МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра вычислительной техники

**Курсовая работа**

**по дисциплине «Базы данных»**

**на тему «Проектирование базы данных «Климатическое оборудование»**

|  |  |
| --- | --- |
| Студент | Дунаев Н. Ю. |
| Группа | АВТ-610 |
| Преподаватель | Трошина Г.В. |
| Вариант | 52 |
|  |  |

Новосибирск 2018

Оглавление

[Задание 3](#_Toc532662005)

[Графическое представление связей между таблицами 4](#_Toc532662006)

[Структуры таблиц баз данных](#_Toc532662007)

[Содержание таблиц баз данных 5](#_Toc532662008)

[Руководство пользователя 7](#_Toc532662009)

[Тексты запросов и примеры работы 10](#_Toc532662010)

[Анализ результатов и выводы 22](#_Toc532662011)

[Список использованной литературы 22](#_Toc532662012)

# Задание

Основная база данных «Климатическое оборудование» (далее КО). Сформировать несколько таблиц. Предусмотреть: ввод данных, редактирование, просмотр данных. Реализовать следующие типы запросов с использованием нескольких таблиц:

1. Для каждого вида КО указать сведения о нем (наименование, дату выпуска, поставщик, цена, дата продажи и т.п.).
2. Для каждого КО выдать список, отсортированный:
   1. По дате выпуска,
   2. В алфавитном порядке по поставщику,
   3. По стоимости,
   4. По дате продажи.
3. Найти самое дорогое КО, самое дешевое, среднюю стоимость по каждому виду и в целом.
4. Найти КО с ценой в заданных пределах (предусмотреть ввод цены с клавиатуры) по каждому виду и в целом.
5. Найти все КО заданного производителя (выбор производителя).
6. Найти долю дешевого КО (меньше заданного, ввод ограничения) от общего числа КО (для каждого вида и в целом).
7. Найти все КО с заданной датой выпуска (ввод даты).
8. Найти все КО, чья дата продажи находится в заданных пределах (ввод интервала) для заданного производителя (выбор производителя) и в целом.
9. Найти долю КО, проданного за определенный период времени (ввод периода) от общего времени продажи.
10. Найти количество бракованного КО, поступившего из заданной страны (выбор страны) для заданного поставщика (выбор поставщика).
11. Найти все КО, поступившее от заданного поставщика, чья стоимость больше, чем стоимость заданного КО (выбор наименования КО), поступившего из заданной страны (выбор страны).
12. Найти долю дешевого КО (чья стоимость меньше заданной, ввод стоимости), поступившего от заданного поставщика (выбор поставщика) и в целом.
13. Найти среднюю стоимость КО, проданного за определенный промежуток времени (ввод интервала).
14. Найти все КО, чья стоимость выше, чем средняя стоимость КО заданного производителя (выбор производителя).
15. Определить долю регулярных поставок заданного КО (выбор) по заданной дате отгрузки (ввод) от общего числа КО.
16. Найти КО, которое лучше всего раскупают, указать параметры (стоимость, производитель, страна и т.д.).

# Графическое представление связей между таблицами

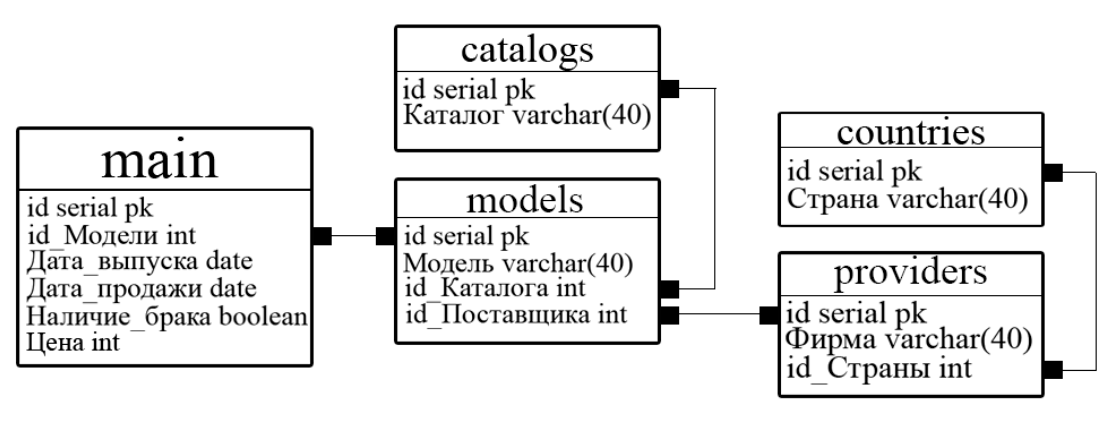


Рисунок 1. Графическое представление связей между таблицами

# Структуры таблиц баз данных

Таблица *main* содержит проданное КО.

