MUNI
FI

lasaris LAB OF SOFTWARE ARCHITECTURES AND INFORMATION SYSTEMS

Data Modeling, Entity-Relationship Diagram

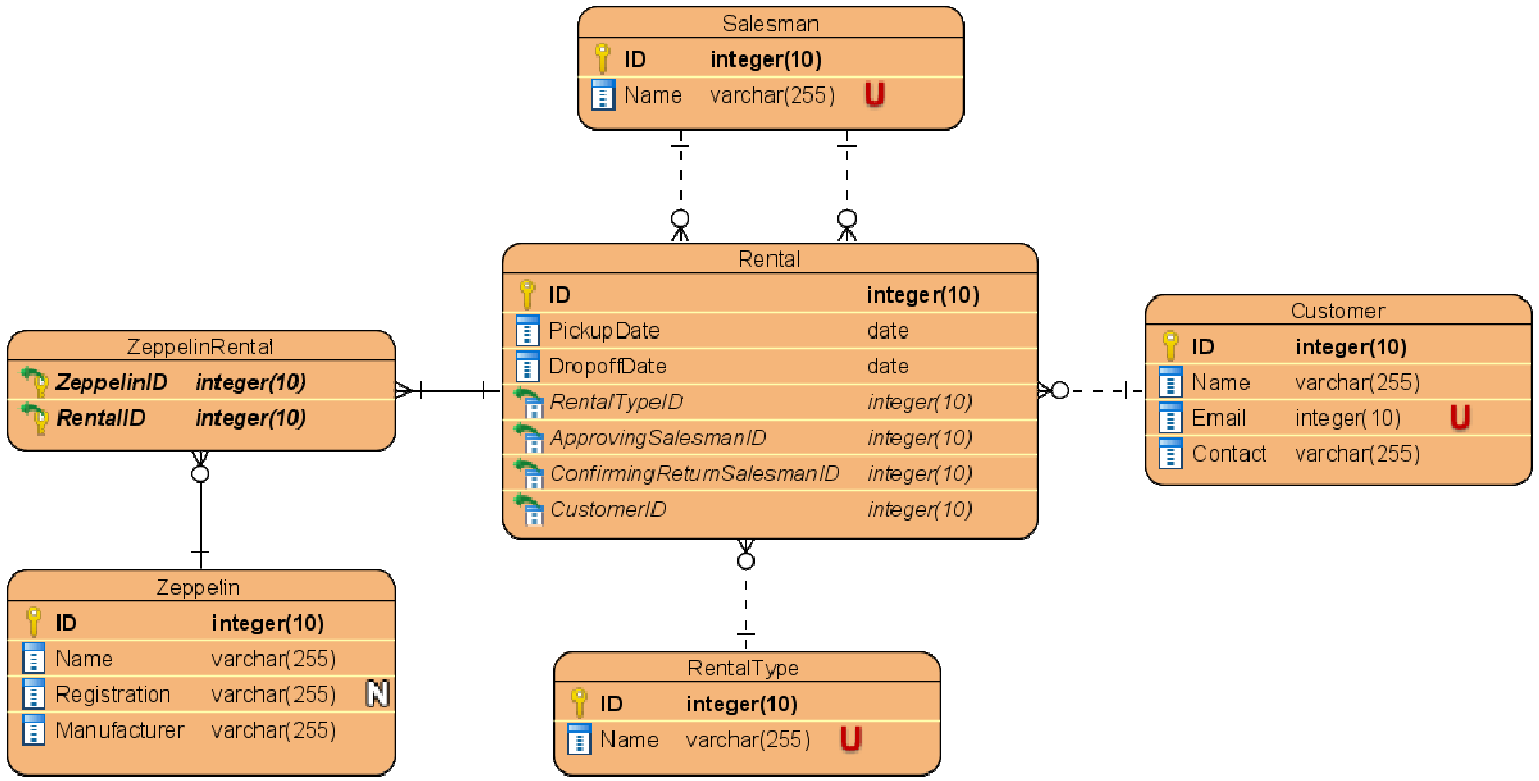
PB007 Software Engineering I

Lukáš Daubner
daubner@mail.muni.cz

