Кафедра вычислительной техники



**КУРСОВАЯ РАБОТА**

**по дисциплине: «***Базы данных***»**

**на тему:** **«***Разработка базы данных «Бытовая техника»»*

Выполнил:Проверил:

Студент гр. «*АВТ-010*»

*«Великжанин С.И.» «Трошина Г.В.»*

«\_\_\_» \_\_\_\_\_\_ 20\_\_г.«\_\_\_» \_\_\_\_\_\_ 20\_\_ г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (подпись)

**Оглавление**

[1. Задание: 3](#_Toc152780975)

[2. Графическое представление связей между таблицами: 4](#_Toc152780976)

[3. Структуры таблиц базы данных: 4](#_Toc152780977)

[4. Содержание таблиц баз данных: 6](#_Toc152780978)

[5. Руководство пользователя: 8](#_Toc152780979)

[6. Тексты запросов и примеры работы: 9](#_Toc152780980)

[7. Анализ результатов и выводы 32](#_Toc152780981)

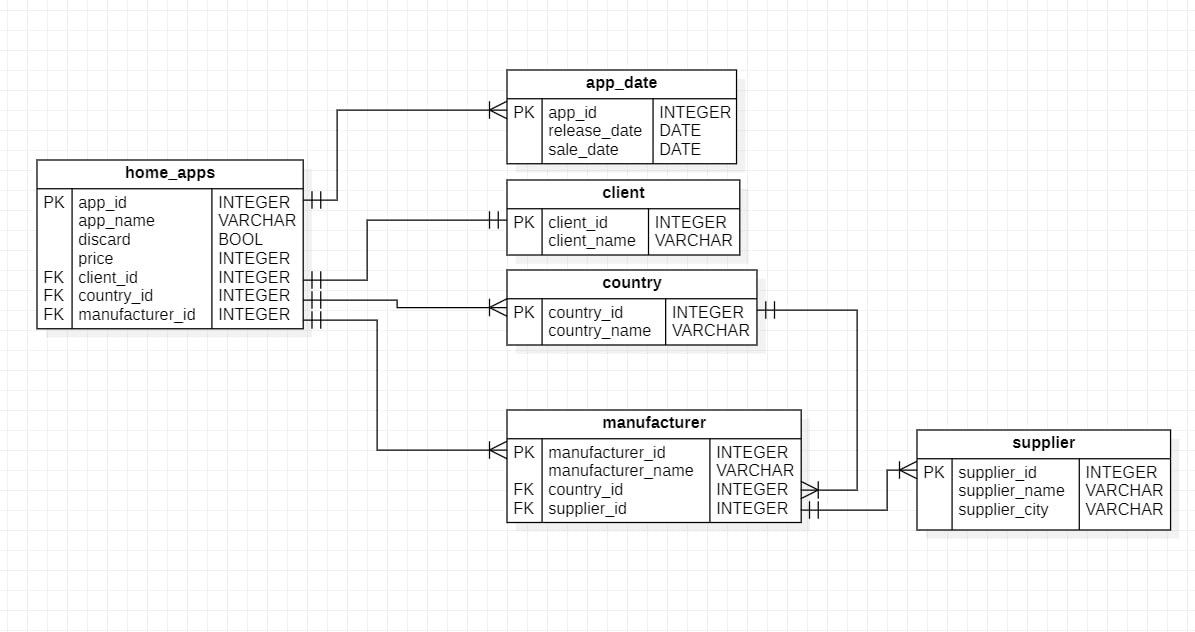
[8. Список использованной литературы 33](#_Toc152780982)

# **Задание:**

Сформировать несколько таблиц. Предусмотреть: ввод данных, редактирование, просмотр данных. Обязательные требования к базе данных: наличие таблиц-справочников и таблиц, использующих справочники; предусмотреть следующие роли: оператор базы данных; пользователь базы данных; администратор БД. Реализовать следующие типы запросов с использованием нескольких таблиц:

* Для каждой бытовой техники указать сведения о ней (наименование, дату выпуска, поставщик, цена, страна и т.п.)
* Получить список, отсортированный: по дате выпуска, по поставщику, по стоимости, по дате продажи.
* Найти самую дорогую бытовую технику, самую дешевую, среднюю стоймость.
* Найти бытовую технику с ценой в заданных пределах (предусмотреть ввод с клавиатуры)
* Найти всю бытовую технику заданного производителя (выбор)
* Найти долю дешевой бытовой техники (меньше заданного) от общего числа бытовой техники
* Найти всю бытовую технику с заданной датой выпуска (ввод даты)
* Найти всю бытовую технику, чья дата продажи находится в заданных пределах для заданного производителя и в целом.
* Найти долю бытовой техники, проданной за определенный период от общего времени продажи.
* Найти самый популярный вид бытовой техники
* Найти всю бытовую технику, поступившую от заданного поставщика, чья стоимость больше, чем средняя стоимость техники, поступившей из заданной страны
* Найти долю дешевой бытовой техники, поступившей от заданного поставщика и в целом
* Найти среднюю стоимость бытовой техники, проданной за определенный промежуток времени
* Найти всю бытовую технику, чья стоимость выше, чем средняя стоимость бытовой техники заданного производителя
* Определить долю регулярных поставок заданной бытовой техники, поступившей из заданного города
* Найти всю бракованную технику, поступившую из заданной страны для заданного поставщика
* Найти всю бытовую технику, проданную заданному клиенту за определенный период от общего времени продажи
* Найти долю дорогой бытовой техники, поступившей из заданной страны и в целом.

