

Business Informatics 2 (PWIN) WS 2025/26

Database Management I Databases & Data-oriented Modelling

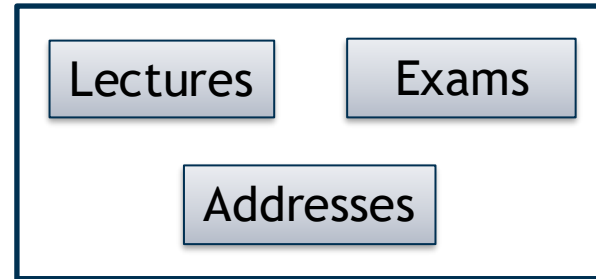
Prof. Dr. Kai Rannenber

Chair of Mobile Business & Multilateral Security
Johann Wolfgang Goethe University Frankfurt a. M.

- Data Organisation Basics
- Databases
- Data-oriented Modelling

Data Organisation Hierarchy and Basic Notions

Database



Dataset

Name	Lecture	Date	Grade
Frank Miller	PWIN	SS10	1.5
Mike Beasley	OWIN	WS10	2.0
John Schulz	PWIN	SS10	2.7

Data
Element

Name	Lecture	Date	Grade
Frank Miller	PWIN	SS10	1.5

Byte

01011101

Bit

0

Data Organisation Hierarchy and Basic Notions

- A **bit** is the smallest data storage unit (1 or 0).
- A **byte** consists of eight bits and represents a character (e.g. ABCD..., 1234..., ?!"\$\$\$, ...).
- A **data element** consists of a series of bytes. It summarises characters as word, number, or group of words.
- Data elements, which have a relationship to each other can be grouped as **data set** or **data table**.
- Data sets or data tables physically or logically belonging together are organised and stored in a **database**.

Further Basic Notions on Data Organisation

- **Entity**

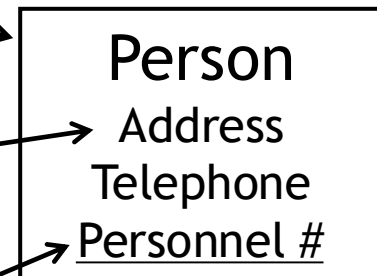
- Phenomena, e.g. a person, a place, a thing or event for which data should be stored.

- **Attribute**

- Property, which describes a specific entity.

- **Primary Key**

- Uniquely, identifiable attribute, e.g. social security number.



Issues of Data Organisation: Redundancy and Inconsistency

- **Data Redundancy** denotes the existence of multiple identical data elements in different data sets or databases.
 - E.g. identical customer contact information stored in multiple databases across an enterprise
 - Consequence:
 - High effort to keep this contact information in sync across all databases at the same time, otherwise data inconsistencies may occur
 - + Can reduce the risk of data loss
- **Data Inconsistency** denotes the mapping of different values to a single attribute of an data element across multiple data sets or databases.
 - Phone Number of Jon Steward in Dataset A: 404-42349234 and in Dataset B: 404-12346323
 - Bank Transfer: Transfer amount got deducted from Account A but was not credited to Account B.
 - Different names or codes for a data element attribute (e.g. product code vs. product ID)

- Data Organisation Basics
- Databases
- Data-oriented Modelling

Database Systems and Databases

- **Database Systems** in general consist of a Database Management System (DBMS) and a group of data (the actual database), which exhibit logical dependencies between each other.
- A **Database** is a collection of data, which is designed to ensure an efficient and concurrent access for multiple users or applications to (as far as possible) redundancy-free data.

- Rather than storing data locally along with individual application systems, databases offer a central, common storage place for all applications.
- Example:
 - A common database containing data of employees such as
 - Employee ID, working hours, salary, tax category ...
 - Unemployment-, retirement-, or health insurance, ...can be accessed by the human resource as well as the accounting department.

- **Data Administration** is a special organisational role for the administration of data resources in an enterprise.
- This role is in charge of the implementing and enforcing policies with regard to data planning, data collection, data quality standards, data maintenance as well as data usage and data transfer.
- In this context, DBMS can be of significant value, if
 - enterprises acknowledge the strategic importance of information,
 - actively maintain information as resource,
 - consider existing information for their planning and activities.

- Data Administration requires specifications for:
 - Data collection planning
 - Coordination of the logical database design
 - Development of data usage policies
 - Monitoring of data usage

Database Management Systems (DBMS)

- The use of databases requires a DBMS.
- A DBMS is defined as follows:
 - Collection of applications for the creation, administration, and use of a database. This collection allows multiple application systems to concurrently store, access and modify their required data without the need to store this data locally.
- A DBMS separates physical and logical data structure
 - *Physical*: Shows how data is actually organised and stored on the physical storage medium
 - *Logical*: Describes the organisation of data based on the perception of a user using logical concepts which abstract from any technical implementation

Database Management Systems (DBMS)

- For the creation, manipulation, and analysis of databases, a DBMS offers multiple components:
 - Data Definition Language (DDL)
 - Defines the structure of the database contents
 - Data Manipulation Language (DML)
 - Used to modify data stored in the database (e.g. SQL)
 - Data Dictionary (DD)
 - Aid for the administration of existing data stocks
 - Generates reports as overview of data stored in a database

- **Hierarchical Data Model**
(out-dated)
- **Network Data Model**
(out-dated)
- **Relational Data Model**
(most commonly used in the industry)
- **Object-oriented Data Model**
(latest model, but compared to the relational model only rarely used)

- Most commonly used concept for the organisation of data in databases
 - Suitable for Ad-hoc queries
 - Flexible with regard to adding new data or combining data from multiple different data sources
 - In relational data models, all data in a database is represented as tables (relations) with a fixed number of columns and a variable number of rows.

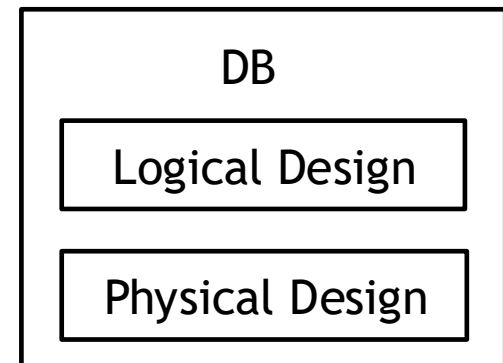
Name	Lecture	Date	Grade
Frank Müller	PWIN	SS10	1.5
Meike Heinrich	OWIN	WS10	2.0
Hans Mann	PWIN	SS10	2.7

- **Columns** represent data elements or attributes respectively, which describe entities.
- **Rows** represent concrete data elements. A row in a relation is called “Tuple”.
- Relations, for instance, could be: order, product, supplier
- A Tuple, for instance, could be:
 - 137;Door lock; 22,50;4058
 - 145;Door handle;26,25;2038
 - ...

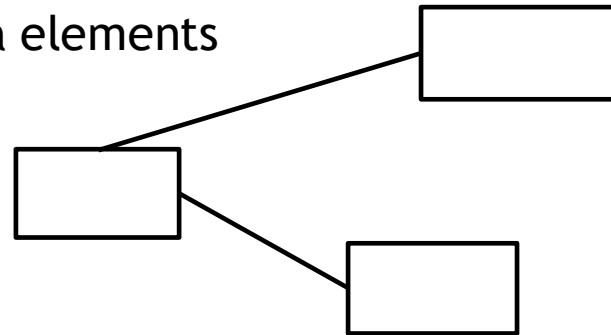
Object-oriented Data Model

- Weaknesses of relational data models with regard to ...
 - Engineering design applications
 - Multimedia applications
 - Architecture concepts, etc.
- With object-oriented modelling, individual limitations of relational data models could be eliminated.
 - For instance, use of complex objects
 - Definition of complex objects, whose attributes itself can be objects
 - Attributes can be simple data types or can be comprised of an indefinite number of data types.
- Solutions: Object-oriented DBMS (OODBMS), such as Matisse, POET or Object Store

- The task of the data administration is the coordination of the database design.
- Database administrators are in charge of defining and structuring database contents as well as of maintenance and administration of the database.
- The approach is divided into two processes
 - Logical database design
 - Physical database design



