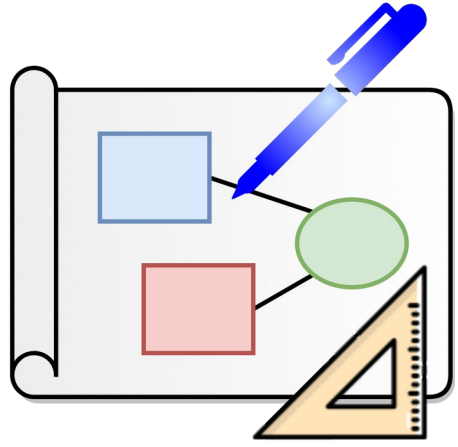
Databázové systémy

Vladislav Novák

1. cvičenie

Obsah cvičení počas semestra

1. polovica semestra
   - dátový model, návrh databáz
   - princípy platné pri tvorbe každého softvéru, nie len pri tom, čo bežne voláme databázy

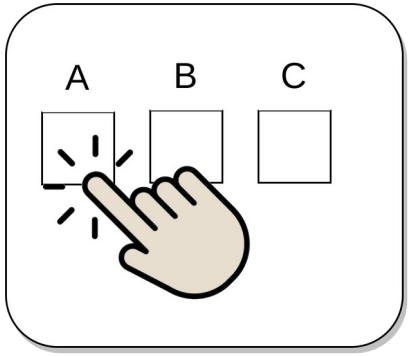


2. polovica semestra
   - relačné databázy
   - programovací jazyk SQL

Web stránka ku cvičeniam v 1. polovici semestra

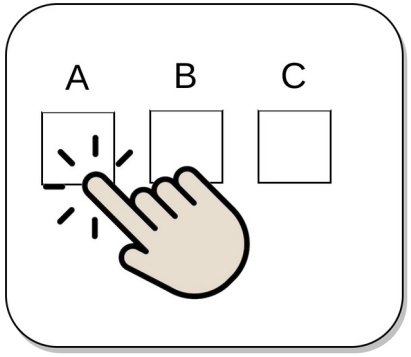
- https://useobjects.net/dbs/
- pracovná verzia učebného textu (môže sa meniť)

Domáce úlohy - testy v moodle



- V testoch základy, na písomkách budú náročnejšie úlohy
- V 1. polovici semestra
- Každý týždeň
- Na vypracovanie približne týždeň
- Každý test je potrebné vypracovať do utorka 23:59 v nasledujúcom týždni
- Každá úloha v teste je za 0,05 bodu
  - spolu zo všetkých testov môžete získať 7 bodov (v skutočnosti trochu viac)
- Test môžete opakovať ľubovoľný počet krát
  - do hodnotenia sa bude brať najlepšie dosiahnuté hodnotenie
- Počet úloh môže byť každý týždeň iný

Domáce úlohy - testy v moodle



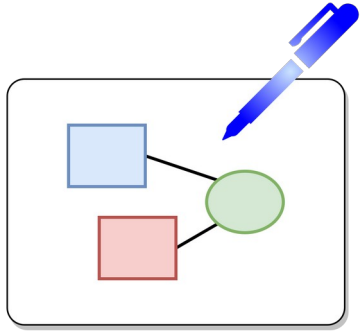
- V testoch základy, na písomkách budú náročnejšie úlohy
- V 1. polovici semestra
- Každý týždeň
- Na vypracovanie približne týždeň
- Každý test je potrebné vypracovať do utorka 23:59 v nasledujúcom týždni
- Každá úloha v teste je za 0,05 bodu
  - spolu zo všetkých testov môžete získať 7 bodov (v skutočnosti trochu viac)
- Test môžete opakovať ľubovoľný počet krát
  - do hodnotenia sa bude brať najlepšie dosiahnuté hodnotenie
- Počet úloh môže byť každý týždeň iný

Moodle

- https://elearn.elf.stuba.sk/moodle/
- Použite prihlasovacie údaje do AIS
  - nevytvárajte si nový používateľský účet!
- Návod
  - linku ste už dostali emailom
  - https://docs.google.com/presentation/d/13CatIADiiap0xR5ZDBEVOApMzS1OMIeZ9kRKe6ADoEI/edit?usp=sharing
- V Moodle je veľa kurzov
- Do kurzu Databázové systémy sa prihláste pomocou prihlasovacieho kľúča
  - bol zaslaný emailom

Bonusové príklady

- Odovzdať na začiatku nasledujúceho cvičenia
- Na papiery
- Kreslené ručne
- Uveďte meno a ID
- Píšte čitateľne
- Hodnotenie môže byť málo prísne
  - v závislosti od typu úlohy – pri tvorivých úlohách body hlavne za snahu
    - ak získate bod, neznamená to, že vaše riešenie je správne
    - ak nezískate bod, vaše riešenie je veľmi zlé
  - na písomke bude hotnotenie výrazne prísnejšie!



Bodové hodnotenie v 1. polovici semestra

- Testy v Moodle
  - 7 bodov (v skutočnosti okolo 7,5 bodu)
- Písomka v polovici semestra
  - 23 bodov
- Bonus
  - ? bodov

Kedy majú študenti voľno (podľa rozvrhu) na písomku

Deň/Hodina	5.00 - 5.50	6.00 - 6.50	7.00 - 7.50	8.00 - 8.50	9.00 - 9.50	10.00 - 10.50	11.00 - 11.50	12.00 - 12.50	13.00 - 13.50	14.00 - 14.50	15.00 - 15.50	16.00 - 16.50	17.00 - 17.50	18.00 - 18.50	19.00 - 19.50	20.00 - 20.50	21.00 - 21.50
Pondelok	100 %	100 %	99 %	97 %	81 %	5 %	7 %	97 %	45 %	45 %	44 %	44 %	100 %	100 %	100 %	100 %	100 %
Utorok	100 %	100 %	98 %	4 %	4 %	2 %	1 %	95 %	2 %	2 %	92 %	92 %	100 %	100 %	100 %	100 %	100 %
Streda	100 %	100 %	99 %	0 %	0 %	0 %	0 %	100 %	8 %	8 %	14 %	14 %	56 %	56 %	93 %	93 %	100 %
Štvrtok	100 %	100 %	93 %	90 %	80 %	45 %	57 %	95 %	48 %	48 %	33 %	33 %	75 %	75 %	100 %	100 %	100 %
Piatok	100 %	100 %	100 %	73 %	73 %	68 %	68 %	100 %	88 %	88 %	95 %	95 %	100 %	100 %	100 %	100 %	100 %

7. týždeň semestra, utorok 31. 3. ?????

Rozvrh

1. polovica semestra

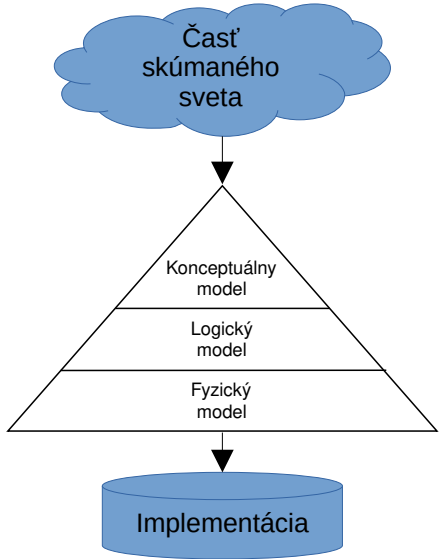
Hod Zac	1 7:00	2 8:00	3 9:00	4 10:00	5 11:00	6 12:00	7 13:00	8 14:00	9 15:00	10 16:00	11 17:00	12 18:00	13 19:00	14 20:00
Pon														
Uto														
Str		prednáška		cvičenie										
Stv														
Pia														

odovzdať bonus

test do 24:00

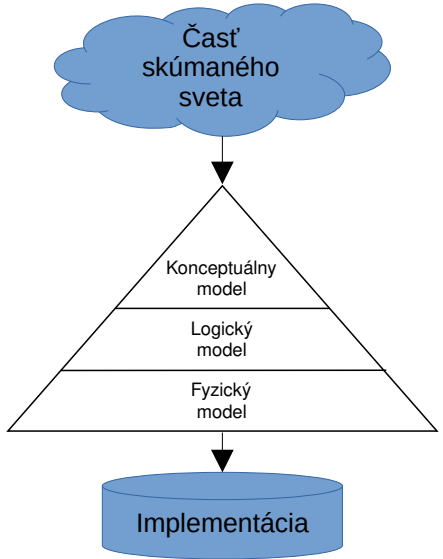
Hierarchia dátových modelov

- Konceptuálny návrh



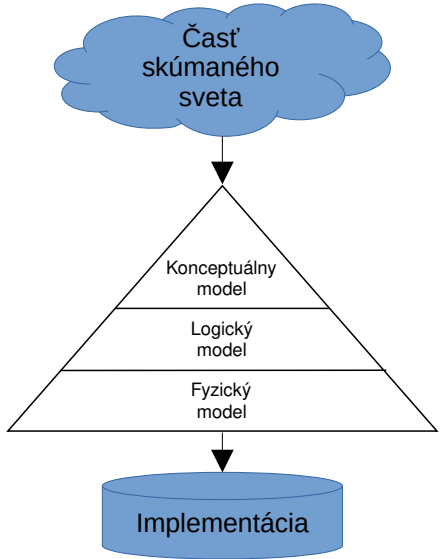
Hierarchia dátových modelov

- Konceptuálny návrh
  - nezávislý na type databázy
- Logický model



Hierarchia dátových modelov

- Konceptuálny návrh
  - nezávislý na type databázy
- Logický model
  - závislý od typu databázy (na akom princípe pracuje)
  - (relačná, grafová, dokumentová, … databáza)
- Fyzický model

