Transformation vom objektorientierten ins relationale Modell

Illustrationen von Paterson, Edlich, Hörnick

N. Höller

Institut für Informationssysteme
Universität zu Lübec

28. November 2007

UNI LÜBECK

Einleitung

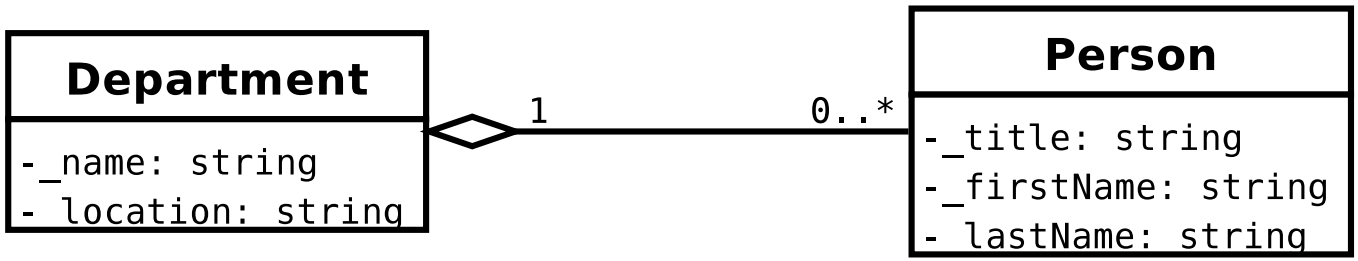
- Eine einfache Klasse kann meist direkt als Tabelle aufgefasst werden.
- Dabei werden allg. Klassenvariablen als Attribute der generierten Tabelle aufgefasst.
- Eine einfach Klasse entspricht demnach einer Entität.
- Die Objektidentität muss jedoch über einen Primärschlüssel abgebildet werden!!
- Bei komplexen Klassenbeziehungen müssen Sonderfälle betrachtet werden.
- Dies gilt vor allem für Aggregationen, Vererbung und Many-Many Beziehungen.

nochmal zur Objektidentität...

- In Relationen werden Primärschlüssel verwendet, um eindeutige Tabelleneinträge zu generieren.
- Im objektorientierten Modell sind Objekte bereits durch ihre Identität eindeutig.
- Zwei gleiche Objekte sind damit nicht identisch!
- Objekte müssen daher durch zusätzliche Schlüsselattribute auf Relationen gemapped werden.

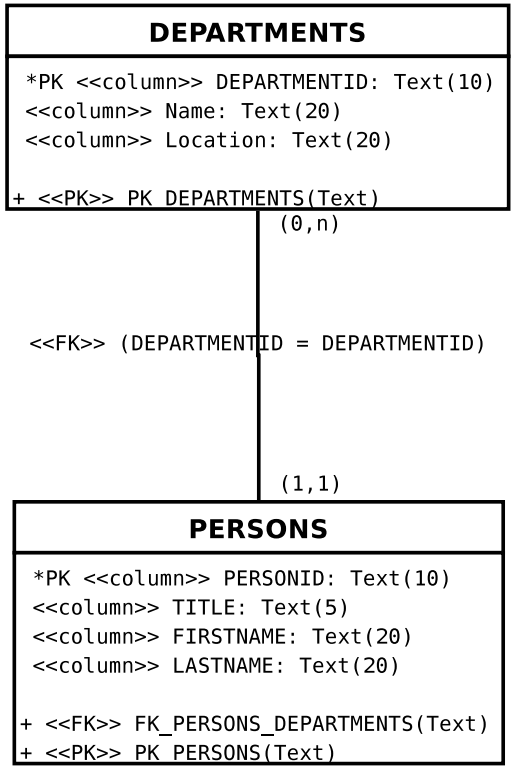
Aggregation

- Eine Aggregation ist eine Teil/Ganze-Beziehung.
- Assoziationen werden gleichermaßen gemapped, da das relationale Modell nicht zwischen Teil/Ganze-Beziehung und einfacher Referenz unterscheidet.



