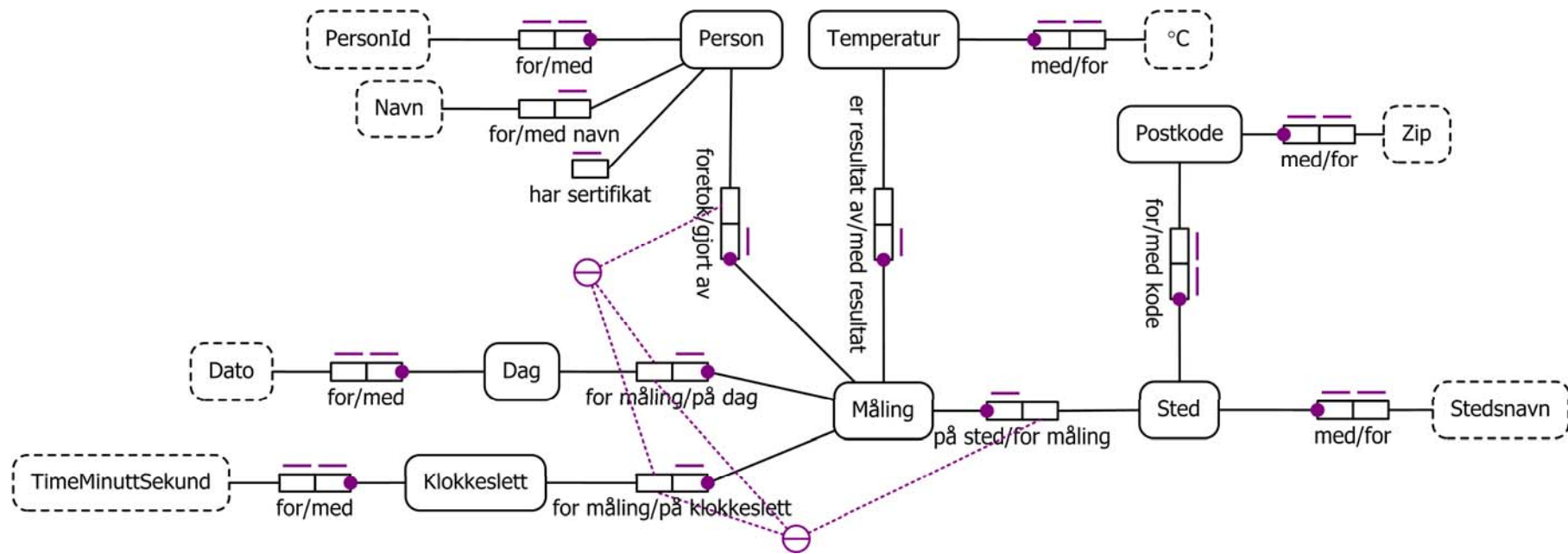
Realiseringsalgoritmen

- 1. Hvert begrep gir opphav til en relasjon (basistabell) med samme navn som begrepet**
- 2. Finn referansemåte for alle begreper og marker alle prefererte referansetyper som brukt (Referansemåtene blir primærnøkler)**
- 3. Grupper resterende broer til sine respektive begreper (Hver bro gir ett attributt i tabellen)**
- 4. Grupper resterende faktatyper (Hver faktatype blir en fremmednøkkel)**
- 5. Overfør skrankene til relasjonsskjemaet**
- 6. Bestem hvilke referanserelasjoner som skal fjernes**