# **Графическое представление связей между таблицами:**



1. *Графическое представление связей между таблицами*

# **Структуры таблиц базы данных:**

Таблица - справочник supplier содержит информацию о поставщиках:

1. *Типы данных и смысловое содержание полей таблицы supplier*

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Пояснения |
| Supplier\_id | SERIAL PRIMARY KEY | идентификатор поставщика |
| Supplier\_name | VARCHAR(255) | название поставщика |
| Supplier\_city | VARCHAR(255) | город поставщика |

Таблица - справочник manufacturer содержит информацию о производителях:

1. *Типы данных и смысловое содержание полей таблицы manufacturer*

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Пояснения |
| manufacturer \_id | SERIAL PRIMARY KEY | идентификатор производителя |
| manufacturer \_name | VARCHAR(255) | название производителя |
| country\_id | INT REFERENCES COUNTRY(ID) | идентификатор страны |
| Supplier\_id | INT REFERENCES SUPPLIER(ID) | идентификатор поставщика |

Таблица - справочник country содержит информацию о странах:

1. *Типы данных и смысловое содержание полей таблицы country*

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Пояснения |
| Country\_id | SERIAL PRIMARY KEY | Идентификатор страны |
| Country\_name | VARCHAR(255) | Название страны |

Таблица – справочник client содержит информацию о клиентах:

1. *Типы данных и смысловое содержание полей таблицы client*

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Пояснения |
| client\_id | SERIAL PRIMARY KEY | Идентификатор клиента |
| сlient\_name | VARCHAR(255) | Имя клиента |

Таблица - справочник app\_date содержит информацию о датах:

1. *Типы данных и смысловое содержание полей таблицы app\_date*

|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Пояснения |
| App\_id | SERIAL PRIMARY KEY | Идентификатор записи |
| Release\_date | DATE | Дата выпуска прибора |
| Sale\_date | DATE | Дата продажи прибора |

Таблица home\_apps содержит информацию о бытовой технике:

*Таблица 6. Типы данных и смысловое содержание полей таблицы home\_apps*

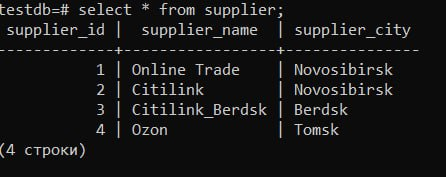
|  |  |  |
| --- | --- | --- |
| Имя поля | Тип данных | Пояснения |
| App\_id | SERIAL PRIMARY KEY | Идентификатор записи |
| App\_name | VARCHAR(255) | Вид техники |
| Discard | BOOLEAN | Брак |
| price | INTEGER | Цена |
| Client\_id | INT REFERENCES CLIENT(ID) | Идентификатор клиента |
| Country\_id | INT REFERENCES COUNTRY(ID) | Идентификатор страны |
| Manufacturer\_id | INT REFERENCES MANUFACTURER(ID) | Идентификатор производителя |

# **Содержание таблиц баз данных:**

1. Заполним таблицу supplier:

CREATE TABLE Supplier(  
supplier\_id serial primary key,  
supplier\_name varchar(255)

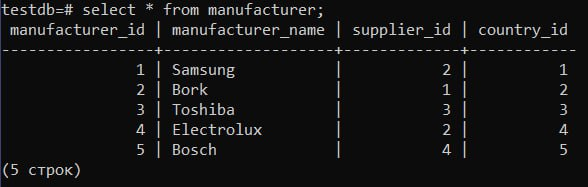
supplier\_city varchar(255));



1. *Содержание таблицы supplier*
2. Заполним таблицу manufacturer:

create table manufacturer (  
manufacturer\_id serial primary key,  
manufacturer\_name varchar(255),

supplier\_id int references Supplier(id)  
country\_id int references Country(id));



1. *Содержание таблицы manufacturer*
2. Заполним таблицу country:

create table Country(

country\_id serial primary key,

country\_name varchar(255));

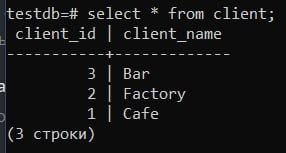


1. *Содержание таблицы country*
2. Заполним таблицу client:

create table Client(

client\_id serial primary key,

client\_name varchar(255));



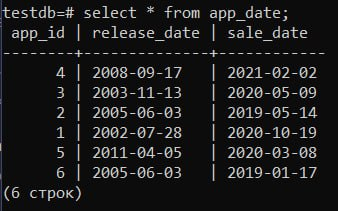
1. *Содержание таблицы* *client*
2. Заполним таблицу app\_date:

create table Types(

app\_id serial primary key,

release\_date date,

sale\_date date);



