Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №4**

По дисциплине «Проектирование баз знаний»

Тема: «Хранимые процедуры, функции и триггеры в MS SQL Server»

**Выполнил:**

Студент 3 курса

Группы ИИ-23

Макаревич Н.Р.

**Проверил:**

Савонюк В.А.

Брест 2024

**Цель работы:** Изучение и решение заданий по реализации хранимых процедур, функций и триггеровразных видов и фильтров в базах данных.

**Ход работы:**

**Вариант 2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Вариант 2 | - | Видеотека |  |  |
| 1 | **№клиента** | ФИО | телефон | адрес |
| 2 | **№клиента** | **№фильма** | дата |  |
| 3 | **№фильма** | режиссер | название | год выпуска |

Задание.

1. Изучить материал, приведенный в “К лабораторной работе 4.doc”.

2. На основании логической модели (в соответствии с вариантом Лабораторной работы №1) создать и проверить на работоспособность хранимые процедуры, функции и триггеры для каждой из таблиц БД: реализовать хранимые процедуры, функции и триггеры разных видов для каждой таблицы .

**Создание таблиц:**

CREATE TABLE Clients (

client\_id INT PRIMARY KEY, -- #№клиента

full\_name VARCHAR(255) NOT NULL, -- ФИО

phone VARCHAR(15) NOT NULL, -- телефон

address VARCHAR(255) NOT NULL -- адрес

);

CREATE TABLE Movies (

movie\_id INT PRIMARY KEY, -- #№фильма

director VARCHAR(255) NOT NULL, -- режиссер

title VARCHAR(255) NOT NULL, -- название

release\_year INT NOT NULL -- год выпуска

);

CREATE TABLE Rentals (

rental\_id INT PRIMARY KEY, -- #№клиента (в качестве внешнего ключа)

client\_id INT, -- #№клиента

movie\_id INT, -- #№фильма

rental\_date DATE NOT NULL, -- дата

FOREIGN KEY (client\_id) REFERENCES Clients(client\_id),

FOREIGN KEY (movie\_id) REFERENCES Movies(movie\_id)

);

**Создание новых строк:**

INSERT INTO Clients (client\_id, full\_name, phone, address) VALUES

(1, 'Иванов Иван Иванович', '123-45-67', 'Москва, ул. Ленина, д. 1'),

(2, 'Петров Петр Петрович','734-56-78','Санкт-Петербург, ул. Пушкина, д. 2'),

(3, 'Сидоров Сидор Сидорович', '345-67-89', 'Казань, ул. Гагарина, д. 3'),

(4, 'Кузнецова Анна Сергеевна', '456-78-90', 'Екатеринбург, ул. Чехова, д. 4'),

(5, 'Смирнов Алексей Викторович', '567-89-01', 'Новосибирск, ул. Толстого, д. 5');

INSERT INTO Movies (movie\_id, director, title, release\_year) VALUES

(1, 'Тарковский Андрей', 'Сталкер', 1979),

(2, 'Станислав Говорухин', 'Место встречи изменить нельзя', 1979),

(3, 'Никита Михалков', 'Утомленные солнцем', 1994),

(4, 'Александр Сокуров', 'Русский ковчег', 2002),

(5, 'Дмитрий Киселев', 'Страна глухих', 1998);

INSERT INTO Rentals (rental\_id, client\_id, movie\_id, rental\_date) VALUES

(1, 1, 1, '2023-10-01'),

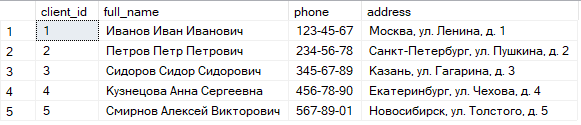
(2, 2, 2, '2023-10-02'),

(3, 3, 3, '2023-10-03'),

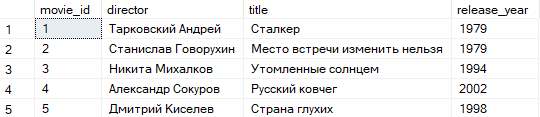
(4, 4, 4, '2023-10-04'),

(5, 5, 5, '2023-10-05');

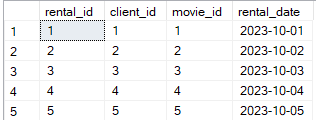
**Таблица клиентов:**



**Таблица фильмов:**



**Таблица билетов:**



**Процедура:**

**-- Создание хранимой процедуры для добавления фильма**

CREATE PROCEDURE добавить\_фильм

@номер\_фильма INT,

@режисер VARCHAR(255),

@название VARCHAR(255),

@год\_выпуска INT

AS BEGIN

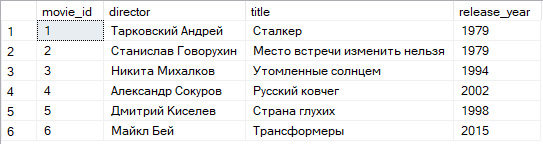
INSERT INTO Movies (movie\_id, director, title, release\_year)

VALUES (@номер\_фильма, @режисер,@название,@год\_выпуска);

END;

**-- Проверка работы процедуры**

EXEC добавить\_фильм @номер\_фильма=6, @режисер="Майкл Бей", @название="Трансформеры", @год\_выпуска=2015;



**-- Создание хранимой процедуры для добавления клиента**

CREATE PROCEDURE добавить\_клиента

@номер INT,

@имя VARCHAR(255),

@телефон VARCHAR(255),

@адрес VARCHAR(255)

