SQL Final Task Report

Afanasiev Sergey

02.08.2021

Contents

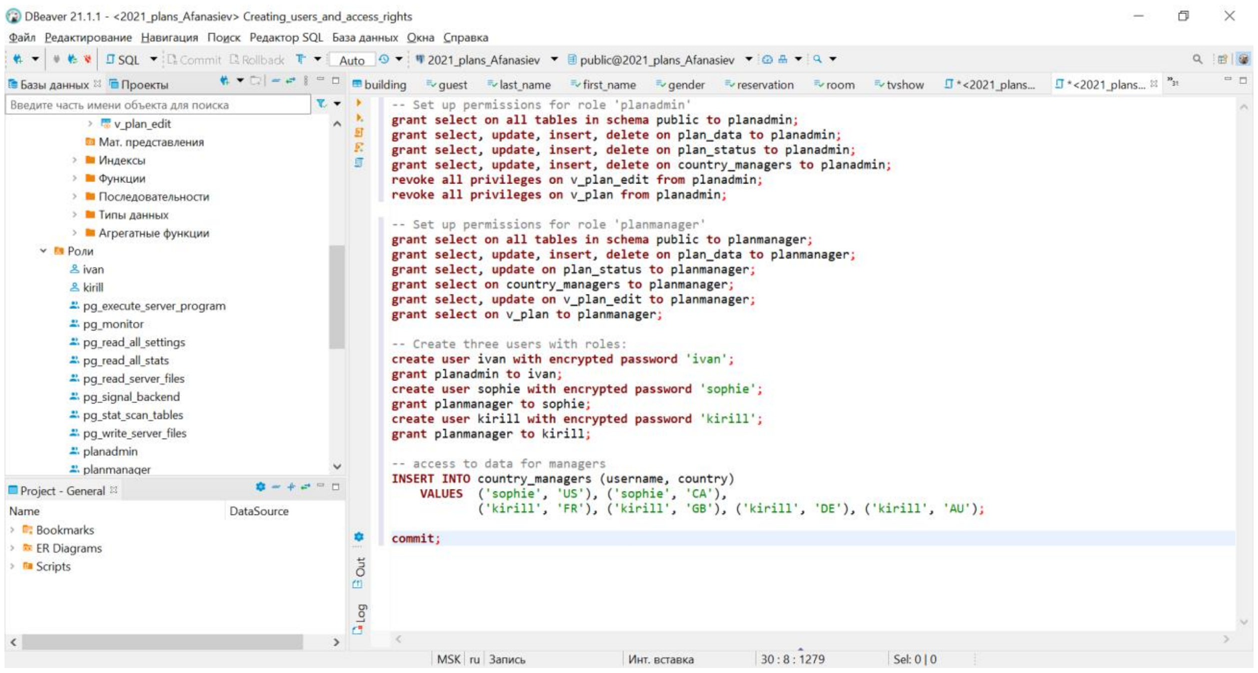
Task №1. Creating users and setting up data access rights

Add the settings script to the report under the heading ‘Task №1. Access settings’. Insert a query\queries into the report to put the data in the ‘country_managers’ table.

1.1. SQL script for creating users and setting up data access rights

```
-- Set up permissions for role 'planadmin'
grant select on all tables in schema public to planadmin;
grant select, update, insert, delete on plan_data to planadmin;
grant select, update, insert, delete on plan_status to planadmin;
grant select, update, insert, delete on country_managers to planadmin;
revoke all privileges on v_plan_edit from planadmin;
revoke all privileges on v_plan from planadmin;

-- Set up permissions for role 'planmanager'
grant select on all tables in schema public to planmanager;
grant select, update, insert, delete on plan_data to planmanager;
grant select, update on plan_status to planmanager;
grant select on country_managers to planmanager;
grant select, update on v_plan_edit to planmanager;
grant select on v_plan to planmanager;

-- Create three users with roles:
create user ivan with encrypted password 'ivan';
grant planadmin to ivan;
create user sophie with encrypted password 'sophie';
grant planmanager to sophie;
create user kirill with encrypted password 'kirill';
grant planmanager to kirill;

-- access to data for managers
INSERT INTO country_managers (username, country)
       VALUES ('sophie', 'US'), ('sophie', 'CA'),
              ('kirill', 'FR'), ('kirill', 'GB'), ('kirill', 'DE'), ('kirill', 'AU');

commit;
```

1.2. Screenshots



```
-- Set up permissions for role 'planadmin'
grant select on all tables in schema public to planadmin;
grant select, update, insert, delete on plan_data to planadmin;
grant select, update, insert, delete on plan_status to planadmin;
grant select, update, insert, delete on country_managers to planadmin;
revoke all privileges on v_plan_edit from planadmin;
revoke all privileges on v_plan from planadmin;

-- Set up permissions for role 'planmanager'
grant select on all tables in schema public to planmanager;
grant select, update, insert, delete on plan_data to planmanager;
grant select, update on plan_status to planmanager;
grant select on country_managers to planmanager;
grant select, update on v_plan_edit to planmanager;
grant select on v_plan to planmanager;

-- Create three users with roles:
create user ivan with encrypted password 'ivan';
grant planadmin to ivan;
create user sophie with encrypted password 'sophie';
grant planmanager to sophie;
create user kirill with encrypted password 'kirill';
grant planmanager to kirill;

-- access to data for managers
INSERT INTO country_managers (username, country)
    VALUES  ('sophie', 'US'), ('sophie', 'CA'),
            ('kirill', 'FR'), ('kirill', 'GB'), ('kirill', 'DE'), ('kirill', 'AU');

commit;
```

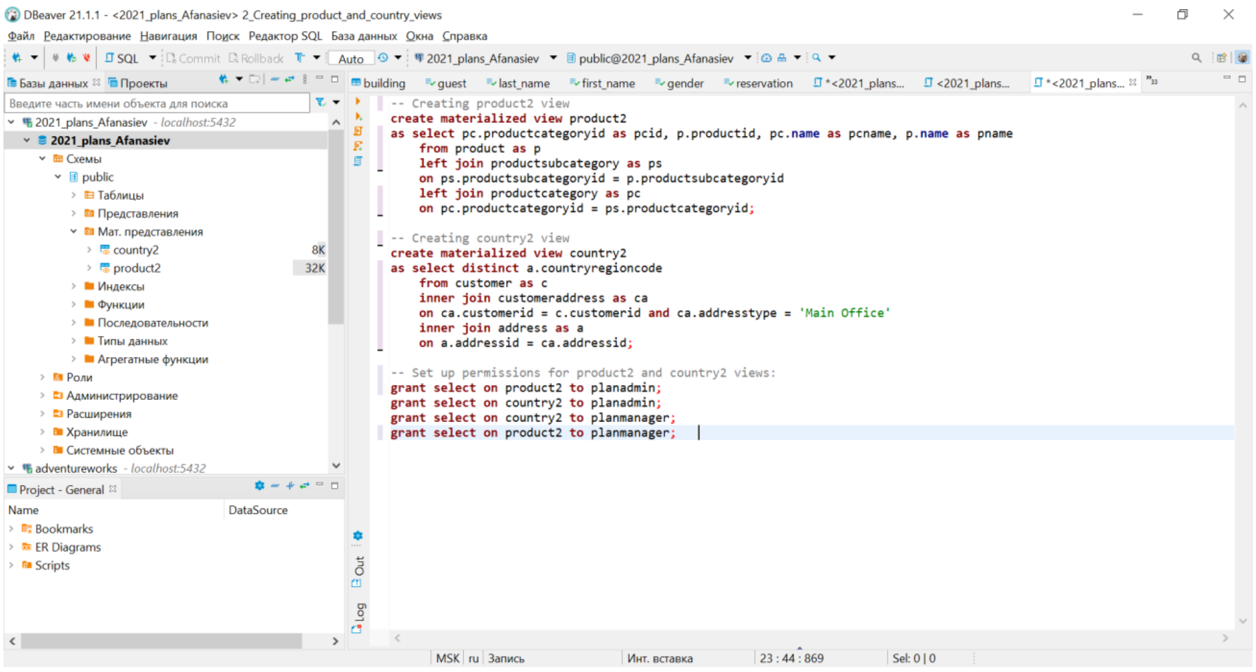
Task №2. Creating product and country views

Add sql code of the views into the report under ‘Task №2. product2 & country 2 materialized views’ heading. Also add commands to set necessary permissions.

