INF115 Lecture 7: *Data Modelling with ER*

Adriaan Ludl
Department of Informatics
University of Bergen

Spring Semester 2021

Deep Concentration & Beneficial Habits

"**… (are) essential and necessary for learning when studying**,"

says Åsa Hammar,

Professor at the Department of Biological and Medical Psychology.

-- https://www.vaergodmotdegselv.no/en/log-off

Practice makes perfect
Research suggests that deep concentration works like a muscle – *it can be trained gradually over time.*

Deep concentration takes place when you can focus on your work to the extent that you get into a flow state and *feel as if everything else around you almost disappears.*

Structure
Try to make a good structure for your day. Be clear about when you are going to work and when you are going to take breaks.

Chapter 7: *Data modelling with ER*

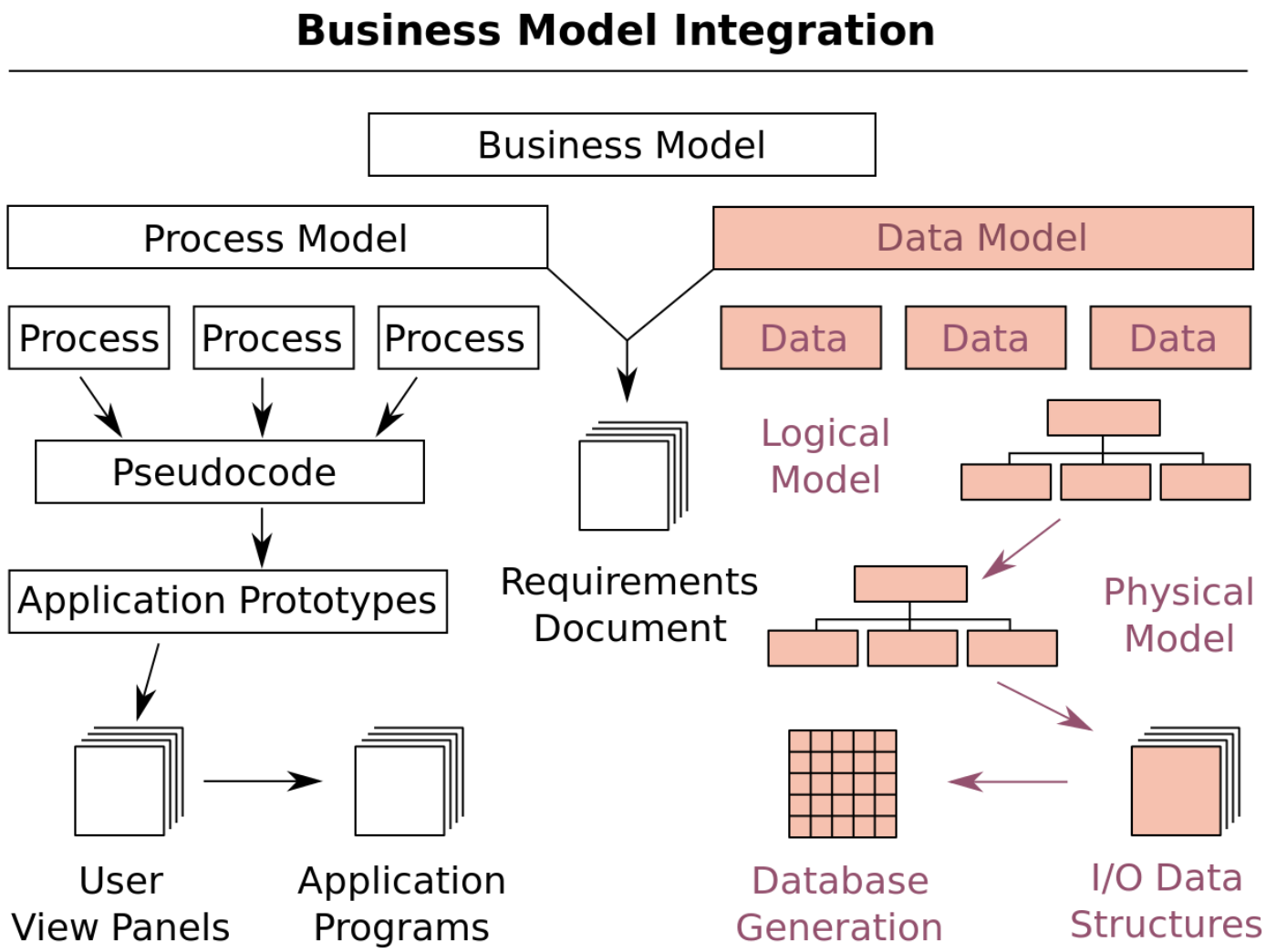
Learning Goals:

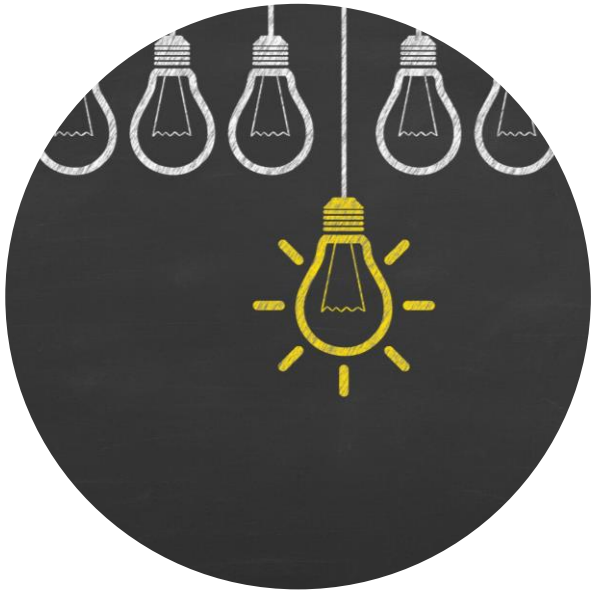
- **Understand** *goals of data modelling and database design*
- Use ER (Entity Relationship) and UML (Unified Modeling Language) to make data models.
  - Use entities with attributes and identifiers
  - One-to-one, one-to-many and many-to-many relationships
  - Weak entities and identifying relationships
  - Multiple value attributes, composed attributes, subtypes, aggregation, and composition in UML
- Frequently used modeling patterns

Data Model

- Logical Model
- Physical Model
- I/O Data Structures
- Database Generation

Image from
https://en.wikipedia.org/wiki/Data_model





The Data Modelling Process and Goal

- ❖ Which tables should the database contain ?
- ❖ Which columns should the tables have ?