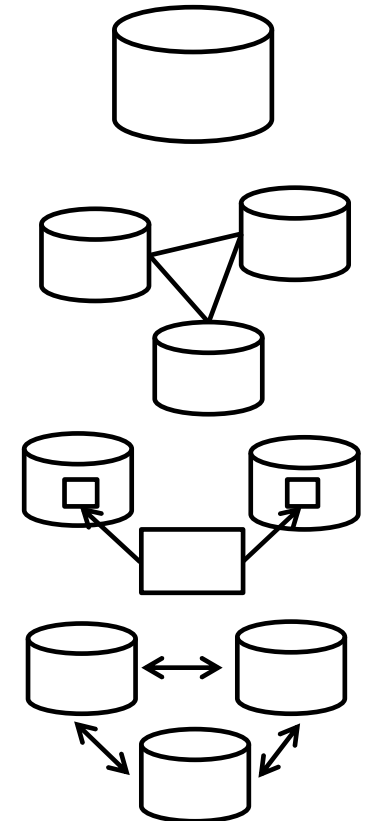
- The design of a database raises some questions:
 - Which relationships exist between the data?
 - Which data types are required?
 - How is the stored data going to be used?
- The logical database design denotes an abstract model of the data to be stored. It contains:
 - Relevant entities
 - Relationships between the data elements



- Modelling Tool: ER-Diagrams

Physical Database Design

- The physical database design addresses the question how the data is stored physically.
 - Central database
 - Distributed database
 - Logically connected, but physically in different places
 - Fragmented or partitioned databases respectively
 - Tuples of a relation are stored in physically different storage places.
 - Replication as alternative to fragmented databases

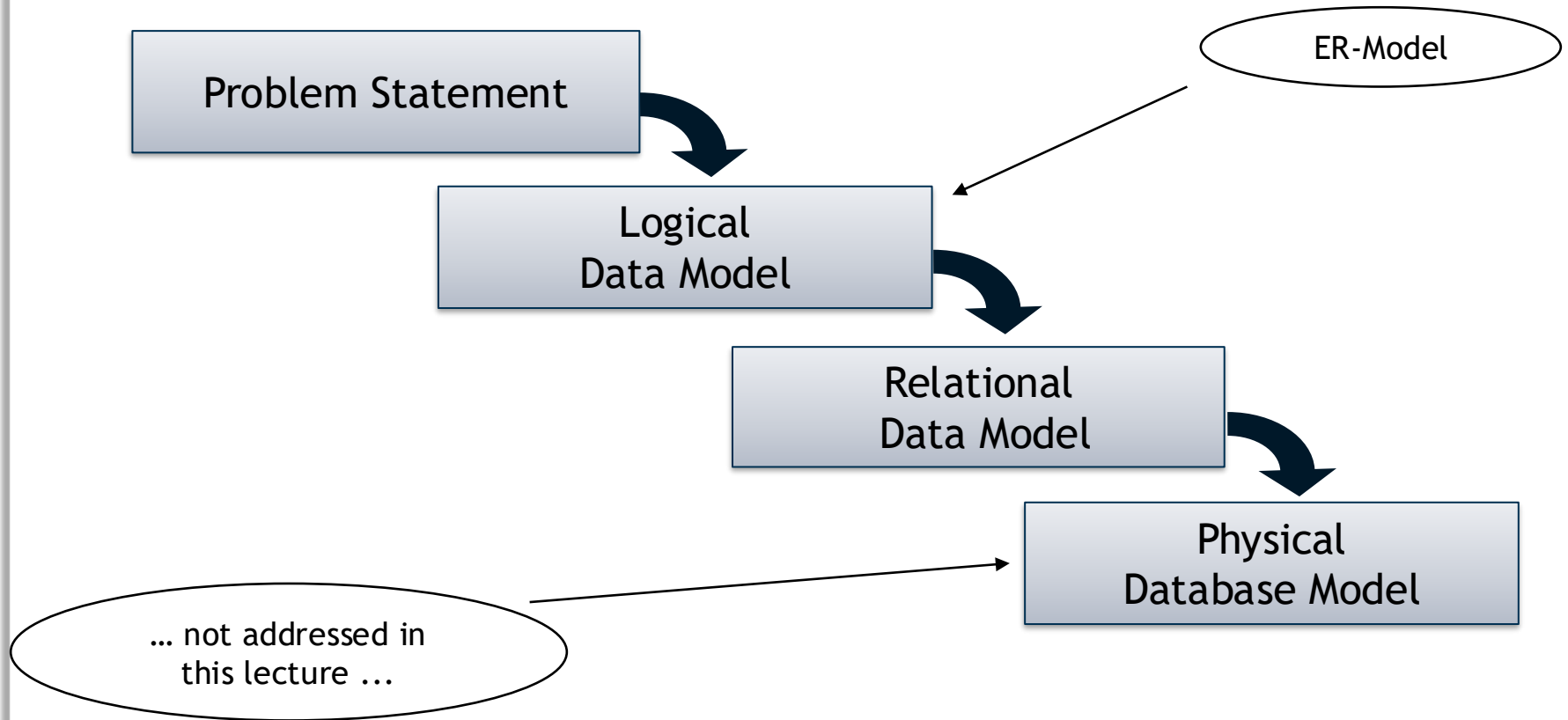


- Development of databases requires, besides the actual database design, also a graphical user interface (GUI).
 - Consists of various forms to display and enter data
 - Allows different perspectives on a physical database
 - e.g. different departments (human resources, accounting) access different types of data in the same database

The image shows a screenshot of a web-based registration form. At the top, there is a header bar with the text 'Registrierung'. Below this, a message in German reads '(fett gekennzeichnete Felder bitte immer ausfüllen)'. The main form area is titled 'Persönliche Daten' and contains several input fields. The fields are: 'Firma' (text input), 'Anrede' (dropdown menu with 'Herr' selected), 'Titel' (text input), 'Vorname' (text input), 'Nachname' (text input), 'E-Mail' (text input), 'Strasse' (text input), 'PLZ' (text input), 'Ort' (text input), 'Land' (dropdown menu with 'Deutschland' selected), 'Telefon' (text input), 'Telefax' (text input), and 'Mobil' (text input). The labels for 'Vorname', 'Nachname', 'E-Mail', 'Strasse', 'PLZ', 'Ort', 'Land', 'Telefon', 'Telefax', and 'Mobil' are bolded in the original image.

- Data Organisation Basics
- Databases
- Data-oriented Modelling

Database Design Process based on the Entity Relationship Model



Excerpt of reality to be reflected in/by the model

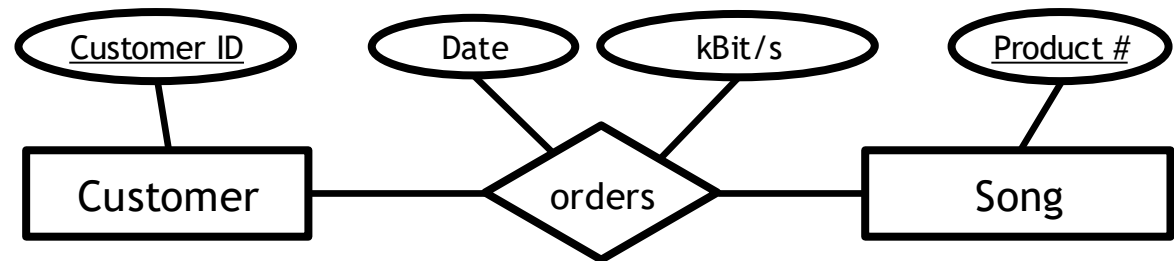
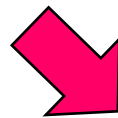


For instance „A customer downloads a song from a music store platform”.