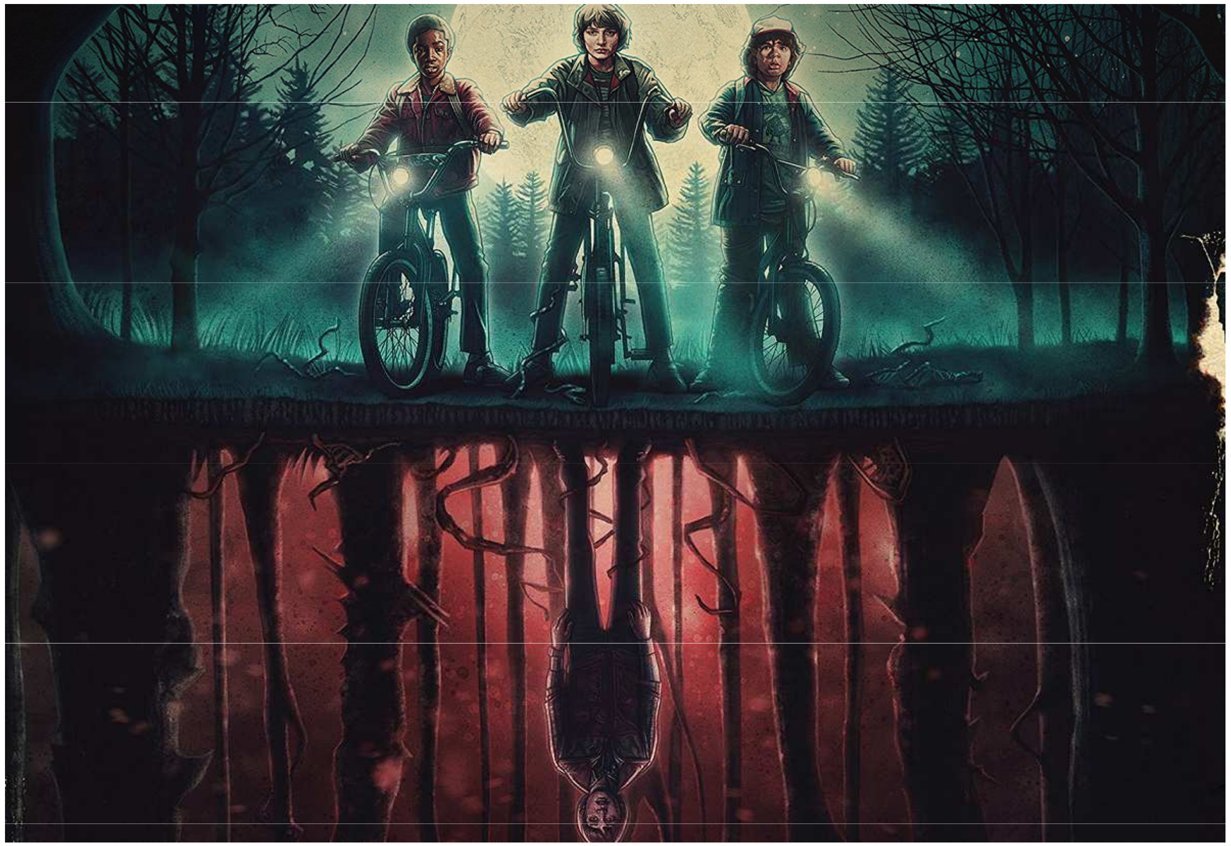
Entity-Relationship Diagram

- Data model
- Not a part of UML
- Representing the logical structure of **relational database**
- Its main components are:
  - Entities
  - Relations
  - Attributes

