

# Úvod do softwarového inženýrství (IUS)

## Druhé cvičení

Brno University of Technology, Faculty of Information Technology  
Božetěchova 1/2, 612 66 Brno - Královo Pole  
Petr Veigend, [iveigend@fit.vut.cz](mailto:iveigend@fit.vut.cz)



- Značná část těchto slidů je převzata a rozšířena ze slidů předmětu IUS (Radek Kočí, Bohuslav Křena, Jaroslav Zendulka)

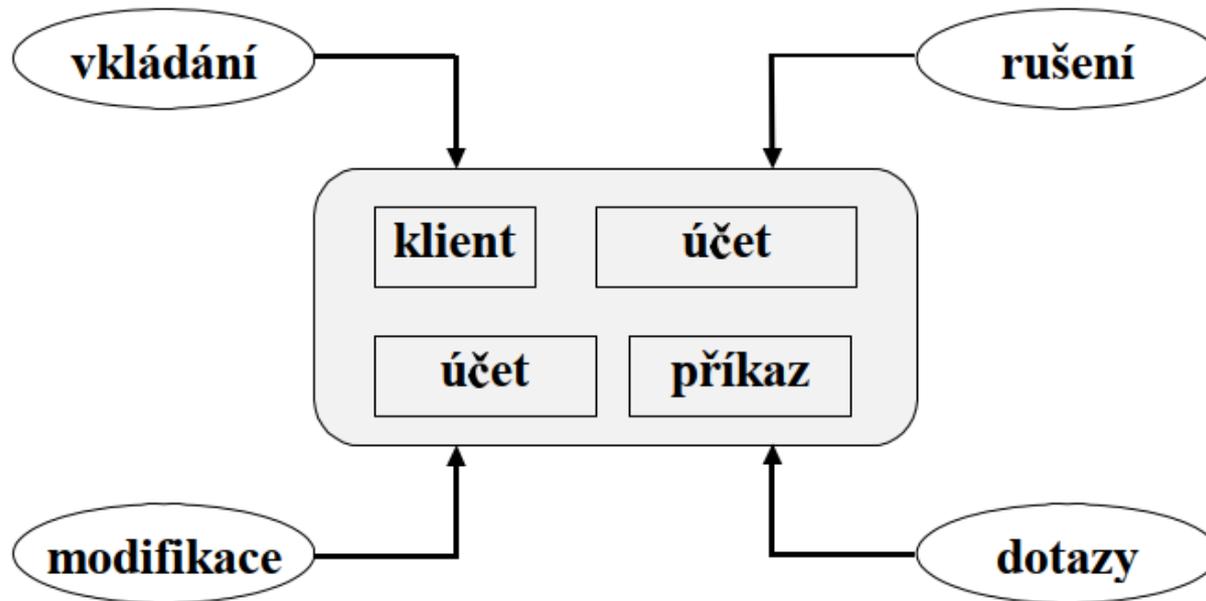
- Jmenuji se **Petr Veigend, studijní poradce**
- **Můj profil:** <https://www.fit.vut.cz/person/iveigend/>
  - Kancelář: A221
  - Konzultační hodiny: po domluvě emailem, případně na kartě
  - Slidy na vizitce
- **Komunikace:**
  - **email – prosím používejte předmět:**  
**IUS - <předmět emailu>**
  - *Discord, Facebook, (ale hůř se dohledává historie a není přesně jasné kdo se ptá, takže „oficiální věci“ raději mailem ... )*

- Cvičení jsou bodovaná (**3 body**)
- Budu hodnotit **aktivní účast**
- **Další informace viz Moodle**
  
- **Domácí úloha**
  - V E-Learningu jste se registrovali na zadání k asistentovi, který bude vaši domácí úlohu konzulovat a opravovat
  - **Deadline: 5.11.2023, 23:59:59**

# DATOVÉ MODELOVÁNÍ

- Cíle
  - Mít všechna potřebná data
  - Nemít nepotřebná data
  - Vyjádřit vztahy mezi daty
  - Popsat, jak se data v systému mění

- ER model slouží k modelování dat aplikační domény “v klidu”