2.1. SQL script for creating product and country views

```sql
-- Creating product2 view
create materialized view product2
as select pc.productcategoryid as pcid, p.productid, pc.name as pcname, p.name as pname
        from product as p
        left join productsubcategory as ps
        on ps.productsubcategoryid = p.productsubcategoryid
        left join productcategory as pc
        on pc.productcategoryid = ps.productcategoryid;

-- Creating country2 view
create materialized view country2
as select distinct a.countryregioncode
        from customer as c
        inner join customeraddress as ca
        on ca.customerid = c.customerid and ca.addresstype = 'Main Office'
        inner join address as a
        on a.addressid = ca.addressid;

-- Set up permissions for product2 and country2 views:
grant select on product2 to planadmin;
grant select on country2 to planadmin;
grant select on country2 to planmanager;
grant select on product2 to planmanager;
```

2.2. Screenshots


DBeaver 21.1.1 - <2021_plans_Afanasiev> 2_Creating_product_and_country_views
Файл Редактирование Навигация Поиск Редактор SQL База данных Окна Справка
SQL Commit Rollback Auto 2021_plans_Afanasiev public@2021_plans_Afanasiev
Базы данных Проекты
building guest last_name first_name gender reservation *<2021_plans... <2021_plans... *<2021_plans...
Введите часть имени объекта для поиска
2021_plans_Afanasiev - localhost:5432
2021_plans_Afanasiev
Схемы
public
Таблицы
Представления
Мат. представления
country2 8K
product2 32K
Индексы
Функции
Последовательности
Типы данных
Агрегатные функции
Роли
Администрирование
Расширения
Хранилище
Системные объекты
adventureworks - localhost:5432
Project - General
Name DataSource
Bookmarks
ER Diagrams
Scripts
Out
Log
MSK ru Запись Инт. вставка 23 : 44 : 869 Sel: 0 | 0

```sql
-- Creating product2 view
create materialized view product2
as select pc.productcategoryid as pcid, p.productid, pc.name as pcname, p.name as pname
    from product as p
    left join productsubcategory as ps
    on ps.productsubcategoryid = p.productsubcategoryid
    left join productcategory as pc
    on pc.productcategoryid = ps.productcategoryid;

-- Creating country2 view
create materialized view country2
as select distinct a.countryregioncode
    from customer as c
    inner join customeraddress as ca
    on ca.customerid = c.customerid and ca.addresstype = 'Main Office'
    inner join address as a
    on a.addressid = ca.addressid;

-- Set up permissions for product2 and country2 views:
grant select on product2 to planadmin;
grant select on country2 to planadmin;
grant select on country2 to planmanager;
grant select on product2 to planmanager;
```

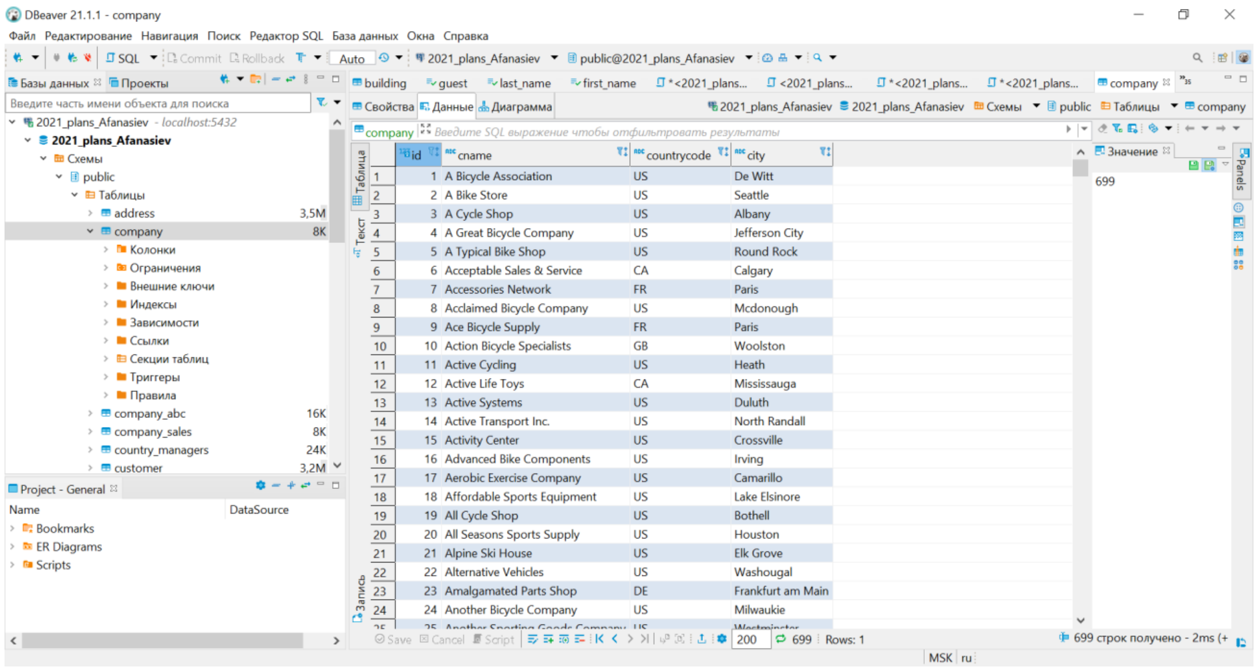
Task №3. Loading data into the company table

Include the prepared query into report under the heading ‘Task №3. Loading data into the company table’.

3.1. SQL script for loading data into the company table

```
insert into company(cname, countrycode, city)
select t.cname, t.countrycode, t.city
from (select row_number() over(partition by c.companyname order by a.countryregioncode,
a.city) as r, c.companyname as cname, a.countryregioncode as countrycode, a.city as city
       from customer c inner join customeraddress ca
       on ca.customerid = c.customerid and ca.addresstype = 'Main Office'
       inner join address a on a.addressid = ca.addressid
       where c.companyname is not null and c.firstname is null) t
where t.r = 1;
commit;
```

3.2. Screenshots



DBeaver 21.1.1 - company

	id	cname	countrycode	city
1	1	A Bicycle Association	US	De Witt
2	2	A Bike Store	US	Seattle
3	3	A Cycle Shop	US	Albany
4	4	A Great Bicycle Company	US	Jefferson City
5	5	A Typical Bike Shop	US	Round Rock
6	6	Acceptable Sales & Service	CA	Calgary
7	7	Accessories Network	FR	Paris
8	8	Acclaimed Bicycle Company	US	Mcdonough
9	9	Ace Bicycle Supply	FR	Paris
10	10	Action Bicycle Specialists	GB	Woolston
11	11	Active Cycling	US	Heath
12	12	Active Life Toys	CA	Mississauga
13	13	Active Systems	US	Duluth
14	14	Active Transport Inc.	US	North Randall
15	15	Activity Center	US	Crossville
16	16	Advanced Bike Components	US	Irving
17	17	Aerobic Exercise Company	US	Camarillo
18	18	Affordable Sports Equipment	US	Lake Elsinore
19	19	All Cycle Shop	US	Bothell
20	20	All Seasons Sports Supply	US	Houston
21	21	Alpine Ski House	US	Elk Grove
22	22	Alternative Vehicles	US	Washougal
23	23	Amalgamated Parts Shop	DE	Frankfurt am Main
24	24	Another Bicycle Company	US	Milwaukie

200 | 699 | Rows: 1

699 строк получено - 2ms (+

Task №4. Company classification by annual amount of orders

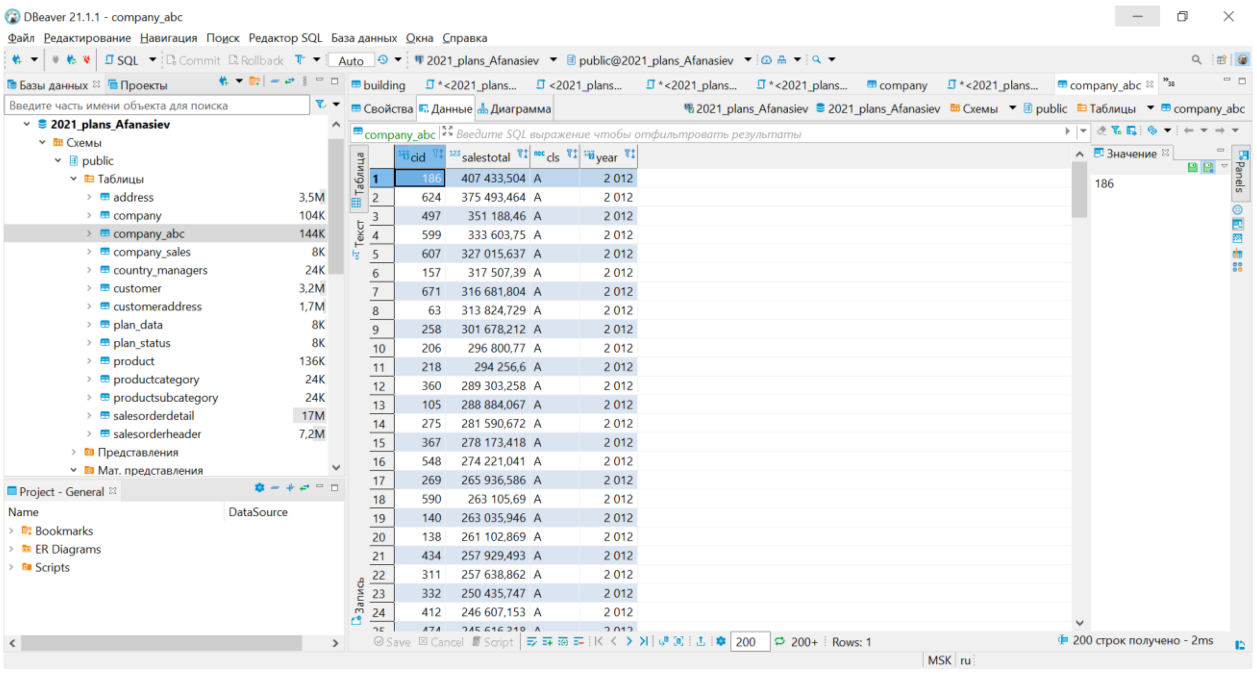
Add the designed SQL queries to the report under ‘Task №4. Company classification’ heading.

