KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI
www.inf.upol.cz

Jan Laštovička
https://www.inf.upol.cz/lide/jan-lastovicka
jan.lastovicka@upol.cz
17. listopadu 12, 771 46 Olomouc

Databáze ◇ poznámky k přednášce

5. Návrh databáze

verze z 22. října 2024

1 Primární klíč

Je dáno relační schéma S a podmnožina $K \subseteq S$.

Řekneme, že dvě n-tice t_1, t_2 nad S se **shodují** v K, pokud $t_1(K) = t_2(K)$. Například n-tice:

- $\{\langle \texttt{title}, \texttt{'The Avengers'}\rangle, \langle \texttt{year}, 2012\rangle, \langle \texttt{length}, 143\rangle\}$,
- $\{\langle \texttt{title}, \texttt{'The Avengers'}\rangle, \langle \texttt{year}, 1998\rangle, \langle \texttt{length}, 89\rangle\}$

nad $\{\texttt{title}, \texttt{year}, \texttt{length}\}$ se shodují v $\{\texttt{title}\}$, ale neshodují v $\{\texttt{title}, \texttt{year}\}$.

Vezměme relaci R nad S. Schéma K nezveme **nadklíčem** relace R, pokud pro každé dvě n-tice $t_1, t_2 \in R$ platí, že jestliže se t_1 shoduje s t_2 v K, pak $t_1 = t_2$. Schéma S je vždy nadklíčem relace R. Například v relaci R:

```
       title       | year | length
-------------------+------+--------
 The Matrix        | 1999 |    136
 The Avengers      | 2012 |    143
 The Avengers      | 1998 |     89
 American Beauty   | 1999 |    122
 Listen to Britain | 1942 |     20
 Khaneh siah ast   | 1963 |     20
(6 rows)
```

jsou $\{\texttt{title}, \texttt{year}\}$ a $\{\texttt{title}, \texttt{year}, \texttt{length}\}$ nadklíče, ale například $\{\texttt{title}\}$ nadklíčem není, protože n-tice:

- $t_1 = \{\langle \texttt{title}, \texttt{'The Avengers'}\rangle, \langle \texttt{year}, 2012\rangle, \langle \texttt{length}, 143\rangle\}$,
- $t_2 = \{\langle \texttt{title}, \texttt{'The Avengers'}\rangle, \langle \texttt{year}, 1998\rangle, \langle \texttt{length}, 89\rangle\}$

z R se shodují v $\{\texttt{title}\}$, ale $t_1 \neq t_2$.

Pro každou základní relaci ***relation*** uvažujeme množinu možných hodnot. Možné hodnoty jsou relace, které mohou být hodnotou relační proměnné ***relation*** — v současnosti nebo v budoucnosti mají charakteristickou vlastnost základní

relace *relation*. Pokud je K nadklíčem každé možné hodnoty základní relace *relation*, pak říkáme, že K je **nadklíčem** základní relace *relation*. Tedy například {title, year} je nadklíčem základní relace movie nad {title, year, length} dané: „Film title vydaný roku year je dlouhý length minut.“, kde předpokládáme, že každý film lze jednoznačně určit názvem a rokem vydání.

Nadklíč K základní relace *relation* se nazývá jejím **kandidátním klíčem**, pokud žádná vlastní podmnožina K není nadklíčem základní relace *relation*. Například u základní relace movie je {title, year} kandidátní klíč, ale {title, year, length} kandidátním klíčem není.

Pokud napíšeme, že $y_1, \ldots, y_n$ je kandidátním klíčem základní relace *relation*, myslíme tím, že $\{y_1, \ldots, y_n\}$ je kandidátním klíčem základní relace *relation* a navíc udáváme pořadí atributů $y_1, \ldots, y_n$ v kandidátním klíči.

Základní relace může mít více kandidátních klíčů. Například základní relace movie nad {movie_id, title, year, length} daná vlastností: „Vlastním film movie_id s názvem title vytvořený roku year, který má délku length minut.“ má dva kandidátní klíče: {movie_id} a {title, year}.

