

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

Отчет

по лабораторной работе №5
«Процедуры, функции, триггеры в PostgreSQL»

по дисциплине «Проектирование и реализация баз данных»

Автор: Пеньков Г.Д.

Факультет: ИКТ

Группа: К3241

Преподаватель: Говорова М. М.



Санкт-Петербург 2023

Цель работы

Овладеть практическими навыками создания и использования процедур, функций и триггеров в базе данных PostgreSQL.

Вариант 3. БД «Библиотека»

Практическое задание:

Вариант 2 (max - 8 баллов)

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).
2. Модифицировать триггер (триггерную функцию) на проверку корректности входа и выхода сотрудника (см. Практическое задание 1 Лабораторного практикума (Приложение)) с максимальным учетом «узких» мест некорректных данных по входу и выходу).
3. Создать авторский триггер по варианту индивидуального задания.

Выполнение

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).
 - 1.1. Для проверки наличия экземпляров заданной книги в библиотеке (процедура должна возвращать количество экземпляров книги).

```
library=# \sf lab3.check_book_availability
CREATE OR REPLACE FUNCTION lab3.check_book_availability(_title character varying, _author character varying)
RETURNS integer
LANGUAGE plpgsql
AS $function$
declare
    num_of_available_copies int;
    _book_id int;
begin
    SELECT book_id into _book_id from lab3.books b
    join lab3.authorships aps on b.book_id = aps.book
    join lab3.authors au on aps.author = au.author_id
    where b.title = _title AND au.name = _author;
    SELECT COUNT(*) INTO NUM_OF_AVAILABLE_COPIES
    FROM LAB3.PUBLICATION_COPIES AS PC
    JOIN
        (SELECT PC.INVENTORY_NUMBER
         FROM LAB3.PUBLICATION_COPIES AS PC
         WHERE PC.WRITE_OFF_ACT IS NULL
         EXCEPT SELECT BOOK
          FROM LAB3.BOOK_LENDING AS BL
          WHERE REAL_RETURN_DATE IS NULL) AS AVAILABLE_COPIES
    ON AVAILABLE_COPIES.INVENTORY_NUMBER = PC.INVENTORY_NUMBER
    JOIN LAB3.PUBLICATIONS AS PUB ON PC.ISBN = PUB.ISBN
    JOIN LAB3.BOOKS AS B ON B.BOOK_ID = PUB.BOOK
    WHERE B.BOOK_ID = _book_id;
    return NUM_OF_AVAILABLE_COPIES;
end;
$function$

library=# select * from lab3.check_book_availability('Илиада', 'Гомер');
check_book_availability
-----
                2
(1 строка)
```

1.2. Для ввода в базу данных новой книги.

```
1 CREATE OR REPLACE PROCEDURE lab3.add_book(  
2     title VARCHAR(100),  
3     original_lang_code varchar(2),  
4     author varchar(50),  
5     year numeric(4) default null,  
6     bbk varchar(50) default null,  
7     udk varchar(50) default null  
8 )  
9 language plpgsql  
10 as $$  
11 declare  
12     _author_id int;  
13     _book_id int;  
14 begin  
15     if author not in (select name from lab3.authors) then  
16         insert into lab3.authors (name) values (author) returning author_id into _author_id;  
17     else  
18         select author_id into _author_id from lab3.authors where name = author;  
19     end if;  
20  
21     if original_lang_code not in (select lang_code from lab3.book_original_languages) then  
22         insert into lab3.book_original_languages (lang_code) values (original_lang_code);  
23     end if;  
24  
25     insert into lab3.books (title, year, bbk, udc, original_language)  
26     values (title, year, bbk, udk, original_lang_code) returning book_id into _book_id;  
27  
28     insert into lab3.authorships (author, book) values (_author_id, _book_id);  
29 end;  
30 $$;  
31
```

```
1 call lab3.add_book(  
2     'Чайка по имени Джонатан Ливингстон',  
3     'en',  
4     'Ричард Бах',  
5     1970  
6 );
```

```
1 select * from lab3.books order by book_id desc;
```

Data Output Messages Notifications

	book_id [PK] integer	title character varying (100)	year numeric (4)	bbk character varying (50)	udc character varying (50)	original_language character
1	19	Чайка по имени Джонатан Ливингстон	1970	[null]	[null]	en
2	18	Параллельное программирование на современном языке C++	2021	32.973.202	004.4	en
3	17	C# 20 в деталях	2020	32.372	004.42	en

```
1 select * from lab3.authors order by author_id desc;
```