1. *Содержание таблицы app\_date*
2. Заполним таблицу home\_apps:

create table Types(

app\_id serial primary key,

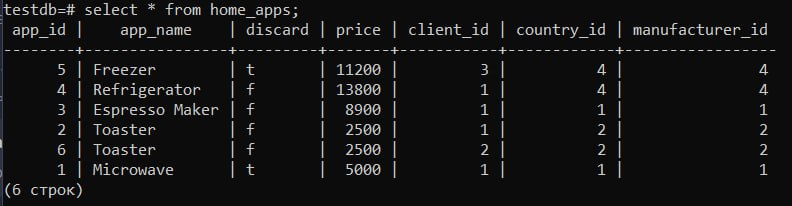
app\_name varchar(255),

discard boolean;

price integer,

client\_id int references Client(client\_id)  
country\_id int references Country(country\_id));

manufacturer\_id int references Manufacturer(manufacturer\_id));



1. *Содержание таблицы home\_apps*

# **Руководство пользователя:**

1. *Описание функций*

|  |  |  |
| --- | --- | --- |
| Наименование функции | Аргумент функции | Описание функции |
| app\_info() | - | Сведения о технике |
| sort\_by\_release() | - | Сортировка по дате выпуска |
| sort\_by\_supplier() | - | Сортировка по поставщикам |
| sort\_by\_price() | - | Сортировка по цене |
| sort\_by\_sale() | - | Сортировка по датам продаж |
| max\_price() | - | Самая дорогая техника |
| min\_price() | - | Самая дешевая техника |
| avg\_price() | - | Средняя цена |
| price\_range(bottom\_price int, top\_price int) | Нижняя, верхняя планки цен | Техника с ценой в заданных пределах |
| manufacturer\_search(manuf\_input varchar(255)) | Производитель | Вся техника для заданного производителя |
| search\_lowcost(input\_price int) | Цена | Доля техники с ценой ниже заданной |
| date\_search(date) | Дата | Поиск по дате выпуска |
| sale\_date\_range\_all(date, date) | Нижняя, верхняя планки дат | Все продажи в пределах указанных дат |
| sale\_date\_range\_name(date, date, varchar(255)) | Нижняя, верхняя планки дат, производитель | Все продажи для производителя в пределах указанных дат |
| sale\_interval\_p(date, date) | Нижняя, верхняя планки дат | Доля продаж от всех за определенный период |
| most\_popular() | - | Самый популярный вид техники |
| from\_manuf\_more\_country\_avg(varchar(255), varchar(255)) | Поставщик, страна | Вся техника поставщика, которая дороже, чем средняя цена техники из страны |
| supp\_cheap\_p(int, varchar(255)) | Цена, поставщик | Доля техники заданного поставщика, дешевле чем заданная цена |
| all\_cheap\_p(int) | Цена | Доля техники дешевле чем заданная цена |
| avg\_price\_interval(date, date) | Нижняя, верхняя планки дат | Средняя стоимость за промежуток времени |
| more\_than\_manuf\_avg(varchar(255)) | Производитель | Техника дороже, чем средняя стоимость по производителю |
| function app\_city\_p(varchar, varchar) | Вид техники, город | Доля регулярных поставок заданной техники из заданного города |
| function discard\_country(varchar, varchar) | Страна, поставщик | Весь брак заданной страны от заданного поставщика |
| function sale\_to\_client\_range(varchar(255), date, date) | Имя клиента, нижняя, верхняя планки дат | Все продажи для клиента за период |
| country\_exp\_p(int, varchar(255)) | Цена, страна | Доля техники из страны дороже, чем задано |
| all\_exp\_p(int) | Цена | Доля техники дороже, чем указано |

# **Тексты запросов и примеры работы:**

* 1. Для каждой бытовой техники указать сведения о ней (наименование, дату выпуска, поставщик, цена, страна и т.п.)

create or replace function app\_info()

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), release date, sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.release\_date, app\_date.sale\_date, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

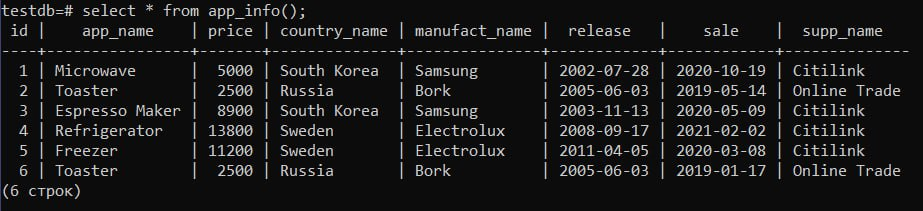
inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

order by app\_id ASC;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции app\_info*
2. Получить список, отсортированный: по дате выпуска, по поставщику, по стоимости, по дате продажи.

create or replace function sort\_by\_release()

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), release date, sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.release\_date, app\_date.sale\_date, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

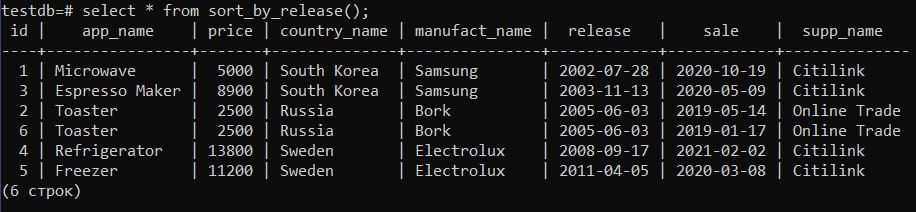
inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

order by release\_date ASC;

end;

$$ LANGUAGE plpgsql; 