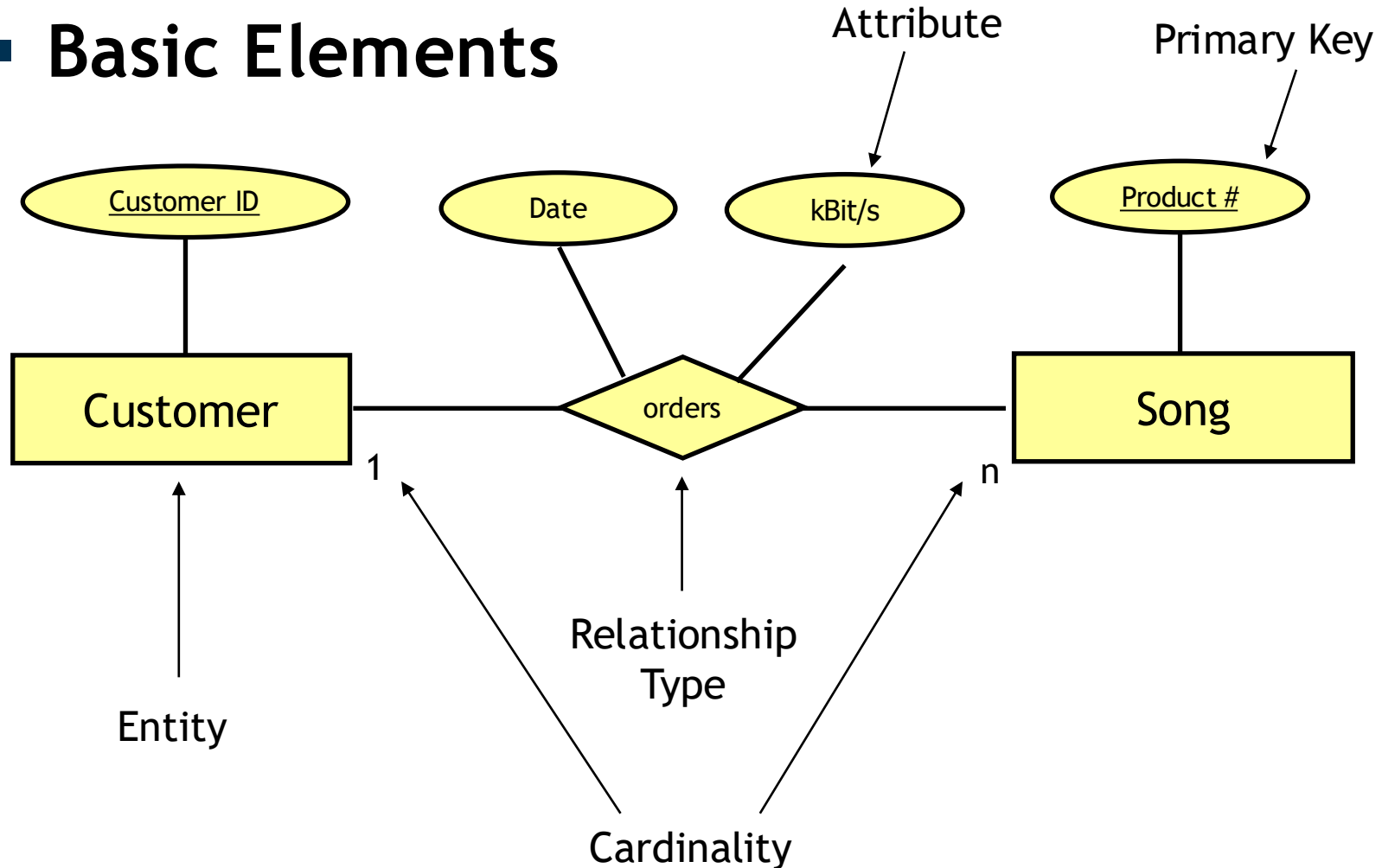
- Modelling of the problem statement from functional perspective
- Abstraction from technical aspects and implementations
- Different modelling concepts (e.g. ERM, SERM, ...) available



A customer downloads a song from a music store platform



■ Basic Elements



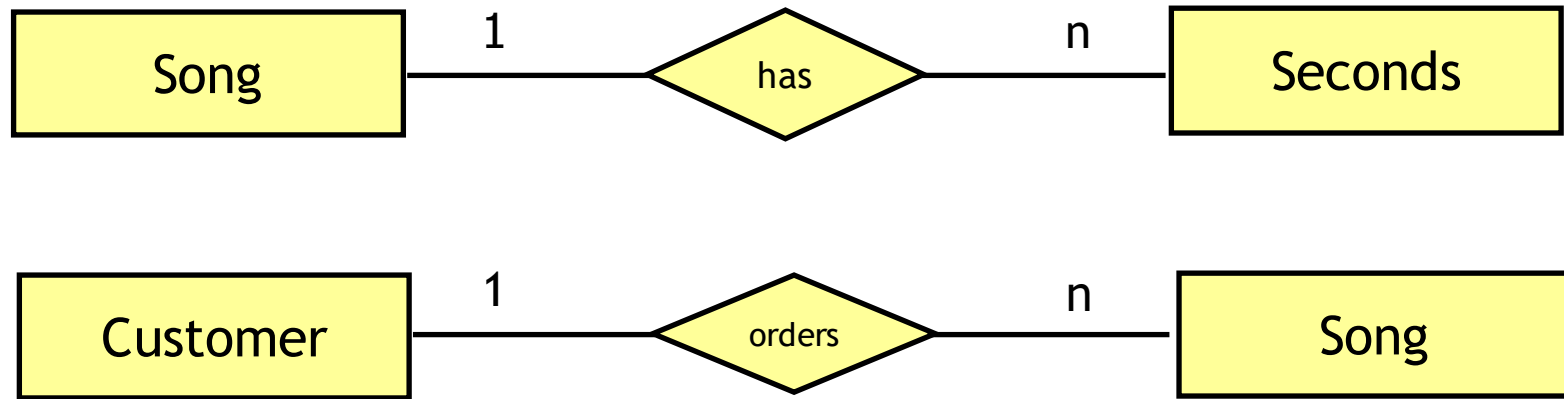
- Advanced Elements
 - Constraints
 - Aggregation
 - Generalisation
 - Recursive Relationship-Types
 - Weak Entities

Cardinalities

- Cardinalities describe the number of relationship instances that an entity can participate in:
 - 1:1 (one-to-one), e.g. student - student card (theoretically)
 - 1:n (one-to-many), e.g. university - student (theoretically)
 - n:m (many-to-many), e.g. professor - student (theoretically)

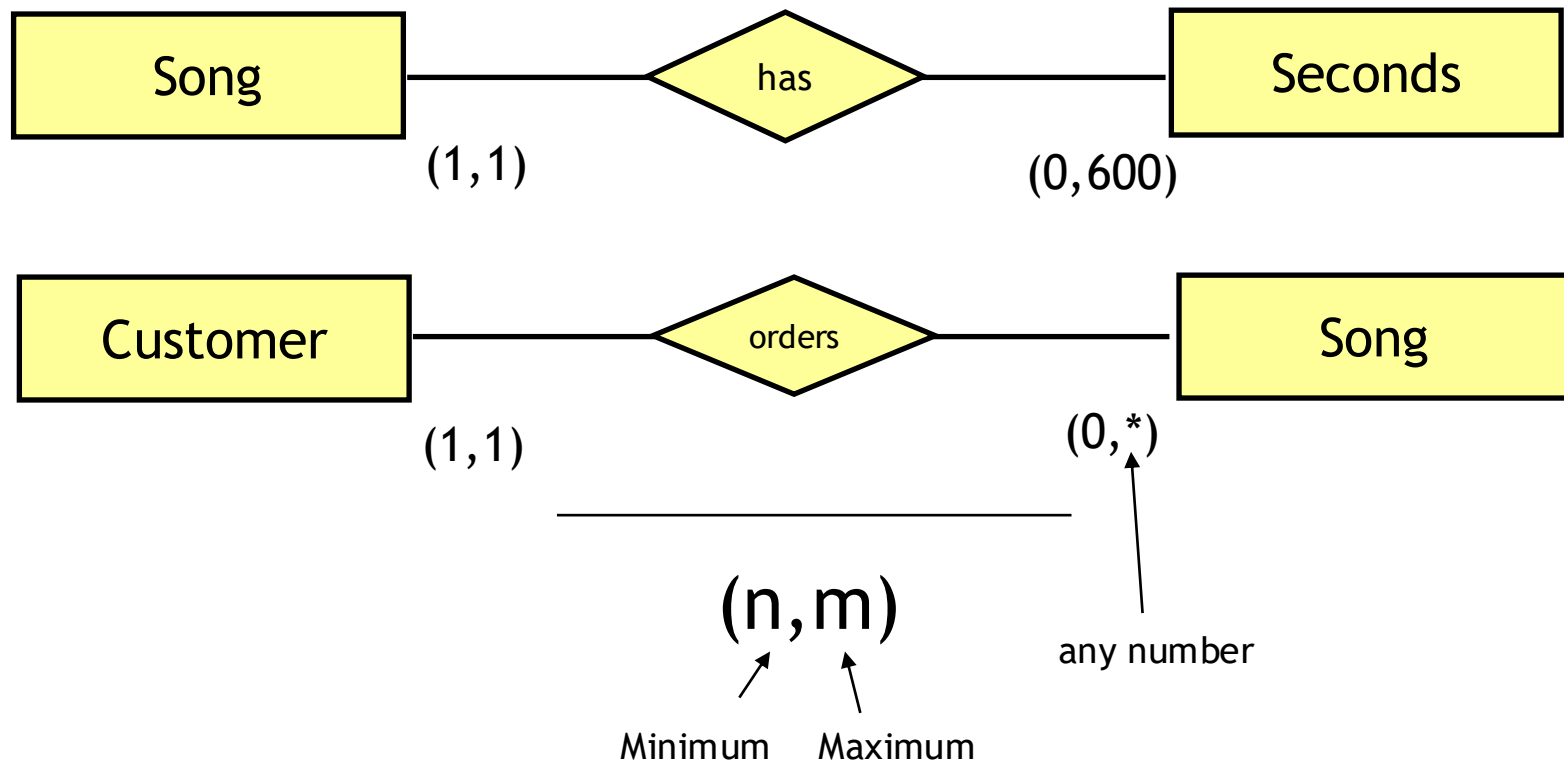
Intervals (min/max notation)

- Intervals allow specifying cardinalities more accurately.
- They specify that each entity participates in at least *min* and at most *max* relationship instances.



