Предметна область: Фільмотека .

Основні предметно-значимі сутності: фільми,актори.

Основні предметно-значимі атрибути сутності:

- фільми – назва фільму, жанр, тривалість, рік виходу, стрічка (кол. чи ч.б.).
- актори – прізвище, дата народження; стать

Основні вимоги до функцій системи:

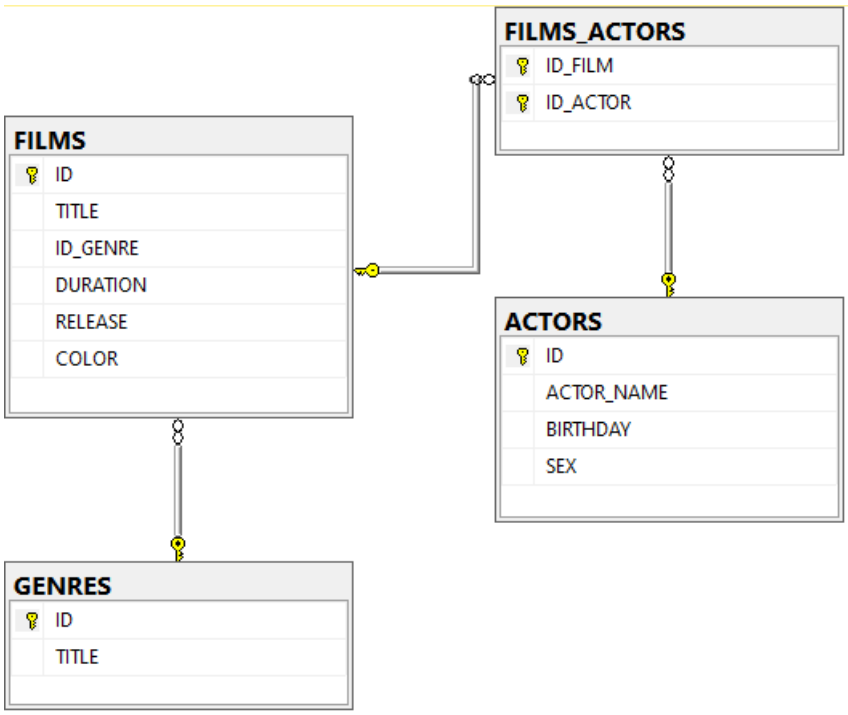
- вибрати всіх акторів, що грали у фільмі вказаного жанру;
- підрахувати кількість фільмів, зіграних кожним актором;
- визначити актора, що зіграв найбільше ролей за останні 10 років.

Тригери:

1. На видалення запису з таблиці «Фільми». Якщо з цим фільмом пов’язані записи в таблиці «АкториФільми», видалити і ці записи.
2. На додавання / оновлення записів в таблиці «АкториФільми». Створити представлення «ІндексАктивності» з полями «код_актора», «ім’я», «Прізвище»,«кількість_фильмів», «рік_виходу», де поле «кількість_фільмів» розраховується на кожний рік, коли знімався даний актор. При додаванні запису в таблицю «АкториФільми» і при оновлені запису, якщо поле «рік_виходу» змінювалось, оновлювати вид «ІндексАктивності».

Збережена процедура:

Процедура повинна повертати кількість фільмів конкретного жанру.



```
CREATE TABLE GENRES(
ID INT PRIMARY KEY IDENTITY,
TITLE VARCHAR(20) NOT NULL);

CREATE TABLE FILMS(
ID INT IDENTITY PRIMARY KEY,
TITLE VARCHAR(20) NOT NULL,
ID_GENRE INT NOT NULL,
DURATION TIME(0) NOT NULL,
RELEASE INT CHECK(RELEASE BETWEEN 1960 AND 2019) NOT NULL,
COLOR VARCHAR(3) CHECK(COLOR IN('YES','NO'))NOT NULL,
CONSTRAINT ID_GENRE_FOREIGN FOREIGN KEY(ID_GENRE) REFERENCES GENRES(ID));

CREATE TABLE ACTORS(
ID INT PRIMARY KEY IDENTITY,
ACTOR_NAME VARCHAR(20) NOT NULL,
BIRTHDAY DATE NOT NULL,
SEX CHAR(1) CHECK(SEX IN ('M','W')) NOT NULL);

CREATE TABLE FILMS_ACTORS(
ID_FILM INT NOT NULL ,
ID_ACTOR INT NOT NULL,
PRIMARY KEY(ID_FILM, ID_ACTOR),
CONSTRAINT FILM_FOREIGN FOREIGN KEY (ID_FILM) REFERENCES FILMS(ID),
CONSTRAINT ACTOR_FOREIGN FOREIGN KEY (ID_ACTOR) REFERENCES ACTORS(ID));
```

```
INSERT INTO ACTORS
VALUES ('PITT','1960-10-5','M');
 INSERT INTO ACTORS
 VALUES ('DICAPRIO','1976-1-2','M');
 INSERT INTO ACTORS
 VALUES ('JOLIE','1980-5-9','W');
 INSERT INTO ACTORS
 VALUES ('DEPP','1972-9-1','M');
 INSERT INTO ACTORS
 VALUES ('ROBERTS','1980-10-5','W');
 INSERT INTO ACTORS
 VALUES ('WATSON','1982-2-4','W');
 INSERT INTO ACTORS
 VALUES ('MERFI','1971-10-5','M');
 INSERT INTO ACTORS
 VALUES ('ZELENSKII','1975-2-4','M');
 INSERT INTO ACTORS
 VALUES ('DOBRYNIN','1969-2-28','M');
 INSERT INTO ACTORS
 VALUES ('BOKLAN','1955-12-11','M');
 INSERT INTO ACTORS
VALUES ('ANDRIEV','1971-2-4','M');
 INSERT INTO ACTORS
 VALUES ('STUPKA','1966-12-5','M');
 INSERT INTO ACTORS
 VALUES ('ALEKSEEV','1951-2-4','M');
 INSERT INTO ACTORS
 VALUES ('ZAVOROTNUIK','1969-2-11','W');
 INSERT INTO ACTORS
 VALUES ('MOGILEVSKA','1955-9-8','W');
```

```
INSERT INTO FILMS
 VALUES ('DE ORIGIN',1,'1:50','2014','YES');
 INSERT INTO FILMS
 VALUES ('FRIDAY 13',2,'2:10','2000','YES');
 INSERT INTO FILMS
 VALUES ('TWELVE CHAIRS',3,'3:00','1980','NO');
 INSERT INTO FILMS
 VALUES ('ONCE IN ODESSA',4,'0:20','2017','YES');
 INSERT INTO FILMS
 VALUES ('RIVERDALE',6,'0:50','2017','YES');
 INSERT INTO FILMS
 VALUES ('KVARTAL',7,'1:30','2009','YES');
 INSERT INTO FILMS
 VALUES ('ALADDIN',9,'1:40','2000','NO');
 INSERT INTO FILMS
 VALUES ('ANGRY BIRDS',10,'1:10','2014','YES');
 INSERT INTO FILMS
 VALUES ('IT',5,'1:50','1960','NO');
 INSERT INTO FILMS
 VALUES ('ASTRAL',8,'2:00','2003','YES');
  INSERT INTO FILMS
 VALUES ('TRANSILVANIA',5,'3:10','2014','YES');
 INSERT INTO FILMS
 VALUES ('IT2',6,'1:50','2015','NO');
 INSERT INTO FILMS
 VALUES ('EXIT',7,'3:00','2005','YES');
```

```
INSERT INTO FILMS_ACTORS
 VALUES('1','2');
 INSERT INTO FILMS_ACTORS
 VALUES('1','3');
 INSERT INTO FILMS_ACTORS
 VALUES('2','3');
 INSERT INTO FILMS_ACTORS
 VALUES('2','1');
  INSERT INTO FILMS_ACTORS
 VALUES('3','13');
  INSERT INTO FILMS_ACTORS
 VALUES('4','8');
 INSERT INTO FILMS_ACTORS
 VALUES('4','12');
 INSERT INTO FILMS_ACTORS
 VALUES('4','11');
 INSERT INTO FILMS_ACTORS
 VALUES('5','6');
  INSERT INTO FILMS_ACTORS
 VALUES('6','8');
  INSERT INTO FILMS_ACTORS
 VALUES('6','9');
 INSERT INTO FILMS_ACTORS
 VALUES('7','7');
 INSERT INTO FILMS_ACTORS
 VALUES('8','3');
```

```
INSERT INTO GENRES
 VALUES ('HORROR');
  INSERT INTO GENRES
 VALUES ('COMEDY');
  INSERT INTO GENRES
 VALUES ('DRAMA');
  INSERT INTO GENRES
 VALUES ('MELODRAMA');
  INSERT INTO GENRES
 VALUES ('FANTASY');
  INSERT INTO GENRES
 VALUES ('SCIENCE');
  INSERT INTO GENRES
 VALUES ('CARTOON');
  INSERT INTO GENRES
 VALUES ('HISTORY');
  INSERT INTO GENRES
 VALUES ('FAMILY');
  INSERT INTO GENRES
 VALUES ('FANTASTIC');
```

```
INSERT INTO FILMS_ACTORS
VALUES('9','5');
INSERT INTO FILMS_ACTORS
VALUES('9','10');
 INSERT INTO FILMS_ACTORS
VALUES('10','9');
INSERT INTO FILMS_ACTORS
VALUES('10','7');
INSERT INTO FILMS_ACTORS
VALUES('11','15');
 INSERT INTO FILMS_ACTORS
VALUES('11','14');
INSERT INTO FILMS_ACTORS
VALUES('11','4');
INSERT INTO FILMS_ACTORS
VALUES('12','5');
 INSERT INTO FILMS_ACTORS
VALUES('12','14');
INSERT INTO FILMS_ACTORS
VALUES('13','15');
```

