МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕВЫСШЕГО ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

|  |
| --- |
| Кафедра вычислительной техники |

**КУРСОВАЯ РАБОТА**

**ПО ДИСЦИПЛИНЕ**

**«БАЗЫ ДАННЫХ»**

|  |  |
| --- | --- |
| Тема: | Проектирование базы данных «Лекарства» |

|  |  |
| --- | --- |
| Студент: | Гунгер А.К. |

|  |  |
| --- | --- |
| Группа: | АВТ-709 |

Вариант: 113

|  |  |
| --- | --- |
| Преподаватель: | Трошина Г.В. |

|  |  |
| --- | --- |
| Представлено к защите: |  |

**Оглавление**

[**1. Задание 3**](#_Toc47916869)

[**2. Структура таблиц базы данных и графическое представление связей 4**](#_Toc47916870)

[**3. Создание таблиц 5**](#_Toc47916871)

[**4. Содержание таблиц баз данных 6**](#_Toc47916872)

[**5. Руководство пользователя 11**](#_Toc47916873)

[**6. Тексты запросов и примеры работы 15**](#_Toc47916874)

[**7. Анализ результатов и выводы 29**](#_Toc47916875)

[**8. Список литературы 30**](#_Toc47916876)

1. **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 каждого типа лекарств (микстура, таблетки, капли и т.д.) указать сведения о нем (наименование, дата выпуска и т.д.).
* Для каждого типа лекарств выдать список, отсортированный:

По дате выпуска, алфавитном порядке по имени, по сроку годности, по стоимости.

* Найти самое дорогое лекарство, самое дешевое, среднюю стоимость для каждого типа лекарств и в целом.
* Для каждого типа заболевания найти лекарства с ценой свыше 20 р. (и любая другая сумма, т.е. предусмотреть ввод цены с клавиатуры).
* Найти все лекарства для заданного типа заболевания (выбор), чья стоимость попадает в заданные пределы (ввод интервала).
* Найти долю дешевых лекарств (меньше заданного, ввод) от общего числа лекарств.
* Определить количество лекарств заданного изготовителя (выбор), чья стоимость находится в заданных предела (ввод интервала).
* Найти все лекарства, поступившие от заданного изготовителя (выбор) для лечения заданного заболевания (выбор).
* Найти долю лекарств заданного изготовителя (выбор) от общего числа лекарств.
* Определить долю регулярных поставок заданного типа (выбор типа).
* Для заданного поставщика (выбор) определить все лекарства, чей температурный режим хранения находится в заданных пределах (ввод интервала).
* Найти все лекарства, поступившие из заданной страны (выбор страны), чья стоимость больше заданной (ввод стоимости).
* Найти долю дешевых лекарств (меньше заданного, ввод ограничения) от общего числа лекарств (для определенного возраста (например, для детей, для тех, кто старше 6 лет, 12 лет и т.д.)).
* Найти самое популярное лекарство, которое покупают чаще всего для заданного сезона.

1. **Структура таблиц базы данных и графическое представление связей**

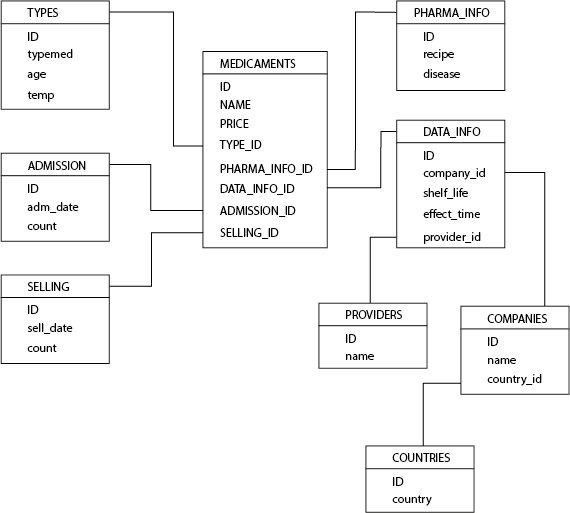


Рисунок 1. Структура базы данных

1. **Создание таблиц**

* Создание таблицы для хранения типов лекарств – types:

