

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего
образования

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет инфокоммуникационных технологий

ОТЧЕТ

О ЛАБОРАТОРНОЙ РАБОТЕ № 3

по теме: процедуры, функции, триггеры в PostgreSQL
по дисциплине: Проектирование и реализация баз данных

Специальность: 45.03.04 Интеллектуальные системы в гуманитарной сфере

Проверила:

Говорова М.М. _____

Выполнила: студентка К3243,
Куканова У.Д. _____

Санкт-Петербург 2021/2022

Создать хранимые процедуры:

- 1) Для снижения цен на книги, которые находятся на базе в количестве, превышающем 1000 штук.

До применения процедуры:

```
1 select * from bookstore.edition
```

	edition [PK] bigint	edition_date date	copies integer	price integer	ISBN bigint	copies_left integer
1	6547836	2020-12-10	5000	683	134610997	[null]
2	8568030	2022-03-03	100	200	2346814907856	[null]
3	7784321	2022-05-24	300	500	5845907888	[null]
4	3146528	2022-04-11	950	1020	9785819907856	[null]
5	1334657	1986-12-03	1500	500	9785845913494	[null]
6	2456781	2012-07-12	8200	800	262017431	[null]
7	3428564	2004-12-31	1200	690	321197844	[null]

После:

```
bookstore/postgres@PostgreSQL 11 v
```

```
1 create procedure lower_1000()  
2 language sql  
3 as $$  
4 update bookstore.edition set price = price - 200  
5 where copies > 1000;  
6 $$;  
7 call lower_1000();  
8 select * from bookstore.edition;  
9
```

	edition [PK] bigint	edition_date date	copies integer	price integer	ISBN bigint	copies_left integer
1	8568030	2022-03-03	100	200	2346814907856	[null]
2	7784321	2022-05-24	300	500	5845907888	[null]
3	3146528	2022-04-11	950	1020	9785819907856	[null]
4	6547836	2020-12-10	5000	483	134610997	[null]
5	1334657	1986-12-03	1500	300	9785845913494	[null]
6	2456781	2012-07-12	8200	600	262017431	[null]
7	3428564	2004-12-31	1200	490	321197844	[null]

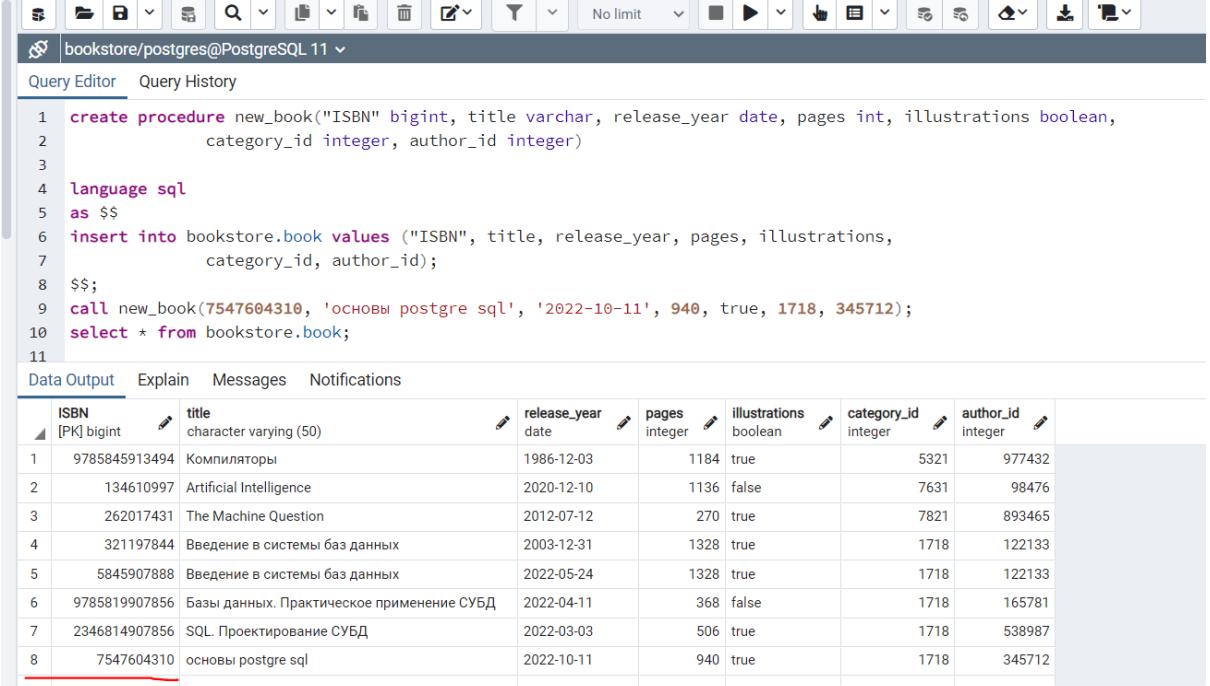
create procedure lower_1000()

```

language sql
as $$
update bookstore.edition set price = price - 200
where copies > 1000;
$$;
call lower_1000();
select * from bookstore.edition;