Add a screenshot of 10-20 records of company_abc.

4.1. SQL script for company classification by annual amount of orders

```
insert into company_abc
with sales as
       (select c.id as cid, extract(year from orderdate) as year, sum(so.subtotal) as
       salestotal
       from company as c
       inner join customer cs on cs.companyname = c.cname
       inner join salesorderheader as so on so.customerid = cs.customerid
              and so.orderdate >= to_date('20120101', 'yyyymmdd')
              and so.orderdate < to_date('20140101', 'yyyymmdd')
       group by c.id, extract(year from orderdate))
select s.cid, s.salestotal as salestotal,
       case
       when sum(s.salestotal) over(partition by s.year order by s.salestotal desc) /
       sum(s.salestotal) over(partition by s.year) <= 0.8 then 'A'
       when sum(s.salestotal) over(partition by s.year order by s.salestotal desc) /
       sum(s.salestotal) over(partition by s.year) <= 0.95 then 'B'
       else 'C'
       end as cls, s.year
from sales as s;
commit;
```

4.2. Screenshots



	cid	salestotal	cls	year
1	186	407 433,504	A	2 012
2	624	375 493,464	A	2 012
3	497	351 188,46	A	2 012
4	599	333 603,75	A	2 012
5	607	327 015,637	A	2 012
6	157	317 507,39	A	2 012
7	671	316 681,804	A	2 012
8	63	313 824,729	A	2 012
9	258	301 678,212	A	2 012
10	206	296 800,77	A	2 012
11	218	294 256,6	A	2 012
12	360	289 303,258	A	2 012
13	105	288 884,067	A	2 012
14	275	281 590,672	A	2 012
15	367	278 173,418	A	2 012
16	548	274 221,041	A	2 012
17	269	265 936,586	A	2 012
18	590	263 105,69	A	2 012
19	140	263 035,946	A	2 012
20	138	261 102,869	A	2 012
21	434	257 929,493	A	2 012
22	311	257 638,862	A	2 012
23	332	250 435,747	A	2 012
24	412	246 607,153	A	2 012

200 строк получено - 2ms

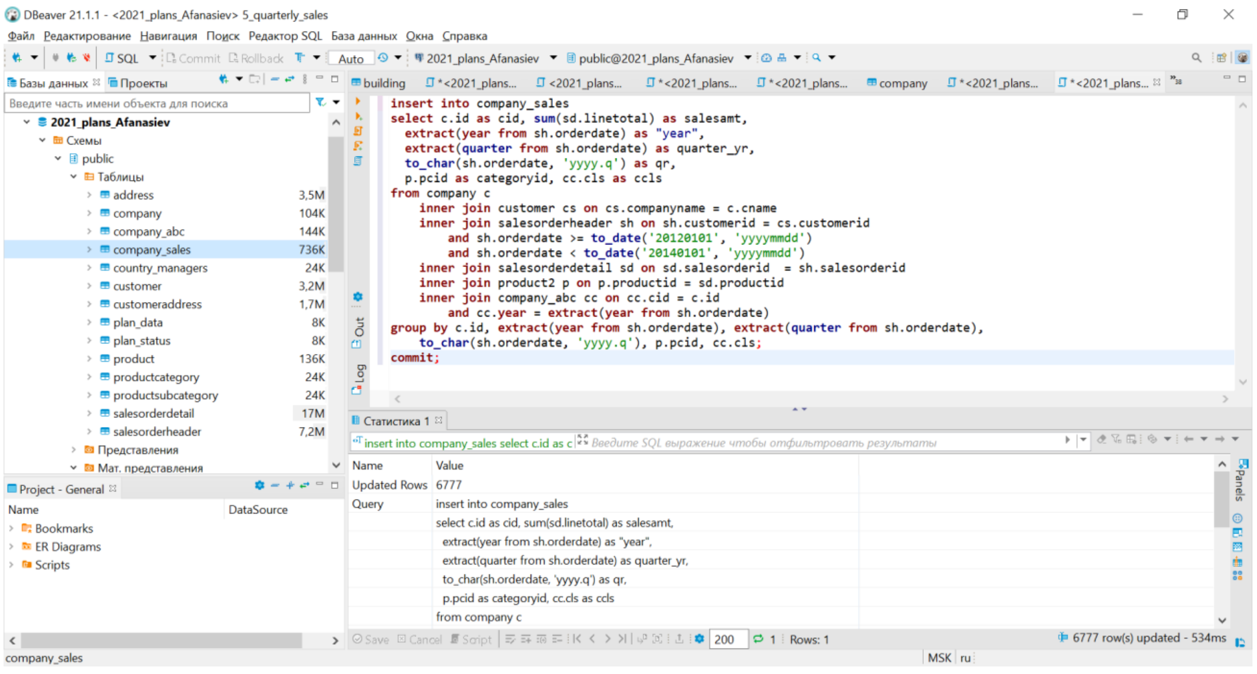
Task №5. Finding quarterly sales amount by company, and product category

Add your query into the report under “Task №5. Finding quarterly sales amount by company, and product category” heading.

5.1. SQL script for finding quarterly sales amount by company, and product category

```
insert into company_sales
select c.id as cid, sum(sd.linetotal) as salesamt, extract(year from sh.orderdate) as "year",
extract(quarter from sh.orderdate) as quarter_yr, to_char(sh.orderdate, 'yyyy.q') as qr,
p.pcid as categoryid, cc.cls as ccls
from company c
        inner join customer cs on cs.companyname = c.cname
        inner join salesorderheader sh on sh.customerid = cs.customerid
                and sh.orderdate >= to_date('20120101', 'yyyymmdd')
                and sh.orderdate < to_date('20140101', 'yyyymmdd')
        inner join salesorderdetail sd on sd.salesorderid = sh.salesorderid
        inner join product2 p on p.productid = sd.productid
        inner join company_abc cc on cc.cid = c.id
                and cc.year = extract(year from sh.orderdate)
group by c.id, extract(year from sh.orderdate), extract(quarter from sh.orderdate),
to_char(sh.orderdate, 'yyyy.q'), p.pcid, cc.cls;
commit;
```

5.2. Screenshots



```
insert into company_sales
select c.id as cid, sum(sd.linetotal) as salesamt,
  extract(year from sh.orderdate) as "year",
  extract(quarter from sh.orderdate) as quarter_yr,
  to_char(sh.orderdate, 'yyyy.q') as qr,
  p.pcid as categoryid, cc.cls as ccls
from company c
    inner join customer cs on cs.companyname = c.cname
    inner join salesorderheader sh on sh.customerid = cs.customerid
        and sh.orderdate >= to_date('20120101', 'yyyymmdd')
        and sh.orderdate < to_date('20140101', 'yyyymmdd')
    inner join salesorderdetail sd on sd.salesorderid  = sh.salesorderid
    inner join product2 p on p.productid = sd.productid
    inner join company_abc cc on cc.cid = c.id
        and cc.year = extract(year from sh.orderdate)
group by c.id, extract(year from sh.orderdate), extract(quarter from sh.orderdate),
    to_char(sh.orderdate, 'yyyy.q'), p.pcid, cc.cls;
commit;
```

Name	Value
Updated Rows	6777
Query	insert into company_sales
	select c.id as cid, sum(sd.linetotal) as salesamt,
	extract(year from sh.orderdate) as "year",
	extract(quarter from sh.orderdate) as quarter_yr,
	to_char(sh.orderdate, 'yyyy.q') as qr,
	p.pcid as categoryid, cc.cls as ccls
	from company c

```
6777 row(s) updated - 534ms
```