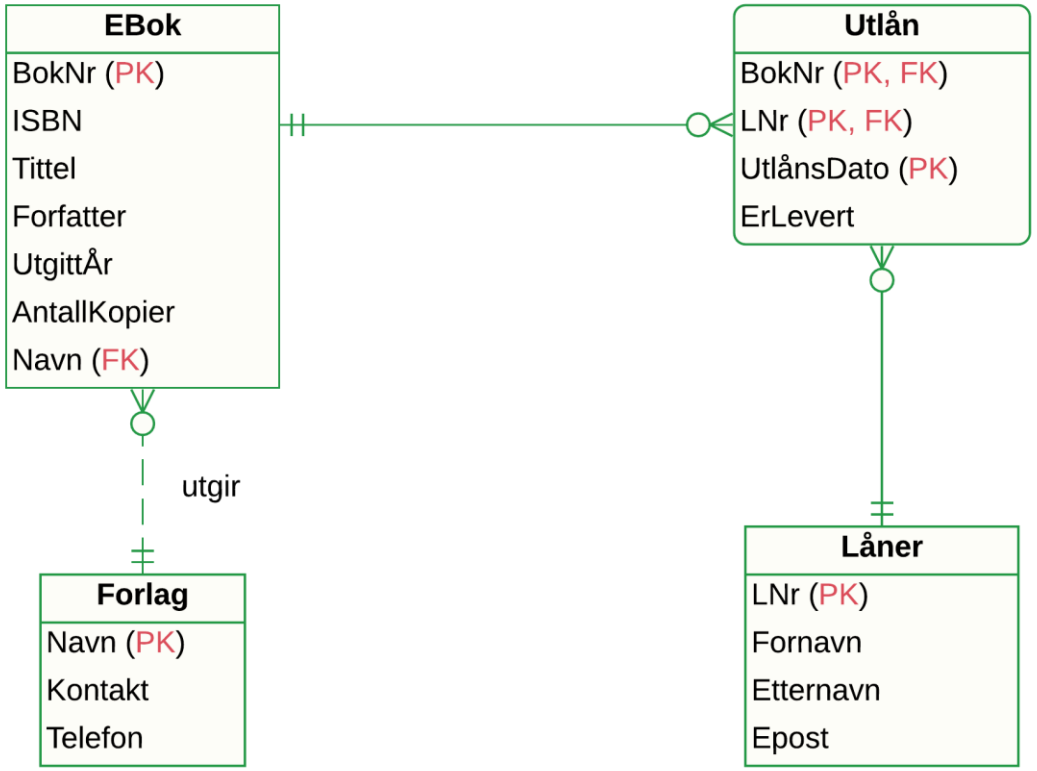
- ➢ We have to try several ideas to find an *appropriate structure*.
- ➢ ***Visual models*** *have proven useful tools in this process.*

- ❖ **ER** (Entity Relationship) is a **visual modelling language**.
  - An **ER-diagram** shows the structure of a database.
  - They are used to design databases.

Introduction to ER-diagrams

ER-diagrams are a visual description of the structure of a database.

- Every **entity** (box) corresponds to a table.
- **Attributes** (lower part of boxes) correspond to columns.
- **Identifiers** (PK) are underlined and represent primary keys.
- ***Relationships*** (lines) correspond to ***foreign keys***.



ER-diagrams are used when we **plan** what the database should contain.

They should:

- Give a **good overview,**
- Be **readable** for non IT-experts.

Requirements analysis and data collection

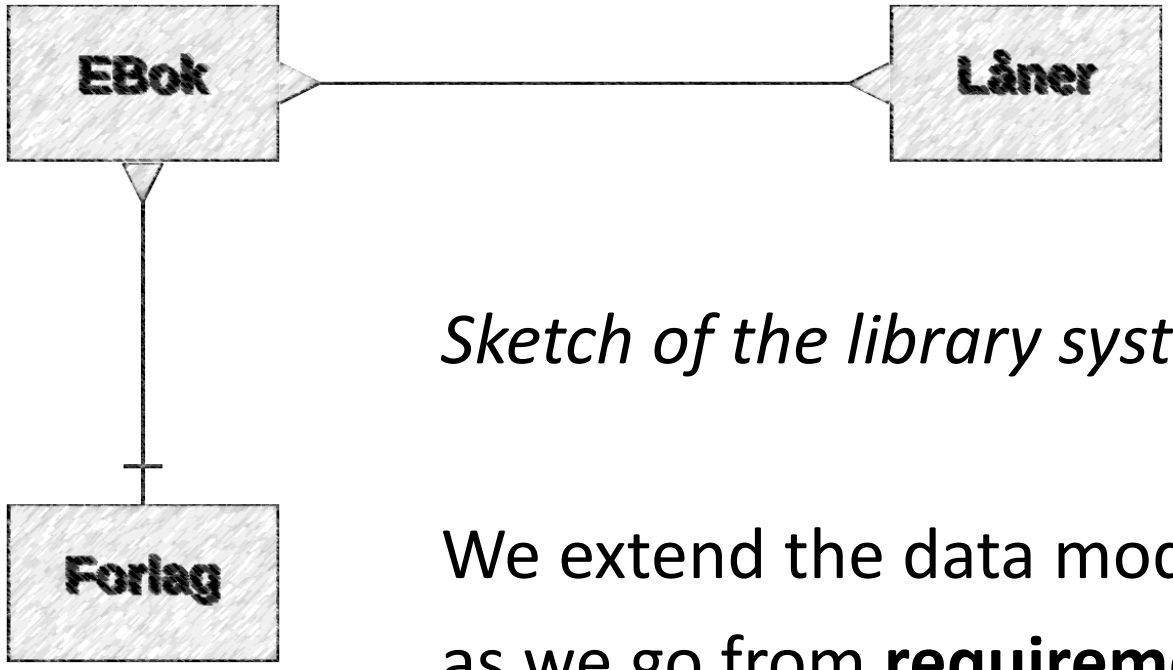
Requirements analysis (Kravanalyse) is used to decide what such a system shall do.

It starts with **collecting information**: study reports, questionnaires, interviews and surveys.

Example: Requirements for a **library system**:

- The system must store information about users (borrowers), books, loans and publishers.
- For every e-book store ISBN, title, authors and year of publication.
- One publisher can publish many e-books (many titles), but one book is provided by one publisher.

From Sketch to Model to Database



Sketch of the library system.

We extend the data model with more details,
as we go from **requirements** to a **sketch** of the system**,**
to gradually more detailed **data models,**
to an actual **database**.

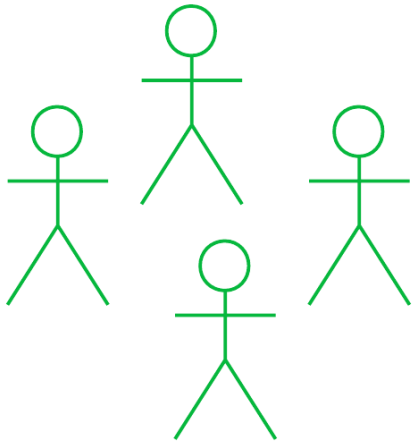
Entities and Attributes

- An **entity** (entitet) is an object that we want to store information about.
- The «interesting» properties of an entity are called **attributes**.
- Attributes should have defined **data types**. We can skip this for now, but it is needed in a *finished database design*.

Student
StudentNr
Fornavn
Etternavn
Adresse
Telefon

Innlegg
Nr INTEGER
Brukernavn VARCHAR(50)
Dato DATE
Melding VARCHAR(140)

Entity Type and Instance



Student

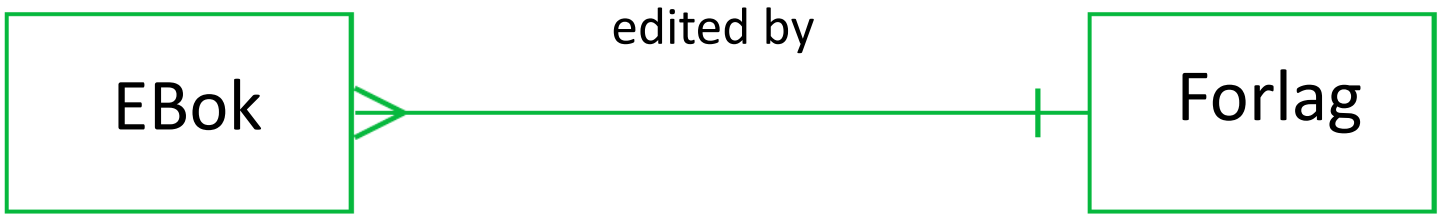
- The table *Student* has many rows.
- Every row describes one person.

