Task №1. Access settings

**Granting Permissions to planadmin**

-- Grant SELECT permission on all tables

**GRANT** **SELECT** **ON** **ALL** **TABLES** **IN** **SCHEMA** public **TO** planadmin;

-- Grant SELECT, UPDATE, INSERT, DELETE on plan\_data

**GRANT** **SELECT**, **UPDATE**, **INSERT**, **DELETE** **ON** plan\_data **TO** planadmin;

-- Grant SELECT, UPDATE, INSERT, DELETE on plan\_status

**GRANT** **SELECT**, **UPDATE**, **INSERT**, **DELETE** **ON** plan\_status **TO** planadmin;

-- Grant SELECT, UPDATE, INSERT, DELETE on country\_managers

**GRANT** **SELECT**, **UPDATE**, **INSERT**, **DELETE** **ON** country\_managers **TO** planadmin;

-- Grant SELECT on v\_plan\_edit, v\_plan

**GRANT** **SELECT** **ON** v\_plan\_edit **TO** planadmin;

**GRANT** **SELECT** **ON** v\_plan **TO** planadmin;

**Granting Permissions to planmanager**

-- Grant SELECT permission on all tables

**GRANT** **SELECT** **ON** **ALL** **TABLES** **IN** **SCHEMA** public **TO** planmanager;

-- Grant SELECT, UPDATE, INSERT, DELETE on plan\_data

**GRANT** **SELECT**, **UPDATE**, **INSERT**, **DELETE** **ON** plan\_data **TO** planmanager;

-- Grant SELECT, UPDATE on plan\_status

**GRANT** **SELECT**, **UPDATE** **ON** plan\_status **TO** planmanager;

-- Grant SELECT on country\_managers

**GRANT** **SELECT** **ON** country\_managers **TO** planmanager;

-- Grant SELECT, UPDATE on v\_plan\_edit

**GRANT** **SELECT**, **UPDATE** **ON** v\_plan\_edit **TO** planmanager;

-- Grant SELECT on v\_plan

**GRANT** **SELECT** **ON** v\_plan **TO** planmanager;

**Creating users**

-- Create user Ivan as an administrator

**CREATE** **USER** ivan **WITH** **PASSWORD** **'password\_for\_ivan'**;

-- Create user Sophie as a manager

**CREATE** **USER** sophie **WITH** **PASSWORD** **'password\_for\_sophie'**;

-- Create user Kirill as a manager

**CREATE** **USER** kirill **WITH** **PASSWORD** **'password\_for\_kirill'**;

**Put the data into the ‘country\_managers’ table**

-- Insert records for Sophie

**INSERT** **INTO** country\_managers (username, country)

**VALUES** (**'sophie'**, **'US'**),

(**'sophie'**, **'CA'**);

-- Insert records for Kirill

**INSERT** **INTO** country\_managers (username, country)

**VALUES** (**'kirill'**, **'FR'**),

(**'kirill'**, **'GB'**),

(**'kirill'**, **'DE'**),

(**'kirill'**, **'AU'**);

Task №2. product2 & country 2 materialized views

**Creating Materialized View product2**

**CREATE** **MATERIALIZED** **VIEW** product2 **AS**

**SELECT**

pc.productcategoryid **AS** pcid,

p.productid,

pc.**name** **AS** pcname,

p.**name** **AS** pname

**FROM**

productcategory pc

**JOIN**

productsubcategory psc **ON** psc.productcategoryid = pc.productcategoryid

**JOIN**

product p **ON** psc.productsubcategoryid = p.productsubcategoryid;

**Creating Materialized View country2**

**CREATE** **MATERIALIZED** **VIEW** country2 **AS**

**SELECT** **DISTINCT**

a.countryregioncode **AS** countrycode

**FROM**

customeraddress ca

**JOIN**

address a **ON** ca.addressid = a.addressid

**WHERE**

ca.addresstype = **'Main Office'**;