Data Output Messages Notifications

	author_id [PK] integer	name character varying (50)
1	21	Ричард Бах
2	19	Райнер Гримм

```
1 select * from lab3.authorships order by authorship_id desc;
```

	authorship_id [PK] integer	author integer	book integer	order_num integer
1	28	21	19	[null]
2	27	16	14	[null]
3	26	16	13	[null]

1.3. Для ввода нового читателя (необходимо проверить наличие читателя в картотеке, чтобы не назначить ему номер вторично).

```
1 create or replace procedure lab3.add_reader_card(
2     _name varchar(50),
3     _passport_info varchar(200),
4     _address varchar(100),
5     _phone_num varchar(20) default null,
6     _email varchar(50) default null,
7     _education varchar(50) default null
8 )
9 language plpgsql
10 as $$
11 begin
12     if _passport_info in (select passport_info from lab3.reader_cards) then
13         raise exception 'Данный читатель уже существует';
14     end if;
15     insert into lab3.reader_cards (passport_info, reader_name, phone_number, address, email, education)
16     values (_passport_info, _name, _phone_num, _address, _email, _education);
17 end; $$;
```

```
1 call lab3.add_reader_card('Куприянов Александр Александрович',
2                             '60 13 099279, выдан ГУ МВД России по г. Саратову, кп 088-049',
3                             'Санкт-Петербург, улица Ленина, дом 10')
```

Data Output	Messages	Notifications
ERROR: Данный читатель уже существует CONTEXT: функция PL/pgSQL lab3.add_reader_card(character varying,character varying,character varying,character varying,character varying,character varying) at lab3.add_reader_card line 15, column 15 RAISE		
ОШИБКА: Данный читатель уже существует SQL state: P0001		

```
1 call lab3.add_reader_card('Горохов Иван Дмитриевич',
2                             '61 14 123279, выдан ГУ МВД России по г. Воронеж, кп 089-087',
3                             'Санкт-Петербург, улица Ленина, дом 12')
4 select * from lab3.reader_cards order by card_id desc;
```

	card_id [PK] integer	passport_info character varying (200)	reader_name character varying (50)
1	30	61 14 123279, выдан ГУ МВД России по г. Воронеж, кп 089-087	Горохов Иван Дмитриевич
2	29	63 24 620360, выдан ГУ МВД России по г. Санкт-Петербургу, кп 364-8...	Скворцов Евгений Александрович
3	28	85 10 671342, выдан ГУ МВД России по г. Уфе, кп 403-504	Гусева Ольга Валдимовна

2. Модификация триггера из практического задания

Для создания триггерной функции мы создали вспомогательную функцию, которая возвращает последнее время выхода (или входа) для заданного работника:

```
1 create or replace function get_last_punch(_employee_id int)
2 returns table (EMPLOYEE_ID int, IS_OUT_PUNCH bool, last_punch_time timestamp)
3 language plpgsql
4 as $$
5 begin
6     return query
7         SELECT T.EMPLOYEE_ID, T.IS_OUT_PUNCH, T.PUNCH_TIME
8         FROM (SELECT TP.EMPLOYEE_ID, MAX(TP.PUNCH_TIME) AS LATEST_TIME
9               FROM TIME_PUNCH TP
10              GROUP BY TP.EMPLOYEE_ID
11              HAVING TP.EMPLOYEE_ID = _employee_id
12             ) LATEST_TIMES
13         JOIN TIME_PUNCH T
14         ON T.EMPLOYEE_ID = LATEST_TIMES.EMPLOYEE_ID AND T.PUNCH_TIME = LATEST_TIMES.LATEST_TIME;
15 end;
16 $$;
17 select * from get_last_punch(1)
```

Data Output Messages Notifications

	employee_id integer	is_out_punch boolean	last_punch_time timestamp without time zone
1	1	true	2023-11-24 14:25:59.656886

Триггерная функция и создание триггера:

```
1 create or replace function fn_check_time_punch() returns trigger as $psql$
2 declare
3     _last_punch_time timestamp;
4     _last_is_out_punch bool;
5 begin
6     select is_out_punch into _last_is_out_punch from get_last_punch(new.employee_id);
7     select last_punch_time into _last_punch_time from get_last_punch(new.employee_id);
8
9     -- нельзя выйти/войти 2 раза подряд
10    if new.is_out_punch = _last_is_out_punch then return null;
11    end if;
12    -- выход не может быть первой записью (сначала нужно войти)
13    if new.is_out_punch = true and _last_is_out_punch is null then return null;
14    end if;
15    -- новый выход/вход должен быть позже последнего
16    if new.punch_time < _last_punch_time then return null;
17    end if;
18    return new;
19 end;
20 $psql$ language plpgsql;
21
22 create trigger check_time_punch before insert on time_punch
23 for each row execute procedure fn_check_time_punch();
```

