

**Липецкий государственный технический университет**  
**Факультет автоматизации и информатики**  
**Кафедра автоматизированных систем управления**

**ЛАБОРАТОРНАЯ РАБОТА №6**  
**По «Бадам данным»**  
**Программирование серверной логики БД**

Студент  
Группа АИ-20

Глубоков Г.В.

Руководитель  
Доцент , к. т. н.

Алексеев В.А.

Липецк 2022 г.

## Цель работы

Изучить возможности программирования серверной логики базы данных с использованием триггеров и хранимых процедур. Получить практические навыки программирования триггеров и хранимых процедур для выбранной СУБД.

### Задание кафедры

Реализовать в БД триггеры для поддержки бизнес-логики информационной системы (не менее 2-х). Реализовать хранимую процедуру для транзакции, разработанной в лабораторной работе №5. Проверить правильность работы созданных объектов БД.

По желанию могут быть реализованы также пользовательские (хранимые) функции БД.

## Ход работы

### 1 Триггеры

#### 1.1 Первый триггер

Создадим триггер, который корректирует вводимые данные в таблице «отели».

Мы не можем добавлять данные о кол-ве звёзд отеля, если вводимое число больше 6 или меньше 1. Если число звёзд отеля больше 6 или меньше 1, то триггер блокирует добавление данных в таблицу и сообщает об ошибке.

отели		
код	отеля	код
города		
название	отеля	
класс	отеля	

Рисунок – 1. Фрагмент физической схемы данных Код для создания функции триггера:

```
CREATE FUNCTION check_class_hotel()
RETURNS TRIGGER
LANGUAGE PLPGSQL
AS
$$
BEGIN
    IF NEW.class_hotel >= 6 OR NEW.class_hotel < 1 THEN RAISE EXCEPTION
'Количество звёзд отеля не может быть больше 6 или меньше 1';
END IF;
RETURN NEW;
END;
$$
```

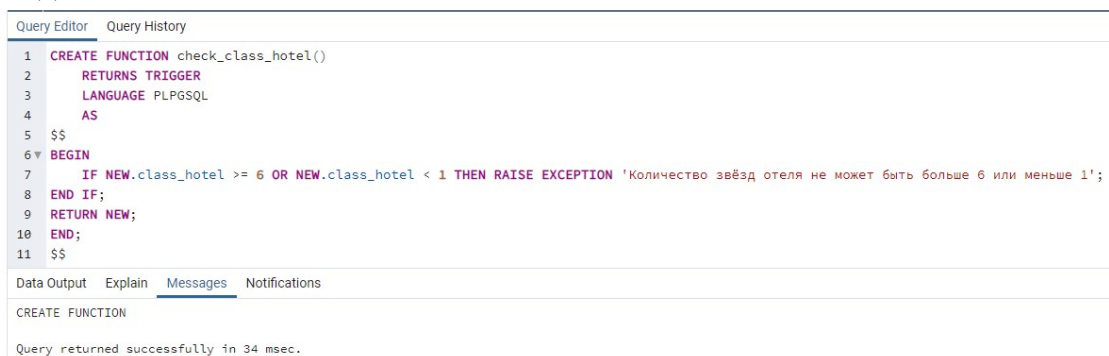


Рисунок – 2. Успешное создание функции

Код для создания триггера: CREATE  
TRIGGER check\_class  
AFTER INSERT OR UPDATE  
ON hotels

```
FOR EACH ROW
EXECUTE PROCEDURE check_class_hotel();
```

```
1 CREATE TRIGGER check_class
2   AFTER INSERT OR UPDATE
3   ON hotels
4   FOR EACH ROW
5   EXECUTE PROCEDURE check_class_hotel();
```

Data Output Explain Messages Notifications

CREATE TRIGGER

Query returned successfully in 34 msec.