* id – идентификационный номер проданного товара
* id\_Модели – идентификационный номер модели товара
* Дата\_выпуска – дата выпуска (или дата поставки для задания 15)
* Дата\_продажи – дата продажи
* Цена – цена, по которой был продан товар
* Наличие\_брака – бракован ли товар

Таблица *models* содержит модели товаров.

* id – идентификационный номер модели
* Модель – название модели
* id\_Поставщика – идентификационный номер фирмы – производителя модели
* id\_Каталога – идентификационный номер каталога, к которому принадлежит модель

Таблица *providers* содержит фирмы (понятие фирмы совмещено с понятиями поставщика и производителя).

* id – идентификационный номер фирмы
* id\_Страны – идентификационный номер страны, к которой относится фирма
* Фирма – название фирмы

Таблица *catalogs* содержит типы оборудования (понятие типа совмещено с понятием вида).

* id – идентификационный номер каталога
* name – название каталога

Таблица *countries* содержит страны.

* id – идентификационный номер страны
* Страна – название страны

# Содержание таблиц баз данных

* Таблица *countries*

CREATE TABLE countries (id SERIAL PRIMARY KEY, Страна varchar(40));

SELECT \* FROM countries;



* Таблица *providers*

CREATE TABLE providers (id SERIAL PRIMARY KEY, Фирма varchar(40), id\_Страны int);

ALTER TABLE providers add constraint countriesfk foreign key (id\_Страны) references countries (id) on update no action on delete cascade;

SELECT \* FROM providers;



* Таблица *catalogs*

CREATE TABLE catalogs (id SERIAL PRIMARY KEY, Каталог varchar(40));

SELECT \* FROM catalogs;

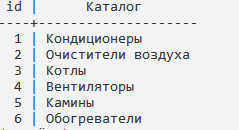


Таблица *models*

CREATE TABLE models (id SERIAL PRIMARY KEY, Модель varchar(40), id\_Каталога int, id\_Поставщика int);

ALTER TABLE models add constraint firmsfk foreign key (id\_Поставщика) references providers (id) match simple on update cascade on delete no action;

ALTER TABLE models add constraint catalogsfk foreign key (id\_Каталога) references catalogs (id) on update no action on delete no action;

SELECT \* FROM models;

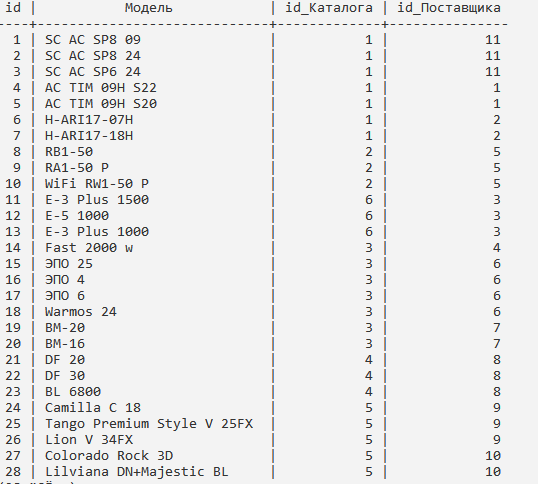


Таблица *main*

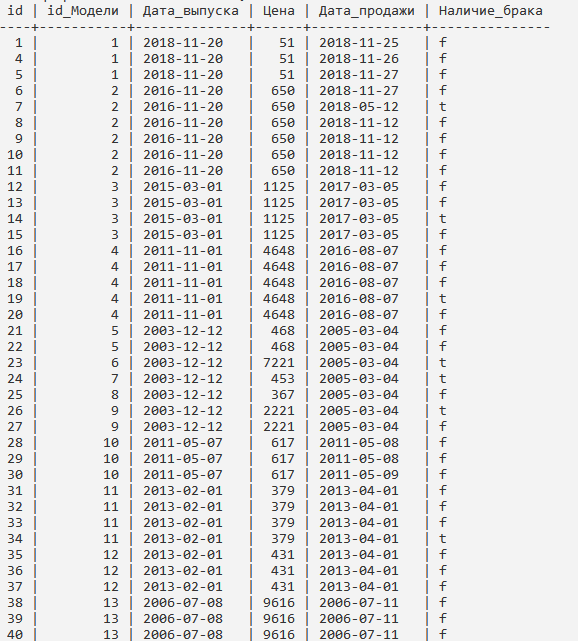
ALTER TABLE main add constraint modelsfk foreign key (id\_Модели) references

models (id) on update no action on delete no action;

ALTER TABLE main add check (Дата\_выпуска < Дата\_продажи);

ALTER TABLE main add check (Цена > 0);

SELECT \* FROM main;



