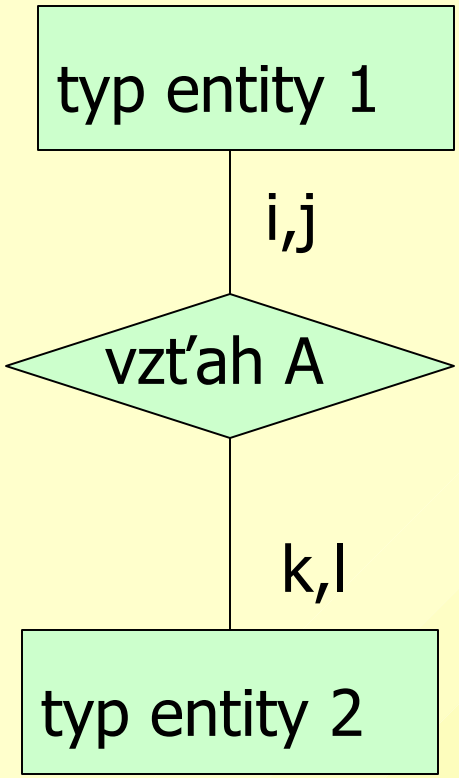
Poloformálne metódy - mapovanie reality

- Entitno-relačný model
- Binárny model
- NIAM
- Sémantický model
- O – O model
- UML – diagram tried
- Sieťový model - Bachmanové diagramy
- Automatické navrhovadlá (designer2000, access, studio, …)
- HIT

Grafická reprezentácia vizualizácia

Entitno relačný (ERA) model



$i, k = \begin{cases} 0 \\ 1 \end{cases}$

0 — Entita daného typu sa nemusí vyskytovať vo vzťahu A

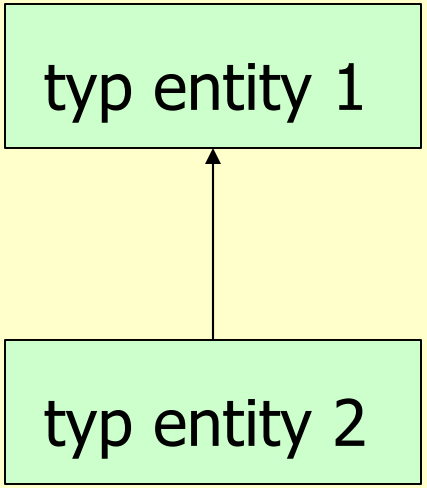
1 — Každá entita daného typu sa musí vyskytnúť vo vzťahu A

$j, l = \begin{cases} 1 \\ n \end{cases}$

1 — Entita daného typu sa môže vyskytovať vo vzťahu A najviac raz

n — Bez ohraničení na počet výskytov entity daného typu vo vzťahu A

Vzťah generalizácie - is a



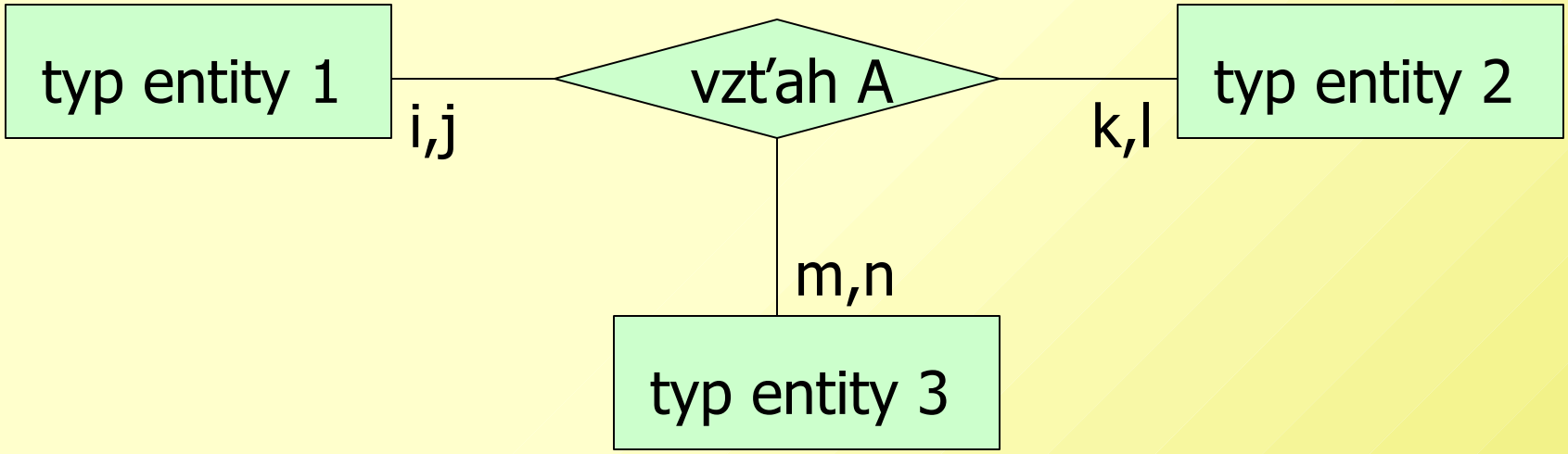
Typ entity 2 je špeciálnym prípadom typu entity 1.

- dedenie atribútov
- discriminated union
- nulové hodnoty

Atribúty - vpisujú sa do typov entít

označenie kľúčov (a cudzích kľúčov)

Ternárne a n-árne vzťahy



Problém ohraničení počtu výskytov

- objektifikácia binárneho vzťahu
- určenie funkčnej závislosti

Binárny model - NIAM

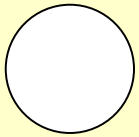
- slovný popis
- grafická reprezentácia

Pojmy:
- Typ
- Populácia
- Výskyt (occurrence)

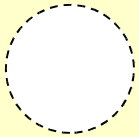
Lexikálne (LOT) a nelexikálne typy objektov (NOLOT)

Typy vzťahov:
- *Idea* - medzi nelexikálnymi typmi objektov
- *Bridge* - medzi nelexikálným a lexikálnym objektom
- Phrase - medzi lexikálnymi objektami

Grafická notácia

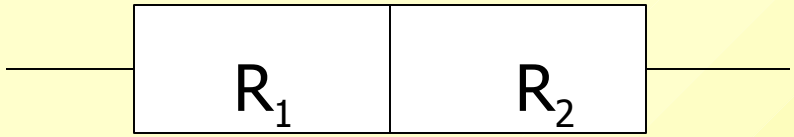


Nelexikálny typ objektu



Lexikálny typ objektu

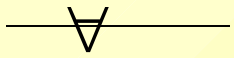
Podtyp (is a)