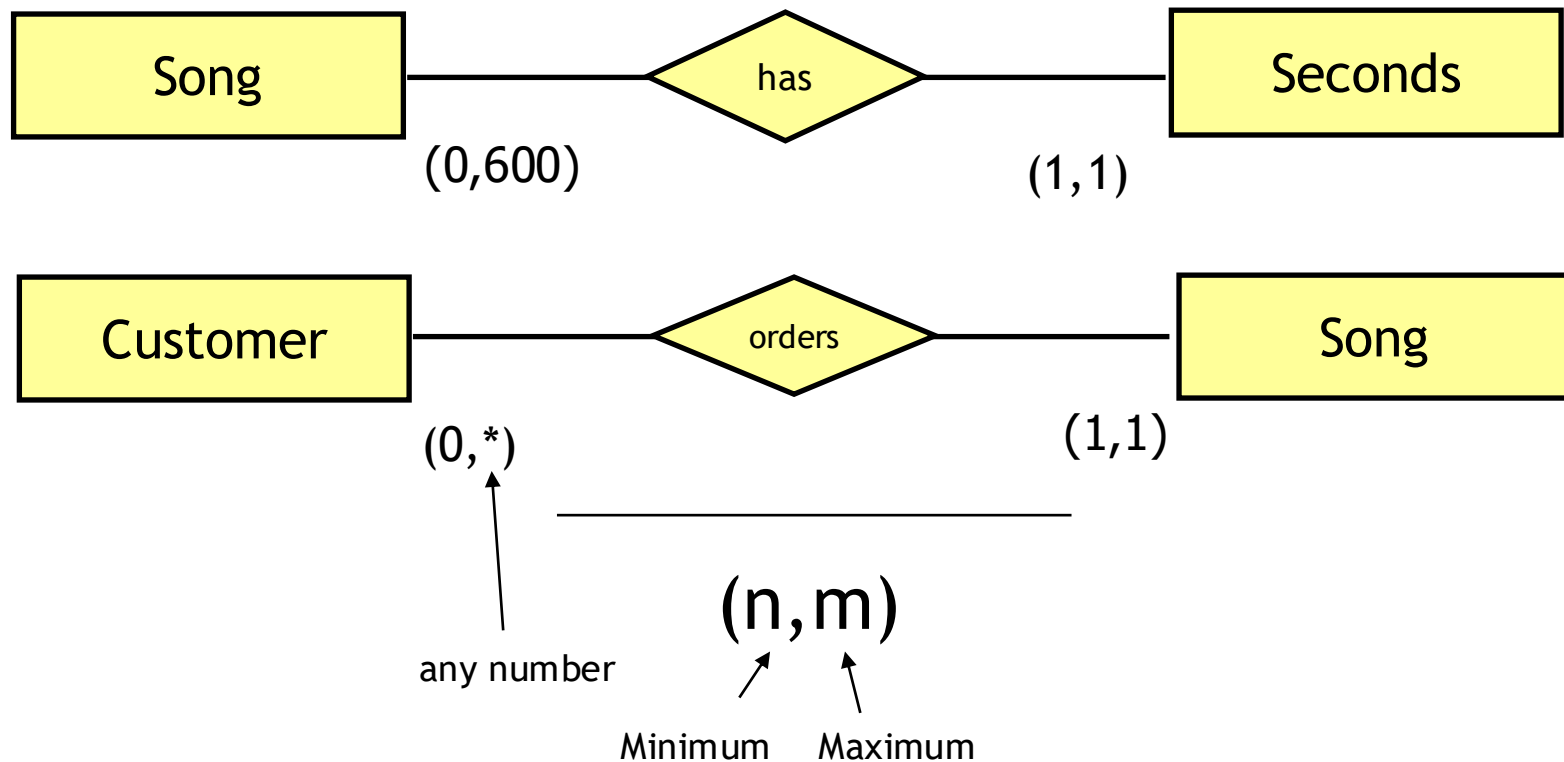
Intervals (according to Schwickert, 2004)

- Intervals allow specifying cardinalities more accurately.



Intervals (according to Ferstl/Sinz 2001)

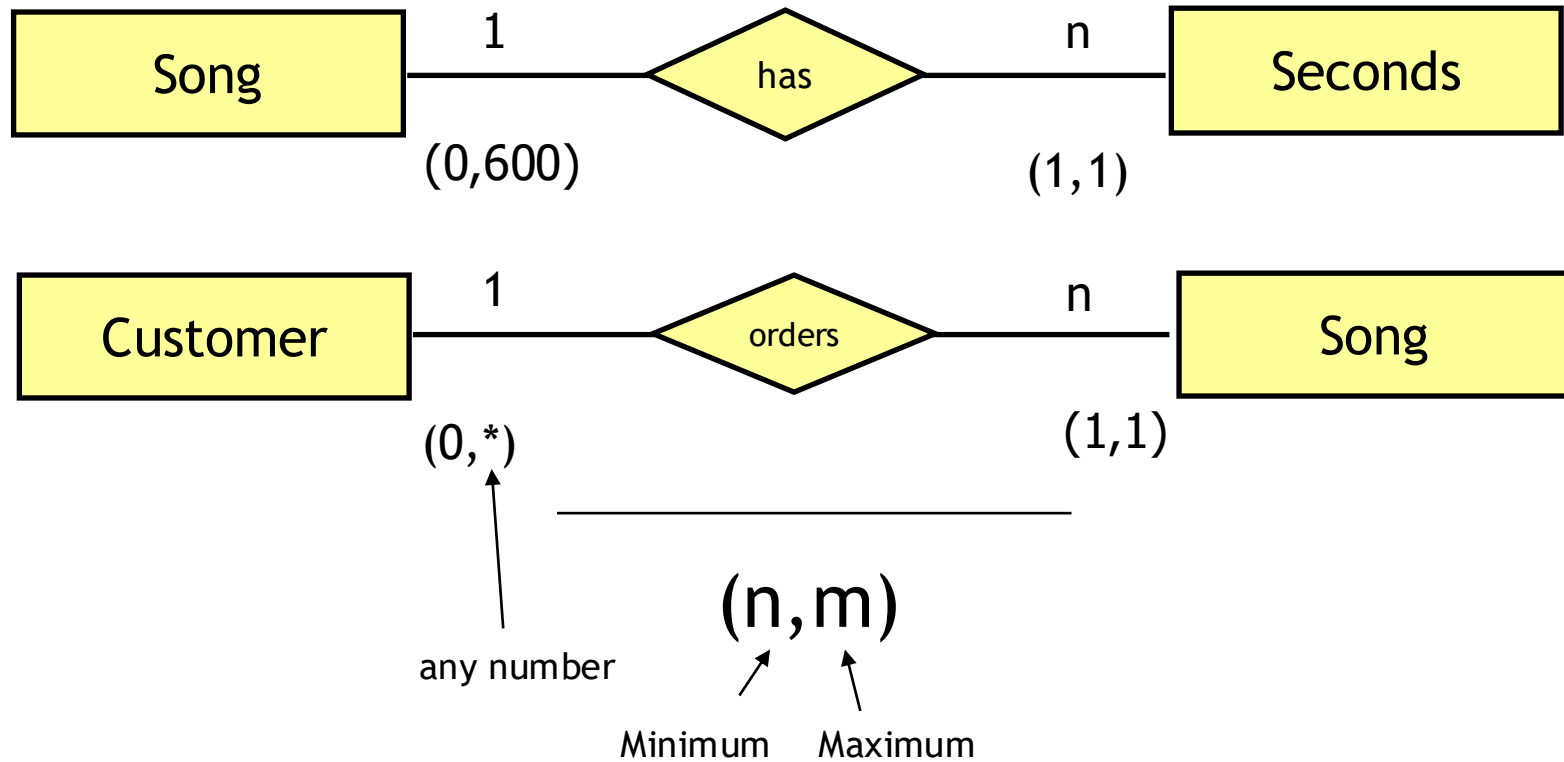
- Intervals allow specifying cardinalities more accurately.