- An **entity instance** (forekomst) corresponds to a **row** in a table.
  - Example: (StudentNr=1, Fornavn='Hans', Etternavn='Hansen', Adresse='Hansegata 3')

- **The entity type** represents the **set** of instances – and thus corresponds to the whole table.

StudentNr	Fornavn	Etternavn	Adresse

Relationships



- Forlag (publisher) and Ebok (ebook) are **entities**.
- There is a **relationship** (forhold) between Forlag and EBok:
  A book is edited by a publisher.

Relationship Cardinality

1:N

(one-to-many)

1:1

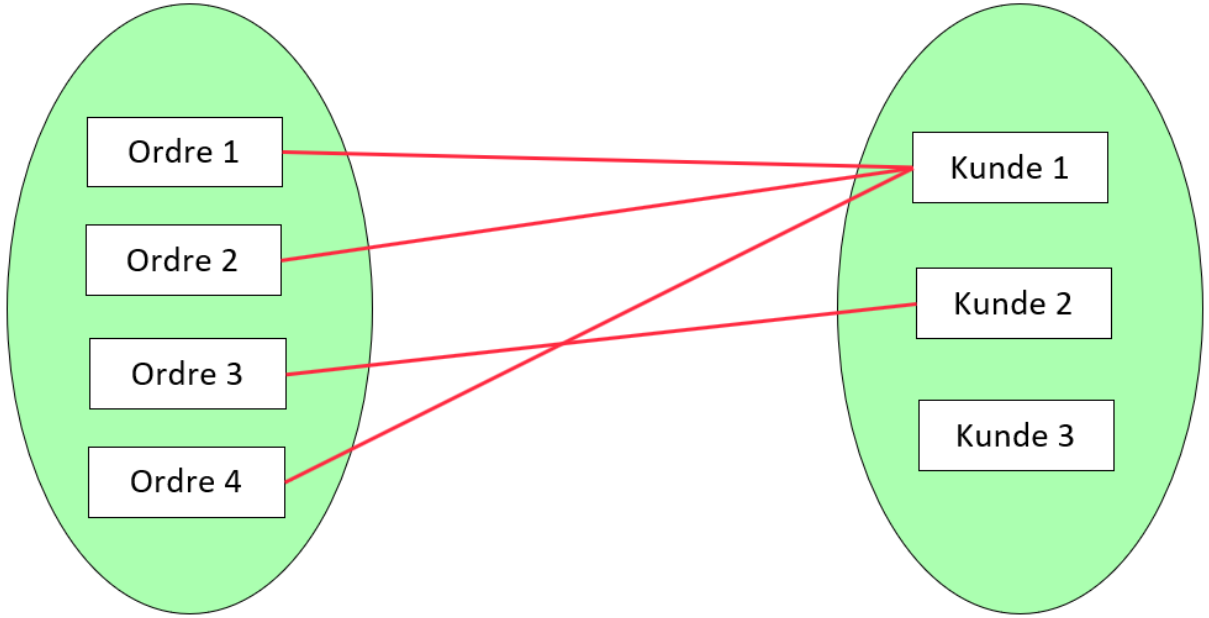
(one-to-one)

M:N

(many-to-many)

Relationship Type and Instance

- **Instance**: the ownership relationship between client 2 and order 3.
- **Type**: the collection of all ownership relationships (all lines).



Minimum and Maximum Cardinality

One department (avdeling) has at a minimum 0 and at maximum many employees, while one employee works in exactly one department (minimum 1 and maximum 1).

- The symbol closer to the entity indicates the **maximum cardinality of the relationship.**
- The **crow's foot** (Kråkefot) means «many», while | means 1.
- The inner symbol shows the **minimum cardinality**.
- Circles stand for 0 ; and | means 1.

Minimum and Maximum Cardinality

Simplified notation (used in some dialects):

- One circle alone stands for maximum=minimum=0.
- One | alone means maximum=minimum=1.

In general, it is possible to use *numbers* other than 0, 1 and «many».

- Such as: 3..7
- The distinction between 0, 1 and «many» determines the table structure.

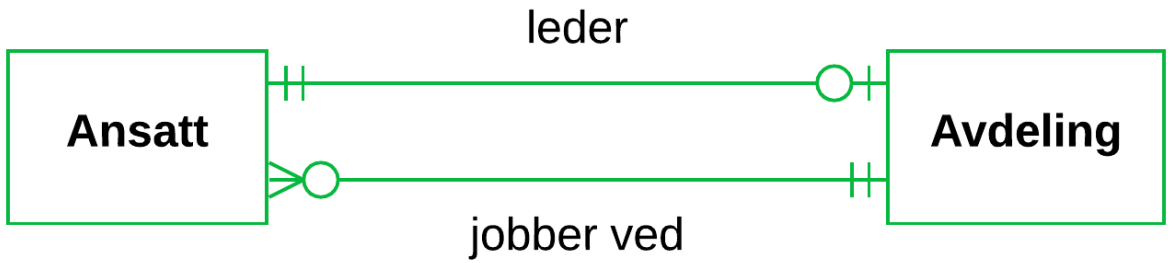
➢ We use both minimum and maximum:

- Circle and crow's foot mean «0 or many».
- Vertical bar and crow's foot mean «1 or many».
- Circle and vertical bar mean «0 or 1».
- Two vertical bars mean «exactly 1».

Roles and Names of Relationships

A named relationship makes the model more **readable**.

- Important when showing several relationships between the same two entities.



We can also use **roles**.

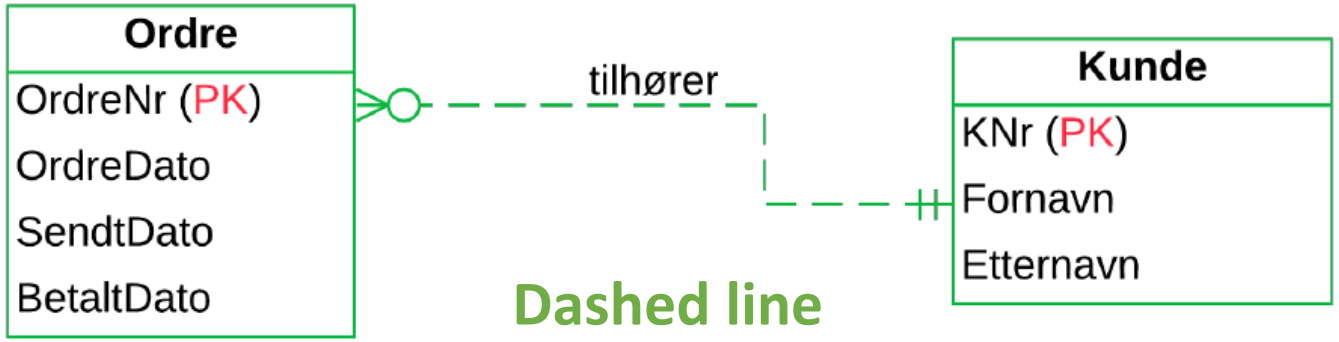
- A teacher can take the role of examiner (sensor).

Exercise for breakout rooms:

Which entities could be used in a database of the university buildings ?

