МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

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



**Курсовая работа**

**По дисциплине**

**«Базы данных»**

|  |  |
| --- | --- |
| Тема: | Разработкабазы данных «Снаряжение для горного туризма» |

|  |  |
| --- | --- |
| Студент | Павлюк А.С. (вариант 163) |

|  |  |
| --- | --- |
| Группа: | АВТ-008 |

|  |  |
| --- | --- |
| Преподаватель: | Трошина Галина Васильевна |

Новосибирск, 2022

Оглавление

[Задание 3](#_Toc119182063)

[Введение 5](#_Toc119182064)

[Графическое представление связей между таблицами 6](#_Toc119182065)

[Структура таблиц базы данных 6](#_Toc119182066)

[Содержание таблиц базы данных 7](#_Toc119182067)

[Руководство пользователя 10](#_Toc119182068)

[Тексты запросов и примеры работы 11](#_Toc119182069)

[Анализ результатов и выводы 33](#_Toc119182070)

[Список использованных источников 34](#_Toc119182071)

# 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 снаряжения для горного туризма указать сведения о нем (наименование, дату выпуска, поставщик, цена, дата продажи и т.п.).
* Получить список, отсортированный: по дате выпуска, в алфавитном порядке по поставщику, по стоимости, по дате продажи.
* Найти самое дорогое снаряжение для горного туризма, самое дешевое, среднюю стоимость.
* Найти снаряжение для горного туризма с ценой в заданных пределах (предусмотреть ввод цены с клавиатуры).
* Найти все снаряжение для горного туризма заданного производителя (выбор производителя).
* Найти долю дешевого снаряжения для горного туризма (меньше заданного, ввод ограничения) от общего числа снаряжения для горного туризма.
* Найти все снаряжение для горного туризма с заданной датой выпуска (ввод даты).
* Найти все снаряжение для горного туризма, чья дата продажи находится в заданных пределах (ввод интервала) для заданного производителя (ввод производителя) и в целом.
* Найти долю снаряжения для горного туризма, проданного за определенный период (ввод периода) от общего времени продажи.
* Найти самое популярное снаряжение для горного туризма (продано наибольшее количество).
* Найти все снаряжение для горного туризма, поступившие от заданного поставщика (выбор поставщика), чья стоимость больше, чем средняя стоимость снаряжения для горного туризма, поступившего из заданной страны (выбор страны).
* Найти долю дорогого снаряжения для горного туризма (чья стоимость больше заданной стоимости), поступившего от заданного поставщика (выбор поставщика) и в целом.
* Найти среднюю стоимость снаряжения для горного туризма, проданного за определенный промежуток времени (ввод интервала).
* Найти все снаряжение для горного туризма, чья стоимость выше, чем средняя стоимость снаряжения для горного туризма заданного производителя (выбор производителя).
* Определить долю регулярных поставок снаряжения для горного туризма от общего числа снаряжения для горного туризма.
* Найти объем продаж снаряжения для горного туризма за месяц, за квартал, за год; за этот же период: найти среднюю цену, самое дорогое снаряжение для горного туризма, самое дешевое снаряжение для горного туризма.

# Введение

В качестве инструмента для проектирования базы данных выбрана одна из самых популярных СУБД PostgreSQL.

СУБД отличается высокой надёжностью и хорошей производительностью. PostgreSQL поддерживает транзакции (ACID). При этом система расширяемая — можно создавать свои типы данных и индексов, а также расширять поведение при помощи языков программирования.

В рамках выполнения курсовой работы необходимо разработать структуру базы данных «Снаряжение для горного туризма», реализовать типовые запросы с использованием нескольких таблиц, оценить полученные результаты.

Цель работы – обобщить и структурировать знания, полученные в рамках дисциплины «Базы данных», научиться применять современные технологии создания баз данных.

