REPORT

Task №1. Access settings

```
insert into country_managers
values ('sophie', 'US'),
       ('sophie', 'CA'),
       ('kirill', 'AU'),
       ('kirill', 'FR'),
       ('kirill', 'GB'),
       ('kirill', 'DE');
```

Task №2. Creating product and country views

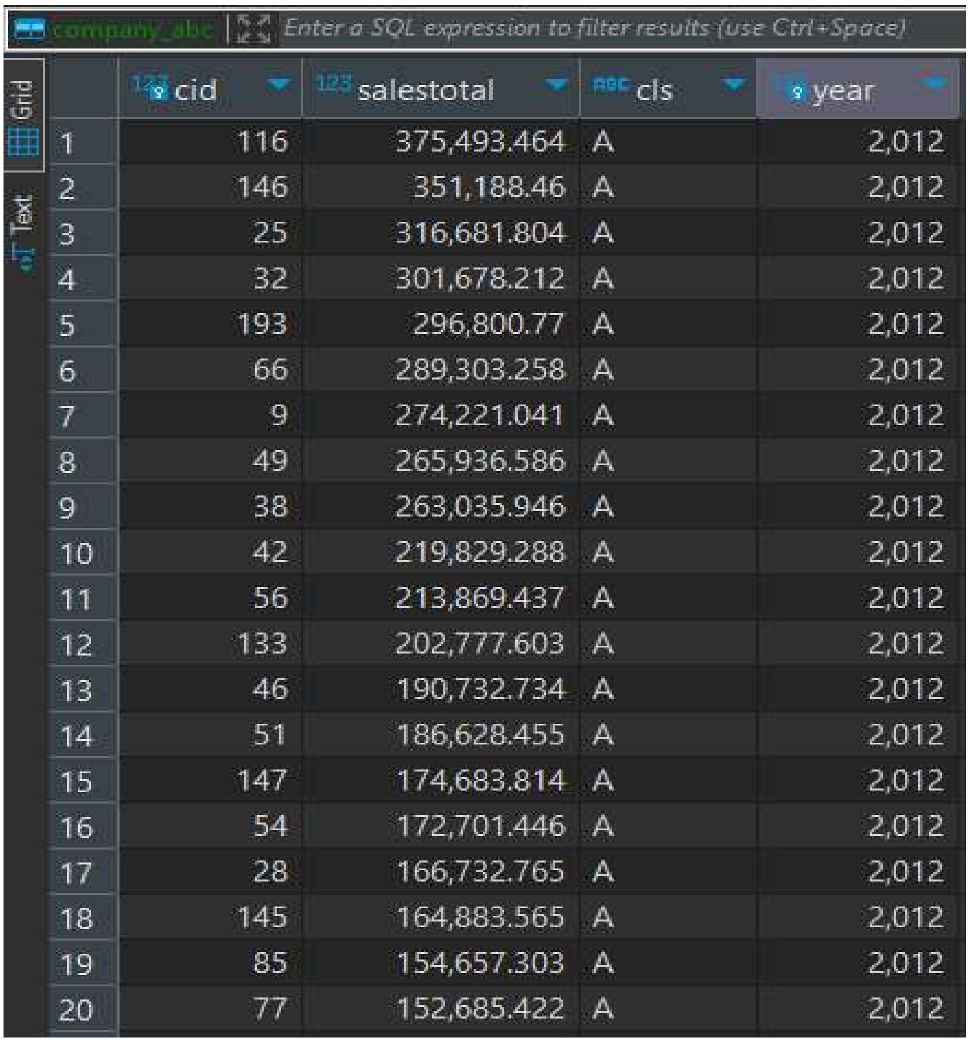
```
create materialized view product2 as
      select
            pc.productcategoryid as pcid,
            p.productid as productid,
            pc."name" as pcname,
            p."name" as pname
      from
            product p
                  join productsubcategory as psc using(productsubcategoryid)
                  join productcategory as pc using(productcategoryid);

create materialized view country2 as
      select distinct a.countryregioncode as countrycode
      from
            customer as c
                  join customeraddress as ca using(customerid)
                  join address as a using(addressid)
      where
            ca.addresstype = 'Main Office';

grant select
on table public.product2, public.country2
to planadmin, planmanager;
```

Task №3. Loading data into the company table

```
insert into company (cname, countrycode, city)
      select
            c.companyname as cname,
            a.countryregioncode as countrycode,
            a.city as city
      from
            customer as c
                  join customeraddress as ca using(customerid)
                  join address as a using(addressid)
      where
            ca.addresstype = 'Main Office';
```