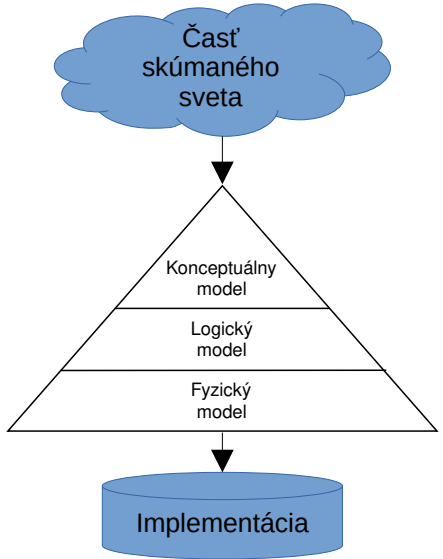


Hierarchia dátových modelov

- Konceptuálny návrh
  - nezávislý na type databázy
- Logický model
  - závislý od typu databázy (na akom princípe pracuje)
  - (relačná, grafová, dokumentová, … databáza)
- Fyzický model
  - závislý od špecifických vlastností implementácie
  - (Oracle, Postgresql, MySQL, SQLite, MongoDB, Neo4j, Redis)

1. až 3. cvičenie

4. až 6. cvičenie



Tvorba informačného systému

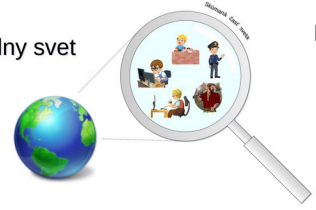
Vladislav Novák

Zadanie:
Vytvorte evidenciu zamestnaní.

Analytik: Čo konktréne treba evidovať?

Zákazník: Kto pracuje v akej profesii.

Reálny svet



Analýza požiadaviek:

na funkcionalitu ← na predmete budeme riešieť len niektoré pravidlá, súvisiace s údajmi

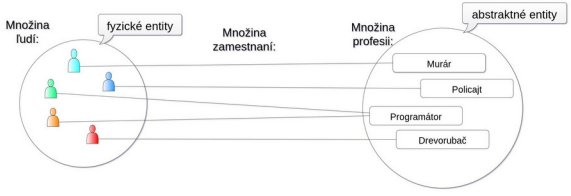
ďalšie požiadavky

na údaje → na tomto predmete

Špecifikácia systému

Návrh:

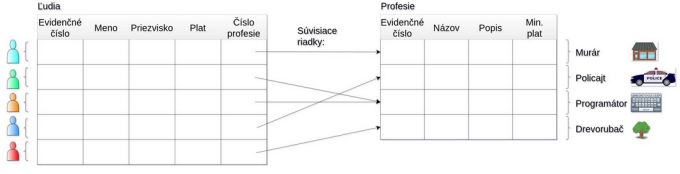
Konceptuálny model:



Analyzujeme a navrhujeme:
- štruktúru údajov:
  typy entít, typy vzťahov
- pravidlá (obmedzenia):
  Môže jeden človek pracovať vo viacerých profesiach? V rôznych časoch? V tom istom čase?
  Môže človek meniť profesiu?
  Môže byť v databáze profesia, ktorú nevykonáva žiadny človek (evidovaný v databáze)?

Musíme vedieť, ktoré údaje reprezentujú, ktoré entitiy reálneho sveta.
Časť údajov musí byť identifikátorom. Používame označenie kľúč.

Logický model relačnej databázy:



Fyzický model relačnej datatabázy:

Databázu upravíme podľa konkrétnych implemenačný detailov použitého systému

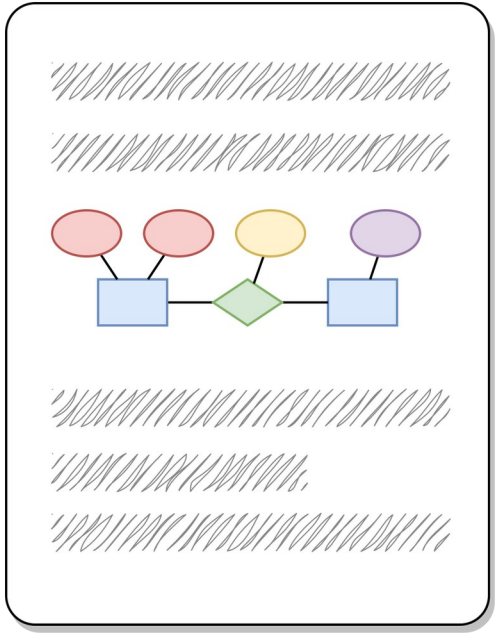
Implementácia

Nasadenie

Údržba

Dátový model

- Diagram
- Textový popis obsahuje to, čo sa nedá zachytiť v diagrame



Konceptuálny model

- Entitno-relačný model

Konceptuálny model

- Entitno-relačný model
  - angl.: entity-relationship model

ERM

Konceptuálny model

- Entitno-relačný model
  - angl.: entity-relationship model
  - grafický zápis: entitno-relačný diagram

ERM

ERD

Konceptuálny model

- Entitno-relačný model
  - angl.: entity-relationship model
  - grafický zápis: entitno-relačný diagram
- Rozšírený entitno-relačný model
  - angl.: enhanced entity-relationship model/diagram
  - skratky: EER model (EERM) / EER diagram (EERD)

ERM

ERD

Konceptuálny model

- Entitno-relačný model
  - angl.: entity-relationship model
  - grafický zápis: entitno-relačný diagram
- Rozšírený entitno-relačný model
  - angl.: enhanced entity-relationship model/diagram
  - skratky: EER model (EERM) / EER diagram (EERD)

ERM

ERD

(E)ER model
nie je štandardizovaný.
V rôznej literatúre,
rôzne grafické značky,
ale aj význam

Konceptuálny model

- Entitno-relačný model
  - angl.: entity-relationship model
  - grafický zápis: entitno-relačný diagram
- Rozšírený entitno-relačný model
  - angl.: enhanced entity-relationship model/diagram
  - skratky: EER model (EERM) / EER diagram (EERD)

ERM

ERD

(E)ER model
nie je štandardizovaný.
V rôznej literatúre,
rôzne grafické značky,
ale aj význam

Na predmete
budeme používať
zvolené značenie
a sémantiku (význam)

Ďalšie prostriedky pre modelovanie

- UML (Unified Modeling Language)
- ORM (Object Role Model)

Príklad 1.1 – stavebná firma

Vytvorte dátový model firmy, pre evidenciu:
- zamestnancov a ich hierarchie,
- projektov, na ktorých firma a jej zamestnanci pracujú,
- pridelenie áut zamestnancom.

