Бази от данни

Упражнение 10:

Ограничения

Димитър Димитров

Мотивация

- При модификация на данните в БД можем да допуснем най-различни нередности:
  - Студенти с еднакви ф.н.
  - Актьор без име
  - Филм е от студио, за което няма запис в Studio
- Решение: ограничения
  - Правила, на които трябва да отговарят данните
  - Не позволяват изпълнението на заявки, които ще доведат до недопустими състояния

Съдържание

- Ще разгледаме следните ограничения:
  - Първичен ключ (PRIMARY KEY)
  - UNIQUE
  - Външен ключ (FOREIGN KEY)
  - NOT NULL
  - CHECK

Първичен ключ на таблица

- Формира се от една или няколко колони
  - Пример: title и year в Movie
- Не се допускат повторения
  - Може филми с еднакви заглавия, но годините трябва да са различни
- Не се допускат NULL стойности
  - За нито една от колоните
- Не може повече от един ПК за една таблица
- Кои заявки в SQL биха могли да не бъдат изпълнени при наличие на ПК?

Деклариране на първичен ключ (1)

- Различни начини:
- ```
  CREATE TABLE Battles (
    name VARCHAR(20) PRIMARY KEY,
    date DATE
  );
  ```
  - Само при една колона – не може да поставим PRIMARY KEY два пъти
- ```
  CREATE TABLE Movie (
    title VARCHAR(255),
    year INTEGER,
    length INTEGER,
    inColor CHAR(1),
    studioName CHAR(50),
    producerC# INTEGER,
    PRIMARY KEY (title, year)
  );
  ```

Деклариране на първичен ключ (2)

- Всяко ограничение има име
  - Ако не го укажем ние, СУБД поставя служебно име
-
```
CREATE TABLE Battles (
  name VARCHAR(20) CONSTRAINT PK_Battles PRIMARY KEY,
  date DATE
);
```
- Най-пълен синтаксис:

```
CREATE TABLE Movie (
  title VARCHAR(255),
  year INTEGER,
  …
  producerC# INTEGER,
  CONSTRAINT PK_Movie PRIMARY KEY (title, year)
);
```

- Демонстрация

Първичен ключ – допълнителен материал

- Сурогатен ключ
- Пример за MS SQL: auto incrementing PK
-
```
create table test1 (
  id int identity primary key,
  name nvarchar(50)
);

insert into test1(name) values('First');
       -- указваме стойност за всички колони освен id
insert into test1(name) values('Second');
insert into test1(name) values('Third');
select * from test1;
delete from test1 where id = 3;
insert into test1(name) values('Fourth');
select * from test1;
```

- В MS SQL сравняването на низове не е case sensitive
  - Същото важи и за първичните ключове върху стрингови колони

UNIQUE

- Върху една или няколко колони на една таблица
- Не се допускат повторения
- Допускат се NULL стойности – зависи от СУБД
  - MS SQL – най-много една стойност NULL
- Може да се дефинират много UNIQUE ограничения за една таблица

UNIQUE – синтаксис

- Същите начини като при PRIMARY KEY, например:

```
CREATE TABLE Movie (
  title VARCHAR(255),
  year INTEGER,
  …
  producerC# INTEGER,
  UNIQUE (title, year)
);
```

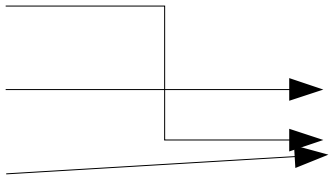
- Следният пример еквивалентен ли е на горния?

```
CREATE TABLE Movie (
  title VARCHAR(255) UNIQUE,
  year INTEGER UNIQUE,
  …
  producerC# INTEGER
);
```

Външен ключ (1)

- Искаме да няма филм, реферираш студио, липсващо в таблицата Studio

Movie		
title	**...**	**studioName**
Gone With the Wind		MGM
Star Wars		Fox
The Usual Suspects		MGM
Outbreak		NULL



Studio	
name	**...**
Disney	
Fox	
MGM	

- Референтна цялостност

Външен ключ (2)

- Върху една или няколко колони
  - В предишния пример – върху studioName
- Реферира първичен ключ в друга или дори същата таблица
- Броят и типът на съставящите го колони трябва да съвпада с тези на PK
  - Да разгледаме външния ключ от StarsIn към Movie
- Допускат се повторения
- Допускат се стойности NULL
- В една таблица може да има много външни ключове

Външен ключ – синтаксис

- Отново различни начини, например:
- ```
  CREATE TABLE Studio (
    name VARCHAR(30) PRIMARY KEY,
    address VARCHAR(255),
    presC# INT REFERENCES MovieExec(cert#)
  );
  ```
- ```
  CREATE TABLE Studio (
    name VARCHAR(30) PRIMARY KEY,
    address VARCHAR(255),
    presC# INT,
    FOREIGN KEY (presC#) REFERENCES MovieExec(cert#)
  );
  ```
- ```
  CREATE TABLE Studio (
    name VARCHAR(30) PRIMARY KEY,
    address VARCHAR(255),
    presC# INT,
    CONSTRAINT FK_Studio_MovieExec FOREIGN KEY (presC#)
      REFERENCES MovieExec(cert#)
  );
  ```
  ```

FK – политики за налагане (1)

- По подразбиране (NO ACTION)
  - Заявка, която би нарушила референтната цялостност, няма да бъде изпълнена
    - Добавяне на филм от несъществуващо студио
    - Изтриване на студио MGM или таблицата Studio
    - Преименуване MGM на Metro-Goldwyn-Mayer в Studio
- Каскадна (CASCADE)
  - При изтриване/преименуване на студио се изтриват/променят всички негови филми
- SET NULL
  - При изтриване/преименуване на студио стойността в studioName за съответните филми се променя на NULL
- SET DEFAULT
  - Като SET NULL, но се използва DEFAULT стойността за съответната колона

FK – политики за налагане (2)

- Пример:

