МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное  
образовательное учреждение высшего образования  
«Самарский национальный исследовательский университет  
имени академика С.П. Королева»  
(Самарский университет)   
  
  
Факультет информатики  
Кафедра программных систем  
  
Дисциплина  
**Базы данных  
  
  
  
ОТЧЕТ**по лабораторной работе №4  
  
Вариант №12

Студенты: Гижевская В.Д.

Петрова А.А.  
Группа: 6313-020302D  
  
Преподаватель: Попова-Коварцева Д.А.  
  
Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2021

**Лабораторная работа №4**

Тема работы: создание и использование триггеров и курсоров.

Цель работы: знакомство с возможностями СУБД по созданию триггеров и курсоров.

Подготовка к работе:

1. Изучите теоретические основы, команды создания последовательностей, триггеров и курсоров рассмотренные в лекционном курсе, методических указаниях или других источниках.
2. Ответьте на вопросы к лабораторной работе.
3. Подготовьте отчет к оформлению хода выполнения работы.

Порядок выполнения работы:

1. Создайте триггер:
2. запрещающий вставку в таблицу новой строки с заданным параметром;
3. запрещающий изменение заданного поля.
4. Внесите изменения в ранее созданный триггер.
5. Создайте необходимые последовательности для суррогатных первичных ключей.
6. Создайте триггеры вставки и изменения записей таблиц, определяющие значение суррогатного первичного ключа по умолчанию.
7. Создайте триггер ведения аудита изменения записей в таблицах.
8. Внесите такие изменения в триггеры вставки и изменения записей таблиц, которые не позволят добавить или изменить записи с дублирующими названиями.
9. Создайте триггеры для не обновляемых представлений, позволяющие изменять данные.
10. Создайте курсор для вывода записей из таблицы, удовлетворяющих заданному условию.
11. Создайте курсор, используемый как выходной параметр процедуры.
12. Оформите отчет о выполнении лабораторной работы.
13. **Триггер:**
14. запрещающий вставку в таблицу новой строки с заданным параметром:

Триггер, запрещающий добавление в таблицу «Происшествие» записи, значение поля «Дата» которой больше текущего

USE [Registration of incident]

GO

CREATE TRIGGER InsertIncident

ON Incident

FOR INSERT

AS

DECLARE @date DATETIME

SELECT @date=Date FROM inserted

IF @date>GETDATE()

BEGIN

ROLLBACK TRANSACTION

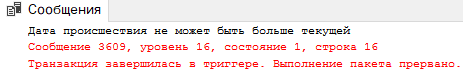
PRINT 'Дата происшествия не может быть больше текущей'

END

GO

INSERT INTO Incident(ID\_incident, Date, Short\_plot, ID\_type, ID\_decision)

VALUES ((SELECT MAX(ID\_incident) FROM Incident)+1,'31-12-2021', 'Гражданин А. напал на гражданина Б.', 8, 3)



1. запрещающий изменение заданного поля:

Триггер, запрещающий изменение поля «Дата» в таблице «Происшествие»

USE [Registration of incident]

GO

CREATE TRIGGER UpdateIncident

ON Incident

FOR UPDATE

AS

IF UPDATE(Date)

BEGIN

ROLLBACK TRANSACTION

PRINT 'Дата происшествия не может быть изменена'

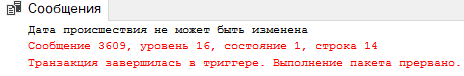
END

GO

UPDATE Incident

SET Date='31-12-2020'

WHERE ID\_incident=3

sql

1. **Внесение изменений в триггер:**

Изменение триггера, запрещающего изменение поля «Дата» в таблице «Происшествие». Теперь данный триггер запрещает изменение поля «Краткая фабула»

USE [Registration of incident]

GO

ALTER TRIGGER UpdateIncident

ON Incident

FOR UPDATE

AS

IF UPDATE(Short\_plot)

BEGIN

ROLLBACK TRANSACTION

PRINT 'Краткая фабула не может быть изменена'

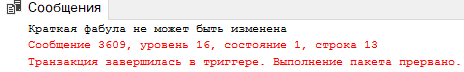
END

GO

UPDATE Incident

SET Short\_plot='Гражданин А. напал на гражданина Б.'

