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

Отчет

по лабораторной работе «Процедуры, функции, триггеры в PostgreSQL»

по дисциплине «**Базы данных**»

Автор: Косенко.Ф

Факультет: ИКТ

Преподаватель: Говорова М.М.

Санкт-Петербург 2023

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

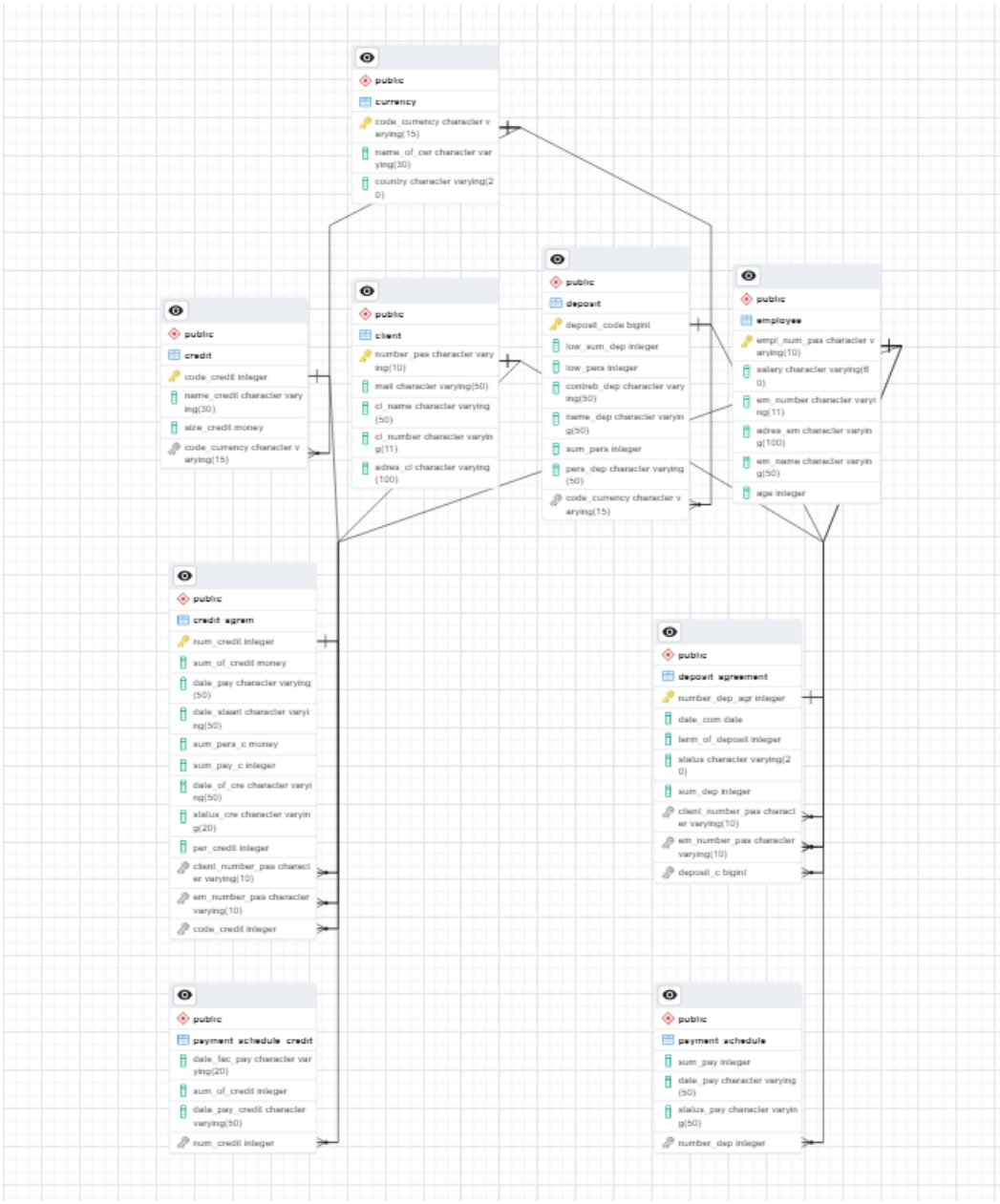
Оборудование: компьютерный класс.

