**пПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение**

**высшего профессионального образования**

**Национальный исследовательский университет**

**«Высшая школа экономики»**

**Московский институт электроники и математики**

**Департамент прикладной математики**

**Образовательная программа «Компьютерная безопасность»**

**Отчёт по практической работе по дисциплине «СУБД»**

**Домашняя практическая работа № 2**

**«Проектирование баз данных»**

**Вариант «Фильмотека»**

**Выполнил:**

**Астраханцев Роман Геннадьевич**

**СКБ-171**

**Проверил:**

**Профессор, Белов А.В.**

**Москва-2020**

**Постановка задачи**

Модель «Фильмотека» должна содержать информацию о кинофильмах, актерах, киностудиях и контрактах между актерами и киностудиями

Требуется построить E-R диаграмму средствами AllFusion Data Modeler (ERwin), определить нормальную форму и сформулировать запросы на языке реляционной алгебры следующие запросы:

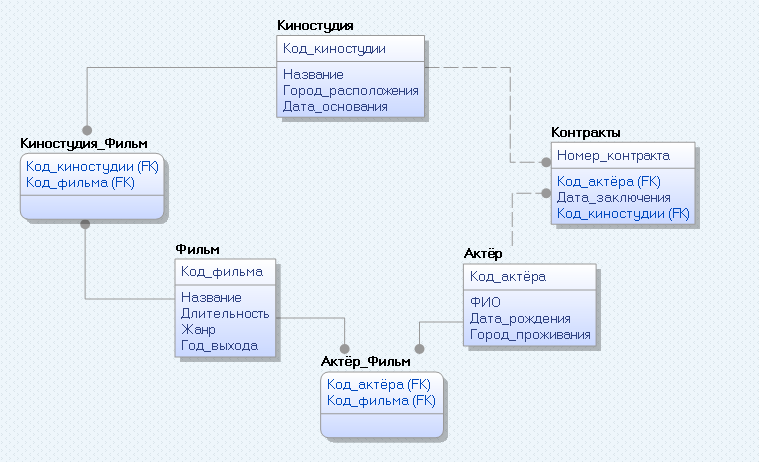
* получить список всех актеров, снимающихся на заданной киностудии;
* получить список кинофильмов, в которых снимаются те же актеры, что и в заданном кинофильме;
* получить пары (Ф.И.О. актера, № контракта), занятых в фильмах, выпущенных на заданной киностудии;
* получить список киностудий, в которых были сняты фильмы по заданной тематике и в заданном году;
* получить список актеров, у которых имеется контракт с киностудией, расположенной в том же городе, в каком проживают эти актеры;
* получить список кинофильмов, в которых заданный актер не снялся ни разу.

*Выбранная среда реализации – Microsoft SQL Server*

**Создание БД**

С помощью мастера была создана база данных в MS SQL сервере с простой моделью восстановления.

С помощью Erwin была спроектирована логическая модель базы данных (см. рис 1), а также сгенерирован код запроса на языке T-SQL (рис. 2), который позже был отредактирован и выполнен на SQL сервере.

  
Рис.1 Логическая модель БД «Фильмотека»

Итоговый код создания базы данных и результат его выполнения приведён в приложении 1. Его функция заключается только создании таблиц данных. Необходимы ограничения и триггеры приведены в дальнейшем пункте.

**Ограничения и триггеры**

Для сохранения целостности данных были созданы ограничения на данные. Так, например, год выхода фильма быть больше 1800 (то есть после появления технологии записи видеозаписей) и быть не сильно больше текущего года (это сделано для анонсов фильмов). Подобные ограничения были сделаны для всех полей.

Так же были предусмотрены случаи удаления данных и их изменения. Так, например, в таблицах выражающие отношения многие ко многим при удаления родительских сущностей данные выставляются в NULL. Это сделано для того, чтобы сохранить информацию о, например, количестве актёров в том или ином фильме.