Примеры выполнения:

1

INSERT INTO TIME_PUNCH (EMPLOYEE_ID, IS_OUT_PUNCH)

2

VALUES (3, true);

3

4

select * from TIME_PUNCH order by id desc;

Data Output

Messages

Notifications

	id [PK] integer	employee_id integer	is_out_punch boolean	punch_time timestamp without time zone
1	26	2	true	2024-11-24 17:42:48
2	25	2	false	2023-11-24 14:42:54.730079
3	24	2	true	2023-11-24 14:28:12.761426

1

INSERT INTO TIME_PUNCH (EMPLOYEE_ID, IS_OUT_PUNCH)

2

VALUES (3, false);

3

4

select * from TIME_PUNCH order by id desc;

Data Output

Messages

Notifications

	id [PK] integer	employee_id integer	is_out_punch boolean	punch_time timestamp without time zone
1	30	3	false	2023-12-27 11:28:33.423699
2	26	2	true	2024-11-24 17:42:48
3	25	2	false	2023-11-24 14:42:54.730079

1

INSERT INTO TIME_PUNCH (EMPLOYEE_ID, IS_OUT_PUNCH, punch_time)

2

VALUES (3, false, '2022-01-01');

3

4

select * from TIME_PUNCH order by id desc;

Data Output

Messages

Notifications

id

[PK] integer

employee_id

integer

is_out_punch

boolean

punch_time

timestamp without time zone

1

36

3

true

2023-12-27 11:31:16.284443

2

30

3

false

2023-12-27 11:28:33.423699

3. Создание триггера по варианту индивидуального задания

```
1 CREATE OR REPLACE FUNCTION FN_CHECK_BOOK_LENDING() RETURNS TRIGGER LANGUAGE PLPGSQL AS $$
2 begin
3     -- нельзя выдать книгу, которая уже выдана
4     if new.book in
5         (select book from lab3.book_lending where real_return_date is null) then
6         return null;
7     end if;
8
9     -- нельзя выдать книгу, которая списана с учета
10    if new.book in
11        (select inventory_number from lab3.publication_copies where write_off_act is not null) then
12        return null;
13    end if;
14
15    -- нельзя выдать книгу, если у человека более чем 3 задолженных книги
16    if (select num_of_fined_books from lab3.debtors where card_id = new.reader_card) > 3 then
17        return null;
18    end if;
19    return new;
20 end; $$;
21
22 create trigger check_book_lending before insert on lab3.book_lending
23 for each row execute procedure fn_check_book_lending();
```

Примеры выполнения:

```
1 insert into lab3.book_lending (reader_card, book, employee)
2 values (24, 89, 4)
3 -- книга с инвентарным номером 89 уже взята
```

Data Output	Messages	Notifications
INSERT 0 0		

```
1 insert into lab3.book_lending (reader_card, book, employee)
2 values (24, 73, 4)
3 -- книга с инвентарным номером 73 снята с учета
```

Data Output	Messages	Notifications
INSERT 0 0		

1 insert into lab3.book_lending (reader_card, book, employee)

2 values (24, 172, 4)

3 |

4 select * from lab3.book_lending order by lending_id desc

Data Output

Messages

Notifications

	<div>lending_id</div> <div>[PK] integer</div>	<div>reader_card</div> <div>integer</div>	<div>book</div> <div>integer</div>	<div>lend_date</div> <div>date</div>	<div>planned_return_date</div> <div>date</div>	<div>real_return_date</div> <div>date</div>	<div>return_info</div> <div>character varying (100)</div>	<div>employee</div> <div>integer</div>	<div>fine</div> <div>integer</div>
1	70	24	172	2023-12-27	2024-01-27	[null]	[null]	4	0
2	67	24	95	2023-12-19	2023-12-30	[null]	[null]	4	0

1	insert into lab3.book_lending (reader_card, book, employee)
2	values (19, 160, 4)
3	

Data Output	Messages	Notifications
INSERT 0 0		

4

select * from lab3.debtors where card_id = 19

Data Output

Messages

Notifications

≡+

▼

▼

	card_id integer	reader_name character varying (50)	total_fine bigint	num_of_fined_books bigint
1	19	Калинина Тамара Ивановна	2255	6

Вывод

В данной лабораторной работе было написано несколько процедур, функций и триггеров. Тем самым, были освоены основы работы с процедурным языком PL/pgSQL.