*create table types(id serial, typeMed char(50), age int, temp int);*

* Создание таблицы для хранения производящих компаний– companies:

*create table companies(id serial, name char(50), country\_id int);*

* Создание таблицы для хранения транспортной и технической информации– data\_info:

*create table data\_info(id serial, company\_id int, date date, shelf\_life int, effect\_time int, provider\_id int);*

* Создание таблицы для хранения характеристических опций– pharma\_info:

*create table pharma\_info(id serial, recipe bit, disease char(50));*

* Создание таблицы для хранения стран из которых возможен импорт – countries:

*create table countries(id serial, country char(50));*

* Создание таблицы для хранения информации о поступлении лекарств– admission:

*create table admission(id serial, adm\_date date, count int);*

* Создание таблицы для хранения информации о продажах– selling:

*create table selling(id serial, sell\_date date, count int);*

* Создание таблицы для хранения возможных поставщиков – providers:

*create table providers(id serial,name char(50));*

* Создание таблицы для хранения полной информации о каждом лекарственном средстве– medicaments:

*create table medicaments(id serial,name char(50), price int, type\_id int, data\_info\_id int, pharma\_info\_id*

*int, admission\_id int, selling\_id int);*

1. **Содержание таблиц баз данных**

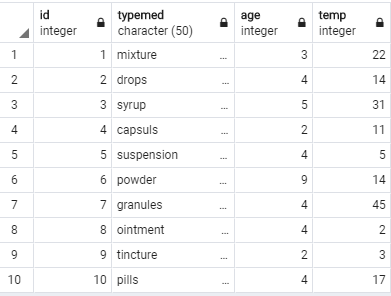


Рисунок 2. Таблица types

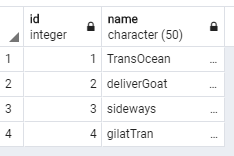


Рисунок 3. Таблица providers

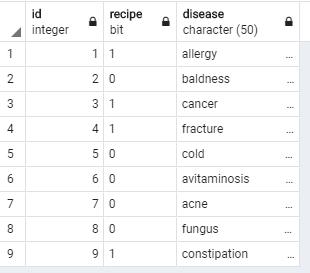
****

Рисунок 4. Таблица pharma\_info

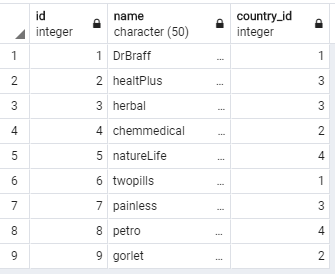


Рисунок 5. Таблица companies

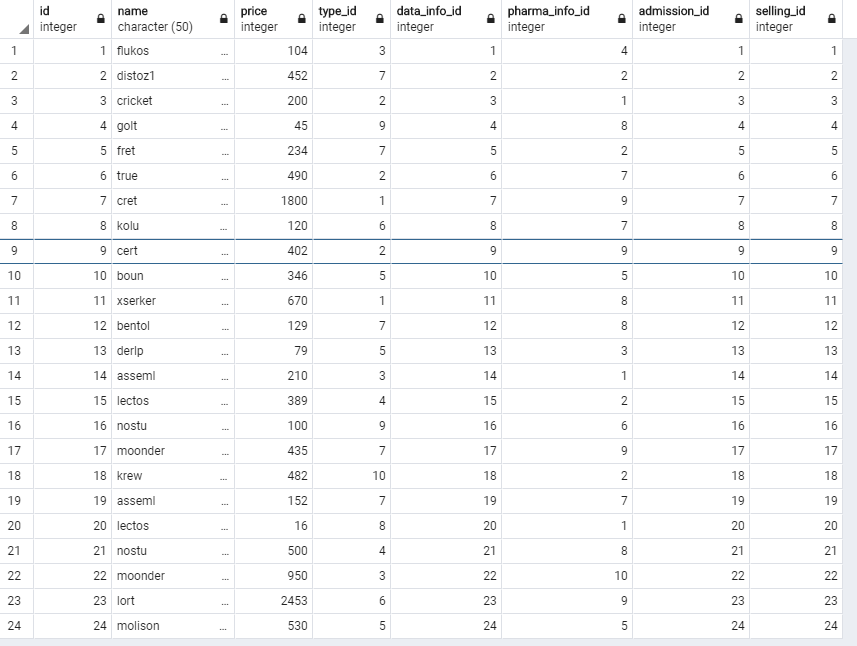


Рисунок 6. Таблица medicaments

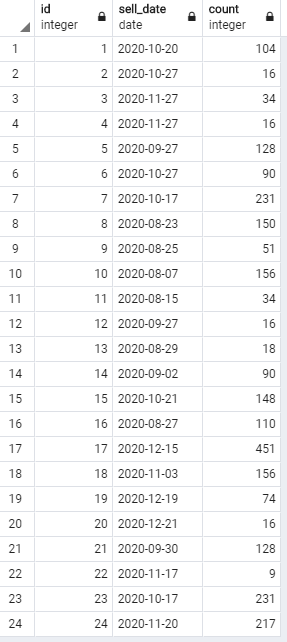


Рисунок 7. Пример таблицы selling

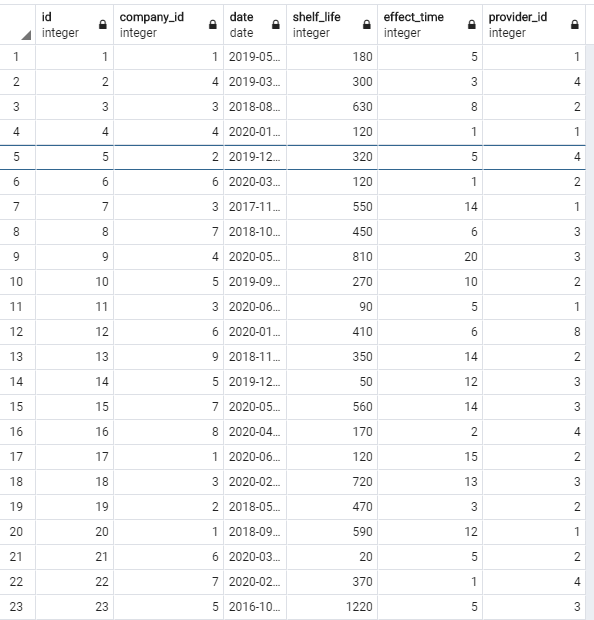


Рисунок 8. Пример таблицы data\_info

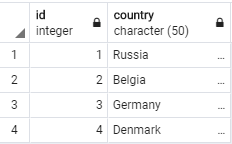


Рисунок 9. Пример таблицы countries

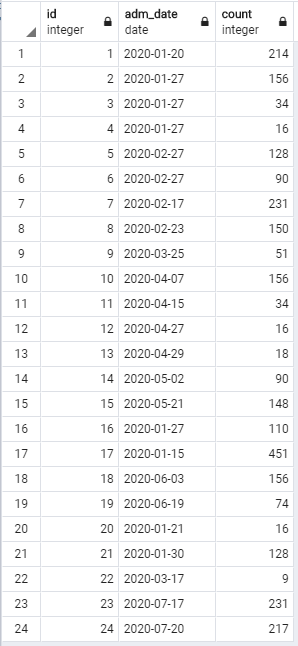


Рисунок 10. Пример таблицы admissions

1. **Руководство пользователя**

Оператор может редактировать, добавлять в справочники и просматривать их. Также ему предоставляется спектр функций для управления и сбора некоторых данных по таблицам.

\c pharma operator;

Пользователь БД может просматривать данные в таблицах и использовать узконаправленные функции для упрощения поиска нужной продукции (Так сказать фильтры).

\c pharma guest;

Администратор может просматривать, удалять, обновлять и добавлять новые записи в любые таблицы и производить любые действия над самими таблицами. Администратору предоставляется весь перечень реализованных функций.