Eksempel

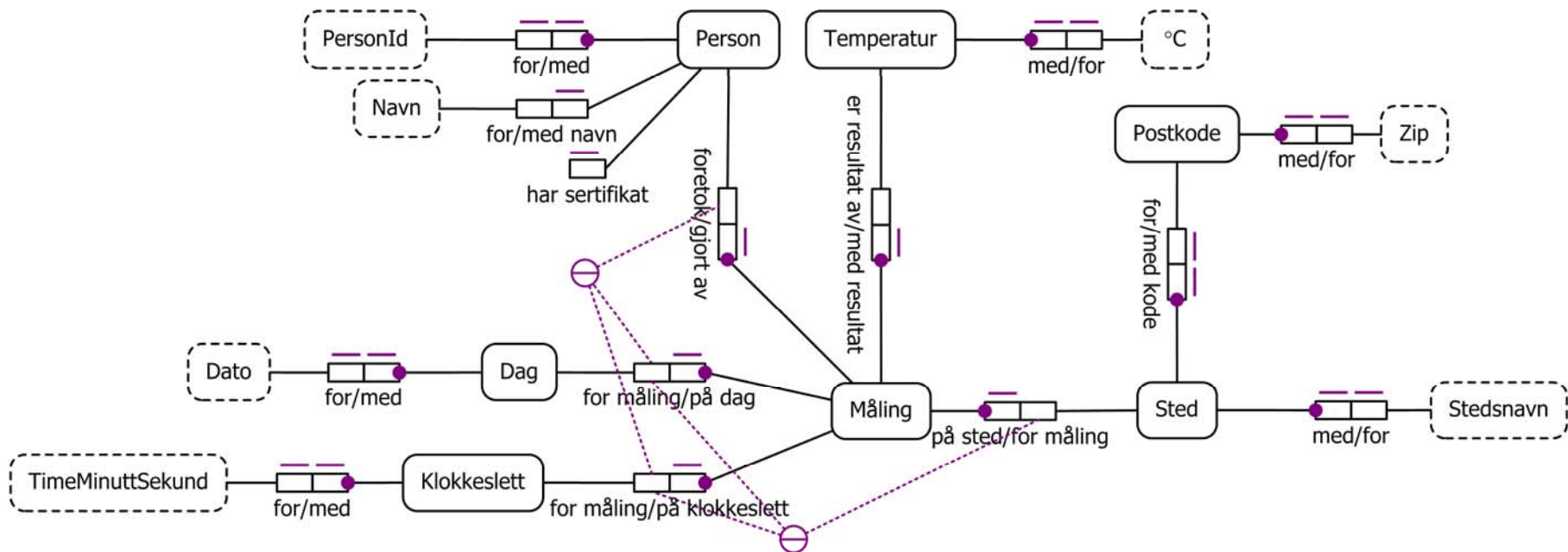


Eksempel – med eksplisitte perfekte broer



Steg 1: Fra begrep til relasjon

1. Hvert begrep gir opphav til en relasjon (basistabell) med samme navn som begrepet



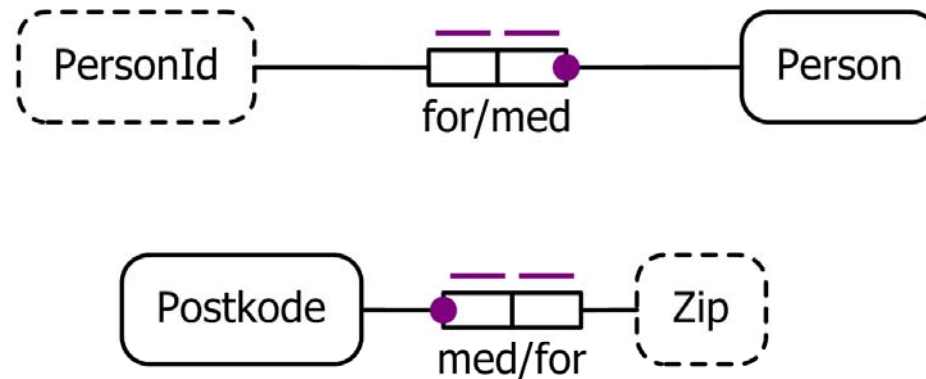
Relasjoner: **Person, Dag, Klokkeslett, Postkode, Sted, Måling, Temperatur**

Steg 2: Valg av referansemåter

2. Finn referansemåte for alle begreper og marker alle prefererte referansetyper som brukt (Referansemåtene blir primærnøkler)

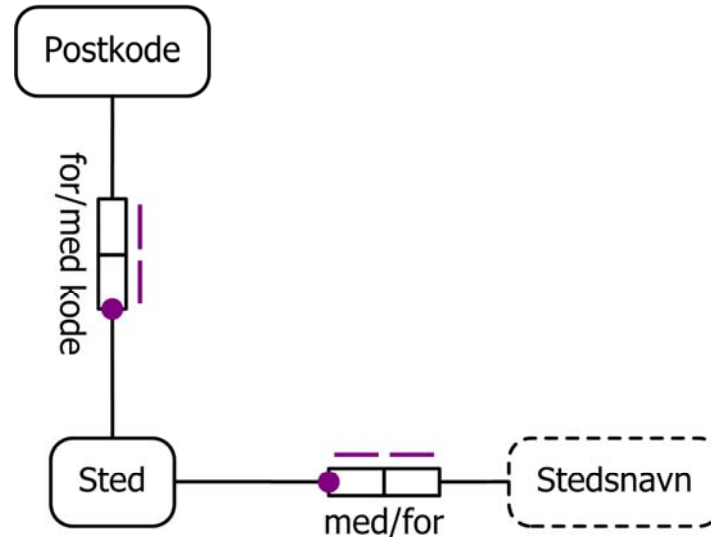
- **Referansemåten** til et begrep er
 - enten: navnet på en representasjon knyttet til begrepet med en perfekt bro
 - eller: referansemåten til en 1:1-faktatype hvor begrepet har en total rolle
 - eller: for begrepsdannelser, samlingen av referansemåtene til de begrepene som utgjør grunnlaget for begrepsdannelsen
 - (detaljer følger)
- Hvis et begrep mangler referansemåte, er ORM-diagrammet ikke refererbart og kan ikke realiseres

