Създаване и изтриване на изглед

Изгледи

- Релациите създадени с командата CREATE TABLE съществуват физически на диска.
- Други обекти в базата от данни, които не съществуват физически на диска, но имат поведение на таблици са изгледите (view).
- Често изгледите са наричани и виртуални таблици.
- Това е така защото те не съдържат данни, а само дефиниция от коя таблица да бъдат взети данните.

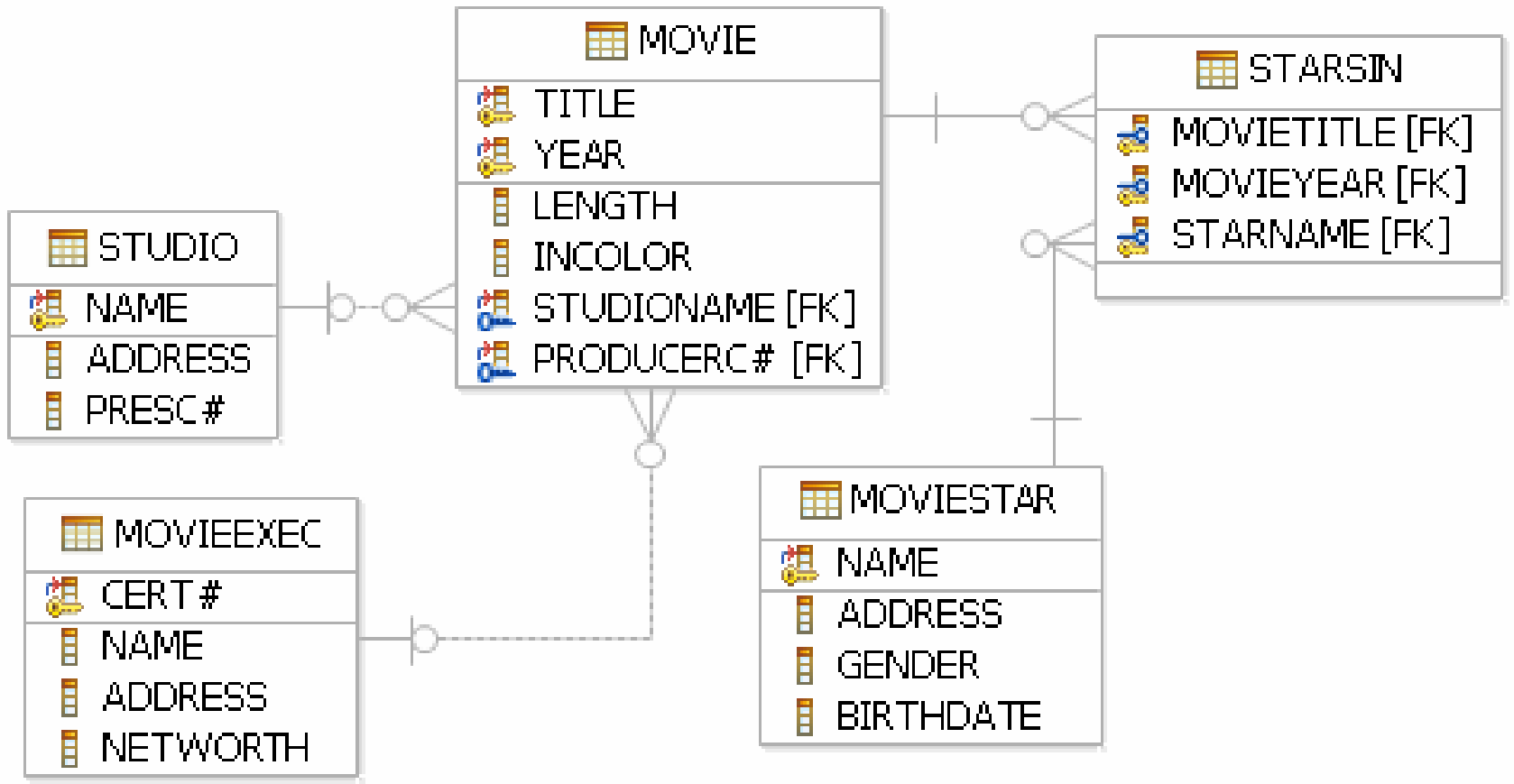
Изгледи

- В SQL изглед се дефинира с:
  - ключовите думи **CREATE VIEW**, последвани от
  - името на изгледа, последвано от
  - ключовата дума *AS*, последвана от
  - SELECT заявка;

```
CREATE VIEW <име_на_изглед>
AS <SELECT заявка>;
```

- Заявката реално е самата дефиниция на изгледа. Тя определя от кои таблици ще бъдат показани данните.