1. *Результат работы функции sort\_by\_release*

create or replace function sort\_by\_supplier()

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), release date, sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.release\_date, app\_date.sale\_date, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

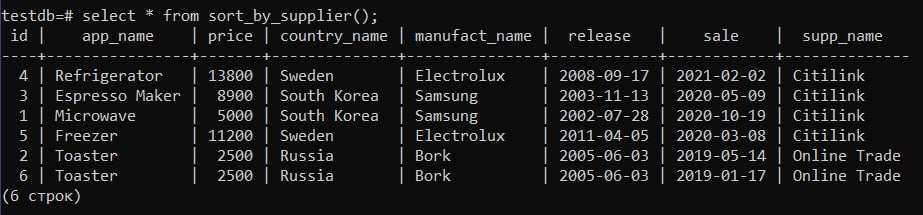
inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

order by supplier\_name ASC;

end;

$$ LANGUAGE plpgsql; 

1. *Результат работы функции sort\_by\_supplier*

create or replace function sort\_by\_price()

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), release date, sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.release\_date, app\_date.sale\_date, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

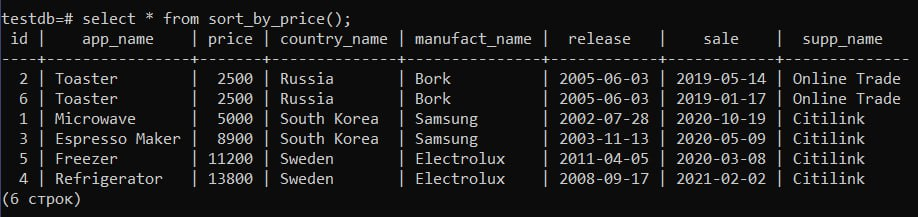
inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

order by price ASC;

end;

$$ LANGUAGE plpgsql; 

1. *Результат работы функции sort\_by\_price*

create or replace function sort\_by\_sale()

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), release date, sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.release\_date, app\_date.sale\_date, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

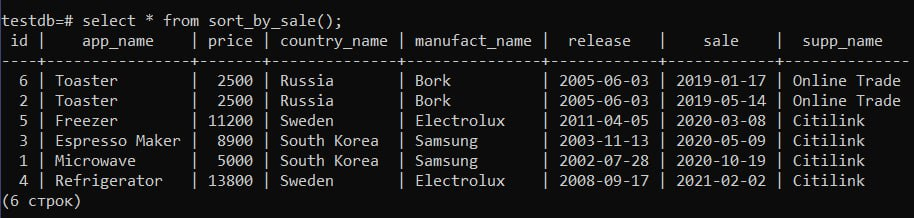
inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

order by sale\_date ASC;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции sort\_by\_sale*
2. Найти самую дорогую бытовую технику, самую дешевую, среднюю стоймость.

* Самая дорогая:

create or replace function max\_price()

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), release date, sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.release\_date, app\_date.sale\_date, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

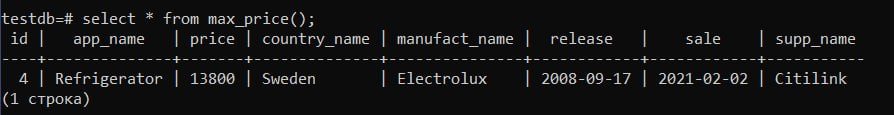
inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

where home\_apps.price = (select max(home\_apps.price) from home\_apps);

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции max\_price*

* Самая дешевая:

create or replace function min\_price()

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), release date, sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.release\_date, app\_date.sale\_date, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

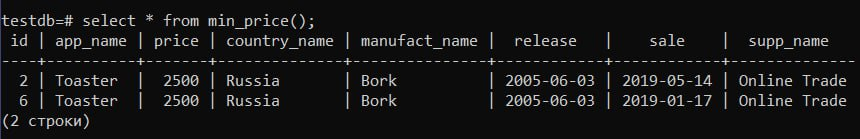
inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

where home\_apps.price = (select min(home\_apps.price) from home\_apps);

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции min\_price*

* Средняя стоимость:

create or replace function avg\_price()

returns float

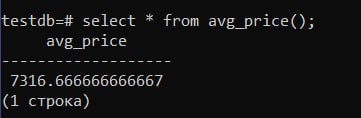
as $$

begin

return (select avg(home\_apps.price) as price from home\_apps);

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции avg\_price*
2. Найти бытовую технику с ценой в заданных пределах (предусмотреть ввод с клавиатуры)

create or replace function price\_range(bottom\_price int, top\_price int)

returns TABLE(id int, app\_name varchar(255), price int)

as $$

begin

return query

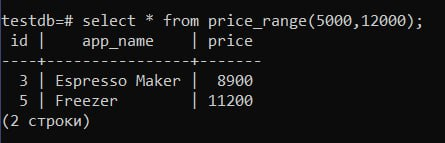
select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price

from home\_apps

group by home\_apps.app\_id having home\_apps.price > bottom\_price AND home\_apps.price < top\_price

order by home\_apps.price;

end;

$$ LANGUAGE plpgsql;

1. *Результат работы функции price\_range*
2. Найти всю бытовую технику заданного производителя (выбор)

create or replace function manufacturer\_search(manuf\_input varchar(255))

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), release date, sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.release\_date, app\_date.sale\_date, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

