

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИТМО»**

Отчет

По Лабораторной работе 3
«Процедуры, функции, триггеры в PostgreSQL»
по дисциплине «Проектирование и реализация баз данных»

Автор: Казанков И

Факультет: ИКТ

Группа: K32402

Преподаватель: Говорова М. М.

Цель работы: овладеть практическими создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

Оборудование: компьютерный класс.

Программное обеспечение: СУБД PostgreSQL, SQL Shell (psql).

Практическое задание:

Вариант 1

2. Создать процедуры/функции согласно индивидуальному заданию и (согласно индивидуальному заданию, часть 4).
- Создать триггер для логирования событий вставки, удаления, редактирования данных в базе данных PostgreSQL (согласно индивидуальному заданию, часть 5). Допустимо создать универсальный триггер или отдельные триггеры на логирование действий.
 -

Выполнение:

Хранимые процедуры:

- 1 - Для повышения цен в пригородные поезда на 20%:

```
1 CREATE OR REPLACE PROCEDURE increase_suburban_prices()  
2 AS  
3 $$  
4 BEGIN  
5     UPDATE ticket  
6     SET price_ticket = price_ticket * 1.2;  
7 END;  
8 $$  
9 LANGUAGE plpgsql;  
10
```

Data Output Сообщения Notifications

CREATE PROCEDURE

Запрос завершён успешно, время выполнения: 371 msec.

1	SELECT price_ticket	1	CALL increase_suburban_prices();
2	FROM ticket;	2	SELECT price_ticket
		3	FROM ticket;

Data Output	Сообщения	Notifications
price_ticket integer		
1	10	
2	15	
3	20	

Data Output	Сообщения	Notifications
price_ticket integer		
1	12	
2	18	
3	24	

2 - Для создания нового рейса на поезд:

```

1 CREATE OR REPLACE PROCEDURE create_new_train_route(
2   p_arrival_time TIME,
3   p_departure_time TIME,
4   p_train_arrival_point VARCHAR(40),
5   p_train_departure_point VARCHAR(40),
6   p_train_name VARCHAR(20),
7   p_arrival_date_time TIMESTAMP,
8   p_departure_date_time TIMESTAMP,
9   p_type_train VARCHAR(20)
10 )
11 AS $$
12 DECLARE
13   v_train_number INTEGER;
14   v_train_id INTEGER;
15 BEGIN
16   INSERT INTO regular_schedule (arrival_time, departure_time, train_arrival_point, train_departure_point)
17   VALUES (p_arrival_time, p_departure_time, p_train_arrival_point, p_train_departure_point)
18   RETURNING train_number INTO v_train_number;
19
20   INSERT INTO train (train_name, train_number, arrival_date_time, departure_date_time, type_train)
21   VALUES (p_train_name, v_train_number, p_arrival_date_time, p_departure_date_time, p_type_train)
22   RETURNING train_id INTO v_train_id;
23
24   RAISE NOTICE 'Created new train flight with train_number %', v_train_number;
25 END;
26 $$ LANGUAGE plpgsql;

```

Data Output Сообщения Notifications

CREATE PROCEDURE

Запрос завершён успешно, время выполнения: 92 msec.

1	SELECT train.train_number, train.train_id, regular_schedule.arrival_time, regular_schedule.departure_
2	FROM train
3	JOIN regular_schedule ON train.train_number = regular_schedule.train_number;
4	

Data Output	Сообщения	Notifications
train_number integer	train_id integer	arrival_time time without time zone
1	1	10:00:00
		departure_time time without time zone
		12:00:00
		train_arrival_point character varying (40)
		Station A
		train_departure_point character varying (40)
		Station B

```

1 CALL create_new_train_route (
2   '10:00',
3   '12:00',
4   'Иркутск',
5   'Омск',
6   'Train 123',
7   '2023-06-30 08:00:00',
8   '2023-06-30 11:00:00',
9   'Express'
10 );

```

Data Output Сообщения Notifications

NOTICE: Created new train flight with train_number 3
CALL

Запрос завершён успешно, время выполнения: 76 мсек.

```

1 SELECT train.train_number, train.train_id, regular_schedule.arrival_time, regular_schedule.dep:
2 FROM train
3 JOIN regular_schedule ON train.train_number = regular_schedule.train_number;
4

