

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

Отчет

по Лабораторной Работе № 4
по дисциплине «**Базы Данных**»

Автор: Акулов Даниил Даниилович

Факультет: ИКТ

Группа: К3239

Преподаватель: Говорова Марина Михайловна

ИТМО

Санкт-Петербург, 2023

Содержание работы

Цель работы:

Овладеть практическими навыками создания представлений и запросов на выборку данных к базе данных PostgreSQL, использования подзапросов при модификации данных и индексов.

Практическое задание:

1. Создать запросы и представления на выборку данных к базе данных PostgreSQL (согласно индивидуальному заданию, часть 2 и 3).
2. Составить 3 запроса на модификацию данных (INSERT, UPDATE, DELETE) с использованием подзапросов.
3. Изучить графическое представление запросов и просмотреть историю запросов.
4. Создать простой и составной индексы для двух произвольных запросов и сравнить время выполнения запросов без индексов и с индексами. Для получения плана запроса использовать команду EXPLAIN.

Вариант 19. БД «Банк»

Схема логической модели базы данных, сгенерированная в Generate ERD указана на рисунке 1.

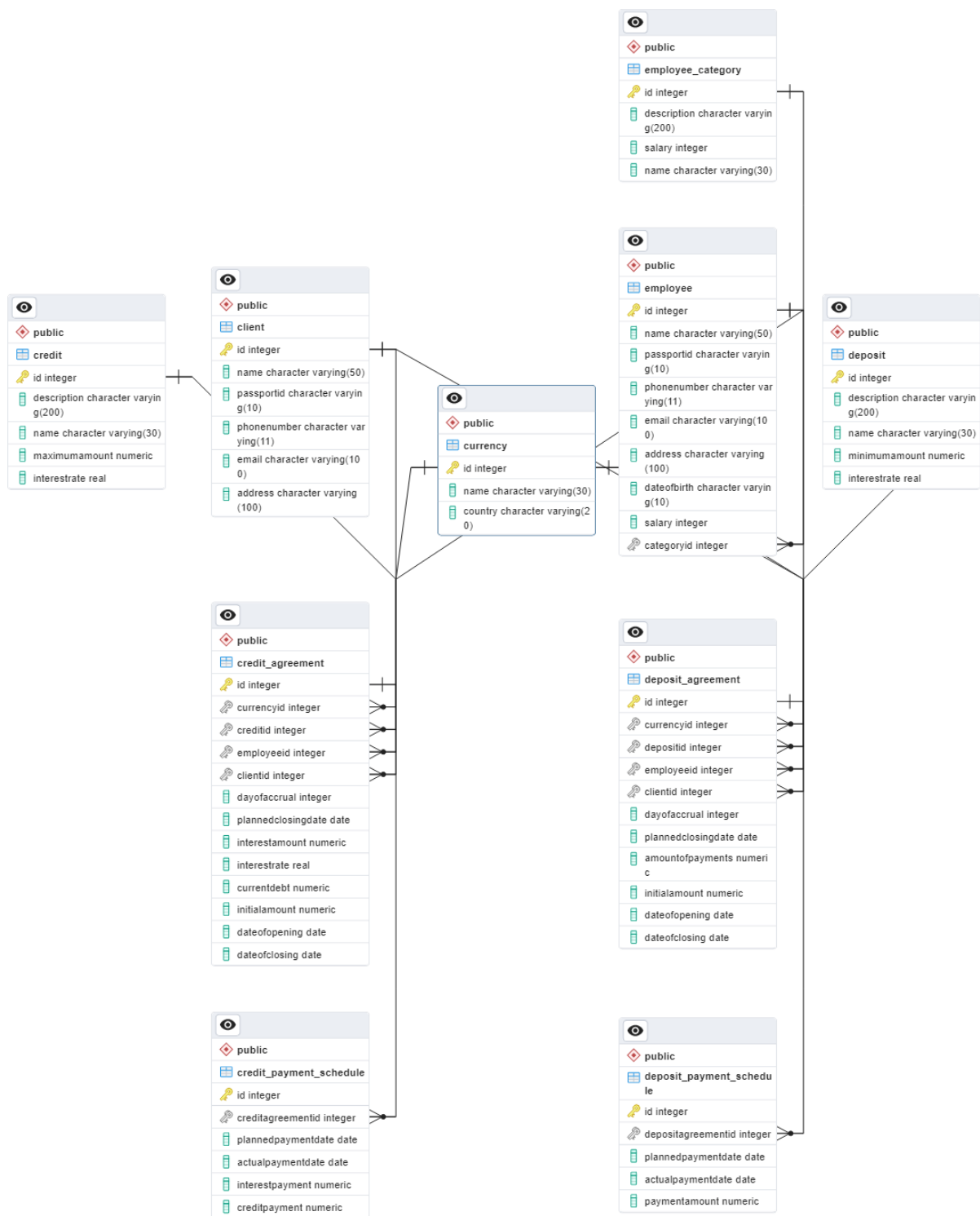


Рисунок 1 – Схема логической модели базы данных.

Выполнение работы:

1. Запросы к базе данных

1. Найти вкладчика, имеющего на текущий день несколько вкладов.

Query

Query History

```
1 select client.id, client.name
2 from client, deposit_agreement
3 where client.id = deposit_agreement.clientid and deposit_agreement.dateofclosing is null
4 group by client.id
5 having count(*) > 1;
```

Data Output

Messages


Notifications

	id [PK] integer	name character varying (50)
1	4	Леонид Арнольдович Федун

```
select deposit_agreement.clientid
from deposit_agreement
group by deposit_agreement.clientid
having count(distinct deposit_agreement.currencyid)=(select count(*) from currency);
```

```
Query  Query History
1  select deposit_agreement.clientid
2  from deposit_agreement
3  group by deposit_agreement.clientid
4  having count(distinct deposit_agreement.currencyid) =
5  (select count(*) from currency);
```

Data Output Messages Notifications

	clientid integer 
1	4

```
select client.id, client.name
```

```

from client, deposit_agreement
where client.id = deposit_agreement.clientid
and deposit_agreement.currencyid = (select id from currency where name='Фунт')