\c pharma admin;

Список функций:

getType(text)

sorted(text,text)

average\_price(text)

max\_min\_price(text)

overPrice(text,integer)

intervalPrice(text,integer,integer)

chipOfAll(integer)

numbByInterval(text,integer,integer)

getNum(text,integer,integer)

getCompany\_desiase(text,text)

partByCompany(text)

partOfProviding(text)

providerByTemp(text,integer,integer)

medFromCountry(text,integer)

partOfChipMeds(integer,integer)

get\_most\_popular(date,date)

**Распределение функций среди ролей**

GUEST:

getType(text),

sorted(text,text),

max\_min\_price(text),

chipOfAll(integer),

overPrice(text,integer),

medFromCountry(text,integer),

OPERATOR:

getType(text),

average\_price(text),

max\_min\_price(text),

sorted(text,text),

max\_min\_price(text),

chipOfAll(integer),

partByCompany(text),

partOfProviding(text),

overPrice(text,integer),

providerByTemp(text,integer,integer),

medFromCountry(text,integer),

get\_most\_popular(date,date)

ADMIN:

getType(text),

sorted(text,text),

average\_price(text),

max\_min\_price(text),

average\_price(text),

overPrice(text,integer),

intervalPrice(text,integer,integer),

chipOfAll(integer),

numbByInterval(text,integer,integer),

getNum(text,integer,integer),

getCompany\_desiase(text,text),

partByCompany(text),

partOfProviding(text),

providerByTemp(text,integer,integer),

medFromCountry(text,integer),

partOfChipMeds(integer,integer),

get\_most\_popular(date,date)

**Назначение и использование функций**

* getType(text)

Выводит список лекарств данного типа (Вход: тип лекарства)

* sorted(text,text)

Получение отсортированного списка по типу лекарства (Вход: тип лекарства, тип сортировки)

* average\_price(text)

Выводит среднюю стоимость по типу или общего списка лекарств (Вход: тип лекарства/ all)

* max\_min\_price(text)

Выводит самое дешевое и дорогое лекарство (Вход: min/max)

* overPrice(text,integer)

Получение таблицы лекарств по болезни и по стоимости выше заданной (Вход: болезнь, цена)

* intervalPrice(text,integer,integer)

Выводит лекарства по типу и по интервалу стоимости (Вход: тип лекарства, цена, цена)

* chipOfAll(integer)

Получении доли дешевых лекарств меньше заданного (Вход: цена)

* numbByInterval(text,integer,integer)

Выводит лекарства заданного изготовителя по интервалу стоимости (Вход: компания, цена, цена)

* getNum(text,integer,integer)

Выводит количество лекарств заданного изготовителя по интервалу стоимости (Вход: тип лекарства)

* getCompany\_desiase(text,text)

Получение таблицы лекарств от компании производителя и заданного заболевания (Вход: компания, заболевание)

* partByCompany(text)

Выводит долю лекарств заданного изготовителя (Вход: тип лекарства)

* partOfProviding(text)

Выводит долю регулярных поставок по типу лекарств (Вход: тип лекарства)

* providerByTemp(text,integer,integer)

Выводит лекарства от поставщика и в интервале заданных температур (Вход: поставщик, температура, температура)

* medFromCountry(text,integer)

Выводит лекарства из определенной страны (Вход: страна)

* partOfChipMeds(integer,integer)

Выводит долю дешевых лекарств по цене меньше заданной и для возраста больше заданного (Вход: цена, возраст)

* get\_most\_popular(date,date)

Выводит самое популярное лекарство за выбранный сезон (Вход: дата, дата)

1. **Тексты запросов и примеры работы**

**Функция 1**. Для каждого типа лекарств (микстура, таблетки, капли и т.д.) указать сведения о нем (наименование, дата выпуска и т.д.).