WHERE ID\_incident=3

****

1. **Последовательности для суррогатных первичных ключей:**

USE [Registration of incident]

GO

CREATE SEQUENCE SequenceType

INCREMENT BY 1

START WITH 13



1. **Триггеры вставки и изменения записей таблиц, определяющие значение суррогатного первичного ключа по умолчанию:**
2. Триггер вставки:

Триггер, срабатывающий при добавлении записи в таблицу «Тип происшествия», который определяет значение суррогатного первичного ключа по умолчанию

USE [Registration of incident]

GO

CREATE TRIGGER InsertTypeBySurrogateKey

ON Type\_of\_incident

INSTEAD OF INSERT

AS

BEGIN

INSERT INTO Type\_of\_incident(ID\_type, Name\_type)

SELECT (NEXT VALUE FOR SequenceType), Name\_type

FROM inserted

END

GO

INSERT INTO Type\_of\_incident(Name\_type)

VALUES('Взрыв')

1. Триггер изменения:

Триггер, срабатывающий при изменении записи в таблицу «Тип происшествия», который определяет значение суррогатного первичного ключа по умолчанию

USE [Registration of incident]

GO

CREATE TRIGGER UpdateTypeBySurrogateKey

ON Type\_of\_incident

INSTEAD OF UPDATE

AS

BEGIN

DECLARE @id INT

SELECT @id = (SELECT ID\_type FROM deleted)

DELETE FROM Type\_of\_incident WHERE ID\_type = @id

INSERT INTO Type\_of\_incident(ID\_type, Name\_type) SELECT (NEXT VALUE FOR SequenceType), Name\_type FROM inserted

END

GO

UPDATE Type\_of\_incident

SET ID\_type = 20, Name\_type='Взрыв с обрушением'

WHERE ID\_type = 13

1. **Триггер ведения аудита изменения записей в таблицах:**

Триггер, срабатывающий после изменения записи в таблице «Тип происшествия», который добавляет запись об изменении в таблицу аудита

USE [Registration of incident]

GO

CREATE TABLE AuditType(Date\_update DATETIME NOT NULL, ID\_type\_old INT NULL, ID\_type\_new INT NULL, Name\_type\_old VARCHAR(20) NULL, Name\_type\_new VARCHAR(20) NULL)

GO

CREATE TRIGGER AuditType

ON Type\_of\_incident

AFTER UPDATE

AS

IF UPDATE(Name\_type) OR UPDATE(ID\_type)

BEGIN

DECLARE @Date\_update DATETIME, @ID\_type\_old INT, @ID\_type\_new INT, @Name\_type\_old VARCHAR(20), @Name\_type\_new VARCHAR(20)

SELECT @ID\_type\_old=(SELECT ID\_type FROM deleted)

SELECT @ID\_type\_new=(SELECT ID\_type FROM inserted)

SELECT @Name\_type\_old=(SELECT Name\_type FROM deleted)

SELECT @Name\_type\_new=(SELECT Name\_type FROM inserted)

INSERT INTO AuditType VALUES (GETDATE(), @ID\_type\_old, @ID\_type\_new, @Name\_type\_old, @Name\_type\_new)

END

GO

UPDATE Type\_of\_incident

SET Name\_type = 'Смертельное ДТП'

WHERE Name\_type = 'ДТП'

1. **Внесение изменений в триггеры вставки и изменения записей таблиц, которые не позволят добавить или изменить записи с дублирующими названиями:**
2. Триггер вставки:

Триггер, срабатывающий при добавлении записи с дублирующимся значением поля «Название типа» в таблицу «Тип происшествия

USE [Registration of incident]

GO

ALTER TRIGGER InsertTypeBySurrogateKey

ON Type\_of\_incident

INSTEAD OF INSERT

AS

IF @@ROWCOUNT >= 1

BEGIN

DECLARE @name VARCHAR(20)

SELECT @name=(SELECT Name\_type FROM inserted)

IF EXISTS(SELECT Name\_type FROM Type\_of\_incident WHERE @name = Name\_type)

BEGIN

PRINT 'Такой тип уже имеется. Дублирование данных запрещено'

ROLLBACK TRANSACTION

END

ELSE

INSERT INTO Type\_of\_incident(ID\_type, Name\_type)

SELECT (NEXT VALUE FOR SequenceType), Name\_type

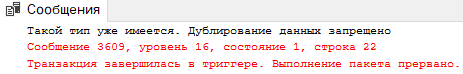
FROM inserted

END

GO

INSERT INTO Type\_of\_incident(Name\_type)