and deposit_agreement.initialamount = (select max(deposit_agreement.initialamount)
from deposit_agreement
where deposit_agreement.currencyid = (select id from currency where name='Фунт'));

```

The screenshot shows a database query editor with two tabs: 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL query. Below the query, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with two columns: 'id' (integer, primary key) and 'name' (character varying (50)).

```

1 select client.id, client.name
2 from client, deposit_agreement
3 where client.id = deposit_agreement.clientid
4 and deposit_agreement.currencyid = (select id from currency where name='Фунт')
5 and deposit_agreement.initialamount = (select max(deposit_agreement.initialamount)
6 from deposit_agreement
7 where deposit_agreement.currencyid = (select id from currency where name='Фунт'));

```

id	name
[PK] integer	character varying (50)

4. Какой из вкладов пользовался наибольшей популярностью за истекший год.

```

SELECT depositid, COUNT(*) as n
FROM deposit_agreement
WHERE dateofopening >= CURRENT_DATE - INTERVAL '1 year'
GROUP BY depositid
ORDER BY n DESC
LIMIT 1;

```

The screenshot shows a database query editor with two tabs: 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL query. Below the query, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table with two columns: 'depositid' (integer) and 'n' (bigint). The table contains one row with the values 1 and 2 respectively.

```

1 SELECT depositid, COUNT(*) as n
2 FROM deposit_agreement
3 WHERE dateofopening >= CURRENT_DATE - INTERVAL '1 year'
4 GROUP BY depositid
5 ORDER BY n DESC
6 LIMIT 1;

```

depositid	n
integer	bigint
1	2

5. Кто из сотрудников заключил максимальное число договоров по кредитам за последний месяц.

```
SELECT employeeid, COUNT(*) AS n
FROM credit_agreement
WHERE credit_agreement.dateofopening >= CURRENT_DATE - INTERVAL '1 month'
GROUP BY employeeid
ORDER BY n DESC
LIMIT 1;
```

The screenshot shows a SQL query editor with a 'Query' tab selected. The query is as follows:

```
1 SELECT employeeid, COUNT(*) AS n
2 FROM credit_agreement
3 WHERE credit_agreement.dateofopening >= CURRENT_DATE - INTERVAL '1 month'
4 GROUP BY employeeid
5 ORDER BY n DESC
6 LIMIT 1;
7
```

