Sales forecasting and planning

Data management

Working with relational databases

14 points

Contents

1. Building an IT-solution for plan vs. actual analysis of bike production company

1.1. Description

The main learning objective of this assignment is to apply knowledge of SQL and foundations of relational databases to create a tool for sales planning and analysis.

You have been given access to a database of a company that sells bikes and related products. The database contains information about orders, products, and customers. Company works with customers of two types. The first group includes individuals and the second contains shops that resell goods bought from the company.

To evaluate the performance of sales department you need to (1) create tools to automate the planning process and (2) prepare data to calculate difference between planned and actual values of amount sold to companies (resellers).

Data manipulations and plan management (like creating new plan periods, data approval or changing status of plan) should be done with SQL.

This assignment includes the following activities:

- Writing queries to process and copy data
- Creating functions in the database
- Creating materialized views
- Setting permissions with SQL

A more detailed explanation of planning process is presented in the sections below. The next chapter covers users of a planning system and their role in the planning process.

1.2. Users of the planning system

There are two user groups working with plan data: administrators and managers.

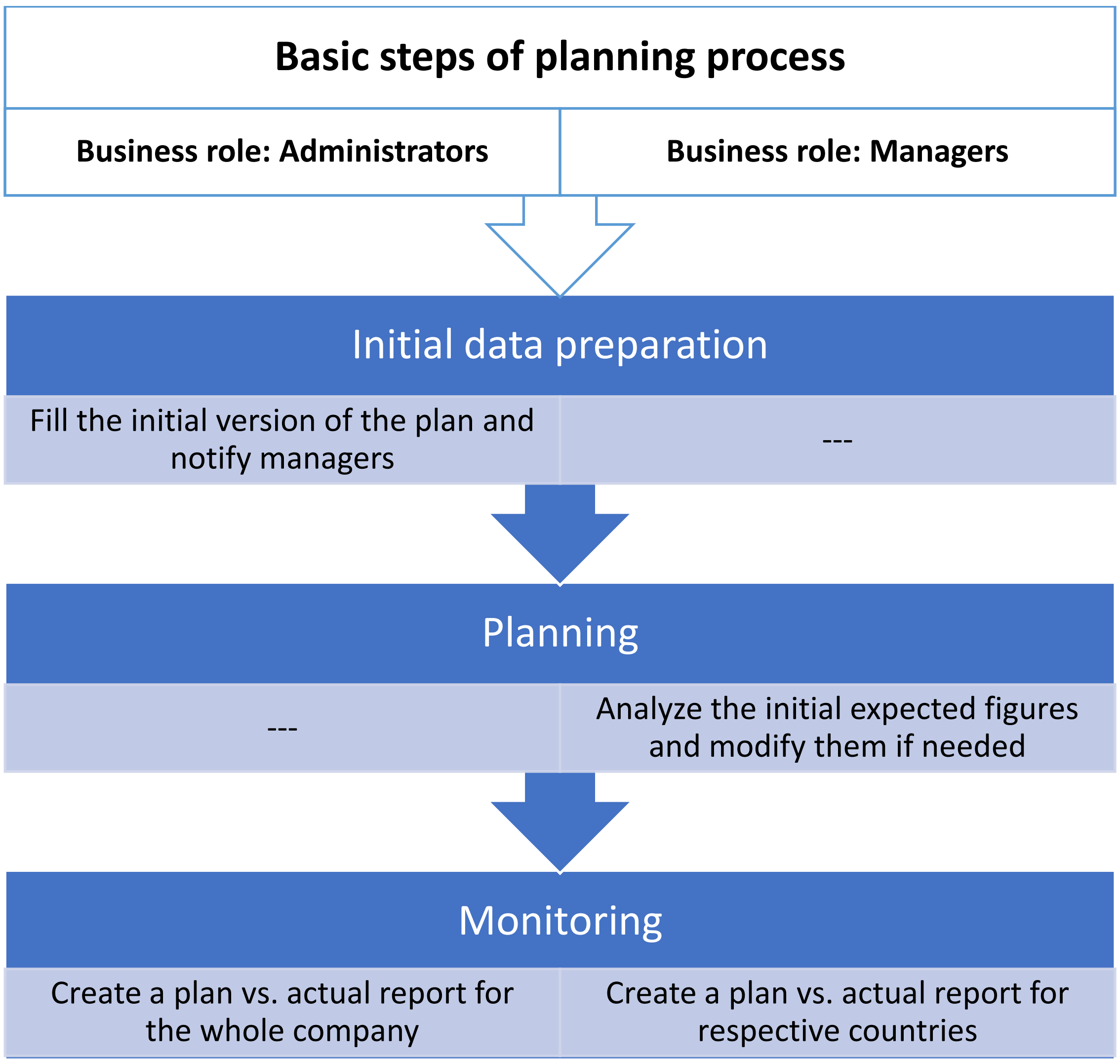
Administrators prepare initial plan data which then becomes available to managers. One planning period contains only one quarter. Administrators have access to all data unlike managers.

Managers plan sales in several countries. Mapping between countries and managers is stored in the system’s settings (in DB). Once the administrator has prepared the initial version of sales plan, managers will be able to modify and confirm the calculated figures. The initial sales plan is based on actual data in two previous years.

Managers can lock their data to avoid unintended corrections by other users. Only locked by manager data is available for correction.

At the end of the planning process managers confirm correctness of plan figures. The approved plan is then used to prepare a plan vs. actual report.

Plan vs. actual comparison report is available to both managers and administrators.