В связи с этим, были созданы триггеры, которые удаляют записи, в котором не осталось информации. Например, если в таблице Актёр\_Фильм оба поля Код\_актёра и Код\_фильма стали NULL, то такая запись удаляется. Код ограничений и триггеров находится в приложении 2.

**Импорт и экспорт данных.**

1. **Используя мастер импорта и экспорта.**

Основные данные такие как таблицы Актёр, Фильм и Киностудия, содержащие информацию о непосредственных сущностях были импортированные с помощью мастера импорта. Результаты представлены на рисунке 2.

1. **Используя язык SQL.**

Таблицы выражающие отношения между сущностями (Актрё\_Фильм и т.д.) были импортированы с помощью команды BULK INSTERT. Пример команды приведён ниже.

BULK INSERT [dbo].[Актёр\_Фильм]

FROM 'Актёр\_Фильм.csv'

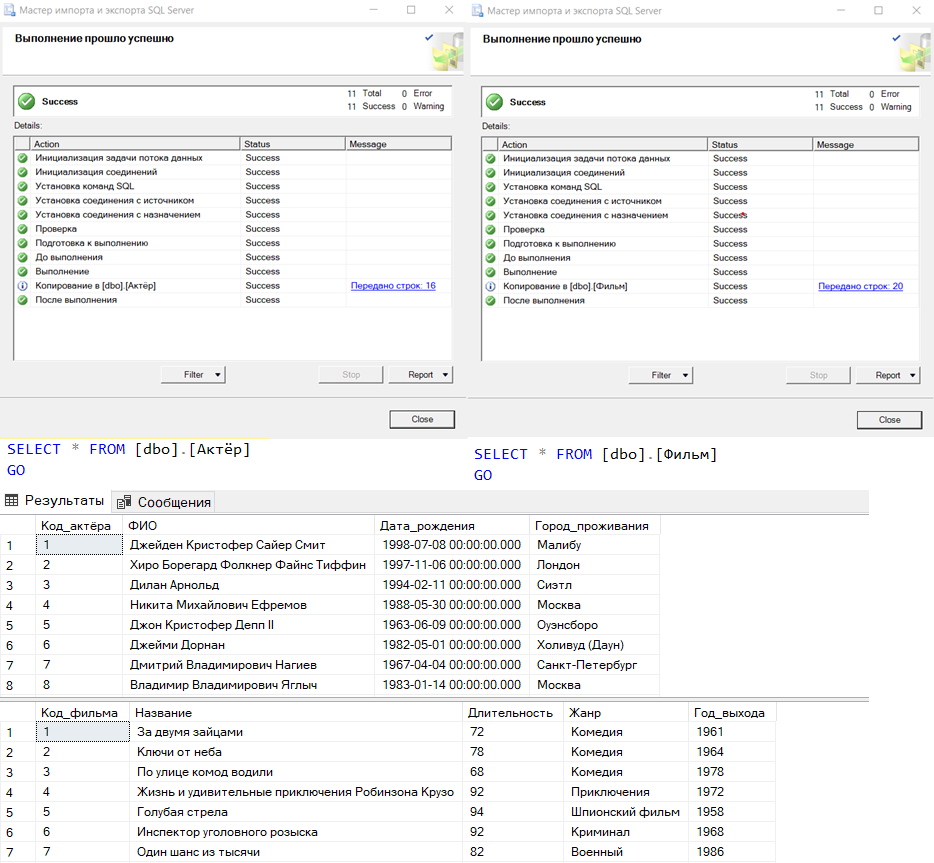
WITH (FORMAT = 'CSV',

CHECK\_CONSTRAINTS,

FIELDTERMINATOR = ';',

FIRSTROW=2

)

  
Рис.2 Импорт данных через мастера.