Task №6. Generating the initial planning data

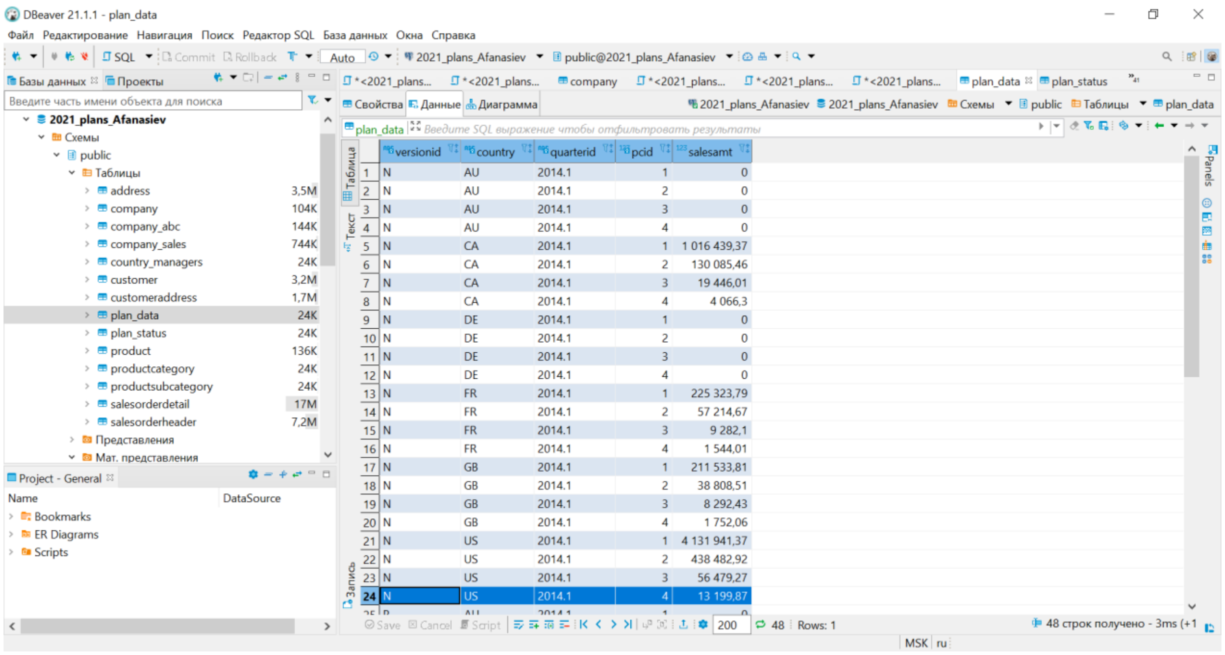
Add the start_planning function to the report under a new header - “Task №6. Initial data preparation”. Write a line with function call that you used to populate the plan_data and plan_status tables.

Add two screenshots of plan_data and plan_status contents, showing results of the function execution (P and N versions of plan should exist, status should be equal to R).

6.1. Python code for generating the initial planning data

```
import psycopg2

def start_planning(year, quarter, user, pwd):
    con = psycopg2.connect(database='2021_plans_Afanasiev', user=user, password=pwd, host='localhost')
    quarterid = str(year) + '.' + str(quarter)
    try:
        with con:
            with con.cursor() as cur:
                cur.execute("""delete from plan_data pd where pd.quarterid = %s""", [quarterid])
                cur.execute("""delete from plan_status ps where ps.quarterid = %s""", [quarterid])
                cur.execute("""insert into plan_status
                                select %s as quarterid, 'R' as status, now() as modifieddatetime,
                                %s as author, c.countryregioncode as country
                                from country2 c""",
                            [quarterid, user])
                cur.execute("""insert into plan_data
                                with country_prod as (select distinct c2.countryregioncode as country, p.pcid
                                from country2 as c2
                                cross join product2 as p
                                where p.pcid is not null),
                                sales as (select c.countrycode as country, cs.qr as quarterid,
                                cs.categoryid as pcid, sum(cs.salesamt) as salesamt
                                from company as c
                                left join company_sales as cs
                                on cs.cid = c.id and cs.ccls in ('A', 'B') and cs.year in (%s-2, %s-1)
                                and cs.quarter_yr = %s
                                group by c.countrycode, cs.qr, cs.categoryid)
                                select 'N' as versionid, cp.country, %s as quarterid, cp.pcid,
                                coalesce(avg(s.salesamt), 0) as salesamt
                                from country_prod as cp
                                left join sales as s on s.country = cp.country and s.pcid = cp.pcid
                                group by cp.country, cp.pcid""",
                            [year, year, quarter, quarterid])
                cur.execute("""insert into plan_data
                                select 'P' as versionid, pd.country as country,
                                pd.quarterid as quarterid, pd.pcid as pcid, pd.salesamt as salesamt
                                from plan_data pd
                                where pd.versionid = 'N' and pd.quarterid = %s""",
                            [quarterid])
    finally:
        con.commit()
        con.close()

if __name__ == '__main__':
    start_planning(2014, 1, 'ivan', 'ivan')
```

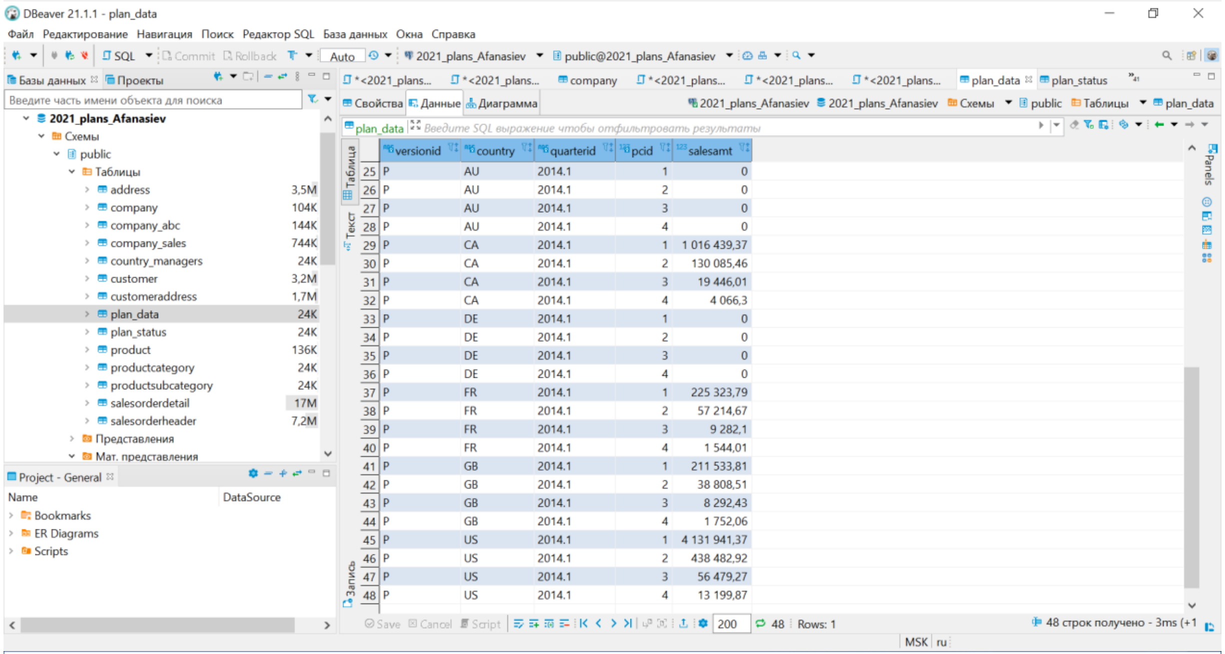
6.2. Screenshots



DBeaver 21.1.1 - plan_data

	versionid	country	quarterid	pcid	salesamt
1	N	AU	2014.1	1	0
2	N	AU	2014.1	2	0
3	N	AU	2014.1	3	0
4	N	AU	2014.1	4	0
5	N	CA	2014.1	1	1 016 439,37
6	N	CA	2014.1	2	130 085,46
7	N	CA	2014.1	3	19 446,01
8	N	CA	2014.1	4	4 066,3
9	N	DE	2014.1	1	0
10	N	DE	2014.1	2	0
11	N	DE	2014.1	3	0
12	N	DE	2014.1	4	0
13	N	FR	2014.1	1	225 323,79
14	N	FR	2014.1	2	57 214,67
15	N	FR	2014.1	3	9 282,1
16	N	FR	2014.1	4	1 544,01
17	N	GB	2014.1	1	211 533,81
18	N	GB	2014.1	2	38 808,51
19	N	GB	2014.1	3	8 292,43
20	N	GB	2014.1	4	1 752,06
21	N	US	2014.1	1	4 131 941,37
22	N	US	2014.1	2	438 482,92
23	N	US	2014.1	3	56 479,27
24	N	US	2014.1	4	13 199,87

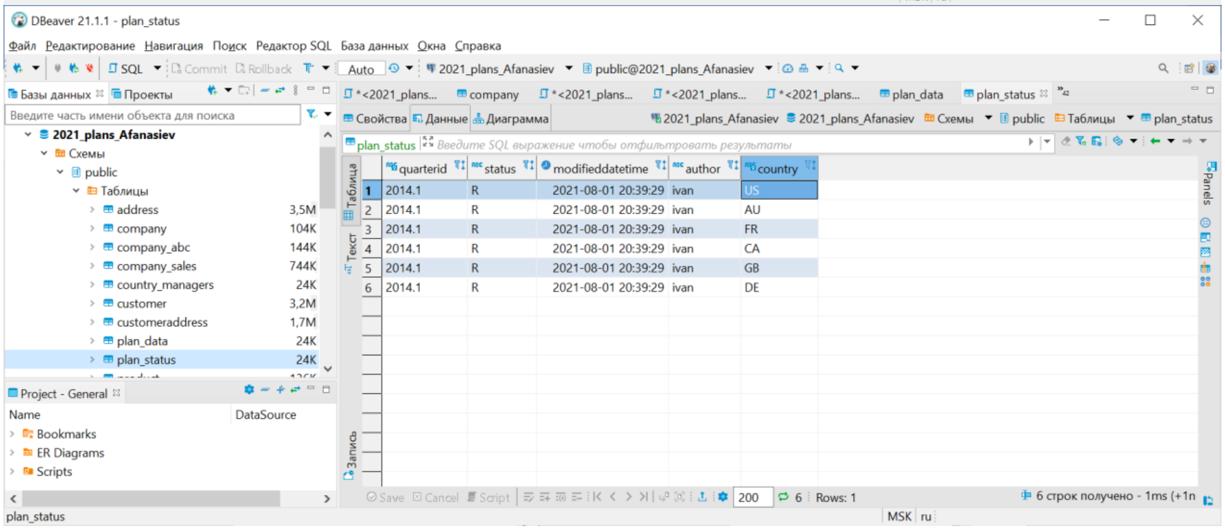
200 48 Rows: 1 — 48 строк получено - 3ms



