**Лабораторная работа №16**

**ИСПОЛЬЗОВАНИЕ ТРИГГЕРОВ.**

1. Модифицировать схему базы данных таким образом, чтобы таблица customers хранила актуальную информацию о дате последнего заказа каждого покупателя.

Листинг 1.1 – Модификация customers

ALTER TABLE customers

ADD c\_last\_order DATE NULL DEFAULT NULL

Листинг 1.2 – Добавление данных о последних заказах

UPDATE c

SET

c.c\_last\_order = o.last\_order

FROM customers c

LEFT JOIN (SELECT

o.o\_client,

MAX(o.o\_date) AS last\_order

FROM orders o

GROUP BY o.o\_client) AS o

ON c.c\_id = o.o\_client

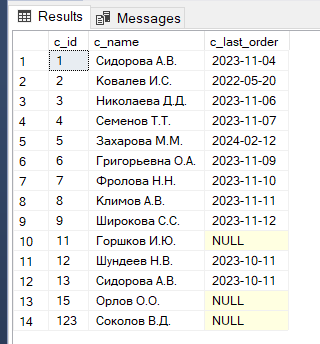


Рис. 1.1 – Вывод информации

Листинг 1.3 – Создание триггера

CREATE TRIGGER last\_order\_on\_customers\_ins\_upd\_del

ON orders

AFTER INSERT, UPDATE, DELETE

AS

UPDATE c

SET

c.c\_last\_order = o.last\_order

FROM customers c

LEFT JOIN (SELECT

o.o\_client,

MAX(o.o\_date) AS last\_order

FROM orders o

GROUP BY o.o\_client) AS o

ON c.c\_id = o.o\_client

Листинг 1.4 – Добавление записи

USE SUPIR;

GO

SET IDENTITY\_INSERT orders ON;

INSERT INTO orders

(o\_id, o\_client, o\_date, o\_status, o\_date\_finish)

VALUES

(200, 11, '2022-01-01', 'Y', NULL)

SET IDENTITY\_INSERT orders OFF;

GO

SELECT

o\_id, o\_client, o\_date, o\_status, o\_date\_finish

FROM orders

WHERE o\_id = 200

SELECT \* FROM customers

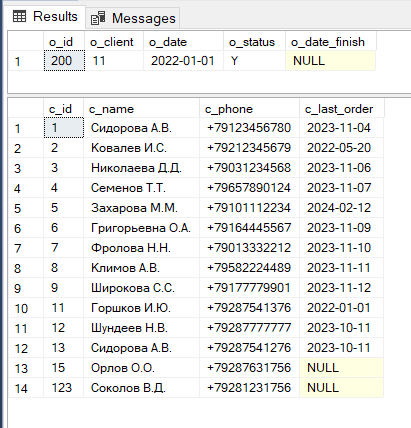


Рис. 1.2 – Вывод информации

Листинг 1.5 – Обновление записи

USE SUPIR;

GO

UPDATE orders

SET o\_client = 15

WHERE o\_id = 200

GO

SELECT

o\_id, o\_client, o\_date, o\_status, o\_date\_finish

FROM orders

WHERE o\_id = 200

SELECT \* FROM customers

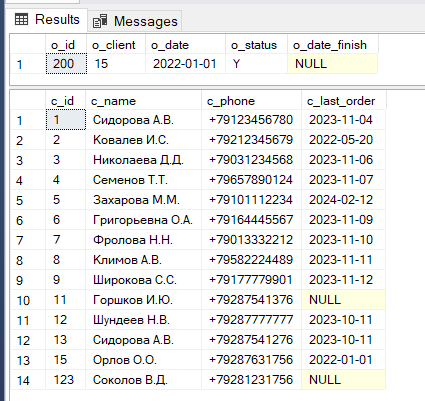


Рис. 1.3 – Вывод информации

Листинг 1.6 – Удаление записи

USE SUPIR;