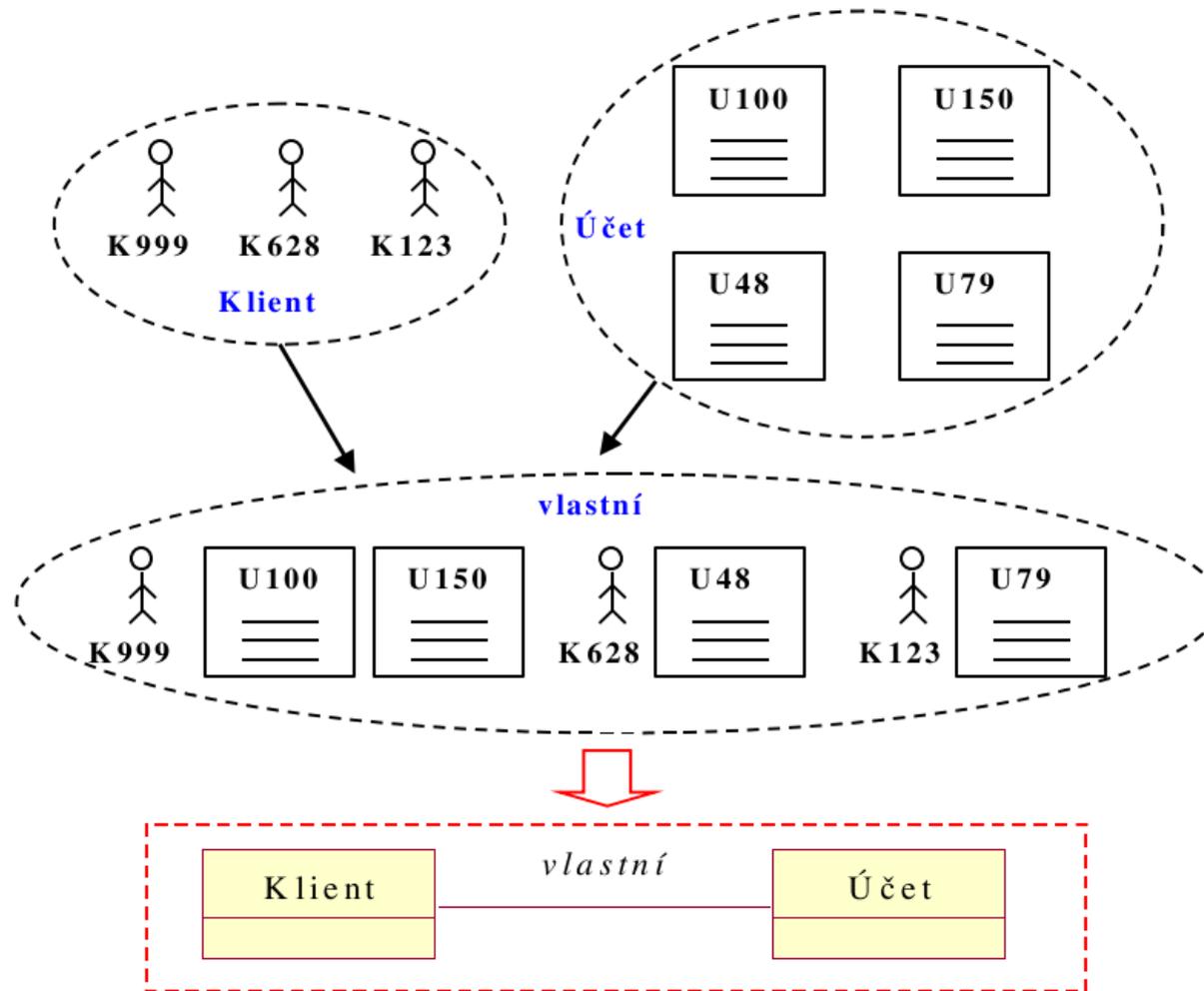


- Otázky, které si pokládáme:
  - Jaká data potřebujeme v systému uchovávat?
  - Jaké vztahy jsou mezi těmito daty?

- Entita
- Entitní množina
- Atribut
- Vztah
- Vztahová množina

- Entita
  - věc reálného světa (objekt), rozlišitelná od jiných objektů
  - Např. klient banky K1, účet v bance U10, předmět P2, ...
- Entitní množina
  - Množina entit téhož typu, sdílí stejné vlastnosti (atributy)
  - Např. Klient, Účet, Předmět, ...
- Atribut
  - Vlastnost entity, která nás zajímá
  - Např. Klient: číslo klienta, jméno, příjmení, adresa, ...

