

## 6. Návrh databáze

verze z 26. října 2023

### 1 Primární klíč

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

Řekneme, že dvě  $n$ -tice  $r_1, r_2$  nad  $R$  se *shodují* v  $S$ , pokud  $r_1(S) = r_2(S)$ .

Například  $n$ -tice

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

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

Vezmeme relaci  $\mathcal{D}$  nad  $R$ , pak podmnožinu  $S \subseteq R$  nezveme *nadklíčem*, pokud pro každé dvě  $n$ -tice  $r_1, r_2 \in \mathcal{D}$  platí, že jestliže se  $r_1$  shoduje s  $r_2$  v  $S$ , pak  $r_1 = r_2$ .

Schéma  $R$  je vždy nadklíčem relace  $\mathcal{D}$ .

Například v relaci  $\mathcal{D}$ :

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

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

v  $\mathcal{D}$  se shodují v  $\{\text{title}\}$ , 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 splňují charakteristickou podmínku základní relace *relation*. Pokud je  $S$  nadklíčem každé možné hodnoty základní relace *relation*, pak říkáme, že  $S$  je *nadklíčem* základní relace *relation*. Tedy například  $\{\text{title}, \text{year}\}$  je nadklíčem základní relace *movie*.

Nadklíč  $S$  základní relace *relation* se nazývá jejím *kandidátním klíčem*, pokud žádná vlastní podmnožina  $S$  není nadklíčem základní relace *relation*.

Například u základní relace *movie* je  $\{\text{title}, \text{year}\}$  kandidátní klíč, ale  $\{\text{title}, \text{year}, \text{length}\}$  kandidátním klíčem není.

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

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

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  $\{\text{movie\_id}\}$  může být primárním klíčem základní relace *movie* a  $\{\text{title}, \text{year}\}$  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 (  $y_1, \dots, y_n$  )
```

deklaruje, že  $\{y_1, \dots, y_n\}$  je primárním klíčem základní relace.

Omezení základní relace

```
UNIQUE (  $y_1, \dots, y_n$  )
```

deklaruje, že  $\{y_1, \dots, y_n\}$  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  $\{\text{movie\_id}\}$  je primárním a  $\{\text{title}, \text{year}\}$  alternativním klíčem základní relace movie.

Tedy do relace:

```
# 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 víme, řádek tabulky pro zadaný sloupec nemusí obsahovat hodnotu. Říkáme pak, že obsahuje *null* hodnotu. V SQL ji zapisujeme NULL. Připomeňme si, že každá položka *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  $\mathcal{D}_1$  nad  $R_1$  a  $\mathcal{D}_2$  nad  $R_2$ , relační schémata  $S_1 \subseteq R_1$  a  $S_2 \subseteq R_2$  a přejmenování atributů  $h: S_1 \rightarrow S_2$ . Schéma  $S_1$  nazýváme *cizím klíčem* relace  $\mathcal{D}_1$  na  $S_2$  relace  $\mathcal{D}_2$  podle  $h$ , pokud pro každou  $n$ -tici  $r_1 \in \mathcal{D}_1$ , existuje právě jedna  $n$ -tice  $r_2 \in \mathcal{D}_2$  taková, že  $\rho_h(r_1(S_1)) = r_2(S_2)$ .

Například pro relaci  $\mathcal{D}_1$ :

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  $\mathcal{D}_2$ :

title	year
The Matrix	1999
Dracula	1992

(2 rows)

je  $\{\text{movie\_title}, \text{movie\_year}\}$  cizím klíčem relace  $\mathcal{D}_1$  na  $\{\text{title}, \text{year}\}$  relace  $\mathcal{D}_2$  podle  $\{\langle \text{movie\_title}, \text{title} \rangle, \langle \text{movie\_year}, \text{year} \rangle\}$ .

Vezměme základní relace *relation1* nad  $R_1$  a *relation2* nad  $R_2$ , pak  $S_1$  je *cizím klíčem* základní relace *relation1* na  $S_2$  základní relace *relation2*, pokud pro libovolné možné hodnoty  $\mathcal{D}_1$  a  $\mathcal{D}_2$  základních relací *relation1* a *relation2* je  $S_1$  cizím klíčem  $\mathcal{D}_1$  na  $S_2$  relace  $\mathcal{D}_2$  podle  $h$ .

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

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

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

### 3.1 Cizí klíč v SQL

Omezení základní relace *relation1*

```
FOREIGN KEY (  $y_1, \dots, y_n$  ) REFERENCES relation2 (  $z_1, \dots, z_n$  )
```