# Графическое представление связей между таблицами

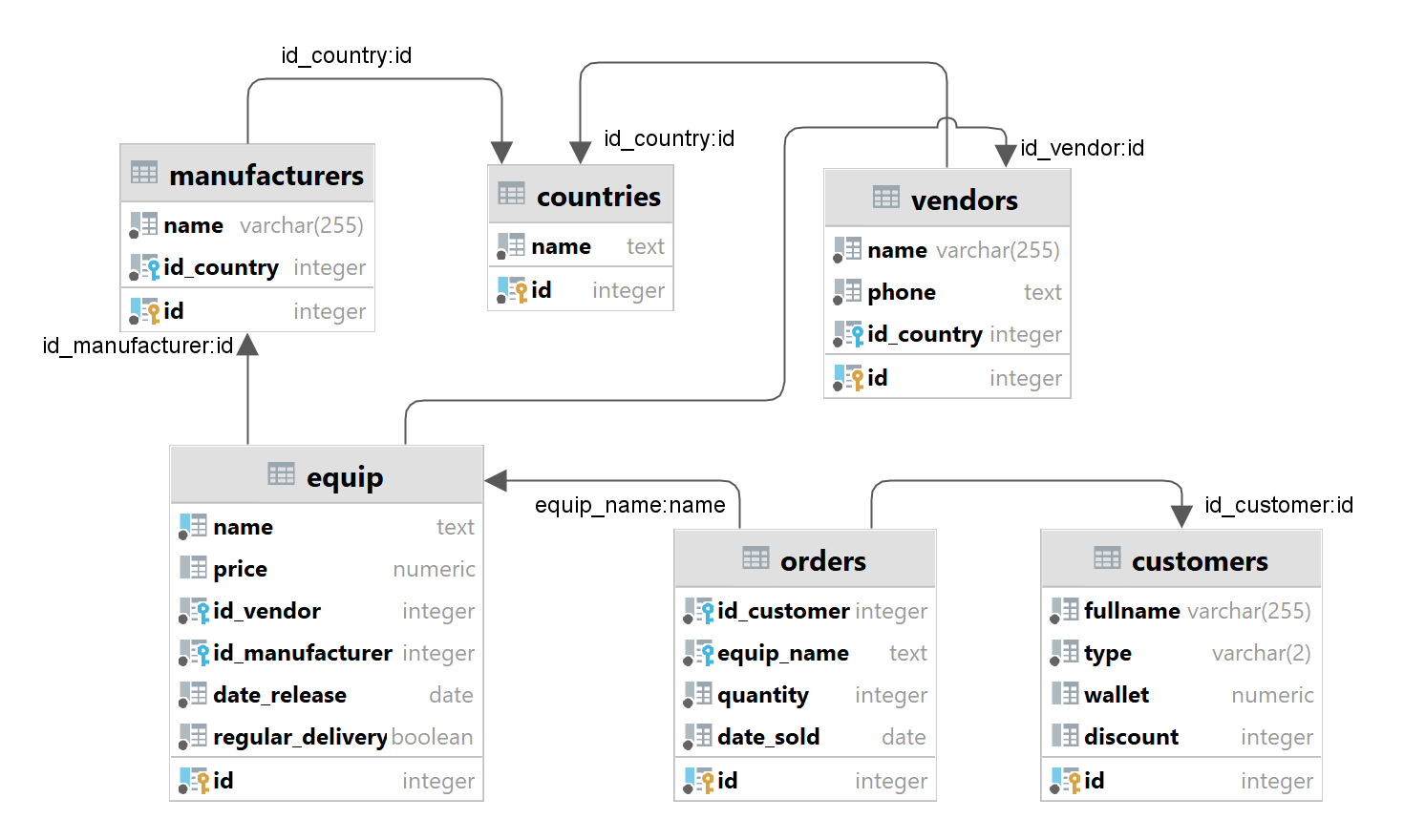


Рисунок 1. Структура базы данных "Снаряжение для горного туризма"

# Структура таблиц базы данных

Таблица-справочник **countries** – список стран:

* id – идентификатор страны
* name – название страны

Таблица-справочник **manufacturers** – список производителей:

* id – идентификатор производителя
* name – название компании производителя
* id\_country – идентификатор страны производителя

Таблица-справочник **vendors** – список поставщиков:

* id – идентификатор поставщика
* name – название компании поставщика
* phone – телефон компании поставщика
* id\_country – идентификатор страны поставщика

Таблица **customers** – информация о покупателях:

* id – идентификатор покупателя
* fullname – имя покупателя
* type – тип покупателя
* wallet – количество денег на счету покупателя
* discount – скидка для покупателя