```

2) Для ввода новой книги



The screenshot shows a PostgreSQL query editor interface. The top toolbar includes icons for file operations, search, and execution. The main area displays a SQL query that creates a procedure named `new_book` to insert new books into the `bookstore.book` table. The procedure takes ISBN, title, release year, pages, illustrations, category ID, and author ID as parameters. Below the query editor, the 'Data Output' tab is active, showing a table with 8 rows of book data. The table has columns for ISBN, title, release_year, pages, illustrations, category_id, and author_id.

	ISBN [PK] bigint	title character varying (50)	release_year date	pages integer	illustrations boolean	category_id integer	author_id integer
1	9785845913494	Компиляторы	1986-12-03	1184	true	5321	977432
2	134610997	Artificial Intelligence	2020-12-10	1136	false	7631	98476
3	262017431	The Machine Question	2012-07-12	270	true	7821	893465
4	321197844	Введение в системы баз данных	2003-12-31	1328	true	1718	122133
5	5845907888	Введение в системы баз данных	2022-05-24	1328	true	1718	122133
6	9785819907856	Базы данных. Практическое применение СУБД	2022-04-11	368	false	1718	165781
7	2346814907856	SQL. Проектирование СУБД	2022-03-03	506	true	1718	538987
8	7547604310	основы postgres sql	2022-10-11	940	true	1718	345712

```

create procedure new_book("ISBN" bigint, title varchar, release_year date, pages int,
illustrations boolean, category_id integer, author_id integer)
language sql
as $$
insert into bookstore.book values ("ISBN", title, release_year, pages, illustrations,
category_id, author_id);
$$;

```

3) Для ввода нового заказа.

bookstore/postgres@PostgreSQL 11

Query Editor Query History

```

1 create procedure new_order(order_id integer, address varchar, "date" date, books_number integer,
2                               customer_id integer, delivery_date date, order_state varchar, total_price integer)
3 language sql
4 as $$
5 insert into bookstore.order values (order_id, address, "date", books_number, customer_id,
6                                     delivery_date, order_state, total_price);
7 $$;
8 call new_order(3455678, 'Гражданский проспект, д.108', '2022-06-01', 26, 220502, '2022-07-12', 'ожидает обработки', 2080);
9 select * from bookstore.order;
10