Idea alebo bridge

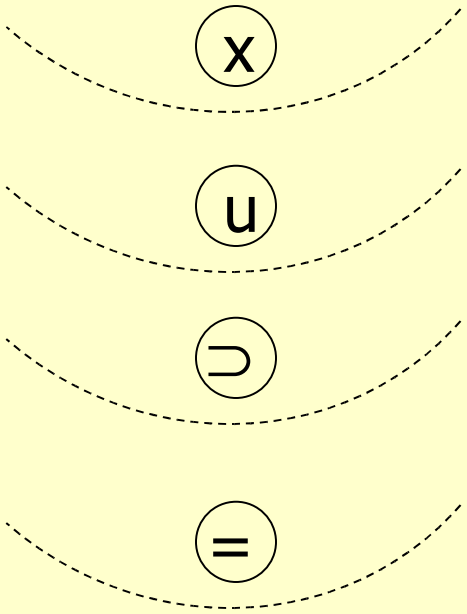
Podmienky - constraints

Nad menom role, znamená že táto rola jednoznačne určuje druhú rolu vo vťahu



surjekcia (totalita)

Podmienky - constraints



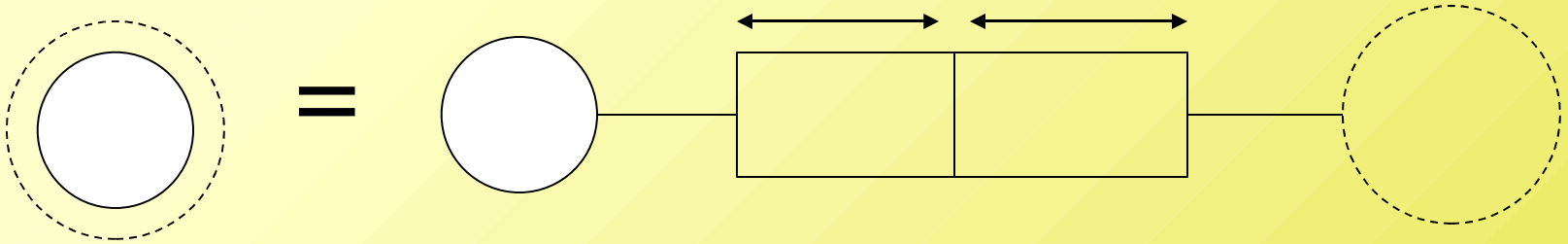
Disjunktnosť (vylúčenie) medzi podtypmi

Jednoznačné určenie výskytu (kombinácie)

Inklúzia medzi populáciami rolí

Rovnosť populácii rolí

Makro



O – O model

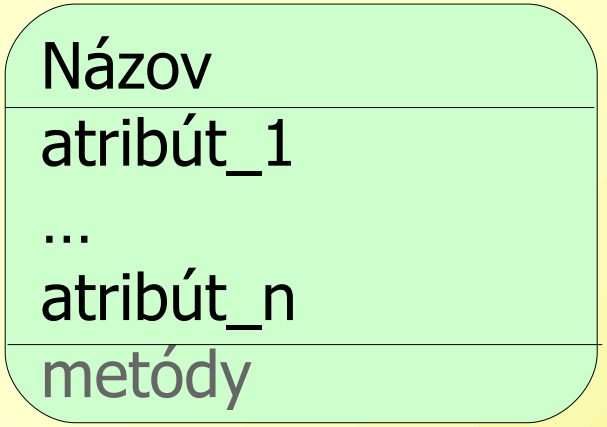
class: Študenti (
attribute string meno;
attribute integer rodné_číslo;
attribute Struct(deň, mesiac, rok) Dátum_narodenia;
association Set(Prednášky) zapísal_si
inverse Prednášky :: majú_zapísané)

V oblasti návrhu objektový model zodpovedá UML diagramu tried. Je podobný binárnemu modelu.

Navyše je detailnejší umožňuje podrobne popísať typy atribútov. Používa konštruktory typov (Set - množina, bag – multimnožina, struct – record, list –zoznam, array – pole, ...).

UML – diagram tried

Class = entity set



SQL:
Create table Názov (atribút_1 domain_1,
… ,
atribút_n domain_n);

Je dobrou paxou nepoužívať SQL dátové typy ako domény atribútov v príkaze create table, ale najprv si vytvoriť vlastné dátové typy príkazom **Create domain !** (SQL 99)

SQL – create domain

CREATE DOMAIN *name* [**AS**] *data_type*
[**DEFAULT** *expression*]
[*constraint* [...]]
where *constraint* is:
[**CONSTRAINT** *constraint_name*]
{ NOT NULL | NULL | **CHECK** (*expression*) }

Syntax SQL – create table

CREATE [[**GLOBAL** | **LOCAL**] { **TEMPORARY** | **TEMP** }]
TABLE *table_name* (
{ *column_name data_type* [**DEFAULT** *default_expr*]
[*column_constraint* [...]]
| *table_constraint*
| **LIKE** *parent_table* [{ **INCLUDING** | **EXCLUDING** }
DEFAULTS] } [, ...]
)
[**INHERITS** (*parent_table* [, ...])]
[**WITH** OIDS | **WITHOUT** OIDS]
[**ON COMMIT** { **PRESERVE ROWS** | **DELETE ROWS** | **DROP** }]

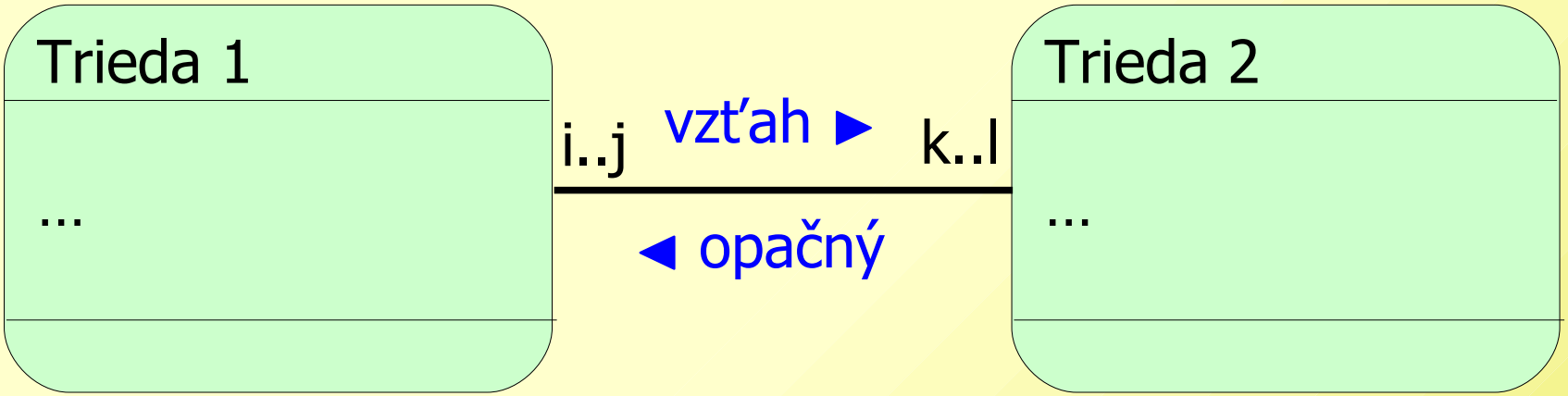
OIDS je synonýmum pre surrogate primary key.