Príklad 1.1.a – stavebná firma

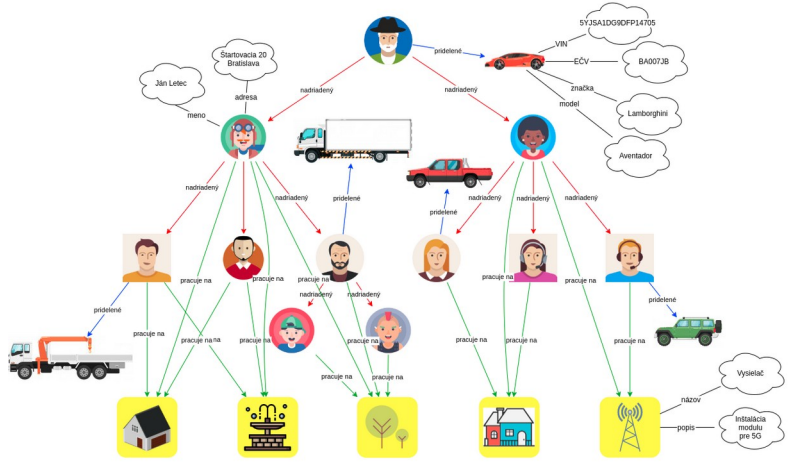
Vytvorte dátový model firmy, pre evidenciu:
- zamestnancov a ich hierarchie,
- projektov, na ktorých firma a jej zamestnanci pracujú,
- pridelenie áut zamestnancom.

a) Identifikujte **typy entít** a **vzťahov** medzi nimi.
V prípade potreby aj **role vo vzťahoch**.
Identifikujte, aké údaje o entitách a vzťahoch bude system obsahovať.
Doplňte model o **atribúty**.

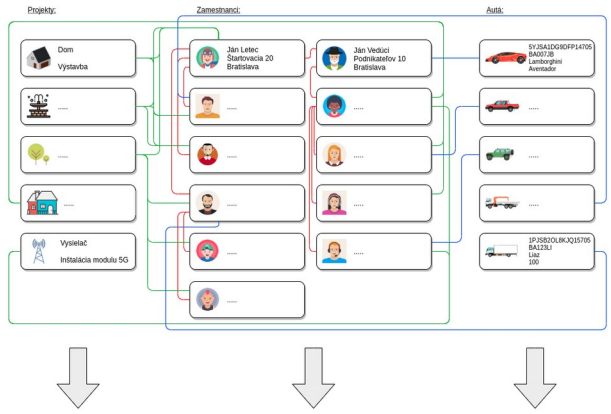
Príklad 1.1.a – stavebná firma – riešenie

Vladislav Novák

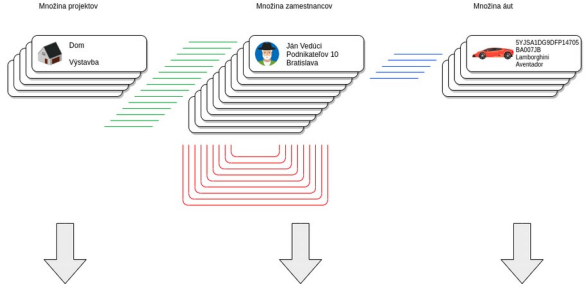
Inštancie v reálnom svete



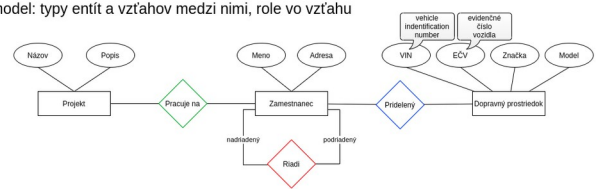
Inštancie ako údaje zapísané na papierových kartách/lístkoch



množiny entít a množiny vzťahov (množiny inštancii)



model: typy entít a vzťahov medzi nimi, role vo vzťahu

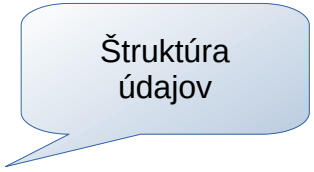


Príklad 1.1.b – stavebná firma

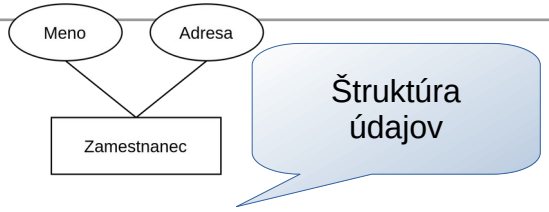
b) Naznačte môžnosti implementácie v C++

Príklad 1.1.b – stavebná firma – riešenie pre C++

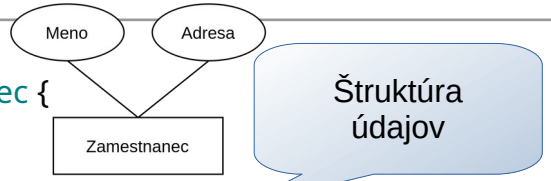
Príklad 1.1.b – stavebná firma – riešenie pre C++



Príklad 1.1.b – stavebná firma – riešenie pre C++



Príklad 1.1.b – stavebná firma – riešenie pre C++

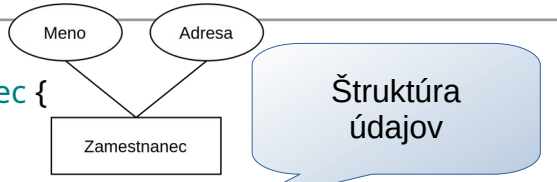


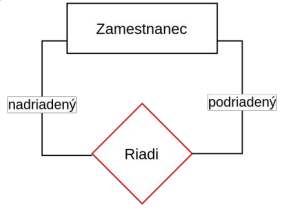
Štruktúra údajov

```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};
```

Príklad 1.1.b – stavebná firma – riešenie pre C++

```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};
```





Príklad 1.1.b – stavebná firma – riešenie pre C++

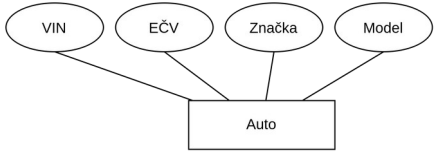
```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};
```

Štruktúra údajov

Príklad 1.1.b – stavebná firma – riešenie pre C++

```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};
```

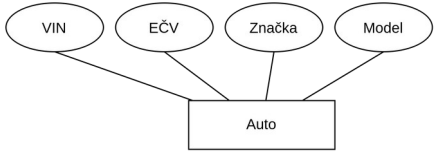
Štruktúra údajov



Príklad 1.1.b – stavebná firma – riešenie pre C++

```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};
```

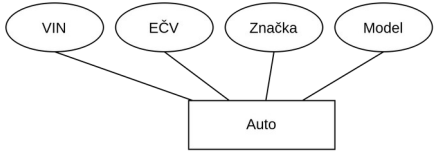
Štruktúra údajov

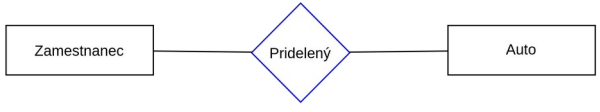


Príklad 1.1.b – stavebná firma – riešenie pre C++


```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};
```

Štruktúra údajov





Príklad 1.1.b – stavebná firma – riešenie pre C++

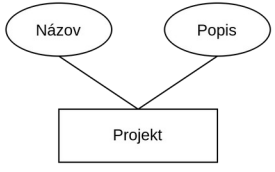

```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};
```

Štruktúra údajov

Príklad 1.1.b – stavebná firma – riešenie pre C++


```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};
```

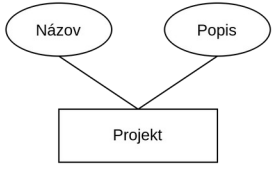
Štruktúra údajov



Príklad 1.1.b – stavebná firma – riešenie pre C++


```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};

struct Projekt {
   string nazov;
   string popis;
   list<Zamestnanec*> zamestnanci;
};
```

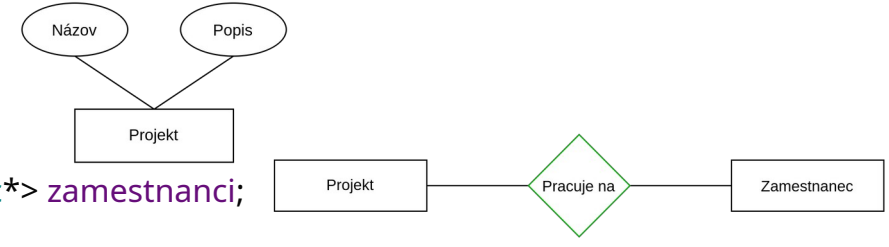
Štruktúra údajov



Príklad 1.1.b – stavebná firma – riešenie pre C++


```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};

struct Projekt {
   string nazov;
   string popis;
   list<Zamestnanec*> zamestnanci;
};
```

Štruktúra údajov



Príklad 1.1.b – stavebná firma – riešenie pre C++


```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;


};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};


struct Projekt {
   string nazov;
   string popis;
   list<Zamestnanec*> zamestnanci;
};
```