Рисунок – 3. Успешное создание триггера

Попробуем добавить в таблицу данные, которые приведут к срабатыванию триггера:

```
Query Editor Query History
1 INSERT INTO hotels(id_hotel,id_city,name_hotel,class_hotel,sale_hotel)
2 VALUES (14,1,'TEST_HOTEL',-10,'3200');
3
```

Data Output Explain Messages Notifications

ERROR: ОШИБКА: Количество звёзд отеля не может быть больше 6 или меньше 1  
CONTEXT: функция PL/pgSQL check\_class\_hotel(), строка 3, оператор RAISE

SQL state: P0001

Рисунок – 4. Ввод неверных данных

Введём данные, которые удовлетворяют условию:

```
Query Editor Query History
1 INSERT INTO hotels(id_hotel,id_city,name_hotel,class_hotel,sale_hotel)
2 VALUES (14,1,'TEST_HOTEL',3,'3200');
3
```

Data Output Explain Messages Notifications

INSERT 0 1

Query returned successfully in 30 msec.

Рисунок – 5. Ввод верных данных

## 1.2 Второй триггер

Создадим триггер, который корректирует вводимые данные в таблице «туры».

Дата окончания тура не может быть раньше даты начала. Поэтому, если дата окончания тура меньше даты начала, то триггер блокирует добавление данных в таблицу и сообщает об ошибке. Таблица, к которой применяется триггер:

туры
код тура код отеля код транспорта код города цена тура название тура описание тура дата начала тура дата окончания тура

Рисунок – 6. Таблица «туры»

Код для создания функции триггера:

```
CREATE FUNCTION check_data_tours()
  RETURNS TRIGGER
  LANGUAGE PLPGSQL
  AS
$$
BEGIN
  IF NEW.date_start_tour > NEW.date_end_tour THEN
    RAISE EXCEPTION 'Дата окончания тура не может быть меньше даты начала';
  END IF;

  RETURN NEW;
END;
$$
```

The screenshot shows a database query editor with a 'Query Editor' tab. The SQL code is as follows:

```
1 CREATE FUNCTION check_data_tours()
2 RETURNS TRIGGER
3 LANGUAGE PLPGSQL
4 AS
5 $$
6 BEGIN
7 IF NEW.start_date_tour > NEW.end_date_tour THEN
8 RAISE EXCEPTION 'Дата окончания тура не может быть меньше даты начала';
9 END IF;
10
11 RETURN NEW;
12 END;
13 $$
14
15 |
16 --SELECT * FROM tours
```

Below the editor, there are tabs for 'Data Output', 'Explain', 'Messages', and 'Notifications'. The 'Messages' tab is selected, showing the message: 'CREATE FUNCTION' and 'Query returned successfully in 36 msec.'

Рисунок – 7. Успешное создание функции

Код для создания триггера:

```
CREATE TRIGGER check_dataInTours
AFTER INSERT ON tour
FOR EACH ROW
EXECUTE PROCEDURE check_data_tours();
```

The screenshot shows a database query editor with a 'Query Editor' tab. The SQL code is as follows:

```
1 CREATE TRIGGER check_dataInTours
2 AFTER INSERT ON tour
3 FOR EACH ROW
4 EXECUTE PROCEDURE check_data_tours();
5
6
7 --SELECT * FROM tour
```

Below the editor, there are tabs for 'Data Output', 'Explain', 'Messages', and 'Notifications'. The 'Messages' tab is selected, showing the message: 'CREATE TRIGGER' and 'Query returned successfully in 34 msec.'

Рисунок – 8. Успешное создание триггера

Попытаемся добавить в таблицу данные, которые приведут к срабатыванию триггера:

```
Query Editor  Query History

1  INSERT INTO tour (id_tour,id_hotel,id_transport,id_city,sale_tour,name_tour,description_tour,date_start_tour,date_end_tour)
2  VALUES(12,11,3,11,'50000','TEST TOUR','TEST_DESCRIPTION','01-01-2022','01-01-2021');
3

Data Output  Explain  Messages  Notifications

ERROR: ОШИБКА: Дата окончания тура не может быть меньше даты начала
CONTEXT:  функция PL/pgSQL check_datatours(), строка 4, оператор RAISE
```

Рисунок – 9. Попытка добавления неверных данных в таблицу «туры»

Введём в таблицу данные, которые не приведут к срабатыванию триггера:

```
Query Editor  Query History

1  INSERT INTO tour (id_tour,id_hotel,id_transport,id_city,sale_tour,name_tour,description_tour,date_start_tour,date_end_tour)
2  VALUES(13,11,3,11,'50000','TEST TOUR_2','TEST_DESCRIPTION_2','01-01-2021','01-01-2022');
3

Data Output  Explain  Messages  Notifications

INSERT 0 1

Query returned successfully in 43 msec.
```

Рисунок – 10. Ввод верных данных в таблицу «туры»



## 2 Транзакция

Необходимо реализовать хранимую процедуру с использованием транзакции.