**Granting permissions**

-- Grant SELECT permission on product2 view to managers and administrators

**GRANT** **SELECT** **ON** product2 **TO** planmanager, planadmin;

-- Grant SELECT permission on country2 view to managers and administrators

**GRANT** **SELECT** **ON** country2 **TO** planmanager, planadmin;

**GRANT** planadmin **TO** ivan;

**GRANT** planmanager **TO** sophie;

**GRANT** planmanager **TO** kirill;

Task №3. Loading data into the company table

**INSERT** **INTO** company (cname, countrycode, city)

**SELECT**

*c*.companyname,

*a*.countryregioncode,

*a*.city

**FROM**

customer *c*

**JOIN**

customeraddress *ca* **ON** *c*.customerid = *ca*.customerid

**JOIN**

address *a* **ON** *ca*.addressid = *a*.addressid

**WHERE**

*ca*.addresstype = **'Main Office'**

**AND** *c*.gender **IS** **NULL**;

Task №4. Company classification

**WITH** *sales\_data* **AS** (

**SELECT**

*c*.id **AS** *cid*,

**EXTRACT**(**YEAR** **FROM** *soh*.orderdate) **AS** **year**,

**SUM**(*soh*.subtotal) **AS** *salestotal*

**FROM**

salesorderheader *soh*

**JOIN**

customer *cust* **ON** *soh*.customerid = *cust*.customerid

**JOIN**

company *c* **ON** *cust*.companyname = *c*.cname

**WHERE**

**EXTRACT**(**YEAR** **FROM** *soh*.orderdate) **IN** (2012, 2013)

**GROUP** **BY**

*c*.id, **EXTRACT**(**YEAR** **FROM** soh.orderdate)

),

totals **AS** (

**SELECT**

**year**,

**SUM**(salestotal) **AS** total\_sales

**FROM**

sales\_data

**GROUP** **BY**

**year**

),

boundaries **AS** (

**SELECT**

**year**,

total\_sales \* 0.8 **AS** Sa,

total\_sales \* 0.95 **AS** Sb

**FROM**

totals

),

grouped **AS** (

**SELECT**

sd.cid,

sd.**year**,

sd.salestotal,

**SUM**(sd.salestotal) **OVER** (**PARTITION** **BY** sd.**year** **ORDER** **BY** sd.salestotal **DESC**) **AS** running\_total,

b.Sa,

b.Sb

**FROM**

sales\_data sd

**JOIN**

boundaries b **ON** sd.**year** = b.**year**

)

**INSERT** **INTO** company\_abc (cid, salestotal, cls, **year**)

**SELECT**

cid,

salestotal,

**CASE**

**WHEN** running\_total <= Sa **THEN** **'A'**

**WHEN** running\_total <= Sb **THEN** **'B'**

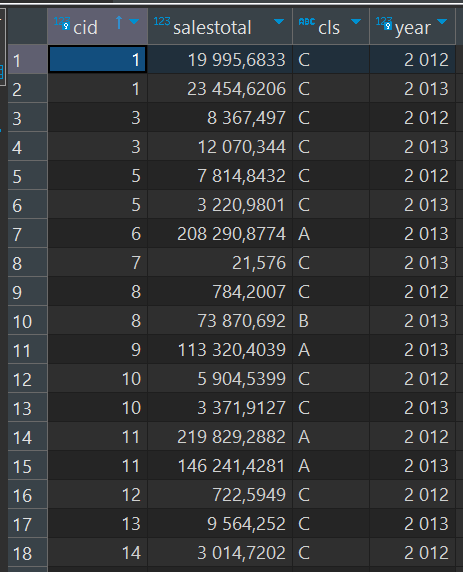
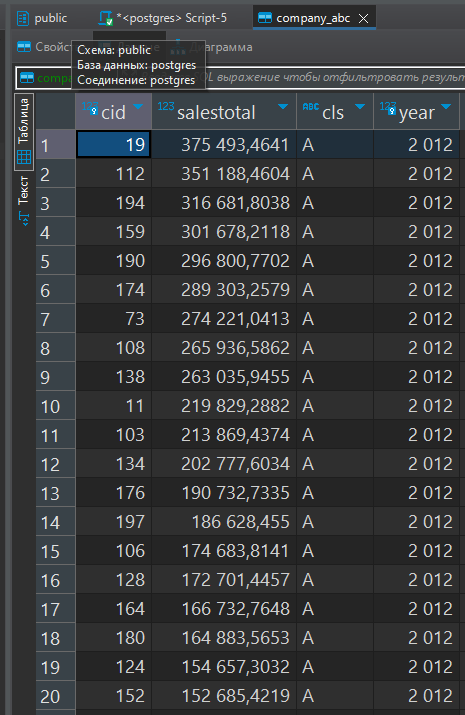
**ELSE** **'C'**