Task №4. Company classification by annual amount of purchases

```
insert into company_abc (cid, salestotal, cls, "year")
      select id, s_t, class, year
      from (
            select *,
                  case
                        when tab1.s_t_r <= tab1.s_a then 'A'
                        when tab1.s_t_r <= tab1.s_b then 'B'
                        else 'C'
                  end as class
            from
                  (select com.id,
                        com.cname,
                        extract (year FROM soh.orderdate) as year,
                        sum(sum(soh.subtotal)) over(partition by extract (year FROM
soh.orderdate)) as S,
                        0.8 * sum(sum(soh.subtotal)) over(partition by extract (year FROM
soh.orderdate)) as S_a,
                        0.95 * sum(sum(soh.subtotal)) over(partition by extract (year FROM
soh.orderdate)) as S_b,
                        sum(soh.subtotal) as S_T,
                        sum(sum(soh.subtotal)) over(partition by extract (year FROM
soh.orderdate) order by sum(soh.subtotal) desc) as S_T_R
                  from
                        company as com
                        join customer as cus on cus.companyname = com.cname
                        join salesorderheader as soh on cus.customerid =
                                                   soh.customerid
                        group by com.id, extract (year FROM soh.orderdate)
                        having extract (year FROM soh.orderdate) in (2012, 2013)) as
                                                                  tab1) as tab2;
```



company_abc | Enter a SQL expression to filter results (use Ctrl+Space)

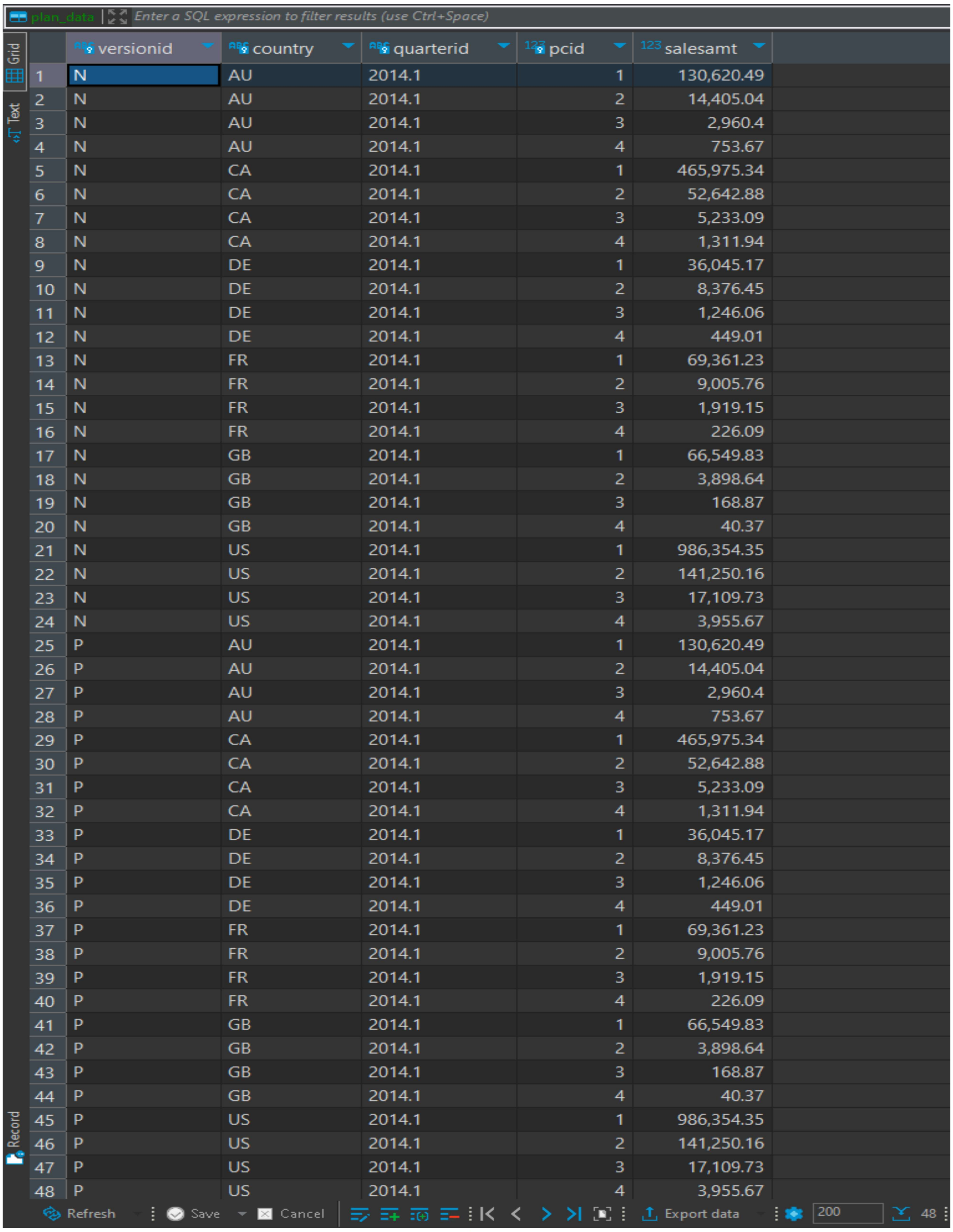
	cid	salestotal	cls	year
1	116	375,493.464	A	2,012
2	146	351,188.46	A	2,012
3	25	316,681.804	A	2,012
4	32	301,678.212	A	2,012
5	193	296,800.77	A	2,012
6	66	289,303.258	A	2,012
7	9	274,221.041	A	2,012
8	49	265,936.586	A	2,012
9	38	263,035.946	A	2,012
10	42	219,829.288	A	2,012
11	56	213,869.437	A	2,012
12	133	202,777.603	A	2,012
13	46	190,732.734	A	2,012
14	51	186,628.455	A	2,012
15	147	174,683.814	A	2,012
16	54	172,701.446	A	2,012
17	28	166,732.765	A	2,012
18	145	164,883.565	A	2,012
19	85	154,657.303	A	2,012
20	77	152,685.422	A	2,012