Syntax SQL – podmienky na stĺpce

[**CONSTRAINT** *constraint_name*]
{ NOT NULL | NULL | **UNIQUE** | **PRIMARY KEY** |
CHECK (*expression*) |
REFERENCES *reftable* [(*refcolumn*)] [MATCH FULL |
MATCH PARTIAL | MATCH SIMPLE]
[**ON DELETE** *action*] [**ON UPDATE** *action*] }
[**DEFERRABLE** | **NOT DEFERRABLE**] [**INITIALLY DEFERRED** | **INITIALLY IMMEDIATE**]

action :: =
NO ACTION | **SET** NULL | **SET** DEFAULT | **CASCADE**

Vzťahy – associations



$i,k \in N$

$j,l \in N - \{0\} \cup \{\infty\}$

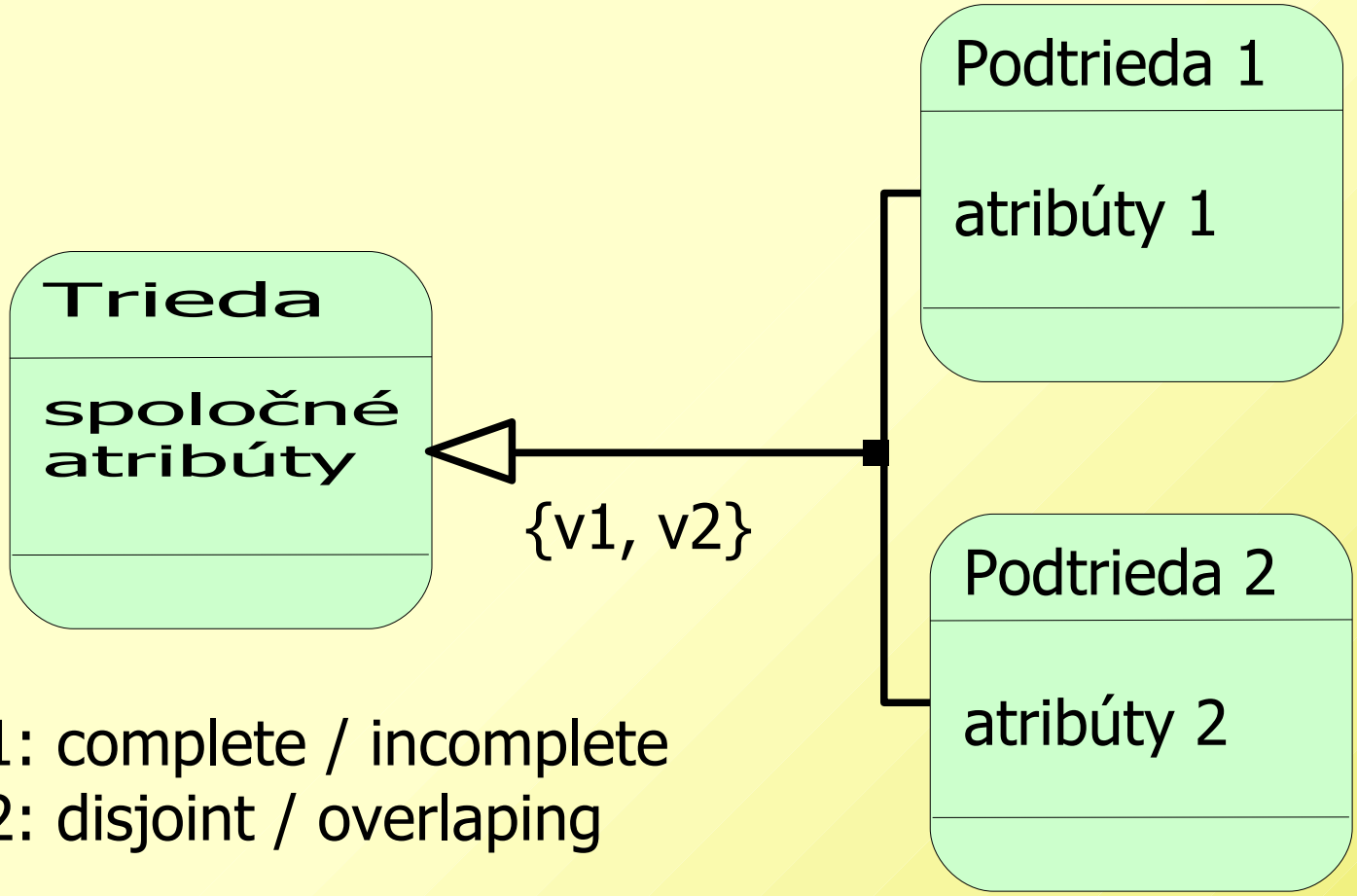
Ak na jednej strane sú hodnoty 0..1 ide o funkčnú závislosť (medzi kľúčami).
1..x znamená inklúznu závislosť.
1..1 je vhodné myslieť na referenčnú integritu.

Syntax SQL – podmienky na tabuľky

[**CONSTRAINT** *constraint_name*]
{ **UNIQUE** (*column_name* [, ...]) |
PRIMARY KEY (*column_name* [, ...]) |
CHECK (*expression*) |
FOREIGN KEY (*column_name* [, ...]) **REFERENCES** *reftable* [(*refcolumn* [, ...])]
[MATCH FULL | MATCH PARTIAL | MATCH SIMPLE]
[**ON DELETE** *action*] [**ON UPDATE** *action*
[**DEFERRABLE** | **NOT DEFERRABLE**] [**INITIALLY DEFERRED** | **INITIALLY IMMEDIATE**]

action :: =
NO ACTION | **SET** NULL | **SET** DEFAULT | **CASCADE**

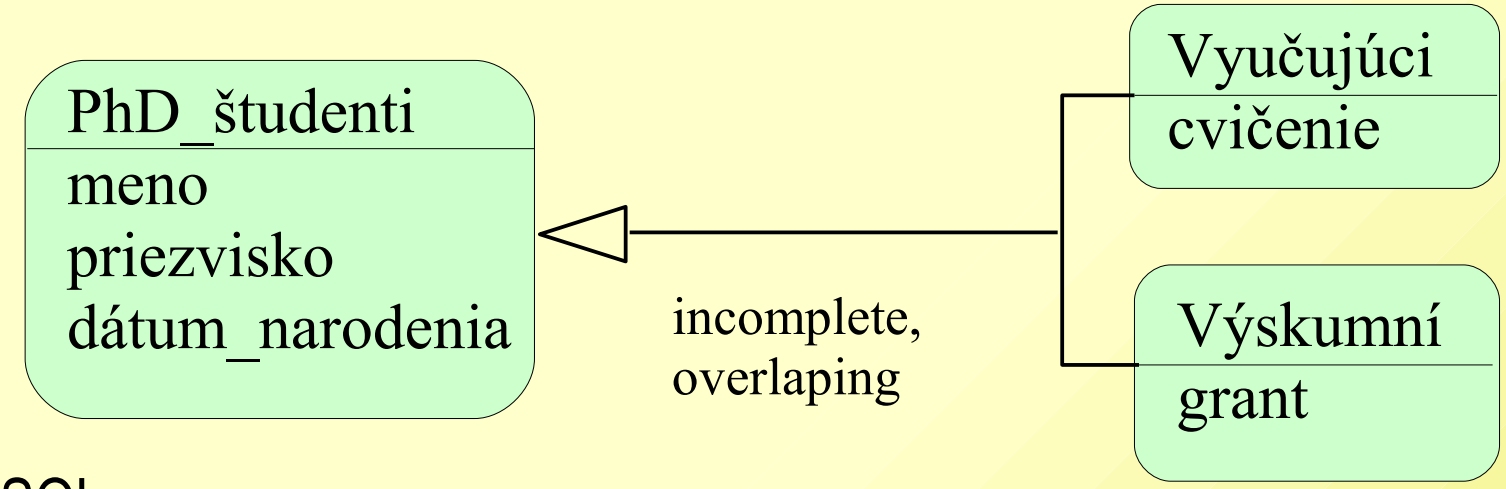
Podtriedy – špecializácia, generalizácia