- Buildings
- Departments
- Courses

Non-Identifying Relationships



We add more details:

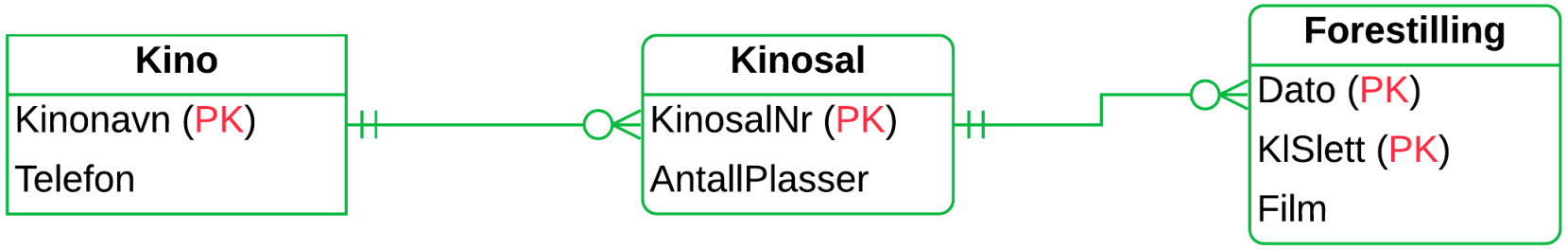
- The entities get **attributes** and we indicate the **primary keys** (denoted as PK).
- We use **dashed lines** for **non-identifying relationships.**

Weak Entities And Identifying Relationships

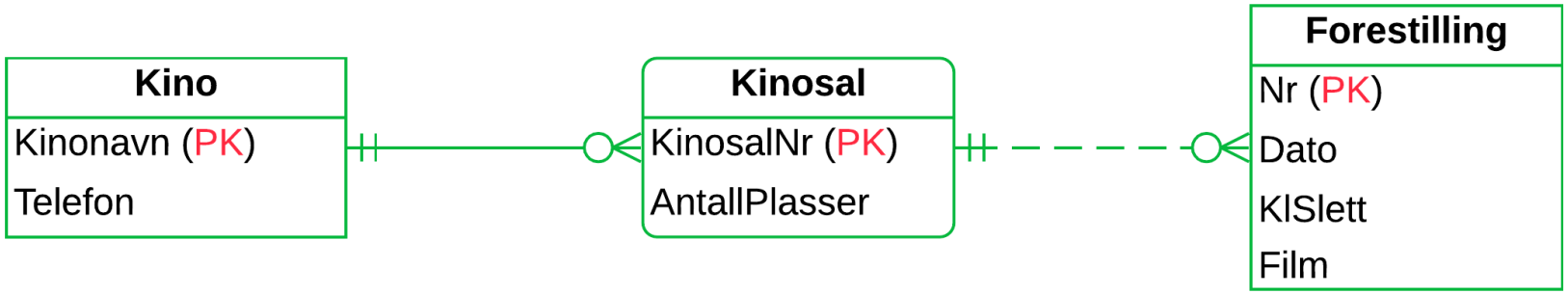
- A **weak entity** *inherits parts of its identifier from another entity* and cannot exist without the other.
  - A cinema hall cannot exist without the corresponding cinema.
  - The identifier (PK) for the cinema hall is partially derived from the identifier of the cinema.
- in MySQL Workbench use **identifying relationship**.

Serial numbers vs. weak entities

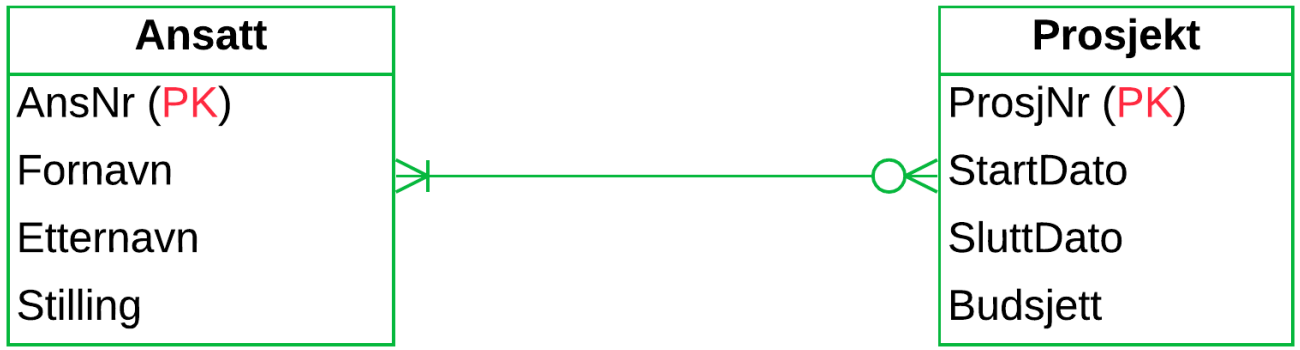
- Extensive use of weak entities can lead to primary keys composed of multiple columns.



  - Can always be replaced by a **surrogate key** (serial number).



Resolving of many-to-many relationships



Are there interesting properties (attributes) of this relationship ?

Quizz on Datamodelling with ER (part 1)

Please answer the practice quizz on mitt.uib now ☺

(you can take it again later if you want)

Link:

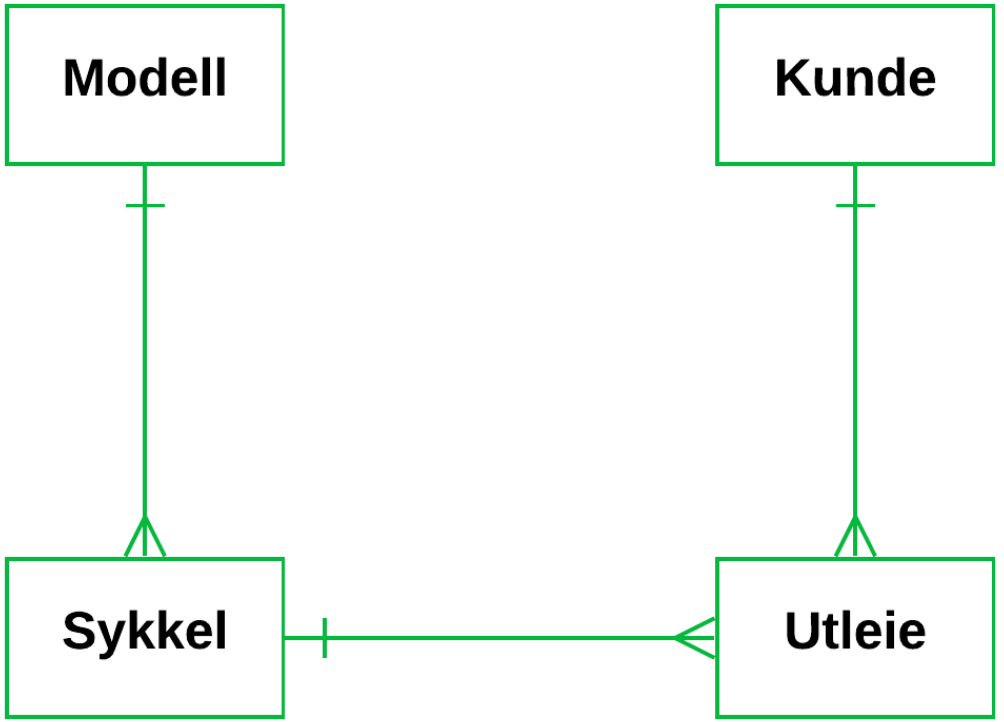
- https://mitt.uib.no/courses/27455/quizzes

(ćÈ³ç Ь
OÈ¿đėćÈ ćÈċėïÈċ Ûð pп ïÛðėđÈċ

Models at different levels of abstraction

Sometimes it is not desirable to show all details in a data model.

- Overview without attributes.
- Does not inform about weak entities and identifying relationships.

