Normalizace

Jiří Zacpal

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

KMI/DATAB Databáze

Příklad

cislo_projektu	jmeno_projektu	cislo_zamestnance	jmeno_zamestnance	pozice	hodinova_mzda	hodiny	vyplata
15	Evergreen	103	Alice Nováková	Elektrikář	200 Kč	23,8	4 760 Kč
15	Evergreen	101	František Bláha	Databázový návrhář	350 Kč	19,4	6 790 Kč
15	Evergreen	105	Jitka Smutná	Databázový návrhář	350 Kč	35,7	12 495 Kč
15	Evergreen	106	Václav Krása	Programátor	150 Kč	12,6	1 890 Kč
15	Evergreen	102	David Skoupil	Analytik	300 Kč	23,8	7 140 Kč
18	Amber	114	Anna Jánská	Návrhář aplikace	120 Kč	24,6	2 952 Kč
18	Amber	118	Jakub Frommer	Brigádník	70 Kč	45,3	3 171 Kč
18	Amber	104	Jana Rámová	Analytik	300 Kč	32,4	9 720 Kč
18	Amber	112	Darina Sladká	Analytik DSS	225 Kč	44	9 900 Kč
22	Rosmary	105	Jitka Smutná	Databázový návrhář	350 Kč	64,7	22 645 Kč
22	Rosmary	104	Jana Rámová	Analytik	300 Kč	48,4	14 520 Kč
22	Rosmary	113	Daniel John	Návrhář aplikace	120 Kč	23,6	2 832 Kč
22	Rosmary	111	Jan Václav	Účetní	130 Kč	22	2 860 Kč
22	Rosmary	106	Václav Krása	Programátor	150 Kč	12,8	1 920 Kč
25	Starflight	107	Marie Aloisová	Programátor	150 Kč	24,6	3 690 Kč
25	Starflight	115	Tomáš Vidím	Analytik	300 Kč	45,8	13 740 Kč
25	Starflight	101	František Bláha	Databázový návrhář	350 Kč	56,3	19 705 Kč
25	Starflight	114	Anna Jánská	Návrhář aplikace	120 Kč	33,1	3 972 Kč
25	Starflight	108	Roman Koubský	Analytik	300 Kč	23,6	7 080 Kč
25	Starflight	118	Jakub Frommer	Brigádník	70 Kč	30,5	2 135 Kč
25	Starflight	112	Darina Sladká	Analytik DSS	225 Kč	41,4	9 315 Kč

Normalizace relačních schémat

- **Nevhodný návrh** relace signalizuje výskyt opakujících se položek v datech, ale také pozorujeme následující potíže :
  - **redundance** - *opakuje se jméno projektu, jméno zaměstnance, …,*
  - nebezpečí vzniku **nekonzistence** při modifikacích jako důsledek redundance - *změníme jméno zaměstnance jen v některých záznamech*
  - **anomálie při vkládání záznamů** - *nemůžeme vložit projekt bez zaměstnance , který jej řeší, neboť by nebyly obsazeny klíčové atributy,*
  - **anomálie při vypouštění záznamů** - *přestanou-li řešit úlohy všichni pracovníci na stejné pozici, ztratíme informaci o platu dané pozice*.
- Problém vyřešíme **dekompozicí** relace za pomoci funkčních závislostí.

Příklad

```
CREATE TABLE prace_na_projektu
(
cislo_projektu INTEGER,
jmeno_projektu VARCHAR(20),
cislo_zamestnance INTEGER,
jmeno_zamestnance VARCHAR(50),
prijmeni VARCHAR(30),
pozice VARCHAR(20),
hodinova_mzda INTEGER,
hodiny numeric(5),
email VARCHAR(50),
vyplata INTEGER
);
```

Příklad

- Smažeme vyplata:

```
ALTER TABLE prace_na_projektu
DROP COLUMN vyplata;
```

1. normální forma

- Relace je 1. NF, jestliže:
  - všechny atributy jsou atomické, tj. dále již nedělitelné,
  - relace má primární klíč.
- Převod:
  1. Nahraďte každý skupinový atribut atomickými atributy.
  2. Určete primární klíč.