VALUES('Смертельное ДТП')



1. Триггер изменения:

Триггер, срабатывающий при изменении записи с дублирующимся значением поля «Название типа» в таблице «Тип происшествия

USE [Registration of incident]

GO

ALTER TRIGGER UpdateTypeBySurrogateKey

ON Type\_of\_incident

INSTEAD OF UPDATE

AS

IF @@ROWCOUNT >= 1

BEGIN

DECLARE @name VARCHAR(20)

SELECT @name=(SELECT Name\_type FROM inserted)

IF EXISTS(SELECT Name\_type FROM Type\_of\_incident WHERE @name = Name\_type)

BEGIN

PRINT 'Такой тип уже имеется. Дублирование данных запрещено'

ROLLBACK TRANSACTION

END

ELSE

INSERT INTO Type\_of\_incident(ID\_type, Name\_type)

SELECT (NEXT VALUE FOR SequenceType), Name\_type FROM inserted

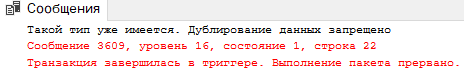
END

GO

UPDATE Type\_of\_incident

SET ID\_type = 20, Name\_type='Взрыв с обрушением'

WHERE ID\_type = 13



1. **Триггеры для не обновляемых представлений, позволяющие изменять данные:**

USE [Registration of incident]

GO

CREATE VIEW AttitudeView

AS

SELECT ID\_attitude, Name\_attitude

FROM Attitude

GROUP BY ID\_attitude, Name\_attitude

GO

CREATE TRIGGER AttitudeViewInsert

ON AttitudeView

INSTEAD OF INSERT

AS

BEGIN

DECLARE @id INT, @name VARCHAR(20)

SELECT @id=(SELECT ID\_attitude FROM inserted)

SELECT @name=(SELECT Name\_attitude FROM inserted)

INSERT INTO Attitude(ID\_attitude, Name\_attitude)

VALUES (@id, @name)

END

GO

INSERT INTO Attitude(ID\_attitude, Name\_attitude)

VALUES ((SELECT MAX(ID\_attitude) FROM Attitude)+1, 'Соучастник



1. **Курсор для вывода записей из таблицы, удовлетворяющих заданному условию:**

Курсор для вывода списка всех участников происшествий, у которых ровно 3 судимости, с указанием количества судимостей и ФИО

USE [Registration of incident]

GO

DECLARE @name VARCHAR(50)

DECLARE @num\_convictions INT

DECLARE @message VARCHAR(100)

PRINT 'Список участников происшествий'

DECLARE ParticipantCursor CURSOR LOCAL FOR

SELECT Name, Number\_of\_convictions

FROM Participant\_in\_incident

WHERE Number\_of\_convictions=3

OPEN ParticipantCursor

FETCH NEXT

FROM ParticipantCursor

INTO @name, @num\_convictions

WHILE @@FETCH\_STATUS=0

BEGIN

SELECT @message='Имя участника: ' + @name + '; Количество судимостей: ' + CAST(@num\_convictions AS VARCHAR(10))

PRINT @message

FETCH NEXT

FROM ParticipantCursor

INTO @name, @num\_convictions

END

CLOSE ParticipantCursor

DEALLOCATE ParticipantCursor



1. **Курсор, используемый как выходной параметр процедуры:**

Курсор для вывода списка всех участников всех происшествий с указанием ФИО

USE [Registration of incident]

GO

ALTER PROCEDURE MyProc

@cursor CURSOR VARYING OUTPUT

AS

BEGIN

DECLARE @name VARCHAR(50)

SET @cursor=CURSOR FORWARD\_ONLY READ\_ONLY

FOR

SELECT Name

FROM Participant\_in\_incident

OPEN @cursor

FETCH NEXT

FROM @cursor

INTO @name

PRINT('Список всех участников происшествий')

WHILE @@FETCH\_STATUS=0

BEGIN

PRINT('Имя участника: ' + @name)

FETCH NEXT

FROM @cursor

INTO @name

END

CLOSE @cursor

DEALLOCATE @cursor

END

GO

DECLARE @MyCursor CURSOR

EXEC MyProc @cursor=@MyCursor OUTPUT