Реализация функции:

create or replace FUNCTION getType(text)

RETURNS table(

type char(15),

medicament char(15),

company char(15),

price int,

shelf\_time int,

disease char(15),

recipe bit

)

AS

$$

BEGIN

RETURN query

select t.typemed, m.name, cs.name, m.price, di.shelf\_life  , pi.disease, pi.recipe

from medicaments m

inner join types t on m.type\_id = t.id

inner join data\_info di on m.data\_info\_id = di.id

inner join companies cs on di.company\_id = cs.id

inner join pharma\_info pi on m.pharma\_info\_id = pi.id

where t.typemed = $1;

END;

$$ LANGUAGE plpgsql;

Результат выполнения:

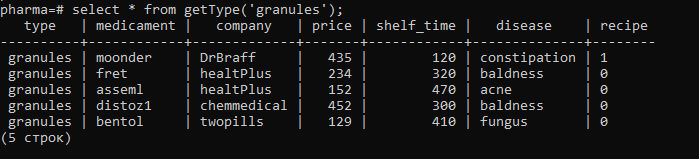


Рисунок 11. Результат выполнения функции

**Функция 2**. Для каждого типа лекарств выдать список, отсортированный:

Реализация функции:

create or replace FUNCTION sorted(text,text)

RETURNS table(

type text,

name text,

price int,

date date,

shelf\_time int

)

AS

$$

BEGIN

if $2 = 'date' then

RETURN query

SELECT t.typemed, m.name, m.price, di.date, di.shelf\_life

from medicaments m

inner join data\_info di on m.data\_info\_id = di.id

inner join types t on m.type\_id = t.id

where t.typemed = $1

order by di.date;

end if;

if $2 = 'alphabet' then

RETURN query

SELECT t.typemed, m.name, m.price, di.date, di.shelf\_life

from medicaments m

inner join data\_info di on m.data\_info\_id = di.id

inner join types t on m.type\_id = t.id

where t.typemed = $1

order by  m.name;

end if;

if $2 = 'shelf' then

RETURN query

SELECT t.typemed, m.name, m.price, di.date, di.shelf\_life

from medicaments m

inner join data\_info di on m.data\_info\_id = di.id

inner join types t on m.type\_id = t.id

where t.typemed = $1

order by di.shelf\_life;

end if;

if $2 = 'price' then

RETURN query

SELECT t.typemed, m.name, m.price, di.date, di.shelf\_life

from medicaments m

inner join data\_info di on m.data\_info\_id = di.id

inner join types t on m.type\_id = t.id

where t.typemed = $1

order by m.price;

end if;

END;

$$ LANGUAGE plpgsql;

Результат выполнения:

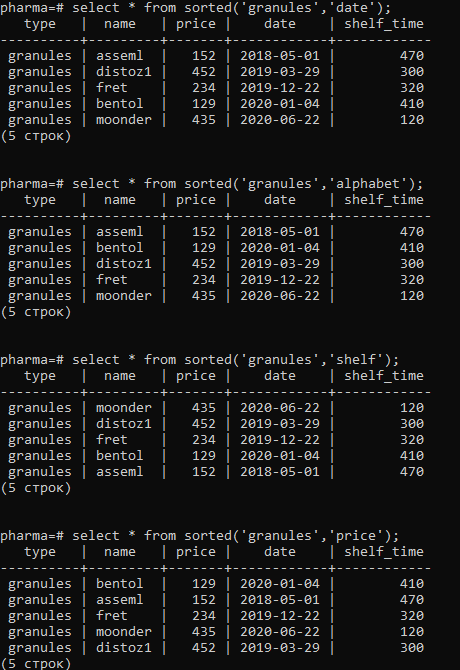


Рисунок 12. Результат выполнения функции

**Функция 3.** Найти самое дорогое лекарство, самое дешевое, среднюю стоимость для каждого типа лекарств и в целом.

Реализация функции:

create or replace FUNCTION average\_price(text)

RETURNS table(

price NUMERIC

)

AS

$$

BEGIN

if $1 = 'all' then

RETURN query

select avg(m.price) as price

from medicaments m

else

RETURN query

select avg(m.price) as price

from medicaments m

inner join types t on m.type\_id = t.id

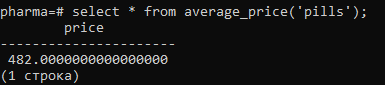
where t.typemed = $1;

end if;

END;

$$ LANGUAGE plpgsql;

Пример выполнения:



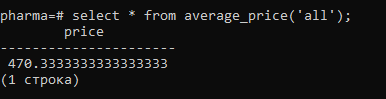


Рисунок 13. Результат выполнения функции

**Функция 4.** Найти самое дорогое лекарство, самое дешевое, среднюю стоимость для каждого типа лекарств и в целом.

Реализация функции:

create or replace FUNCTION max\_min\_price(text)

RETURNS table

(

"Oname" text,

"Oprice" int

)

as

$$

BEGIN

if $1 = 'max' then

RETURN query

SELECT name, price

from medicaments where price = (select max(price) from medicaments);

end if;

if $1 = 'min' then

RETURN query

SELECT name, price

from medicaments where price = (select min(price) from medicaments);

end if;

end;

$$ language plpgsql;

Результат выполнения:

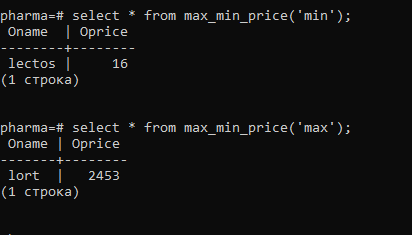


Рисунок 14. Результат выполнения функции

**Функция 5.** Для каждого типа заболевания найти лекарства с ценой свыше 20 р. (и любая другая сумма, т.е. предусмотреть ввод цены с клавиатуры).

Реализация функции:

create or replace FUNCTION overPrice(text,integer)

RETURNS table

(

P\_desiase text,

P\_NAME text,

P\_PRICE int

)

as

$$

BEGIN

return query

SELECT disease, name, price

from medicaments m

inner join pharma\_info pi on m.pharma\_info\_id = pi.id

 where price>$2 and disease = $1;

end;

$$ language plpgsql;

Результат выполнения:

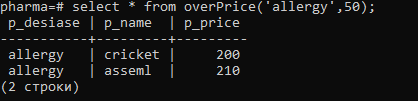


Рисунок 15. Результат выполнения функции

**Функция 6.** Найти все лекарства для заданного типа заболевания (выбор), чья стоимость попадает в заданные пределы (ввод интервала).

Реализация функции:

create or replace FUNCTION intervalPrice(text,integer,integer)

RETURNS table

(

P\_TYPE text,

P\_NAME text,

P\_PRICE int

)

as

$$

BEGIN

return query

SELECT typemed, name, price

from medicaments m

inner join types t on m.type\_id = t.id

 where price BETWEEN $2 and $3 and typemed = $1;

end;

$$ language plpgsql;

Результат выполнения:

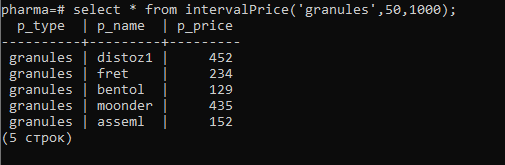


Рисунок 16. Результат выполнения функции

**Функция 7.** Найти долю дешевых лекарств (меньше заданного, ввод) от общего числа лекарств.

Реализация функции:

create or replace FUNCTION chipOfAll(integer)

RETURNS REAL

as

$$

DECLARE

total REAL;

chip REAL;

BEGIN

SELECT count(\*) into total

from  medicaments;

SELECT count(\*) into chip

from  medicaments where price < $1;

return chip/total;

end;

$$ language plpgsql;

Результат выполнения:

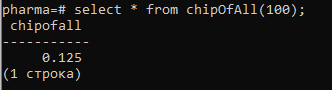


Рисунок 17. Результат выполнения функции

**Функция 8.** Определить количество лекарств заданного изготовителя (выбор), чья стоимость находится в заданных предела (ввод интервала).

Реализация функции:

create or replace FUNCTION numbByInterval(text,integer,integer)

RETURNS table

(

P\_count int,

P\_name text,

P\_Number int

)

as

$$

BEGIN

return query

SELECT count, cs.name, price

from  medicaments m

inner join data\_info dii on m.data\_info\_id = dii.id

inner join admission ad on m.admission\_id = ad.id

inner join companies cs on dii.company\_id = cs.id

 where m.price BETWEEN $2 and $3 and cs.name = $1;

end;

$$ language plpgsql;

Результат выполнения:

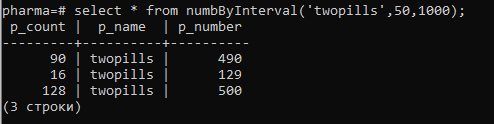


Рисунок 18. Результат выполнения функции

**Функция 9.** Определить количество лекарств заданного изготовителя (выбор), чья стоимость находится в заданных предела (ввод интервала).