**Запросы к базе на языке SQL.**

1. получить список всех актеров, снимающихся на заданной киностудии;

WITH [Код\_заданной\_киностудии] AS (

SELECT [Код\_киностудии]

FROM [Киностудия]

WHERE [Название]=<заданная киностудия>

),

[Коды\_искомых\_актёров] AS (

SELECT [Код\_актёра]

FROM [Контракты]

WHERE [Код\_киностудии] IN

(SELECT \* FROM [Код\_заданной\_киностудии] )

)

SELECT [A].\*

FROM [Коды\_ искомых\_актёров] AS [КИА]

INNER JOIN [Актёр] AS [А]

ON [КИА].[Код\_актёра]=[А].[Код\_актёра]

1. получить список кинофильмов, в которых снимаются те же актеры, что и в заданном кинофильме;

WITH [Заданный\_фильм] AS (

SELECT [Код\_фильма]

FROM [Фильм]

WHERE [Название]=<заданный кинофильм>

) ,

[Все\_актёры\_фильма] AS (

SELECT [Код\_актёра]

FROM (

SELECT [АФ].\*

FROM [Заданный\_фильм] AS [ЗФ]

INNER JOIN [Актёр\_Фильм] AS [АФ]

ON [ЗФ].[Код\_фильма]=[АФ].[Код\_фильма]

) AS [Коды\_актёров\_фильма]

),

[Коды\_искомых\_фильмов] AS (

SELECT [Код\_фильма]

FROM (

SELECT [АФ].\*

FROM [Все\_актёры\_фильма] AS [А]

INNER JOIN [Актёр\_Фильм] AS [АФ]

ON [А].[Код\_актёра]=[АФ].[Код\_актёра]

) AS [КФ]

GROUP BY [Код\_фильма]

)

SELECT [Ф].\*

FROM [Фильм] AS [Ф]

INNER JOIN [Коды\_искомых\_фильмов] AS [КИФ]

ON [КИФ].[Код\_фильма]=[Ф].[Код\_фильма]

1. получить пары (Ф.И.О. актера, № контракта), занятых в фильмах, выпущенных на заданной киностудии;

WITH [Код\_заданной\_киностудии] AS (

SELECT [Код\_киностудии]

FROM [Киностудия]

WHERE [Название]=<заданная киностудия>

),

[Коды\_фильмов\_киностудии] AS (

SELECT [Код\_фильма]

FROM [Код\_заданной\_киностудии] AS [КЗК]

INNER JOIN [Киностудия\_Фильм]

ON [КЗК].[Код\_киностудии]=[Киностудия\_Фильм].[Код\_киностудии]

),

[Коды\_актёров\_фильмов] AS (

SELECT [Код\_актёра]

FROM [Коды\_фильмов\_киностудии] AS [КФК]

INNER JOIN [Актёр\_Фильм] AS [АФ]

ON [КФК].[Код\_фильма]=[АФ].[Код\_фильма]

)

SELECT [ФИО], [Номер\_контракта]

FROM [Коды\_актёров\_фильмов] AS [КАФ]

INNER JOIN [Актёр] AS [А]

ON [КАФ].[Код\_актёра]=[А].[Код\_актёра]

INNER JOIN [Контракты] AS [К]

ON [А].[Код\_актёра]=[К].[Код\_актёра]

1. получить список киностудий, в которых были сняты фильмы по заданной тематике;

WITH [Коды\_заданных\_фильмов] AS (

SELECT [Код\_фильма]

FROM [Фильм]

WHERE [Жанр]=<заданная тематика>

)

WITH [Коды\_искомых \_киностудий] AS (

SELECT [Код\_киностудии]

FROM [Коды\_заданных\_фильмов] AS [КЗФ]

INNER JOIN [Киностудия\_Фильм] AS [КФ]

ON [КЗФ].[Код\_фильма]=[КФ].[Код\_фильма]

)

SELECT \*

FROM [Коды\_искомых \_киностудий] AS [КИК]

INNER JOIN [Киностудия] AS [К]

ON [КИК].[Код\_киностудии]=[К].[Код\_киностудии]