```

Data Output Explain Messages Notifications

	order_id [PK] integer	address character varying (30)	date date	books_number integer	customer_id integer	delivery_date date	order_state character varying (30)	total_price integer
1	1873673	Итмо, коворкинг на Ломоносова	2022-03-16	25	111581	2022-03-26	в процессе сборки	17075
2	1144567	Колпино	2022-02-14	60	142581	2022-04-18	отправлен в пункт выдачи	42000
3	1873567	Лесная	2021-11-28	2	220502	2022-03-21	собран	2000
4	6683567	Коммendantский проспект д.32	2022-05-21	30	220502	2022-06-24	собран	30000
5	3455678	Гражданский проспект, д.108	2022-06-01	26	220502	2022-07-12	ожидает обработки	2080

```

create procedure new_order(order_id integer, address varchar, "date" date, books_number
integer, customer_id integer, delivery_date date, order_state varchar, total_price integer)
language sql
as $$
insert into bookstore.order values (order_id, address, "date", books_number, customer_id,
delivery_date, order_state, total_price);
$$;

```

Создать триггерную функцию

```

1 CREATE OR REPLACE FUNCTION add_to_log() RETURNS
2 TRIGGER AS $$
3 DECLARE
4 mstr varchar(30);
5 astr varchar(100);
6 retstr varchar(254);
7 BEGIN
8 IF TG_OP = 'INSERT' THEN
9 astr = NEW;
10 mstr := 'Add new';
11 retstr := mstr || astr;
12 INSERT INTO
13 bookstore.logs(book_info, log_time) values
14 (retstr, NOW());
15 RETURN NEW;
16 ELSIF TG_OP = 'UPDATE' THEN
17 astr = NEW;
18 mstr := 'Update';
19 retstr := mstr || astr;
20 INSERT INTO
21 bookstore.logs(book_info, log_time) values
22 (retstr, NOW());
23 RETURN NEW;
24 ELSIF TG_OP = 'DELETE' THEN
25 astr = OLD;
26 mstr := 'Remove';
27 retstr := mstr || astr;
28 INSERT INTO
29 bookstore.logs(book_info, log_time) values
30 (retstr, NOW());
31 RETURN OLD;
32 END IF;
33 END;
34 $$ LANGUAGE plpgsql;
35

```

```

CREATE OR REPLACE FUNCTION add_to_log() RETURNS
TRIGGER AS $$
DECLARE
mstr varchar(30);
astr varchar(100);
retstr varchar(254);
BEGIN
IF TG_OP = 'INSERT' THEN
astr = NEW;
mstr := 'Add new';

```

```

retstr := mstr || astr;
INSERT INTO
bookstore.logs(book_info, log_time) values
(retstr,NOW());
RETURN NEW;
ELSIF TG_OP = 'UPDATE' THEN
astr = NEW;
mstr := 'Update';
retstr := mstr || astr;
INSERT INTO
bookstore.logs(book_info, log_time) values
(retstr,NOW());
RETURN NEW;
ELSIF TG_OP = 'DELETE' THEN
astr = OLD;
mstr := 'Remove';
retstr := mstr || astr;
INSERT INTO
bookstore.logs(book_info, log_time) values
(retstr,NOW());
RETURN OLD;
END IF;
END;
$$ LANGUAGE plpgsql;

```

```

1  create trigger b_books after insert or update or delete
2  on bookstore.book
3  for each row execute procedure add_to_log();
4

```

Data Output Explain **Messages** Notifications

CREATE TRIGGER

Query returned successfully in 82 msec.

Query Editor Query History

```

1  INSERT INTO bookstore.book ("ISBN", title, release_year, pages, illustrations, category_id, author_id)
2  VALUES (45566739476, 'SQL и не только', '2013-07-09', 458, true, 1718, 988761);
3

```

Data Output Explain **Messages** Notifications



INSERT 0 1

Query returned successfully in 79 msec.

Query Editor Query History

```
1  select * from bookstore.logs
```

Data Output Explain Messages Notifications

	book_info text	 log_time timestamp with time zone 
1	Add new(45566739476,"SQL и не только",2013-07-09,458,t,1718,988761)	2022-06-16 14:47:19.259209+03
2	Remove(45566739476,"SQL и не только",2013-07-09,458,t,1718,988761)	2022-06-16 15:52:46.790241+03