Štruktúra údajov

Príklad 1.1.b – stavebná firma – riešenie pre C++


```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;
  // Auto *prideleneAuto;
  // list<Projekt*> projekty;
};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};


struct Projekt {
   string nazov;
   string popis;
   list<Zamestnanec*> zamestnanci;
};
```

Štruktúra údajov

Alternatívna implementácia vzťahov

Príklad 1.1.b – stavebná firma – riešenie pre C++

Štruktúra údajov

```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;
  // Auto *prideleneAuto;
  // list<Projekt*> projekty;
};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};


struct Projekt {
   string nazov;
   string popis;
   list<Zamestnanec*> zamestnanci;
};
```

Príklad 1.1.b – stavebná firma – riešenie pre C++

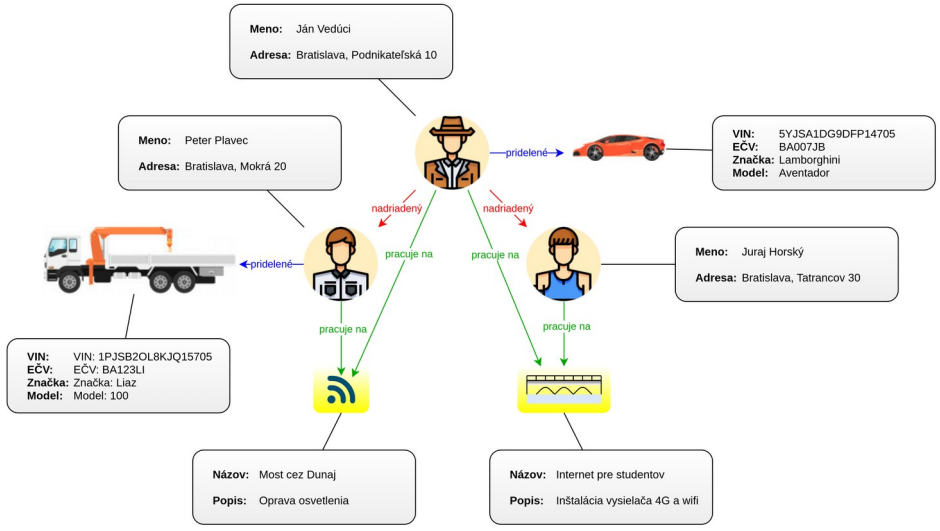
Štruktúra údajov

```
struct Zamestnanec {
    string meno;
    string adresa;
    Zamestnanec *nadriadeny;
   // Auto *prideleneAuto;
   // list<Projekt*> projekty;
};

struct Auto {
    string vin;
    string ecv;
    string znacka;
    string model;
    Zamestnanec *zamestnanec;
};

struct Projekt {
    string nazov;
    string popis;
    list<Zamestnanec*> zamestnanci;
};
```

Údaje

```
Zamestnanec jan;
Zamestnanec peter;
Zamestnanec juraj;
Auto sportiak;
Auto nakladiak;
Projekt most;
Projekt vysielac;
```

Pre prehľadnosť bude v implementácii menej údajov



Príklad 1.1.b – stavebná firma – riešenie pre C++

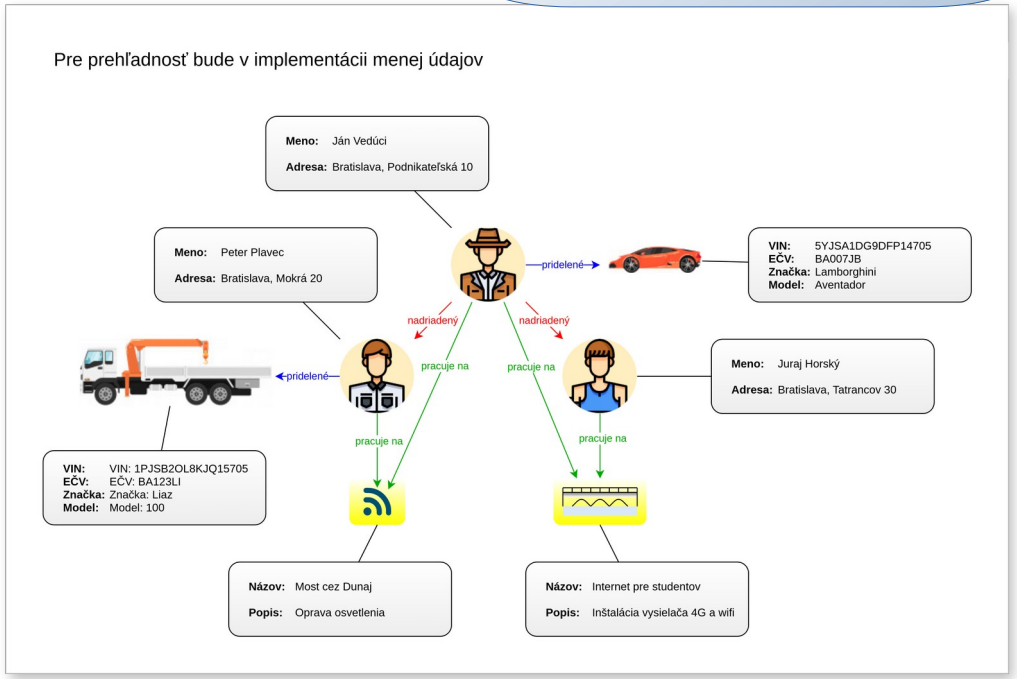
Štruktúra údajov

```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;
  // Auto *prideleneAuto;
  // list<Projekt*> projekty;
};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};

struct Projekt {
   string nazov;
   string popis;
   list<Zamestnanec*> zamestnanci;
};
```

Údaje

```
Zamestnanec jan;
Zamestnanec peter;
Zamestnanec juraj;
Auto sportiak;
Auto nakladiak;
Projekt most;
Projekt vysielac;
```

Napríklad premenné vo funkcii `main`



Príklad 1.1.b – stavebná firma – riešenie pre C++


```
struct Zamestnanec {
   string meno;
   string adresa;
   Zamestnanec *nadriadeny;
  // Auto *prideleneAuto;
  // list<Projekt*> projekty;
};

struct Auto {
   string vin;
   string ecv;
   string znacka;
   string model;
   Zamestnanec *zamestnanec;
};

struct Projekt {
   string nazov;
   string popis;
   list<Zamestnanec*> zamestnanci;
};
```

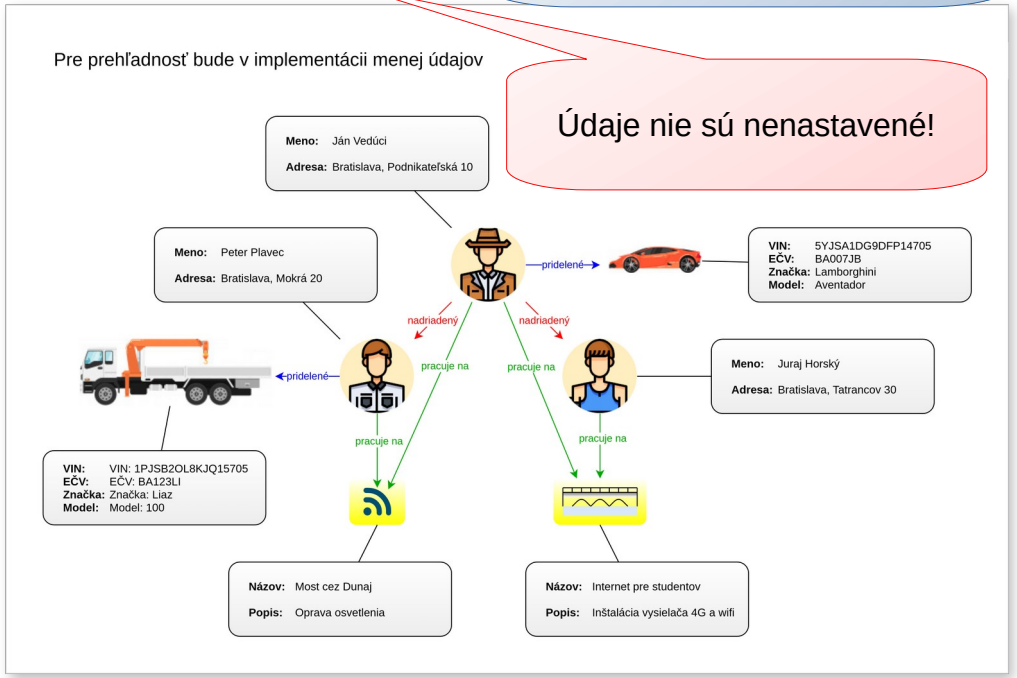
Štruktúra údajov

```
Zamestnanec jan;
Zamestnanec peter;
Zamestnanec juraj;
Auto sportiak;
Auto nakladiak;
Projekt most;
Projekt vysielac;
```

Údaje

Napríklad premenné vo funkcii `main`

Údaje nie sú nenastavené!