GO

DELETE FROM orders

WHERE o\_id = 200

GO

SELECT

o\_id, o\_client, o\_date, o\_status, o\_date\_finish

FROM orders

WHERE o\_id > 50

SELECT \* FROM customers

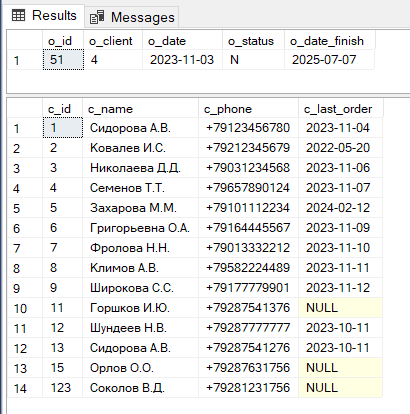


Рис. 1.4 – Вывод информации

1. Модифицировать схему базы данных таким образом, чтобы таблица customers хранила актуальную информацию о том, сколько заказов для каждого клиента со статусом «В пути» (‘N’).

Листинг 2.1 – Модификация customers

ALTER TABLE customers

ADD c\_quantity\_orders INT NOT NULL DEFAULT 0

Листинг 2.2 – Добавление данных

USE SUPIR;

GO

UPDATE c

SET c.c\_quantity\_orders = pd.c\_q\_o

FROM customers c

JOIN ( SELECT

o.o\_client,

COUNT(o.o\_id) AS c\_q\_o

FROM orders o

WHERE o.o\_status = 'N'

GROUP BY o.o\_client) AS pd

ON pd.o\_client = c.c\_id

SELECT \* FROM customers

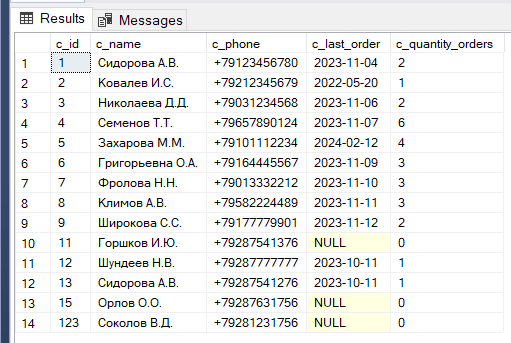


Рис. 2.1 – Вывод информации

Листинг 2.3 – Создание триггера на добавление

CREATE TRIGGER c\_quantity\_orders\_on\_orders\_ins

ON orders

AFTER INSERT

AS

UPDATE c

SET c.c\_quantity\_orders = c.c\_quantity\_orders + pd.new\_order

FROM customers c

JOIN ( SELECT

i.o\_client,

COUNT(i.o\_id) AS new\_order

FROM inserted i

WHERE i.o\_status = 'N'

GROUP BY i.o\_client) AS pd

ON pd.o\_client = c.c\_id

Листинг 2.4 – Создание триггера на удаление

CREATE TRIGGER c\_quantity\_orders\_on\_orders\_del

ON orders

AFTER INSERT

AS

UPDATE c

SET c.c\_quantity\_orders = c.c\_quantity\_orders - pd.old\_order

FROM customers c

JOIN ( SELECT

d.o\_client,

COUNT(d.o\_id) AS old\_order

FROM deleted d

WHERE d.o\_status = 'N'

GROUP BY d.o\_client) AS pd

ON pd.o\_client = c.c\_id

Листинг 2.5 – Создание триггера на обновление

CREATE TRIGGER c\_quantity\_orders\_on\_orders\_upd

ON orders

AFTER UPDATE

AS

UPDATE c

SET c.c\_quantity\_orders = c.c\_quantity\_orders - pd.old\_order

FROM customers c

JOIN ( SELECT

d.o\_client,

COUNT(d.o\_id) AS old\_order

FROM deleted d

WHERE d.o\_status = 'N'