Pokud nevíte proč, podívejte se na tuto možnou hodnotu:

```
 movie_id |       title       | year | length
----------+-------------------+------+--------
        1 | The Matrix        | 1999 |    136
        2 | The Avengers      | 2012 |    143
        3 | The Avengers      | 1998 |     89
        4 | American Beauty   | 1999 |    122
        5 | Listen to Britain | 1942 |     20
        6 | Khaneh siah ast   | 1963 |     20
(6 rows)
```

Jeden z kandidátních klíčů základní relace vybereme a prohlásíme ho za **primární klíč**. Zbylé klíče se nazývají **alternativní klíče**. Například {movie_id} může být zvolen primárním klíčem základní relace movie a {title, year} klíčem alternativním.

2 Omezení základní relace

Některé hodnoty základní relace můžeme vyloučit tak, že u základní relace uvedeme **omezení**. Omezení základní relace můžeme uvést při její deklaraci:

```
CREATE TABLE relation (
  attributes,
  constraints
);
```

kde *constraints* jsou omezení základní relace oddělená čárkou.

Omezení základní relace:

```
PRIMARY KEY ( y1, ..., yn )
```

deklaruje, že $y_1, \ldots, y_n$ je primárním klíčem základní relace. Omezení základní relace:

```
UNIQUE ( y1, ..., yn )
```

deklaruje, že $y_1, \ldots, y_n$ je alternativním klíčem základní relace. Například:

```
CREATE TABLE movie (
    movie_id integer,
    title text,
    year integer,
    length integer,
    PRIMARY KEY ( movie_id ),
    UNIQUE ( title, year )
);
```

deklaruje, že {movie_id} je primárním a {title, year} alternativním klíčem základní relace movie. Do proměnné:

```
# TABLE movie;

 movie_id |    title     | year | length
----------+--------------+------+--------
        1 | The Matrix   | 1999 |    136
        2 | The Avengers | 2012 |    143
(2 rows)
```

nemůžeme přidat:

```
# INSERT INTO movie VALUES
       ( 2, 'The Avengers', 1998, 89 );

ERROR:  duplicate key value violates unique constraint "movie_pkey"
DETAIL:  Key (movie_id)=(2) already exists.
```

ani:

```
# INSERT INTO movie VALUES
       ( 3, 'The Avengers', 2012, 89 );

ERROR:  duplicate key value violates unique constraint "movie_title_year_key"
DETAIL:  Key (title, year)=(The Avengers, 2012) already exists.
```

Pokud se omezení týká jediného atributu, můžeme jej uvést rovnou v deklaraci atributu:

```
y scalar_type constraints
```

Omezení atributu píšeme za sebe a oddělujeme mezerou. Omezení atributu `PRIMARY KEY` prohlásí {y} za primární klíč a `UNIQUE` za alternativní klíč. Například:

```
CREATE TABLE movie (
    movie_id integer PRIMARY KEY,
    title text,
    year integer,
    length integer,
    UNIQUE ( title, year )
);
```

Jak bylo dříve zmíněno, řádek tabulky SQL pro zadaný sloupec nemusí obsahovat hodnotu. Říkáme pak, že obsahuje **null** hodnotu. V SQL ji zapisujeme `NULL`. Připomeňme si, že každá komponenta n-tice v relaci má vždy hodnotu. Omezení atributu `NOT NULL` zaručí, že ve sloupci nikdy nebude null hodnota. Například:

```
CREATE TABLE movie (
    movie_id integer PRIMARY KEY,
    title text NOT NULL,
    year integer NOT NULL,
    length integer,
    UNIQUE ( title, year )
);
```

Nelze přidat null hodnotu do sloupce `title`:

```
# INSERT INTO movie VALUES
        ( 1, NULL, 1999, 136 ),
        ( 2, 'The Avengers', 2012, 143 );

ERROR:  null value in column "title" of relation "movie"
        violates not-null constraint
DETAIL:  Failing row contains (1, null, 1999, 136).
```

V žádném sloupci uvedeném v primárním klíči nesmí být null hodnoty. Proto nelze:

```
# INSERT INTO movie VALUES
       ( NULL, 'The Avengers', 1998, 89 );

ERROR:  null value in column "movie_id" of relation "movie"
        violates not-null constraint
```

Omezení PRIMARY KEY je ekvivalentní kombinaci omezení NOT NULL a UNIQUE.