Task №5. Finding quarterly volume of purchases made by each company, and the product category

```
insert into company_sales
	select
		cid,
		salesamt,
		qr_sales.year,
		qr_sales.quarter_yr,
		qr_sales.qr,
		qr_sales.categoryid,
		cabc.cls as ccls
	from
		(select
			cus.companyname as cname,
			pr2.pcid as categoryid,
			extract(year from soh.orderdate) as year,
			extract(quarter from soh.orderdate) as quarter_yr,
			extract(year from orderdate)::varchar(4)||'.'||extract(quarter from
			orderdate)::varchar(1) as qr,
			sum(sod.linetotal) as salesamt
		from
			salesorderheader soh
				join salesorderdetail sod using(salesorderid)
				join product2 pr2 using(productid)
				join customer as cus using(customerid)
		where extract(year from soh.orderdate) in (2012, 2013)
		group by
			extract(year from soh.orderdate),
			extract(quarter from soh.orderdate),
			qr,
			cus.companyname,
			pr2.pcid) as qr_sales
	join company as com using(cname)
	join company_abc cabc on (com.id = cabc.cid and qr_sales.year = cabc.year)
```

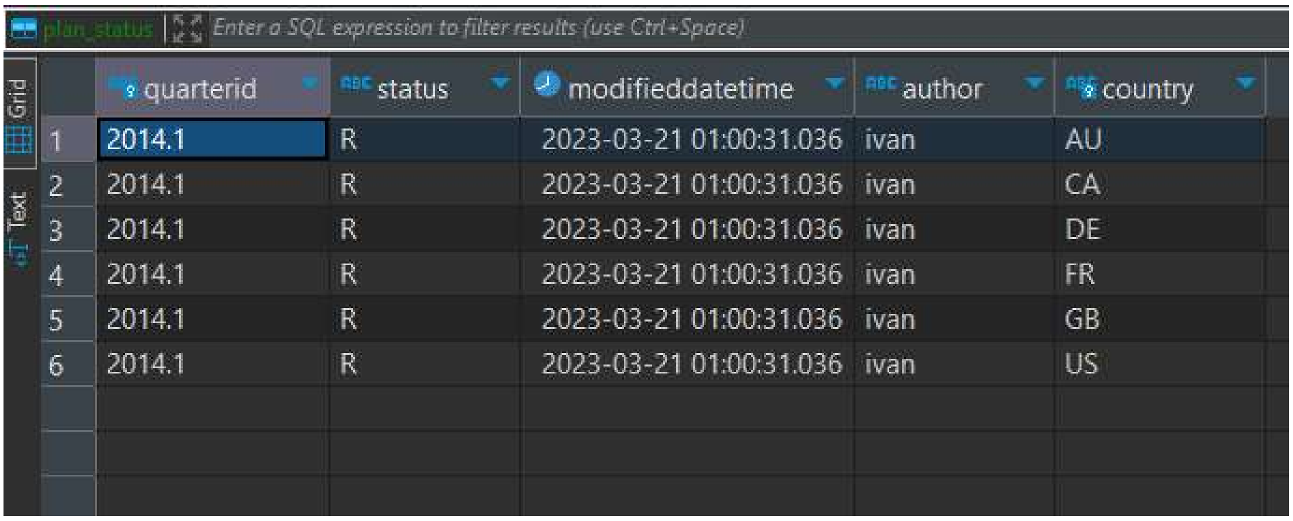
Task №6. Generating the initial planning data

```python
import psycopg2

def start_planning(year, quarter, user, pwd):
    # Create a connection
    con = psycopg2.connect(database='2023_plans_Lazarev',
                           user=user,
                           password=pwd,
                           host='localhost')
    # Create a client-side cursor
    cur = con.cursor()

    # Delete plan data from the plan_data table related to the target year and quarter
    quarterid = f'{year}.{quarter}'
    query1 = f'delete from plan_data where quarterid = %s;'
    cur.execute(query1, [quarterid])

    # Delete all records related to the target quarter from the plan_status table
    query2 = f"delete from plan_status where quarterid like '____.%s';"
    cur.execute(query2, [quarter])

    query3 = f'select * from country2'
    cur.execute(query3)

    countries = [record[0] for record in cur]

    planning_status = 'R'
    query4 = '''
             insert into plan_status (quarterid, status, country)
             values (%s, %s, %s);
    '''
    for country in countries:
        cur.execute(query4, [quarterid, planning_status, country])

    not_changed_version = 'N'
    query5 = '''
       insert into public.plan_data
            select %s, countrycode as country, %s, categoryid as pcid, plan as salesamt
            from
            (select *,
                case
                    when tab3.flag = 1 and tab3."axs_plan_2014.1" is null then salesamt
                    when tab3.flag = 1 and tab3."axs_plan_2014.1" is not null then
                        "axs_plan_2014.1"
                end as plan
            from
            (select *,
                case