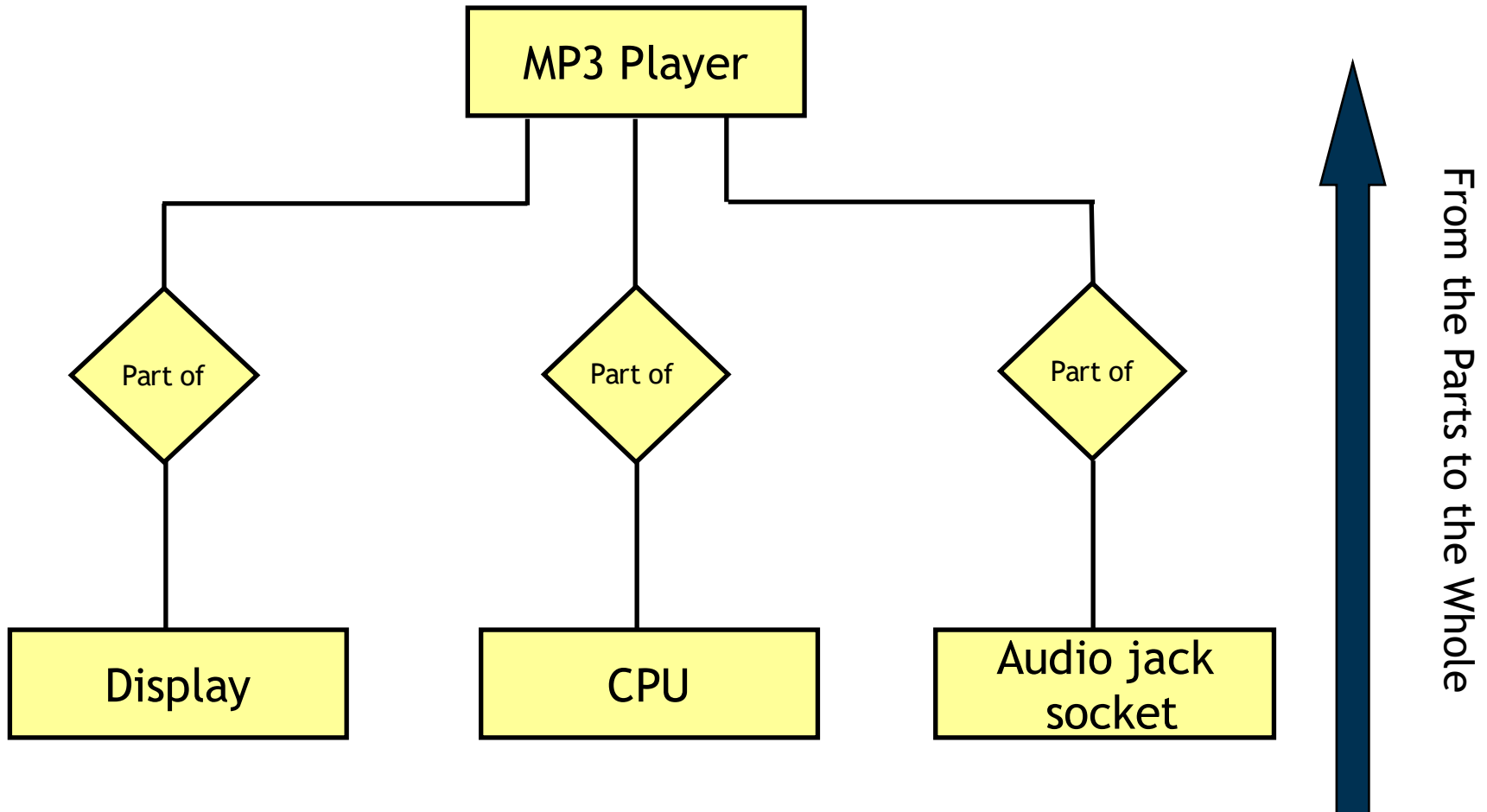
- NOTE:** This lecture follows the widely used approach of Ferstl/Sinz (2001)

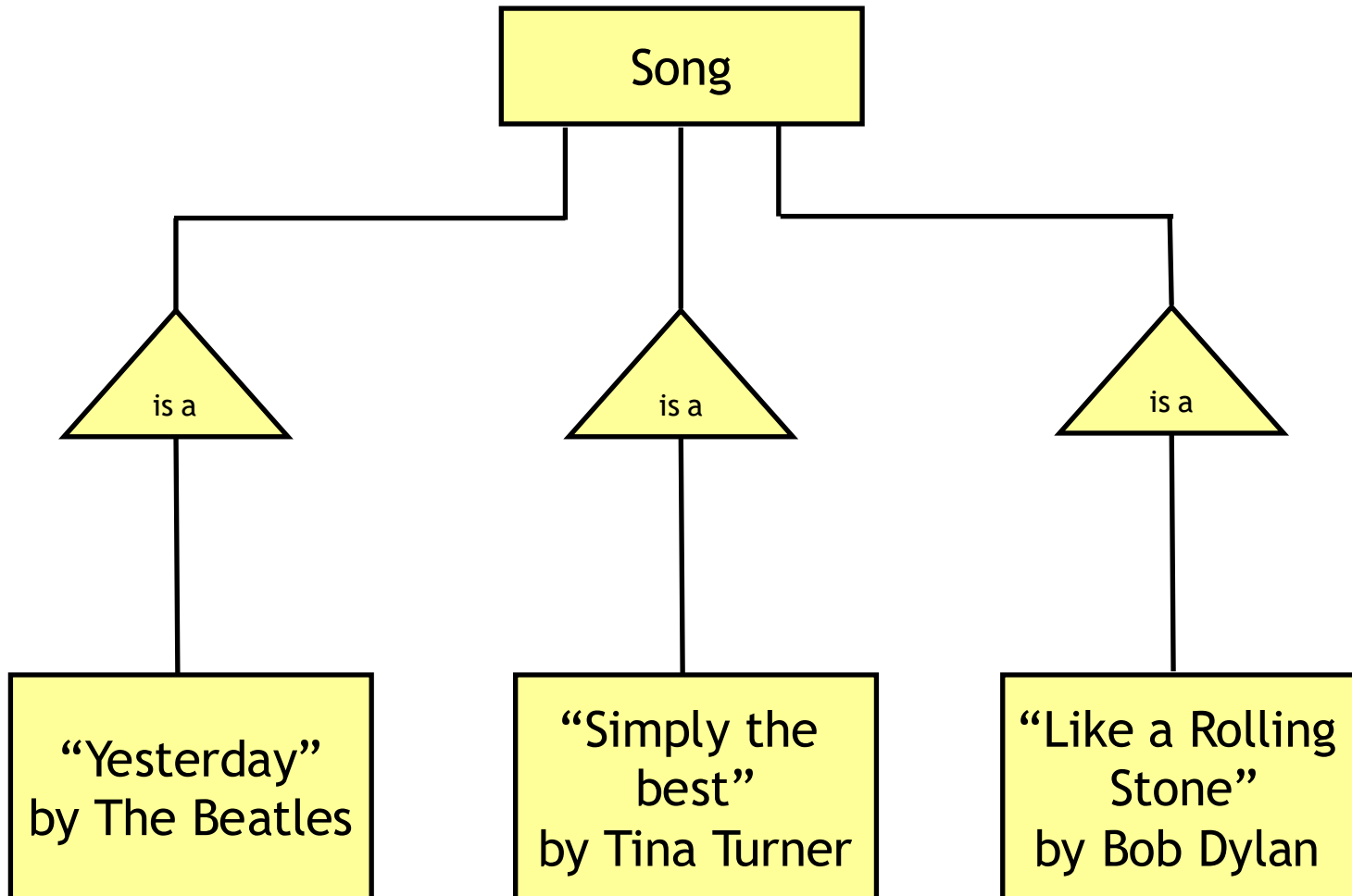
Cardinalities and Intervals (according to Ferstl/Sinz 2001)

- Intervals allow specifying cardinalities more accurately.