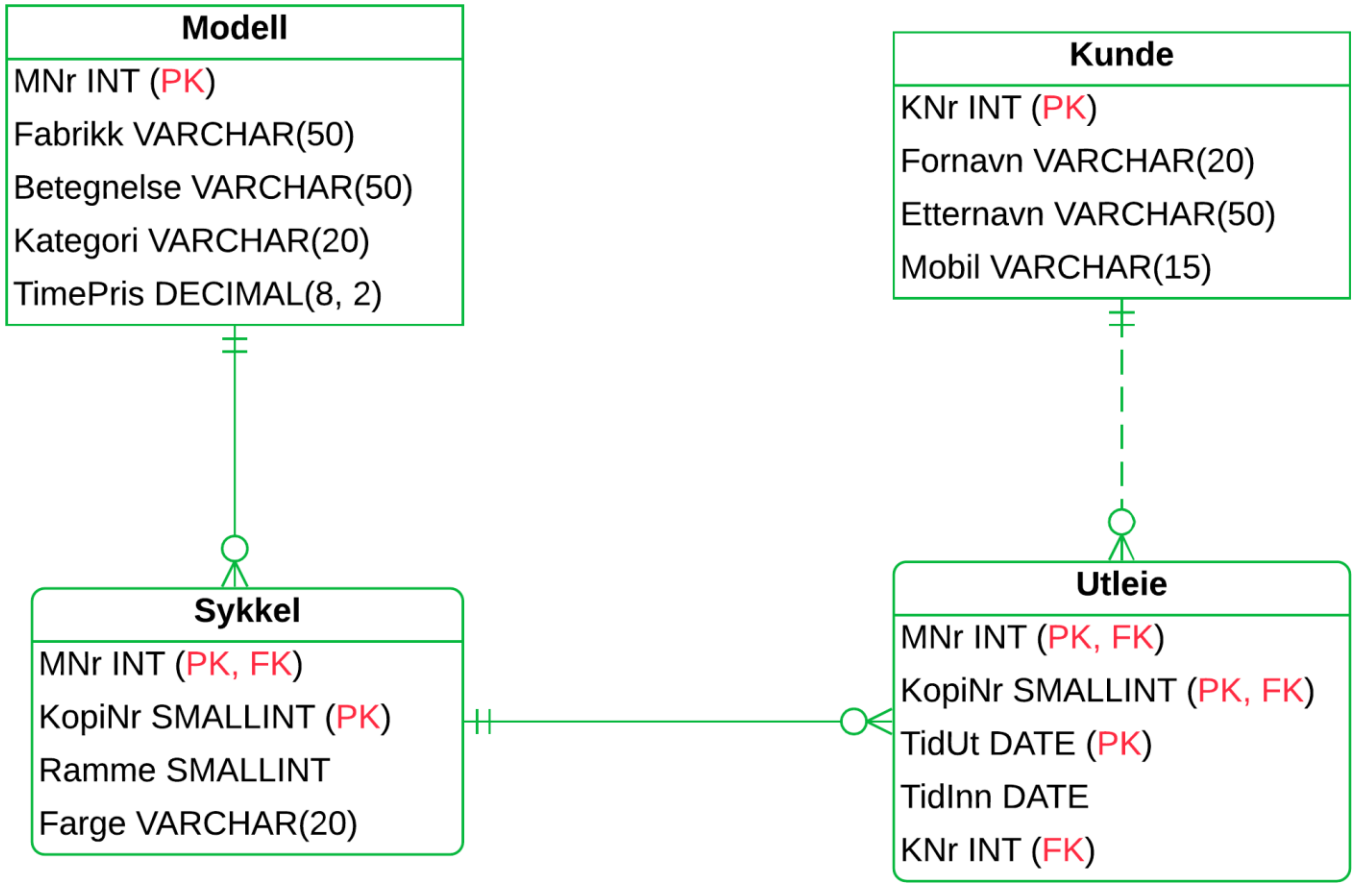


More detailed data models

A **logical data model** gives a *precise description* of the database with:

- **Primary Keys** (PK),
- **Foreign Keys** (FK),
- and **Data Types**.

In Chapter 8 we will go from
the *conceptual*
to the *logical* data model.



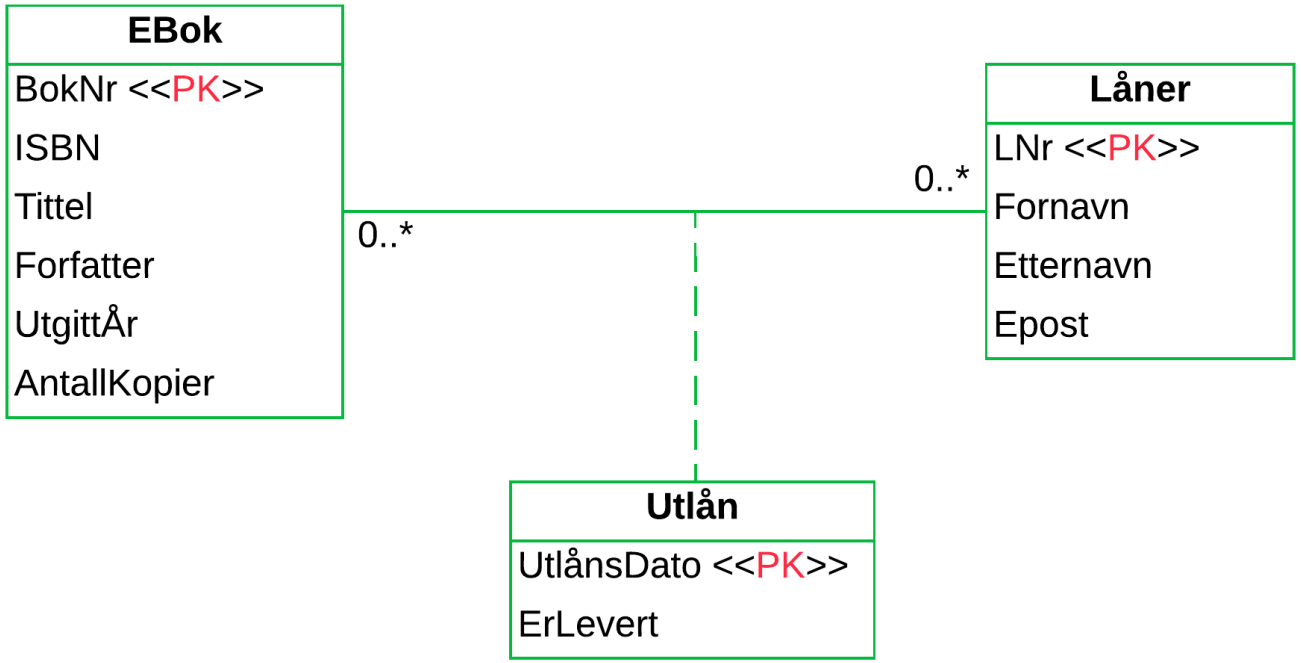
Data modelling with UML

There are many ways to draw ER-diagrams: many «dialects».

- **UML** (Unified Modeling Language) is a **visual modelling language** with a variety of uses in system design.
- **Class diagrams** can be used for database design.
  - *Relationships* are called *associations* in UML.
  - The notation is a bit different for identifiers (<<PK>>) and cardinalities (1..1, 0..*), and we can indicate a «reading direction».

Properties of Relationships

- They can be shown in UML by putting attributes in an **associative entity** (class)
- Which is connected ot the ***association*** by a dashed line.