Basic steps of planning process
Business role: Administrators
Business role: Managers
Initial data preparation
Fill the initial version of the plan and notify managers

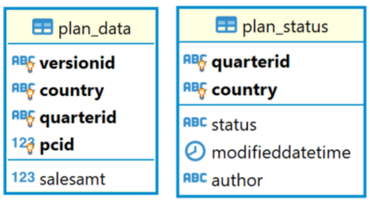
Planning

Analyze the initial expected figures and modify them if needed
Monitoring
Create a plan vs. actual report for the whole company
Create a plan vs. actual report for respective countries

Figure 1 High-level process overview

1.3. Implementation of the planning process in a database

Additional tables were designed to store planning data as well as information on its status on a given planning period.



Plan data has three versions stored in a single table - N (initial calculation), P (edited) and A (confirmed and considered as the result of planning). Version number is written in the *versionid* field.

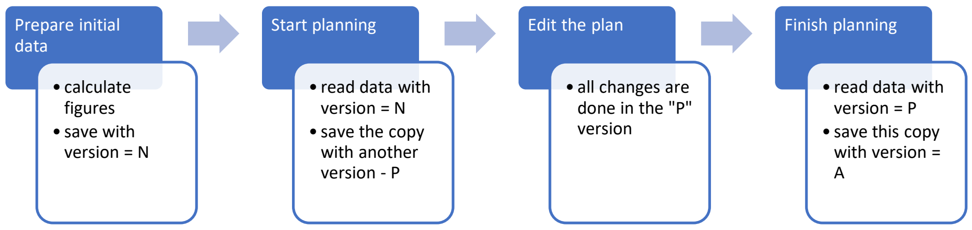

Prepare initial data
calculate figures
save with version = N
Start planning
read data with version = N
save the copy with another version - P
Edit the plan
all changes are done in the "P" version
Finish planning
read data with version = P
save this copy with version = A

Figure 2 Working with data versions

Aggregated sales data is divided into groups of rows (slices) by quarter and country. Quarter is stored in YYYY.Q format.

The status of each data slice can be set individually. The valid status names are:

- R – ready for work
- L – locked by manager and available for editing
- A – approved

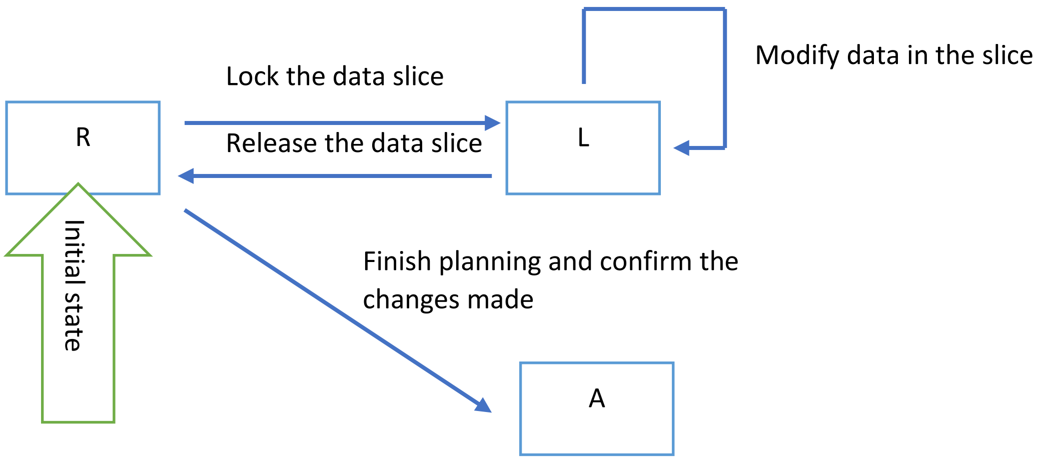


Figure 3 State chart for a data slice

This list contains an example of manipulations with plan data:

- The planning period (quarter and year) and a list of countries are chosen.
- The administrator creates records in *plan_status* table for the current planning period with “R” status. Number of rows to be added depends on countries which participate in planning procedure. One new record should be created for each country.
- Initial plan calculation is performed, and the results are saved as “N” version. Version “N” is read-only.
- Plan data is copied from ‘N’ to ‘P’ version.
- Managers change data in the “P” version. The respective data slice should be locked prior to editing. If the data was locked by another person before, then editing is impossible.

- Managers release locks on their data. Removing data locks means changing plan status from ‘L’ to ‘R’. The lock can be removed only if the slice’s status is ‘L’ and the author of last changes is the current user.
- Then the data is copied to the ‘A’ version by managers which means that planning is complete. The respective planning status records of data slices are changed from ‘R’ to ‘A’. Data in the ‘A’ version is then used to prepare a report on differences between planned and actual sales amounts.

1.4. Calculating the initial planning values

To evaluate the expected volume of sales to shops, you need to use actual sales to companies, that make the most of total yearly sales.

Company (shop) grouping is conducted with help of ABC-analysis based on sales data of each year. Only sales to customers from A and B groups will be used to prepare the initial plan data.

The planning year is further denoted as ***y***.

To determine the initial expected sales amount, it is necessary to calculate the quarterly sales amounts in ***y - 1*** and ***y - 2*** years (excluding customers in the group C).

$amount_{yi,q,c,pc,comp}$ - actual sales amount related to year *yi* (or *y*), quarter number *q*, product category *pc*, sold to country *c* and company *comp*.

$$amount1_{yi,q,c,pc} = \sum_{comp \in \{Aclass_{yi}, Bclass_{yi}\}} amount_{yi,q,c,pc,comp}$$

$Aclass_{yi}$ contains companies that belong to A class in *yi* year.

$N_{yi,q,c,pc}$ is number of positive values in $\{amount1_{yi-1,q,c,pc}; amount1_{yi-2,q,c,pc}\}$.

$$amount2_{y,q,c,pc} = (amount1_{y-1,q,c,pc} + amount1_{y-2,q,c,pc})/N_{y,q,c,pc}$$

amount2 should be calculated for all combinations of country and product category. The two parameters *y* and *q* are year and quarter of planning period.