База от данни - Movies



Създаване на изглед

- Например изглед, които ни дава информация за всички филми, актьорите които участват в тях и имената на студията, които ги продуцират

```
CREATE VIEW V_MOVIE_INFO
AS
SELECT TITLE, YEAR, STUDIONAME, STARNAME
FROM MOVIE, STARSIN
WHERE TITLE = MOVIETITLE
AND YEAR = MOVIEYEAR
```

- Към изглед се обръщаме по същият начин, както към таблица.
- Например:

```
SELECT * FROM V_MOVIE_INFO;
```

Обръщане към изглед в заявка

- Всеки път когато правим заявка към изгледа, всъщност правим заявка към таблиците, които участват в дефиницията на изгледа.
- В действителност изгледа не съдържа кортежи.
- За това когато правим заявка към изглед, резултатното множество се връща от таблицата (таблиците) участващи в дефиницията на изгледа.
- Така в резултат на една и съща заявка към изгледа могат да бъдат върнати различни резултатни множества, особено ако таблицата е била модифицирана между двете заявки.
- Ако се изтрие някоя от таблиците, които участват в дефиницията на изгледа, то изгледът става неактивен.
- Това означава, че изгледът трябва да се изтрие и създаде наново.

Именуване на колоните в изгледа

Понякога е удобно още със създаването на изгледа да дадем нови имена на атрибутите, които се връщат в резултат на SELECT заявката. Това става със следният синтаксис:

```
CREATE VIEW my_view(A1, A2 …, An)
AS <SELECT заявка>;
```

- За горният пример, синтаксисът ще изглежда така:

```
CREATE VIEW V_MOVIE_INFO(TITLE, YEAR, NAME, ACTOR)
AS
SELECT TITLE, YEAR, STUDIONAME, STARNAME
FROM MOVIE, STARSIN
WHERE TITLE = MOVIETITLE
AND YEAR = MOVIEYEAR
```

Изгледи

- Всяка колона в изгледа трябва да има име, ако в SELECT клаузата има изрази, задължително трябва или с AS да се даде име на колоната, или при CREATE VIEW да се изброят имената на всички колони.
- Същото важи и ако в SELECT клаузата има колони с едни и същи имена. Причината е очевидна - изгледите трябва да могат да се използват в различни заявки, както и таблиците и ако някоя колона на изгледа няма име, няма да можем да я достъпваме.
- Изгледите не могат да бъдат променяни. Ако по някаква причина искаме да променим дефиницията на изглед, тогава трябва да изтрием съществуващият изглед и да го създадем наново със същото име и нова дефиниция.
- Изглед се изтрива с ключовите думи **DROP VIEW**. Например:

```
DROP VIEW V_MOVIE_INFO;
```

Предимства на изгледите

- Едно от предимствата на изгледите е, че могат да бъдат използвани за реализиране на сигурност в базите от данни.
- Изгледите могат да ограничат достъпа на даден потребител, като сведат неговата “видимост” до определени атрибути или кортежи на дадена релация.
- Например изгледът, който създадохме по-горе. Ако имаме потребител, който да има права да прави SELECT върху изгледа и няма права върху таблицата MOVIE, то създаденият от нас изглед ограничава достъпа на потребителя само до име на филм и година на филм. Другата информация за филма е скрита от този потребител.
- Предимство на изгледите е, че могат да бъдат използвани и за ускоряване на заявките. Това е в сила, когато използваме една и съща заявка често пъти. СУБД имат механизъм за кеширане на такива заявки, по този начин изгледите могат да доведат до ускоряване на изпълнението на съответната заявка.

Updatable и read-only изгледи

- Според дефиницията на изгледа, изгледите могат да бъдат updatable (т.е. такива през, които може да бъде променена оригиналната таблица) и read-only (т.е. такива през, които може само да се извежда информация от оригиналната таблица, но таблицата не може да бъде променяна).
- Има определени изисквания, на които трябва да отговаря дефиницията на един изглед за да бъде той updatable. Ако ги обобщим, те включват следните ограничения:
  - Заявката на updatable изглед не може да включва съединения, подзаявки, групиране и агрегатни функции
  - В общи линии заявката на updatable изглед трябва да бъде само върху една таблица, като трябва да включва всички колони от таблицата. Позволено е да не се включват само тези колони които имат стойности по подразбиране или позволяват NULL.
- Всички останли изгледи, които не отговарят на горните условия са read-only.