1. получить список актеров, у которых имеется контракт с киностудией, расположенной в том же городе, в каком проживают эти актеры;

SELECT [Код\_актёра], [ФИО], [Дата\_рождения], [Город\_проживания]

FROM [Актёр] AS [А]

INNER JOIN [Контракты] AS [Ктр]

ON [А].[Код\_актёра]=[Ктр].[Код\_актёра]

INNER JOIN [Киностудия] AS [Кино]

ON [Ктр].[Код\_киностудии]=[Кино].[Код\_киностудии]

WHERE [Город\_проживания]=[Город\_расположения]

1. получить список кинофильмов, в которых заданный актер не снялся ни разу.

WITH [Коды\_всех\_фильмов\_актёра] AS (

SELECT [Код\_фильма]

FROM [Актёр] AS [А]

INNER JOIN [Актёр\_Фильм] AS [АФ]

ON [А].[Код\_актёра]=[АФ].[Код\_актёра]

WHERE [ФИО]='Дмитрий Владимирович Нагиев'

)

SELECT \*

FROM [Фильм]

WHERE [Код\_фильма] NOT IN

(SELECT [Код\_фильма] FROM [Коды\_всех\_фильмов\_актёра])

**Хранимая процедура по внесению новой записи**

Была создана хранимая процедура по внесению информации о том, что актёр играл в фильме. На вход процедуре должно подаваться ФИО актёра и название фильма, а также опциально информацию об актёре и фильме.

Данная процедура добавляет актёра и фильм в соответствующие таблицы, если таковых в них нет. Кроме того, процедура добавляет информацию о связи в таблицу Актёр\_Фильм, если такая связь ещё не отражена этой в таблице. Код процедуры написан в приложении 3.

**Пользователи БД**

Были созданы следующие пользователи БД:

* Data\_scientist – человек, который будет читать из базы для проведения исследований в области машинного обучения.
* Owner – владелец базы данных. Данный пользователь был выделен с целью возможности продажи базы данных.
* Data\_processor – человек, ответственный за поиск новых данных и загрузку их в базу.

Процесс создания пользователей можно увидеть в приложении 4.

**Шифрование атрибута**

В качестве шифруемого атрибута была выбрана дата заключения контракта. В качестве шифрования был выбран симметричный способ по алгоритму AES\_256. Этот симметричный ключ был зашифрован с помощью ассиметричного ключа, выработанного по алгоритму RSA\_2048. Так же для защиты этого ключа уже был создан пароль для мастерключа SQL. Было создано представление, которое показывает данные в расшифрованном виде. Код скрипта и результаты его работы можно посмотреть в приложении 5.

**Создание резервной копии**

Была создана резервная копия базы данных по простой схеме. После чего база данных была удалена и восстановлена. Рисунок ниже показывает код создания резервной копии и её восстановления.