Příklad

- Vytvoříme primární klíč:

```
ALTER TABLE prace_na_projektu
ADD CONSTRAINT projekt_zamestnanec_pkey PRIMARY KEY
(cislo_projektu,cislo_zamestnance);
```

Příklad

- Vytvoříme atributy pro jméno a příjmení:

```
ALTER TABLE prace_na_projektu
ADD COLUMN jmeno VARCHAR(20);

ALTER TABLE prace_na_projektu
ADD COLUMN prijmeni VARCHAR(30);
```

- Rozdělíme jmeno_zamestnance na jméno a příjmení:

```
UPDATE prace_na_projektu SET jmeno=left(jmeno_zamestnance,
position(' ' in jmeno_zamestnance)-1),
prijmeni=right(jmeno_zamestnance,char_length(jmeno_zamestnance
)-position(' ' in jmeno_zamestnance));
```

Příklad

- Smažeme jmeno_zamestnance:

```
ALTER TABLE prace_na_projektu
DROP COLUMN jmeno_zamestnance;
```

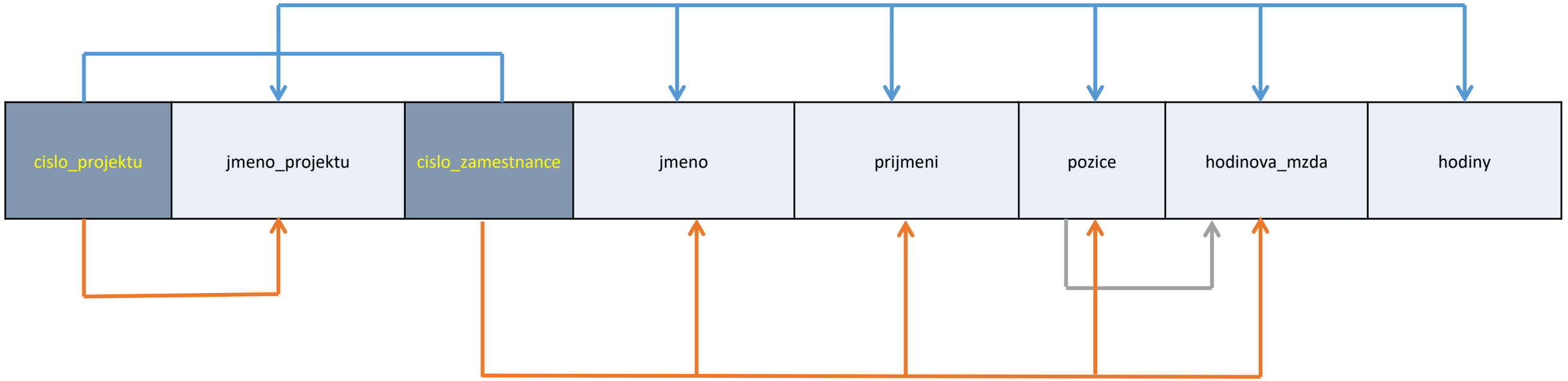
Funkční závislost

- funkční závislost je definována mezi dvěma podmnožinami **atributů** v rámci jedné relace
- nechť X, Y jsou podmnožiny množiny jmen atributů $\{A_1, \ldots, A_n\}$ relace R.
- X určuje Y (tj. Y je funkčně závislý na X), jestliže všechny záznamy (řádky) v tabulce, které mají stejnou hodnotu atributu X má stejnou hodnotu atributu Y.
- píšeme $\boldsymbol{X \rightarrow Y}$
- je-li $Y \subset Y$ říkáme, že závislost $X \rightarrow Y$ je triviální
- příklad:

  cislo_zamestnance →prijmeni (naopak to neplatí)

  cislo_projektu→jmeno_projektu

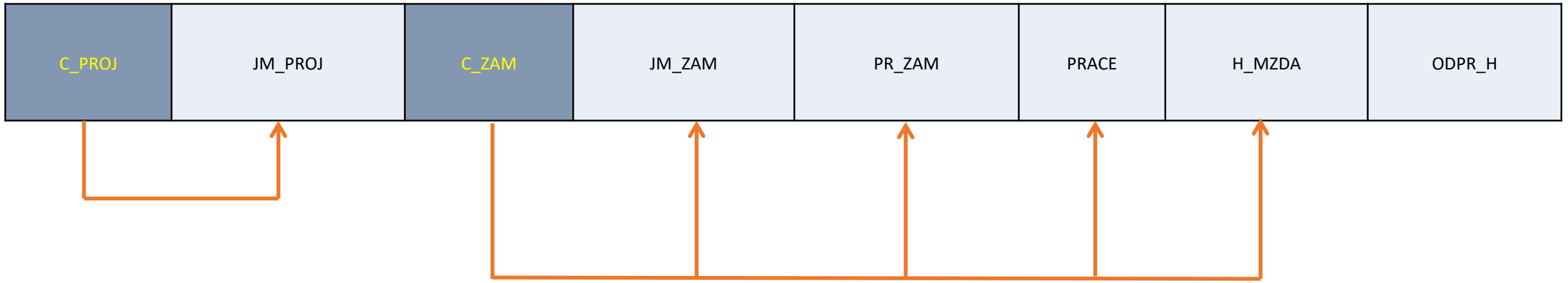
Funkční závislosti



2. normální forma

- Relace je v 2. NF, jestliže:
  - je v 1. NF,
  - neobsahuje částečné závislosti (atribut, který není primárním klíčem závislý na celém primárním klíči).
- Převod:
  1. Určete všechny částečné závislosti.
  2. Pro každou částečnou závislost vytvořte zvláštní relaci.