v1: complete / incomplete
v2: disjoint / overlaping

Samostatné tabuľky, s rovnakým primárnym kľúčom

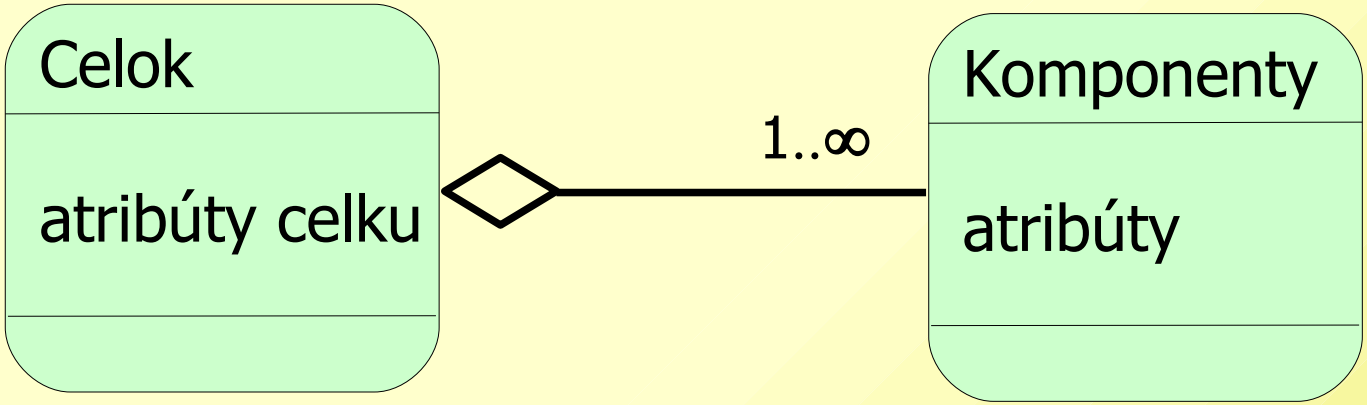
Príklad

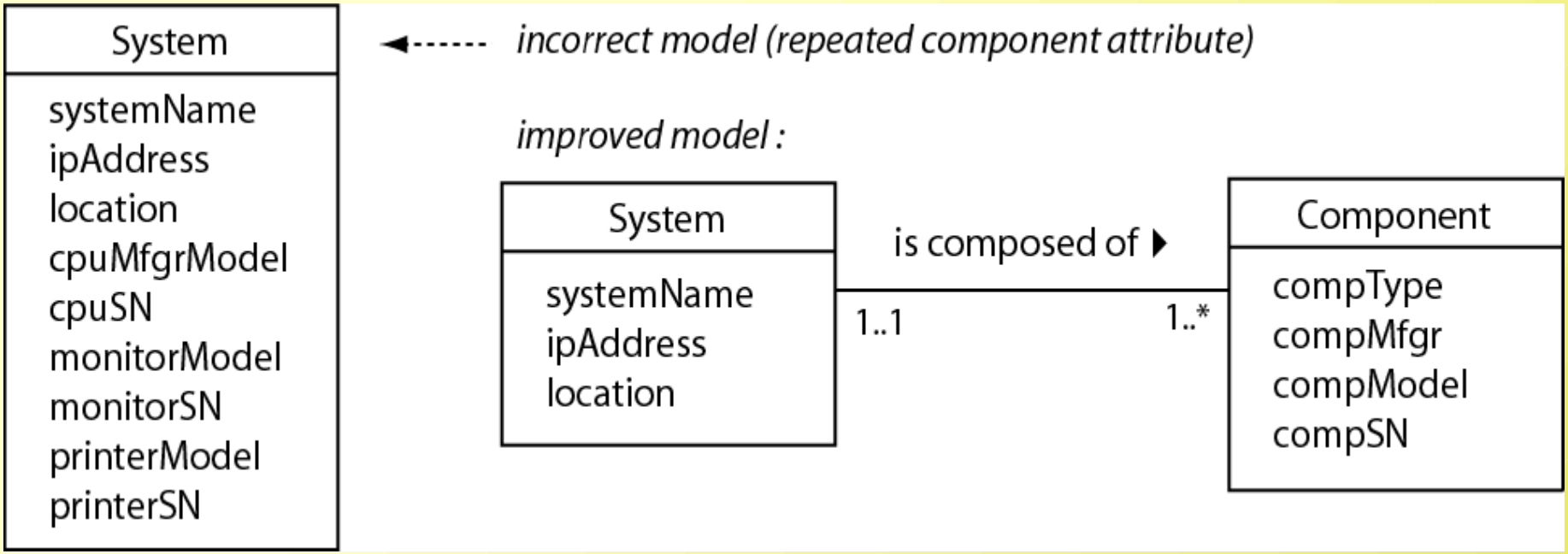


SQL:

create table PhD_študenti (ids, meno, priezvisko, dátum_narodenia,
constraint primary key (ids));
create table Vyučujúci (ids, cvičenie,
constraint primary key (ids), **check** (**exists**
(**select** * **from** PhD_študenti P **where** P.ids = ids);
create table Výskumní (ids, grant,
constraint primary key(ids), **check** (**exists**
(**select** * **from** PhD_študenti P **where** P.ids = ids));

Kusovník – aggregation, composition





Primárny kľúč

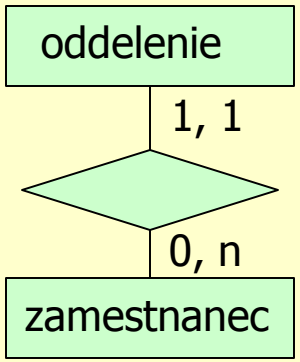
- Vybrať z predmetných (externých) kľúčov
- Surrogate – umelý kľúč (väčšinou číslo) generovaný db-systémom
- Substitute PK – jeden atribút (obvykle skratka) napr. trojpísmenové označenie letísk

Výber PK treba starostlivo zvážiť. Často je vhodné použiť surogát, aj keď prirodzený externý kľúč existuje. Dôvod: db neumožňuje modifikovať primárny kľúč.