AS BEGIN

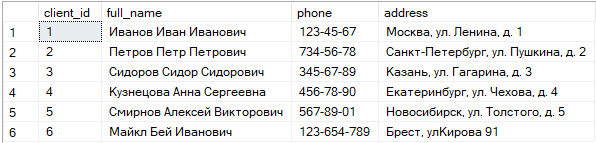
INSERT INTO Clients (client\_id, full\_name, phone, address)

VALUES (@номер, @имя,@телефон,@адрес);

END;

**-- Проверка работы процедуры**

EXEC добавить\_клиента @номер=6, @имя="Майкл Бей Иванович", @телефон="123-654-789", @адрес="Брест, улКирова 91";



**-- Создание хранимой процедуры для добавления билета**

CREATE PROCEDURE добавить\_билета

@номер\_билета INT,

@номер\_клиента INT,

@номер\_фильма INT,

@дата DATE

AS BEGIN

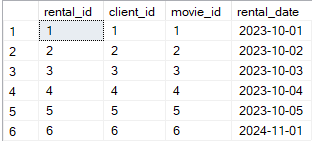
INSERT INTO Rentals (rental\_id, client\_id, movie\_id, rental\_date)

VALUES (@номер\_билета, @номер\_клиента,@номер\_фильма,@дата);

END;

**-- Проверка работы процедуры**

EXEC добавить\_билета @номер\_билета=6, @номер\_клиента=6, @номер\_фильма=6, @дата="2024-11-01";



**Функция:**

**-- Создание функции для подсчета количества фильмов выпущенных в выбранный год:**

CREATE FUNCTION количество\_фильмов (@год INT)

RETURNS INTEGER

AS

BEGIN

DECLARE @количество INTEGER;

SELECT @количество = COUNT(\*) FROM Movies WHERE release\_year = @год;

RETURN @количество;

END;

**-- Проверка работы функции**

SELECT dbo.количество\_фильмов(1979) AS Количество\_фильмов;



**-- Создание функции для получения ФИО клиента по заданному номеру:**

CREATE FUNCTION получить\_имя\_клиента (@client\_id INT)

RETURNS VARCHAR(255)

AS

BEGIN

DECLARE @full\_name VARCHAR(255);

SELECT @full\_name = full\_name

FROM Clients

WHERE client\_id = @client\_id;

RETURN @full\_name;

END;

**-- Проверка работы функции**

SELECT dbo.получить\_имя\_клиента(1) AS ИмяКлиента;



**-- Создание функции для подсчета количества билетов клиента с выбранным номером:**

CREATE FUNCTION количество\_арендованных\_фильмов (@client\_id INT)

RETURNS INT

AS BEGIN

DECLARE @количество INT;

SELECT @количество = COUNT(\*)

FROM Rentals

WHERE client\_id = @client\_id;

RETURN @количество;

END;

**-- Проверка работы функции**

SELECT dbo.количество\_арендованных\_фильмов(2) AS КоличествоАрендованныхФильмов;



**Триггер:**

**-- Создание триггера для проверки наличия билетов у удаляемого клиента:**

CREATE TRIGGER trg\_BeforeDelete\_Clients

ON Clients

INSTEAD OF DELETE

AS BEGIN

IF EXISTS (SELECT \* FROM Rentals WHERE client\_id IN (SELECT client\_id FROM deleted))

BEGIN

RAISERROR('Невозможно удалить клиента, у него есть активные аренды.', 16, 1);

END

ELSE BEGIN

DELETE FROM Clients WHERE client\_id IN (SELECT client\_id FROM deleted);

END

END;

**-Проверка**

DELETE FROM Clients

WHERE phone = '123-45-67';



**-- Создание триггера для проверки отсутствия дубликации названия и года нового фильма:**

CREATE TRIGGER trg\_BeforeInsert\_Movies

ON Movies

INSTEAD OF INSERT AS BEGIN

IF EXISTS (SELECT \* FROM Movies WHERE title IN (SELECT title FROM inserted) AND release\_year IN (SELECT release\_year FROM inserted))

BEGIN

RAISERROR('Фильм с таким названием и годом выпуска уже существует.', 16, 1);

END

ELSE BEGIN

INSERT INTO Movies (movie\_id, director, title, release\_year)

SELECT movie\_id, director, title, release\_year

FROM inserted;

END

END;

**-Проверка**  
INSERT INTO Movies (movie\_id, director, title, release\_year) VALUES

(7, 'Тарковский Андрей', 'Сталкер', 1979);



**Создание триггера для обновления даты билета(нельзя поменять на более ранюю дату)**

CREATE TRIGGER trg\_BeforeUpdate\_Rentals

ON Rentals

INSTEAD OF UPDATE AS BEGIN

IF EXISTS (

SELECT \*

FROM inserted i

JOIN deleted d ON i.rental\_id = d.rental\_id

WHERE i.rental\_date < d.rental\_date

)

BEGIN

RAISERROR('Невозможно установить более раннюю дату аренды.', 16, 1);

END

ELSE BEGIN

UPDATE Rentals

SET client\_id = i.client\_id,

movie\_id = i.movie\_id,

rental\_date = i.rental\_date

FROM inserted i

WHERE Rentals.rental\_id = i.rental\_id;

END

END;

**-Проверка**  
UPDATE Rentals

SET rental\_date = '2022-10-01'

WHERE client\_id = 6;



**Вывод:** Изучил и решил задания по реализации хранимых процедур, функций и триггеровразных видов и фильтров в базах данных.