

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: Чебан И.В.

Факультет: ИКТ

Группа: К3241

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

Оглавление

Цель работы:.....	3
Практическое задание.....	3
Выполнение.....	3
Вывод.....	8

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 (max - 6 баллов)

1. Создать процедуры/функции согласно индивидуальному заданию (часть 4).
2. 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

Создайте хранимые процедуры:

1. Функция для снижения цены на заданный процент для товаров, у которых срок пребывания на складе превысил заданный норматив.

```
CREATE OR REPLACE FUNCTION reduce_price(days_limit INT,  
percent_amount REAL)  
RETURNS VOID AS $$  
BEGIN  
UPDATE shipment_content  
SET price = price * (1 - percent_amount / 100)  
FROM shipment  
WHERE shipment.shipment_id = shipment_content.shipment_id AND  
current_date - shipment.delivery_date > days_limit;  
END;  
$$ LANGUAGE plpgsql;
```

```

postgres=# CREATE OR REPLACE FUNCTION reduce_price(days_limit INT,
postgres(# percent_amount REAL)percent_amount REAL)
postgres=# RETURNS VOID AS $$
postgres$# BEGIN
postgres$# UPDATE shipment_content
SET price = price * (1 - percent_amount / 100)
FROM shipmentUPDATE shipment_content
postgres$# SET price = price * (1 - percent_amount / 100)
FROM shipmentSET price = price * (1 - percent_amount / 100)
postgres$# FROM shipmentFROM shipment
postgres$# WHERE shipment.shipment_id = shipment_content.shipment_id AND
postgres$# current_date - shipment.delivery_date > days_limit;
END;current_date - shipment.delivery_date > days_limit;
postgres$# END;END;
postgres$# $$ LANGUAGE plpgsql;
CREATE FUNCTION

```

2) Функция для расчета стоимости всех партий товаров, проданных за прошедшие сутки.

```

CREATE OR REPLACE FUNCTION total_sales_last_day()

RETURNS DECIMAL AS $$

DECLARE total DECIMAL;

BEGIN

SELECT SUM(custom_content.price *

custom_content.amount_of_goods) INTO total

FROM custom_content

JOIN custom ON custom.purchase_id = custom_content.purchase_id

WHERE custom.invoice_creation_date = current_date - interval '1 day';

RETURN total;

END;


$$ LANGUAGE plpgsql;

```

```

postgres=# CREATE OR REPLACE FUNCTION total_sales_last_day()
postgres=# RETURNS DECIMAL AS $$
postgres$# DECLARE total DECIMAL;
postgres$# BEGIN
postgres$# SELECT SUM(custom_content.price *
postgres$# custom_content.product_amount) INTO total
postgres$# FROM custom_content
postgres$# JOIN custom ON custom.custom_id = custom_content.custom_id
postgres$# WHERE custom.invoice_creation_date = current_date - interval '1 day';
postgres$# RETURN total;
postgres$# END;
postgres$# $$ LANGUAGE plpgsql;
CREATE FUNCTION

```

	total_sales_last_day 
	numeric
1	500

Создайте необходимые триггеры:

1) Создадим таблицу для записи логирования:

```

CREATE TABLE log_table (
log_id BIGINT GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
table_name VARCHAR(100),
operation VARCHAR(10),
old_data TEXT,
new_data TEXT,
log_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

```

2) Создадим функции и триггеры

```

CREATE OR REPLACE FUNCTION product_insert_trigger()
RETURNS TRIGGER AS $$
BEGIN
INSERT INTO log_table (table_name, operation, new_data)
VALUES ('product', 'INSERT', row_to_json(NEW)::text);
RETURN NEW;
END;
$$ LANGUAGE plpgsql;

```

```

CREATE TRIGGER product_insert_log
AFTER INSERT ON product
FOR EACH ROW EXECUTE FUNCTION product_insert_trigger();

```

```

CREATE OR REPLACE FUNCTION product_update_trigger()
RETURNS TRIGGER AS $$
BEGIN
INSERT INTO log_table (table_name, operation, old_data, new_data)
VALUES ('product', 'UPDATE', row_to_json(OLD)::text,
row_to_json(NEW)::text);
RETURN NEW;

```

```
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER product_update_log
AFTER UPDATE ON product
FOR EACH ROW EXECUTE FUNCTION product_update_trigger();
```

```
CREATE OR REPLACE FUNCTION product_delete_trigger()
RETURNS TRIGGER AS $$
BEGIN
INSERT INTO log_table (table_name, operation, old_data)
VALUES ('product', 'DELETE', row_to_json(OLD)::text);
RETURN OLD;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER product_delete_log
AFTER DELETE ON product
FOR EACH ROW EXECUTE FUNCTION product_delete_trigger();
```

```

-- CUSTOMER FUNCTIONS
CREATE TRIGGER custom_insert_log
AFTER INSERT ON custom
FOR EACH ROW EXECUTE FUNCTION custom_insert_trigger();
CREATE OR REPLACE FUNCTION custom_update_trigger()
RETURNS TRIGGER AS $$
BEGIN
INSERT INTO log_table (table_name, operation, old_data, new_data)
VALUES ('custom', 'UPDATE', row_to_json(OLD)::text,
row_to_json(NEW)::text);
RETURN NEW;
END;
$$ LANGUAGE plpgsql;
CREATE TRIGGER custom_update_log
AFTER UPDATE ON custom
FOR EACH ROW EXECUTE FUNCTION custom_update_trigger();
CREATE OR REPLACE FUNCTION custom_delete_trigger()
RETURNS TRIGGER AS $$
BEGIN
INSERT INTO log_table (table_name, operation, old_data)
VALUES ('custom', 'DELETE', row_to_json(OLD)::text);
RETURN OLD;
END;
$$ LANGUAGE plpgsql;
CREATE TRIGGER custom_delete_log
AFTER DELETE ON custom
FOR EACH ROW EXECUTE FUNCTION custFOR EACH ROW EXECUTEVALUES ('custom', 'INSERT', row_to_json(NEW)::text);
postgres## RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

	log_id [PK] bigint	table_name character varying (100)	operation character varying (10)	old_data text
1	1	custom	INSERT	[null]
2	2	custom	UPDATE	{'custom_id': 'custom10', 'client_id': 'client2', 'manager_id': 'manager1', 'invoice_creation_date': '2023-12-1
3	3	custom	DELETE	{'custom_id': 'custom10', 'client_id': 'client1', 'manager_id': 'manager1', 'invoice_creation_date': '2023-12-1

Вывод

В ходе лабораторной работы была освоена работа с процедурами и триггерами.