Below the query editor, the 'Data Output' tab is selected, showing the result of the query. The result is a table with two columns: 'employeeid' (integer) and 'n' (bigint). The first row shows employeeid 1 and n 7.

	employeeid integer	n bigint
1	7	1

6. Вывести список вкладчиков, у которых срок вклада истекает завтра и суммы начислений, которые могут быть ими востребованы.

```
SELECT cl.id, cl.name, da.initialamount + SUM(dps.paymentamount)
AS summ
FROM client cl, deposit_agreement da, deposit_payment_schedule dps
WHERE cl.id = da.clientid
AND da.id = dps.depositagreementid
AND da.plannedclosingdate = (CURRENT_DATE + interval '1 DAY')
GROUP BY cl.id, da.initialamount;
```

Query

Query History

1

SELECT cl.id, cl.name, da.initialamount + SUM(dps.paymentamount)

2

AS summ

3

FROM client cl, deposit_agreement da, deposit_payment_schedule dps

4

WHERE cl.id = da.clientid

5

AND da.id = dps.depositagreementid

6

AND da.plannedclosingdate = (CURRENT_DATE + interval '1 DAY')

7

GROUP BY cl.id, da.initialamount;

Data Output

Messages

Notifications

≡+

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

	id integer	🔒	name character varying (50)	🔒	summ integer	🔒
1	4		Леонид Арнольдович Федун		300000	

7. Вывести список сотрудников, заключивших договоры по вкладам на максимальную сумму за последний месяц.

SELECT employeeid

FROM deposit_agreement

WHERE deposit_agreement.initialamount =

(SELECT MAX(initialamount)

FROM deposit_agreement

WHERE deposit_agreement.dateofopening >=

CURRENT_DATE - INTERVAL '1 month')

Query

Query History

1

2

3

4

5

6

7

SELECT

employeeid

FROM

deposit_agreement

WHERE

deposit_agreement.initialamount =

(SELECT

MAX(initialamount)

FROM

deposit_agreement

WHERE

deposit_agreement.dateofopening >=

CURRENT_DATE - INTERVAL '1 month')

Data Output

Messages

Notifications

employeeid

integer

1

7

2. Представления

1. Содержащее сведения обо всех сотрудниках банка и заключенных ими договорах по кредитам за прошедший месяц.

```
CREATE VIEW employeeandcredits AS
```

```
SELECT e.id, e.name, e.passportid, e.phonenumber, e.email, e.address, e.dateofbirth,
e.salary, e.categoryid,
```

```
ca.id as creditagreementid, ca.currencyid, ca.creditid, ca.clientid, ca.dayofaccrual,
ca.plannedclosingdate,
```

```
ca.interestamount, ca.currentdebt, ca.initialamount, ca.dateofopening, ca.dateofclosing
```

```
FROM credit_agreement ca, employee e
```

```
WHERE ca.dateofopening >= (CURRENT_DATE - INTERVAL '1 month')
```

The screenshot shows a SQL query editor with a tab labeled 'Query'. The query text is as follows:

```
1 drop view if exists employeeandcredits;
2 CREATE VIEW employeeandcredits AS
3 SELECT e.id, e.name, e.passportid, e.phonenumber, e.email, e.address, e.dateofbirth, e.salary, e.categoryid,
4 ca.id as creditagreementid, ca.currencyid, ca.creditid, ca.clientid, ca.dayofaccrual, ca.plannedclosingdate,
5 ca.interestamount, ca.currentdebt, ca.initialamount, ca.dateofopening, ca.dateofclosing
6 FROM credit_agreement ca, employee e
7 WHERE ca.dateofopening >= (CURRENT_DATE - INTERVAL '1 month')
```

Below the query editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing the message: 'CREATE VIEW' and 'Query returned successfully in 96 msec.'