Částečné závislosti



Příklad

- Vytvoříme relaci projekt:

```
CREATE TABLE projekt
(
cislo_projektu INTEGER,
jmeno_projektu VARCHAR(20),
CONSTRAINT cislo_pkey PRIMARY KEY (cislo_projektu)
);
```

- Vložíme n-tice:

```
INSERT INTO projekt SELECT DISTINCT
cislo_projektu,jmeno_projektu FROM prace_na_projektu;
```

- Odstraníme atribut jmeno_projektu:

```
ALTER TABLE prace_na_projektu
DROP COLUMN jmeno_projektu;
```

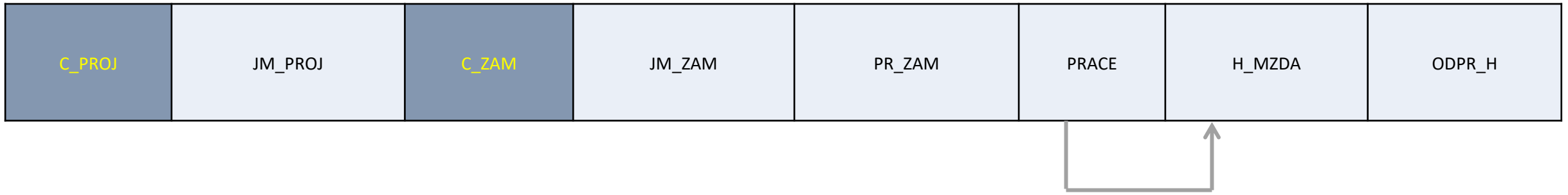
Úkol

Převeďte relaci do 2. NF.

3. normální forma

- Relace je v 3. NF, jestliže:
  - je v 2. NF,
  - neobsahuje transitivní závislosti (nejsou zde závislosti mezi neklíčovými atributy).
- Převod:
  1. Určete všechny tranzitivní závislosti.
  2. Pro každou tranzitivní závislost vytvořte zvláštní relaci.

Tranzitivní závislosti



Úkol

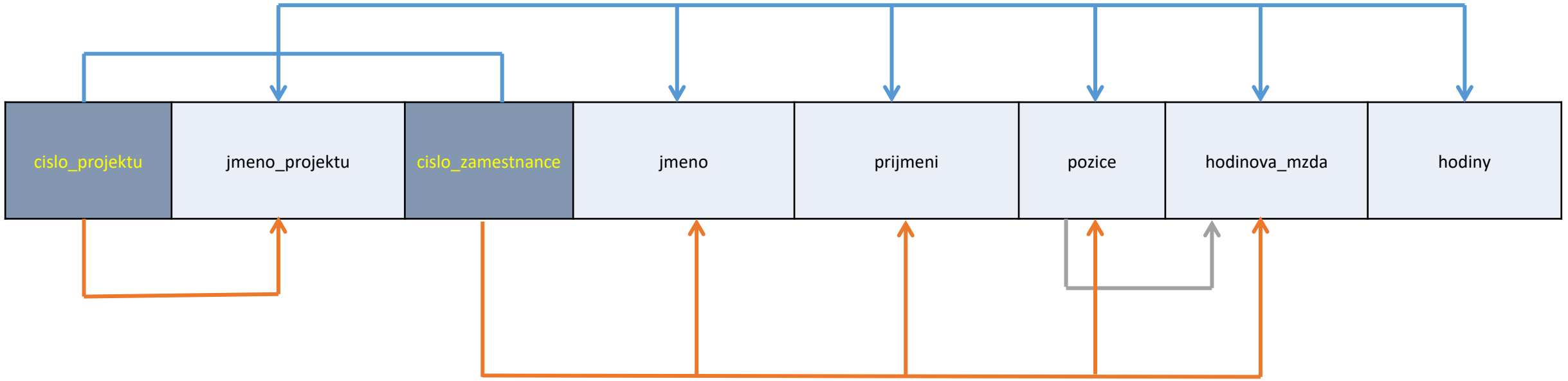
Převeďte relaci do 3. NF.

Boyceho–Coddova normální forma

- Relace je v BCNF, jestliže:
  - pro každou netriviální funkční závislost X -> Y relační proměnné R platí, že X je nadklíč (superklíč) R.

- Převod:
  1. Určete všechny tyto závislosti.
  2. Pro každou závislost vytvořte zvláštní relaci.

Funkční závislosti



Úkol

Nastavte pro všechny relace referenční integritu.

Bodovaný úkol

Normalizujte relaci knihovna ze souboru

datab_08_normalizace_ukol.sql

do BCNF nebo 3. normální formy.