The administrator saves the initial data as ‘N’ version of the plan. The same data is copied to the ‘P’ version which is supposed to be edited by the managers.

Example: calculations for the first quarter of 2020 will require data from the first quarter of 2018 and the first quarter of 2019.

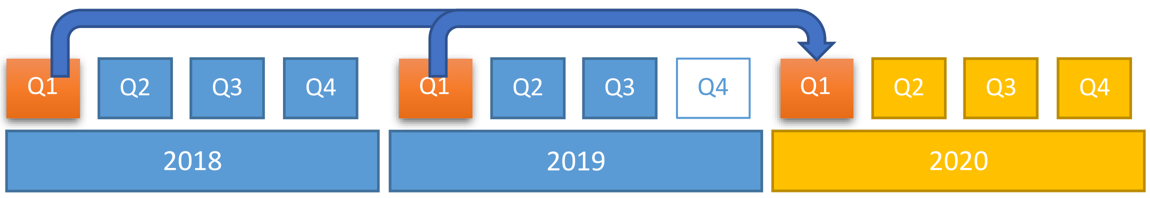


Figure 4 Quarterly planning based on two previous years (2020 is a year to be planned. 2019.Q4 data is unknown)

1.5. Company classification by annual orders

The analysis is done annually. In this task the planning year is 2014.

It is necessary to split the buyer companies into three groups by their volume of orders in ***y* - 1** (2013) and ***y* - 2** (2012) years.

Company can belong to different classes depending on year.

The following algorithm should be used to define members of the three groups for one year:

1. Calculate the annual value of sales amount. This value is denoted as S_{yi}
2. Calculate the upper boundary of groups A and B: $SA_{yi} = 0.8 * S_{yi}$, $SB_{yi} = 0.95 * S_{yi}$
3. Sort companies in descending order by their total sales ($ST_{yi, comp}$).
4. Apply running total calculation to $ST_{yi, comp}$ in the result set of the previous step. The new column is denoted as $SRT_{yi,comp}$. $SRT_{yi,comp}$ of a single company in this case equals the amount of orders of this company *comp* plus sales of all companies that are higher in this list
5. Rate the company *comp* according to the following rules:
    a. If $SRT_{yi,comp} <= SA_{yi}$, then the company is marked as A
    b. If $SRT_{yi,comp} <= SB_{yi}$ and $SRT_{yi,comp} > SA_{yi}$, then the company is marked as B
    c. Otherwise, the company belongs to C class.

1.6. Comparison of planned and actual sales

As soon as a manager has finished editing the plan and confirmed it, the data becomes available for plan vs. actual analysis.

The plan vs. actual analysis report uses plan values that are marked as approved by the manager.

Comparison of planned and actual values is made in terms of year, quarter, country, and product category. Actual figures are taken from the information about sales in the ***y*** year to shops that were members of groups A or B in the previous ***(y – 1)*** year.

If report date the first quarter of 2020, then the actual data will be selected from the 1st quarter of 2020 for companies that were in groups A or B as of 2019.

A report template is shown below:

Table 1

Field	**Quarter**	**Country**	**Category name**	**Dev.**	**Dev., %**
Number	1	2	3	4	5
Description	Quarter key in YYYY.Q format	Code of country	Category name	=Plan – Actual Plan is chosen from the A version of plan data. Actual is taken from fact data (linetotal)	=(Plan-Actual)/Plan

Data examples	2014.1, 2014.2, ...	AU, DE, …	Bikes,…	10000, -3000,…	10%, -30%, …

Table 5 Plan vs. actual comparison report

Important. If the approved plan data cannot be found, then null should be shown in columns 4 and 5.

2. About this task

2.1. Instructions

While preparing this assignment you will modify access settings and write SQL code to manage plan data and create additional database objects. Then you will develop several functions to automate activities related to manipulations with plan data.

The code created will be used to prepare a plan for the first quarter of 2014. The last step includes comparing planned and actual sales figures.

2.2. How to submit

To get mark for this assignment you need to prepare two files and upload them to the platform.

1. Create a report in docx or pdf format. Tasks include hints on what should be put in the report.

For example:

Add 'Confirmation of Plan Data' to the report. Insert the function code into this chapter.

1.3. How this assignment will be graded?

This assignment has ten tasks. Instructions begin on page 12.

First, the completeness of the report is considered. To get a high score, all results marked with *color and italics* must be present in the report and well formed.

The backup should include all the results mentioned in the report. Code from the report should align with requirements written in this document.

Develop functions and queries that are compatible with the database.

You get 1 point for tasks 1, 2, 3, 5, 7, 8 and 2 points for 4, 6, 9.

Task	Grading rules	Points
1	0.6 – sufficient permissions are granted	1

	0.4 – appropriate lines are added into settings table, users are created	
2	0.4 – ‘product2’ is correct 0.3 – ‘country2’ is correct 0.3 – ‘permissions are set’	1
3	1 - query is correct (0.5 – if the query can be executed but is incorrect)	1
4	0 – if query does not work, else: 1 – granularity is correct 0.5 – classification is correct 0.2 – single query 0.3 – screenshot is present	2
5	0 – if query does not work, else: 0.5 – correct calculations in expressions 0.5 – classification table is used as required	1
6	0.4 – function clears data 0.5 – function correctly sets statuses 0.5 – function loads necessary data into ‘N’ version of plan 0.4 – function copies data into ‘P’ version of plan 0.2 – screenshot contains correct data	2
7	0.3 – set_lock function changes status correctly 0.3 – remove_lock function changes status correctly 0.2 – functions refresh attributes of status 0.2 – screenshot is present	1
8	0.2 – the function clears data 0.4 – the function changes status correctly 0.4 – the function creates the ‘A’ version	1
9	1 – trigger was developed correctly 1 – excel report meets all requirements	2
10	0.5 – company class is chosen correctly when preparing actual data 0.5 – plan data is connected to actual figures 0.5 – excel report meets all requirements 0.5 – expressions meet the requirements in 1.6.	2