GROUP BY d.o\_client) AS pd

ON pd.o\_client = c.c\_id

UPDATE c

SET c.c\_quantity\_orders = c.c\_quantity\_orders + pd.new\_order

FROM customers c

JOIN ( SELECT

i.o\_client,

COUNT(i.o\_id) AS new\_order

FROM inserted i

WHERE i.o\_status = 'N'

GROUP BY i.o\_client) AS pd

ON pd.o\_client = c.c\_id

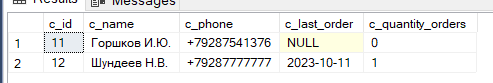


Рис. 2.2 – Вывод пользователей до внесения изменений

Листинг 2.6 – Добавление данных и вывод информации.

SET IDENTITY\_INSERT orders ON

INSERT INTO orders (o\_id, o\_client, o\_date, o\_status, o\_date\_finish)

VALUES

(1011, 11, '2022-10-01', 'N', '2026-10-01'),

(1012, 12, '2022-10-01', 'N', '2026-10-01');

SET IDENTITY\_INSERT orders OFF

SELECT \* FROM customers

WHERE c\_id = 11 OR c\_id = 12

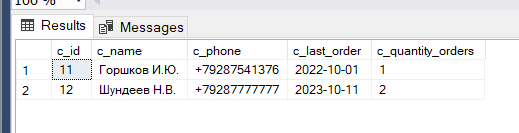


Рис. 2.3 – Вывод пользователей после добавления записей

Листинг 2.7 – Обновление данных и вывод информации.

UPDATE o

SET o.o\_client = 12

FROM orders o

WHERE o.o\_id = 1011

SELECT \* FROM customers

WHERE c\_id = 11 OR c\_id = 12

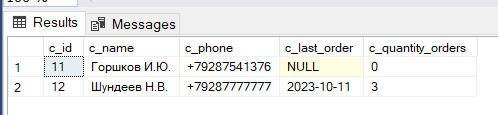


Рис. 2.3 – Вывод пользователей после обновления записей

Листинг 2.8 – Удаление данных и вывод информации.

DELETE FROM orders

WHERE o\_id IN (1011, 1012);

SELECT \* FROM customers

WHERE c\_id = 11 OR c\_id = 12

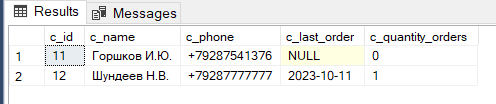


Рис. 2.4 – Вывод пользователей после удаления записей

1. Создать триггер, не позволяющий добавить в базу данных информацию о выдаче книги, если выполняется хотя бы одно из условий:
   1. Дата отправки в будущем;
   2. Дата доставки в прошлом (только для вставленных данных и для записей со статусом “N”)
   3. Дата доставки меньше даты отправки

Листинг 3.1 – Создание триггера

CREATE TRIGGER orders\_control

ON orders

AFTER INSERT, UPDATE

AS

--

DECLARE @bad\_records NVARCHAR(max);

DECLARE @msg NVARCHAR(max);

-- ДАТА ОТПРАВКИ В БУДУЩЕМ

SELECT @bad\_records =

(

SELECT STRING\_AGG(CAST(o\_id AS NVARCHAR) + ' (' + CAST(o\_date AS NVARCHAR) + ')', ', ')

FROM inserted

WHERE o\_date > CONVERT(date, GETDATE())

);

IF LEN(@bad\_records) > 0

BEGIN

SET @msg =

CONCAT('The following orders'' activation dates are in the future', @bad\_records);

RAISERROR(@msg, 16, 1);

ROLLBACK TRANSACTION;

RETURN

END;

--ДАТА ДОСТАВКИ В ПРОШЛОМ

DECLARE @deleted\_records INT;

DECLARE @inserted\_records INT;

SELECT @deleted\_records =

(

SELECT COUNT(\*) FROM deleted

);