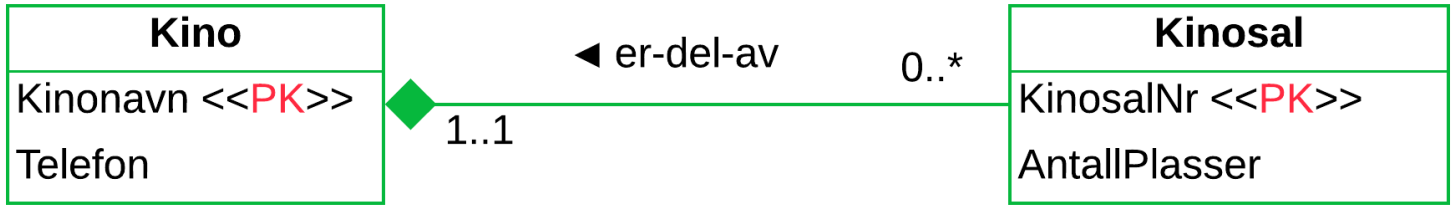
Aggregation and composition

- **Aggregation** is used to model that an entity is a part of another entity, for example that a municipality (kommune) is a part of a county (fylke).

  This is drawn in UML with a little **empty diamond**.

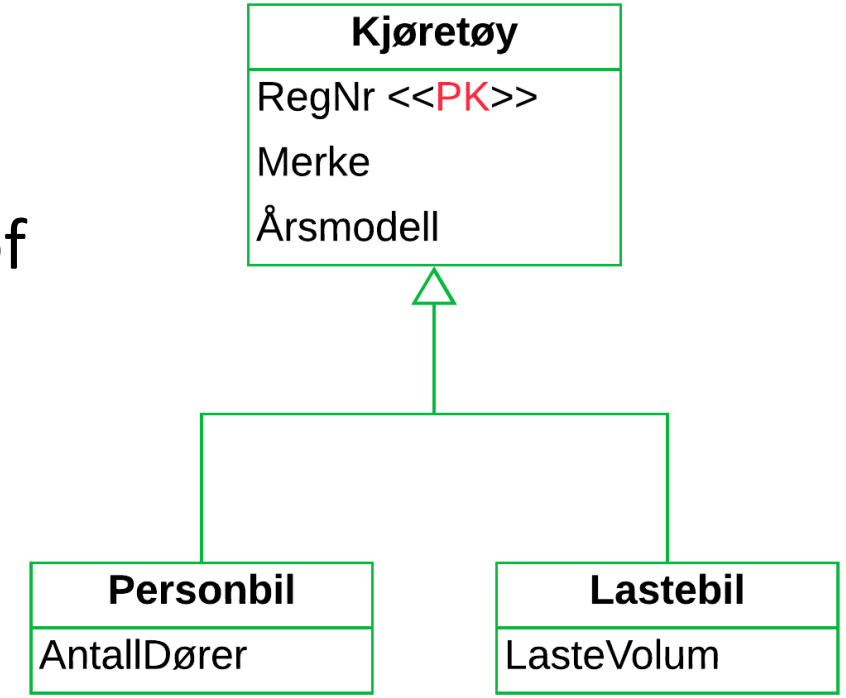
- **Composition** is a stronger form of aggregation that is used when the life cyle of the two entities coincides. A cinema hall ceases to exist when the cinema is torn down.

  This is drawn with a **filled diamond**.



Subtypes

- A **subtype** is a specialistion of another entity.
- Substypes **inherit** properties of the entity and can also have **additional attributes** and take part in **other relationships**.
- Subtypes correspond to **subsets** of the sets of instances.
- Check it yourself by replacing in «A is a B».
  - A car is a vehicle.
  - A truck is a vehicle.
  - But: a client is **not** an oder!



Common abstraction methods

❖ **Specialisation / generalisation**
- *Seller* is a **specialisation** of *Employee*.
- *Employee* is a **generalisation** of *Seller*.

«Is-an»

❖ **Aggregation / decomposition**
- *Bicycle* is an **aggregation** of *wheel, frame, seat, steer* and *pedals*.
- **Decomposition** of *Bicycle* yields …

«Has-an»

❖ **Categorisation / instantiation**
- *Employee* is a **categorisation** of Hans, Lise, …
- Lise is an **instance** of *Employee*.
- Compare with type and instance.

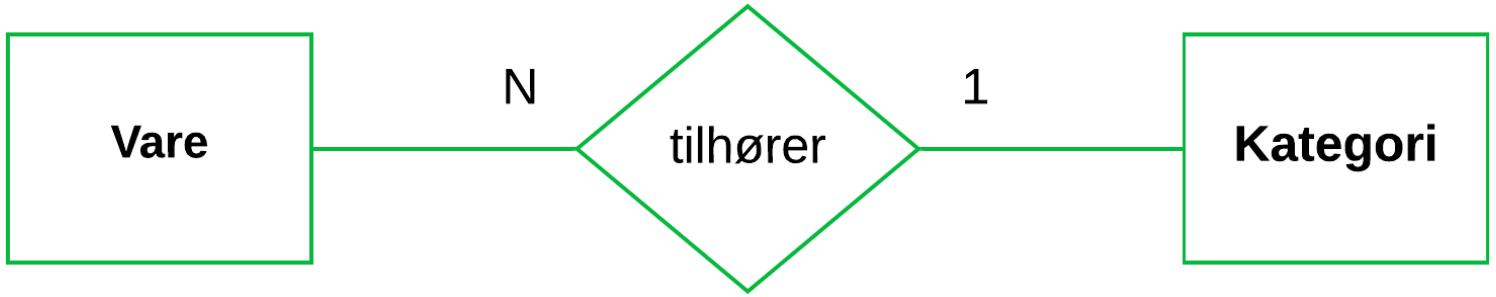
Chen-notation

ER was invented by Peter Chen.

The crow's foot notation, UML og Chen-notation are «dialects» of ER.

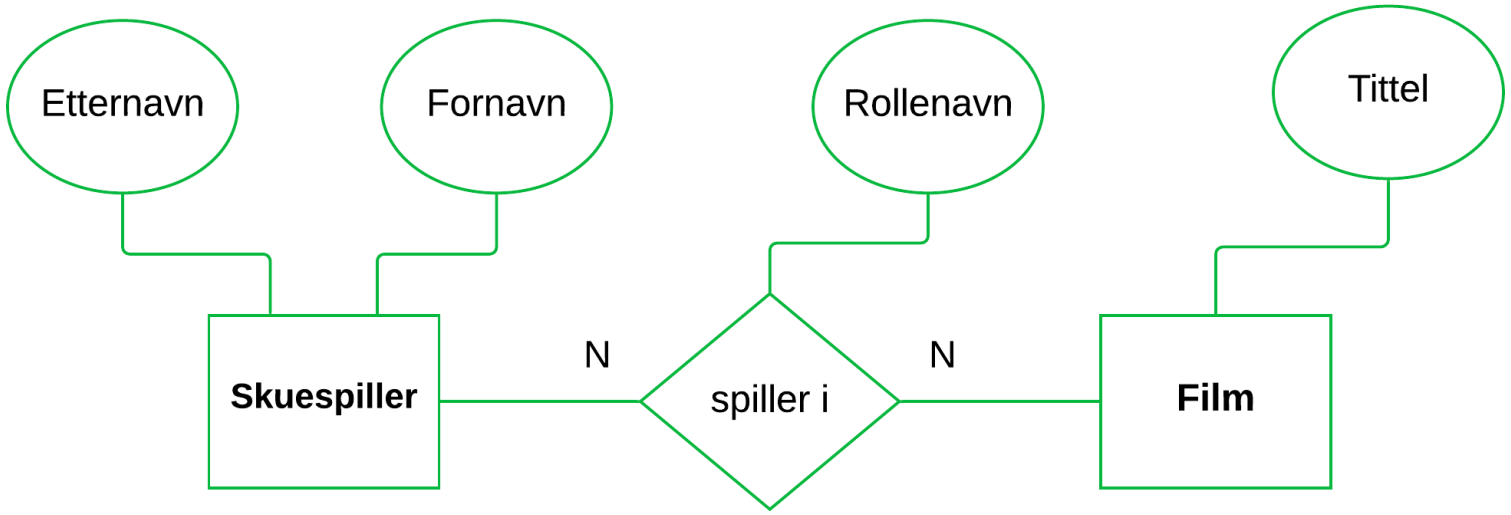
- ❖ A relationship is drawn as a diamond.
- ❖ Cardinality is given as «1» and «N».