- Vztah
  - Asociace mezi několika entitami
  - Např. klient s číslem K100 vlastní účet U10
- Vztahová množina
  - Množina vztahů téhož typu, které sdílí tytéž vlastnosti
  - Např. Klient vlastní Účet (pro vztah mezi entitami typu Klient a Účet)



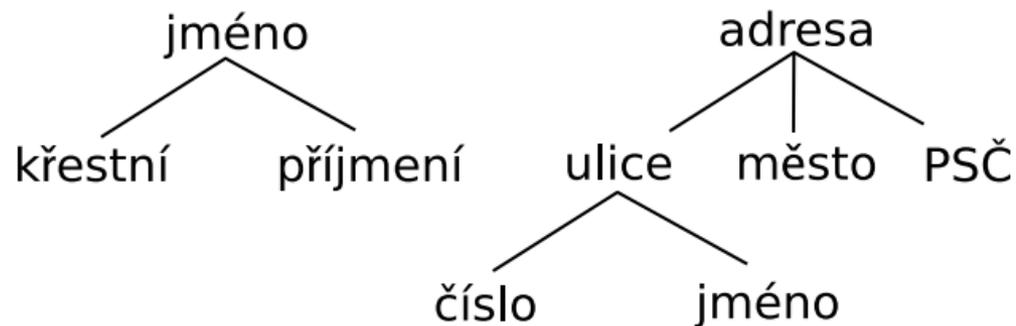
- Jednoduché a složené atributy
  - Složeným se v **reálném návrhu chceme vyhnout**, v IUS stačí použít **atribut složený**