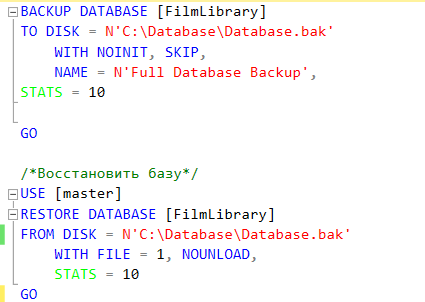
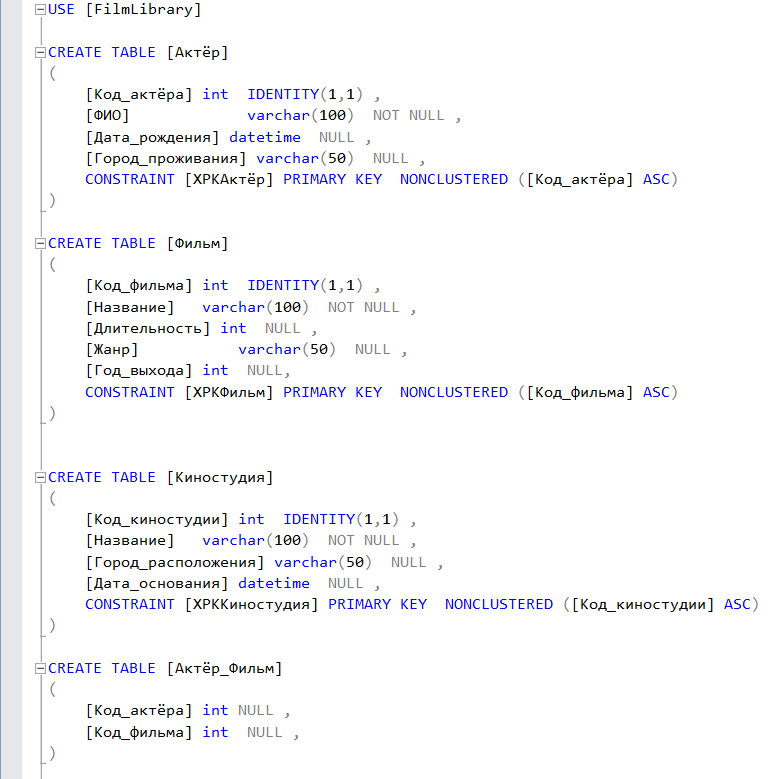
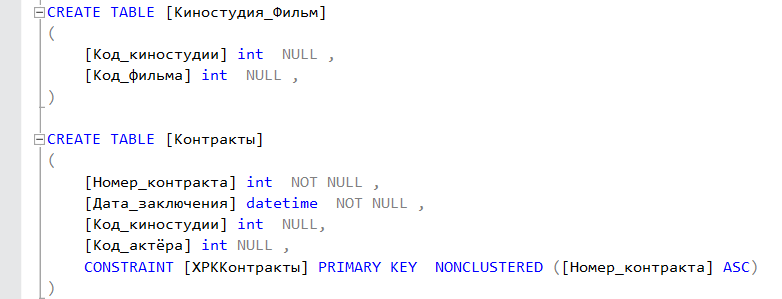
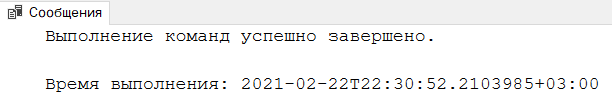


Рис.3 Создание резервной копии и восстановление базы.