Реализация функции:

create or replace FUNCTION getNum(text,integer,integer)

RETURNS integer

as

$$

DECLARE

summing INTEGER;

BEGIN

 SELECT sum(count) into summing

from  medicaments m

inner join data\_info dii on m.data\_info\_id = dii.id

inner join admission ad on m.admission\_id = ad.id

inner join companies cs on dii.company\_id = cs.id

 where m.price BETWEEN $2 and $3 and cs.name = $1;

return summing;

end;

$$ language plpgsql;

Пример выполнения:

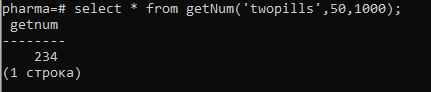


Рисунок 19. Результат выполнения функции

**Функция 10.** Найти долю лекарств заданного изготовителя (выбор) от общего числа лекарств.

Реализация функции:

create or replace FUNCTION getCompany\_desiase(text,text)

RETURNS table

(

P\_medicament text,

P\_company text,

P\_disease text

)

as

$$

BEGIN

return query

SELECT m.name, cs.name, pi.disease

from  medicaments m

inner join data\_info di on m.data\_info\_id = di.id

inner join companies cs on di.company\_id = cs.id

inner join pharma\_info pi on m.pharma\_info\_id = pi.id

 where cs.name = $1 and pi.disease = $2;

end;

$$ language plpgsql;

Результат выполнения:

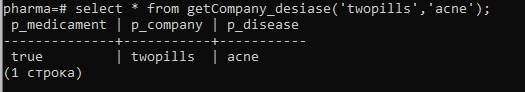


Рисунок 20. Результат выполнения функции

**Функция 11.** Найти долю лекарств заданного изготовителя (выбор) от общего числа лекарств.

Реализация функции:

create or replace FUNCTION partByCompany(text)

RETURNS REAL

as

$$

DECLARE

total REAL;

current REAL;

BEGIN

SELECT count(\*) into total

from  medicaments;

SELECT count(\*) into current

from  medicaments m

inner join data\_info di on m.data\_info\_id = di.id

inner join companies cs on di.company\_id = cs.id

where cs.name = $1;

return current/total;

end;

$$ language plpgsql;

Результат выполнения:

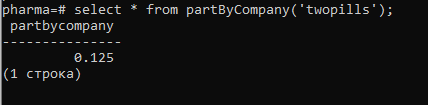


Рисунок 21. Результат выполнения функции

**Функция 12.** Определить долю регулярных поставок заданного типа (выбор типа).

Реализация функции:

create or replace FUNCTION partOfProviding(text)

RETURNS REAL

as

$$

DECLARE

total REAL;

disease REAL;

BEGIN

SELECT count(ad.adm\_date) into total

from  medicaments m

inner join admission ad on m.admission\_id = ad.id

;

SELECT count(ad.adm\_date) into disease

from  medicaments m

inner join admission ad on m.admission\_id = ad.id

inner join types t on m.type\_id = t.id

where t.typemed = $1;

return disease/total;

end;

$$ language plpgsql;

Результат выполнения:

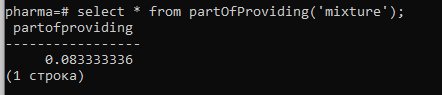


Рисунок 22. Результат выполнения функции

**Функция 13.** Для заданного поставщика (выбор) определить все лекарства, чей температурный режим хранения находится в заданных пределах (ввод интервала).