                    when lead(tab2.categoryid) over (partition by tab2.countrycode,
                                tab2.categoryid order by qr) = tab2.categoryid then 0
                    else 1
                end    as flag
                from
                (select *,  0.5*(tab1.salesamt + lag(tab1.salesamt, 1) over (partition
            by tab1.countrycode, tab1.categoryid order by qr)) as "axs_plan_2014.1"
                    from (
                            select cs.year, cs.quarter_yr, cs.qr, c.countrycode,
                                   cs.categoryid, sum(salesamt) as salesamt
                            from company_sales cs join company c on cs.cid = c.id
                            where cs.ccls != 'C'
                            group by cs.year, cs.quarter_yr, cs.qr, c.countrycode,
                                     cs.categoryid
                            having cs.quarter_yr = %s
                            order by categoryid, countrycode, year
                        ) as tab1) as tab2) as tab3) as tab4
            where plan is not null;
    '''
    cur.execute(query5, [not_changed_version, quarterid, quarter])
    changed_version = 'P'
    query6 = '''
    insert into public.plan_data
        select %s, country, quarterid, pcid, salesamt
        from plan_data;
    '''
    cur.execute(query6, [changed_version])
    con.commit()
    con.close()

Function call:
start_planning(2014, 1, 'ivan', 'ivan')
```



	versionid	country	quarterid	pcid	salesamt
1	N	AU	2014.1	1	130,620.49
2	N	AU	2014.1	2	14,405.04
3	N	AU	2014.1	3	2,960.4
4	N	AU	2014.1	4	753.67
5	N	CA	2014.1	1	465,975.34
6	N	CA	2014.1	2	52,642.88
7	N	CA	2014.1	3	5,233.09
8	N	CA	2014.1	4	1,311.94
9	N	DE	2014.1	1	36,045.17
10	N	DE	2014.1	2	8,376.45
11	N	DE	2014.1	3	1,246.06
12	N	DE	2014.1	4	449.01
13	N	FR	2014.1	1	69,361.23
14	N	FR	2014.1	2	9,005.76
15	N	FR	2014.1	3	1,919.15
16	N	FR	2014.1	4	226.09
17	N	GB	2014.1	1	66,549.83
18	N	GB	2014.1	2	3,898.64
19	N	GB	2014.1	3	168.87
20	N	GB	2014.1	4	40.37
21	N	US	2014.1	1	986,354.35
22	N	US	2014.1	2	141,250.16
23	N	US	2014.1	3	17,109.73
24	N	US	2014.1	4	3,955.67
25	P	AU	2014.1	1	130,620.49
26	P	AU	2014.1	2	14,405.04
27	P	AU	2014.1	3	2,960.4
28	P	AU	2014.1	4	753.67
29	P	CA	2014.1	1	465,975.34
30	P	CA	2014.1	2	52,642.88
31	P	CA	2014.1	3	5,233.09
32	P	CA	2014.1	4	1,311.94
33	P	DE	2014.1	1	36,045.17
34	P	DE	2014.1	2	8,376.45
35	P	DE	2014.1	3	1,246.06
36	P	DE	2014.1	4	449.01
37	P	FR	2014.1	1	69,361.23
38	P	FR	2014.1	2	9,005.76
39	P	FR	2014.1	3	1,919.15
40	P	FR	2014.1	4	226.09
41	P	GB	2014.1	1	66,549.83
42	P	GB	2014.1	2	3,898.64
43	P	GB	2014.1	3	168.87
44	P	GB	2014.1	4	40.37
45	P	US	2014.1	1	986,354.35
46	P	US	2014.1	2	141,250.16
47	P	US	2014.1	3	17,109.73
48	P	US	2014.1	4	3,955.67