deklaruje, že  $y_1, \dots, y_n$  je cizím klíčem základní relace *relation1* na  $z_1, \dots, z_n$  základní relace *relation2*. Schéma  $\{z_1, \dots, z_n\}$  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 (  $z_1$  )
```

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 z množiny  $\{0-1, 1, N\}$ ,



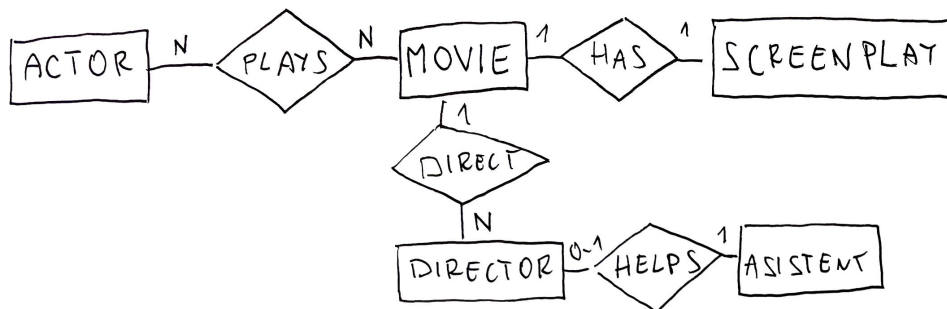
a dále platí, že

- 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 kosočtvercem postaveným na vrchol. Názvy vrcholů vypisujeme dovnitř čtyřúhelníků. Orientaci hran není potřeba indikovat šipkou – vyplývá z názvů vztahů mezi entitními typy.

Příklad grafu entitně vztahového modelu:



Každý vrchol typu entitní typ označuje množinu entit daného typu. Například vrchol *movie* označuje množinu všech filmů (v naší omezené části světa). Vezměme vztah *relationship* mezi entitními typy  $entity\_type1$  a  $entity\_type2$  a množiny  $X_1$  a  $X_2$ , kde  $X_1$  je množina entit typu  $entity\_type1$  a  $X_2$  množina entit typu  $entity\_type2$ . Vztah *relationship* mezi  $entity\_type1$  a  $entity\_type2$  pak označuje binární relaci mezi  $X_1$  a  $X_2$ :

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

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 necht  $X_1$  a  $X_2$  jsou množiny entit typů *entity\_type1* a *entity\_type2* a  $R$  je binární relace označená *relationship*. Pak musí platit:

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

Například vztah *direct* mezi *director* a *movie* s kardinalitou  $\langle \mathbb{N}, 1 \rangle$  vynucuje, že každý film režíroval právě jeden režisér.

Jistě platí:

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

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  $x = y = 1$ , spojíme základní relace pro *entity\_type1* a *entity\_type2*. Primární klíč jedné základní relace převedeme na alternativní klíč.
  - (b) Pokud  $x = 1$  a  $y \neq 1$ , do základní relace pro *entity\_type1* přidáme cizí klíč na základní relaci pro *entity\_type2*.
  - (c) Pokud  $y = 1$  a  $x \neq 1$ , do základní relace pro *entity\_type2* přidáme cizí klíč na základní relaci pro *entity\_type1*.
  - (d) Pokud  $x \neq 1$  a  $y \neq 1$ , 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  $x = 0-1$ , prohlásíme cizí klíč na základní relaci pro *entity\_type1* alternativním klíčem.
- (f) Pokud  $y = 0-1$ , prohlásíme cizí klíč na základní relaci pro *entity\_type2* alternativním klíčem.

Provedeme převod pro výše uvedený diagram.

Fakt, že základní relace *relation* je nad  $R = \{y_1, \dots, y_n\}$  a  $\{y_1, \dots, y_m\} \subseteq R$  je primární klíč můžeme zapsat následovně.

```
relation( $y_1$ , ...,  $y_m$ ,  $y_{m+1}$ , ...,  $y_n$ )
```

Například: *movie*(movie\_id, title, year, length)

Nejprve vytvoříme základní relace:

- *movie*(movie\_id, title, year)  
„Film *movie\_id* má název *title* a byl vytvořený roku *year*.“
- *actor*(actor\_id, name, born)  
„Herce *actor\_id* se jmenuje *name* a narodil se roku *born*.“
- *screenplay*(screenplay\_id, scene\_count)  
„Scénář *screenplay\_id* má *scene\_count* scén.“
- *director*(director\_id, name, oscar\_count)  
„Režisér *director\_id* se jmenuje *name* a získal *oscar\_count* Oskarů.“
- *asistent*(asistent\_id, name)  
„Asistent *asistent\_id* se jmenuje *name*.“

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*:

```
movie(movie_id, title, year, 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.“
  - Alternativní klíče: {*screenplay\_id*}
2. Pro vztah *direct* mezi *director* a *movie* s kardinalitou  $\langle N, 1 \rangle$  podle bodu (c) přidáme cizí klíč do základní relace *movie* na základní relaci *director*:

```
movie(movie_id, title, year, 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.“
  - Alternativní klíče: {`screenplay_id`}
  - Cizí klíče: {`director_id`} na `director`
3. Pro vztah `helps` mezi `asistent` a `director` s kardinalitou  $\langle 1, 0-1 \rangle$  podle bodu (b) přidáme cizí klíč {`director_id`} na základní relaci `director` do základní relace `asistent` a podle bodu (f) přidáme do základní relace `asistent` alternativní klíč {`director_id`}:

```
asistent(asistent_id, name, director_id)
```

- Charakteristická vlastnost: „Asistent `asistent_id` se jmenuje `name` a je k ruce režisérovi `director_id`.“
  - Alternativní klíče: {`director_id`}
  - Cizí klíče: {`director_id`} na `director`
4. Pro vztah `plays` mezi `actor` a `movie` s kardinalitou  $\langle N, N \rangle$  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`:

```
movie_cast(actor_id, movie_id)
```

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

Tím algoritmus 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".

```