Entity-Relationship Diagram



Two Worlds Collide



Two Worlds Collide

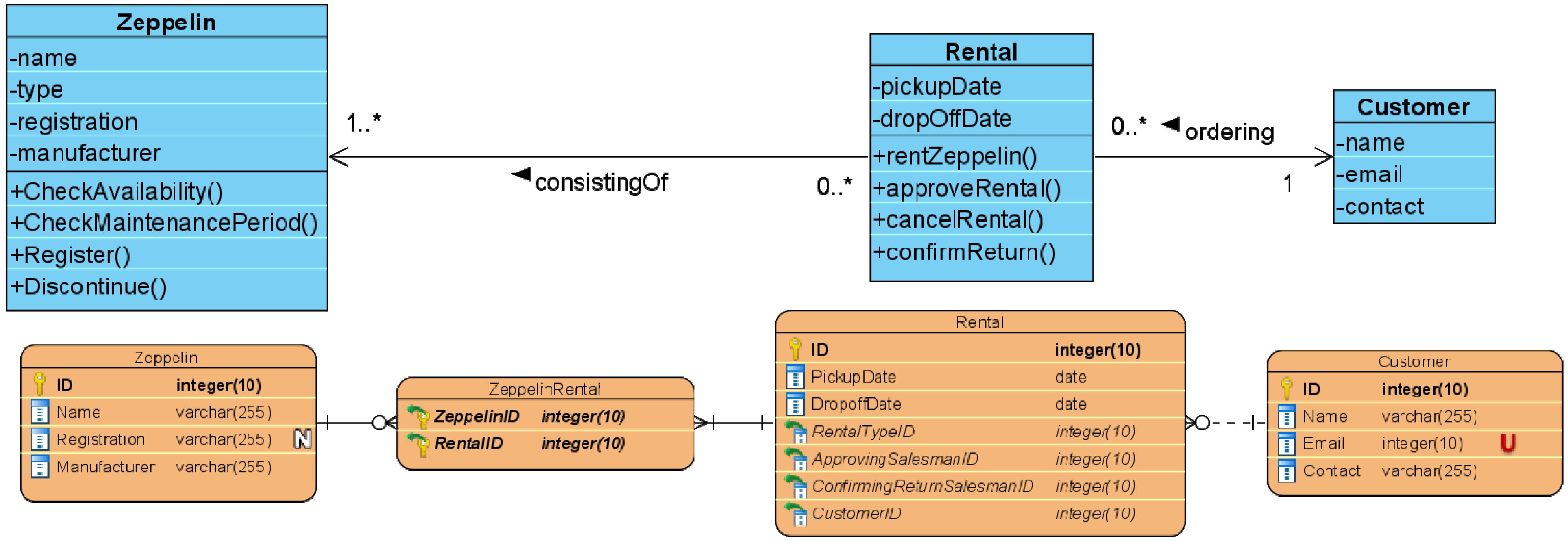
- World of Objects – Class Diagram
  - Captures data and operations
  - Classes are connected with different relationships with different semantics
  - Objects have own dynamic lifecycle
  - Manipulation with data through object interaction
- World of Data – Entity-Relationship Diagram
  - Captures just data
  - Simple relationships
  - Represents tables in relational database
  - Manipulation with data through relational algebra

Object-Relational Mapping

- Conversion „between the worlds“
  - Persistent class ~ Entity type (table)
  - Object ~ Entity (table row)
  - Class attribute ~ Entity attribute (table column)
  - Association/Aggregation/Composition ~ Relation (connection via foreign keys)
  - Inheritance ~ … (manual work needed, see following slides)
- Mapping is not always 1:1!
  - Single class can be mapped to multiple tables
  - And vice versa
  - Not all classes are **persistent** (objects stored in database)