Príklad 1.1.b – stavebná firma – riešenie pre C++

Príklad 1.1.b – stavebná firma – riešenie pre C++

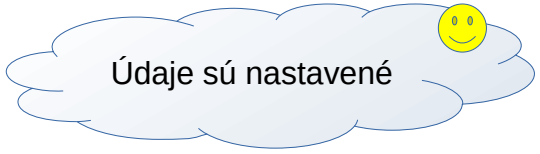

```
Zamestnanec jan {
    .meno = "Jan Hlavny",
    .adresa = "Bratislava, Podnikatelska 10",
    .nadriadeny = nullptr
};
Zamestnanec peter {
    .meno = "Peter Plavec",
    .adresa = "Bratislava, Mokra 20",
    .nadriadeny = &jan
};
Zamestnanec juraj {
    .meno = "Juraj Horsky",
    .adresa = "Bratislava, Tatrancov 30",
    .nadriadeny = &jan
};
```

Príklad 1.1.b – stavebná firma – riešenie pre C++


```
Zamestnanec jan {
   .meno = "Jan Hlavny",
   .adresa = "Bratislava, Podnikatelska 10",
   .nadriadeny = nullptr
};
Zamestnanec peter {
   .meno = "Peter Plavec",
   .adresa = "Bratislava, Mokra 20",
   .nadriadeny = &jan
};
Zamestnanec juraj {
   .meno = "Juraj Horsky",
   .adresa = "Bratislava, Tatrancov 30",
   .nadriadeny = &jan
};
```

```
Projekt most {
   .nazov = "Most cez Dunaj",
   .popis = "Oprava .....",
   .zamestnanci = {&jan, &peter}
};
Projekt vysielac {
   .nazov = "Internet pre studentov",
   .popis = "Instalacia 5G .....",
   .zamestnanci = {&jan, &juraj}
};
```

Príklad 1.1.b – stavebná firma – riešenie pre C++




```
Zamestnanec jan {
    .meno = "Jan Hlavny",
    .adresa = "Bratislava, Podnikatelska 10",
    .nadriadeny = nullptr
};
Zamestnanec peter {
    .meno = "Peter Plavec",
    .adresa = "Bratislava, Mokra 20",
    .nadriadeny = &jan
};
Zamestnanec juraj {
    .meno = "Juraj Horsky",
    .adresa = "Bratislava, Tatrancov 30",
    .nadriadeny = &jan
};
```

```
Projekt most {
    .nazov = "Most cez Dunaj",
    .popis = "Oprava .....",
    .zamestnanci = {&jan, &peter}
};
Projekt vysielac {
    .nazov = "Internet pre studentov",
    .popis = "Instalacia 5G .....",
    .zamestnanci = {&jan, &juraj}
};
```

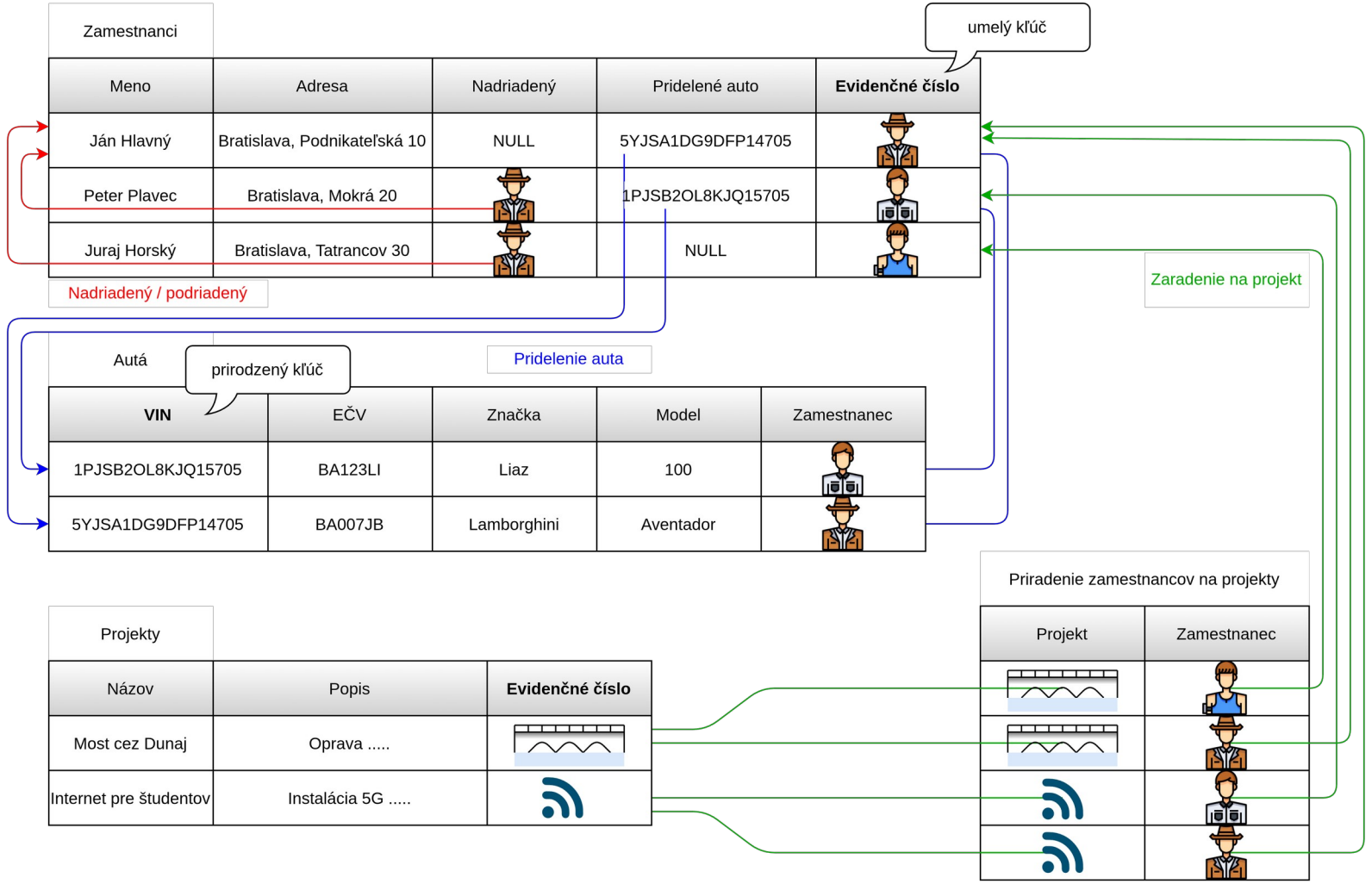
```
Auto sportiak {
    .vin = "5YJSA1DG9DFP14705",
    .ecv = "BA007JB",
    .znacka = "Lamborghini",
    .model = "Aventador",
    .zamestnanec = &jan
};
Auto nakladiak {
    .vin = "1PJSB2OL8KJQ15705",
    .ecv = "BA123LI",
    .znacka = "Liaz",
    .model = "100",
    .zamestnanec = &peter
};
```

Príklad 1.1.c – stavebná firma

c) Naznačte môžnosti implementácie v tabuľkovom editore.

animácia

Príklad 1.1.c – stavebná firma – tabuľkový editor



Príklad 1.1.c – stavebná firma – tabuľkový edito

animácia

V tabuľkovom editore určenom pre kancelárske účely môžeme do bunky v tabuľke napísať ľubovolný text. V relačných databázach, ktoré údaje uchovávajú v tabuľkách, ale štandardné platí pravidlo, že do bunky môžeme zadať len jednu hodnotu. V implementáciach relačných databáz existujú aj rozšírenia, kde to neplatí.