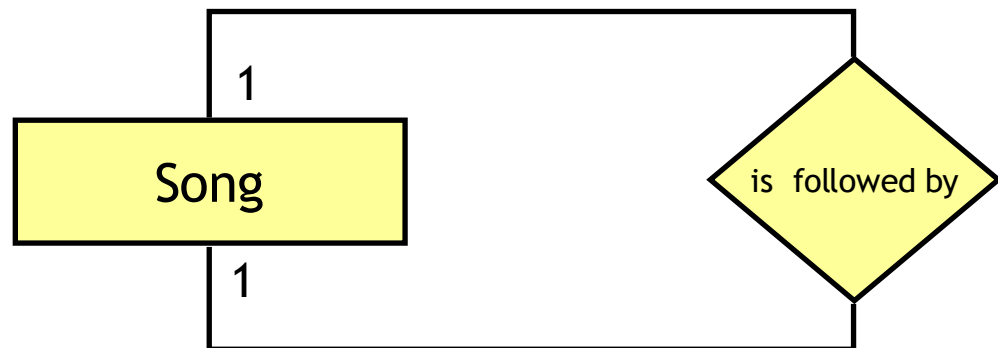
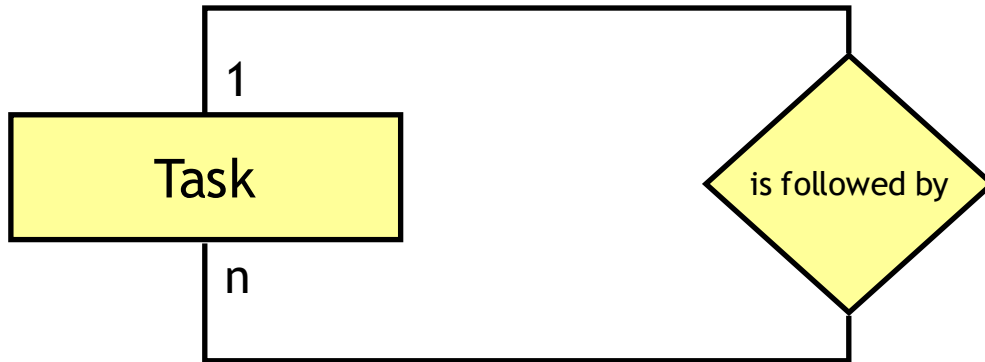
- NOTE:** This lecture follows the widely used approach of Ferstl/Sinz (2001)



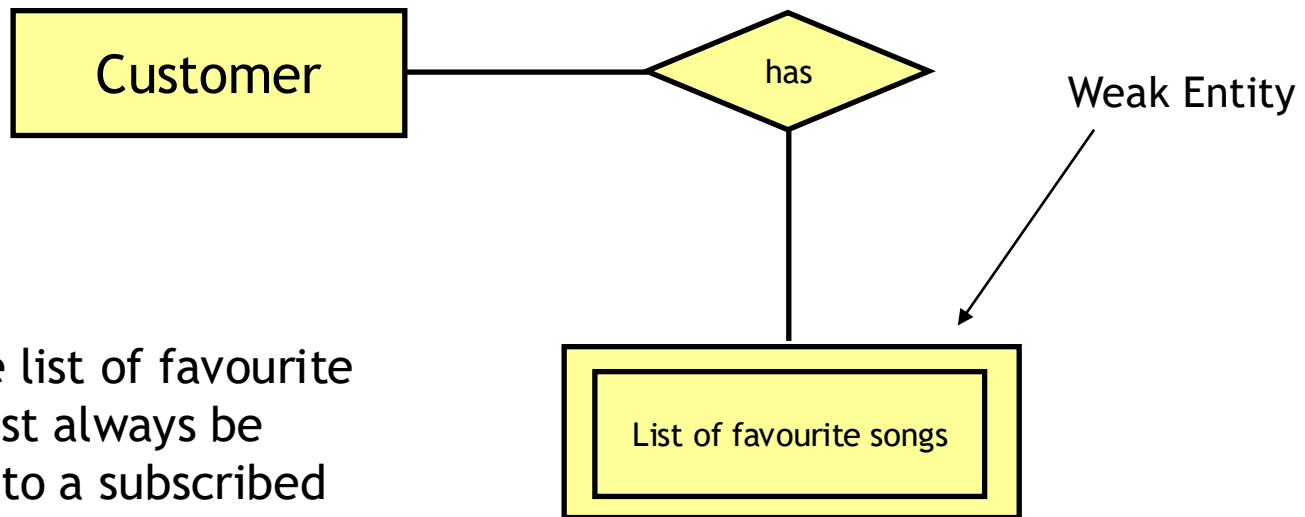


From the Special to the General

Recursive Types of Relationship

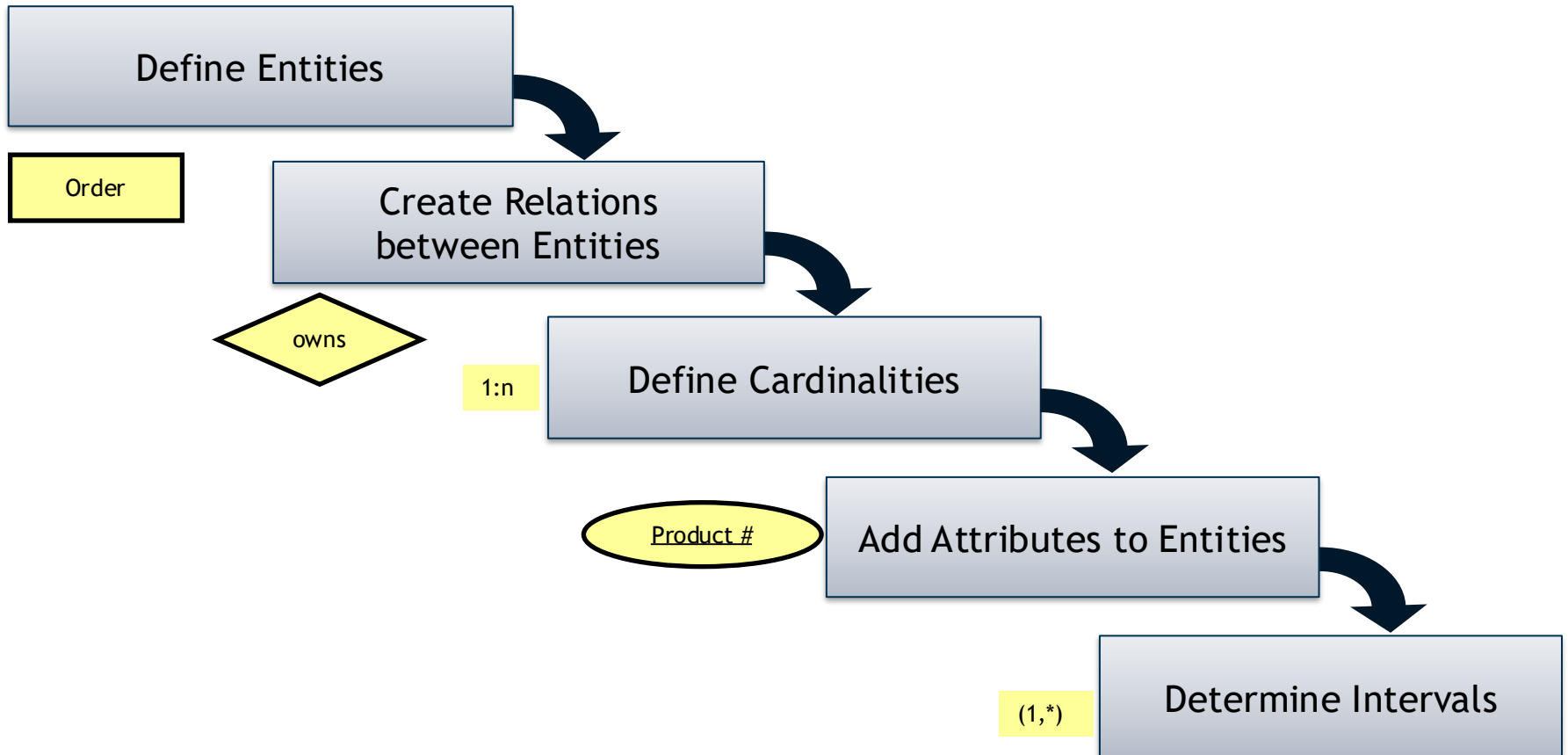


Weak entities depend on at least one entity and consequently cannot exist without them.