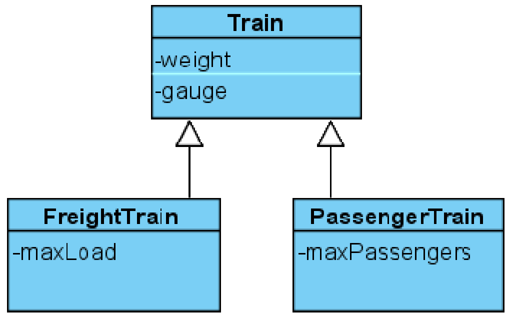
Object-Relational Mapping

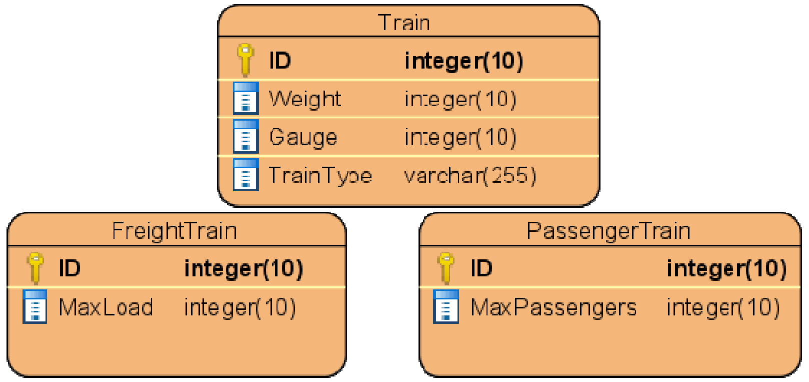
Example



Object-Relational Mapping – Inheritance

1:1 Mapping

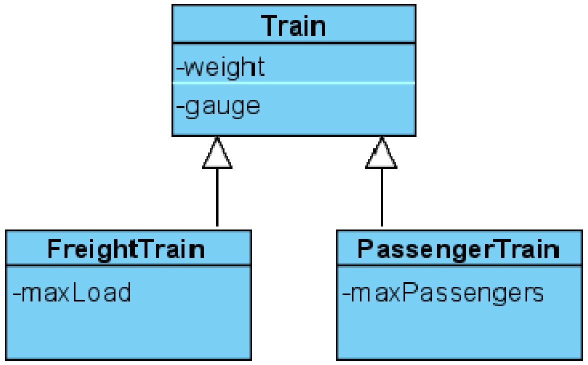


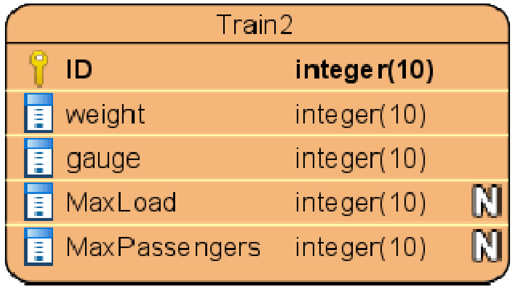


- Each class becomes a table
- Type attribute differentiates the subclass type
- One object instance in multiple tables
  - More difficult data access

Object-Relational Mapping – Inheritance

Merge to superclass





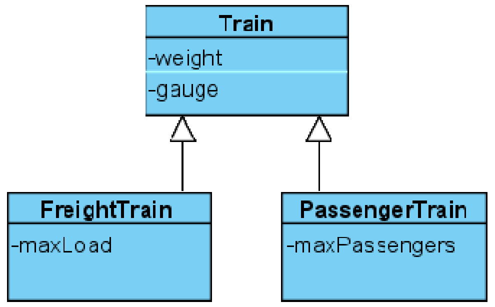
Train2

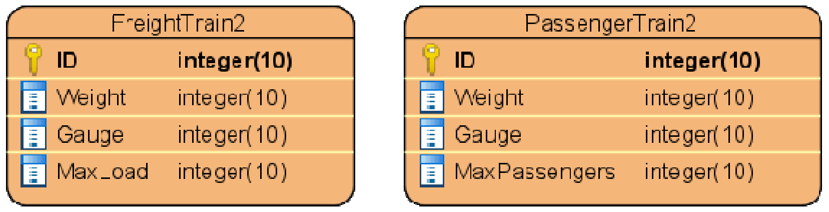
ID	integer(10)	
weight	integer(10)	
gauge	integer(10)	
MaxLoad	integer(10)	N
MaxPassengers	integer(10)	N

- All attributes in one table
- Some will have NULL value
  - Breaks the 4.NF
- Suitable for small number of subclasses and few attributes

Object-Relational Mapping – Inheritance

Propagation to subclasses





FreightTrain2	
ID	**integer(10)**
Weight	integer(10)
Gauge	integer(10)
MaxLoad	integer(10)

PassengerTrain2	
ID	**integer(10)**
Weight	integer(10)
Gauge	integer(10)
MaxPassengers	integer(10)

- Superclass attributes are copied to non-abstract subclass tables
- Suitable if:
  - Superclass has few attributes
  - Many subclasses
  - Subclasses have many attributes

Normal Forms

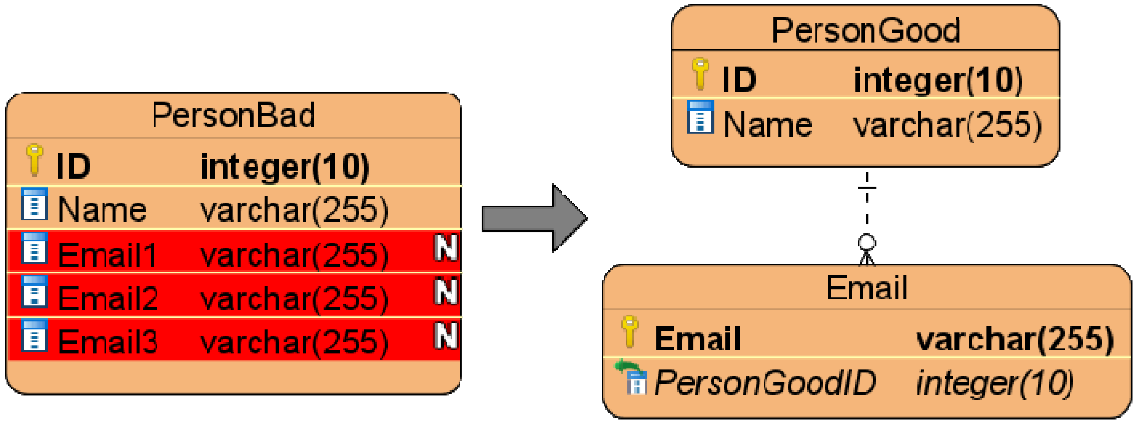
- Technique for data organization and good database design
- Elimination of repetitive data
- Reduction of table complexity
- Problem prevention
  - E.g., update anomalies