Entitní množina

Klient

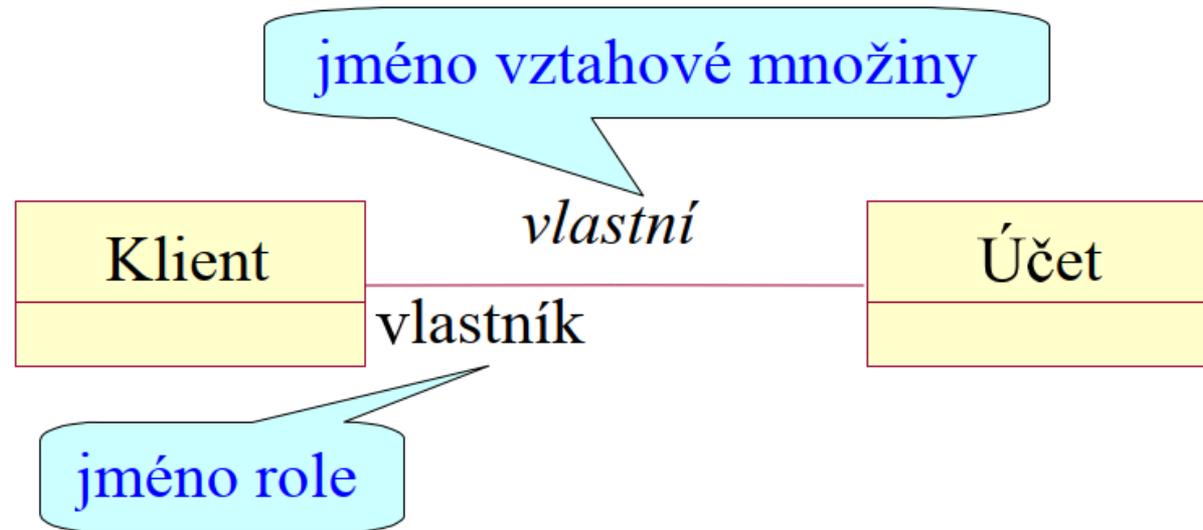
Složené atributy

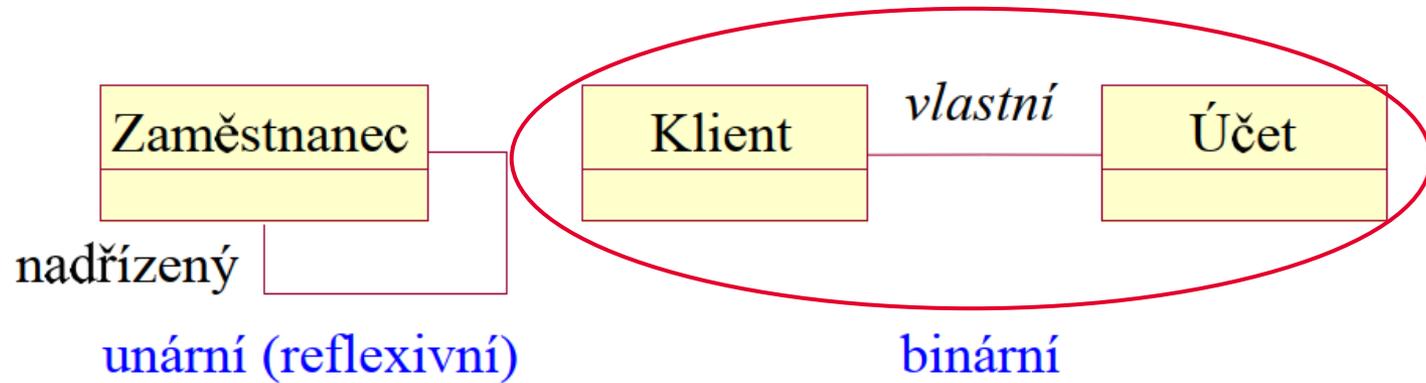
Složky atributu



- Jednohodnotové a vícehodnotové
  - Telefon – více čísel
- Prázdné (NULL) atributy
  - Mohou nabývat speciální hodnoty NULL
  - Může zastupovat
    - Chybějící hodnotu – existuje, ale neznáme ji
    - Neznámou hodnotu – nevíme, jestli existuje
- Odvozené atributy
  - Hodnotu lze odvodit od jiných atributů nebo entit
  - Např.: datumNarození → věk

- Jméno vztahové množiny i jméno role vyjadřuje **význam vztahu**

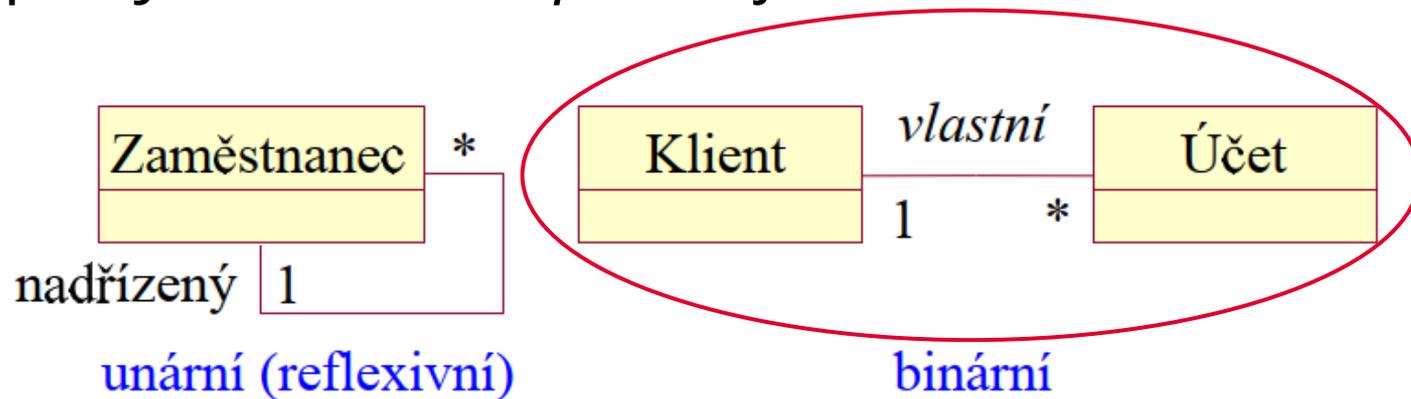




- Jak čteme tuto vazbu (tj. vazební množinu)?

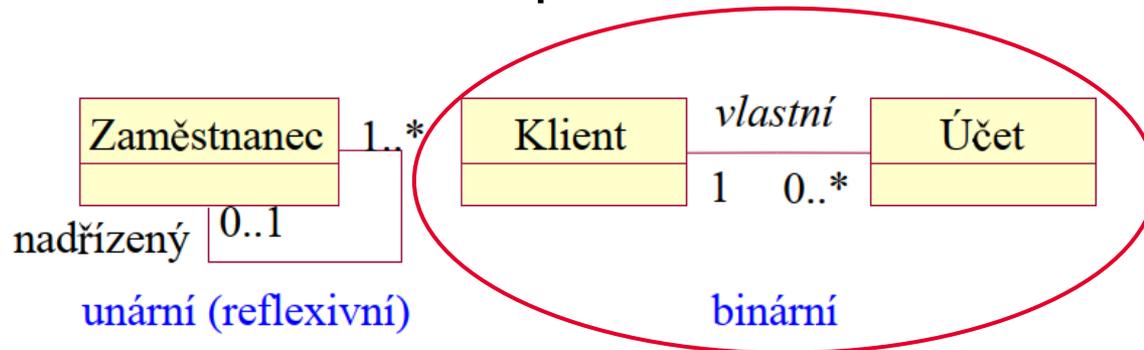


- Kardinalita je **maximální** počet vztahů daného typu (vztahové množiny), kterých se může účastnit jedna entita
- Typicky: 1, M ( $n$ ,  $*$ ), *přesněji*