DBeaver 21.1.1 - plan_data

	versionid	country	quarterid	pcid	salesamt
25	P	AU	2014.1	1	0
26	P	AU	2014.1	2	0
27	P	AU	2014.1	3	0
28	P	AU	2014.1	4	0
29	P	CA	2014.1	1	1 016 439,37
30	P	CA	2014.1	2	130 085,46
31	P	CA	2014.1	3	19 446,01
32	P	CA	2014.1	4	4 066,3
33	P	DE	2014.1	1	0
34	P	DE	2014.1	2	0
35	P	DE	2014.1	3	0
36	P	DE	2014.1	4	0
37	P	FR	2014.1	1	225 323,79
38	P	FR	2014.1	2	57 214,67
39	P	FR	2014.1	3	9 282,1
40	P	FR	2014.1	4	1 544,01
41	P	GB	2014.1	1	211 533,81
42	P	GB	2014.1	2	38 808,51
43	P	GB	2014.1	3	8 292,43
44	P	GB	2014.1	4	1 752,06
45	P	US	2014.1	1	4 131 941,37
46	P	US	2014.1	2	438 482,92
47	P	US	2014.1	3	56 479,27
48	P	US	2014.1	4	13 199,87

200 48 Rows: 1 — 48 строк получено - 3ms



DBeaver 21.1.1 - plan_status

	quarterid	status	modifieddatetime	author	country
1	2014.1	R	2021-08-01 20:39:29	ivan	US
2	2014.1	R	2021-08-01 20:39:29	ivan	AU
3	2014.1	R	2021-08-01 20:39:29	ivan	FR
4	2014.1	R	2021-08-01 20:39:29	ivan	CA
5	2014.1	R	2021-08-01 20:39:29	ivan	GB
6	2014.1	R	2021-08-01 20:39:29	ivan	DE

200 6 Rows: 1 — 6 строк получено - 1ms

Task №7. Changing the plan data

Add set_lock and remove_lock code into your report under “Changing plan data” header. Also provide a screenshot of v_plan_edit contents when logged in as kirill.

The screenshot should show the changed data before executing the remove_lock function.

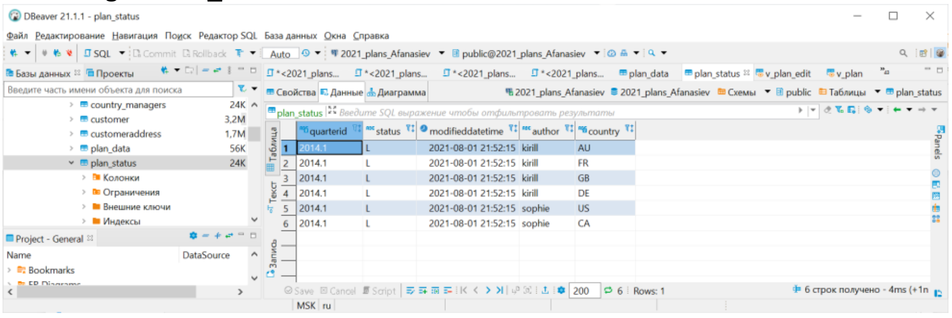
7.1. Python code for changing the plan data

```
def set_lock(year, quarter, user, pwd):
    con = psycopg2.connect(database='2021_plans_Afanasiev', user=user, password=pwd, host='localhost')
    quarterid = str(year) + '.' + str(quarter)
    try:
        with con:
            with con.cursor() as cur:
                cur.execute("""update plan_status as ps
                                set status = 'L', modifieddatetime = now(), author = current_user
                                where ps.quarterid = %s and ps.status = 'R'
                                and exists (select 1
                                            from country_managers as cm
                                            where cm.country = ps.country
                                            and cm.username = current_user)""",
                            [quarterid])
    finally:
        con.commit()
        con.close()

#=============================

def remove_lock(year, quarter, user, pwd):
    con = psycopg2.connect(database='2021_plans_Afanasiev', user=user, password=pwd, host='localhost')
    quarterid = str(year) + '.' + str(quarter)
    try:
        with con:
            with con.cursor() as cur:
                cur.execute("""update plan_status as ps
                                set status = 'R', modifieddatetime = now(), author = current_user
                                where ps.quarterid = %s and ps.status = 'L'
                                and exists (select 1
                                            from country_managers cm
                                            where cm.country = ps.country
                                            and cm.username = current_user)""",
                            [quarterid])
    finally:
        con.commit()
        con.close()

#======================================

if __name__ == '__main__':
    set_lock(2014, 1, 'kirill', 'kirill')
    set_lock(2014, 1, 'sophie', 'sophie')

#======================================

if __name__ == '__main__':
    remove_lock(2014, 1, 'kirill', 'kirill')
    remove_lock(2014, 1, 'sophie', 'sophie')
```

7.2. Screenshots

Step 1

Executing the set_lock function:

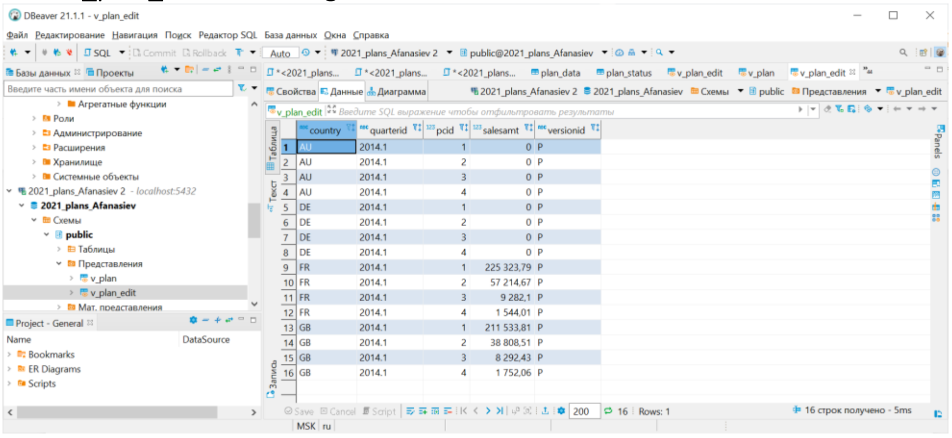


DBeaver 21.1.1 - plan_status

	quarterid	status	modifieddatetime	author	country
1	2014.1	L	2021-08-01 21:52:15	kirill	AU
2	2014.1	L	2021-08-01 21:52:15	kirill	FR
3	2014.1	L	2021-08-01 21:52:15	kirill	GB
4	2014.1	L	2021-08-01 21:52:15	kirill	DE
5	2014.1	L	2021-08-01 21:52:15	sophie	US
6	2014.1	L	2021-08-01 21:52:15	sophie	CA

6 строк получено - 4ms (+1n

View v_plan_edit for manager ‘kirill’:

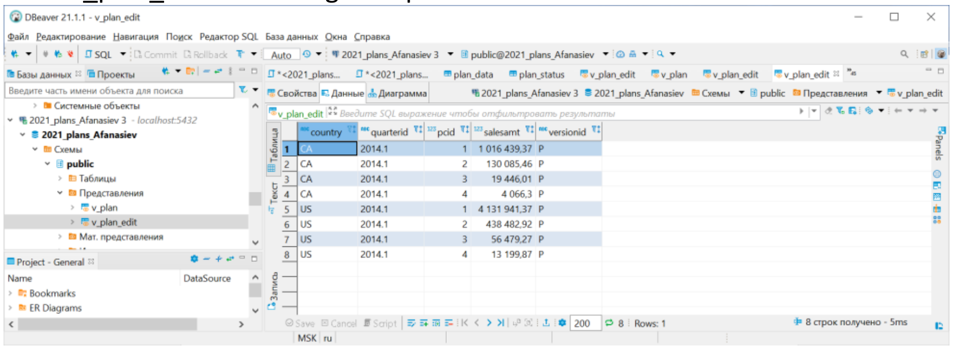


DBeaver 21.1.1 - v_plan_edit

	country	quarterid	pcid	salesamt	versionid
1	AU	2014.1	1	0	P
2	AU	2014.1	2	0	P
3	AU	2014.1	3	0	P
4	AU	2014.1	4	0	P
5	DE	2014.1	1	0	P
6	DE	2014.1	2	0	P
7	DE	2014.1	3	0	P
8	DE	2014.1	4	0	P
9	FR	2014.1	1	225 323,79	P
10	FR	2014.1	2	57 214,67	P
11	FR	2014.1	3	9 282,1	P
12	FR	2014.1	4	1 544,01	P
13	GB	2014.1	1	211 533,81	P
14	GB	2014.1	2	38 808,51	P
15	GB	2014.1	3	8 292,43	P
16	GB	2014.1	4	1 752,06	P