Normal Forms

1. Normal Form

- Satisfies 0. NF (yes, it actually exists)

- Each attribute is atomic

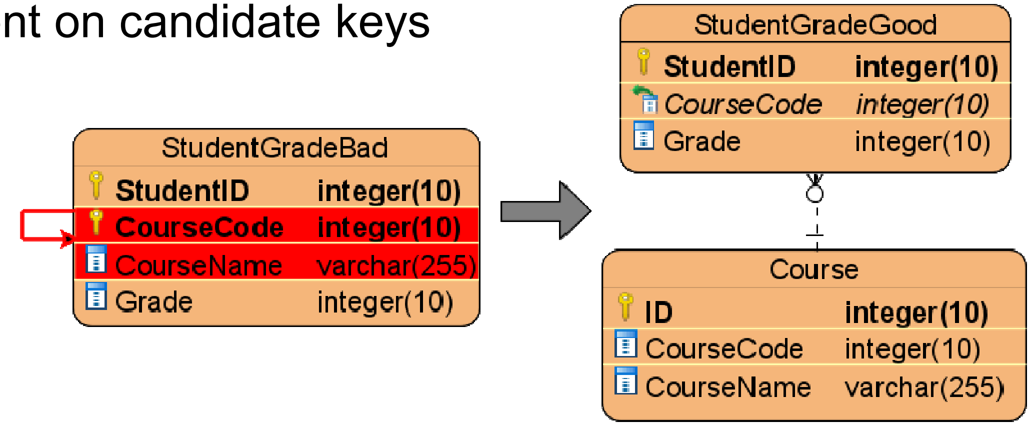


Normal Forms

2. Normal Form

- Satisfies 1. NF

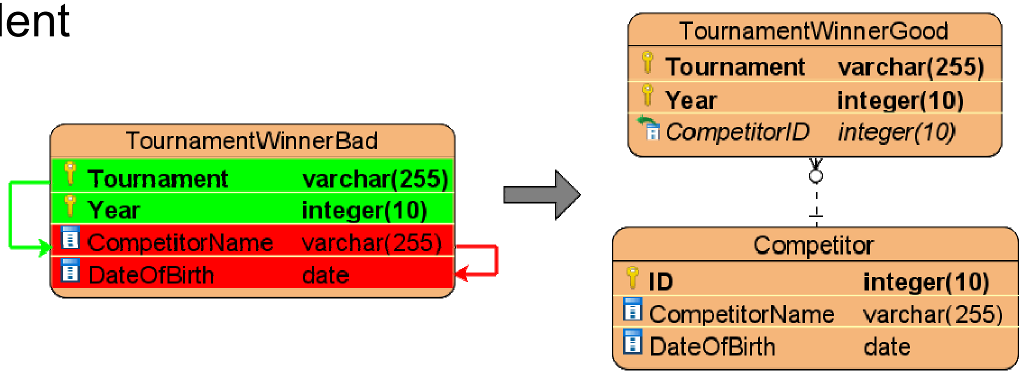
- No partial dependency
  - Each non-key attribute are fully dependent on candidate keys



Normal Forms

3. Normal Form

- Satisfies 2. NF
- No transitive dependency
  - Each non-key attribute is dependent on primary key (and candidate keys) only
  - Non-key attributes are mutually independent



Task for this week

You gotta do what you gotta do

- Process the feedback

- Create ERD based on the class diagram
  - Keep it consistent – you model the same system
  - Decompose M:N relationships using entities
  - Normalize to 3. NF

- Based on the EDR create separate example violating 3. and 2. NF
  - Example – don‘t need to create whole new diagram
  - Add notes explaining the NF violations