Zamestnanci

umelý kľúč

Meno	Adresa	Nadriadený	Pridelené auto	**Evidenčné číslo**
Ján Hlavný	Bratislava, Podnikateľská 10	NULL	5YJSA1DG9DFP14705	
Peter Plavec	Bratislava, Mokrá 20		1PJSB2OL8KJQ15705	
Juraj Horský	Bratislava, Tatrancov 30		NULL	

Nadriadený / podriadený

Zaradenie na projekt

Autá

prirodzený kľúč

Pridelenie auta

VIN	EČV	Značka	Model	Zamestnanec
1PJSB2OL8KJQ15705	BA123LI	Liaz	100	
5YJSA1DG9DFP14705	BA007JB	Lamborghini	Aventador	

Projekty

Názov	Popis	**Evidenčné číslo**
Most cez Dunaj	Oprava	
Internet pre študentov	Instalácia 5G	

Priradenie zamestnancov na projekty

Projekt	Zamestnanec

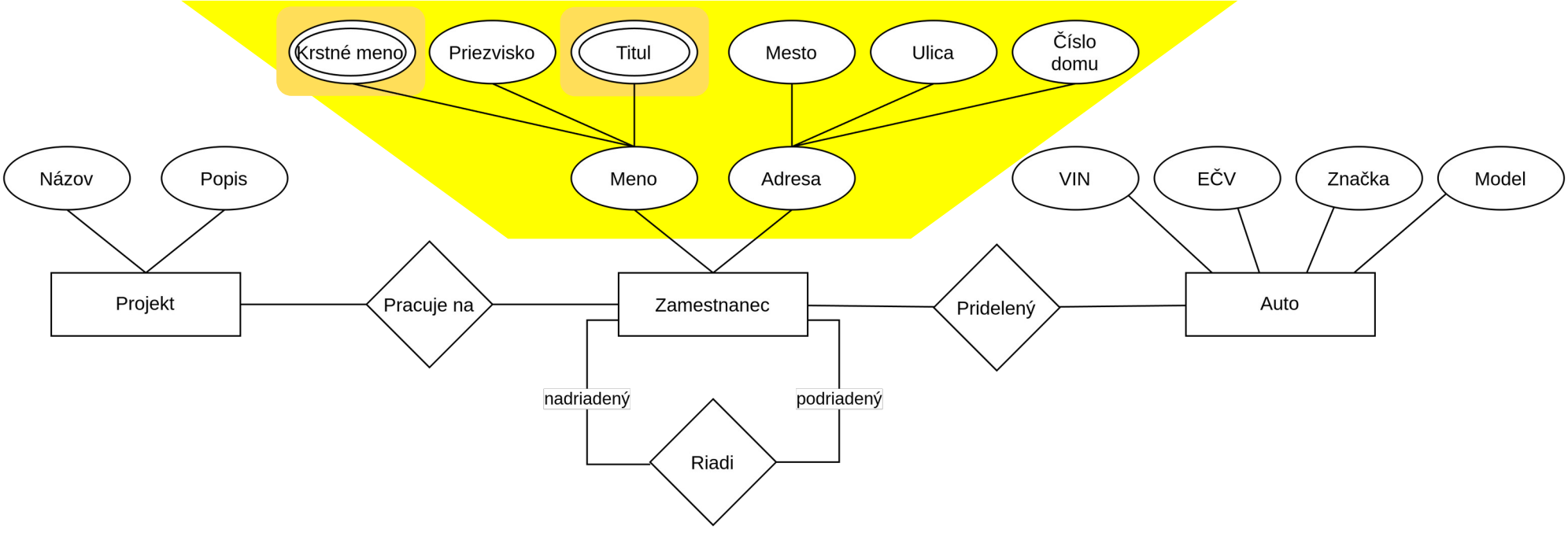
Príklad 1.1.d – stavebná firma

d) Rozložte zložené atribúty.

Identifikujte viachodnotové atribúty.

Príklad 1.1.d – stavebná firma

- Meno a Adresa – môžu byť **zloženými atribútmi**
- Krstné meno a Titul – môžu byť **viachodnotovými atribútmi**



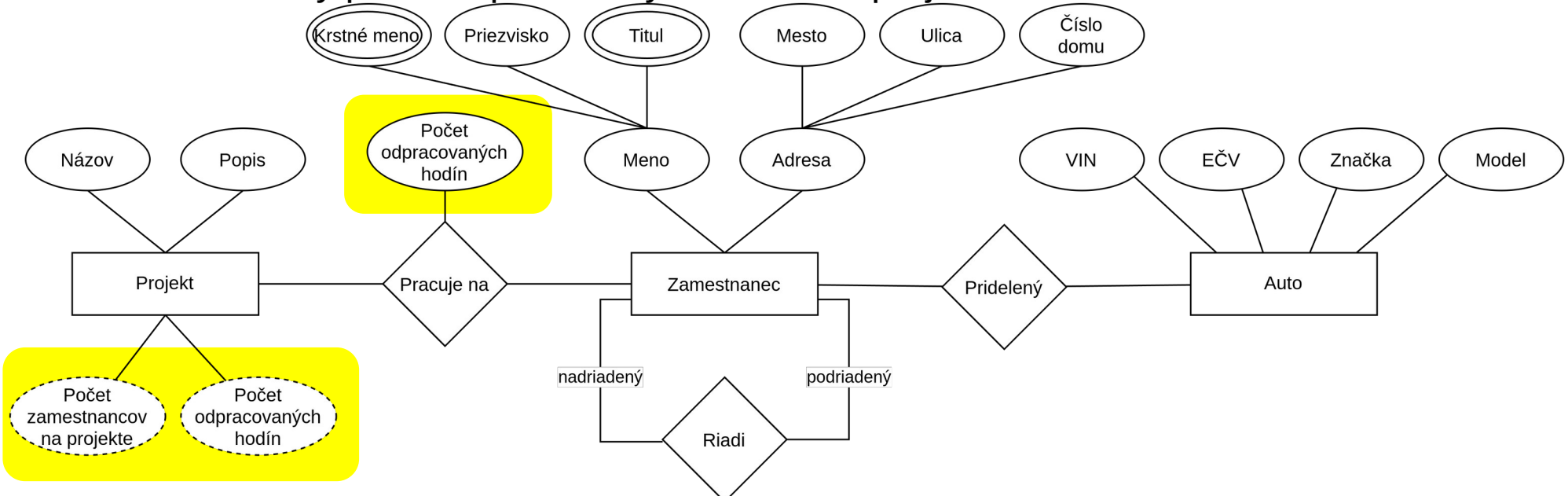
Príklad 1.1.e – stavebná firma

e) Doplňte evidenciu o:

- počet odpracovaných hodín zamestnantcov pre jednotlivé projekty,
- počet zamestnancov pracujúcich na projekte
- a celkový počet hodín odpracovaných na každom projekte.

Príklad 1.1.e – stavebná firma

- **Atribút vzťahu**
  - Odpracované hodiny zamestnanca na projekte
- **Odvodené atribúty**
  - Počet zamestnancov pracujúcich na projekte
  - a celkový počet odpracovaných hodín na projekte