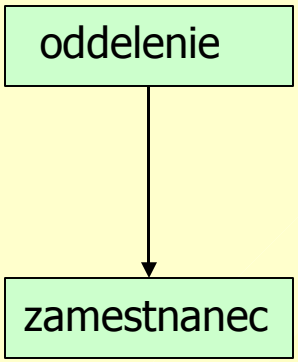
Príklad nevhodného externého primárneho kľúča je napr. rodné číslo, psč, … .

Základné konštrukcie I

ER-model



Sieťový

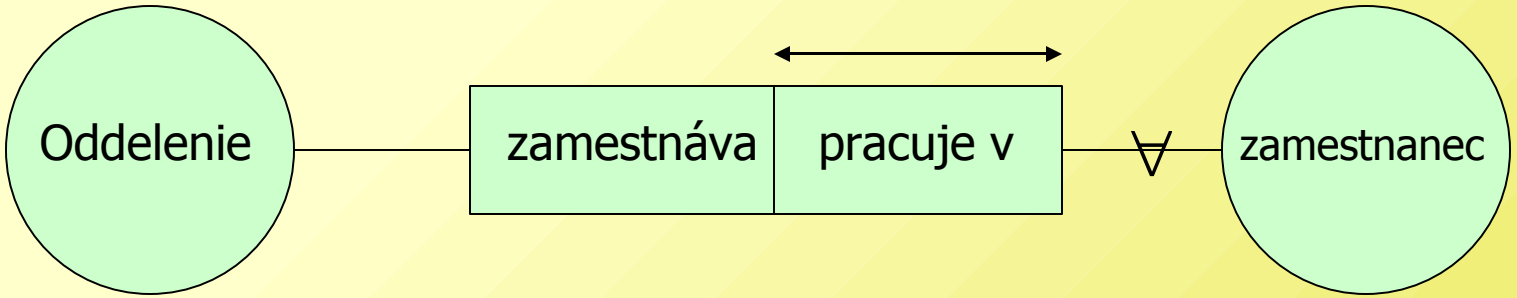


Relačný

Oddelenie(ČísOdd, …)

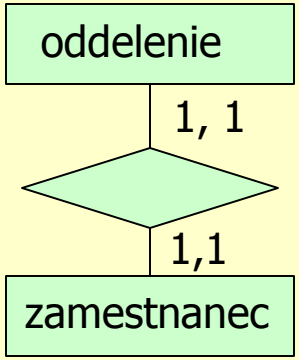
Zamestnanci(IdZam, ČísOdd, …)

Binárny model

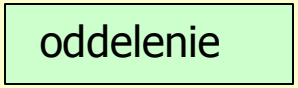


Základné konštrukcie II

ER-model



Sieťový



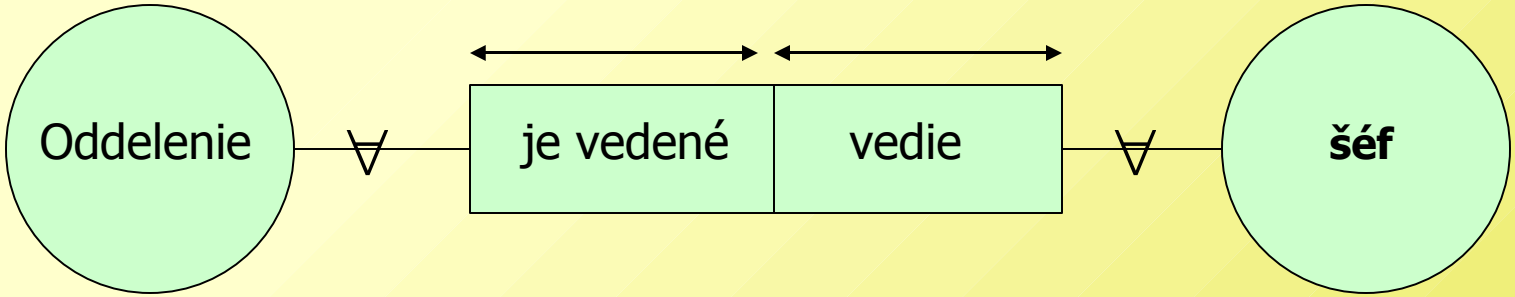
Record type
oddelenie(ČísOdd, IdŠéfa, ...)

Relačný

oddelenie(ČísOdd, IdŠéfa, ...)

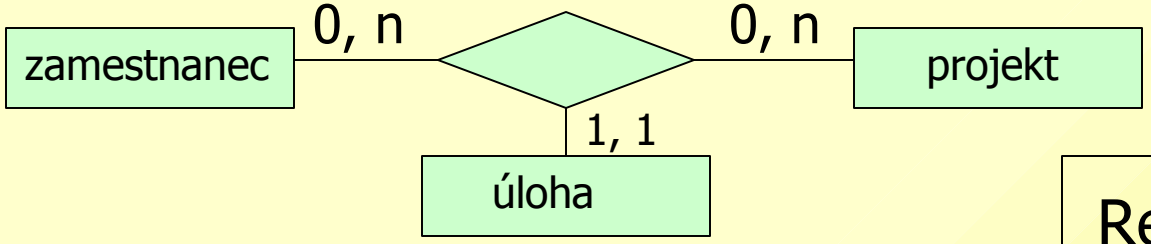
Existencia samostatných typov viet pre oddelenie a šéfa je možná, ale nie nutná.