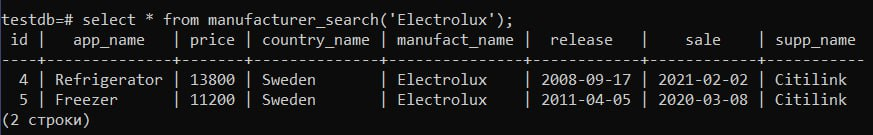
inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

and manufacturer.manufacturer\_name = manuf\_input;

end;

$$ LANGUAGE plpgsql;

1. *Результат работы функции manufacturer\_search*
2. Найти долю дешевой бытовой техники (меньше заданного) от общего числа бытовой техники

create or replace function search\_lowcost(input\_price int)

returns float

as $$

declare

a float;

b float;

begin

select count(\*) into a from home\_apps

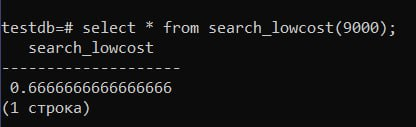
where home\_apps.price <= input\_price;

select count (\*) into b from home\_apps;

return a/b;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции search\_lowcost*
2. Найти всю бытовую технику с заданной датой выпуска (ввод даты)

create or replace function date\_search(date)

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), release date, sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.release\_date, app\_date.sale\_date, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

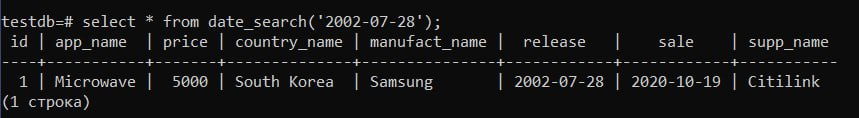
inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

where app\_date.release\_date = $1;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции date\_search*
2. Найти всю бытовую технику, чья дата продажи находится в заданных пределах для заданного производителя и в целом.

create or replace function sale\_date\_range\_all(date, date)

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.sale\_date, client.client\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

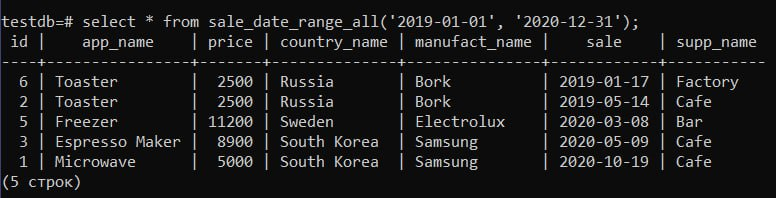
inner join client on home\_apps.client\_id = client.client\_id

where app\_date.sale\_date >= $1 and app\_date.sale\_date <= $2

order by app\_date.sale\_date;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции sale\_date\_range\_all*

create or replace function sale\_date\_range\_name(date, date, varchar(255))

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), sale date, supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.sale\_date, client.client\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

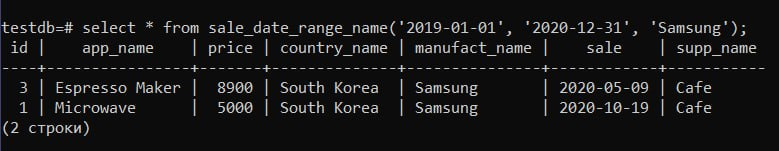
inner join client on home\_apps.client\_id = client.client\_id

where app\_date.sale\_date >= $1 and app\_date.sale\_date <= $2 and manufacturer.manufacturer\_name = $3

order by app\_date.sale\_date;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции sale\_date\_range\_name*
2. Найти долю бытовой техники, проданной за определенный период от общего времени продажи.

create or replace function sale\_interval\_p(date, date)

returns float

as $$

declare

a float;

b float;

begin

select count(\*) into a from app\_date

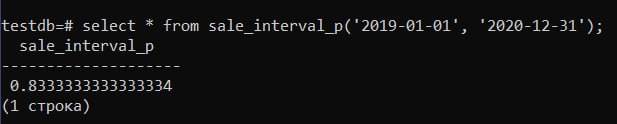
where app\_date.sale\_date >= $1 and app\_date.sale\_date <= $2;

select count (\*) into b from app\_date;

return a/b;

end;

$$ LANGUAGE plpgsql;

$$ LANGUAGE plpgsql; 

1. *Результат работы функции sale\_interval\_p*
2. Найти самый популярный вид бытовой техники

create or replace function most\_popular()

returns table(app\_name varchar(255), total bigint)

as $$

begin

return query

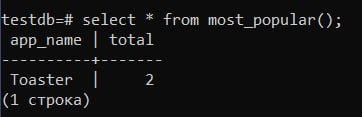
select home\_apps.app\_name, count (\*) as total from home\_apps

group by home\_apps.app\_name

limit 1;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции most\_popular*
2. Найти всю бытовую технику, поступившую от заданного поставщика, чья стоимость больше, чем средняя стоимость техники, поступившей из заданной страны

create or replace function from\_manuf\_more\_country\_avg(varchar(255), varchar(255))

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), sale date, supp\_name varchar(255), avg\_price numeric)

as $$

begin

return query

with avg\_country\_price as(

select avg(home\_apps.price) as avg\_price

from home\_apps

inner join country on home\_apps.country\_id = country.country\_id

where country.country\_name = $2

)