**END** **AS** cls,

**year**

**FROM**

grouped;



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

**INSERT** **INTO** company\_sales (cid, salesamt, **year**, quarter\_yr, qr, categoryid, ccls)

**SELECT**

***c***.id **AS** *cid*,

**SUM**(***sod***.linetotal) **AS** *salesamt*,

**EXTRACT**(**YEAR** **FROM** ***soh***.orderdate) **AS** **year**,

**EXTRACT**(**QUARTER** **FROM** ***soh***.***orderdate***) **AS** quarter\_yr,

**CONCAT**(**EXTRACT**(**YEAR** **FROM** soh.orderdate), **'.'**, **EXTRACT**(**QUARTER** **FROM** soh.orderdate)) **AS** qr,

p.pcid **AS** categoryid,

cab.cls **AS** ccls

**FROM**

salesorderdetail sod

**JOIN**

salesorderheader soh **ON** sod.salesorderid = soh.salesorderid

**JOIN**

product2 p **ON** sod.productid = p.productid

**JOIN**

customer cust **ON** soh.customerid = cust.customerid

**JOIN**

company c **ON** cust.companyname = c.cname

**LEFT** **JOIN**

company\_abc cab **ON** c.id = cab.cid **AND** **EXTRACT**(**YEAR** **FROM** soh.orderdate) = cab.**year**

**WHERE**

**EXTRACT**(**YEAR** **FROM** soh.orderdate) **IN** (2012, 2013)

**GROUP** **BY**

c.id, **EXTRACT**(**YEAR** **FROM** soh.orderdate), **EXTRACT**(**QUARTER** **FROM** soh.orderdate), p.pcid, cab.cls;

Task №6. Initial data preparation

**Python code**

import psycopg2

from psycopg2.extensions import cursor

def del\_plan\_data(cur: cursor, year: int, quarter: int):

    delete\_plan\_data\_query = """

    DELETE FROM plan\_data

    WHERE quarterid = %s;"""

    cur.execute(delete\_plan\_data\_query, (f"{year}.{quarter}",))

def del\_plan\_status(cur: cursor, year: int, quarter: int):

    delete\_plan\_status\_query = """

    DELETE FROM plan\_status

    WHERE quarterid = %s;"""

    cur.execute(delete\_plan\_status\_query, (f"{year}.{quarter}",))

def insert\_plan\_dstatus(cur: cursor, year: int, quarter: int, user: str):

    insert\_plan\_status\_query = """

    INSERT INTO plan\_status

        (quarterid, country, status, modifieddatetime, author)

    SELECT DISTINCT

    %s AS quarterid, countrycode AS country, 'R' AS status, NOW(), %s AS author

    FROM country2;"""

    cur.execute(insert\_plan\_status\_query, (f"{year}.{quarter}", user))

def insert\_plan\_data(cur: psycopg2.extensions.cursor, year: int, quarter: int):

    generate\_and\_copy\_plan\_data\_query = """

    WITH avg\_sales AS (

        SELECT c.countrycode AS country, cs.qr AS quarterid, cs.categoryid AS pcid, AVG(cs.salesamt)

            FILTER (WHERE cs.ccls IN ('A', 'B'))

            OVER (PARTITION BY c.countrycode, cs.categoryid) AS avg\_sales

        FROM company\_sales cs

        JOIN company c ON cs.cid = c.id

        -- JOIN plan\_status ps ON ps.quarterid = cs.qr AND ps.country = c.countrycode

        -- WHERE quarterid = %s

    )

    INSERT INTO plan\_data (versionid, country, quarterid, pcid, salesamt)

    SELECT 'N' AS versionid, country, %s AS quarterid, pcid, COALESCE(avg\_sales, 0)