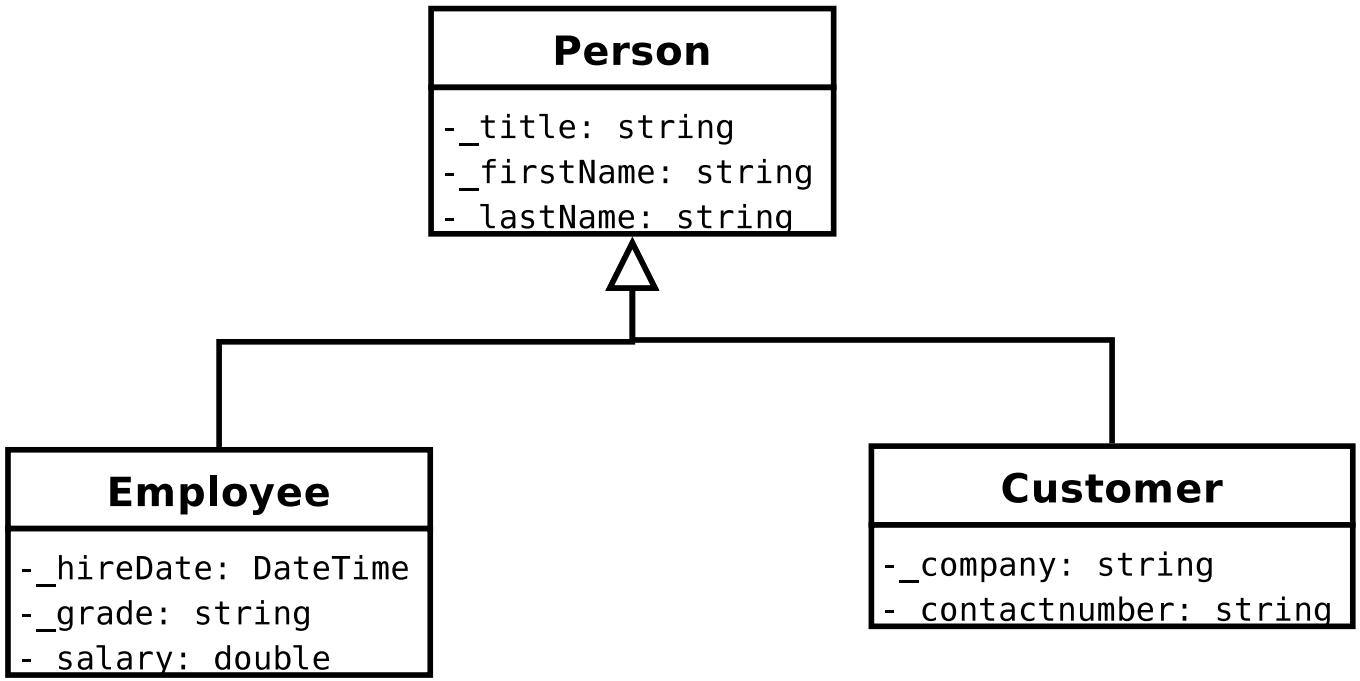
Aggregation Mapping

- Mapping über Fremdschlüssel in der Tabelle auf der Ganzes-Seite der Beziehung.