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

Практическое задание:
Вариант 1

- скрипты кода разработанных объектов (процедур/функций и триггера на логирование действий) и подтверждающие скриншоты работы и результатов в psql согласно индивидуальному заданию (часть 4 и 5).

Ход работы:



Создать хранимые процедуры:
о текущей сумме вклада и сумме начисленного за месяц процента для заданного клиента;

```
CREATE OR REPLACE FUNCTION get_deposit_info(client_id INTEGER)
RETURNS TABLE (deposit_amount NUMERIC) AS
$$
DECLARE
  current_date DATE := now()::DATE;
BEGIN
  SELECT
    SUM(sum_dep) INTO deposit_amount
  FROM
    deposit_agreement
  WHERE
    client_number_pas = CAST(get_deposit_info.client_id AS character varying);

  RETURN QUERY SELECT deposit_amount;
END;
$$
LANGUAGE plpgsql;
```

Вывод:

```
1  SELECT * FROM get_deposit_info(953145)
```

Data Output | Сообщения | Notifications

	deposit_amount numeric
1	2000

Найти клиента банка, имеющего максимальное количество кредитовна текущий день;

```
CREATE OR REPLACE FUNCTION get_max_credit_client()
RETURNS TABLE (client_name VARCHAR(100), credit_count BIGINT)
AS $$
BEGIN
    RETURN QUERY
        SELECT client.cl_name, COUNT(deposit_agreement.number_dep_agr) AS credit_count
        FROM deposit_agreement
        INNER JOIN client ON deposit_agreement.client_number_pas = client.cl_name
        GROUP BY client.cl_name
        ORDER BY credit_count DESC
        LIMIT 1;
END;
$$ LANGUAGE plpgsql;
```

Вывод:

```
1  SELECT * from get_max_credit_client()
```

Data Output | Сообщения | Notifications

	client_name character varying	credit_count bigint
1	Коля	3

Найти клиентов банка, не имеющих задолженности по кредитам

```
CREATE OR REPLACE FUNCTION get_clients_without_debt()
RETURNS TABLE (client_name VARCHAR(100))
AS $$
BEGIN
    RETURN QUERY
        SELECT DISTINCT client.cl_name
        FROM client
        LEFT JOIN credit_agreement ON client.cl_name = credit_agreement.client_number_pas
        WHERE credit_agreement.number_credit_agr IS NULL OR credit_agreement.debt_amount = 0;
END;
$$ LANGUAGE plpgsql;
```

Вывод

```
1  SELECT * FROM get_clients_without_debt()
```

Data Output | Сообщения | Notifications

	client_name character varying
1	Александр
2	Кирилл
3	Коля
4	Ольга
5	Павел
6	Семен

Триггер для логирования событий вставки, удаления и обновления данных в таблице

```
CREATE OR REPLACE FUNCTION LogTriggerFunction()
RETURNS TRIGGER AS $$
BEGIN
  IF (TG_OP = 'INSERT') THEN
    INSERT INTO LogTable (TableName, ActionType)
    VALUES (TG_TABLE_NAME, 'Insert');
  ELSIF (TG_OP = 'UPDATE') THEN
    INSERT INTO LogTable (TableName, ActionType)
    VALUES (TG_TABLE_NAME, 'Update');
  ELSIF (TG_OP = 'DELETE') THEN
    INSERT INTO LogTable (TableName, ActionType)
    VALUES (TG_TABLE_NAME, 'Delete');
  END IF;

  RETURN NEW;
END;
$$ LANGUAGE plpgsql;

-- Привязка триггера к таблице
CREATE TRIGGER LogTrigger
AFTER INSERT OR UPDATE OR DELETE ON client
FOR EACH ROW
EXECUTE FUNCTION LogTriggerFunction();
```

Проверим работу:

```
1  insert into student
2  values (22 ,'Ron', '10-01-2023');
```

Data Output | Сообщения | Notifications

	logid [PK] integer	tablename character varying (100)	actiontype character varying (50)	actiondate timestamp without time zone
1	1	student	Insert	2023-05-27 15:54:24.916548

В ходе лабораторной работы я научился создавать и использовать процедуры, функции и триггеры в базе данных PostgreSQL. Также, я понял, что функции и процедуры в SQL недостаточно гибкие, как в ЯП.