Pokud základní relace má deklarovaný primární klíč a každý sloupec má omezení, že nesmí obsahovat null hodnoty, pak je zaručeno, že její hodnota bude relace. Například:

```
CREATE TABLE movie (
    movie_id integer PRIMARY KEY,
    title text NOT NULL,
    year integer NOT NULL,
    length integer NOT NULL,
    UNIQUE ( title, year )
);
```

3 Cizí klíč

Vezměme dvě relace R_1 nad schématem S_1 a R_2 nad schématem S_2. Dále vezměme relační schémata $K_1 \subseteq S_1$ a $K_2 \subseteq S_2$ a přejmenování atributů $h\colon K_1 \to K_2$. Schéma K_1 nazýváme **cizím klíčem** relace R_1 na K_2 relace R_2 podle h, pokud pro každou n-tici $t_1 \in R_1$ existuje právě jedna n-tice $t_2 \in R_2$ taková, že $\rho_h(t_1(K_1)) = t_2(K_2)$.

Například pro relaci R_1:

```
     actor_name     | movie_title | movie_year
--------------------+-------------+------------
 Keano Reeves       | The Matrix  |       1999
 Keano Reeves       | Dracula     |       1992
 Laurence Fishburne | The Matrix  |       1999
 Gary Oldman        | Dracula     |       1992
(4 rows)
```

a relaci R_2:

```
   title    | year
------------+------
 The Matrix | 1999
 Dracula    | 1992
(2 rows)
```

je $\{\texttt{movie\_title}, \texttt{movie\_year}\}$ cizím klíčem relace R_1 na $\{\texttt{title}, \texttt{year}\}$ relace R_2 podle $\{\langle \texttt{movie\_title}, \texttt{title}\rangle, \langle \texttt{movie\_year}, \texttt{year}\rangle\}$.

Vezměme základní relace ***relation1*** nad S_1 a ***relation2*** nad S_2, pak K_1 je **cizím klíčem** základní relace ***relation1*** na K_2 základní relace ***relation2***, pokud pro libovolnou možnou dvojici hodnot R_1 a R_2 základních relací ***relation1*** a ***relation2*** je K_1 cizím klíčem R_1 na K_2 relace R_2 podle h.

Pokud napíšeme, že y_1, ..., y_n je cizím klíčem základní relace ***relation1*** na z_1, ..., z_n základní relace ***relation2***, pak tím myslíme, že $\{y_1, \ldots, y_n\}$ je cizí klíč základní relace ***relation1*** na $\{z_1, \ldots, z_n\}$ základní relace ***relation2*** podle přejmenování $h = \{\langle y_1, z_1\rangle, \ldots, \langle y_n, z_n\rangle\}$. Například `movie_title`, `movie_year` je cizím klíčem základní relace `movie_cast` na `title`, `year` základní relace relace `movie`. Nevyjádřené h je zde:

$$\{\langle \texttt{movie\_title}, \texttt{title}\rangle, \langle \texttt{movie\_year}, \texttt{year}\rangle\}.$$

Pokud navíc z_1, ..., z_n je primární klíč základní relace ***relation2***, stačí říci pouze, že y_1, ..., y_n je cizím klíčem základní relace ***relation1*** na ***relation2***.

3.1 Cizí klíč v SQL

Omezení základní relace ***relation1***:

```
FOREIGN KEY ( y1, ..., yn ) REFERENCES relation2 ( z1, ..., zn )
```

deklaruje, že y_1, ..., y_n je cizím klíčem základní relace ***relation1*** na z_1, ..., z_n základní relace ***relation2***. Schéma $\{z_1, \ldots, z_n\}$ musí být deklarováno jako kandidátní klíč základní relace ***relation2***. Například:

```
CREATE TABLE movie (
    title text,
    year integer,
    PRIMARY KEY ( title, year )
);

CREATE TABLE movie_cast (
  actor_name text,
  movie_title text,
  movie_year integer,
  PRIMARY KEY ( actor_name, movie_title, movie_year ),
  FOREIGN KEY ( movie_title, movie_year ) REFERENCES movies ( title, year )
);
```