```

Data Output Сообщения Notifications

	train_number integer	train_id integer	arrival_time time without time zone	departure_time time without time zone	train_arrival_point character varying (40)	train_departure_point character varying (40)
1	1	1	10:00:00	12:00:00	Station A	Station B
2	2	2	10:00:00	12:00:00	Иркутск	Омск

3 - Для формирования общей выручки по продаже билетов за сутки.

```

1 CREATE OR REPLACE FUNCTION calculate_daily_revenue(date_param DATE)
2 RETURNS INTEGER AS $$
3 DECLARE
4     total_revenue INTEGER;
5 BEGIN
6     SELECT SUM(price_ticket)
7     INTO total_revenue
8     FROM ticket
9     WHERE date_of_purchase = date_param;
10
11 IF total_revenue IS NULL THEN
12     total_revenue := 0;
13 END IF;
14
15 RETURN total_revenue;
16 END;
17 $$ LANGUAGE PLPGSQL;

```

Data Output Сообщения Notifications

CREATE FUNCTION

Запрос завершён успешно, время выполнения: 62 msec.

```

1 SELECT calculate_daily_revenue('2023-06-29');

```

Data Output Сообщения Notifications

<div> <div>☰+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>▼</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> </div>	
	<div>calculate_daily_revenue</div> <div>integer</div> <div>🔒</div>
1	0

Создание триггера для логирования событий вставки, удаления и редактирования данных в базе данных PostgreSQL

Создание триггера для таблицы regular_schedule:

```

1 CREATE TABLE log_regular_schedule (
2     log_id SERIAL PRIMARY KEY,
3     event_type VARCHAR(20) NOT NULL,
4     table_name VARCHAR(50) NOT NULL,
5     record_id INTEGER,
6     event_time TIMESTAMP DEFAULT NOW(),
7     user_name VARCHAR(50)
8 );
9
10 CREATE OR REPLACE FUNCTION log_event_regular_schedule()
11     RETURNS TRIGGER AS
12 $BODY$
13 BEGIN
14     IF (TG_OP = 'INSERT') THEN
15         INSERT INTO log_regular_schedule (event_type, table_name, record_id, user_name)
16         VALUES ('INSERT', TG_TABLE_NAME, NEW.train_number, current_user);
17         RETURN NEW;
18     ELSIF (TG_OP = 'DELETE') THEN
19         INSERT INTO log_regular_schedule (event_type, table_name, record_id, user_name)
20         VALUES ('DELETE', TG_TABLE_NAME, OLD.train_number, current_user);
21         RETURN OLD;
22     ELSIF (TG_OP = 'UPDATE') THEN
23         INSERT INTO log_regular_schedule (event_type, table_name, record_id, user_name)
24         VALUES ('UPDATE', TG_TABLE_NAME, NEW.train_number, current_user);
25         RETURN NEW;
26     END IF;
27     RETURN NULL;
28 END;
29 $BODY$
30 LANGUAGE plpgsql;
31
32 CREATE TRIGGER regular_schedule_trigger
33 AFTER INSERT OR DELETE OR UPDATE ON regular_schedule
34 FOR EACH ROW
35 EXECUTE FUNCTION log_event_regular_schedule();
36

```

Вставляем данные в таблицу для проверки триггера

```

1 INSERT INTO regular_schedule (arrival_time, departure_time, train_arrival_point, train_departure_point)
2 VALUES ('10:00:00', '12:00:00', 'Point A', 'Point B');

```

Data Output Сообщения Notifications

INSERT 0 1

Запрос завершён успешно, время выполнения: 51 мсек.

```

1 SELECT * FROM log_regular_schedule;
2

```

Data Output Сообщения Notifications

	log_id [PK] integer	event_type character varying (20)	table_name character varying (50)	record_id integer	event_time timestamp without time zone	user_name character varying (50)
1	3	INSERT	regular_schedule	5	2023-06-29 09:22:54.644587	postgres
2	4	INSERT	regular_schedule	6	2023-06-29 09:25:29.152083	postgres

Выводы:

По результатам данной лабораторной работы были получены навыки создания функций, процедур и триггеров в PostgreSQL, созданы необходимые функции в соответствии с заданием, а также авторский триггер.