Реализация функции:

create or replace FUNCTION providerByTemp(text,integer,integer)

RETURNS table

(

P\_medicament text,

P\_provider text,

P\_temp int

)

as

$$

BEGIN

return query

SELECT m.name, pr.name, ty.temp

from  medicaments m

inner join data\_info di on m.data\_info\_id = di.id

inner join providers pr on di.provider\_id = pr.id

inner join types ty on m.type\_id = ty.id

 where ty.temp BETWEEN $2 and $3 and pr.name = $1 ;

end;

$$ language plpgsql;

Результат выполнения:

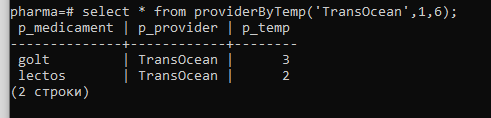


Рисунок 23. Результат выполнения функции

**Функция 14.** Найти все лекарства, поступившие из заданной страны (выбор страны), чья стоимость больше заданной (ввод стоимости).

Реализация функции:

create or replace FUNCTION medFromCountry(text,integer)

RETURNS table

(

P\_medicament text,

P\_country text,

P\_price int

)

as

$$

BEGIN

return query

SELECT m.name, co.country, m.price

from  medicaments m

inner join data\_info di on m.data\_info\_id = di.id

inner join companies cs on di.company\_id = cs.id

inner join countries co on cs.country\_id = co.id

 where co.country = $1 and m.price > $2 ;

end;

$$ language plpgsql;

Результат выполнения:

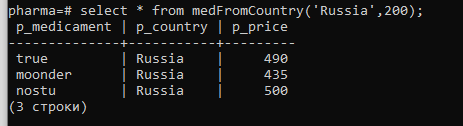


Рисунок 24. Результат выполнения функции

**Функция 15.** Найти долю дешевых лекарств (меньше заданного, ввод ограничения) от общего числа лекарств (для определенного возраста (например, для детей, для тех, кто старше 6 лет, 12 лет и т.д.)).

Реализация функции:

create or replace FUNCTION partOfChipMeds(integer,integer)

RETURNS REAL

as

$$

DECLARE

total REAL;

goodAge REAL;

BEGIN

SELECT count(t.age) into total

from  medicaments m

inner join types t on m.type\_id = t.id

;

SELECT count(t.age) into goodAge

from  medicaments m

inner join types t on m.type\_id = t.id

where t.age > $2 and m.price < $1;

return goodAge/total;

end;

$$ language plpgsql;

Результат выполнения:

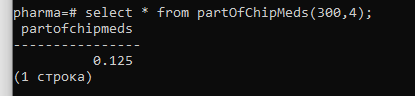


Рисунок 25. Результат выполнения функции

**Функция 16.** Найти самое популярное лекарство, которое покупают чаще всего для заданного сезона.

Реализация функции:

create or replace FUNCTION get\_most\_popular(date,date)

RETURNS table

(

P\_medicament text,

P\_date date,

P\_count int

)

as

$$

BEGIN

return query

SELECT m.name, s.sell\_date, s.count

from  medicaments m

inner join selling s on m.selling\_id = s.id

 where s.count = (select max(count) from selling where sell\_date BETWEEN $1 and $2) ;

end;

$$ language plpgsql;

Результат выполнения:

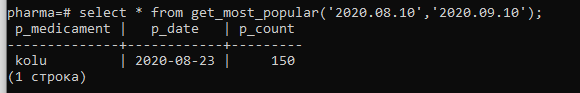


Рисунок 26. Результат выполнения функции

1. **Анализ результатов и выводы**

В результате выполнения курсовой работы была разработана база данных «Лекарства» с использованием СУБД PostgreSQL и языка SQL.

Были реализованы все необходимые типы запросов с использованием нескольких таблиц. В процессе выполнения работы были закреплены навыки и приемы разработки архитектуры баз данных, написания функций.

1. **Список литературы**
2. PostgreSQL [Электронный ресурс]: Documentation. – Режим доступа: <http://www.postgresql.org/docs/>. – загл. с экрана. - (Дата обращения: 20.03.2020).
3. Unetway [Электронный ресурс]: Documentation. – Режим доступа: <http://unetway.com/public/tutorial/sql-funkcii>– загл. с экрана. - (Дата обращения: 12.07.2020).