Binárny model

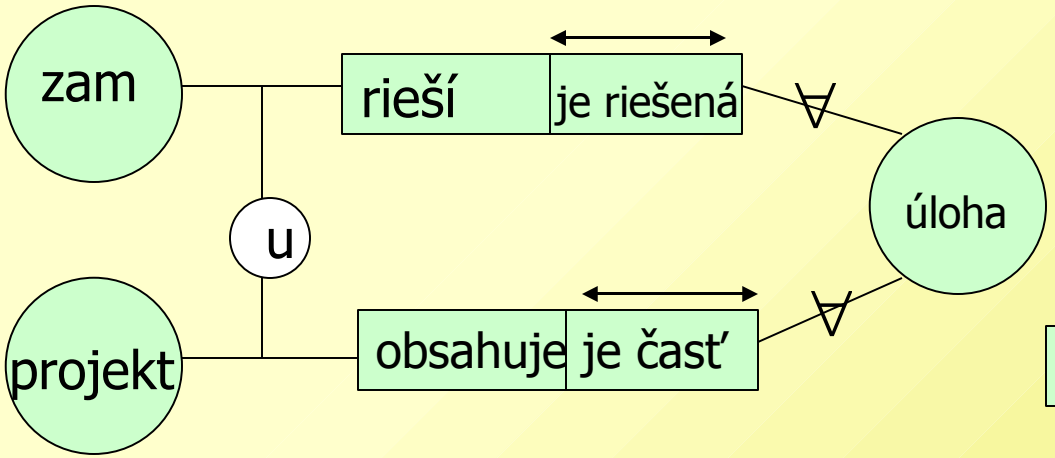


Ternárne vzťahy I

ER-model



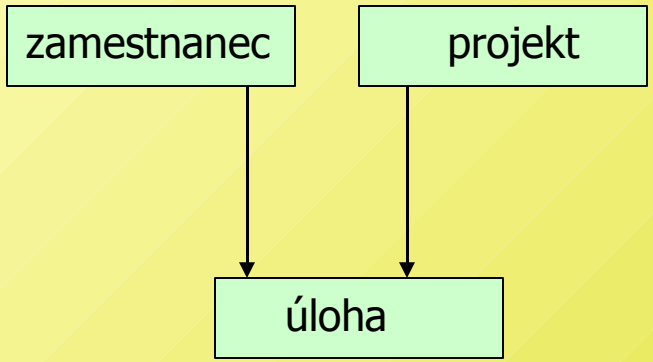
Binárny model



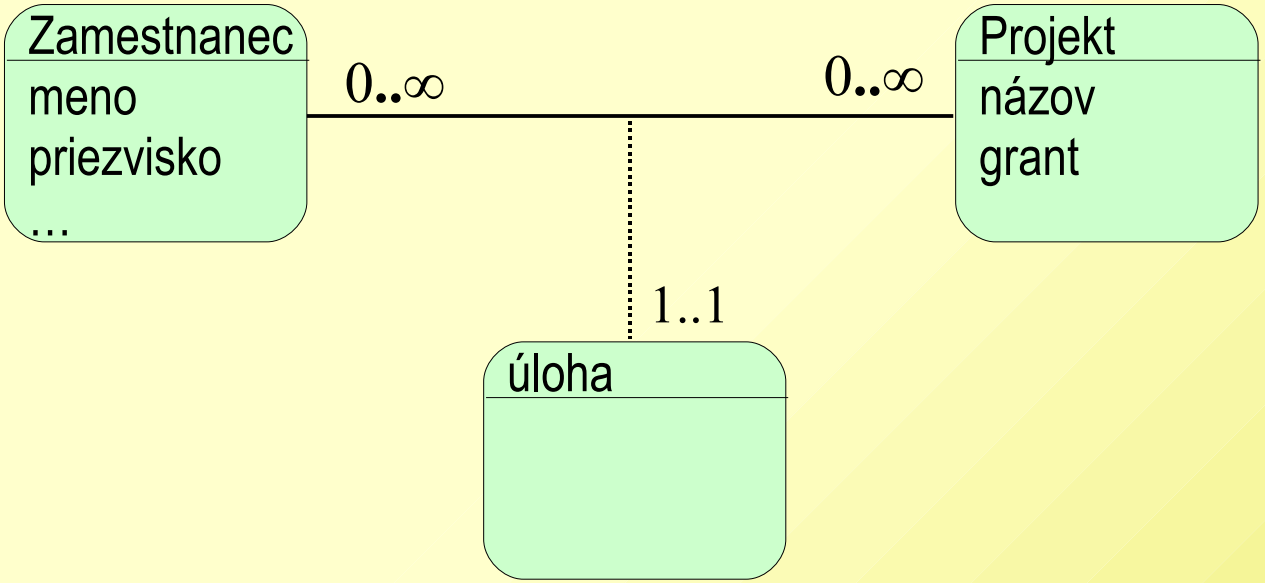
Relačný model

zamestnanci(IdZam, …)
projekty(ČísProj, …)
úlohy(IdZam, ČísProj, …)

Sieťový model



Ternárne vzťahy I – UML, SQL

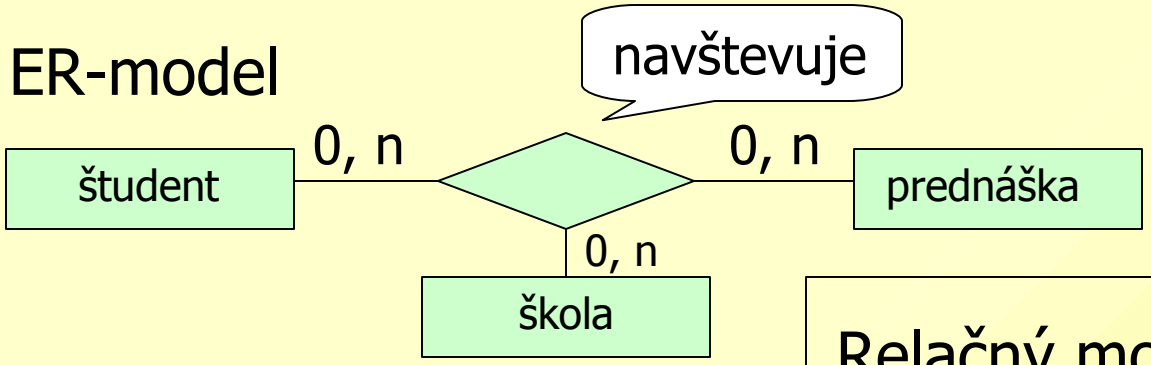


SQL:

create table Zamestnanec(idz, meno, priezvisko, ...,
primary key (idz));
create table Projek(čp, meno, priezvisko,
constraint primary key (čp));
create table Úloha(idz, čp, **primary key**(idz, čp),
foreign key idz **references** Zamestnanec, **on delete cascade**
foreign key (čp) **references** Projekt);

