# 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 «Базы данных»

Автор: Демша Евгения

Факультет: ИКТ

Группа: К32422

Преподаватель: Говорова М.М.



Санкт-Петербург, 2023

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

#### Практическое задание:

- 1 .Создать процедуры/функции согласно индивидуальному заданию
- 2. Создать триггер для логирования событий вставки, удаления, редактирования данных в базе данных PostgreSQL
- 1. Создать хранимые процедуры:
  - О текущей сумме вклада и сумме начисленного за месяц процента для заданного клиента;

Так как база данных заполнялась месяц назад, последние начисления по вкладу указаны в апреле 2023 года. В запросе есть допущение, за текущий (последний) месяц считается апрель (4). В настоящем запросе месяц должен быть равен текущему.

• Найти клиента банка, имеющего максимальное количество кредитов на текущий день;

```
bank=# CREATE OR REPLACE FUNCTION max_credit_count() RETURNS TABLE(client_name fio) AS $$
bank$# BEGIN
bank$#
           RETURN QUERY
        (SELECT full_name
bank$#
bank$# FROM client
bank$# JOIN
         (SELECT client_passport
bank$#
bank$#
bank$#
                 (SELECT client passport,
bank$#
                                  COUNT(credit_contract.contract_number)
bank$#
                  FROM credit_contract
                  GROUP BY client_passport) AS cr
bank$#
        WHERE COUNT =
bank$#
bank$#
                   (SELECT MAX (COUNT)
bank$#
bank$#
                          (SELECT client_passport,
bank$#
                                           COUNT(credit_contract.contract_number)
                            FROM credit_contract
bank$#
                            WHERE status = 'active'
bank$#
bank$#
                           GROUP BY client passport) AS cr)) AS mxcr ON mxcr.client passport=client.passport);
bank$# END;
bank$# $$ LANGUAGE PLPGSQL;
CREATE FUNCTION
bank=# SELECT max credit count();
   max_credit_count
ЦЁёсыютр Lы№сшэр +уюЁютэр
(1 ё€Ёюър)
```

Функция вернула полное имя человека с наибольшим количеством кредитов...

• найти клиентов банка, не имеющих задолженности по кредитам.

```
bank=# CREATE OR REPLACE FUNCTION no debt() RETURNS TABLE(client name fio) AS $$
  bank$# BEGIN
bank$# RETURN QUERY
bank$# SELECT DISTINCT(full_name)
bank$# FROM payment_schedule
bank$# JOIN credit_contract_USING(contract_number)
 bank$# JOIN client ON credit contract.client passport = client.passport
 bank$# WHERE real_date IS NOT NULL
 bank$#
                                         AND status = 'active';
 bank$# END;
 bank$# $$ LANGUAGE PLPGSQL;
  CREATE FUNCTION
  bank=# SELECT no_debt();
                                                                     no_debt
       Lь эўхт Һыхс ⊥рышыхтшў
       ∦хьхэъют тшьеЁ ±ртыютшў
    TSEATARE TARABLE TARABLE TO THE TARABLE T
       1<sub>€ъ°</sub> 1<sub>рьшы№</sub> Ё№хтшў
       \pmхёхэсеш<br/>ішёэахёх\pm qeш<br/>ішёэахёх
       детоухиф фишь фетер
фетоухиф фетер
фетоуска
```

### 2. Создать триггеры

• Триггер, который не позволит вставить в таблицу с начислениями по вкладу начисление, дата которого раньше последнего.

```
bank=# create or replace function fn check accrual date() returns trigger as $psql$
bank$#
         begin
bank$#
           if new.date_accrual < (
               select date_accrual
from interest accrual acc
bank$#
bank$#
                where acc.contract_number = new.contract_number
order by acc.date_accrual desc limit 1
bank$#
bank$#
bank$#
             ) then
bank$#
                   return null;
              end if;
bank$#
bank$#
               return new;
bank$#
          end;
bank$# $psql$ language plpgsql;
CREATE FUNCTION
bank=# create trigger check accrual date before insert on interest accrual
bank-# for each row execute procedure fn check accrual date();
CREATE TRIGGER
bank=# INSERT INTO interest_accrual(date_accrual, contract_number, month_accrual, current_sum)
bank-# VALUES('2023-02-11',45044273,30000,3053530);
INSERT 0 0
bank=# INSERT INTO interest accrual(date accrual, contract number, month accrual, current sum)
bank-# VALUES('2023-05-11',\overline{4}5044273,3000\overline{0},3053530);
INSERT 0 1
bank=#
```

Так как последнее начисление для вклада с этим номером контракта произошло в апреле, мы не смогли вставить новую запись с датой в феврале, но смогли с датой в мае.

• После добавления новой валюты в таблицу с валютами, триггер создаст новый вклад в этой валюте

```
bank=# CREATE OR REPLACE FUNCTION currency_insert_trigger_fnc()
bank-# RETURNS trigger AS $$
bank$# BEGIN
bank$# INSERT INTO deposit(deposit_name, currency_code)
bank$# VALUES(CONCAT('Вклад в валюте ', NEW.currency_name, '"') ,NEW.currency_code);
bank$# RETURN NEW;
bank$# END;
bank$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
bank=# CREATE TRIGGER currency_insert_trigger
bank-# AFTER INSERT
bank-# ON currency
bank-# FOR EACH ROW
bank-# EXECUTE PROCEDURE currency_insert_trigger_fnc();
CREATE TRIGGER
bank=# INSERT INTO currency(currency_code, currency_name, country)
bank-# VALUES ('MNT','Тугрик','Монголия')
bank-# ;
INSERT 0 1
bank=# SELECT * FROM deposit;
    deposit_name | currency_code | min_sum | min_term |
                      description
_____
√еў°шщ тъырф
              | RUB     | 100000 | 1 mon   | ∯Сюс√ яюыеўшЄ№ ыръёшыры№эс∎ ёЄртъє 9,5% Ёр
фхэ№уш, ъюЄюЁ√х т√ эх їЁрэшыш эр фЁєушї тр°шї тъырфрї т <sup>⊥</sup>рэъх т яЁхф√фє шх 3 ърыхэфрЁэ√ї ьхё Ўр ш юЄъЁющ€
(5 ëEËюъ)
```

#### Вывод:

Хранимые процедуры, функции и триггеры помогают упростить и автоматизировать работу с базой данных.