Pokud:

```
# TABLE movie;

   title    | year
------------+------
 The Matrix | 1999
 Dracula    | 1992
(2 rows)
```

```
# TABLE movie_cast;

     actor_name     | movie_title | movie_year
--------------------+-------------+------------
 Keano Reeves       | The Matrix  |       1999
 Keano Reeves       | Dracula     |       1992
 Laurence Fishburne | The Matrix  |       1999
 Gary Oldman        | Dracula     |       1992
(4 rows)
```

pak nelze odstranit:

```
# DELETE FROM movie WHERE title = 'Dracula';

ERROR:  update or delete on table "movie" violates
        foreign key constraint "movie_cast_movie_fkey" on table "movie_cast"
DETAIL:  Key (title, year)=(Dracula, 1992) is still referenced
         from table "casting".
```

ani změnit:

```
# UPDATE movie
  SET    title = 'Drákula'
  WHERE  title = 'Dracula';

ERROR:  update or delete on table "movie" violates
        foreign key constraint "movie_cast_movie_fkey" on table "movie_cast"
DETAIL:  Key (title, year)=(Dracula, 1992) is still referenced
         from table "casting".
```

ani přidat:

```
# INSERT INTO movie_cast VALUES
        ( 'Gary Oldman', 'Batman Begins', 2005);

ERROR:  insert or update on table "casting" violates
        foreign key constraint "movie_cast_movie_fkey"
DETAIL:  Key (movie_title, movie_year)=(Batman Begins, 2005)
         is not present in table "movies".
```

Pokud by cizí klíč obsahoval jediný atribut, můžeme použít omezení atributu y_1:

```
REFERENCES relation2 ( z1 )
```

Například:

```
CREATE TABLE movie (
    movie_id integer PRIMARY KEY,
    title text NOT NULL,
    year integer NOT NULL,
    UNIQUE ( title, year )
);

CREATE TABLE movie_cast (
  actor_name text,
  movie_id integer REFERENCES movies ( movie_id ),
  PRIMARY KEY ( actor_name, movie_id )
);
```

Pokud vynecháme odkazované atributy, doplní se primární klíč relace. Tedy

```
REFERENCES movie
```

znamená:

```
REFERENCES movie ( movie_id )
```

a podobně u dřívějšího příkladu omezení tabulky:

```
FOREIGN KEY ( movie_title, movie_year ) REFERENCE movie
```

znamená:

```
FOREIGN KEY ( movie_title, movie_year ) REFERENCES movie ( title, year )
```

4 Entitně vztahový model

Entitně vztahový model je dán orientovaným grafem, kde

- každý vrchol je buď typu **entitní typ**, nebo **vztah**,
- každý vrchol má název,
- každá hrana je ohodnocena prvkem množiny $\{0\text{-}1, 1, N\}$,

a dále platí, že

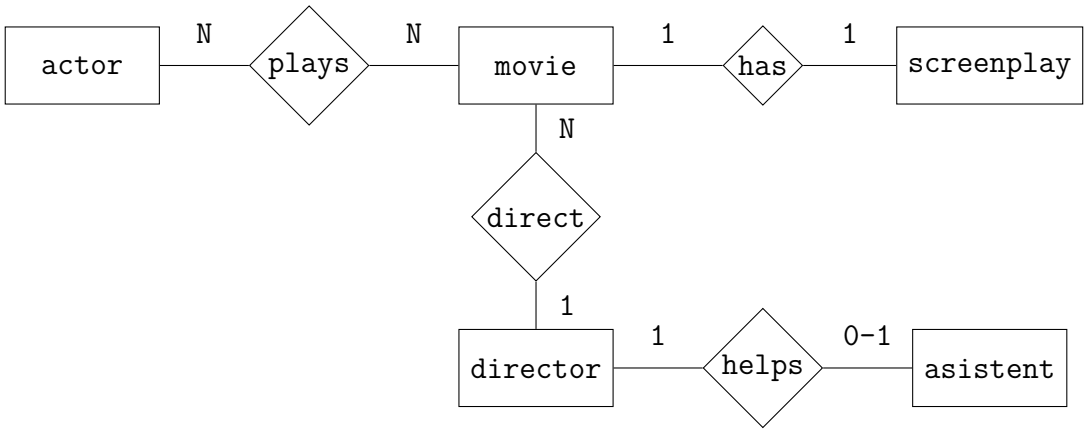
- do každého vrcholu typu vztah vede právě jedna hrana a to z vrcholu typu entitní typ,
- z každého vrcholu typu vztah vede právě jedna hrana a to do vrcholu typu entitní typ,

- každý vrchol typu entitní typ má za sousedy pouze vrcholy typu vztah.