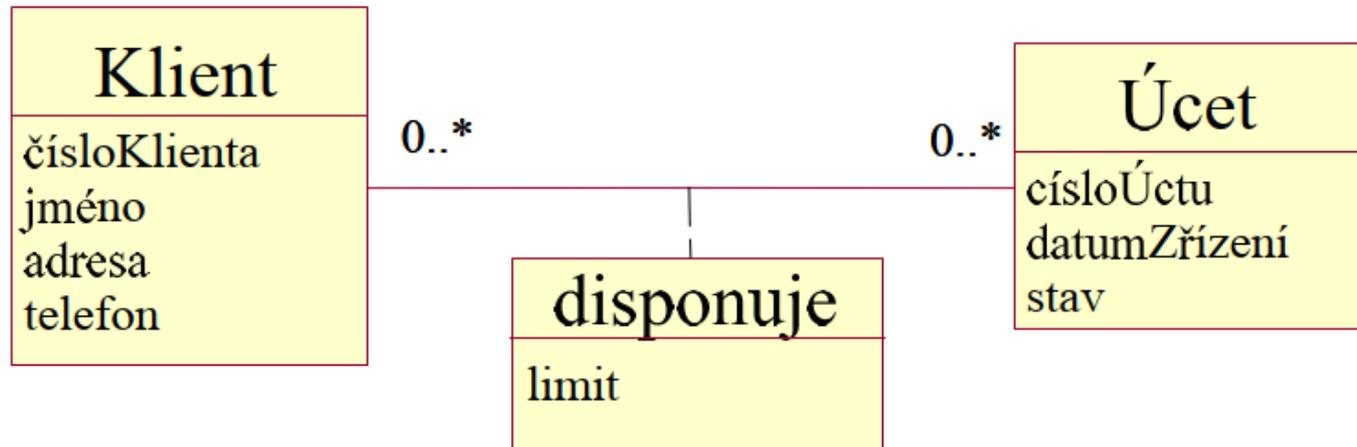
- Jak čteme tyto kardinality?

- Členství / účast je **minimální** počet vztahů daného typu (vztahové množiny), ve kterých může participovat jedna entita.
- Typicky: 0 – volitelné, 1 – povinné



- Jak čteme tato členství?

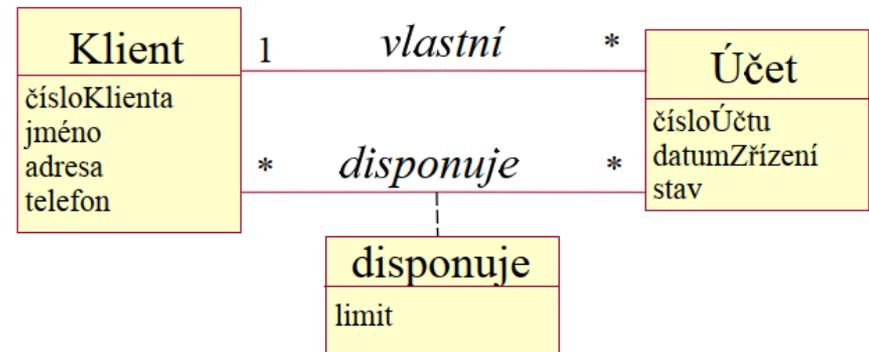
- Pokud atribut **nelze přiřadit** ani jedné z entit
- Vztah povýšený na entitu



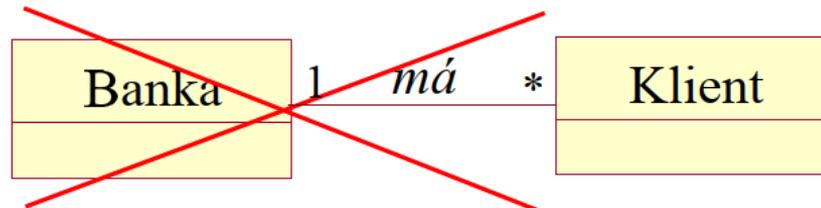
- Proč je u tohoto příkladu atribut vztahu potřeba?
- U vztahových množin jakého typu najdeme atribut vztahu?