(фрагмент таблицы, всего 76 строк)

# Руководство пользователя

|  |  |  |
| --- | --- | --- |
| Название функции | Аргументы | Описание |
| showEquip() | — | Показать сведения о проданном КО |
| sort\_release\_date() | — | Показать список КО, отсортированный по дате выпуска |
| sort\_provider() | — | Показать список КО, отсортированный по производителю |
| sort\_price() | — | Показать список КО, отсортированный по цене |
| sort\_sale\_date() | — | Показать список КО, отсортированный по дате продажи |
| max\_price() | — | Показать самое дорогое КО |
| min\_price() | — | Показать самое дешевое КО |
| avg\_price() | — | Показать среднюю стоимость по всему КО |
| avg\_price\_in() | — | Показать среднюю стоимость КО по каждому каталогу |
| price\_between\_for\_each\_catalog  (int, int) | Нижний предел цены  Верхний предел цены | Показать список КО в заданных пределах для каждого каталога |
| price\_between(int, int) | Нижний предел цены  Верхний предел цены | Показать список КО в заданных пределах в целом |
| all\_equip\_of\_provider(varchar) | Название фирмы | Показать список КО заданного поставщика |
| low\_part(int) | Значение цены | Показать долю КО с ценой, меньше заданной от общего числа КО |
| low\_part\_in\_catalog(varchar,int) | Название каталога  Значение цены | Показать долю КО с ценой, меньше заданной от общего числа КО по выбранному каталогу |
| find\_by\_release\_date(date) | Дата выпуска | Показать список КО с заданной датой выпуска |
| find\_if\_sale\_date\_between(date, date) | Нижний предел даты продажи  Верхний предел даты продажи | Показать список КО, произведенный в заданном промежутке времени |
| find\_if\_sale\_date\_between\_of\_provider  (date, date, varchar) | Нижний предел даты продажи  Верхний предел даты продажи  Название фирмы | Показать список КО, произведенный в заданном промежутке времени указанным поставщиком |
| sale\_date\_part(date, date) | Нижний предел даты продажи  Верхний предел даты продажи | Показать долю КО, проданного в заданный период времени от общего времени продажи |
| defective(varchar, varchar) | Название страны  Название фирмы | Показать количество бракованного КО, поступившего из заданной страны для заданного поставщика |
| find\_by\_provider\_price\_country  (varchar, int, varchar) | Название фирмы  Цена  Название страны | Показать список КО, поступившего от заданного поставщика больше заданной цеы из заданной страны |
| low\_part\_from\_provider(int, varchar) | Значение цены  Название фирмы | Показать долю КО, стоимость которого меньше заданной, произведенного заданным поставщиком |
| avg\_price\_between(date,date) | Нижний предел даты продажи  Верхний предел даты продажи | Показать среднюю стоимость КО, проданного за определенный период времени |
| price\_more\_avg\_of(varchar) | Имя фирмы | Показать список КО, чья стоимость выше, чем средняя стоимость КО заданного производителя |
| part\_of\_provider\_by\_date  (varchar, date) | Название модели  Дата выпуска | Показать долю регулярных поставок заданного КО по заданной дате отгрузки от общего числа КО. |
| popular\_equip() | — | Показать наиболее часто скупаемое КО |

# Тексты запросов и примеры работы

1) Для каждого вида КО указать сведения о нем (наименование, дату выпуска, поставщик, цена, дата продажи и т.п.).

CREATE OR REPLACE FUNCTION showEquip() RETURNS TABLE

(id int, Поставщик varchar(40), Модель varchar(40), Дата\_выпуска date, Дата\_продажи date, Цена int)

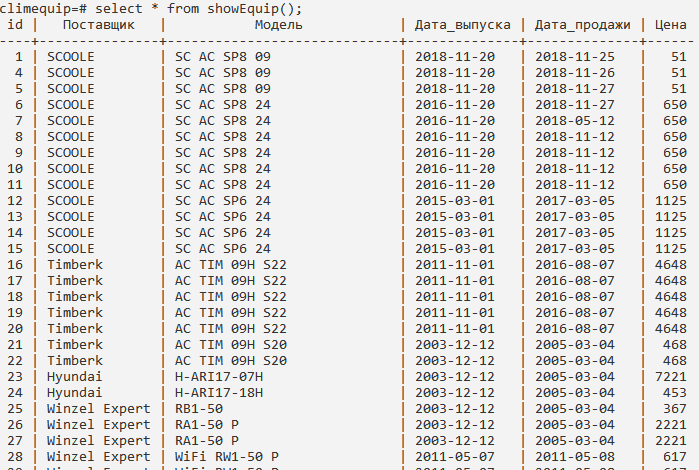
AS $$ BEGIN RETURN QUERY

SELECT main.id, providers.Фирма, models.Модель, main.Дата\_выпуска, main.Дата\_продажи, main.Цена