Лабораторна робота 4

1. Вибрати всіх акторів, що грали у фільмі вказаного жанру.

```
SELECT ACTOR_NAME FROM ACTORS
 INNER JOIN FILMS_ACTORS ON ACTORS.ID=FILMS_ACTORS.ID_ACTOR
  INNER JOIN FILMS ON FILMS.ID=FILMS_ACTORS.ID_FILM
   INNER JOIN GENRES ON FILMS.ID_GENRE=GENRES.ID WHERE GENRES.TITLE='FANTASY';
```

2. Підрахувати кількість фільмів, зіграних кожним актором.

```
SELECT ACTOR_NAME,COUNT(ACTORS.ID) AS ROLES FROM ACTORS
 INNER JOIN FILMS_ACTORS ON ACTORS.ID=FILMS_ACTORS.ID_ACTOR
  GROUP BY ACTOR_NAME ORDER BY ROLES;
```

3. Визначити актора, що зіграв найбільше ролей за останні 10 років.

```
SELECT TOP 1 ACTOR_NAME,COUNT(ACTORS.ID) AS ROLES FROM ACTORS
 INNER JOIN FILMS_ACTORS ON ACTORS.ID=FILMS_ACTORS.ID_ACTOR
  INNER JOIN FILMS ON FILMS.ID=FILMS_ACTORS.ID_FILM WHERE FILMS.RELEASE >= '2009'
   GROUP BY ACTOR_NAME ORDER BY ROLES DESC;
```

Лабораторна робота 5

Визначити акторів, що зіграли найбільше ролей.

```
SELECT ACTOR_NAME,COUNT(ACTORS.ID) AS ROLES FROM ACTORS
 INNER JOIN FILMS_ACTORS ON ACTORS.ID=FILMS_ACTORS.ID_ACTOR
 GROUP BY ACTOR_NAME
          HAVING COUNT(ACTORS.ID) >= ALL
           (SELECT COUNT(ACTORS.ID) FROM ACTORS
 INNER JOIN FILMS_ACTORS ON ACTORS.ID=FILMS_ACTORS.ID_ACTOR
  INNER JOIN FILMS ON FILMS.ID=FILMS_ACTORS.ID_FILM
          GROUP BY ACTOR_NAME);
```

Лабораторна робота 6

Створити представлення «ІндексАктивності» з полями «код_актора», «ім'я», «Прізвище»,«кількість_фильмів», «рік_виходу», де поле «кількість_фільмів» розраховується на кожний рік, коли знімався даний актор. При додаванні запису в таблицю «АкториФільми» і при оновлені запису, якщо поле «рік_виходу» змінювалось, оновлювати вид «ІндексАктивності».

```
CREATE VIEW INDEX_ACTIVITY AS
SELECT ACTORS.ID, ACTOR_NAME, COUNT(ACTORS.ID) AS ROLES, FILMS.RELEASE FROM ACTORS
 INNER JOIN FILMS_ACTORS ON ACTORS.ID=FILMS_ACTORS.ID_ACTOR
  INNER JOIN FILMS ON FILMS_ACTORS.ID_FILM=FILMS.ID
   GROUP BY FILMS.RELEASE, ACTORS.ID, ACTOR_NAME;
```

Лабораторна робота 7

Процедура повинна повертати кількість фільмів конкретного жанру.

```
CREATE PROCEDURE SAVED_PROC @GENRESTITLE VARCHAR(20)OUTPUT
AS
IF EXISTS (SELECT 'TRUE' FROM GENRES WHERE GENRES.TITLE = @GENRESTITLE)
 SELECT GENRES.TITLE, COUNT(FILMS.ID_GENRE) AS AMOUNT FROM GENRES
  INNER JOIN FILMS ON GENRES.ID = FILMS.ID_GENRE
   WHERE GENRES.TITLE = @GENRESTITLE
   GROUP BY GENRES.TITLE
ELSE
 BEGIN
  PRINT 'DOES NOT EXIST'
 END
```

Лабораторна робота 8

На видалення запису з таблиці «Фільми». Якщо з цим фільмом пов'язані записи в таблиці «АкториФільми», видалити і ці записи.

```
CREATE TRIGGER FILM_DELETE ON FILMS
INSTEAD OF DELETE
AS
 DECLARE @ID INT
 SELECT @ID=ID FROM FILMS
IF EXISTS (SELECT 'TRUE' FROM FILMS_ACTORS WHERE ID_FILM=@ID)
BEGIN
 DELETE FROM FILMS_ACTORS WHERE ID_FILM=@ID
END
```