    FROM avg\_sales

    UNION ALL

    SELECT 'P' AS versionid, country, %s AS quarterid, pcid, COALESCE(avg\_sales, 0)

    FROM avg\_sales

    ON CONFLICT (quarterid, country, pcid, versionid) DO NOTHING;"""

    # SELECT \* FROM avg\_sales;"""

    cur.execute(

        generate\_and\_copy\_plan\_data\_query,

        (f"{year}.{quarter}", f"{year}.{quarter}", f"{year}.{quarter}"),

    )

def start\_planning(year, quarter, user, pwd):

    # Establish a connection to the database

    C = "postgres"

    conn = psycopg2.connect(dbname=C, user=user, password=pwd, host="localhost")

    # Create a cursor object

    cur = conn.cursor()

    try:

        del\_plan\_data(cur, year, quarter)

    except Exception as e:

        conn.commit()

        print(f"del\_plan\_data failed: {e}")

        return

    try:

        del\_plan\_status(cur, year, quarter)

    except Exception as e:

        conn.commit()

        print(f"del\_plan\_status failed: {e}")

        return

    try:

        insert\_plan\_dstatus(cur, year, quarter, user)

    except Exception as e:

        print(f"insert\_plan\_dstatus failed: {e}")

        return

    try:

        insert\_plan\_data(cur, year, quarter)

    except Exception as e:

        print(f"insert\_plan\_data failed: {e}")

        return

    try:

        # Commit the transaction

        conn.commit()

        print("Planning data successfully initialized.")

    except Exception as e:

        print(f"An error occurred: {e}")

        conn.rollback()

    finally:

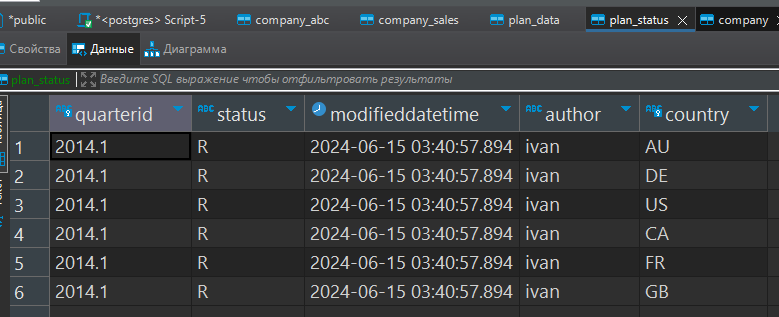
        # Close the cursor and connection

        cur.close()

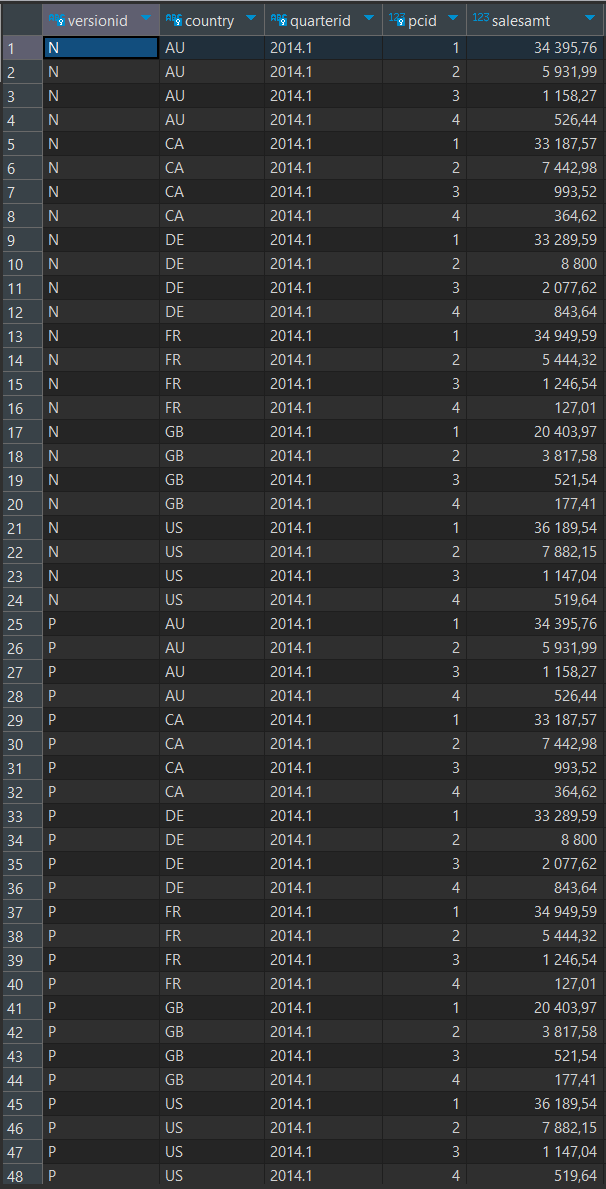
        conn.close()