FROM main, providers, models

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id);

return; end; $$ language 'plpgsql';



2. Для каждого КО выдать список, отсортированный:

2) Для каждого КО выдать список, отсортированный:

2.1) По дате выпуска

CREATE OR REPLACE FUNCTION sort\_release\_date() RETURNS TABLE

(id int, Поставщик varchar(40), Модель varchar(40), Дата\_выпуска date, Дата\_продажи date, Цена int)

AS $$ BEGIN RETURN QUERY

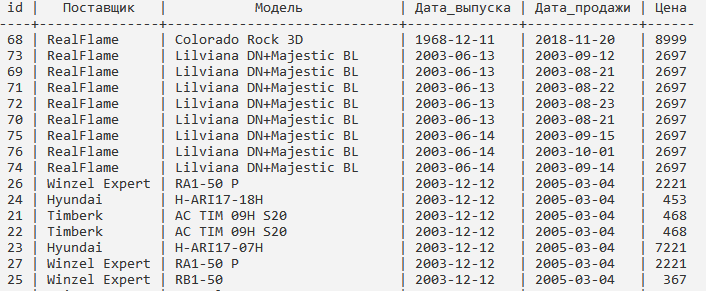
SELECT main.id, providers.Фирма, models.Модель, main.Дата\_выпуска, main.Дата\_продажи, main.Цена

FROM main, providers, models

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id)

ORDER BY Дата\_выпуска;

return; end; $$ language 'plpgsql';



2.2) В алфавитном порядке по поставщику

CREATE OR REPLACE FUNCTION sort\_provider() RETURNS TABLE

(id int, Поставщик varchar(40), Модель varchar(40), Дата\_выпуска date, Дата\_продажи date, Цена int)

AS $$ BEGIN RETURN QUERY

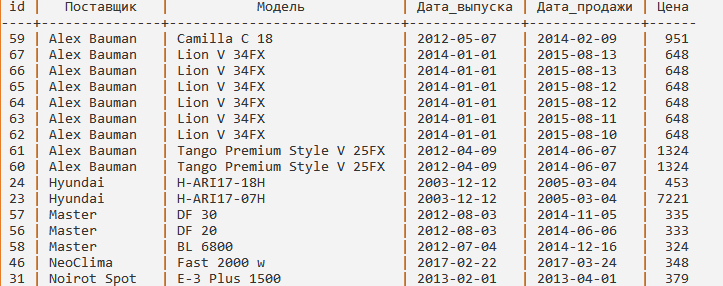
SELECT main.id, providers.Фирма, models.Модель, main.Дата\_выпуска, main.Дата\_продажи, main.Цена

FROM main, providers, models

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id)

ORDER BY providers.Фирма;

return; end; $$ language 'plpgsql';



2.3) По стоимости

CREATE OR REPLACE FUNCTION sort\_price() RETURNS TABLE

(id int, Поставщик varchar(40), Модель varchar(40), Дата\_выпуска date, Дата\_продажи date, Цена int)

AS $$ BEGIN RETURN QUERY

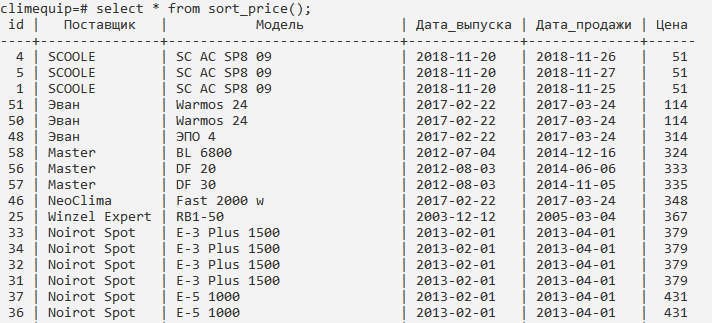
SELECT main.id, providers.Фирма, models.Модель, main.Дата\_выпуска, main.Дата\_продажи, main.Цена

FROM main, providers, models

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id)

ORDER BY main.Цена;

return; end; $$ language 'plpgsql';



2.4) По дате продажи

CREATE OR REPLACE FUNCTION sort\_sale\_date() RETURNS TABLE

(id int, Поставщик varchar(40), Модель varchar(40), Дата\_выпуска date, Дата\_продажи date, Цена int)