```
CREATE TABLE Studio (
  name VARCHAR(30) CONSTRAINT PK_Studio PRIMARY KEY,
  address VARCHAR(255),
  presC# INT,
  CONSTRAINT FK_Studio_MovieExec
    FOREIGN KEY (presC#)
    REFERENCES MovieExec(cert#)
    ON DELETE CASCADE
    ON UPDATE SET NULL
);
```

NOT NULL

- Не позволява в дадена колона да има стойности NULL
- Примери:

```
address VARCHAR(255) NOT NULL,
presC# INT REFERENCES MovieExec(cert#) NOT NULL
```

  - Към една колона може да укажем множество ограничения – разделяме ги с интервали
- NOT NULL vs DEFAULT

CHECK

- Стойностите в таблицата трябва да отговарят на булево условие
- Ако условието засяга само една колона:

```
inColor CHAR(1) CHECK (inColor in ('Y', 'N'))
```

- Ако засяга повече от една колона:
-
```
CREATE TABLE Movie (
  …
  inColor CHAR(1),
  producerC# INTEGER,
  CHECK (inColor = 'N' OR inColor = 'Y' AND year >= 1918)
);
```

  - Да си припомним курса по УП – какъв е приоритетът на операциите?
- Указването на име на ограничението се извършва по познатия начин

CHECK – особености

- Форматът на условието е познатият ни от WHERE клаузите
- В MS SQL не е позволено условието да съдържа подзаявки
  - Допълнителен материал: заобикаляне с функция, дефинирана от потребителя
- Допълнителен материал: MySQL игнорира CHECK органиченията

Модификация на ограничения

- При вече създадена таблица:
- Добавяне на ограничение:

```
ALTER TABLE Movie
ADD CONSTRAINT PK_Movie
    PRIMARY KEY (title, year);
```

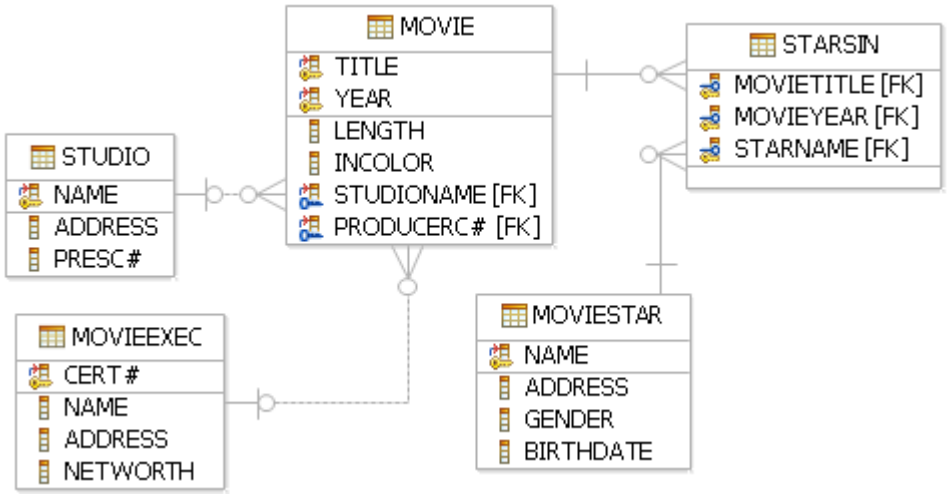
- Премахване на ограничения:
- **ALTER TABLE Studio**
  DROP CONSTRAINT PK_Studio, FK_Studio_MovieExec;
- За NOT NULL синтаксисът е друг, не е необходимо да се знае

Въпроси?

Следват задачи

Задачи - Movies

1. а) Да се направи така, че да не може два филма да имат еднаква дължина.
   б) Да се направи така, че да не може едно студио да има два филма с еднаква дължина.
2. Изтрийте ограниченията, създадени в зад. 1.



Задачи

Зад. 3.
а) За всеки студент се съхранява следната информация:
- факултетен номер - от 0 до 99999, първичен ключ;
- име - до 100 символа;
- ЕГН - точно 10 символа, уникално;
- e-mail - до 100 символа, уникален;
- рождена дата;
- дата на приемане в университета - трябва да бъде поне 18 години след рождената;
За всички атрибути задължително трябва да има зададена стойност (не може NULL)

б) добавете валидация за e-mail адреса - да бъде във формат <низ>@<низ>.<низ>

в) създайте таблица за университетски курсове - уникален номер и име.

Всеки студент може да се запише в много курсове и във всеки курс може да има записани много студенти.

При изтриване на даден курс автоматично да се отписват всички студенти от него.