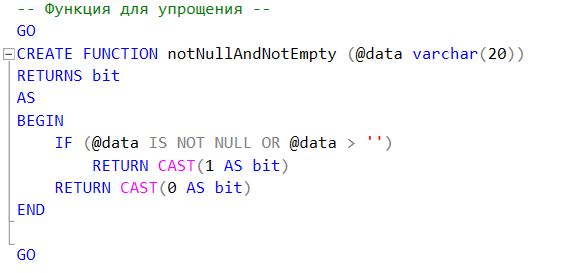
**Приложение 1**



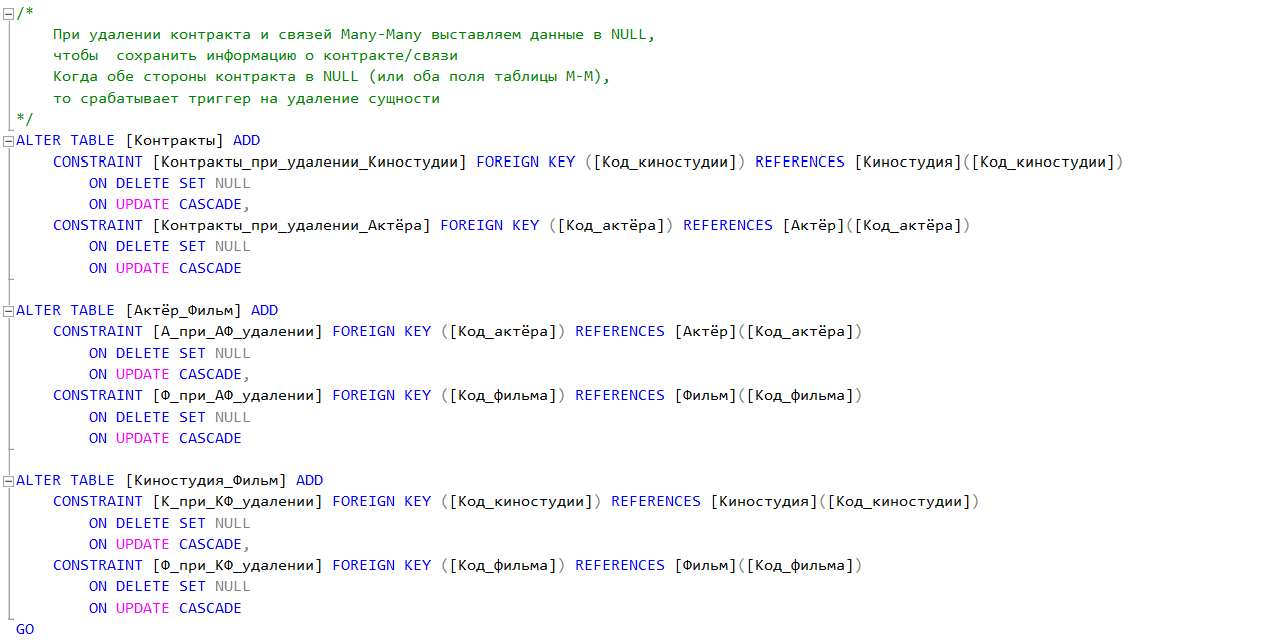


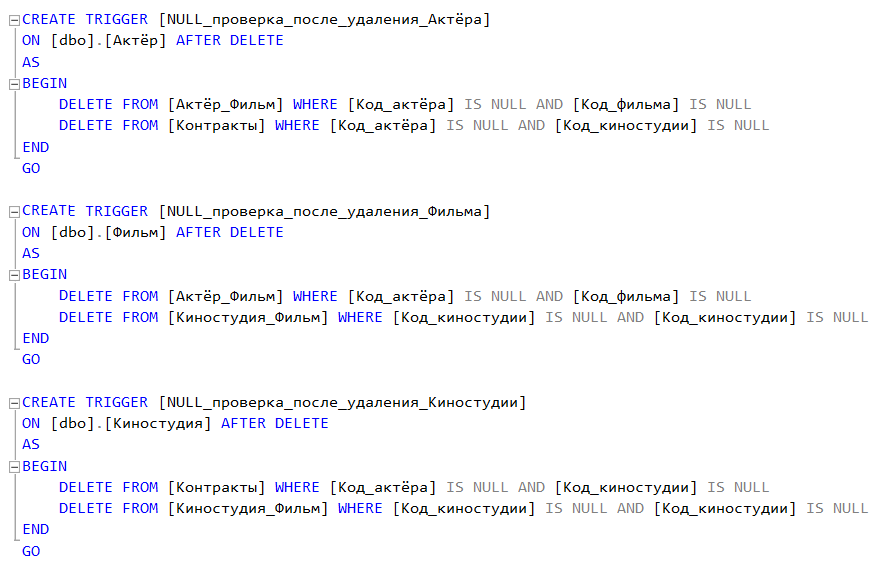