start\_planning(2014, 1, "ivan", "password\_for\_ivan")

**Output:** Planning data successfully initialized.



*plan\_status*



*plan\_data*

Task №7. Changing 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.

Task №8. Plan data approval

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

I will choose option 2, calculating actual data using salesorderheader and ordersalesdetail tables without using company\_sales. This approach allows us to directly utilize the detailed sales order information to compute the actual sales for A- and B-class companies in 2014, based on their classification in 2013.

**CREATE** **MATERIALIZED** **VIEW** mv\_plan\_fact\_2014\_q1 **AS**

**SELECT**

p.quarterid,

p.country,

p.category\_name,

**COALESCE**(p.plan\_sales, 0) - **COALESCE**(a.actual\_sales, 0) **AS** dev,

**CASE**

**WHEN** p.plan\_sales **IS** **NULL** **THEN** **NULL**

**ELSE** ((**COALESCE**(p.plan\_sales, 0) - **COALESCE**(a.actual\_sales, 0)) / **COALESCE**(p.plan\_sales, 0)) \* 100

**END** **AS** dev\_percent

**FROM**

(**SELECT**

**'2014.1'**::**text** **AS** quarterid,

company.countrycode **AS** country,

product2.pcname **AS** category\_name,

**SUM**(plan\_data.salesamt) **AS** plan\_sales

**FROM**

plan\_data

**JOIN**

company **ON** plan\_data.country = company.countrycode

**JOIN**

product2 **ON** plan\_data.pcid = product2.pcid

**WHERE**

plan\_data.versionid = **'A'** **AND**

plan\_data.quarterid = **'2014.1'**

**GROUP** **BY**

company.countrycode, product2.pcname) p

**LEFT** **JOIN**

(**SELECT**

company.countrycode **AS** country,

product2.pcname **AS** category\_name,

**SUM**(salesorderdetail.linetotal) **AS** actual\_sales

**FROM**

salesorderheader

**JOIN**

salesorderdetail **ON** salesorderheader.salesorderid = salesorderdetail.salesorderid

**JOIN**

product2 **ON** salesorderdetail.productid = product2.productid

**JOIN**

company **ON** salesorderheader.customerid = company.id

**JOIN**

company\_sales **ON** company\_sales.cid = company.id

**WHERE**

**EXTRACT**(**YEAR** **FROM** salesorderheader.orderdate) = 2014 **AND**

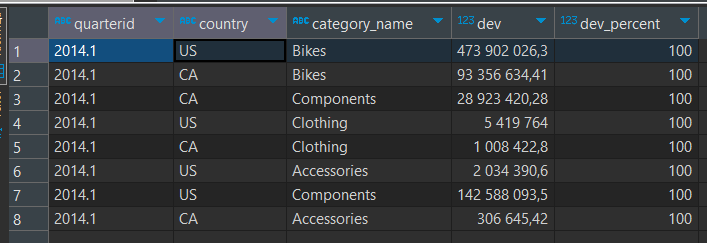
**EXTRACT**(**MONTH** **FROM** salesorderheader.orderdate) **BETWEEN** 1 **AND** 3 **AND**

company\_sales.**year** = 2013 **AND** company\_sales.ccls **IN** (**'A'**, **'B'**)

**GROUP** **BY**

company.countrycode, product2.pcname) a

**ON** p.country = a.country **AND** p.category\_name = a.category\_name;



*mv\_plan\_fact\_2014\_q1*