- Zobrazuje **pouze data a vztahy mezi nimi**, ne procesy
- Každý atribut je zobrazen **pouze jednou**
- Data se seskupují pro účely databáze, ne výstupních sestav
- Zobrazujeme pouze datové objekty, které jsou v systému trvale
- Zobrazujeme pouze nezbytně nutné vztahy
  - Učitel učí Předmět, který má zapsaný Student
  - Učitel učí Student → redundantní
- **Pozor na entity**
  - Bez atributů
  - Které mají pouze identifikátor
  - Které se vyskytují pouze jednou
  - Které obsahují cizí atributy (patří jiným entitám)

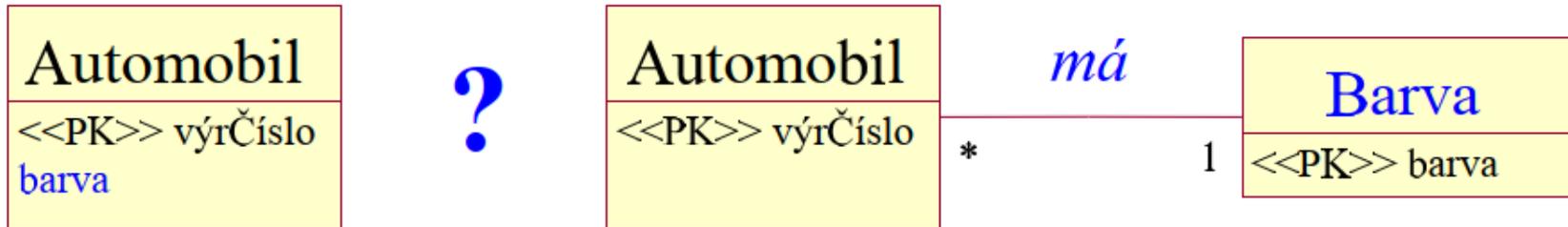
- **Jména**
  - Musí být srozumitelná a musí vyjadřovat význam entitních a vztahových množin.
  - Entitní množiny: **podstatná jména**
  - Vztahové množiny: **slovesa, předložky**
  - V předmětu IUS se vztahy **pojmenovávají vždy**
- Mezi stejnými entitními množinami **může být několik** vztahových množin.



- Identifikátor (klíč, **primární klíč - <<PK>>**)
  - Entity a vztahy musí být identifikovatelné
  - Hodnota identifikátoru **musí být unikátní** (a minimální)
    - **POZOR:** v rámci zadání jsou jejich názvy jednoznačně určeny
  - Identifikátorem je jednoduchý nebo složený atribut
  - Hodnota **musí být unikátní** pouze v rámci vyvíjeného systému
- Celkový systém by neměl být zahrnut do ERD.

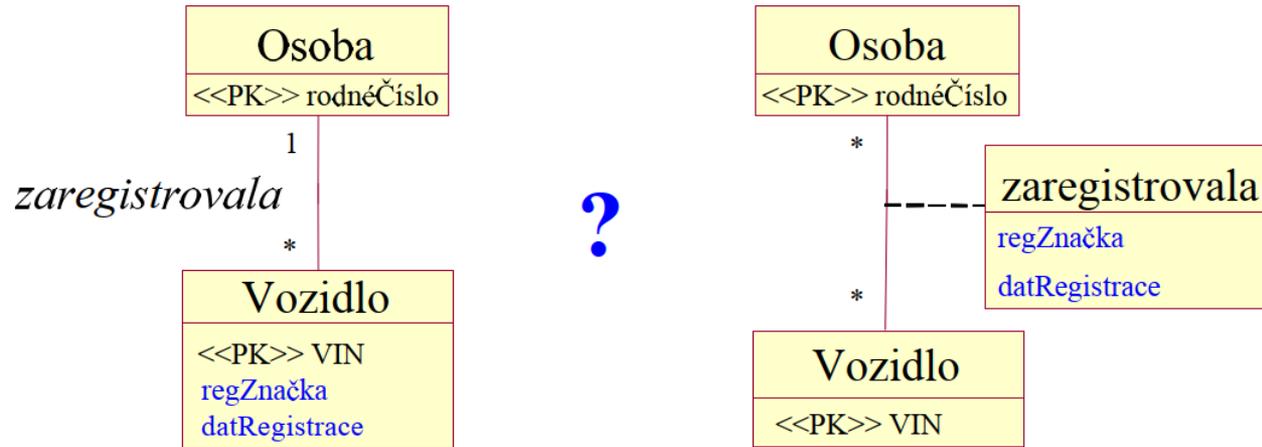


- Entitní množina nebo atribut?