Создадим процедуру для удаления информации о городе и связанных с ним данных.

В процессе работы процедуры мы находим город с необходимым ID. После чего мы удаляем информацию о городе в таблице «отели», «туры» и затем в «городах»

Структура таблиц и связей, затрагиваемая процедурой удаления записей.

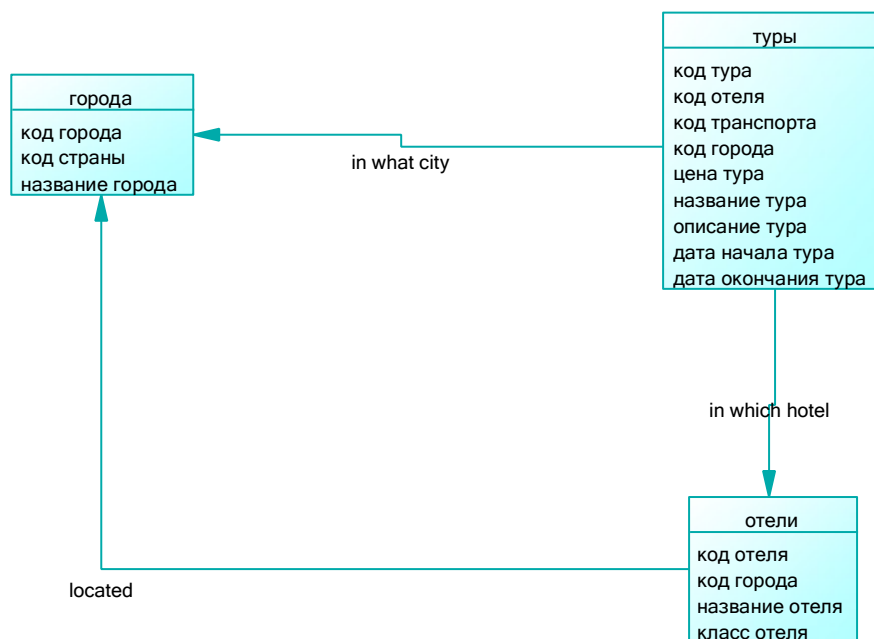


Рисунок – 11. Фрагмент структуры таблиц, обрабатываемый в процедуре

Код для создания процедуры:

```
CREATE PROCEDURE delete_city(city_id INT)
  LANGUAGE PLPGSQL
  AS
  $$
  DECLARE
  BEGIN
    DELETE FROM hotels WHERE id_city = city_id;
    DELETE FROM tour WHERE id_city = city_id;
    DELETE FROM city WHERE id_city = city_id;
  END;
  $$;
```

```
Query Editor  Query History

1  CREATE PROCEDURE delete_city(city_id INT)
2      LANGUAGE PLPGSQL
3      AS
4  $$
5  DECLARE
6
7  BEGIN
8      delete from hotels where id_city = city_id;
9      delete from tour  where id_city = city_id;
10     delete from citie  where id_city = city_id;
11 END;
12 $$;

Data Output  Explain  Messages  Notifications

CREATE PROCEDURE

Query returned successfully in 30 msec.
```

## Рисунок – 12. Создание процедуры

Проверим работу транзакции на тестовых данных:

## Попробуем удалить город с id = 10



Data Output		Explain	Messages	Notifications			
	id_tour [PK] integer 	id_hotel integer 	id_transport integer 	id_city integer 	sale_tour numeric (8,2) 	name_tour character varying (50) 	description_tour text
1	4	4	10	4	90000.00	Novotel Beijing	[null]
2	5	7	1	7	30000.00	Holiday Inn	[null]
3	6	12	3	1	60000.00	Millésime	[null]
4	7	10	6	10	73000.00	Riviera Travel	[null]
5	10	6	8	6	91000.00	Camino Real Aeropuerto	[null]
6	11	11	4	11	70400.00	Grand Via	[null]