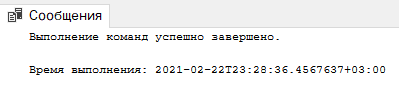
**Приложение 2**



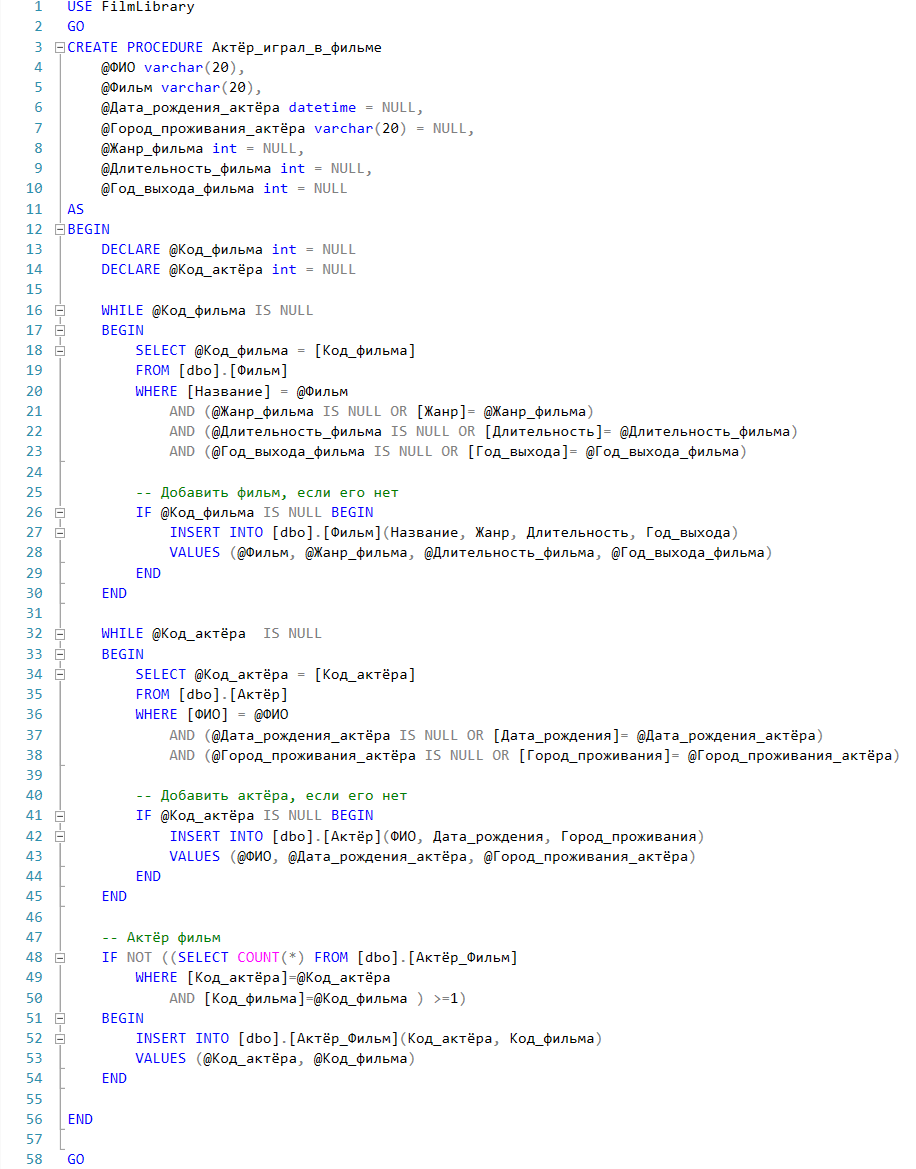




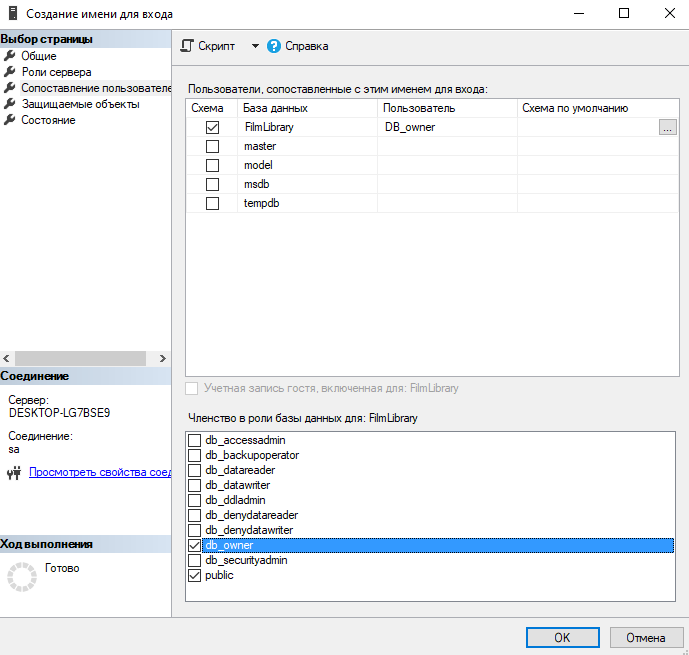