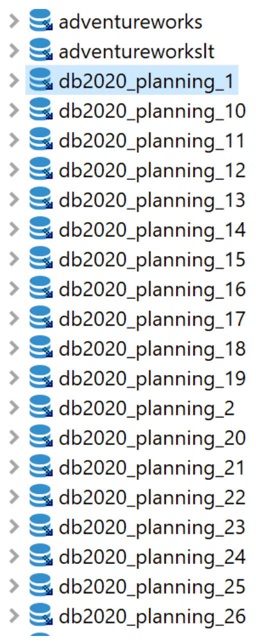
2. Before you begin

You will need a client software to communicate with the server (for example, DBeaver).

The DDL script to create additional tables can be found in the following file (attached to this course):

- Planning application - DDL script.txt

You can find the database for your assignment on the common server. Please, ask teacher assistants for the database name. Typically, it will start with ***db<year>_planning_<your number>***.



adventureworks
adventureworkslt
db2020_planning_1
db2020_planning_10
db2020_planning_11
db2020_planning_12
db2020_planning_13
db2020_planning_14
db2020_planning_15
db2020_planning_16
db2020_planning_17
db2020_planning_18
db2020_planning_19
db2020_planning_2
db2020_planning_20
db2020_planning_21
db2020_planning_22
db2020_planning_23
db2020_planning_24
db2020_planning_25
db2020_planning_26

Your database will contain tables as shown on the diagram below (Figure , page 10):

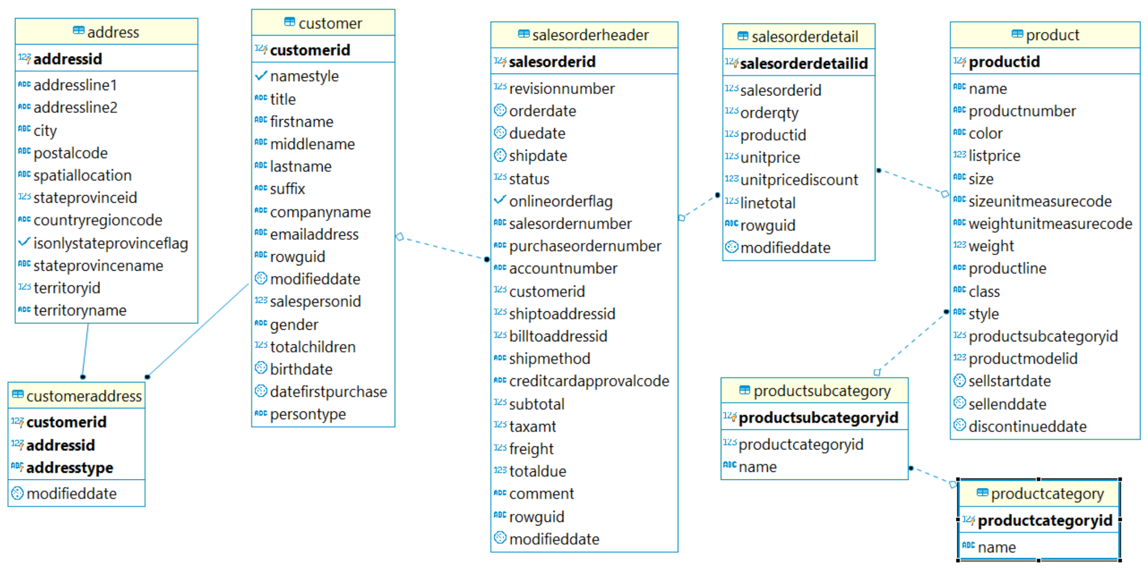


Figure 1 Tables with business data

In addition to the tables in the diagram, you will need tables and views below, which should be added from the corresponding file with DDL. Run code from the file to create the following database objects:

1. Table which stores status of plan data fragments – ‘plan_status’
2. Table of versioned plan data – ‘plan_data’
3. Data access settings defining each manager’s permissions to view and edit sales plans in specific countries – ‘country_managers’
4. Table of companies. Shops that resell products – ‘company’
5. Aggregated sales table in terms of companies, product categories, years, and quarters – ‘fact_sales’
6. Company classification by total annual cost of orders – ‘company_abc’
7. A view to edit planned data by administrators and managers – ‘v_plan_edit’
8. A view to read approved planning data based on user’s permissions – ‘v_plan’
9. Two roles – planadmin### (administrators) and planmanager### (managers).

3. Graded tasks

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

Set up permissions for roles as written in the table below:

Table 2 Planning system users' rights

	User role	
DB object	planadmin	planmanager
all tables	S	S
plan_data	SUID	SUID
plan_status	SUID	SU
country_managers	SUID	S
v_plan_edit	-	SU
v_plan	-	S

Legend:
S – select
U – update
I – insert
D - delete

Create three users:

- Administrator:
    - ivan###
- Managers:
    - sophie###
    - kirill###

### is your number.

Manager sophie### has access to data of US and CA countries. Manager kirill### works with sales data in FR, GB, DE, AU countries. Put this information in the 'country_managers' table, which stores data about the anchoring of managers for certain countries.

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.

4.2. Task №2. Creating product and country views

Add two materialized views – 'product2', 'country2'.

The 'product2' view should contain the product and its category.

The 'country2' view should be filled with unique codes of the countries where the shops are located (the type of address is Main Office).

Allow managers and administrators to read from these views.