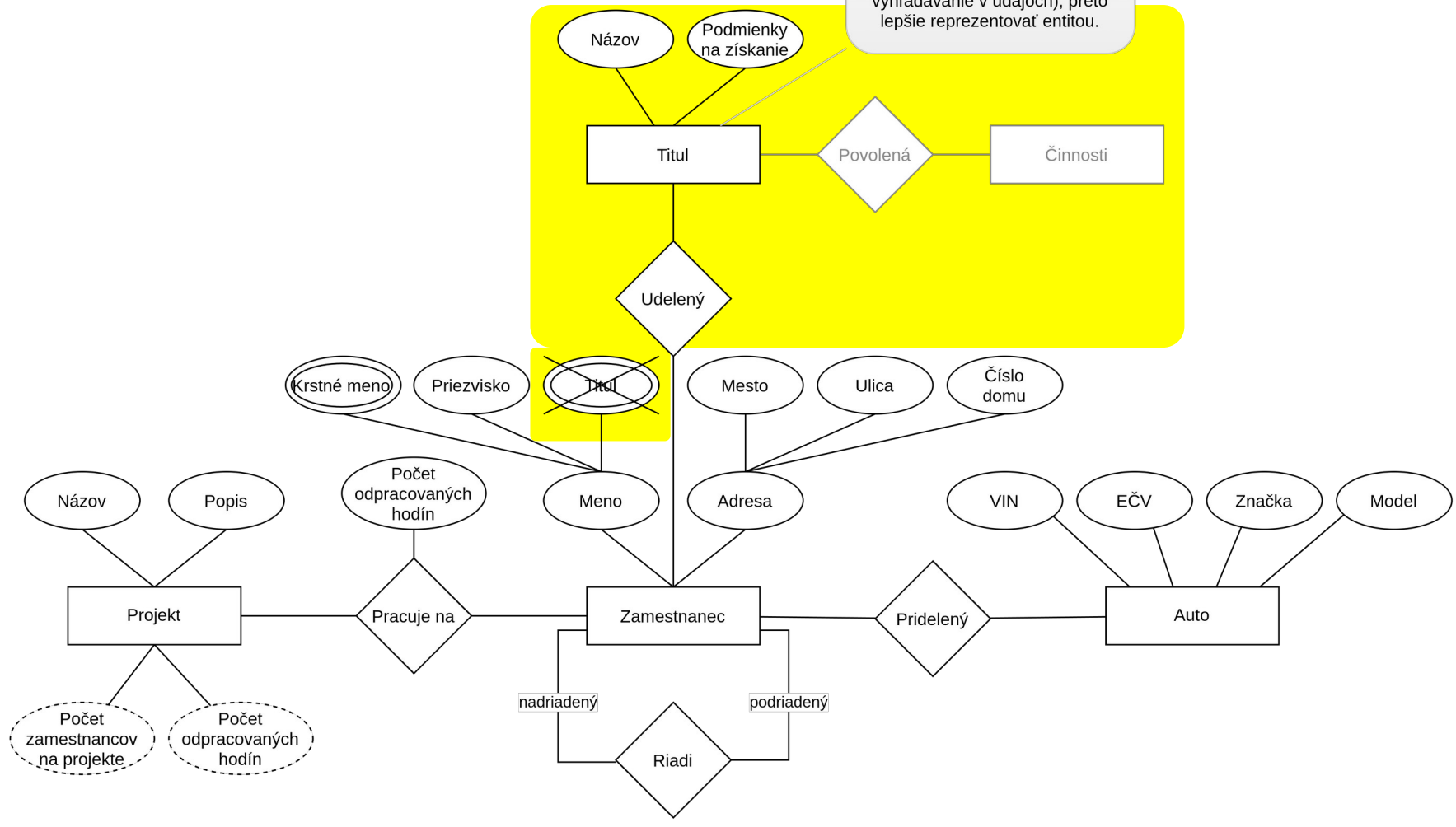
Príklad 1.1.f – stavebná firma

f) Kedy použiť entitu a kedy atribút?

Príklad 1.1.f – stavebná firma

animácia

Titul v súkromnej firme nemusí byť (pre vyhľadávanie v údajoch) dôležitý. Preto môže stačiť ako atribút. Na univerzite je akademický titul dôležitý (pre vyhľadávanie v údajoch), preto lepšie reprezentovať entitou.

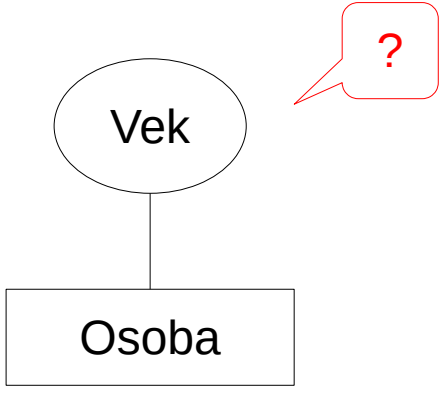


Príklad 1.2 – Vek osoby

Reprezentujte osobu a jej vek

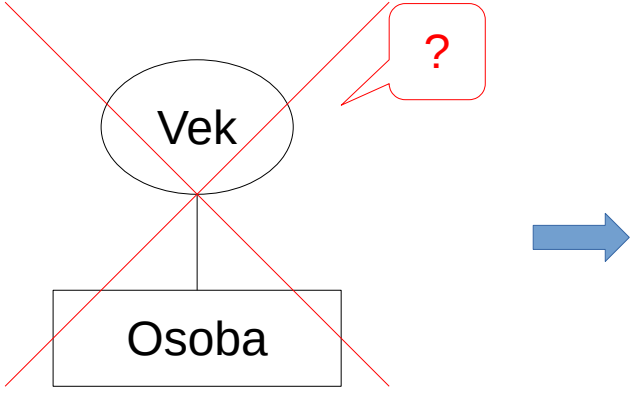
Príklad 1.2 – Vek osoby

Reprezentujte osobu a jej vek



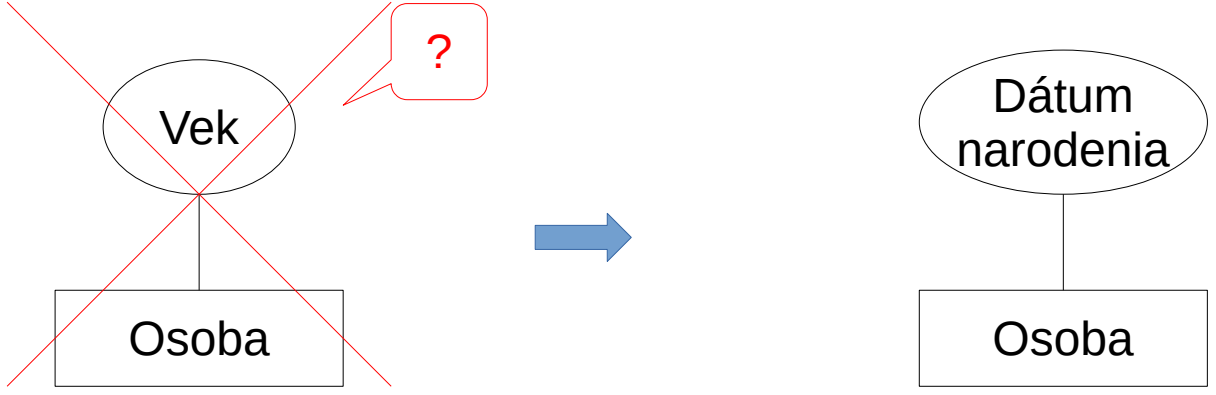
Príklad 1.2 – Vek osoby

Reprezentujte osobu a jej vek



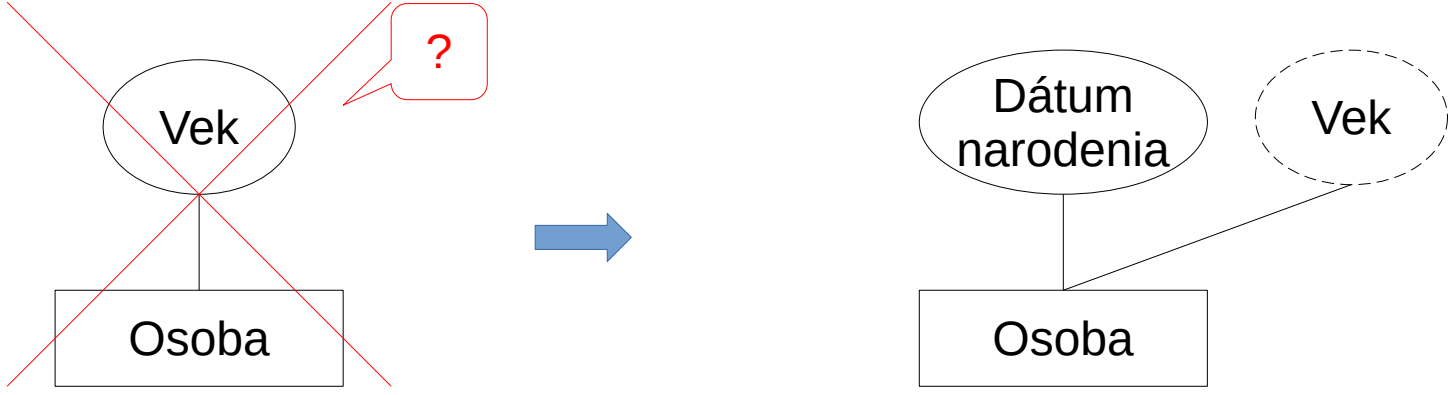
Príklad 1.2 – Vek osoby

Reprezentujte osobu a jej vek



Príklad 1.2 – Vek osoby

Reprezentujte osobu a jej vek



Príklad 1.3 – nedefinovaná hodnota atribútu

Hodnota atribútu nemusí byť v databáze vždy zaznamenaná.