16 строк получено - 5ms

View v_plan_edit for manager ‘sophie’:



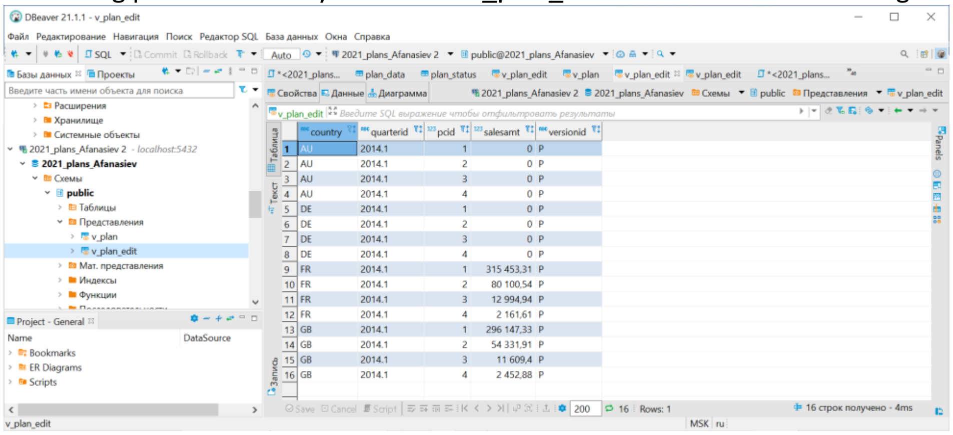
DBeaver 21.1.1 - v_plan_edit

	country	quarterid	pcid	salesamt	versionid
1	CA	2014.1	1	1 016 439,37	P
2	CA	2014.1	2	130 085,46	P
3	CA	2014.1	3	19 446,01	P
4	CA	2014.1	4	4 066,3	P
5	US	2014.1	1	4 131 941,37	P
6	US	2014.1	2	438 482,92	P
7	US	2014.1	3	56 479,27	P
8	US	2014.1	4	13 199,87	P

8 строк получено - 5ms

Step 2

Increasing planned sales by 40% in the v_plan_edit view on behalf of manager 'kirill':

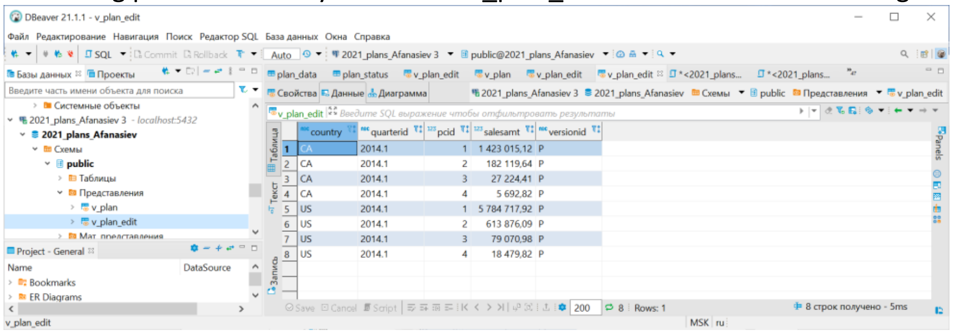


DBeaver 21.1.1 - v_plan_edit

	country	quarterid	pcid	salesamt	versionid
1	AU	2014.1	1	0	P
2	AU	2014.1	2	0	P
3	AU	2014.1	3	0	P
4	AU	2014.1	4	0	P
5	DE	2014.1	1	0	P
6	DE	2014.1	2	0	P
7	DE	2014.1	3	0	P
8	DE	2014.1	4	0	P
9	FR	2014.1	1	315 453,31	P
10	FR	2014.1	2	80 100,54	P
11	FR	2014.1	3	12 994,94	P
12	FR	2014.1	4	2 161,61	P
13	GB	2014.1	1	296 147,33	P
14	GB	2014.1	2	54 331,91	P
15	GB	2014.1	3	11 609,4	P
16	GB	2014.1	4	2 452,88	P

16 строк получено - 4ms

Increasing planned sales by 40% in the v_plan_edit view on behalf of manager 'sophie':

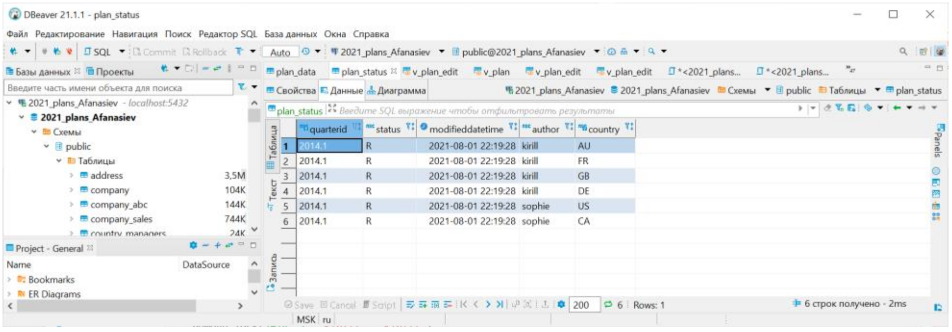

DBeaver 21.1.1 - v_plan_edit

	country	quarterid	pcid	salesamt	versionid
1	CA	2014.1	1	1 423 015,12	P
2	CA	2014.1	2	182 119,64	P
3	CA	2014.1	3	27 224,41	P
4	CA	2014.1	4	5 692,82	P
5	US	2014.1	1	5 784 717,92	P
6	US	2014.1	2	613 876,09	P
7	US	2014.1	3	79 070,98	P
8	US	2014.1	4	18 479,82	P

8 строк получено - 5ms

Step 3

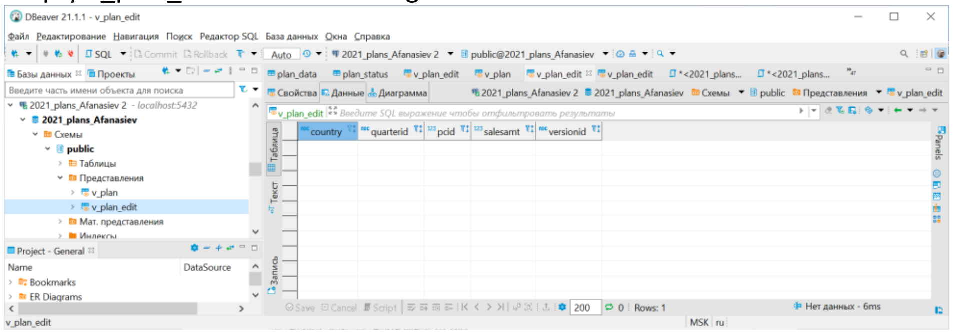
Running the remove_lock function to mark Q1 2014 as not in use (as ‘kirill’ and then as ‘sophie’):


DBeaver 21.1.1 - plan_status

	quarterid	status	modifieddatetime	author	country
1	2014.1	R	2021-08-01 22:19:28	kirill	AU
2	2014.1	R	2021-08-01 22:19:28	kirill	FR
3	2014.1	R	2021-08-01 22:19:28	kirill	GB
4	2014.1	R	2021-08-01 22:19:28	kirill	DE
5	2014.1	R	2021-08-01 22:19:28	sophie	US
6	2014.1	R	2021-08-01 22:19:28	sophie	CA

6 строк получено - 2ms

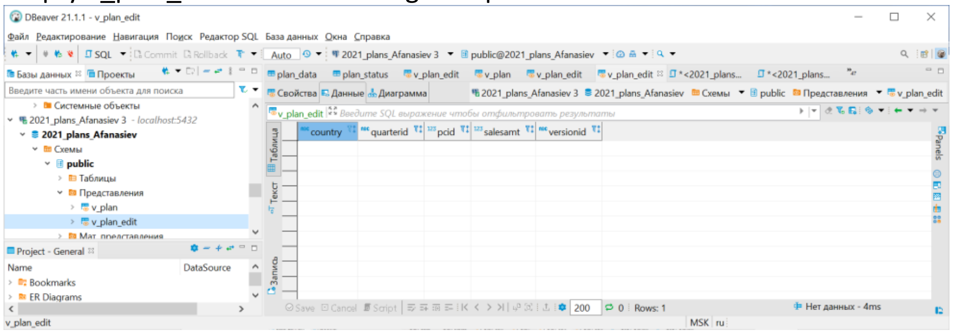
Empty v_plan_edit view for manager ‘kirill’:


DBeaver 21.1.1 - v_plan_edit

country	quarterid	pcid	salesamt	versionid

Нет данных - 6ms

Empty v_plan_edit view for manager ‘sophie’:


DBeaver 21.1.1 - v_plan_edit

country	quarterid	pcid	salesamt	versionid

Нет данных - 4ms

Task №8. Plan data approval

Add accept_plan function code to the report under “Plan data approval” heading. Also include a function call as kirill and sophie.

After logging in as sophie add a screenshot of rows in the v_plan view.