Ternárne vzťahy II

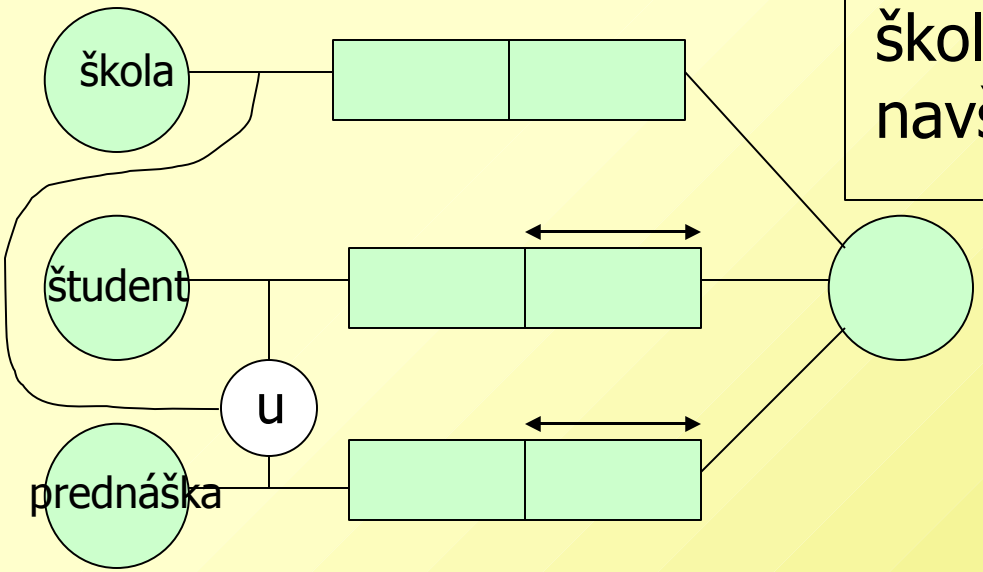
ER-model



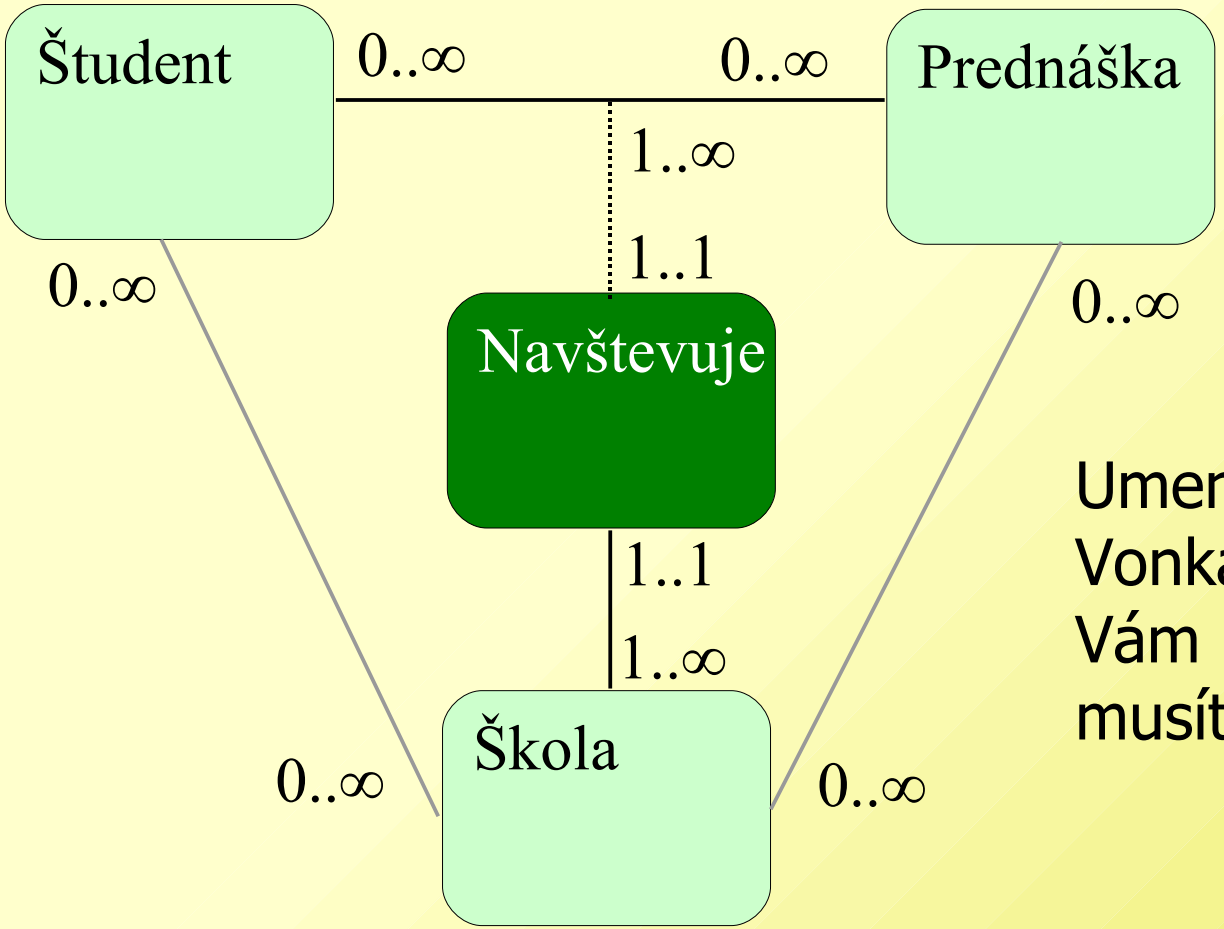
Relačný model

študent(<u>RodČís</u>, …)
prednáška(<u>názov</u>, …)
škola(<u>IČO</u>, …)
navštevuje(<u>názov, RodČís,</u> IČO)

Binárny model



Ternárne vzťahy II – UML



Umenie návrhu:
Vonkajší trojuholník
Vám hovoria, vnútro
musíte objaviť.

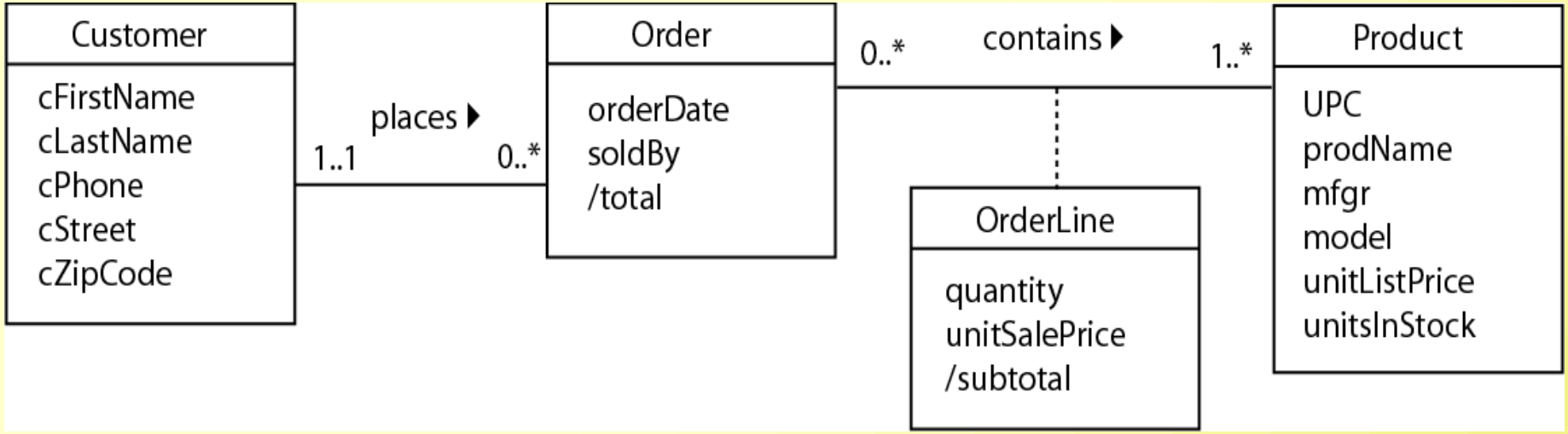
Ternárne vzťahy II – SQL

Create table Študent (rč, meno, priezvisko, … ,
primary key(rč));
create table Prednáška (názov, … ,
primary key(názov));
create table Škola (IČO, … ,
primary key(IČO));
create table Navštevuje (rč, názov, IČO,
constraints primary key(rč,názov),
foreign key (rč) references Študent,
foreign key (názov) references Prednáška,
foreign key (IČO) references Škola);