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.sale\_date, client.client\_name, avg\_country\_price.avg\_price

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join client on home\_apps.client\_id = client.client\_id

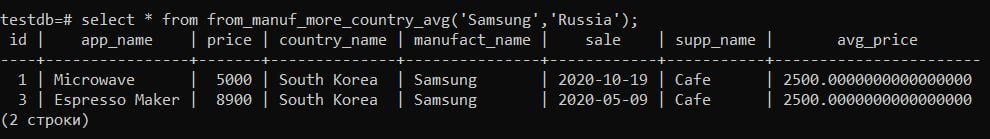
inner join avg\_country\_price on 1=1

where manufacturer.manufacturer\_name = $1 and home\_apps.price >= avg\_country\_price.avg\_price

order by home\_apps.app\_id;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции from\_manuf\_more\_country\_avg*
2. Найти долю дешевой бытовой техники, поступившей от заданного поставщика и в целом

* От заданного

create or replace function supp\_cheap\_p(int, varchar(255))

returns float

as $$

declare

a float;

b float;

begin

select count(\*) into a from home\_apps

inner join manufacturer on home\_apps.manufacturer\_id = manufacturer.manufacturer\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

where home\_apps.price < $1

and supplier.supplier\_id in(

select supplier.supplier\_id from supplier

where supplier.supplier\_name = $2);

select count (\*) into b from home\_apps

inner join manufacturer on home\_apps.manufacturer\_id = manufacturer.manufacturer\_id

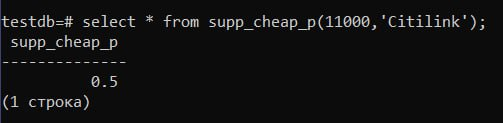
inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

where supplier.supplier\_name = $2;

return a/b;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции supp\_cheap\_p*

* В целом

create or replace function all\_cheap\_p(int)

returns float

as $$

declare

a float;

b float;

begin

select count(\*) into a from home\_apps

inner join manufacturer on home\_apps.manufacturer\_id = manufacturer.manufacturer\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

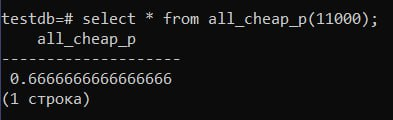
where home\_apps.price < $1;

select count (\*) into b from home\_apps;

return a/b;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции all\_cheap\_p*
2. Найти среднюю стоимость бытовой техники, проданной за определенный промежуток времени

create or replace function avg\_price\_interval(date, date)

returns float

as $$

begin

return (

select avg(home\_apps.price) from home\_apps

where home\_apps.app\_id in(

select app\_date.app\_id from app\_date

where app\_date.sale\_date >$1 and app\_date.sale\_date < $2));

end;

$$ LANGUAGE plpgsql; 

1. *Результат работы функции avg\_price\_interval*
2. Найти всю бытовую технику, чья стоимость выше, чем средняя стоимость бытовой техники заданного производителя

create or replace function more\_than\_manuf\_avg(varchar(255))

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), sale date, supp\_name varchar(255), avg\_price numeric)

as $$

begin

return query

with avg\_manuf\_price as(

select avg(home\_apps.price) as avg\_price

from home\_apps

inner join manufacturer on home\_apps.manufacturer\_id = manufacturer.manufacturer\_id

where manufacturer.manufacturer\_name = $1

)

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.sale\_date, client.client\_name, avg\_manuf\_price.avg\_price

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join client on home\_apps.client\_id = client.client\_id

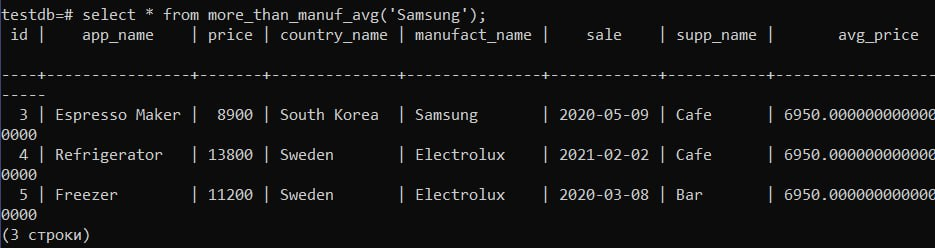
inner join avg\_manuf\_price on 1=1

where home\_apps.price >= avg\_manuf\_price.avg\_price

order by home\_apps.app\_id;

end;

$$ LANGUAGE plpgsql;



1. *Результат работы функции more\_than\_manuf\_avg*
2. Определить долю регулярных поставок заданной бытовой техники, поступившей из заданного города

create or replace function app\_city\_p(varchar, varchar)

returns float

as $$

declare

a float;

b float;

begin

select count(\*) into a from home\_apps

inner join manufacturer on home\_apps.manufacturer\_id = manufacturer.manufacturer\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

where home\_apps.app\_name = $1 and supplier.supplier\_city = $2;

select count (\*) into b from home\_apps

inner join manufacturer on home\_apps.manufacturer\_id = manufacturer.manufacturer\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

where supplier.supplier\_city = $2;

return a/b;

end;

$$ LANGUAGE plpgsql; 