Vererbung

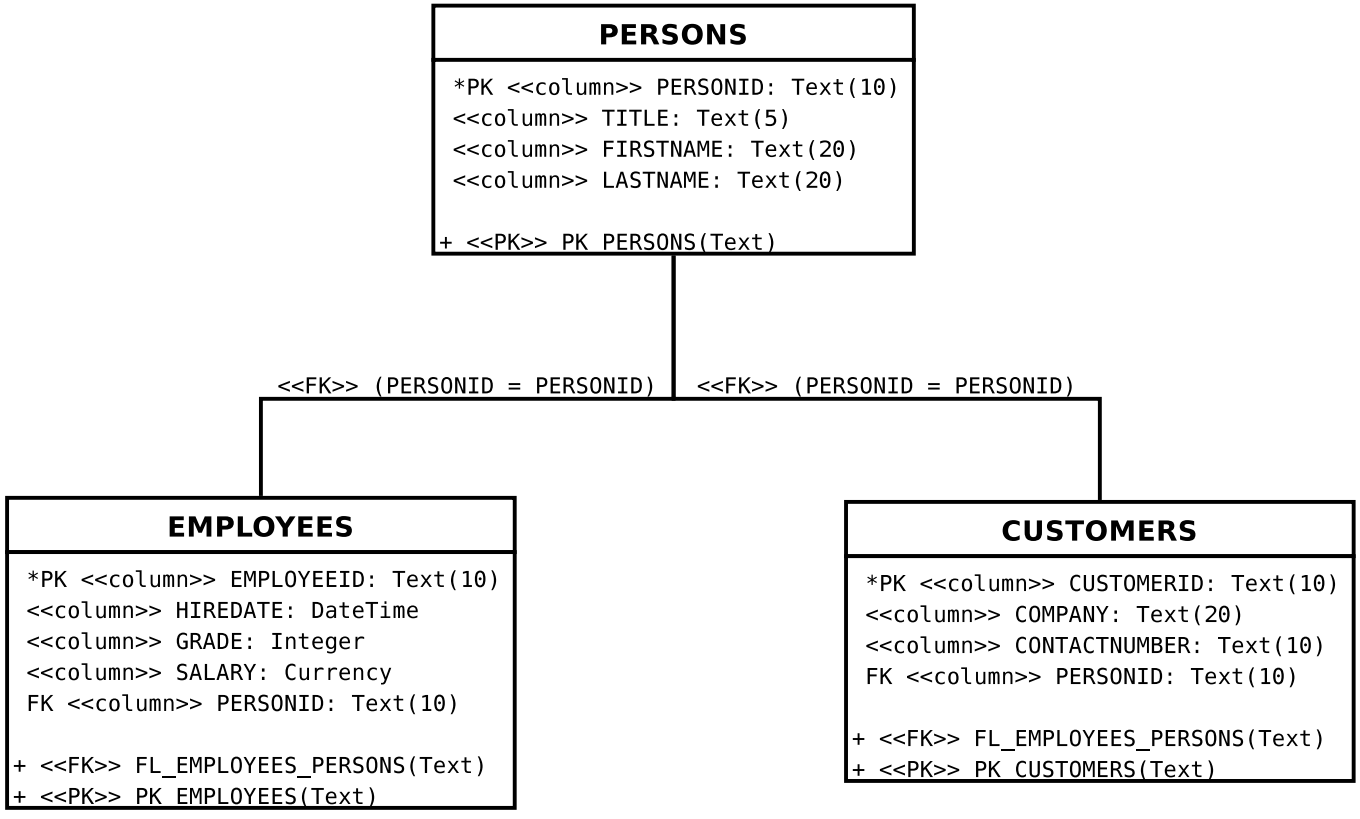
- Im relationalen Modell gibt es kein äquivalentes Konzept zur objektorientierten Vererbung.
- Es gibt mehrere Möglichkeiten eine Vererbung aufzulösen.
- Die jeweilige Strategie hängt vom Vererbungsbaum ab.



Vertical Mapping: One Table per Class

- Es wird eine Tabelle für jede Klasse und Unterklasse erstellt.
- Unterklassen werden mit Fremdschlüsseln mit der Oberklasse in Beziehung gebracht.
- Es müssen jeweils Schlüssel für jede Ausprägung erstellt werden (PersonID).
- Um ein `Employee` Objekt zu erstellen, müssen die Tabellen `Person` und `Employee` gejoined werden.
- Komplizierter Vorgang, aber gute Wartungsmöglichkeiten (Robust gegen Änderungen).