Fields of ‘product2’ are shown in the table:

Table 3

Field	**Description**	**Rule**
Pcid	Product category key	Load from Productcategory.productcategoryid
productid	Product key	Load from Product.productid
pcname	Category name	Load from Productcategory.name
pname	Product name	Load from Product.name

Fields of 'country2' are shown in the table:

Table 4

Field	**Description**	**Rule**
countrycode	Code of country	Load unique values from address.countreyregioncode

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.

Tables with source data

Diagrams of needed tables are presented below.

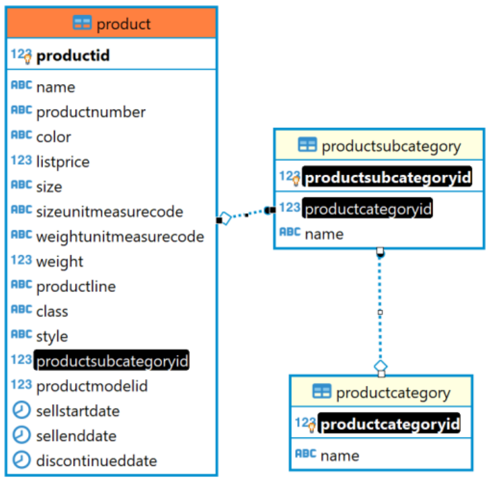


Figure 6 Tables for product2

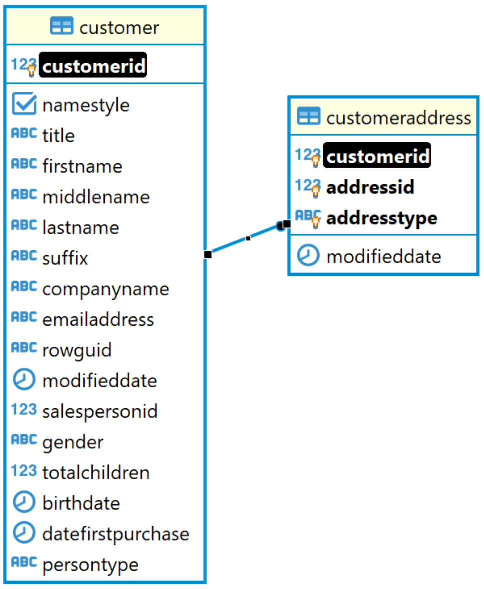


Figure 3 Tables for country2

4.3. Task №3. Loading data into the company table

In the current database the *customer* table contains information about two categories of buyers - individuals and companies. However, we consider only companies in this task. For the convenience of further development, fill the company table with data.

Values of *companyname* field should be included in the list of companies. The country and the city should be taken from the *address* table. Develop a single query to load the country table.

Follow these rules in the table below. Use addresses of ‘Main Office’ type to find the appropriate information.

Table 5 ‘company’ table fields

Field	**Description**	**Rule**
id	Company key	Auto-generated value. No need to populate it manually
cname	Company name	Company name ‘customer.companyname’
countrycode	Code of country	Head office country ‘address.countryregioncode’
city	City name	Head office city ‘address.city’

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

Tables with source data

The following tables can be used to form the ‘company’ table: customer, customeraddress, address. You can find them on the diagram below.

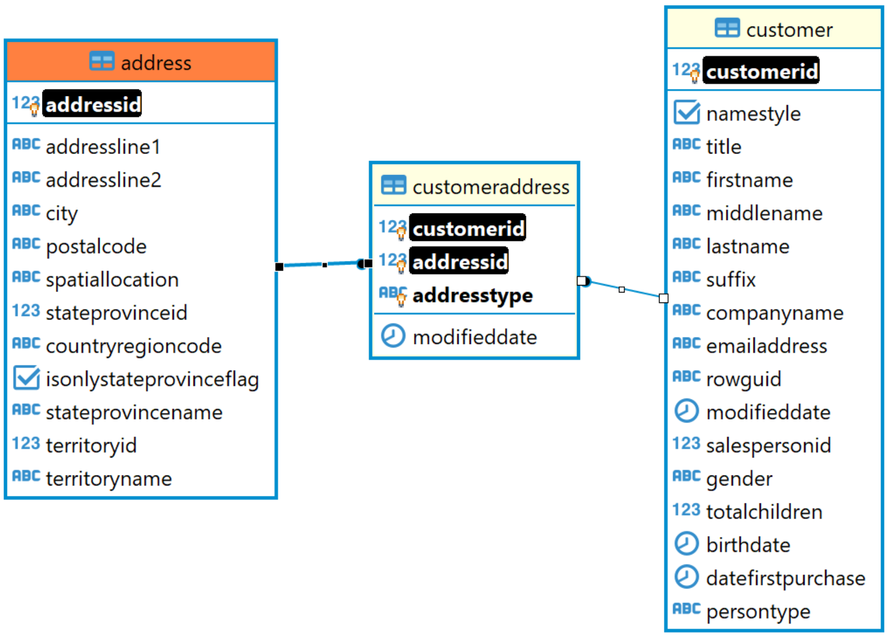


Figure 4 Source data to fill ‘company’ table

4.4. Task №4. Company classification by annual amount of orders

Split the companies into three groups according to algorithm in section 1.5 (on page 7) for 2012 and 2013.

Fill the ‘company_abc’ table using SQL. All calculations should be done in one query.

Table 6 company_abc’ attribute loading rules

Field	**Description**	**Rule**
cid	Company key	Key from ‘company’ table. Find the *companyname* from ‘*customer*’ in the ‘company’ table to retrieve its key
salestotal	Total sales per year	Calculated as *sum(subtotal)* from *‘salesoderheader’* table
cls	Company class	Use the algorithm from section 1.5.
year	Year	Get year from *orderdate* field of *‘salesoderheader’* table

Add the designed SQL queries to the report under ‘Task №4. Company classification’ heading. Add a screenshot of 10-20 records of company_abc.