AS $$ BEGIN RETURN QUERY

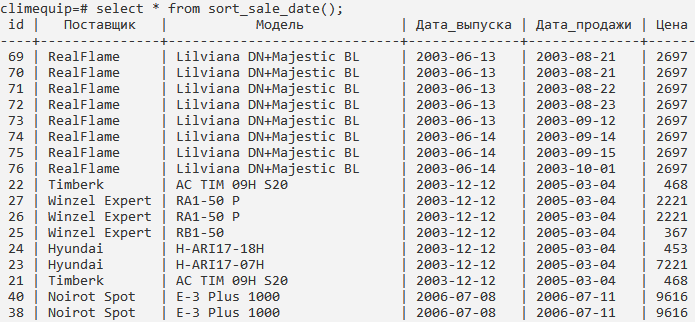
SELECT main.id, providers.Фирма, models.Модель, main.Дата\_выпуска, main.Дата\_продажи, main.Цена

FROM main, providers, models

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id)

ORDER BY main.Дата\_продажи;

return; end; $$ language 'plpgsql';



3) Найти КО:

3.1) Самое дорогое

CREATE OR REPLACE FUNCTION max\_price() RETURNS TABLE

(Фирма varchar(40), Каталог varchar(40), Модель varchar(40), Цена int)

AS $$ BEGIN RETURN QUERY

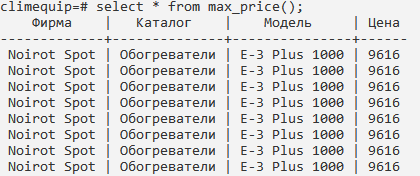
SELECT providers.Фирма, catalogs.Каталог, models.Модель, main.Цена

FROM providers, catalogs, models, main

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id

AND main.Цена = (SELECT MAX(main.Цена) FROM main));

return; end; $$ language 'plpgsql';



3.2) Самое дешевое

CREATE OR REPLACE FUNCTION min\_price() RETURNS TABLE

(Фирма varchar(40), Каталог varchar(40), Модель varchar(40), Цена int)

AS $$ BEGIN RETURN QUERY

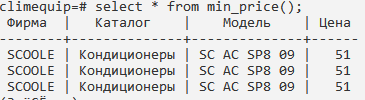
SELECT providers.Фирма, catalogs.Каталог, models.Модель, main.Цена

FROM providers, catalogs, models, main

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id

AND main.Цена = (SELECT MIN(main.Цена) FROM main));

return; end; $$ language 'plpgsql';



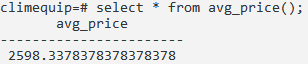
3.3) Средняя стоимость в целом

CREATE OR REPLACE FUNCTION avg\_price() RETURNS numeric

AS $$ DECLARE n numeric;

BEGIN n:=(SELECT AVG(main.Цена) FROM main);

return n; end; $$ language 'plpgsql';



3.4) Средняя стоимость по типам (Каталогам)

CREATE OR REPLACE FUNCTION avg\_price\_in() RETURNS TABLE

(Каталог varchar(40), Средняя\_стоимость numeric)

AS $$ BEGIN RETURN QUERY

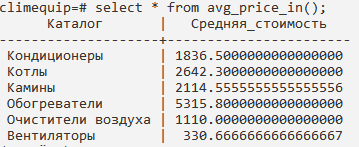
SELECT catalogs.Каталог AS Каталог, AVG(main.Цена) AS Средняя\_стоимость

FROM catalogs, models, main

WHERE(main.id\_Модели = models.id AND models.id\_Каталога = catalogs.id)

GROUP BY catalogs.Каталог;

return; end; $$ language 'plpgsql';



4)Найти КО с ценой в заданных пределах (предусмотреть ввод цены с клавиатуры)

4.1) По каждому виду

CREATE OR REPLACE FUNCTION price\_between\_for\_each\_catalog(int, int) RETURNS TABLE

(Фирма varchar(40), Каталог varchar(40), Модель varchar(40), Цена int)

AS $$ BEGIN RETURN QUERY

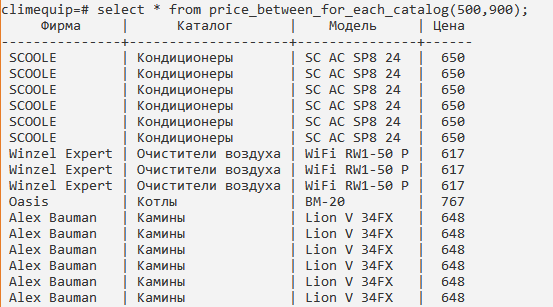
SELECT providers.Фирма, catalogs.Каталог, models.Модель, main.Цена

FROM providers, catalogs, models, main

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id

AND main.Цена > $1 AND main.Цена < $2);

return; end; $$ language 'plpgsql';



4.2) В целом

CREATE OR REPLACE FUNCTION price\_between(int, int) RETURNS TABLE

(Фирма varchar(40), Каталог varchar(40), Модель varchar(40), Цена int)