SELECT @inserted\_records =

(

SELECT COUNT(\*) FROM inserted

);

SELECT @bad\_records =

(

SELECT STRING\_AGG(CAST(o\_id AS NVARCHAR) + ' (' + CAST(o\_date AS NVARCHAR) + ' )', ', ')

FROM inserted

WHERE o\_date\_finish < CONVERT(date, GETDATE())

AND o\_status = 'N'

);

IF

(

(LEN(@bad\_records) > 0) AND

(@deleted\_records = 0) AND

(@inserted\_records > 0)

)

BEGIN

SET @msg =

CONCAT('The following orders'' deactivation dates are in the past: ', @bad\_records);

RAISERROR(@msg, 16, 1);

ROLLBACK TRANSACTION;

RETURN

END;

-- ДАТА ДОСТАВКИ МЕНЬШЕ ДАТЫ ОТПРАВКИ

SELECT @bad\_records =

(

SELECT STRING\_AGG(CAST(o\_id AS NVARCHAR) + ' (' + CAST(o\_date AS NVARCHAR) + ' )', ', ')

FROM inserted

WHERE o\_date > o\_date\_finish

);

IF LEN(@bad\_records) > 0

BEGIN

SET @msg =

CONCAT('The following orders'' deactivation dates are less than activation dates: ', @bad\_records);

RAISERROR(@msg, 16, 1);

ROLLBACK TRANSACTION;

RETURN

END

Листинг 3.2 – Проверка работы триггера (отправка в будущем)

SET IDENTITY\_INSERT orders ON

INSERT INTO orders (o\_id, o\_client, o\_date, o\_status, o\_date\_finish)

VALUES

(1012, 12, '2026-10-01', 'N', '2028-10-01');

SET IDENTITY\_INSERT orders OFF

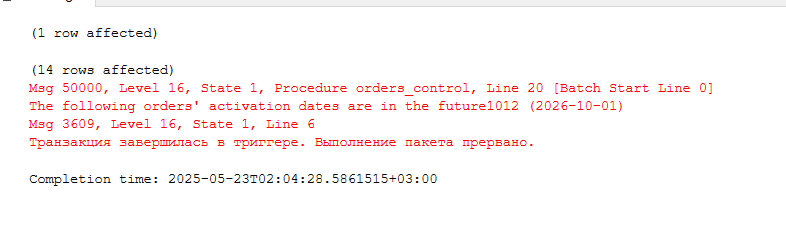


Рис. 3.1 – Проверка работы триггера

Листинг 3.3 – Проверка работы триггера (получение в прошлом)

SET IDENTITY\_INSERT orders ON

INSERT INTO orders (o\_id, o\_client, o\_date, o\_status, o\_date\_finish)

VALUES

(1012, 12, '2026-10-01', 'N', '2028-10-01');

SET IDENTITY\_INSERT orders OFF

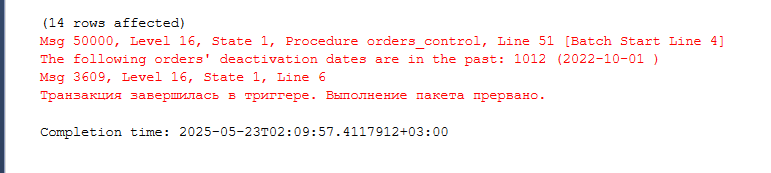


Рис. 3.2 – Проверка работы триггера

Листинг 3.4 – Проверка работы триггера (получение в прошлом)

SET IDENTITY\_INSERT orders ON

INSERT INTO orders (o\_id, o\_client, o\_date, o\_status, o\_date\_finish)

VALUES

(1012, 12, '2024-10-01', 'Y', '2022-10-01');

SET IDENTITY\_INSERT orders OFF

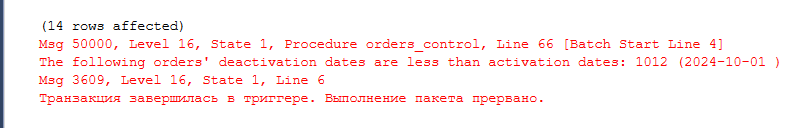


Рис. 3.3 – Проверка работы триггера

1. Создать триггер, проверяющий наличие точки в конце инициалов имени мастера и добавляющий таковую при её отсутствии.