Tables with source data

Use the following tables:

salesorderheader, customer, company

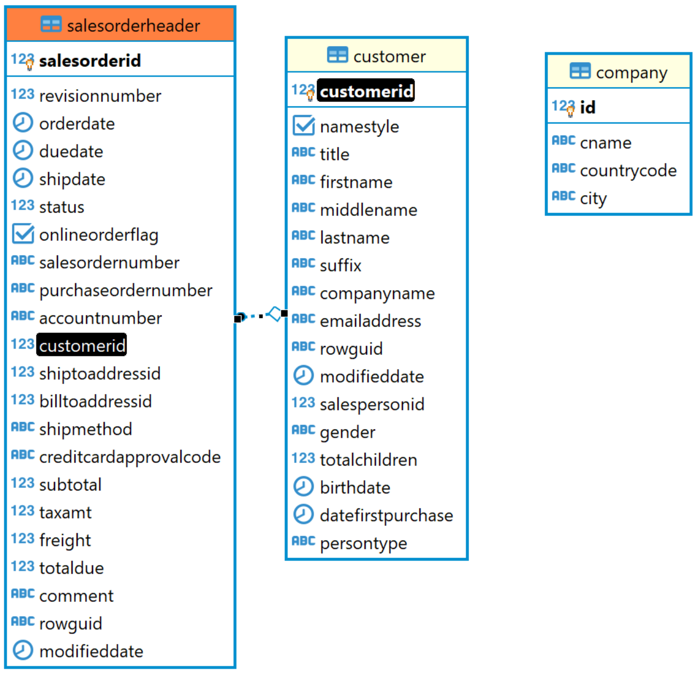


Figure 5 Tables for company segmentation

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

Calculate quarterly sales amount before taxes in 2012 and 2013 individually. Fill the *company_sales* table using data about orders, companies, and classification results in the respective year.

The table below contains comments on filling the fields of the *company_sales* table.

Table 7 Data loading rules for company_sales table

Field	**Description**	**Rules**
cid	Company key	*company.id* Find company’s identifier through *customer* and *company* tables
salesamt	Total amount sold (before taxes)	Calculate as sum(*salesorderdetail.linetotal*) for combinations of year&quarter, company and category. The *salesorderdetail*

		table contains product sales data in quantity and cash.
year	Year	Numeric year of *salesorderheader.orderdate*
quarter_yr	Quarter of year	Numeric quarter number of *salesorderheader.orderdate*
qr	Quarter	String representation of quarter in "YYYY.Q" format extracted from *salesorderheader.orderdate.*
categoryid	Product category's key	Use *product2.pcid.* Connect *product2* with *salesorderdetail*
cls	Company's class code	String name of a class to which a company belongs. Use *company_abc.cls* for the year from *salesorderheader.orderdate*

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

Source data

Use the following tables and views to complete the task:

Customer, company, company_abc, salesorderheader, salesorderdetail, product2

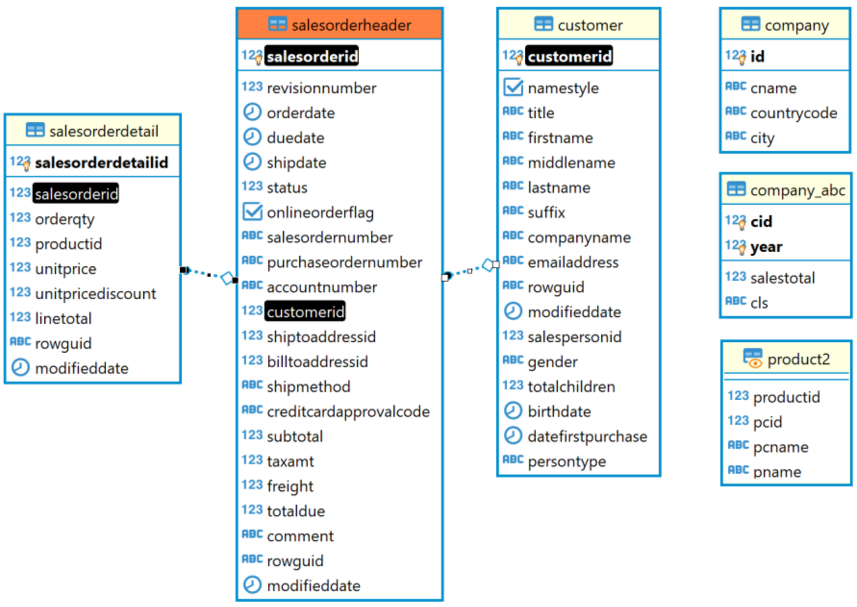


Figure 6 Source data location in the database

4.6. Writing functions to automate planning process

Now it's time to write functions. You can use the same file to develop functions and execute them.

4.6.1. Task №6. Generating the initial planning data

Write a function in PostgreSQL to help managers start working with plan data in new quarter. Each planning period can be identified by two values – year and quarter. E.g., 2021.1 (first quarter of 2021) or 2022.4 (fourth quarter of 2022).

Parameters of the *start_planning(year, quarter)* function are:

- year – target year of planning period
- quarter – target quarter of the planning period

These parameters have the same meaning in the next tasks.

The function should implement the following steps:

1. Delete plan data from the *plan_data* table related to the planning year and quarter.
2. In the plan_status table delete records related to the target quarter
3. Create planning status records (*plan_status* table) for the selected quarter. The number of records added equals the number of countries in which companies are situated.
4. Generate version 'N' of planning data in the *plan_data* table. Use the calculation algorithm described in section 1.4.
5. Copy data from version 'N' into version 'P' in the *plan_data* table.

Name of the current user who called the function should be stored in the *author* column of records of *plan_status* table.

If initial planning data cannot be generated for some record in *plan_status* (e.g., if no data can be found in the *company_sales* table), then add rows with 0 in *salesamt* column for related combination of country and category.