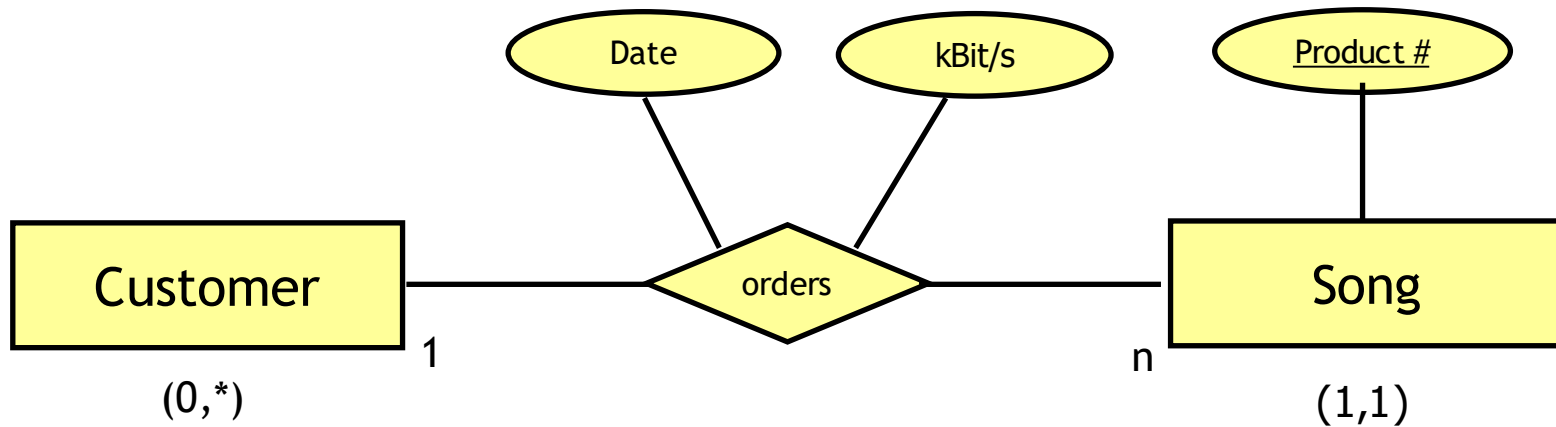


An active list of favourite songs must always be assigned to a subscribed customer.

Process for the Development of an ER-Model/Diagram

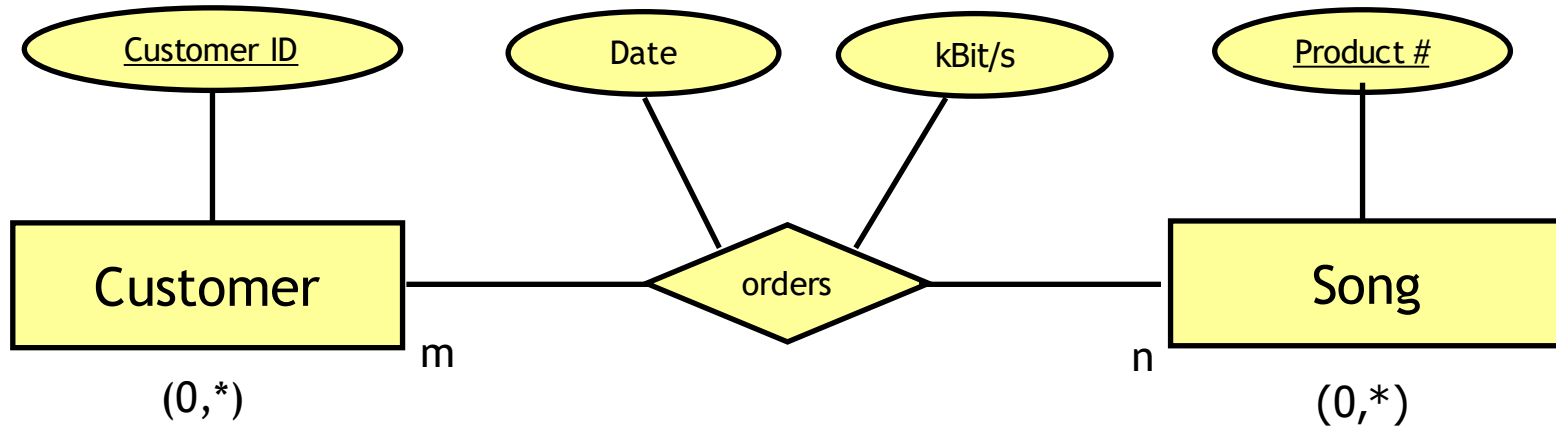


Process for the Development of an ER-Model/Diagram

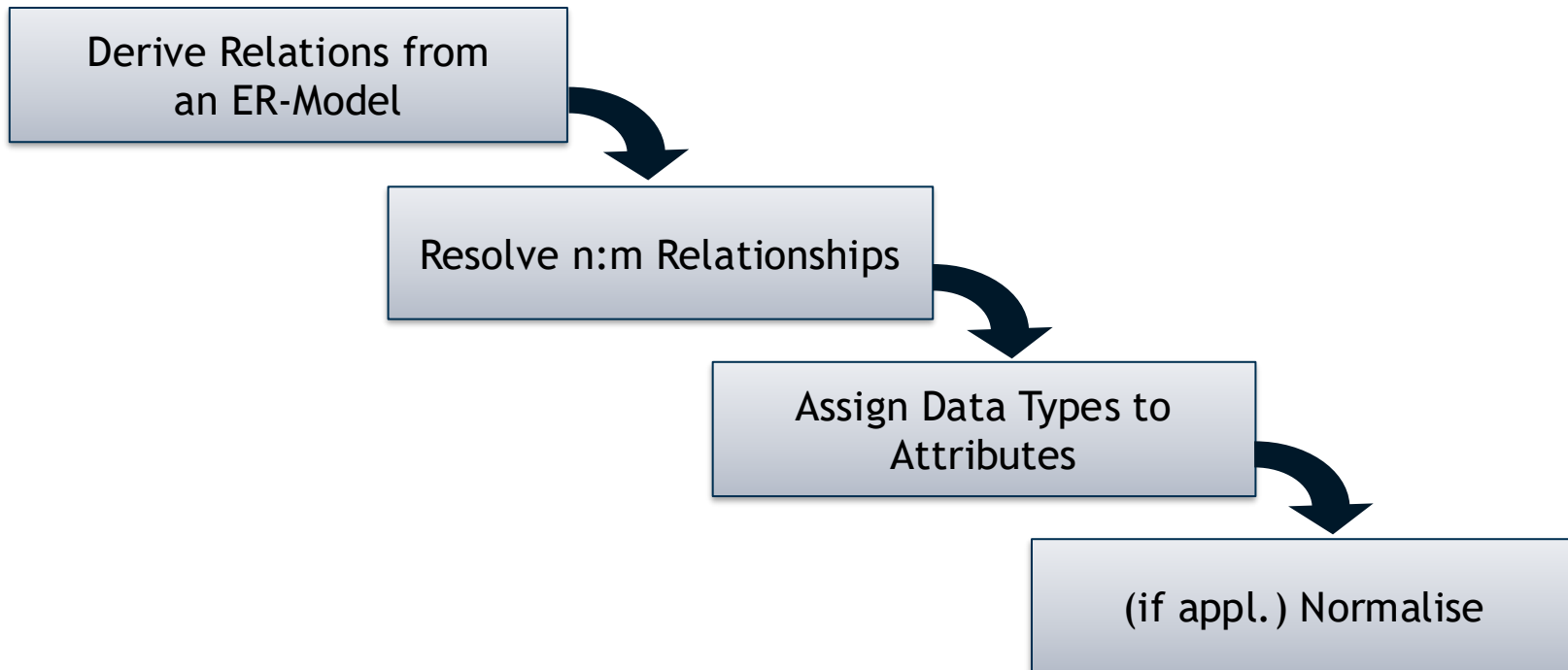


Process for the Development of an ER-Model/Diagram

- New example with m customers



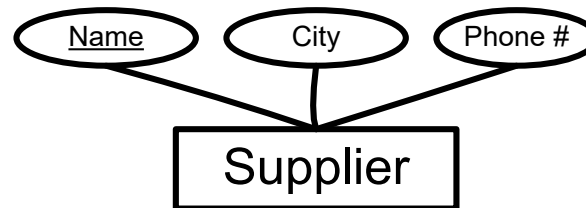
Derivation of Relation Models from an ER-Model/Diagram



Derive Relations from an ER-Model

- The relation type with its corresponding attributes is derived from the entity type.