Referanse måte via perfekt bro



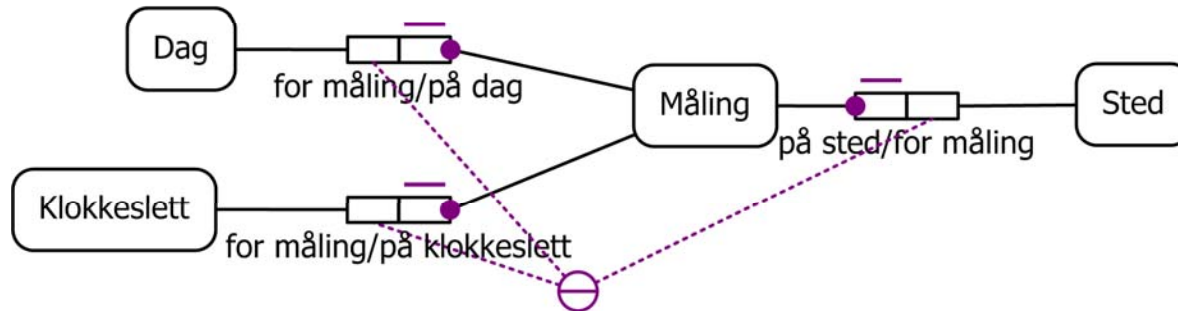
- Referanse måten til **Person** er **PersonId**
- Referanse måten til **Postkode** er **Zip**

Referanse måte via 1:1-faktatype



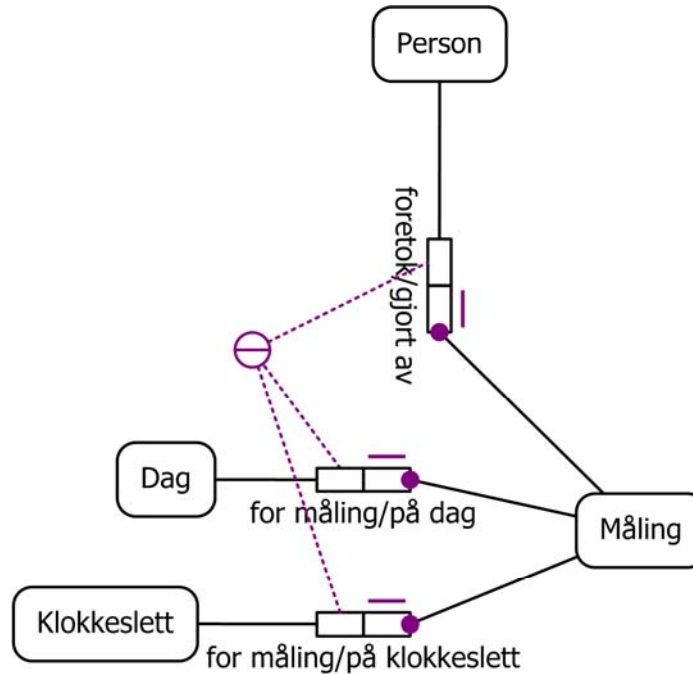
- For begrepet [Sted](#) har vi to mulige referanse måter:
Via en perfekt bro til [Stedsnavn](#)
Via en 1:1-faktatype med total rolle til [Postkode](#)
- Vi velger f.eks. den perfekte broen; referanse måten til [Sted](#) blir da [Stedsnavn](#)
- Alternativt kan vi velge 1:1-faktatypen; da arver [Sted](#) referanse måte fra [Postkode](#) og får referanse måten [Zip](#)