AS $$ BEGIN RETURN QUERY

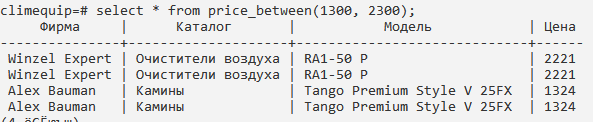
SELECT providers.Фирма, catalogs.Каталог, models.Модель, main.Цена

FROM providers, catalogs, models, main

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id

AND main.Цена > $1 AND main.Цена < $2);

return; end; $$ language 'plpgsql';



5) Найти все КО заданного производителя (выбор производителя)

CREATE OR REPLACE FUNCTION all\_equip\_of\_provider(varchar) RETURNS TABLE

(Фирма varchar(40), Каталог varchar(40), Модель varchar(40), Цена int)

AS $$ BEGIN RETURN QUERY

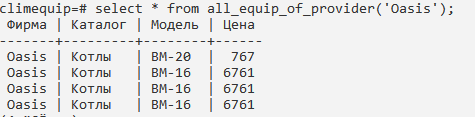
SELECT providers.Фирма, catalogs.Каталог, models.Модель, main.Цена

FROM providers, catalogs, models, main

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id

AND providers.Фирма = $1);

return; end; $$ language 'plpgsql';



6) Найти долю дешевого КО (меньше заданного, ввод ограничения) от общего числа КО

6.1) Для каждого вида (каталога)

CREATE OR REPLACE FUNCTION low\_part\_in\_catalog(varchar, int) RETURNS numeric

AS $$ DECLARE n numeric;

BEGIN

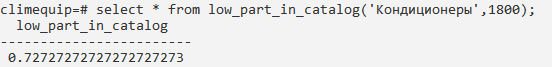
n:=((SELECT COUNT(\*) FROM main, models, catalogs

WHERE(main.id\_Модели = models.id AND models.id\_Каталога = catalogs.id AND main.Цена <=$2 AND catalogs.Каталог = $1))::numeric

/(SELECT COUNT(\*) FROM main,models, catalogs

WHERE(main.id\_Модели = models.id AND models.id\_Каталога = catalogs.id AND catalogs.Каталог = $1)))::numeric;

return n; end; $$ language 'plpgsql';



6.2) В целом

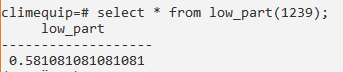
CREATE OR REPLACE FUNCTION low\_part(int) RETURNS numeric

AS $$ DECLARE n numeric;

BEGIN n:=(SELECT COUNT(\*) FROM main

WHERE main.Цена <=$1)/(SELECT COUNT(\*) FROM main)::numeric;

return n; end; $$ language 'plpgsql';



7) Найти все КО с заданной датой выпуска

CREATE OR REPLACE FUNCTION find\_by\_release\_date(date) RETURNS TABLE

(id int, Цена int, Дата\_выпуска date, Фирма varchar(40), Каталог varchar(40), Модель varchar(40))

AS $$ BEGIN RETURN QUERY

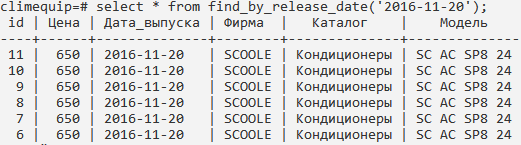
SELECT main.id, main.Цена,main.Дата\_выпуска, providers.Фирма, catalogs.Каталог, models.Модель

FROM main, providers, catalogs, models

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id

AND main.Дата\_выпуска = $1);

return; end; $$ language 'plpgsql';



8) Найти все КО, чья дата продажи находится в заданных пределах

8.1) В целом

CREATE OR REPLACE FUNCTION find\_if\_sale\_date\_between(date, date) RETURNS TABLE

(id int, Цена int, Дата\_продажи date, Фирма varchar(40), Каталог varchar(40), Модель varchar(40))

AS $$ BEGIN RETURN QUERY

SELECT main.id, main.Цена,main.Дата\_продажи, providers.Фирма, catalogs.Каталог, models.Модель

FROM main, providers, catalogs, models

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id

AND main.Дата\_продажи > $1 AND main.Дата\_продажи < $2);

return; end; $$ language 'plpgsql';



8.2) Для заданного производителя

CREATE OR REPLACE FUNCTION find\_if\_sale\_date\_between\_of\_provider(date, date, varchar) RETURNS TABLE

(id int, Цена int, Дата\_продажи date, Фирма varchar(40), Каталог varchar(40), Модель varchar(40))