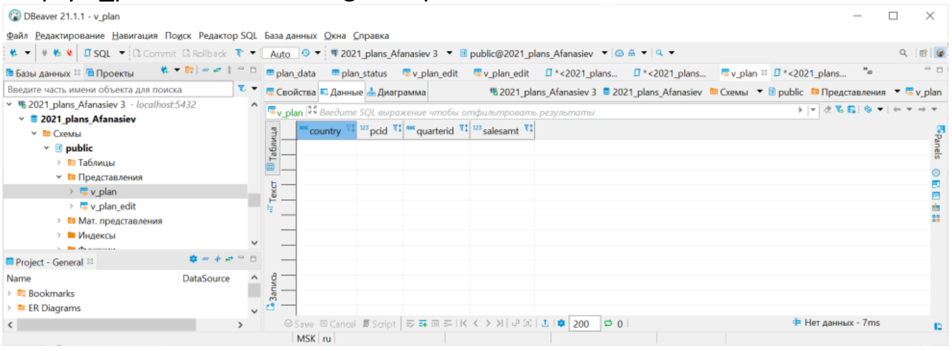
8.1. Python code for plan data approval

```
def accept_plan(year, quarter, user, pwd):
    con = psycopg2.connect(database='2021_plans_Afanasiev', user=user, password=pwd, host='localhost')
    quarterid = str(year) + '.' + str(quarter)

    try:
        with con:
            with con.cursor() as cur:
                cur.execute("""delete from plan_data as pd
                                where pd.quarterid = %s and pd.versionid = 'A'
                                and exists (select 1 from country_managers as cm
                                            where cm.country = pd.country
                                            and cm.username = current_user)
                                and exists (select 1 from plan_status as ps
                                            where ps.quarterid = pd.quarterid
                                            and ps.country = pd.country and ps.status = 'A')""",
                            [quarterid])
                cur.execute("""update plan_status as ps
                                set status = 'R', modifieddatetime = now(), author = current_user
                                where ps.quarterid = %s and ps.status = 'A'
                                and exists (select 1 from country_managers as cm
                                            where cm.country = ps.country
                                            and cm.username = current_user)""",
                            [quarterid])
                cur.execute("""insert into plan_data
                                select 'A' as versionid, pd.country, pd.quarterid, pd.pcid, pd.salesamt
                                from plan_data as pd
                                where pd.quarterid = %s and pd.versionid = 'P'
                                and exists (select 1 from country_managers as cm
                                            where cm.country = pd.country
                                            and cm.username = current_user)
                                and exists (select 1 from plan_status as ps
                                            where ps.quarterid = pd.quarterid
                                            and ps.country = pd.country and ps.status = 'R')""",
                            [quarterid])
                cur.execute("""update plan_status as ps set status = 'A',
                                modifieddatetime = now(), author = current_user
                                where ps.quarterid = %s and ps.status = 'R'
                                and exists (select 1
                                            from country_managers as cm
                                            where cm.country = ps.country
                                            and cm.username = current_user)""",
                            [quarterid])
    finally:
        con.commit()
        con.close()

#=======================================

if __name__ == '__main__':
    accept_plan(2014, 1, "kirill", "kirill")
    accept_plan(2014, 1, "sophie", "sophie")
```

8.2. Screenshots

Login as manager 'sophie'

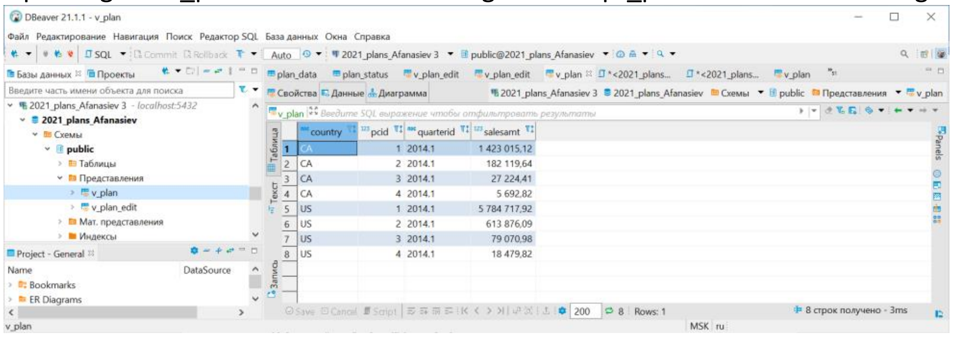
Empty v_plan view for manager 'sophie':



	country	pcid	quarterid	salesamt

Нет данных - 7ms

Updating the v_plan view after executing the accept_plan function for manager 'sophie':

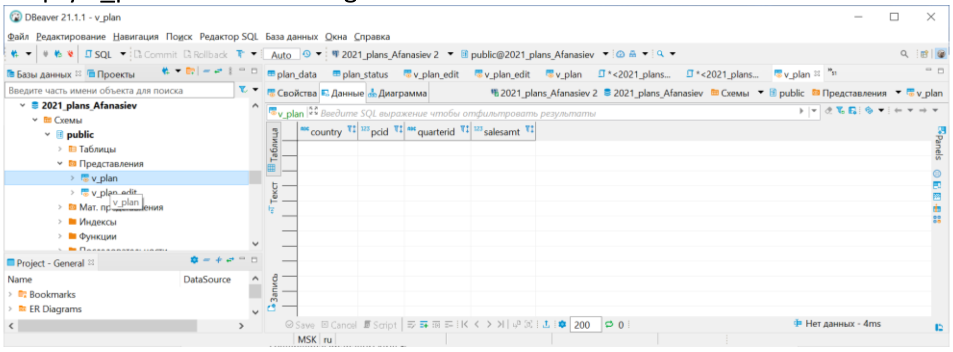


	country	pcid	quarterid	salesamt
1	CA	1	2014.1	1 423 015,12
2	CA	2	2014.1	182 119,64
3	CA	3	2014.1	27 224,41
4	CA	4	2014.1	5 692,82
5	US	1	2014.1	5 784 717,92
6	US	2	2014.1	613 876,09
7	US	3	2014.1	79 070,98
8	US	4	2014.1	18 479,82

8 строк получено - 3ms

Login as manager 'kirill'

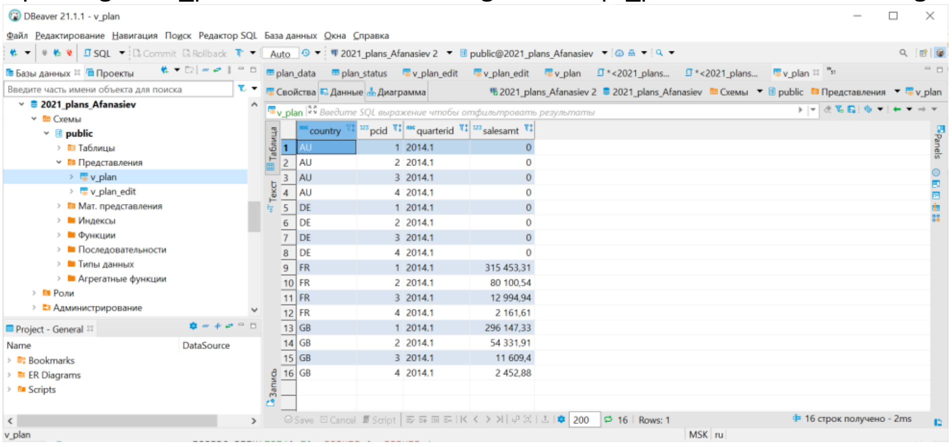
Empty v_plan view for manager 'kirill':



	country	pcid	quarterid	salesamt

Нет данных - 4ms

Updating the v_plan view after executing the accept_plan function for manager 'kirill':



	country	pcid	quarterid	salesamt
1	AU	1	2014.1	0
2	AU	2	2014.1	0
3	AU	3	2014.1	0
4	AU	4	2014.1	0
5	DE	1	2014.1	0
6	DE	2	2014.1	0
7	DE	3	2014.1	0
8	DE	4	2014.1	0
9	FR	1	2014.1	315 453,31
10	FR	2	2014.1	80 100,54
11	FR	3	2014.1	12 994,94
12	FR	4	2014.1	2 161,61
13	GB	1	2014.1	296 147,33
14	GB	2	2014.1	54 331,91
15	GB	3	2014.1	11 609,4
16	GB	4	2014.1	2 452,88

16 строк получено - 2ms

Task №9. Data preparation for plan-fact analysis in Q1 2014

Add a header “Data preparation for plan-fact analysis in Q1 2014”.

Write which approach you chose.