Referansemåte via begrepsdannelse



- Begrepet **Måling** er en begrepsdannelse basert på begrepene **Dag**, **Klokkeslett** og **Sted**
 - **Måling** har entydige totale roller **på dag**, **på klokkeslett**, **på sted**
 - Det er en ekstern entydighetsskranke over rollene til **Dag**, **Klokkeslett** og **Sted**
- Referansemåten til **Dag** er **Dato**
Referansemåten til **Klokkeslett** er **TimeMinuttSekund**
Referansemåten til **Sted** er valgt til **Stedsnavn**
En referansemåte til **Måling** er derfor
(**Dato**, **TimeMinuttSekund**, **Stedsnavn**)

Referansemåte via begrepsdannelse



- Alternativt kunne vi valgt referansemåte til **Måling** basert på begrepene **Dag**, **Klokkeslett** og **Person**; syntaktisk er det ikke noe i veien for dette, men begrepsmessig er det ikke like naturlig

Referanseemåtene blir primærnøkler

Referanseemåtene blir **primærnøkler** i de tilhørende relasjonene:

Person(PersonId)

Dag(Dato)

Klokkeslett(TimeMinuttSekund)

Postkode(Zip)

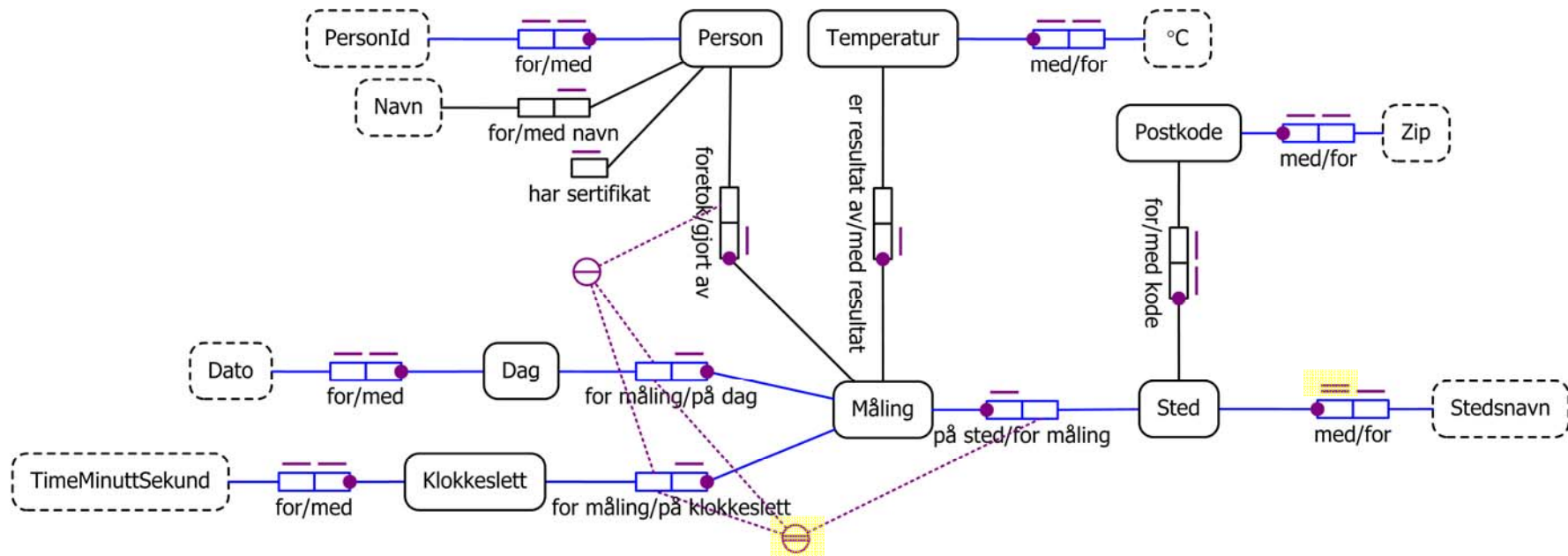
Sted(Stedsnavn)

Måling(Dato, TimeMinuttSekund, Stedsnavn)

Temperatur(°C)