Пример за updatable и read-only изгледи

- Пример за read-only изглед е изгледът V_MOVIE_INFO Той включва съединение на две таблици, като ползва само част от колоните на тези таблици.
- По тези причини, през този изглед не могат да бъдат обновени нито една от двете таблици от дефиницията.
- Пример за updatable изглед е следният:

```
CREATE VIEW V_MOVIE_1980
AS
SELECT *
FROM MOVIE
WHERE YEAR > 1980
```

Пример за updatable и read-only изгледи

- През този изглед могат да бъдат правени следните заявки:

```
INSERT INTO V_MOVIE_1980
  VALUES ('Lego', 2016, 100, 'Y', 'MGM', 199);

DELETE FROM V_MOVIE_1980
 WHERE TITLE = 'Lego';

UPDATE V_MOVIE_1980
  SET LENGTH = NULL
  WHERE TITLE = 'Lego';
```

Пример за updatable и read-only изгледи

- През горният изглед могат да бъдат направени и следните промени:

```
UPDATE V_MOVIE_1980
   SET LENGTH = NULL
   WHERE YEAR = 1977
```

- Въпреки че, дефиницията на изгледа ни ограничава да виждаме само записите след 1980 година, с горната заявка обновихме запис от таблицата, който не се вижда от изгледа.

Пример за updatable и read-only изгледи

- Аналогична е ситуацията и при следния INSERT:
- Тоест отново вмъкваме запис през изгледа, а самият изглед не може да види този запис.

```
CREATE VIEW V_MOVIE_1980
AS
SELECT *
FROM MOVIE
WHERE YEAR > 1980


INSERT INTO V_MOVIE_1980
VALUES('NEW MOVIE', 1967, 120, 'Y', 'MGM', 123)
```

Изгледи и опция CHECK

- При такива случаи, ако искаме да избегнам подобни нежелани ефекти се използва опцията WITH CHECK OPTION, която се записва при дефинирането на изгледа.
- Смисълът на тази опция е да не позволява през изгледа да бъдат променяни редове от оригиналната таблица, които след промяната няма да отговарят на условието в WHERE клаузата на изгледа и съответно няма да могат да бъдат видени от него. Например:

```
CREATE VIEW V_MOVIE_1980
AS
SELECT *
FROM MOVIE
WHERE YEAR > 1980
WITH CHECK OPTION;
```

- Сега ако повторим горните заявки, СУБД ще изведе съобщение за грешка, че INSERT заявката нарушава условието на CHECK опцията от дефиницията на изгледа.

Дефиниране на изглед от изглед

- Изгледите са виртуални таблици, това прави възможно дефинирането на изглед в чиято дефиниция участва изглед. Например изгледът:

```
CREATE VIEW V_MOVIE_MGM
AS
SELECT *
FROM V_MOVIE_1980
WHERE STUDIONAME = 'MGM';
```

- Със такива изгледи се работи по същият начин, както и с представените до момента изгледи.
- За тях важат същите правила за updatable и read-only изгледи, както и за WITH CHECK OPTION.

Изгледи и опция CASCADED CHECK

- Някои СУБД подържат и опцията WITH CASCADED CHECK OPTION, чиито смисъл е да не позволява през изгледите - текущият и този върху които е дефиниран съответния изглед, да бъдат променяни редове от оригиналната таблица, които след промяната няма да отговарят на условията в WHERE клаузата на изгледите.
- Понеже проверката се извършва каскадно за всички изгледи от тук идва и думата CASCADED в опцията.
- Например:

```
CREATE VIEW V_MOVIE_MGM
AS
SELECT *
FROM V_MOVIE_1980
WHERE STUDIONAME = 'MGM'
WITH CASCADED CHECK OPTION;
```

Изгледи и опция CASCADED CHECK

- Този изглед ще позволява само заявки, при които годината на филма е < 1980 и името на студиото е MGM

```
INSERT INTO V_MOVIE_MGM
VALUES('NEW MOVIE', 1967, 120, 'Y', 'MGM', 123);
```

- Отново ще доведат до извеждане на съобщение за грешка от СУБД, че INSERT заявката нарушава условието на CHECK опцията от дефиницията на изгледа.