If the function works correctly, the data will appear in the *plan_data* and *plan_status* tables. The 'N' and 'P' versions of plan data will be created.

Rules for loading data into the *plan_status* tables are shown below. Plan_status holds information about all plan data slices, which are restricted by quarter and year and country:

Table 8

Column	**Description**	**Rules**
quarterid	Key of planning quarter	String identifier of a quarter with "YYYY.Q" value template
country	Country of a shop which orders goods	company.country
status	Planning data slice status	'R'
modifieddatetime	Time when the record was changed or created	Value of current_timestamp
author	User that changed the record	Value of current_user

Rules for loading data into the *plan_data* tables are shown here:

Table 9

Column	**Description**	**Rules**
versionid	Version of plan data	Initial data generation – 'N';

		Copy from ‘N’ version – ‘P’;
country	Country of a shop which orders goods	country2.countrycode
quarterid	Key of planning quarter	String identifier of a quarter with “YYYY.Q” value template
pcid	Product category’s key	product2.pcid
salesamt	Sales amount before taxes	Average of total quarterly sales amount in A and B groups. Source field - company_sales.salesamt

Call this function on behalf of ivan##. The target planning period is 1st quarter of 2014.

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’).

Source data

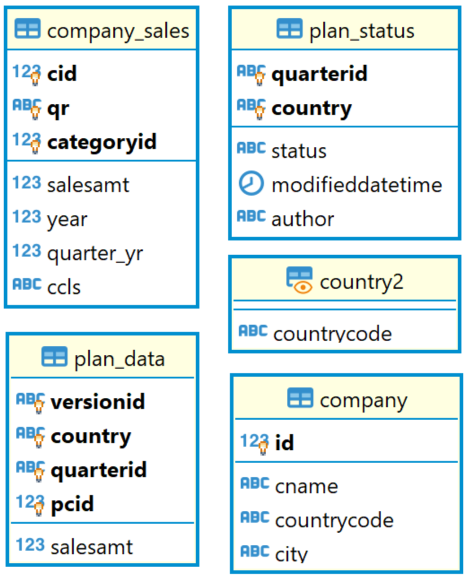


Figure 7 Tables with source data

4.6.2. Task №7. Changing the plan data

Write two functions in PostgreSQL:

- *set_lock(year, quarter)*, which will change status from ‘R’ to ‘L’ for data slices, that are associated with the target quarter and year, and connected to the current user in the *country_managers* configuration table. To obtain the name of the current user, use current_user in your query. Also write a timestamp of modification to the modifieddatetime field.

- *remove_lock(year, quarter)* function, that will change the planning data status from ‘L’ to ‘R’. associated with the current user through the *country_managers* table. Write a change time stamp in the *modifieddatetime* field.

Execute the *set_lock* function to lock the plan data in the first quarter of 2014 on behalf of “kirill###” user, and then – “sophie###”. If everything is done correctly, data will appear in the *v_plan_edit* view if logged on as kirill### or sophie###.

Increase planned sales by about 30-50% in the *v_plan_edit* view on behalf of two managers. You can edit data through the view in DBeaver using a virtual key (it must contain all fields except *salesamt*).

Run the function *remove_lock* to mark Q1 2014 as free. Run this function as “kirill###” and then as “sophie###”. Now the *v_plan_edit* view will return no records.

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.

Rules for updating the *plan_status* table are listed below:

Table 10 Update rules for plan_status

Column	Description	Rules
quarterid	Key of planning quarter	No changes
sountry	Country of a shop which orders goods	No changes
status	Plan data slice status	‘L’
modifieddatetime	Time when the record was changed or created	Value of current_timestamp
author	User that changed the record	Value of current_user

Tables used

Tables are shown on the picture below. *Plan_status* table should be updated with SQL. *Country_managers* table stores information about users’ permissions regarding plan sales to different countries.

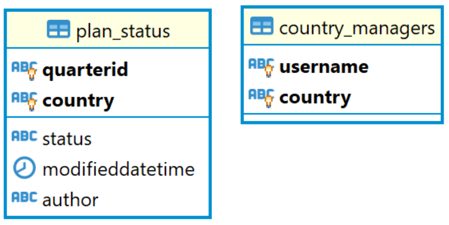


Figure 8 Tables to use inside functions that manage planning status

4.6.3. Task №8. Plan data confirmation

Write an SQL function - accept_plan(year, quarter, user, pwd). The function is designed to copy the corrected data into the actual version of the plan.

The function should select records from plan_data table that meet the following requirements:

- Planning quarter (*quarterid* column) is equal to combination of year and quarter from the function's arguments.
- Version is 'P' (corrected version of the plan).
- Data slice status (in *plan_status* table) equals 'R'.
- The current user has a permission to access the plan data according to the settings in the *country_managers* table.

  Implement these processing steps in the *accept_plan* function:
- Clear the ‘A’ version of plan data for specific quarter and countries accessible to the current user
- Read data available to the current user from the version ‘P’ and save its copy into as the version ‘A’
- Change the status of data being processed from 'R' to 'A'

When updating the status, also save a time stamp in *modifiedtimestamp* column. Use the developed function to approve the plan of Q1 2014 on behalf of each manager. Check whether the data is visible through the *v_plan* view:

- The administrator has access to the entire plan
- Manager can view only data he/she is permitted to read and change.

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.

The next two tables represent comments on how to update plan tables.