1. *Результат работы функции app\_city\_p*
2. Найти всю бракованную технику, поступившую из заданной страны для заданного поставщика

create or replace function discard\_country(varchar, varchar)

returns TABLE(id int, app\_name varchar(255), price int, discard bool, country\_name varchar(255),

manufact\_name varchar(255), supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, home\_apps.discard, country.country\_name, manufacturer.manufacturer\_name, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

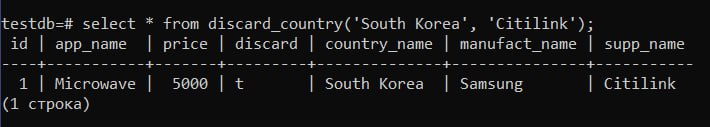
inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

where home\_apps.discard = true and country.country\_name = $1 and supplier.supplier\_name = $2

order by home\_apps.app\_id;

end;

$$ LANGUAGE plpgsql; 

1. *Результат работы функции discard\_country*
2. Найти всю бытовую технику, проданную заданному клиенту за определенный период от общего времени продажи:

create or replace function sale\_to\_client\_range(varchar(255), date, date)

returns TABLE(id int, app\_name varchar(255), price int, country\_name varchar(255),

manufact\_name varchar(255), sale date, client varchar(255), supp\_name varchar(255))

as $$

begin

return query

select home\_apps.app\_id, home\_apps.app\_name, home\_apps.price, country.country\_name, manufacturer.manufacturer\_name, app\_date.sale\_date, client.client\_name, supplier.supplier\_name

from home\_apps

inner join country ON home\_apps.country\_id = country.country\_id

inner join manufacturer ON home\_apps.manufacturer\_id= manufacturer.manufacturer\_id

inner join app\_date ON home\_apps.app\_id = app\_date.app\_id

inner join client on home\_apps.client\_id = client.client\_id

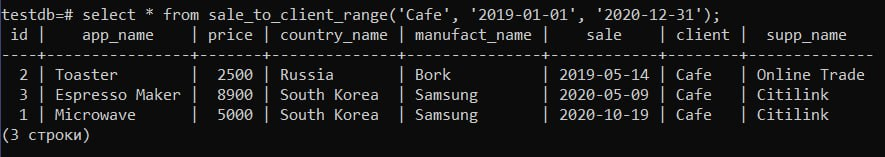
inner join supplier on manufacturer.supplier\_id = supplier.supplier\_id

where client.client\_name = $1 and app\_date.sale\_date >= $2 and app\_date.sale\_date <= $3

order by app\_date.sale\_date;

end;

$$ LANGUAGE plpgsql;



*Рисунок 31. Результат работы функции sale\_to\_client\_range*

1. Найти долю дорогой бытовой техники, поступившей из заданной страны и в целом.

create or replace function country\_exp\_p(int, varchar(255))

returns float

as $$

declare

a float;

b float;

begin

select count(\*) into a from home\_apps

inner join manufacturer on home\_apps.manufacturer\_id = manufacturer.manufacturer\_id

inner join country on home\_apps.country\_id = country.country\_id

where home\_apps.price > $1

and country.country\_id in(

select country.country\_id from supplier

where country.country\_name = $2);

select count (\*) into b from home\_apps

inner join manufacturer on home\_apps.manufacturer\_id = manufacturer.manufacturer\_id

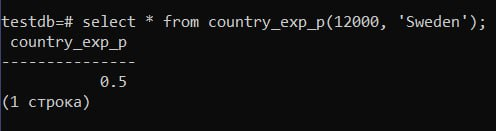
inner join country on home\_apps.country\_id = country.country\_id

where country.country\_name = $2;

return a/b;

end;

$$ LANGUAGE plpgsql;



*Рисунок 32. Результат работы функции country\_exp\_p*

create or replace function all\_exp\_p(int)

returns float

as $$

declare

a float;

b float;

begin

select count(\*) into a from home\_apps

inner join manufacturer on home\_apps.manufacturer\_id = manufacturer.manufacturer\_id

inner join country on home\_apps.country\_id = country.country\_id

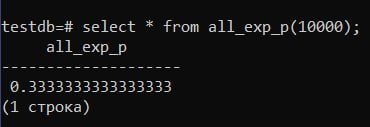
where home\_apps.price > $1 ;

select count (\*) into b from home\_apps;

return a/b;

end;

$$ LANGUAGE plpgsql;



*Рисунок 33. Результат работы функции all\_exp\_p*

1. Создание ролей для базы данных

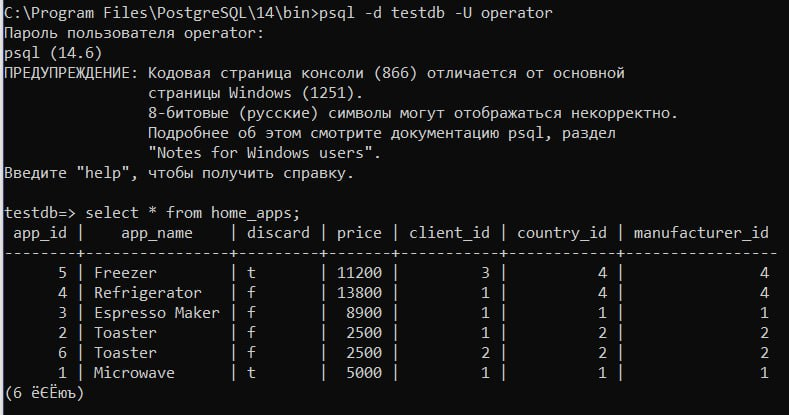
create role operator with password '1' nosuperuser nocreatedb nocreaterole login;

create role user\_db with password '1' nosuperuser nocreatedb nocreaterole login;

create role admin with password '4' superuser createdb createrole login;