Picture 1: plan_data table



plan_status | Enter a SQL expression to filter results (use Ctrl+Space)

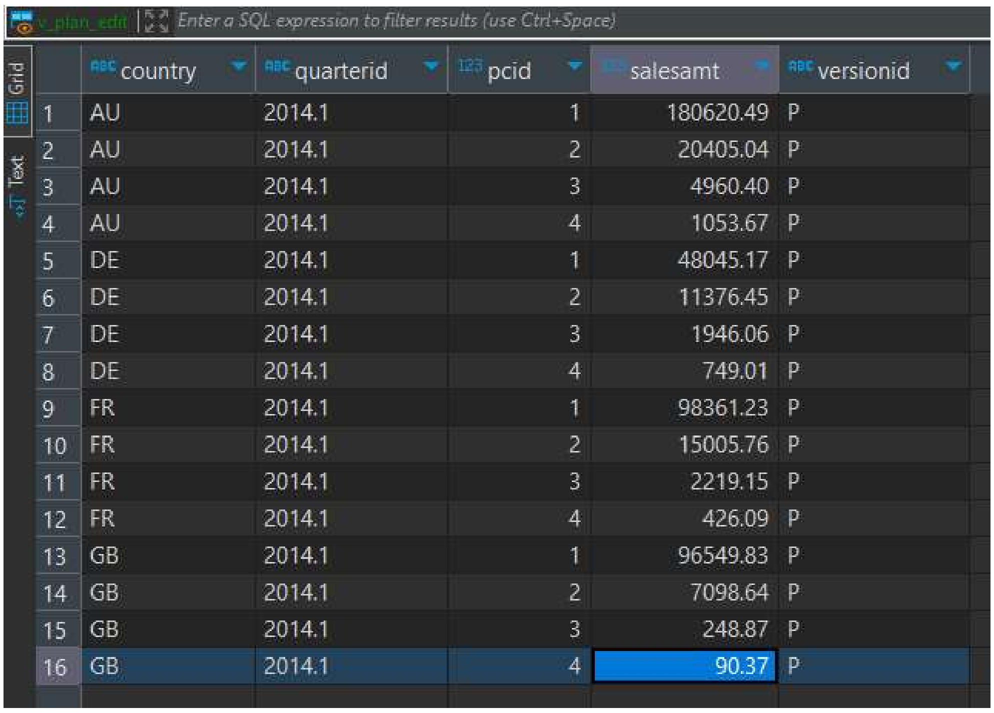
	quarterid	status	modifieddatetime	author	country
1	2014.1	R	2023-03-21 01:00:31.036	ivan	AU
2	2014.1	R	2023-03-21 01:00:31.036	ivan	CA
3	2014.1	R	2023-03-21 01:00:31.036	ivan	DE
4	2014.1	R	2023-03-21 01:00:31.036	ivan	FR
5	2014.1	R	2023-03-21 01:00:31.036	ivan	GB
6	2014.1	R	2023-03-21 01:00:31.036	ivan	US