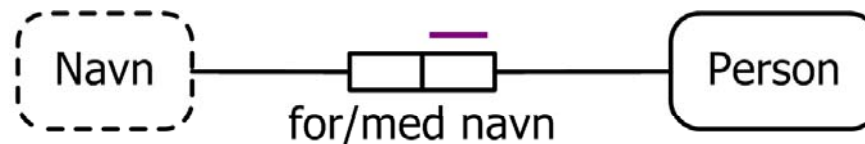
Status etter steg 2

- Alle gjenværende faktatyper er unære eller binære og har minst én kort entydighetsskranke
- Alle gjenværende broer har kort entydighetsskranke på begrepsrollen
- I ORM kan man, hvis det er flere alternativer, angi valg av referansemåte med dobbel entydighetsskranke. Under er i tillegg de brukte setningstypene markert med blått (de gjenværende er fortsatt svarte):



Steg 3: Gruppering av broer

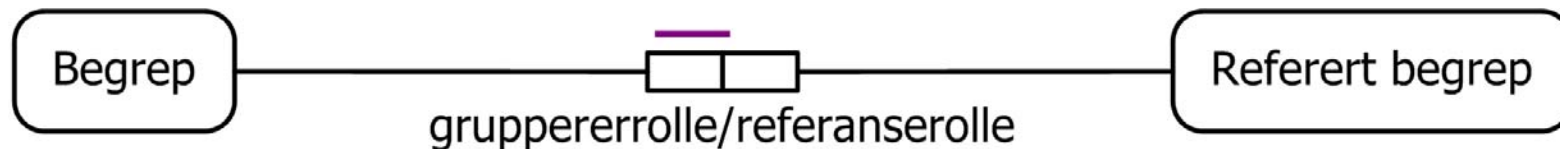
3. Grupper resterende broer til sine respektive begreper
(Hver bro gir ett attributt i tabellen)



- Relasjonen **Person** får attributtet **Navn_for**:
Person(PersonId, Navn_for)
- Hvis begrepsrollen hadde vært total, ville nullverdier ikke vært tillatt i **Navn_for**

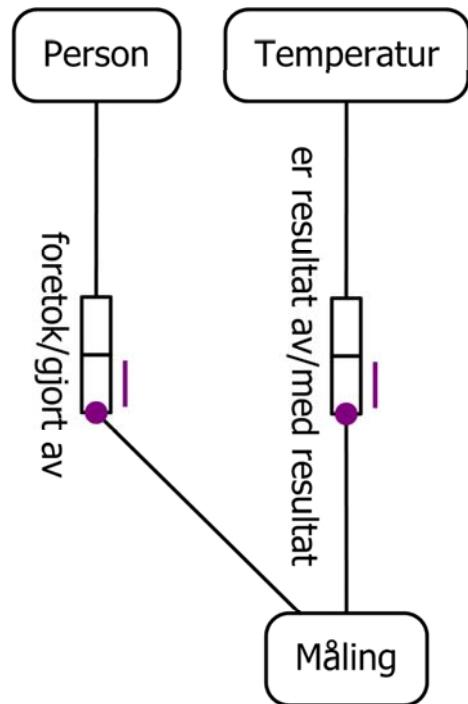
Steg 4: Gruppering av faktatyper

4. Grupper resterende faktatyper (Hver faktatype blir en fremmednøkkel)



- I binære faktatyper velges en **entydig** rolle som **gruppererrolle**; den andre kalles **referanserollen**
 - Hvis begge rollene er entydige og en av dem er total, velges den totale rollen som gruppererrolle
- Relasjonen til gruppererrollens begrep får en fremmednøkkel til det refererte begrepets relasjon

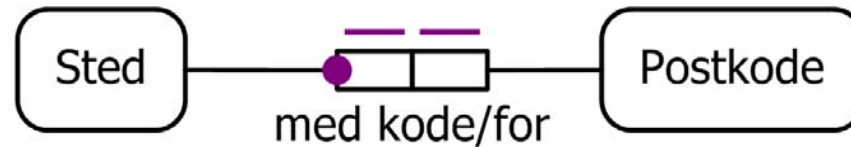
Gruppering av binære faktatyper



- Relasjonen **Måling** får et attributt **Person_foretok**
- Attributtet **Person_foretok** blir fremmednøkkel til relasjonen **Person**
- Tilsvarende får **Måling** attributtet **Temperatur_er_resultat_av** som er fremmednøkkel til relasjonen **Temperatur**

Måling(Dato, TimeMinuttSekund, Zip, Temperatur_er_resultat_av, Person_foretok)