The screenshot shows a SQL query editor with a tab labeled 'Query'. The query text is:

```
1 SELECT * FROM employeeandcredits
```

Below the query editor, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, displaying a table with the following data:

	id integer	name character varying (50)	passportid character varying (10)	phonenumber character varying (11)	email character varying (100)	address character varying (100)	dateofbirth character varying (10)
1	7	Игорь Викторович Макаров	1919385795	78843693165	markovbankir@bank.ru	Воронеж, ул. Карла Маркса, 67/2	12.05.1989

2. Найти клиентов банка, имеющих задолженности по кредитам.

```
CREATE VIEW clientdebt AS
```

```
SELECT cl.*
```

```
FROM credit_agreement ca, client cl, credit_payment_schedule cps
```

```
WHERE ca.id = cps.creditagreementid
```

```
AND ca.clientid = cl.id
```


AND cps.plannedpaymentdate < CURRENT_DATE

AND cps.actualpaymentdate IS NULL

```
Query  Query History
1  drop view if exists clientdebt;
2  CREATE VIEW clientdebt AS
3  SELECT cl.*
4  FROM credit_agreement ca, client cl, credit_payment_schedule cps
5  WHERE ca.id = cps.creditagreementid
6  AND ca.clientid = cl.id
7  AND cps.plannedpaymentdate < CURRENT_DATE
8  AND cps.actualpaymentdate IS NULL

Data Output  Messages  Notifications
CREATE VIEW
Query returned successfully in 75 msec.
```

3. Запросы на модификацию данных

1. INSERT - Создать договор о кредите на 500000 руб. на клиента с номером паспорта 2809346819.

INSERT INTO credit_agreement(currencyid, creditid, clientid, employeeid, dayofaccrual, plannedclosingdate, interestamount, currentdebt, initialamount, dateofopening)

VALUES((SELECT id FROM currency WHERE name = 'rub'), 2, (SELECT id FROM client WHERE passportid = '2809346819'), 7, EXTRACT(DAY FROM CURRENT_DATE), CURRENT_DATE + INTERVAL '1 YEAR', 0, 0, 500000, CURRENT_DATE)

```
Query  Query History
1  INSERT INTO credit_agreement(currencyid, creditid, clientid, employeeid, dayofaccrual,
2  plannedclosingdate, interestamount, currentdebt, initialamount, dateofopening)
3  VALUES((SELECT id FROM currency WHERE name = 'rub'), 2,
4  (SELECT id FROM client WHERE passportid = '2809346819'), 7, EXTRACT(DAY FROM CURRENT_DATE),
5  CURRENT_DATE + INTERVAL '1 YEAR', 0, 0, 500000, CURRENT_DATE)
6

Data Output  Messages  Notifications
INSERT 0 1
Query returned successfully in 60 msec.
```

2. UPDATE - Обновить сумму начисленных процентов у всех договоров о вкладе

UPDATE deposit_agreement da

SET amountofpayments = (SELECT COALESCE(SUM(paymentamount), 0)

FROM deposit_payment_schedule dps

WHERE dps.depositagreementid = da.id AND dps.actualpaymentdate IS NOT NULL)



The screenshot shows a SQL query execution window with tabs for 'Query', 'Query History', 'Data Output', 'Messages', and 'Notifications'. The 'Query' tab is active, displaying the following SQL code:

```
1 UPDATE deposit_agreement da
2 SET amountofpayments = (SELECT COALESCE(SUM(paymentamount), 0) FROM deposit_payment_schedule dps
3 WHERE dps.depositagreementid = da.id AND dps.actualpaymentdate IS NOT NULL)
```

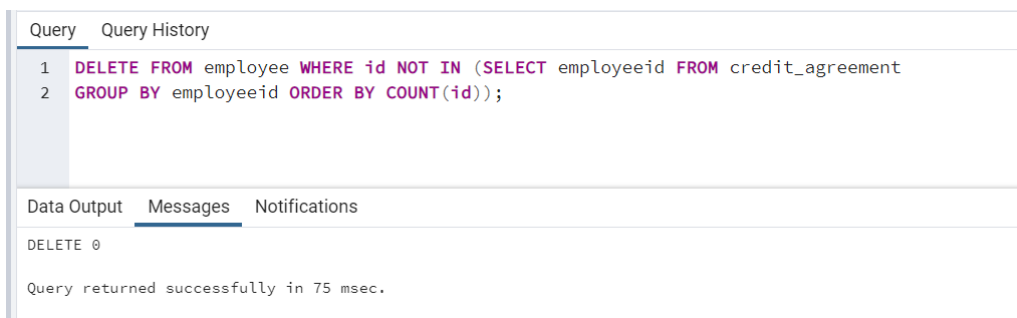
The 'Messages' tab is also active, showing the following output:

```
UPDATE 1
Query returned successfully in 91 msec.
```