Picture 2 plan_status table

Task №7. Changing the plan data

```python
def set_lock(year, quarter, user, pwd):
    # Create a connection
    con = psycopg2.connect(database='2023_plans_Lazarev',
                           user=user,
                           password=pwd,
                           host='localhost')
    # Create a client-side cursor
    cur = con.cursor()

    quarterid = f'{year}.{quarter}'

    cur.execute('select current_user;')
    current_user = list(cur)[0][0]

    cur.execute('select current_timestamp;')
    current_time = list(cur)[0][0]

    query1 = 'select * from country_managers;'
    cur.execute(query1)

    countries = tuple(record[1] for record in cur
                      if current_user in record)

    query2 = '''
        update
            plan_status
        set
            status = %s,
            modifieddatetime = %s,
            author = %s
        where
            quarterid = %s and
            country in %s;
    '''
    lock_status = 'L'
    cur.execute(query2, [lock_status, current_time,
                         current_user, quarterid,
                         countries])
    con.commit()
    con.close()
```

```
def remove_lock(year, quarter, user, pwd):
    # Create a connection
    con = psycopg2.connect(database='2023_plans_Lazarev',
                           user=user,
                           password=pwd,
                           host='localhost')
    # Create a client-side cursor
    cur = con.cursor()

    quarterid = f'{year}.{quarter}'
    cur.execute('select current_user;')
    current_user = list(cur)[0][0]

    cur.execute('select current_timestamp;')
    current_time = list(cur)[0][0]

    query1 = 'select * from country_managers;'
    cur.execute(query1)

    countries = tuple(record[1] for record in cur
                       if current_user in record)



    query2 = '''
            update
                plan_status
            set
                status = %s,
                modifieddatetime = %s,
                author = %s
            where
                quarterid = %s and
                country in %s;
        '''
    planning_status = 'R'
    cur.execute(query2, [planning_status, current_time,
                         current_user, quarterid,
                         countries])
    con.commit()
    con.close()
```

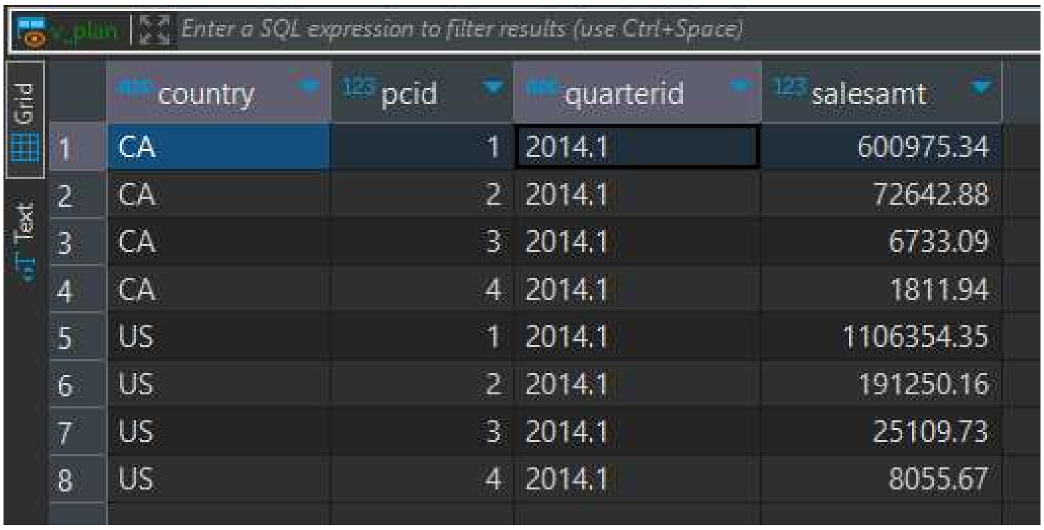


	country	quarterid	pcid	salesamt	versionid
1	AU	2014.1	1	180620.49	P
2	AU	2014.1	2	20405.04	P
3	AU	2014.1	3	4960.40	P
4	AU	2014.1	4	1053.67	P
5	DE	2014.1	1	48045.17	P
6	DE	2014.1	2	11376.45	P
7	DE	2014.1	3	1946.06	P
8	DE	2014.1	4	749.01	P
9	FR	2014.1	1	98361.23	P
10	FR	2014.1	2	15005.76	P
11	FR	2014.1	3	2219.15	P
12	FR	2014.1	4	426.09	P
13	GB	2014.1	1	96549.83	P
14	GB	2014.1	2	7098.64	P
15	GB	2014.1	3	248.87	P
16	GB	2014.1	4	90.37	P

Picture 3: v_plan_edit view for user "kirill"