Table 11 Rules for changing records in the plan_status table

Field	**Description**	**Rules**
quarterid	Key of planning quarter	No Changes
country	Country of a shop which orders goods	No Changes
status	Planning data slice status	‘A’
modifieddatetime	Time when the record was changed or created	Value of current_timestamp
author	User that changed the record	Value of current_user

Table 12 Rules for loading data in the plan_data table

Field	**Description**	**Rules**
versionid	Version of plan	‘A’
country	Country of a shop which orders goods	Copy from version ‘P’
quarterid	Key of planning quarter	Copy from version ‘P’
pcid	Product category’s key	Copy from version ‘P’
salesamt	Sales amount before taxes	Copy from version ‘P’

Tables used

Tables are shown on the picture below:

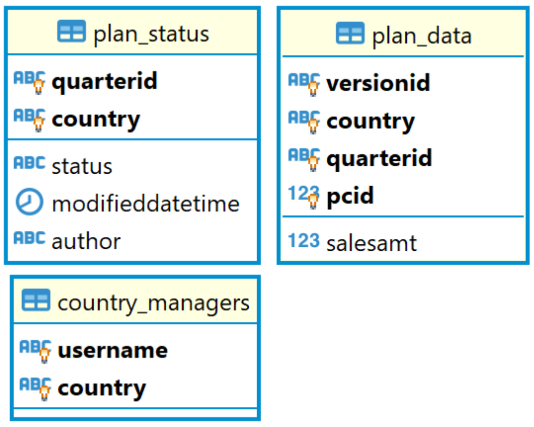


Figure 9 Tables for accept_plan

4.7. Task №9. Tracking changes in the plan table with trigger

Write a trigger to track all changes in the 'P' version. If some figure was modified, then previous and new values should be written in a separate table along with user's name and timestamp.

Please, create this table to store changes – plan_data_history

Add the following fields:

- country – country code char(5)
- quarterid – planning period char(6)
- pcid – category key int
- author – user who changed sales amount figure varchar(25)
- modified_time – timestamp when a figure was changed timestamp
- salesamt – new value numeric(18, 2)
- salesamt_old – previous value numeric(18, 2)

Primary key of this table will contain five columns – {country, quarterid, pcid, modified_time, author}.

Test the trigger.

Connect to the database from MS Excel and create a plain table report showing changes made by users. Use ODBC driver for PostgreSQL. The report should display the following information:

- user
- time of modification (sort the table in reverse chronological order)
- difference between previous and new sales amount
- category name
- country code
  Enable user input control to filter the table by category and user names. Data slicers in MS Excel provide the required functionality.

Add a header "Tracking changes in plan table". Paste the code of the trigger in your document. Take a screenshot of Excel report.

4.8. Task №10. Data preparation for plan vs. actual analysis in Q1 2014

Create a materialized view mv_plan_fact_2014_q1 to compare planned and observed sales before taxes in 1st quarter of 2014. The view itself should show the difference between actual figures and plan.

The report requirements are described in the section 1.6. on the page 7.

Use the results of ABC-analysis from 2013 to find actual sales of A- and B-class companies in 2014.

You can choose one of these approaches (does not affect your grade):

1. Load data of 2014 into the company_sales table and include this table in the view

2. Calculate actual data using salesorderheader and ordersalesdetail tables without using company_sales.

Connect to the database from MS Excel and create a cross-table report showing planned and actual sales amounts with categories and countries. Use ODBC driver for PostgreSQL.

Add a header "Data preparation for plan-fact analysis in Q1 2014". Write which approach you have chosen. Include SQL code of the new materialized view together with a screenshot showing data in mv_plan_fact_2014_q1 view.

Take a screenshot of the Excel report.

Tables used

Here you can find all tables and views that can serve as a data source for the considered report.

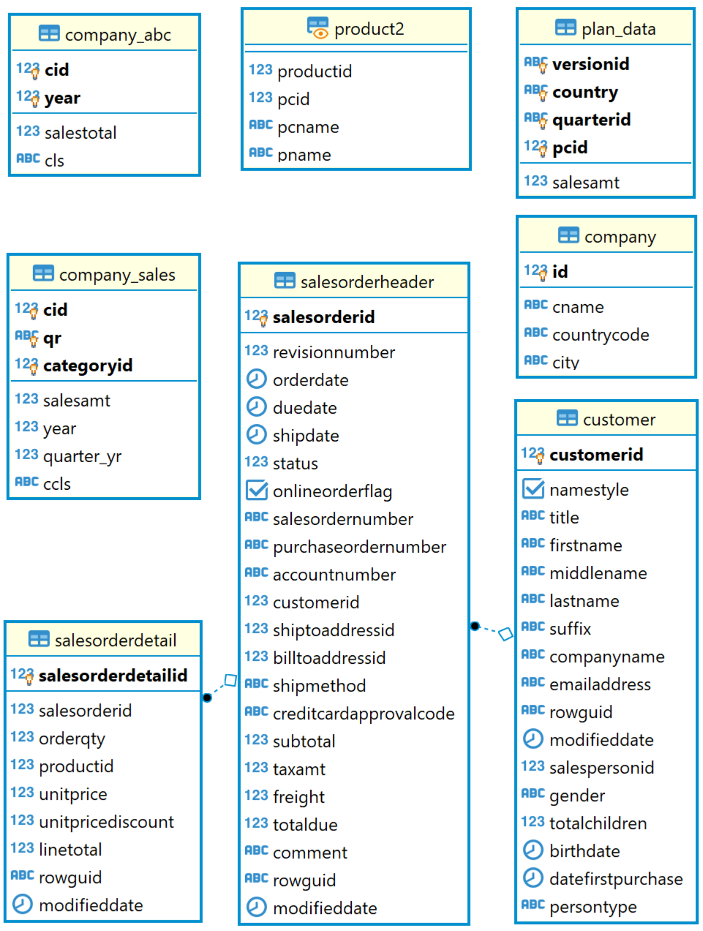


Figure 10 Tables and views available for plan-fact analysis

Database submission

Make sure your database exists on the common server.

Created database should meet the requirements presented in your report – contain necessary tables with data and views.

SQL queries provided in the report are expected to run successfully with your database. Functions should be contained in the database.