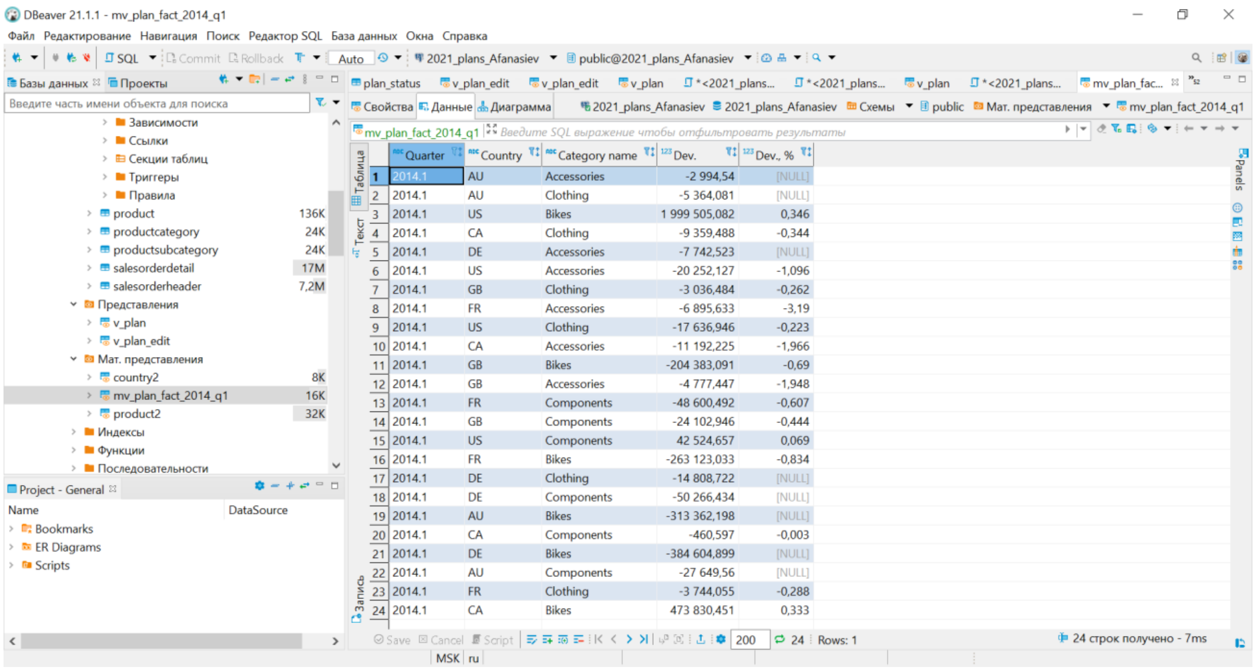
Include SQL code of the new materialized view together with a screenshot showing data in mv_plan_fact_2014_q1 view.

I chose approach 2: calculating actual data using salesorderheader and ordersalesdetail tables without using company_sales.

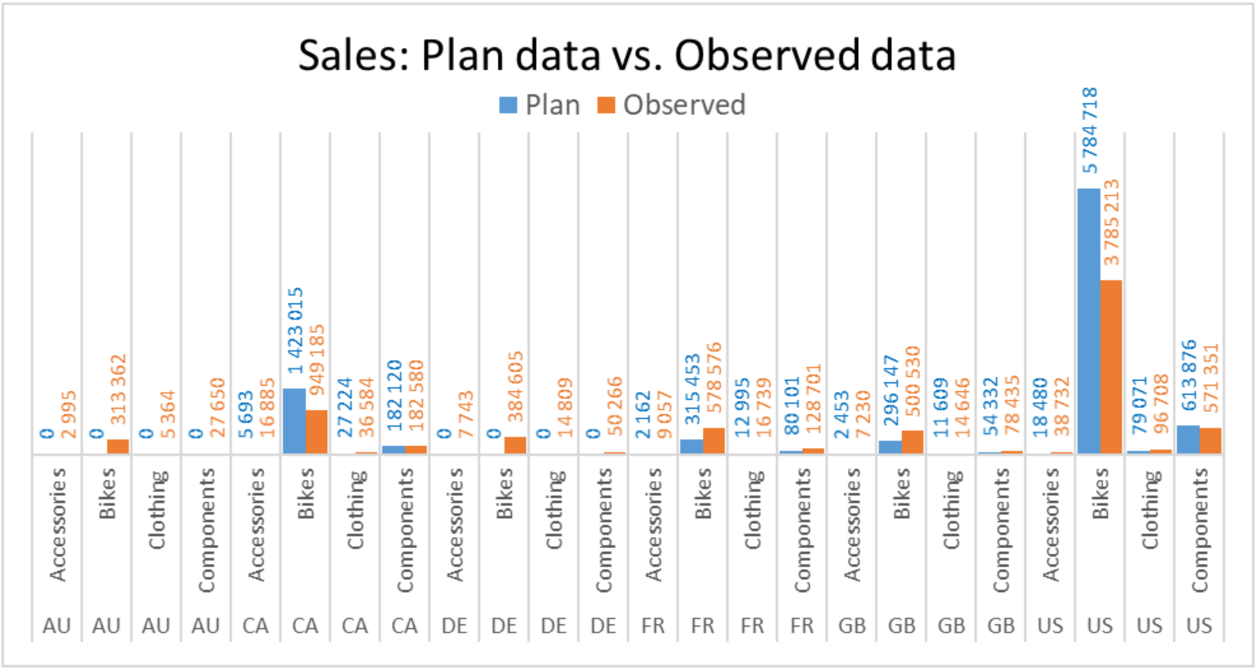
9.1. SQL script for data preparation for plan-fact analysis in Q1 2014

```
create materialized view mv_plan_fact_2014_q1
as with country_prod as
        (select distinct '2014.1' as quarterid, c2.countryregioncode as crc, p2.pcid,
        p2.pcname
        from country2 as c2 cross join product2 as p2
        where p2.pcid is not null),
        observed_data as
                (select to_char(sh.orderdate, 'yyyy.q') as quarterid, c.countrycode,
                p2.pcid, sum(sd.linetotal) as sales
                from company c
                inner join customer as cs on cs.companyname = c.cname
                inner join salesorderheader as sh on sh.customerid = cs.customerid
                        and sh.orderdate >= to_date('20140101', 'yyyymmdd')
                        and sh.orderdate < to_date('20140401', 'yyyymmdd')
                inner join salesorderdetail as sd on sd.salesorderid  = sh.salesorderid
                inner join product2 as p2 on p2.productid = sd.productid
                where exists (select 1
                        from company_abc cc
                        where cc.cid = c.id and cc.year = 2013 and cc.cls in ('A', 'B'))
        group by to_char(sh.orderdate, 'yyyy.q'), c.countrycode, p2.pcid)
select
        cp.quarterid as "Quarter",
        cp.crc as "Country",
        cp.pcname as "Category name",
        pd.salesamt - od.sales as "Dev.",
        case when coalesce(pd.salesamt, 0) = 0 then null
                else (pd.salesamt - od.sales) / pd.salesamt
                end as "Dev., %"
from country_prod cp
        left join plan_data pd on pd.quarterid = cp.quarterid
                and pd.country = cp.crc and pd.pcid = cp.pcid and pd.versionid = 'A'
        left join observed_data od on od.countrycode = cp.crc
                and od.quarterid = cp.quarterid and od.pcid = cp.pcid;

grant select on mv_plan_fact_2014_q1 to planadmin;

grant select on mv_plan_fact_2014_q1 to planmanager;
```

9.2. Screenshots



	Quarter	Country	Category name	Dev.	Dev., %
1	2014.1	AU	Accessories	-2 994,54	[NULL]
2	2014.1	AU	Clothing	-5 364,081	[NULL]
3	2014.1	US	Bikes	1 999 505,082	0,346
4	2014.1	CA	Clothing	-9 359,488	-0,344
5	2014.1	DE	Accessories	-7 742,523	[NULL]
6	2014.1	US	Accessories	-20 252,127	-1,096
7	2014.1	GB	Clothing	-3 036,484	-0,262
8	2014.1	FR	Accessories	-6 895,633	-3,19
9	2014.1	US	Clothing	-17 636,946	-0,223
10	2014.1	CA	Accessories	-11 192,225	-1,966
11	2014.1	GB	Bikes	-204 383,091	-0,69
12	2014.1	GB	Accessories	-4 777,447	-1,948
13	2014.1	FR	Components	-48 600,492	-0,607
14	2014.1	GB	Components	-24 102,946	-0,444
15	2014.1	US	Components	42 524,657	0,069
16	2014.1	FR	Bikes	-263 123,033	-0,834
17	2014.1	DE	Clothing	-14 808,722	[NULL]
18	2014.1	DE	Components	-50 266,434	[NULL]
19	2014.1	AU	Bikes	-313 362,198	[NULL]
20	2014.1	CA	Components	-460,597	-0,003
21	2014.1	DE	Bikes	-384 604,899	[NULL]
22	2014.1	AU	Components	-27 649,56	[NULL]
23	2014.1	FR	Clothing	-3 744,055	-0,288
24	2014.1	CA	Bikes	473 830,451	0,333


Sales: Plan data vs. Observed data
🟦 Plan 🟧 Observed
AU Accessories 0 2 995
AU Bikes 0 313 362
AU Clothing 0 5 364
AU Components 0 27 650
CA Accessories 5 693 16 885
CA Bikes 1 423 015 949 185
CA Clothing 27 224 36 584
CA Components 182 120 182 580
DE Accessories 0 7 743
DE Bikes 0 384 605
DE Clothing 0 14 809
DE Components 0 50 266
FR Accessories 2 162 9 057
FR Bikes 315 453 578 576
FR Clothing 12 995 16 739
FR Components 80 101 128 701
GB Accessories 2 453 7 230
GB Bikes 296 147 500 530
GB Clothing 11 609 14 646
GB Components 54 332 78 435
US Accessories 18 480 38 732
US Bikes 5 784 718 3 785 213
US Clothing 79 071 96 708
US Components 613 876 571 351