Task №8. Plan data approval

```
def accept_plan(year, quarter, user, pwd):
    # Create a connection
    con = psycopg2.connect(database='2023_plans_Lazarev',
                           user=user,
                           password=pwd,
                           host='localhost')
    # Create a client-side cursor
    cur = con.cursor()

    quarterid = f'{year}.{quarter}'

    cur.execute('select current_user;')
    current_user = list(cur)[0][0]

    cur.execute('select current_timestamp;')
    current_time = list(cur)[0][0]

    query1 = 'select * from country_managers;'
    cur.execute(query1)

    countries = tuple(record[1] for record in cur
                      if current_user in record)
    # Clear the A version of plan data for specific quarter and countries
    # accessible to the current user
    a_version = 'A'
    query2 = '''
        delete
            from plan_data
            where
                versionid = %s and
                quarterid = %s and
                country in %s;
    '''
    cur.execute(query2, [a_version, quarterid,
                         countries])
    # Read data available to the current user
    # from the version P and save its copy
    # as the version A
    p_version = 'P'
    query3 = '''
        insert into public.plan_data
                select %s, country, quarterid, pcid, salesamt
                from plan_data
                where
                    versionid = %s and
                    quarterid = %s and
                    country in %s;
    '''
    cur.execute(query3, [a_version, p_version,
                         quarterid, countries])
    # Change the status of the processed from 'R' to 'A'
    query4 = '''
        update plan_status
        set
            status = %s,
            modifieddatetime = %s,
            author = %s
        where
            quarterid = %s and
            country in %s;
            '''
    planning_status = 'A'
    cur.execute(query4, [planning_status, current_time,
                         current_user, quarterid,
                         countries])
    con.commit()
    con.close()

Function call:
accept_plan(2014, 1, 'sophie', 'sophie')
accept_plan(2014, 1, 'kirill', 'kirill')
```



	country	pcid	quarterid	salesamt
1	CA	1	2014.1	600975.34
2	CA	2	2014.1	72642.88
3	CA	3	2014.1	6733.09
4	CA	4	2014.1	1811.94
5	US	1	2014.1	1106354.35
6	US	2	2014.1	191250.16
7	US	3	2014.1	25109.73
8	US	4	2014.1	8055.67

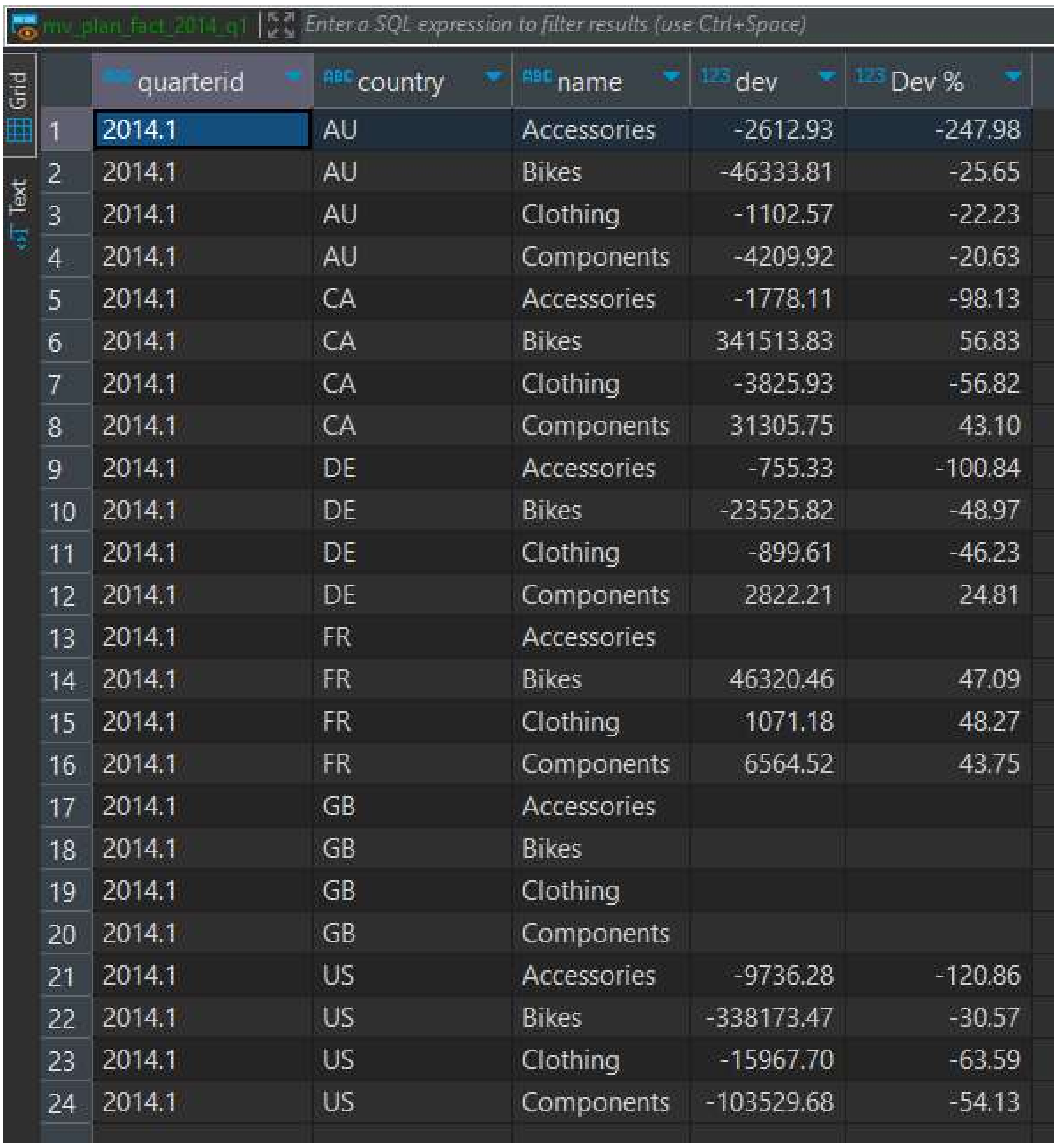
Picture 4: v_plan view for user "sophie"