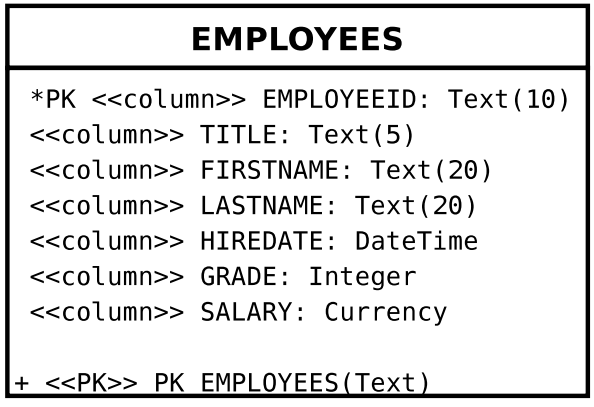
Vertical Mapping: One Table per Class

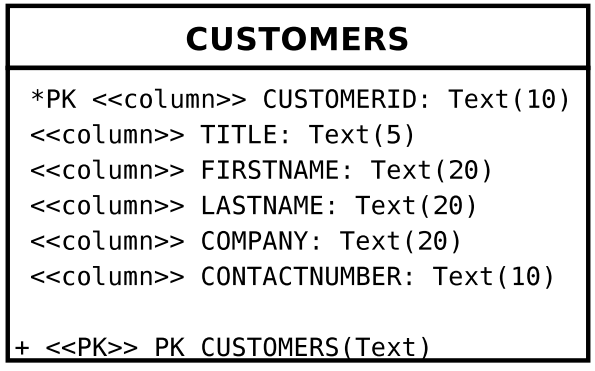


Horizontal Mapping: One Table per Concrete Class

- Es wird eine Tabelle für jede konkrete Klasse erstellt.
- Hierzu wird der Vererbungsbaum aufgelöst und jede mögliche Klassenausbildung betrachtet.
- Es ist eine einfache Möglichkeit eine Vererbung aufzulösen.
- Probleme
  - Redundanz
  - Wenn Änderungen in Oberklassen auftreten müssen alle Tabellen geändert werden.
  - Viel Arbeit bei Änderungen des Schemas.