Листинг 4.1 – Создание триггера на добавление

CREATE TRIGGER masters\_name\_lp\_ins

ON masters

INSTEAD OF INSERT

AS

DECLARE @bad\_records NVARCHAR(max);

DECLARE @msg NVARCHAR(max);

SELECT @bad\_records =

(

SELECT STRING\_AGG('[' + m\_name + '] -> [' + m\_name +'.]', ', ')

FROM inserted

WHERE RIGHT(m\_name, 1) <> '.'

);

IF (LEN(@bad\_records) > 0)

BEGIN

SET @msg =

CONCAT('Some values were changed: ', @bad\_records);

PRINT (@msg);

RAISERROR(@msg, 16, 0);

END;

SET IDENTITY\_INSERT masters ON;

INSERT INTO masters (m\_id, m\_name, m\_phone)

SELECT

(

CASE

WHEN m\_id IS NULL OR m\_id = 0

THEN IDENT\_CURRENT('masters') +

IDENT\_INCR('masters') +

ROW\_NUMBER() OVER (ORDER BY (SELECT 1)) -1

ELSE m\_id

END

) AS m\_id,

(

CASE

WHEN RIGHT(m\_name, 1) <> '.'

THEN CONCAT(m\_name, '.')

ELSE m\_name

END

) AS m\_name,

m\_phone

FROM inserted;

SET IDENTITY\_INSERT masters OFF;

Листинг 4.2 – Создание триггера на обновление

CREATE TRIGGER masters\_name\_lp\_upd

ON masters

INSTEAD OF UPDATE

AS

DECLARE @bad\_records NVARCHAR(max);

DECLARE @msg NVARCHAR(max);

IF (UPDATE([m\_id]))

BEGIN

RAISERROR ('Please, do NOT update surrogate PK on table[masters]!', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END;

SELECT @bad\_records =

(

SELECT STRING\_AGG('[' + m\_name + '] -> [' + m\_name +'.]', ', ')

FROM inserted

WHERE RIGHT(m\_name, 1) <> '.'

);

IF (LEN(@bad\_records) > 0)

BEGIN

SET @msg =

CONCAT('Some values were changed: ', @bad\_records);

PRINT @msg;

RAISERROR (@msg, 16, 0);

END;

UPDATE masters

SET masters.m\_name =

(

CASE

WHEN RIGHT(inserted.m\_name, 1) <> '.'

THEN CONCAT(inserted.m\_name, '.')

ELSE inserted.m\_name

END

)

FROM masters

JOIN inserted ON inserted.m\_id = masters.m\_id

Листинг 4.3 – Вставка данных

INSERT INTO masters (m\_name, m\_phone)

VALUES

('Плющев А.Ю', '+79289873685');

SELECT \* FROM masters

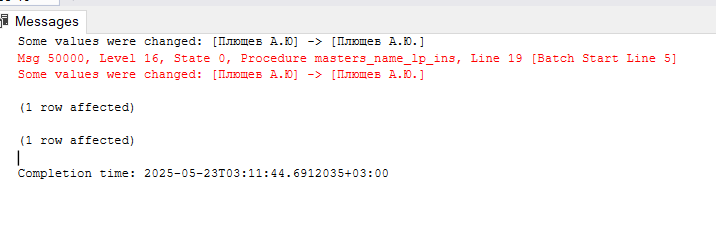


Рис. 4.1 – Сообщение триггера



Рис. 4.2 – Проверка работы триггера