3. DELETE - Удалить сотрудников, которые не заключили ни одного договора о кредите

DELETE FROM employee WHERE id NOT IN (SELECT employeeid FROM credit_agreement

GROUP BY employeeid ORDER BY COUNT(id));



The screenshot shows a SQL query execution window with tabs for 'Query', 'Query History', 'Data Output', 'Messages', and 'Notifications'. The 'Query' tab is active, displaying the following SQL code:

```
1 DELETE FROM employee WHERE id NOT IN (SELECT employeeid FROM credit_agreement
2 GROUP BY employeeid ORDER BY COUNT(id));
```

The 'Messages' tab is also active, showing the following output:

```
DELETE 0
Query returned successfully in 75 msec.
```

4. Создание индексов

Найти номера, имена, номера телефона и текущие долги клиентов по кредиту:

SELECT ca.clientid, cl.name, cl.phonenumber, SUM(dps.creditpayment +
dps.interestpayment) as payment

FROM client cl, credit_agreement ca, credit_payment_schedule dps

WHERE cl.id = ca.clientid AND dps.creditagreementid = ca.id

AND dps.actualpaymentdate IS NULL AND dps.plannedpaymentdate < CURRENT_DATE

GROUP BY(ca.clientid, cl.name, cl.phonenumber, dps.creditpayment + dps.interestpayment)

Создание индексов:

CREATE INDEX idx_client_id ON credit_agreement (clientid);

CREATE INDEX idx_payment_dates ON credit_payment_schedule (plannedpaymentdate, actualpaymentdate);

CREATE INDEX idx_payment_credit_agreement_id ON credit_payment_schedule (creditagreementid)

Без индекса:

Query Query History

```
1 SELECT ca.clientid, cl.name, cl.phonenumber, SUM(dps.creditpayment + dps.interestpayment) as payment
2 FROM client cl, credit_agreement ca, credit_payment_schedule dps
3 WHERE cl.id = ca.clientid AND dps.creditagreementid = ca.id
4 AND dps.actualpaymentdate IS NULL AND dps.plannedpaymentdate < CURRENT_DATE
5 GROUP BY(ca.clientid, cl.name, cl.phonenumber, dps.creditpayment + dps.interestpayment)
```

Data Output Messages Notifications

	clientid integer	name character varying (50)	phonenumber character varying (11)	payment numeric
1	3	Михаил Дмитриевич Прохоров	72298165946	36000

Total rows: 1 of 1 Query complete 00:00:00.181 Ln 1, Col 1

С индексом:

Query Query History

```
1 SELECT ca.clientid, cl.name, cl.phonenumber, SUM(dps.creditpayment + dps.interestpayment) as payment
2 FROM client cl, credit_agreement ca, credit_payment_schedule dps
3 WHERE cl.id = ca.clientid AND dps.creditagreementid = ca.id
4 AND dps.actualpaymentdate IS NULL AND dps.plannedpaymentdate < CURRENT_DATE
5 GROUP BY(ca.clientid, cl.name, cl.phonenumber, dps.creditpayment + dps.interestpayment)
6
7
8
```

Data Output Messages Notifications

	clientid integer	name character varying (50)	phonenumber character varying (11)	payment numeric
1	3	Михаил Дмитриевич Прохоров	72298165946	36000

Total rows: 1 of 1 Query complete 00:00:00.078 Ln 6, Col 1

Удаление индексов:

```
DROP INDEX idx_client_id;
```

```
DROP INDEX idx_payment_dates;
```

```
DROP INDEX idx_payment_credit_agreement_id
```

Вывод

В ходе лабораторной работы были изучены возможности написания различных запросов в PostgreSQL, а именно различные вариации запросов на выборку, включающие в себя группировку, фильтрацию, агрегацию, сортировку и присоединение. В процессе лабораторной работы были написаны, в соответствии с индивидуальным заданием запросы на выбор данных из таблицы и создание представлений. Были написаны запросы на модификацию данных с помощью подзапросов, что позволяли сделать эти запросы более автоматическими. Также, были написаны простые и составные индексы для таблиц с данными и проведено сравнение скорости выполнения запросов до и после создания индексов. Результаты показали прирост скорости выполнения запросов на чтение после использования индексов.