Horizontal Mapping: One Table per Concrete Class

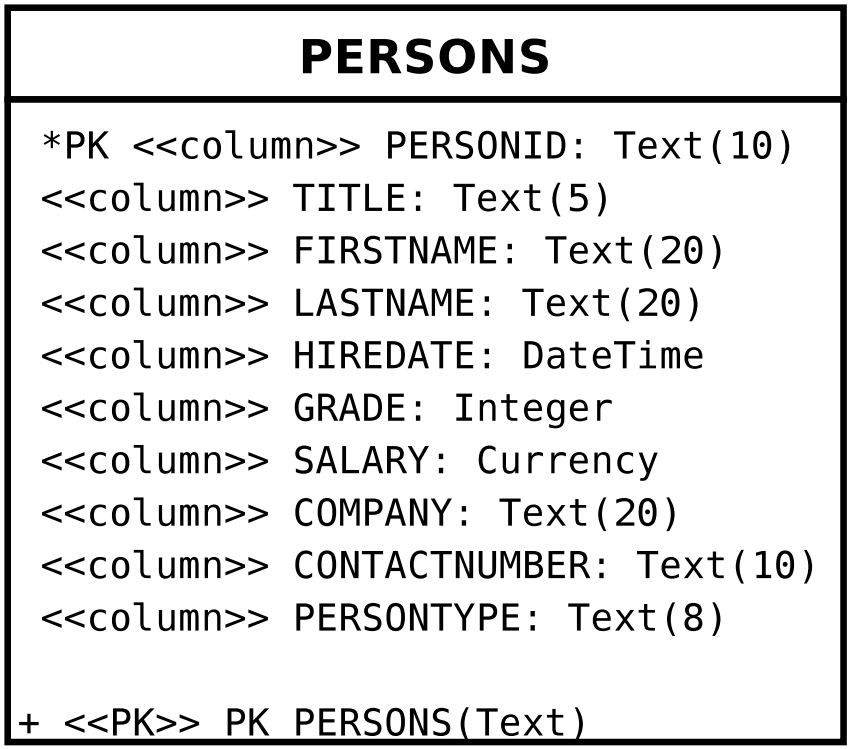




Filtered Mapping: One Table per Tree

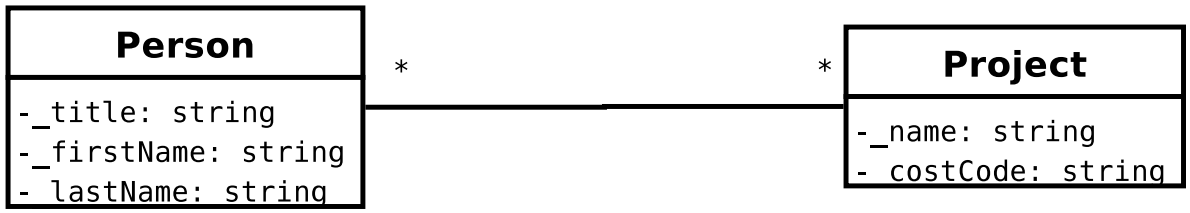
- Joins und separate Tabellen werden umgangen.
- Ein Filterattribut (`PERSONTYPE`) sorgt dafür, zwischen unterschiedlichen Ausprägungen unterscheiden zu können.
- Vorteil: einfacherer Zugriff, Objektbaum = eine Entität
- Nachteil:
  - große Relation.
  - für Unterklassen werden unnötige Attribute mitverwendet (meist NULL).
  - Typsicherheit muss genauer betrachtet werden.

Filtered Mapping: One Table per Tree



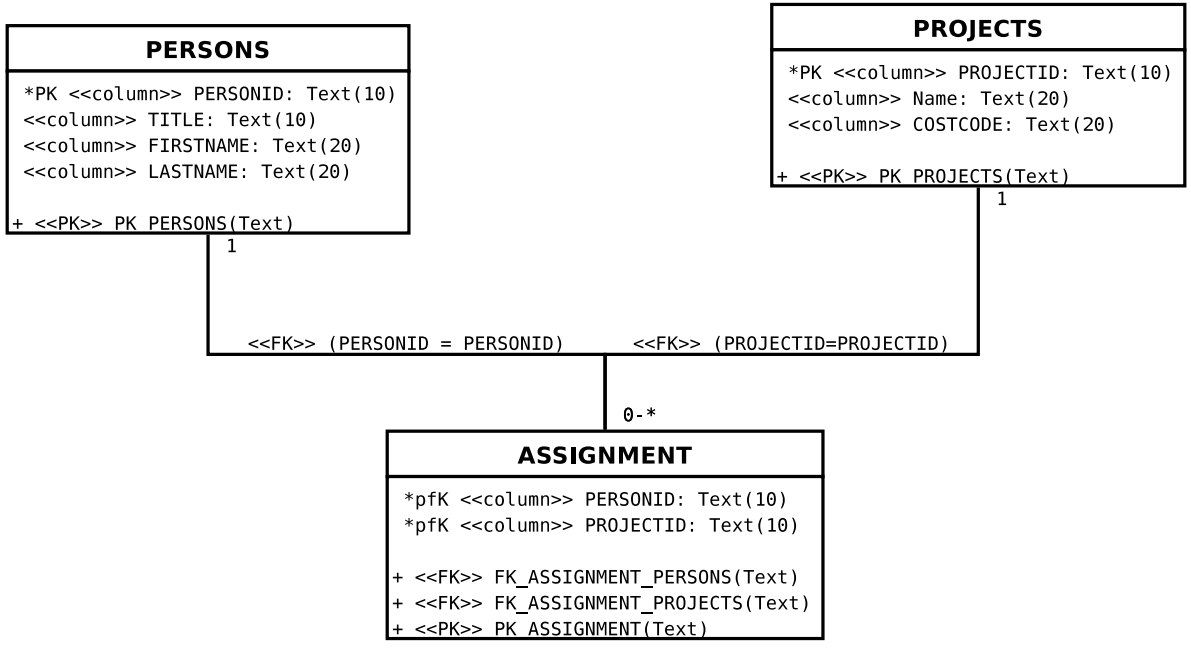
Many-Many Beziehungen

- Ein Beispiel für eine Many-Many Beziehung
- Eine Person kann an mehreren Projekten arbeiten, während an einem Projekt beliebig viele Personen arbeiten



Many-Many Beziehungen

- Wie bei der Umsetzung im ER Modell muss eine separate Tabelle als Beziehungstabelle fungieren.
- Dieses Mapping existiert daher schon länger als das objektorientierte Modell.



UNI LÜBECK