Example:

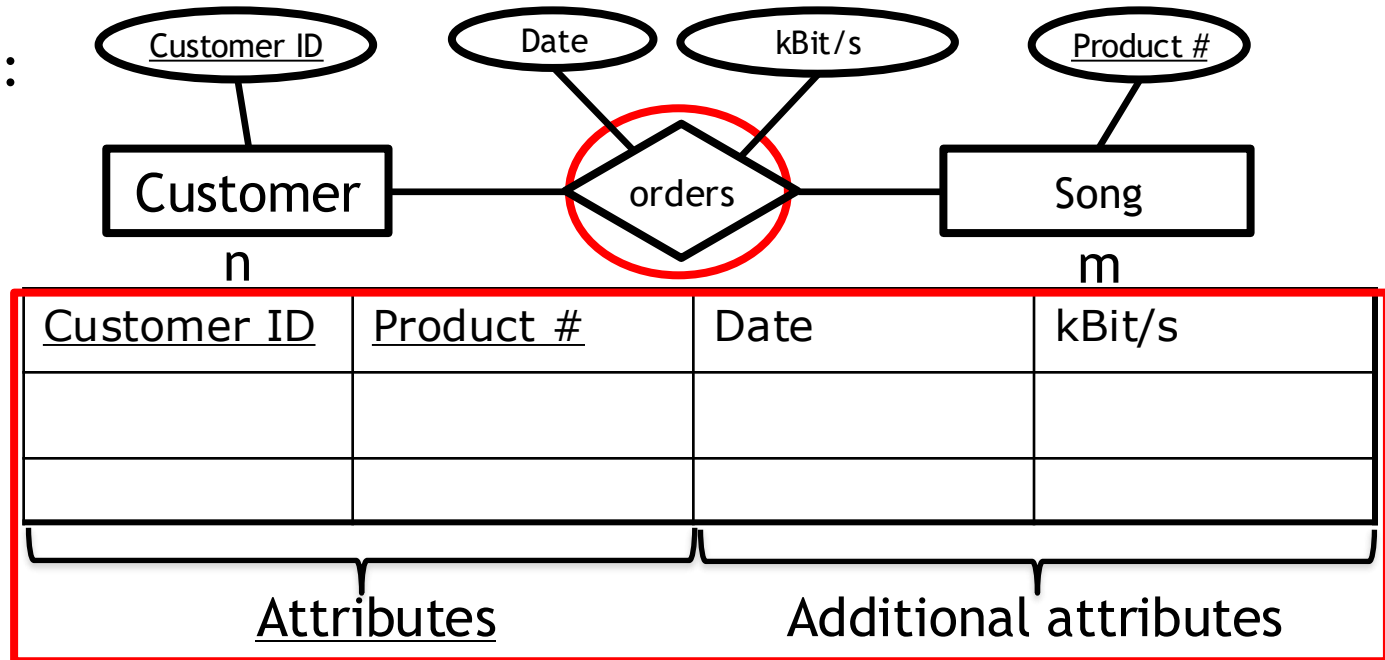


<u>Name</u>	City	Phone#

Derive Relations from an ER-Model

- An n:m-relationship type induces an additional relation-type.
- The relation contains
 - primary keys of involved entity types as attributes
 - and additional attributes of the relation types

Example:

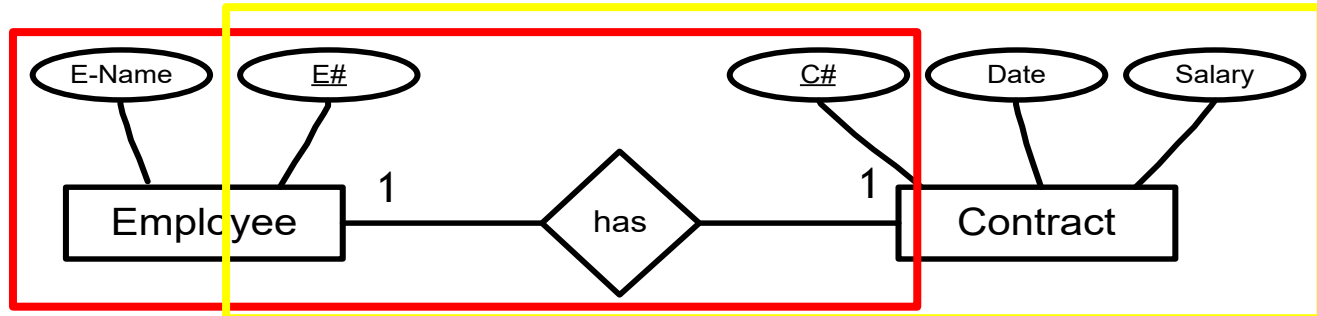


Note: In order to reflect the complete ER-Model above, two more relations (Customer (Customer ID) and Song (Product #)) are required. The relation above connects both Customer and Song entities.

Derive Relations from an ER-Model

- A 1:1 relationship type does NOT become a relation on its own.
- The information is to be “attached” to one of the involved entity types.

Example:



Alternative 1:

<u>E#</u>	E-Name	C#

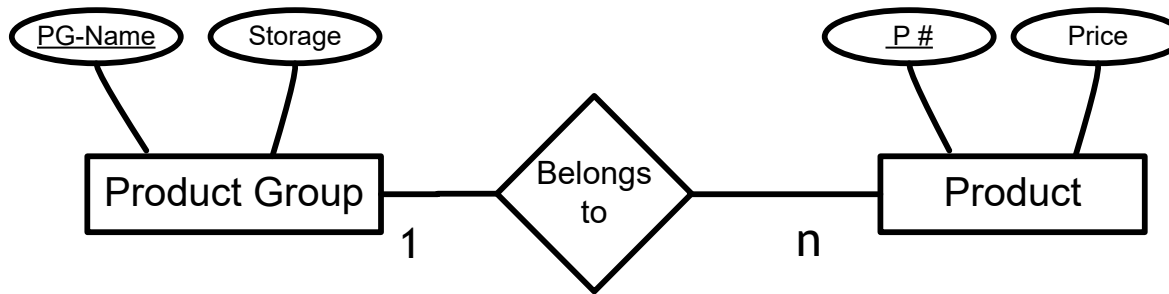
Alternative 2:

<u>C#</u>	Date	Salary	E#

Derive Relations from an ER-Model

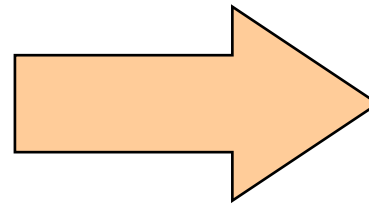
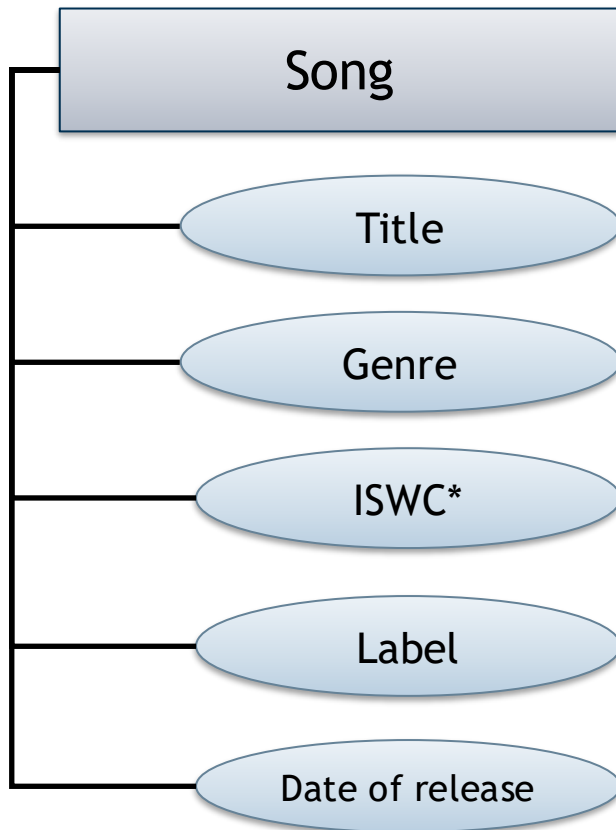
- A 1:n relationship type does NOT become a relation on its own.
- The information is to be “attached” to that relation that corresponds to the entity type with the n-signed edge.

Example:



<u>P #</u>	Price	PG-Name

Assign Data Types to Attributes



Relation „Song“

- Title: CHAR(40)
- Genre: CHAR(20)
- ISWC*: INT
- Label: CHAR(30)
- Date of release: DATE

*International Standard Musical Work Code (9 digits)

- Objectives
 - Each relation contains only data of one meaning.
 - Redundancy free storage of data
 - Prevention of anomalies caused by redundancies and inconsistencies
- Normalisation Forms
 - Non-normalised relations
 - Normalised relations in the 1st - 3rd Normalised Form (NF)
 - Additional Normal Forms (e.g. 4th+5th Normal Form or Boyce Codd)
- Normalisation Process
 - Non-normalised -> 1st NF -> 2nd NF -> 3rd NF
 - Relations derived from ER-Model already in 2nd NF



- Ferstl, Otto K., Sinz, Elmar J. (2001) „Grundlagen der Wirtschaftsinformatik“, 4. Auflage, München
- Laudon, Kenneth C., Laudon, Jane P., Schoder, Detlef (2010) „Wirtschaftsinformatik - Eine Einführung“
- Schwickert, Axel (2004) „Modellierung von IuK-Systemen“, Universität Gießen