Tříprvkový sled ***entity_type1***, ***relationship***, ***entity_type2***, kde ***entity_type1*** a ***entity_type2*** jsou vrcholy typu entitní typ a ***relationship*** je vrchol typu vztah, nazýváme **vztahem** ***relationship*** mezi entitními typy ***entity_type1*** a ***entity_type2***. Dvojici $\langle x, y \rangle$, kde x je ohodnocení hrany z ***entity_type1*** do ***relationship*** a y z ***relationship*** do ***entity_type2***, nazýváme **kardinalitou** vztahu.

Vrcholy typu entitní typ zobrazujeme obdélníkem a vrcholy typu vztah čtvercem postaveném na vrchol. Názvy vrcholů vypisujeme dovnitř čtyřúhelníků. Orientaci hran není potřeba indikovat šipkou – vyplývá z názvů vztahů mezi entitními typy.

Příklad entitně vztahového modelu:



Každý vrchol typu entitní typ označuje množinu entit daného typu. Například vrchol `movie` označuje množinu všech filmů (v námi vytyčené části světa). Vezměme vztah ***relationship*** mezi entitními typy ***entity_type1*** a ***entity_type2***. Množinu entit typu ***entity_type1*** si označme X_1 a množinu entit typu ***entity_type2*** si označíme X_2. Vztah ***relationship*** mezi ***entity_type1*** a ***entity_type2*** pak označuje binární relaci mezi X_1 a X_2:

$$\{\langle e_1, e_2 \rangle \in X_1 \times X_2 \mid \ e_1 \text{ je ve vztahu } \textbf{\textit{relationship}} \text{ s } e_2\}$$

Například vztah `plays` mezi `actor` a `movie` označuje binární relaci mezi množinou herců A a množinou filmů M: $\{\langle a, m \rangle \in A \times M \mid \text{herec } a \text{ hraje ve filmu } m\}$.

Vezměme libovolný vztah ***relationship*** mezi entitními typy ***entity_type1*** a ***entity_type2*** s kardinalitou $\langle x, y \rangle$ a nechť X_1 a X_2 jsou množiny entit typů ***entity_type1*** a ***entity_type2*** a R je binární relace označená ***relationship***. Pak musí platit:

- Pokud $x = \mathtt{1}$, pak pro každé $x_2 \in X_2$ existuje právě jedno $x_1 \in X_1$ takové, že $\langle x_1, x_2 \rangle \in R$.
- Pokud $y = \mathtt{1}$, pak pro každé $x_1 \in X_1$ existuje právě jedno $x_2 \in X_2$ takové, že $\langle x_1, x_2 \rangle \in R$.

- Pokud $x = \texttt{0-1}$, pak pro každé $x_2 \in X_2$ existuje nejvýše jedno $x_1 \in X_1$ takové, že $\langle x_1, x_2 \rangle \in R$.
- Pokud $y = \texttt{0-1}$, pak pro každé $x_1 \in X_1$ existuje nejvýše jedno $x_2 \in X_2$ takové, že $\langle x_1, x_2 \rangle \in R$.

Například vztah `direct` mezi `director` a `movie` s kardinalitou $\langle \texttt{1}, \texttt{N} \rangle$ vynucuje, aby každý film režíroval právě jeden režisér, ale režisér může režírovat libovolný počet filmů.

Platí následující.

- Pokud $x = \texttt{1}$, pak R^{-1} je zobrazení X_2 do X_1.
- Pokud $y = \texttt{1}$, pak R je zobrazení X_1 do X_2.
- Pokud $x = y = \texttt{1}$, pak R je bijekce.
- Pokud $x = \texttt{0-1}$, pak R^{-1} je částečné zobrazení X_2 do X_1.
- Pokud $y = \texttt{0-1}$, pak R je částečné zobrazení X_1 do X_2.

5 Převod entitně vztahového modelu na relační model

Postup převedení entitně vztahového modelu na základní relace je následující.