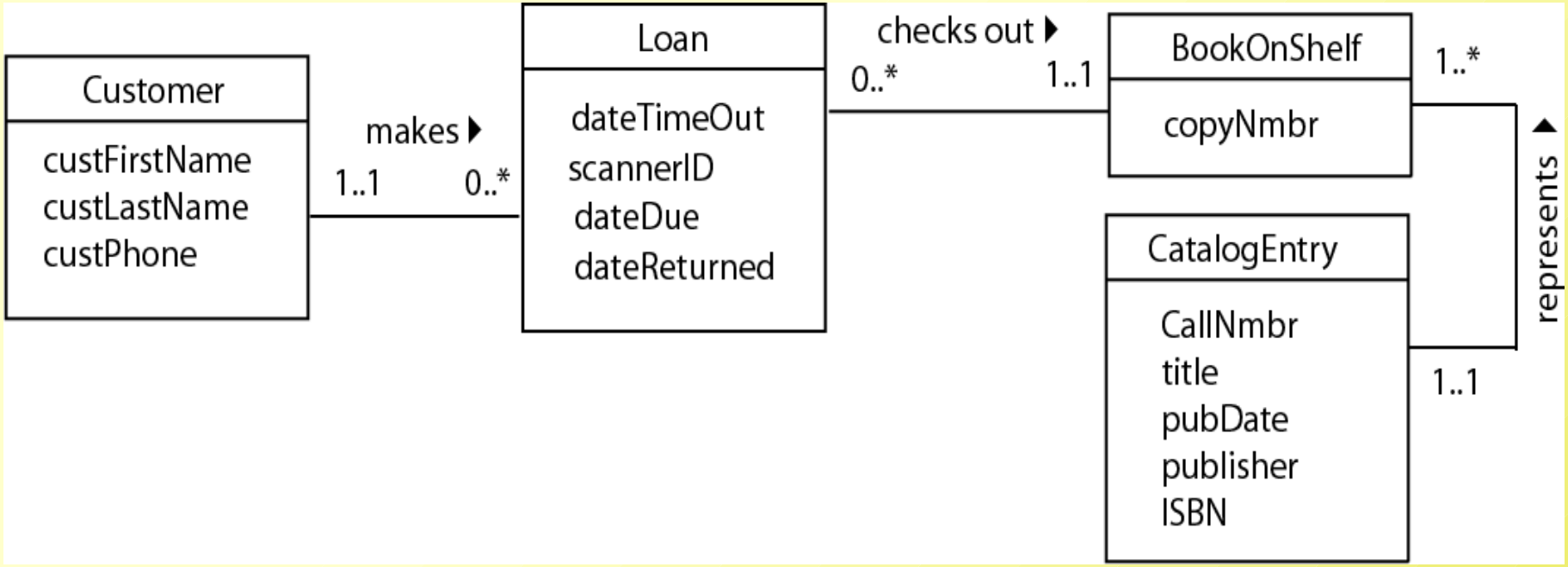
Návrhové vzory

- Je vhodné zbierať dobré riešenia typických situácií.
- Zdroje
  - literatúra
  - skúsenosti
  - reverzné inžinierstvo fungujúcich systémov
- Bez veľkých skúseností je ťažké rozoznať, či je to hlboký poznatok, alebo skostnatená byrokracia.

Zákazník, objednávka, výrobok

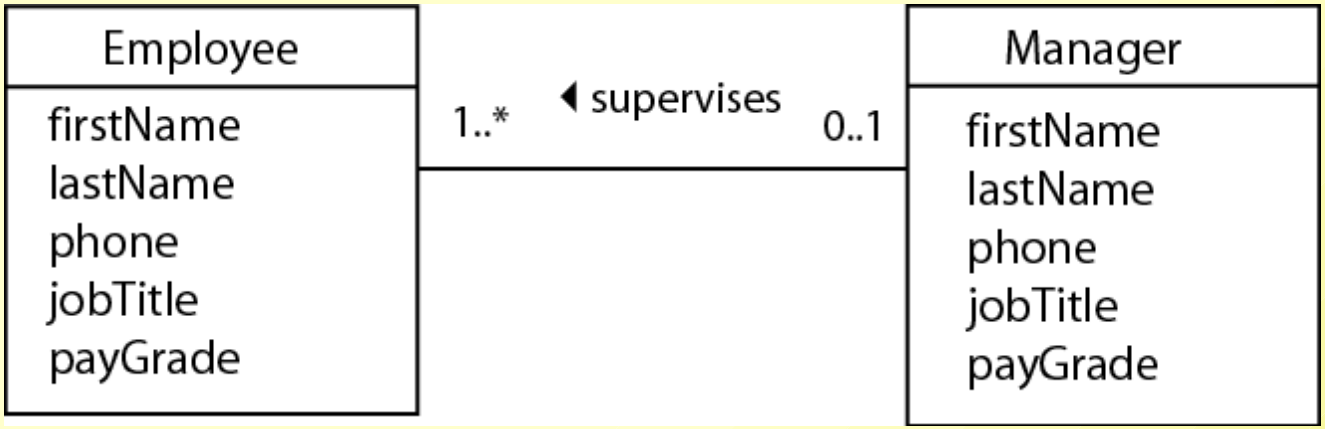


Výpožičky v knižnici

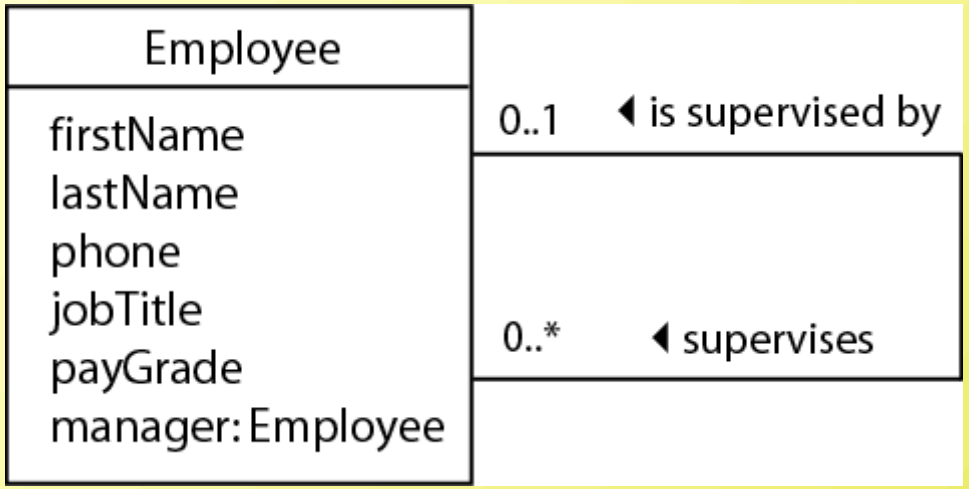


Šéfovská hierarchia

Nesprávny návrh:



Správny návrh:



Preklad grafickej notácie do relačného modelu a SQL

- Väčšinou je to priamočiara záležitosť
- Množiny entít, triedy prekladáme priamo na relácie, tabuľky
  - problém je voľba primárneho kľúča (automatické systémy uprednostňujú surogátny pk)
  - ostatné podmienky je potrebné zadávať interaktívne
- Vťahy sa prekladajú binárnymi reláciami, kľúč sú oba atribúty v prípade (many to many) inak je na strane one.
- V prípade may to one vzťahu možno binárnu reláciu nahradiť importovaním cudzieho kľúča do podriadenej relácie