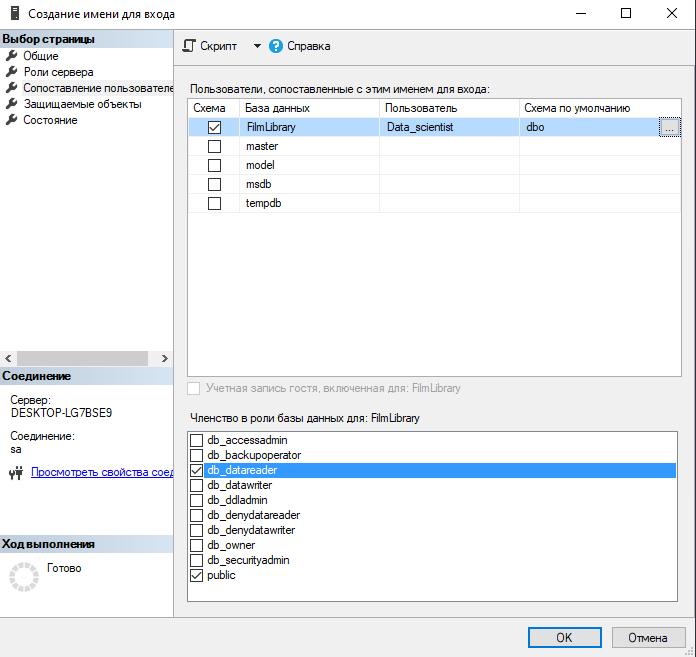


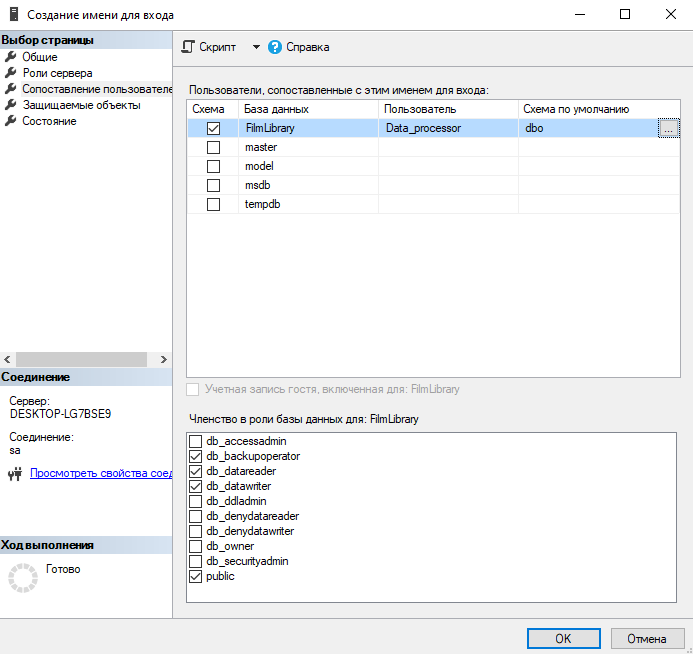
**Приложение 3**



**Приложение 4**







**Приложение 5**