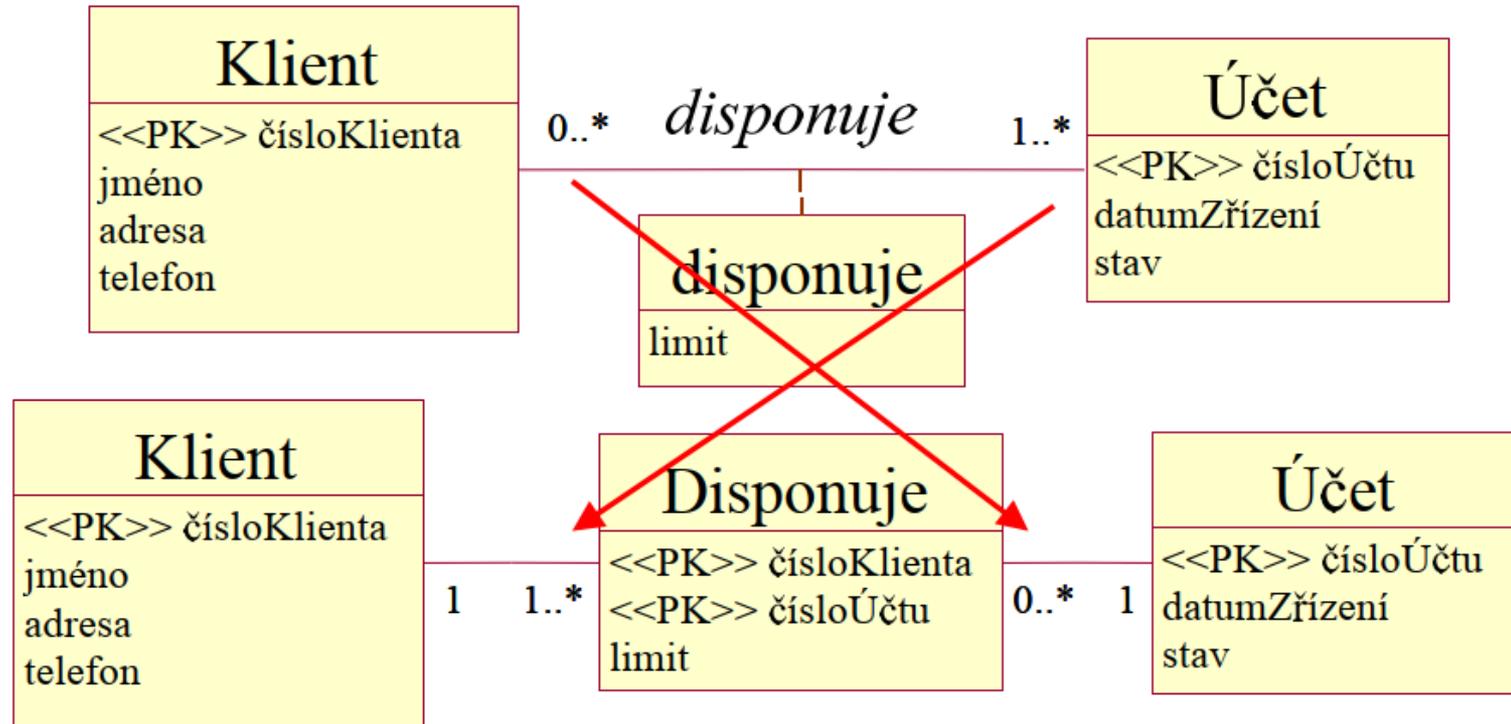
- Kdy bychom potřebovali barvu definovat jako vlastní entitní množinu?
- Pokud je hodnota atributu důležitá, i když neexistuje žádná entita s touto vlastností, pak bychom ji měli modelovat jako entitu