Task №9. Comparison between the planned and actual sales in Q1 2014

I chose an option : Calculate actual data using salesorderheader and salesorderdetail tables without using company_sales.

```
create materialized view mv_plan_fact_2014_q1 as
select pcid_only.quarterid, pcid_only.country, pcat."name", pcid_only.Dev, pcid_only."Dev %"
from
      (select vp.quarterid,
              vp.country,
               vp.pcid,
          case
                when vp.salesamt is not null and fact.salesamt_fact is not null then
                        round(vp.salesamt - fact.salesamt_fact, 2)
                else null
                end as Dev,
                case
                when vp.salesamt is not null and fact.salesamt_fact is not null then
                round((vp.salesamt - fact.salesamt_fact) / vp.salesamt * 100, 2)
                else null
                end as "Dev %"
          from v_plan vp
          left join
          (select countrycode as country, categoryid as pcid, qr quarterid, sum(salesamt)
          salesamt_fact
                from (
                        select cus.companyname as cname, pr2.pcid as categoryid,
                        extract(year from orderdate)::varchar(4)||'.'||extract(quarter from
                        orderdate)::varchar(1) as qr,
                        sum(sod.linetotal) as salesamt
                        from salesorderheader soh
                        join salesorderdetail sod using(salesorderid)
                        join product2 pr2 using(productid)
                        join customer as cus using(customerid)
                        where extract(year from soh.orderdate) = 2014 and
                              extract(quarter from soh.orderdate) = 1
          group by extract(year from soh.orderdate), extract(quarter from soh.orderdate), qr,
                               cus.companyname, pr2.pcid) as sales_2014
      join company as com using(cname)
```

```
        where id in (select distinct cid from company_abc ca
                     where cls in ('A', 'B') and year = 2013
                     order by cid)
        group by categoryid, qr, countrycode
        order by countrycode) as fact
        on fact.country = vp.country and fact.quarterid = vp.quarterid and vp.pcid = fact.pcid)
        as pcid_only
    join productcategory as pcat on pcat.productcategoryid = pcid_only.pcid
    order by pcid_only.country, pcat."name";
```



	quarterid	country	name	dev	Dev %
1	2014.1	AU	Accessories	-2612.93	-247.98
2	2014.1	AU	Bikes	-46333.81	-25.65
3	2014.1	AU	Clothing	-1102.57	-22.23
4	2014.1	AU	Components	-4209.92	-20.63
5	2014.1	CA	Accessories	-1778.11	-98.13
6	2014.1	CA	Bikes	341513.83	56.83
7	2014.1	CA	Clothing	-3825.93	-56.82
8	2014.1	CA	Components	31305.75	43.10
9	2014.1	DE	Accessories	-755.33	-100.84
10	2014.1	DE	Bikes	-23525.82	-48.97
11	2014.1	DE	Clothing	-899.61	-46.23
12	2014.1	DE	Components	2822.21	24.81
13	2014.1	FR	Accessories		
14	2014.1	FR	Bikes	46320.46	47.09
15	2014.1	FR	Clothing	1071.18	48.27
16	2014.1	FR	Components	6564.52	43.75
17	2014.1	GB	Accessories		
18	2014.1	GB	Bikes		
19	2014.1	GB	Clothing		
20	2014.1	GB	Components		
21	2014.1	US	Accessories	-9736.28	-120.86
22	2014.1	US	Bikes	-338173.47	-30.57
23	2014.1	US	Clothing	-15967.70	-63.59
24	2014.1	US	Components	-103529.68	-54.13

Picture 5: mv_plan_fact_2014_q1 materialized view