1. Pro každý entitní typ vytvoříme základní relaci a určíme její primární klíč.
2. Pro každý vztah *relationship* mezi entitními typy *entity_type1* a *entity_type2* s kardinalitou $\langle x, y \rangle$ provedeme:
   (a) Pokud $x = y = \texttt{1}$, pak spojíme základní relace pro *entity_type1* a *entity_type2*. Primární klíč jedné základní relace převedeme na alternativní klíč.
   (b) Pokud $x = \texttt{1}$ a $y \neq \texttt{1}$, pak do základní relace pro *entity_type2* přidáme cizí klíč na základní relaci pro *entity_type1*.
   (c) Pokud $y = \texttt{1}$ a $x \neq \texttt{1}$, pak do základní relace pro *entity_type1* přidáme cizí klíč na základní relaci pro *entity_type2*.
   (d) Pokud $x \neq \texttt{1}$ a $x \neq \texttt{1}$, pak vytvoříme základní relaci pro *relationship* s cizím klíčem na základní relaci pro *entity_type1* a s cizím klíčem na základní relaci pro *entity_type2*.
   (e) Pokud $x = \texttt{0-1}$, pak prohlásíme cizí klíč na základní relaci pro *entity_type2* za alternativní klíč.
   (f) Pokud $y = \texttt{0-1}$, pak prohlásíme cizí klíč na základní relaci pro *entity_type1* za alternativní klíč.

Převod si demonstrujeme na výše uvedeném diagramu. Nejprve vytvoříme základní relace pro entitní typy:

- Základní relace `movie`:
  - Schéma: {`movie_id`, `title`, `year`, `length`}
  - Charakteristická vlastnost: „Film `movie_id` má název `title` a byl vytvořený roku `year`.“
  - Primární klíč: {`movie_id`}
- Základní relace `actor`:
  - Schéma: {`actor_id`, `name`, `born`}
  - Charakteristická vlastnost: „Herce `actor_id` se jmenuje `name` a narodil se roku `born`.“
  - Primární klíč: {`actor_id`}
- Základní relace `screenplay`:
  - Schéma: {`screenplay_id`, `scene_count`}
  - Charakteristická vlastnost: „Scénář `screenplay_id` má `scene_count` scén.“
  - Primární klíč: {`screenplay_id`}
- Základní relace `director`:
  - Schéma: {`director_id`, `name`, `oscar_count`}
  - Charakteristická vlastnost: „Režisér `director_id` se jmenuje `name` a získal `oscar_count` Oskarů.“
  - Primární klíč: {`director_id`}
- Základní relace `asistent`:
  - Schéma: {`asistent_id`, `name`}
  - Charakteristická vlastnost: „Asistent `asistent_id` se jmenuje `name`.“
  - Primární klíč: {`asistent_id`}

Postupně zpracujeme všechny vztahy.

1. Pro vztah `has` mezi `movie` a `screenplay` s kardinalitou $\langle 1, 1 \rangle$ podle bodu (a) spojíme základní relace `movie` a `screenplay`.

   Základní relace `movie`:

   - Schéma: {`movie_id`, `title`, `screenplay_id`, `scene_count`}
   - Charakteristická vlastnost: „Film `movie_id` s názvem `title` z roku `year` byl natočený podle scénáře `screenplay_id` obsahující `scene_count` scén.“

- Primární klíč: {movie_id}
- Alternativní klíče: {screenplay_id}

2. Pro vztah direct mezi director a movie s kardinalitou ⟨1, N⟩ podle bodu (b) přidáme cizí klíč do základní relace movie na základní relaci director.

   Základní relace movie:

   - Schéma: {movie_id, title, screenplay_id, scene_count, director_id}
   - Charakteristická vlastnost: „Film movie_id s názvem title z roku year byl natočený režisérem director_id podle scénáře screenplay_id obsahující scene_count scén.“
   - Primární klíč: {movie_id}
   - Alternativní klíče: {screenplay_id}
   - Cizí klíče: {director_id} na director

3. Pro vztah helps mezi asistent a director s kardinalitou ⟨0-1, 1⟩ podle bodu (c) přidáme cizí klíč {director_id} na základní relaci director do základní relace asistent a podle bodu (e) přidáme do základní relace asistent alternativní klíč {director_id}.

   Základní relace asistent:

   - Schéma: {asistent_id, name, director_id}
   - Charakteristická vlastnost: „Asistent asistent_id se jmenuje name a je k ruce režisérovi director_id.“
   - Primární klíč: {asistent_id}
   - Cizí klíče: {director_id} na director

4. Pro vztah plays mezi actor a movie s kardinalitou ⟨N, N⟩ podle bodu (d) vytvoříme základní relaci movie_cast pro plays obsahující cizí klíč {actor_id} na základní relaci actor a cizí klíč {movie_id} na základní relaci movie.