- Kardinalita a umístění atributů



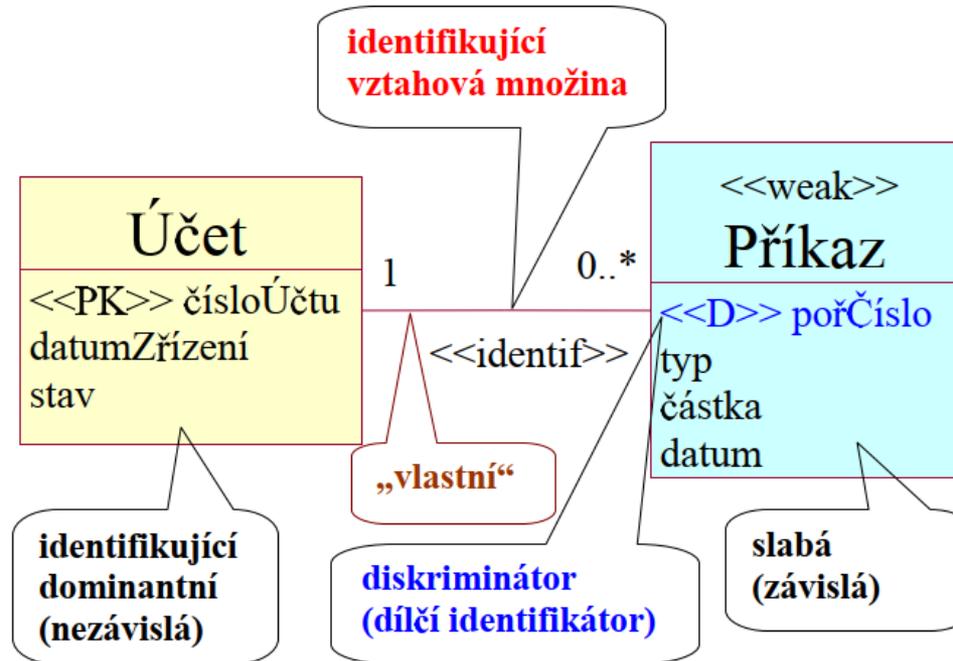
- Jak se od sebe liší oba diagramy?
- Proč by u tohoto příkladu bylo výhodné použít atribut vztahu?
- Existuje ještě nějaké jiné řešení?

- Náhrada vztahů M:M (\*:\*) **vazební entitní množinou**



- Jsou obě tyto možnosti ekvivalentní?

- Entity modelované slabou entitní množinou **nemohou existovat** bez silné entitní množiny.



- Rysy slabé entitní množiny:
  - Identifikátor = identifikátor\_dominantní + diskriminátor
  - Existenční závislost slabé na identifikující
- Slabá nebo silná entitní množina?
  - Jsme-li na pochybách, modelujeme jako **silnou entitní množinu**
  - Další viz slidy IUS

- 1) Zvolte jednu entitu ze specifikace požadavků
- 2) Určete atributy entity, označte kandidátní klíče
- 3) Prověřte atributy, zda je potřeba zaznamenat informace o některém z atributů v samostatné entitě
- 4) Další entita → krok 1)
- 5) Vytvořte vztahy mezi entitami
- 6) Určete, zda některé atributy potřebují být identifikovány pomocí více entit → atribut přiřadte vztahu, který spojuje příslušné entity
- 7) Identifikujte a odstraňte redundantní vztahy

Tento algoritmus **není potřeba** přesně dodržovat.

# UČEBNY

Navrhněte modul fakultního informačního systému, který bude umožňovat správu učeben a laboratoří na FITu. Systém musí uchovávat základní informace o učebnách, jejich umístění, kapacitu a vybavení (projektor, klimatizace, kamera, počet a typ tabulí, ...). Dále musí systém umožňovat rezervaci učeben pro různé akce (zkoušky, semináře, cvičení, ...). Rezervace učeben a laboratoří mohou provádět jen akademičtí pracovníci, změny informací o učebnách a laboratořích provádí pouze správce systému.

Systém musí umožňovat vypisovat rozvrh konkrétní učebny a také rozvrh po oborech, ročnících, nebo předmětech. Navíc musí umožnit zjistit volné učebny v daném termínu, případně s konkrétním vybavením. Systém také musí uchovávat informace o tom, kdo a kdy danou učebnu rezervoval.

# DISTRIBUTOR ČAJE

Vytvořte návrh informačního systému pro distributora čajů. Distributor má na skladě různé druhy čajů (zelený, černý, oolong, Pu-erh, bylinný, ovoněný, ...). Každý druh čaje je dodán z některé země, volitelně z oblasti v dané zemi (např. oblasti Darjeeling či Assam v Indii). U oblastí je nutné uchovávat popis oblasti a charakteristiku typického čaje z dané oblasti. U některých druhů čajů je dána jejich kvalita podle stupnice (OP, FOP, ...), dodavatel, doba louhování, atp. Pro čaje na skladě je nutné mít přehled o tom, ze které várky pocházejí. Cena druhu čaje se liší podle várky, nelze tudíž míchat více várek dohromady. Odběratelé si pak v systému mohou vytvořit objednávku na určité množství konkrétní várky. V rámci jedné objednávky si lze objednat různé čaje z různých várek. Celá objednávka je poté fakturována v rámci jedné faktury. Odběratel má možnost vypsát si druhy čajů na skladě a ke každému čaji i várky a množství čaje v jednotlivých várkách. Distributor má možnost staré čaje zlevnit a odběratelé vidí jak zlevněnou, tak i původní cenu.

Děkuji za pozornost