Оператор может редактировать и просматривать справочники:

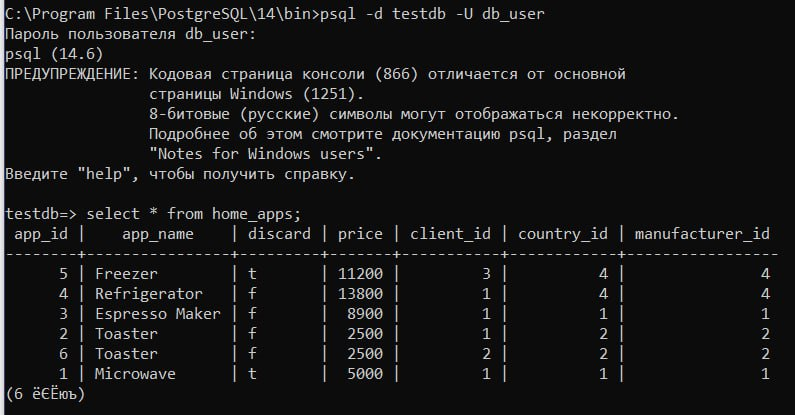
Grant insert, delete, update, select on table app\_date, client, country, manufacturer, supplier, home\_apps to operator;



*Рисунок 34. Демонстрация прав доступа роли operator*

Пользователь может просматривать, удалять, обновлять и добавлять записи во всех таблицы, кроме таблицы-журнала и справочников. Справочники он может только просматривать:

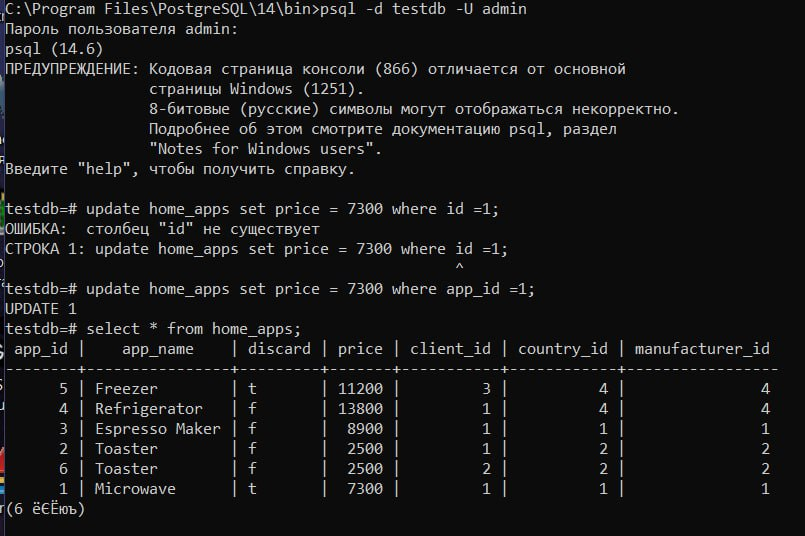
Grant update, select on table home\_apps, app\_date to db\_user;



*Рисунок 35. Демонстрация прав доступа роли user\_db*

Администратор может совершать любые действия над всеми таблицами.

Grant insert, delete, update, select on table home\_apps, app\_date, country, supplier, manufacturer, client to admin;



*Рисунок 36.Демонстрация прав доступа роли admin*

# **Анализ результатов и выводы**

Построена база данных в СУБД PostgreSQL. В базе данных присутствуют таблицы -справочники и таблицы, использующие справочники. В пояснительной записке представлена структура таблиц и связи между ними. Написаны функции для реализации 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. Базы данных: методические указания по выполнению курсовой работы для 3 курса факультета автоматики и вычислительной техники по направлению "Информатика и вычислительная техника" / Новосиб. гос. техн. ун-т ; [сост.: В. А. Астапчук, В. Б. Панова]. - Новосибирск, 2012. - 60, [1] с. : ил., табл.. - Режим доступа: http://elibrary.nstu.ru/source?bib\_id=vtls000173910 – (Дата обращения: 25.09.2023)
2. Трошина Г. В. Базы данных [Электронный ресурс] : электронный учебно-методический комплекс / Г. В. Трошина ; Новосиб. гос. техн. ун-т. - Новосибирск, [2011]. - Режим доступа: http://elibrary.nstu.ru/source?bib\_id=vtls000152242. - Загл. с экрана – (Дата обращения: 25.09.2023)
3. Функции SQL : справочник программиста : [пер. с англ.] / Эйри Джоунс [и др.]. - М., 2007. - 760 с. : табл.
4. «Курс SQL для начинающих» [видеоролики с канала EngineerSpock - IT & программирование]:  
   https://www.youtube.com/watch?v=HVQNxdI6fqY&list=PLBheEHDcG7-k1Y\_Uy04Dj2ylWhcfSfqoF (Дата обращения: 16.10.2023)