- In the example diagram below the attributes are omitted.



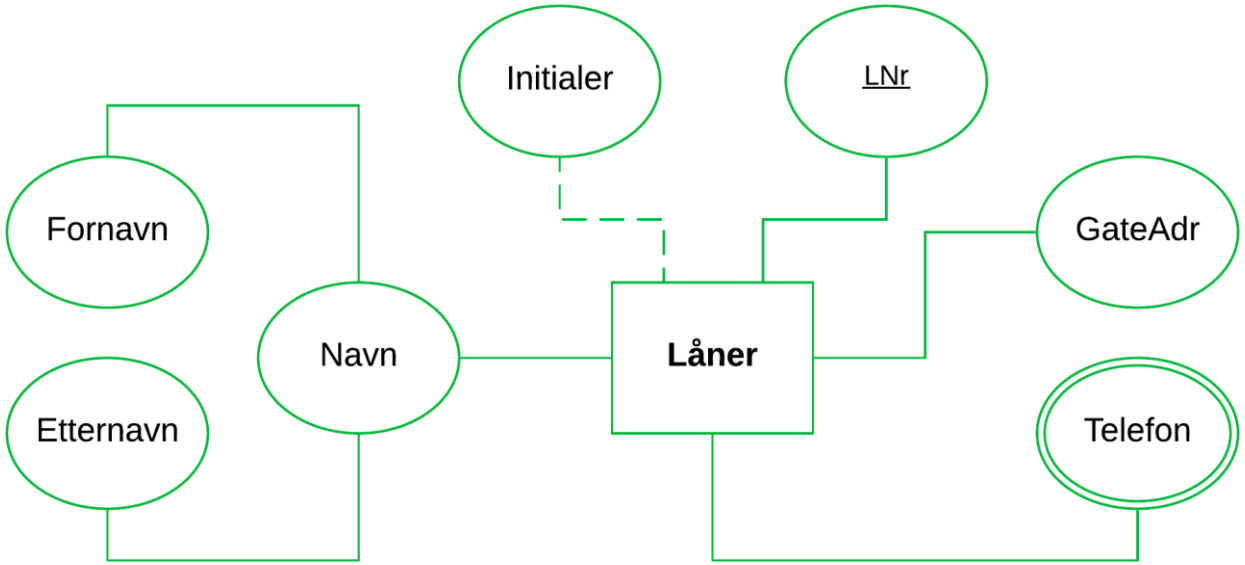
Relationships in Chen-notation

- Attributes a drawn in ovals outside entities.
- One can connect attributes to relationships (e.g. *Rollenavn*).

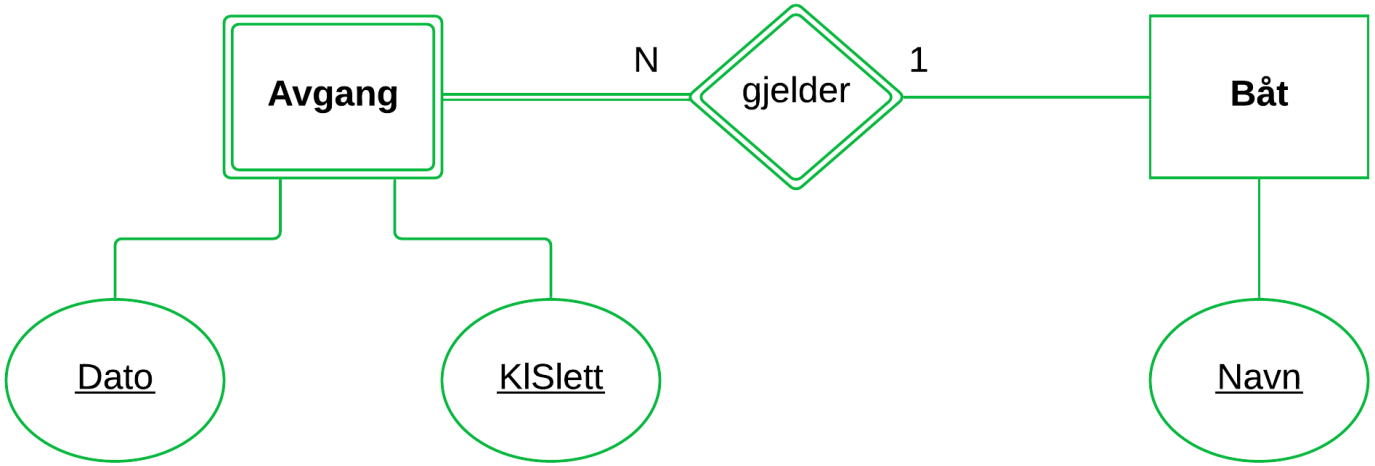


Non-atomic attributes (Chen)

- **Composite attributes**: Names can be decomposed into first names and last names.
- **Derived attributes**: *Initialer* can be computed.
- **Multivalued attributes**: *Telefon* (numbers) can be a list of values.



Weak Entities (Chen)



- **Double lines** indicate weak entities.