Príklad 1.3 – nedefinovaná hodnota atribútu

Hodnota atribútu nemusí byť v databáze vždy zaznamenaná.

a) Uveďte príklad:

- neznámej hodnoty atribútu
- nedefinovanej hodnoty atribútu

Príklad 1.3 – nedefinovaná hodnota atribútu

Hodnota atribútu nemusí byť v databáze vždy zaznamenaná.

a) Uveďte príklad:

- neznámej hodnoty atribútu
- nedefinovanej hodnoty atribútu

Neznáma hodnota atribútu:

- Hmotnosť výrobku (existuje ale nemusíme ju poznať)

Príklad 1.3 – nedefinovaná hodnota atribútu

Hodnota atribútu nemusí byť v databáze vždy zaznamenaná.

a) Uveďte príklad:

- neznámej hodnoty atribútu
- nedefinovanej hodnoty atribútu

Neznáma hodnota atribútu:

- Hmotnosť výrobku (existuje ale nemusíme ju poznať)

Nedefinovaná hodnota atribútu:

- Ulica v adrese, ak obec nemá ulice (neexistuje)

Príklad 1.3 – nedefinovaná hodnota atribútu

Hodnota atribútu nemusí byť v databáze vždy zaznamenaná.

a) Uveďte príklad:

- neznámej hodnoty atribútu
- nedefinovanej hodnoty atribútu

Neznáma hodnota atribútu:

- Hmotnosť výrobku (existuje ale nemusíme ju poznať)

Nedefinovaná hodnota atribútu:

- Ulica v adrese, ak obec nemá ulice (neexistuje)

b) Ako označujeme neznámu alebo nedefinovanú hodnotu atribútu?

Príklad 1.3 – nedefinovaná hodnota atribútu

Hodnota atribútu nemusí byť v databáze vždy zaznamenaná.

a) Uveďte príklad:

- neznámej hodnoty atribútu
- nedefinovanej hodnoty atribútu

Neznáma hodnota atribútu:

- Hmotnosť výrobku (existuje ale nemusíme ju poznať)

Nedefinovaná hodnota atribútu:

- Ulica v adrese, ak obec nemá ulice (neexistuje)

b) Ako označujeme neznámu alebo nedefinovanú hodnotu atribútu?

Neznáma alebo nedefinovaná hodnota atribútu je označovaná **NULL**.

Príklad 1.4 – softvérová firma

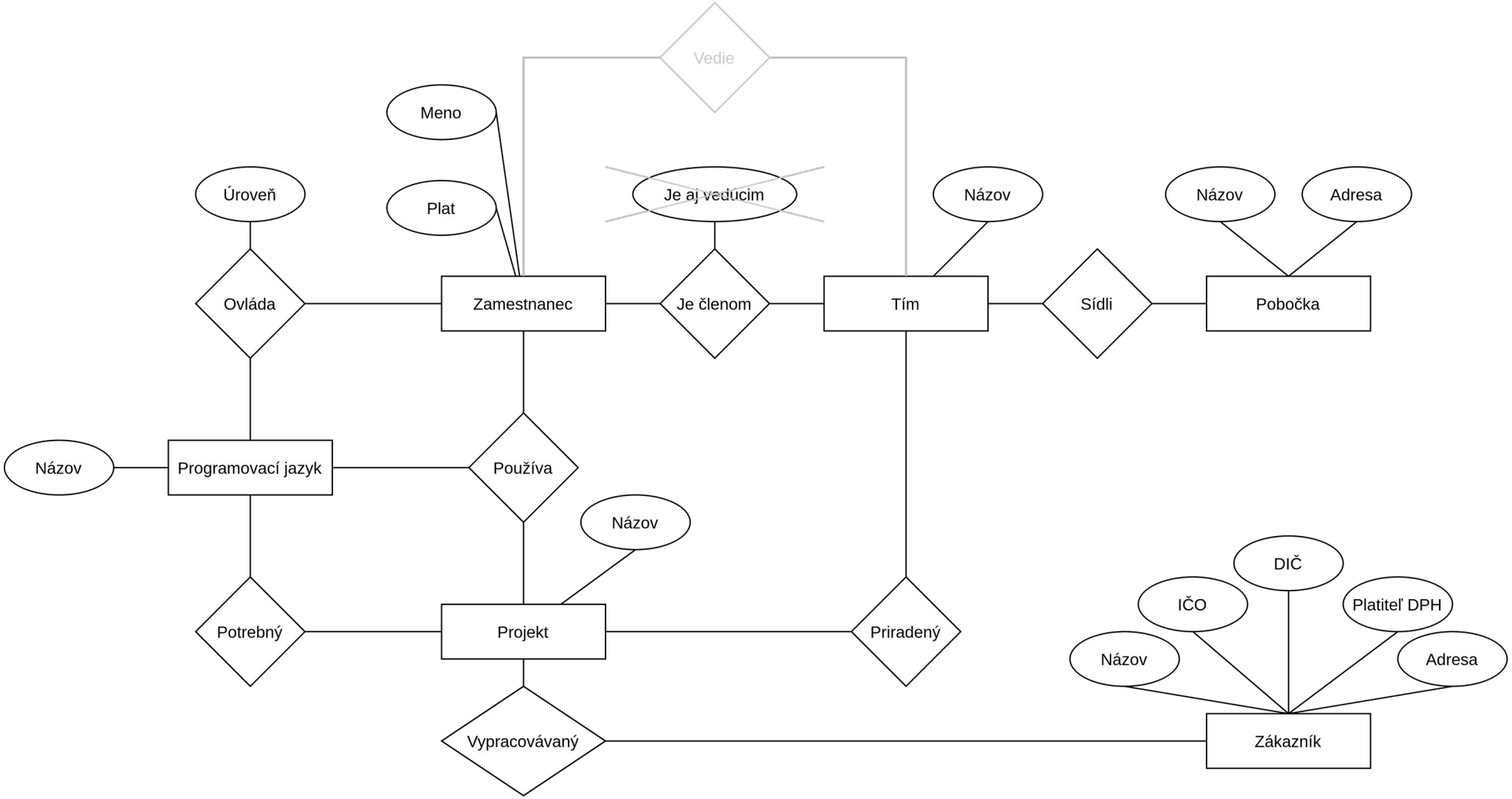
Navrhnite konceptuálny model databázy firmy. Databáza bude obsahovať informácie o zamestnancoch, ich zaradení do tímov a informácie o vedúcich tímov. Firma má viacero pobočiek. Každý tím pracuje na jednej z pobočiek. Každý tím môže pracovať na viacerých projektoch. Viaceré projekty môžu byť zadané tým istým zákazníkom. Databáza obsahuje informáciu o úrovni ovládania programovacích jazykov zamestnancami. Každému projektu sú pridelené aj programovacie jazyky, ktoré sú pre jeho realizáciu potrebné. Tiež obsahuje informáciu o tom, ktorý zamestnanec používa ktoré jazyky na ktorých projektoch.

Príklad 1.4 – softvérová firma

Navrhnite konceptuálny model databázy firmy. Databáza bude obsahovať informácie o zamestnancoch, ich zaradení do tímov a informácie o vedúcich tímov. Firma má viacero pobočiek. Každý tím pracuje na jednej z pobočiek. Každý tím môže pracovať na viacerých projektoch. Viaceré projekty môžu byť zadané tým istým zákazníkom. Databáza obsahuje informáciu o úrovni ovládania programovacích jazykov zamestnancami. Každému projektu sú pridelené aj programovacie jazyky, ktoré sú pre jeho realizáciu potrebné. Tiež obsahuje informáciu o tom, ktorý zamestnanec používa ktoré jazyky na ktorých projektoch.

Často platí:
podstatné meno → typ entity alebo atribút
sloveso → typ vzťahu

Príklad 1.4 – softvérová firma



Príklad 1.4 – softvérová firma

Množiny entít

Množina zamestnancov = { , , , , , , , , , , , , }

Množina programoacích jazykov = { , , , , , , , , TS }

Množina tímov = { , , , , }

Množina projektov = { , , , , }

Na slajde nie sú všetky množiny definované modelom

Príklad obsahu databázy

Množiny vzťahov

Množina členstiev v tímoch = {
(,),
(,),
(,),
(,),
(,),
(,),
(,),
(,),
(,),
(,),
}

binárny vzťah - množina dvojic

Zamestnanec ovláda = {
(,),
(,),
(,),
(,),
(,),
(,),
(,),
(,),
(,),
(,),
(,),
(,),
(,),
(,),
atď.
}

binárny vzťah - množina dvojic

Zamestnanec používa na projekte = {
(, ,),
(, ,),
(, ,),
(, ,),
(, ,),
(, ,),
(, ,),
(, ,),
(, ,),
(, ,),
(, ,),
(, ,),
atď.
}

ternárny vzťah - množina trojíc

Bonusová úloha

- Na web-stránke cvičení
- Odovzdajte na začiatku 2. cvičenia

Zdroje obrázkov

- Aplikácia https://app.diagrams.net
- https://medium.com/swlh/10-things-every-programmer-should-know-26ba37cfcaf4
- https://emojiterra.com/package/
- https://stock.adobe.com/search?k=indian+village
- https://www.shutterstock.com/search/student-writing-cartoon