   Základní relace movie_cast:

   - Schéma: {actory_id, movie_id}
   - Charakteristická vlastnost: „Herec actor_id hrál ve filmu movie_id.“
   - Primární klíč: {actory_id, movie_id}
   - Cizí klíče: {actor_id} na actor, {movie_id} na movie

Tím proces převodu končí. Výsledné definice základních relací v SQL:

```
CREATE TABLE director (
  director_id integer PRIMARY KEY,
  name text,
  oscar_count integer
);
```

```
-- Režisér "director_id" se jmenuje "name"
-- a získal "oscar_count" Oskarů.

CREATE TABLE movie (
    movie_id integer PRIMARY KEY,
    title text,
    year integer,
    screenplay_id integer UNIQUE,
    scene_count integer,
    director_id integer REFERENCES director
);
-- Film "movie_id" s názvem "title"
-- byl natočený režisérem "director_id" roku "year"
-- podle scénáře "screenplay_id"
-- obsahující "scene_count" scén.

CREATE TABLE actor (
  actor_id integer PRIMARY KEY,
  name text,
  born integer
);
-- Herce "actor_id" se jmenuje "name"
-- a narodil se roku "born".

CREATE TABLE movie_cast (
  actor_id integer REFERENCES actor,
  movie_id integer REFERENCES movie,
  PRIMARY KEY ( actor_id, movie_id )
);
-- Herec "actor_id" hrál ve filmu "movie_id".

CREATE TABLE asistent (
  asistent_id integer PRIMARY KEY,
  name text,
  director_id integer UNIQUE REFERENCES director
);
-- Asistent "asistent_id" se jmenuje "name"
-- a je k ruce režisérovi "director_id".
```

Otázky a úkoly na cvičení

1. Určete všechny nadklíče následující relace.

name	born	nationality
Helena Bonham Carter	1966	England
Emma Watson	1990	English
Kyle MacLachlan	1959	USA
Hugh Laurie	1959	English

2. Vezměme relační proměnnou `actor` danou vlastností „Herec `actor_id` jménem `name` se narodil roku `born` a je národnosti `nationality`“. Předpokládejte, že nemůžou existovat dva herci stejného jména narození ve stejný rok. Určete nadklíče a kandidátní klíče proměnné `actor`. Který z kandidátních klíčů byste zvolili za primární klíč?

3. Vezměme relaci z prvního příkladu a označme si ji R_1. Dále uvažujme relace R_2:

actor	movie
Emma Watson	Little Women
Kyle MacLachlan	Dune
Hugh Laurie	101 Dalmatians

R_3:

actor	movie
Helena Bonham Carter	The King's Speech
Emma Watson	Little Women
Kyle MacLachlan	Dune
Hugh Laurie	101 Dalmatians

a R_4:

actor	movie
Helena Bonham Carter	The King's Speech
Emma Watson	Little Women
Emma Watson	Beauty and the Beast
Kyle MacLachlan	Dune
Hugh Laurie	101 Dalmatians

Rozhodněte, zda je $\{\texttt{name}\}$ cizím klíčem relace R_1 na $\{\texttt{actor}\}$ relace R_2, R_3 a R_4 podle $h = \{\langle \texttt{name}, \texttt{actor} \rangle\}$.

4. Vezměme základní relaci `actor` danou vlastností „Herec `actor` narozený roku `born` žije ve městě `town` v zemi `country`.“ a základní relaci `town` danou vlastností „Město `name` nacházející se v zemi `country` má `population` obyvatel.“ Předpokládejte, že v žádné zemi neexistují dvě města stejného názvu. Je `town`, `country` cizím klíčem základní relace `actor` na `name`, `country` základní relace `town`?

5. Vytvořte entitně vztahový model pro databázi knih v knihovně. V modelu musí být zachyceny všechny svazky, které knihovna nabízí. Dále u každé knihy musí být uvedeno její umístění v regálu. Různé kopie téže knihy nemohou být v různých regálech. U knihy dále chcete uchovávat jejího autora a jeho rok narození. Musí být možné identifikovat každou kopii knihy.

6. Převeďte entitně vztahový model z předchozího úkolu na relační model.