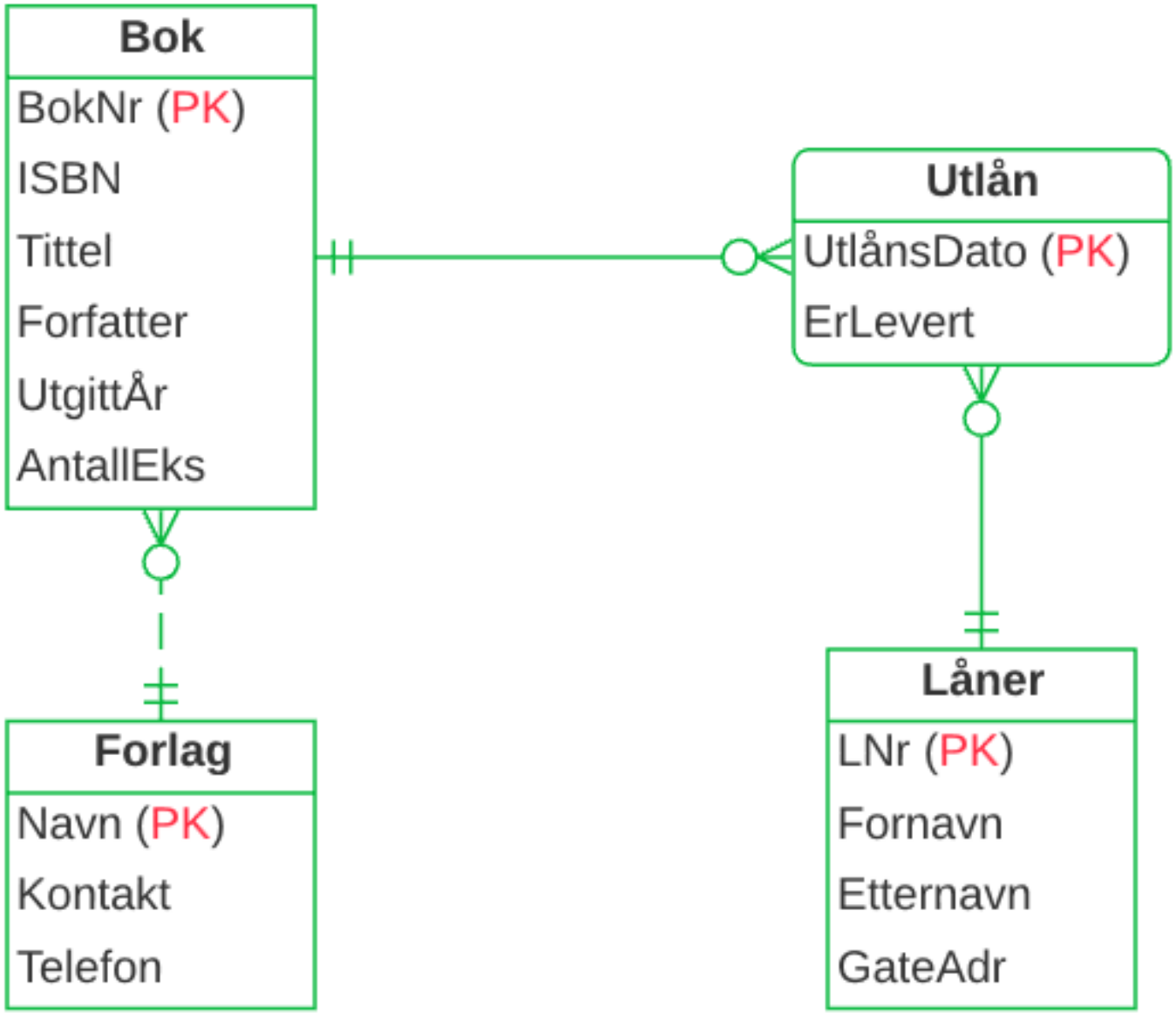
- *Avgang* is a weak entity.
- *Avgang* inherits part of the identifier of *Båt.*
- The primary key (identifier) for Avgang is Navn + Dato + KlSlett.

Non-atomic attributes with UML

Informally we could extend ER/UML to include non-atomic attributes.

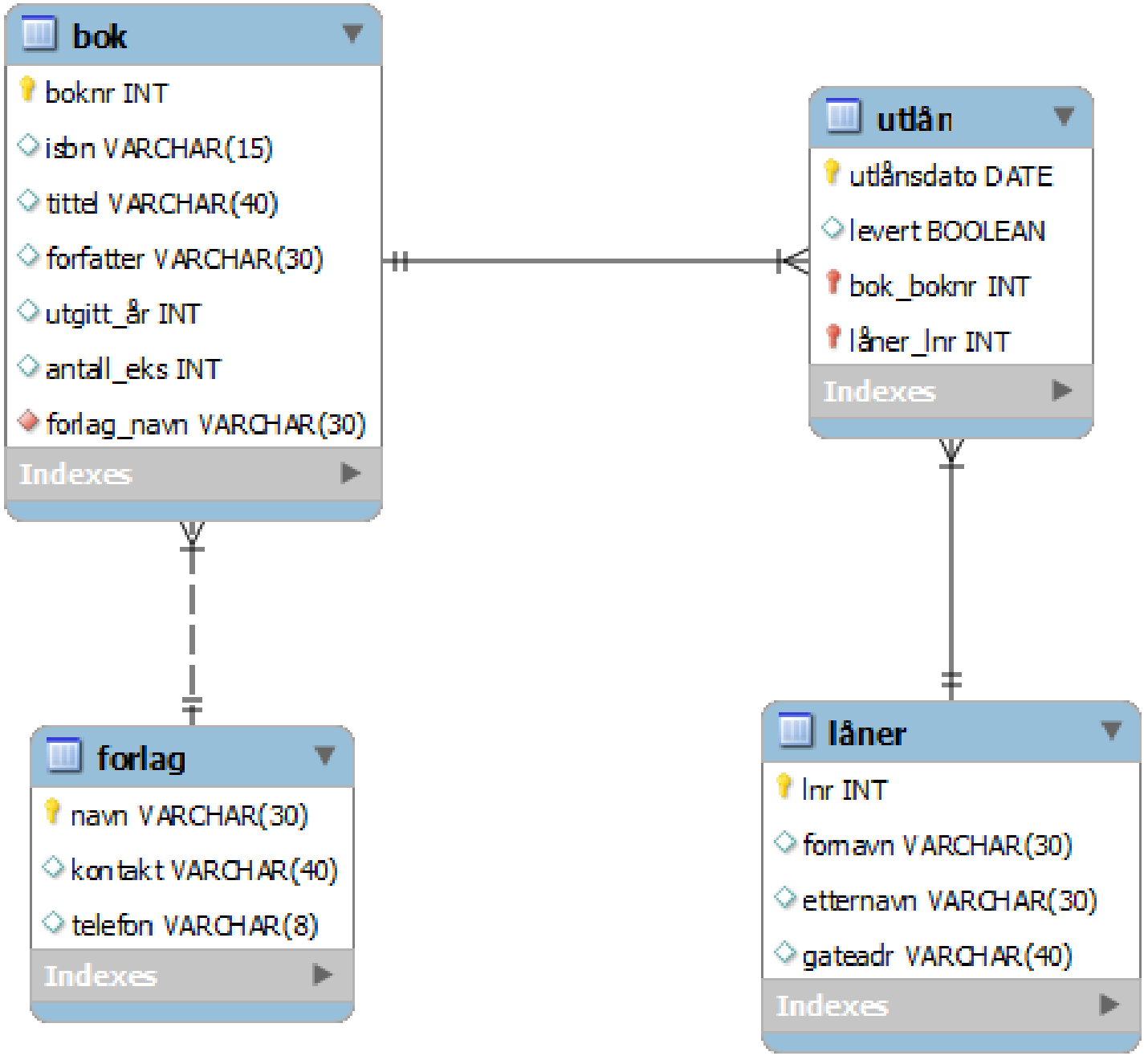
- **Composite attributes**: indentation.
- **Derived attributes**: preceeding slash.
- **Multivalued attributes**: square brackets.

A *conceptual* ER-diagram



The same ER-diagram in MySQL Workbench

Note the *different symbols* and *additional details*.



Differences between conceptual and Workbench diagrams

- Workbench is designed for **MySQL**.
  - The conceptual ER is independent of the DBMS / tool.
- Workbench shows **datatypes** for the attributes.
- Workbench shows **foreign keys** (with a red icon) in the model.
  - But we do not manually set up foreign keys.
  - We draw relationships and Workbench takes care of the foreign keys !
- Workbench shows **primary keys** with a yellow *key icon* (red icon for columns that are also foreign keys).
  - In some versions the inherited primary keys are not shown.
  - The conceptual diagram indicates them with **PK.**

Quizz on Datamodelling with ER (part 2)

Please answer the practice quizz on mitt.uib now ☺

(you can take it again later if you want)

Link:

- https://mitt.uib.no/courses/27455/quizzes

Chapter 7: *Data Modelling with ER*

Summary of part 1:

- Goals of data modelling and database design.
- **ER** and **UML** are **visual modelling languages**.
- **Entities** with attributes and identifiers (PK).
- One-to-one, one-to-many and many-to-many **relationships**.
- **Weak** entities and **identifying** relationships.
- **Common abstraction methods.**