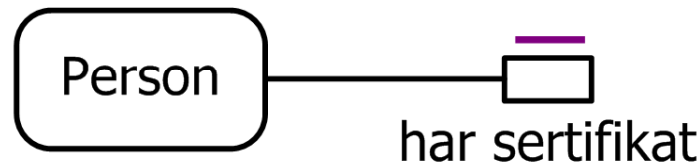
Gruppering av 1:1-faktatyper



- Med to entydige roller kan begge velges som gruppererrolle
- Hvis én av rollene er total (som tilfellet er for **med kode**), bør denne velges
- På grunn av entydighetsskranken over referanserollen er fremmednøkkelen **Postkode_for** entydig. I dette tilfellet blir **Postkode_for** derfor en kandidatnøkkel for **Sted**

Sted(Stedsnavn, Postkode_for)

Gruppering av unære faktatyper



- Relasjonen **Person** får et Boolesk attributt **har_sertifikat**:
Person(personId, Navn_for, har_sertifikat)
- Nullverdier er aldri tillatt for Booleske attributter som stammer fra unære faktatyper
- Rollen i en unær faktatype kan aldri være total
- Vi kan klare oss uten unære faktatyper:



Status etter steg 4

Etter steg 4 ser relasjonsdatabaseskjemaet slik ut (i tillegg kommer fremmednøkler):

Person(PersonId, Navn_for, har_sertifikat)

Dag(Dato)

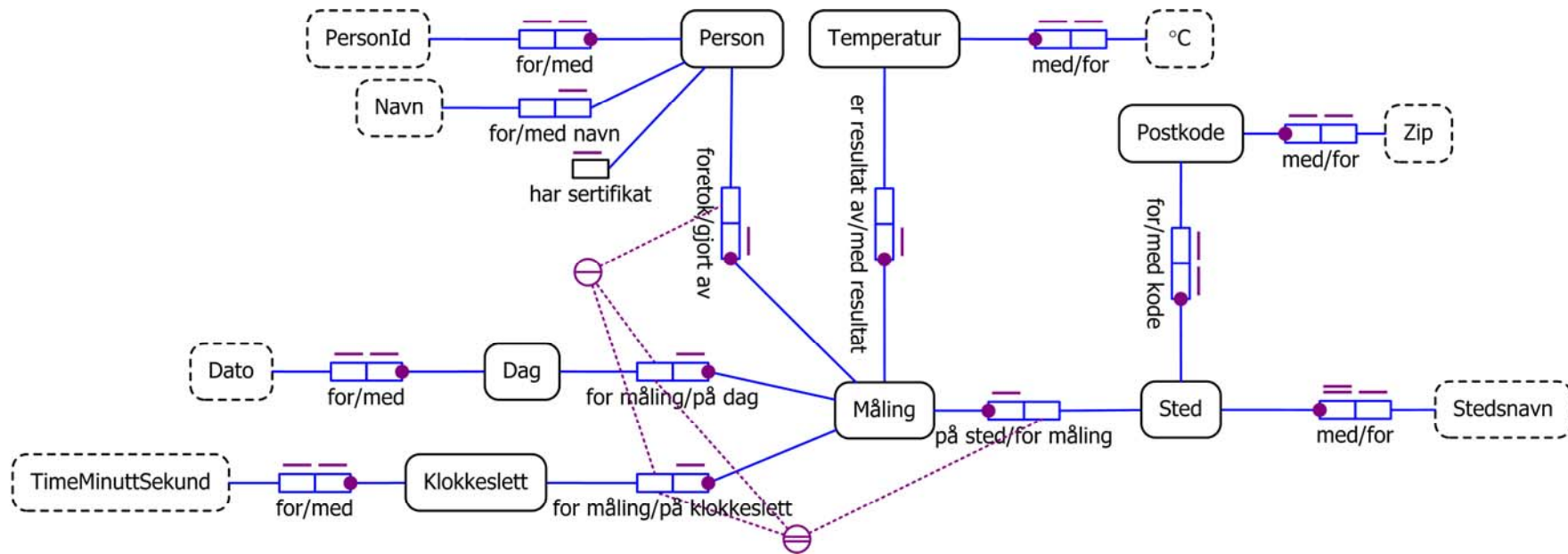
Klokkeslett(TimeMinuttSekund)

Postkode(Zip)

Sted(Stedsnavn, Postkode_for)

Måling(Dato, TimeMinuttSekund, Stedsnavn, Temperatur_er_resultat_av, Person_foretok)

Temperatur(°C)



Fremmednøkler