AS $$ BEGIN RETURN QUERY

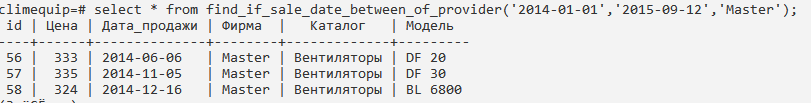
SELECT main.id, main.Цена,main.Дата\_продажи, providers.Фирма, catalogs.Каталог, models.Модель

FROM main, providers, catalogs, models

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id

AND main.Дата\_продажи > $1 AND main.Дата\_продажи < $2 AND providers.Фирма = $3);

return; end; $$ language 'plpgsql';



9) Найти долю КО, проданного за определенный период времени (ввод периода) от общего времени продажи

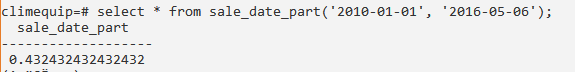
CREATE OR REPLACE FUNCTION sale\_date\_part(date, date) RETURNS numeric

AS $$ DECLARE n numeric;

BEGIN n:=(SELECT COUNT(\*) FROM main

WHERE main.Дата\_выпуска > $1 AND main.Дата\_выпуска < $2)/(SELECT COUNT(\*) FROM main)::numeric;

return n; end; $$ language 'plpgsql';



10) Найти количество бракованного КО, поступившего из заданной страны (выбор страны) для заданного поставщика (выбор поставщика).

CREATE OR REPLACE FUNCTION defective(varchar, varchar) RETURNS int

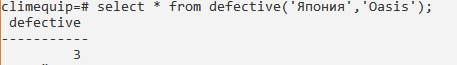
AS $$ DECLARE n int;

BEGIN n:=(SELECT COUNT(\*) FROM main, models, providers, countries

WHERE main.Наличие\_брака = true AND main.id\_Модели = models.id AND models.id\_Поставщика = providers.id

AND providers.id\_Страны = countries.id AND countries.Страна = $1 AND providers.Фирма = $2);

return n; end; $$ language 'plpgsql';



11) Найти все КО, поступившее от заданного поставщика, чья стоимость больше, чем заданная стоимость КО (ввод цены), поступившего из заданной страны (выбор страны).

CREATE OR REPLACE FUNCTION find\_by\_provider\_price\_country(varchar, int, varchar) RETURNS TABLE

(id int, Цена int, Дата\_продажи date, Фирма varchar(40), Каталог varchar(40), Модель varchar(40))

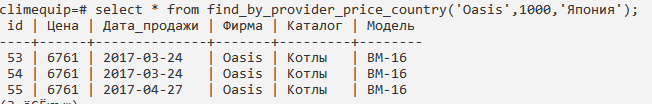
AS $$ BEGIN RETURN QUERY SELECT main.id, main.Цена,main.Дата\_продажи, providers.Фирма, catalogs.Каталог, models.Модель

FROM main, providers, catalogs, models, countries

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id

AND providers.id\_Страны = countries.id AND main.Цена > $2 AND providers.Фирма = $1 AND countries.Страна = $3);

return; end; $$ language 'plpgsql';



12) Найти долю дешевого КО (чья стоимость меньше заданной, ввод стоимости), поступившего от заданного поставщика (выбор поставщика).

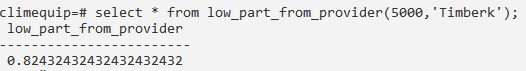
CREATE OR REPLACE FUNCTION low\_part\_from\_provider(int, varchar) RETURNS numeric

AS $$ DECLARE n numeric;

BEGIN n:=(SELECT COUNT(\*) FROM main, providers

WHERE (main.Цена <=$1 AND providers.Фирма = $2))/(SELECT COUNT(\*) FROM main)::numeric;

return n; end; $$ language 'plpgsql';



13) Найти среднюю стоимость КО, проданного за определенный промежуток времени (ввод интервала).

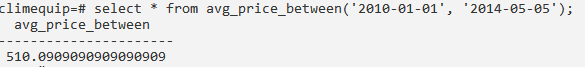
CREATE OR REPLACE FUNCTION avg\_price\_between(date,date) RETURNS numeric

AS $$ DECLARE n numeric;

BEGIN n:=(SELECT AVG(main.Цена) FROM main

WHERE (main.Дата\_продажи>$1 AND main.Дата\_продажи<$2))::numeric;

return n; end; $$ language 'plpgsql';



14) Найти все КО, чья стоимость выше, чем средняя стоимость КО заданного производителя (выбор производителя).