Рисунок – 13. Таблица «туры» до исполнения транзакции

	Data Output	Explain	Messages	Notifications
	 id_hotel [PK] integer 	id_city integer 	name_hotel character varying (50) 	class_hotel integer 
1	1	1	Измайлово	3
2	2	2	Mercure Tokyo Ginza	4
3	3	3	Sunworld Hotel	4
4	4	4	Hyatt House	3
5	5	5	Fairmont Nile City	5
6	6	6	Pepper My Love	3
7	7	7	Lord Elgin Hotel	4
8	8	8	GLAD Yeouido	4
9	9	9	Riad Dar Rabiaa	3
10	10	10	Hotel Brasil 21	4
11	11	11	Pestana CR7 Gran Vía Madrid	4
12	12	12	Hotel Trianon Rive Gauche	4
13	14	1	TEST_HOTEL	3

Рисунок – 14. Таблица «отели» до исполнения транзакции





Data Output		Explain	Messages	Notifications
	<b>id_city</b> [PK] integer 	<b>id_country</b> integer 	<b>name_city</b> character varying (50) 	
1	1	1	Москва	
2	2	2	Токио	
3	3	3	Пекин	
4	4	4	Вашингтон	
5	5	5	Каир	
6	6	6	Мехико	
7	7	7	Оттава	
8	8	8	Сеул	
9	9	9	Рабат	
10	10	10	Бразилиа	
11	11	11	Мадрид	
12	12	12	Париж	

Рисунок – 15. Таблица «города» до исполнения транзакции

```
1 CALL delete_city(10);
```

Data Output Explain Messages Notifications

CALL

Query returned successfully in 32 msec.

Рисунок – 16. Успешно выполненная процедура

	Data Output	Explain	Messages	Notifications
				
	id_hotel [PK] integer	id_city integer	name_hotel character varying (50)	class_hotel integer
1	1	1	Измайлово	3
2	2	2	Mercure Tokyo Ginza	4
3	3	3	Sunworld Hotel	4
4	4	4	Hyatt House	3
5	5	5	Fairmont Nile City	5
6	6	6	Pepper My Love	3
7	7	7	Lord Elgin Hotel	4
8	8	8	GLAD Yeouido	4
9	9	9	Riad Dar Rabiaa	3
10	11	11	Pestana CR7 Gran Vía Madrid	4
11	12	12	Hotel Trianon Rive Gauche	4
12	14	1	TEST_HOTEL	3

Рисунок – 17. Таблица «отели» после вызова процедуры








Data Output		Explain	Messages	Notifications			
	 id_tour [PK] integer 	id_hotel integer 	id_transport integer 	id_city integer 	sale_tour numeric (8,2) 	name_tour character varying (50) 	
1	4	4	10	4	90000.00	Novotel Beijing	
2	5	7	1	7	30000.00	Holiday Inn	
3	6	12	3	1	60000.00	Millésime	
4	10	6	8	6	91000.00	Camino Real Aeropuerto	
5	11	11	4	11	70400.00	Grand Via	
6	2	2	2	2	80000.00	Japan Wonder Travel	
7	1	1	5	2	50000.00	Grand Muthu Cayo	
8	3	5	7	5	30000.00	Swiss Inn Pyramids	
9	8	8	2	8	85000.00	Conrad tr	
10	13	11	3	11	50000.00	TEST_TOUR_2	

Рисунок – 18. Таблица «туры» после вызова процедуры

	Data Output	Explain	Messages	Notifications
	id_city [PK] integer	id_country integer	name_city character varying (50)	
1	1	1	Москва	
2	2	2	Токио	
3	3	3	Пекин	
4	4	4	Вашингтон	
5	5	5	Каир	
6	6	6	Мехико	
7	7	7	Оттава	
8	8	8	Сеул	
9	9	9	Рабат	
10	11	11	Мадрид	
11	12	12	Париж	

Рисунок – 19. Таблица «города» после вызова процедуры

Как мы видим, город с id = 10 был удалён из трёх таблиц, следовательно процедура отработала верно.

## Вывод

В ходе работы получены практические навыки работы с хранимыми процедурами и триггерами базы данных PostgreSQL, составляющих основу для описания и разработки логики обработки данных на стороне PostgreSQL.