CREATE OR REPLACE FUNCTION price\_more\_avg\_of(varchar)

RETURNS TABLE(id int, Модель varchar, Фирма varchar, Каталог varchar, Цена int)

AS $$ BEGIN RETURN QUERY

SELECT main.id, models.Модель, providers.Фирма, catalogs.Каталог, main.Цена

FROM main, models, providers, catalogs

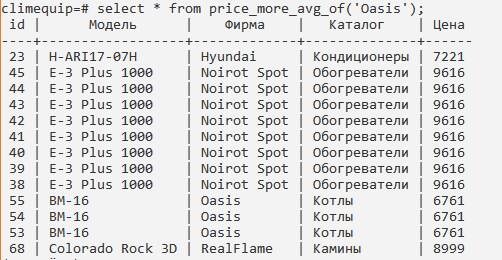
WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id

AND main.Цена > (SELECT AVG(main.Цена)

FROM main, models, providers

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND providers.Фирма = $1)));

return; end; $$ language 'plpgsql';



15) Определить долю регулярных поставок заданного КО (выбор) по заданной дате отгрузки (ввод) от общего числа КО.

CREATE OR REPLACE FUNCTION part\_of\_provider\_by\_date(varchar, date) RETURNS numeric

AS $$ DECLARE n numeric;

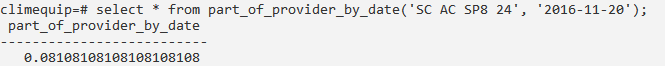
BEGIN n:=((SELECT COUNT(\*)

FROM main, models

WHERE main.id\_Модели = models.id AND models.Модель = $1 AND main.Дата\_выпуска = $2))::numeric /

(SELECT COUNT(\*) FROM main);

return n; end; $$ language 'plpgsql';



16) Найти КО, которое лучше всего раскупают, указать параметры (стоимость, производитель, страна и т.д.).

CREATE OR REPLACE FUNCTION popular\_equip() RETURNS TABLE(Модель varchar, Фирма varchar, Страна varchar, Цена int, Количество bigint)

AS $$ BEGIN RETURN QUERY

SELECT models.Модель, providers.Фирма, countries.Страна, main.Цена, COUNT(main.id\_Модели) AS cnt

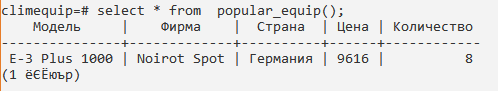
FROM providers, models, countries, main, catalogs

WHERE (main.id\_Модели = models.id AND models.id\_Поставщика = providers.id AND models.id\_Каталога = catalogs.id AND providers.id\_Страны = countries.id)

GROUP BY main.id\_Модели, providers.Фирма, countries.Страна, main.Цена, models.Модель

ORDER BY cnt desc limit 1;

return; end; $$ language 'plpgsql';

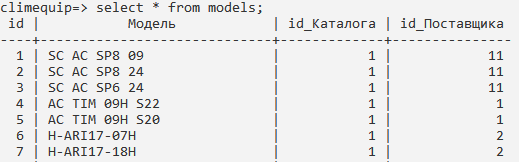


**Роли**

Пользователь

CREATE ROLE userdb WITH PASSWORD '00000' NOSUPERUSER NOCREATEDB LOGIN;

GRANT SELECT ON models, catalogs, providers, countries TO userdb;

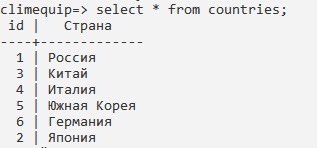




Оператор

CREATE ROLE operator WITH PASSWORD '00000' NOSUPERUSER NOCREATEDB LOGIN;

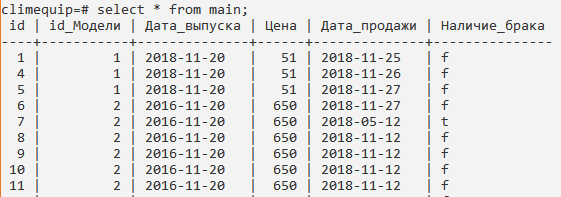
GRANT INSERT, UPDATE, DELETE, SELECT ON models, catalogs, providers, countries TO operator;





Администратор

CREATE ROLE admin with PASSWORD '00000' SUPERUSER CREATEDB LOGIN;



# Анализ результатов и выводы

Построена база данных в СУБД PostgreSQL с таблицами по заданной теме, представлена структура таблиц и связи между ними. Выполнены все предложенные задания: написаны функции для выполнения запросов с использованием процедурного языка plpgsql, приведены примеры результатов выполнения функций. Для реализации курсовой работы были применены знания и навыки разработки БД, выполнения запросов, написания функций, полученные в ходе выполнения лабораторных работ по курсу и изучения документации PostgreSQL.

# Список использованной литературы

1. Методические указания для выполнения лабораторных работ по курсу «Базы данных».

2. Документация по PostgreSQL [Электронный ресурс] // PostgreSQL.ru.net – сообщество русскоязычных пользователей PostgreSQL. – Режим доступа: <http://postgresql.ru.net/manual/index.html>. – Загл. с экр.