Таблица **equip** – информация о снаряжении для горного туризма:

* id – идентификатор снаряжения
* price – цена снаряжения
* id\_vendor – идентификатор поставщика
* id\_manufacturer – идентификатор производителя
* date\_release – дата привоза снаряжения
* regular\_delivery – флаг регулярных доставок

Таблица **orders** – информация о заказах:

* id – идентификатор заказа
* id\_customer – идентификатор покупателя
* equip\_name – название снаяржения
* quantity – количество снаряжения
* date\_sold – дата продажи

# Содержание таблиц базы данных

* база данных tour\_equip

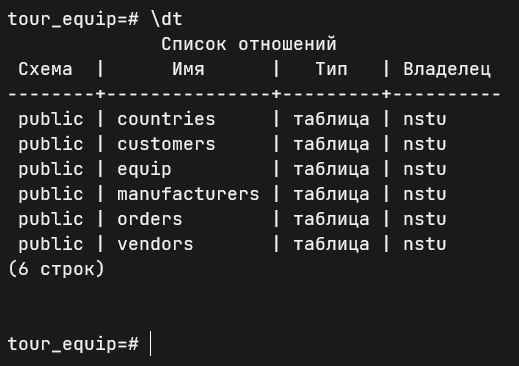


Рисунок 2. Список таблиц базы данных

* countries

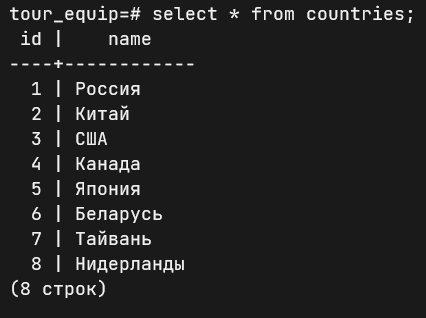


Рисунок 3. Содержание таблицы стран

* manufacturers

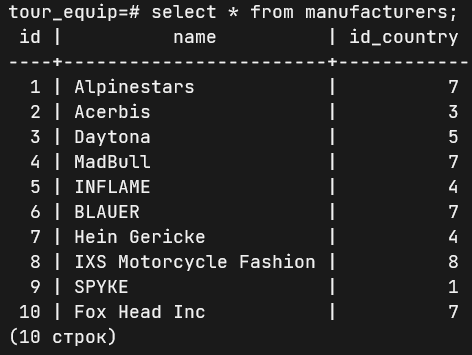


Рисунок 4. Содержание таблицы производителей

* vendors

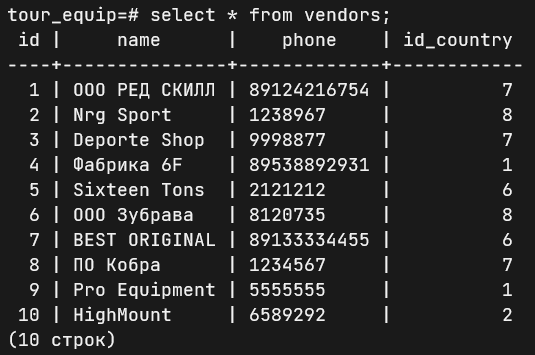


Рисунок 5. Содержание таблицы поставщиков

* customers

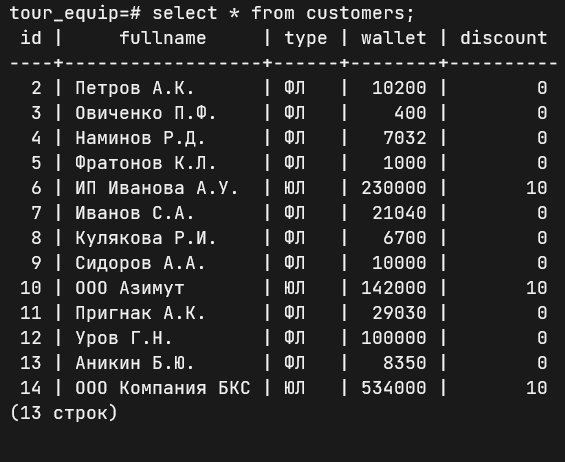


Рисунок 6. Содержание таблицы покупателей

* equip

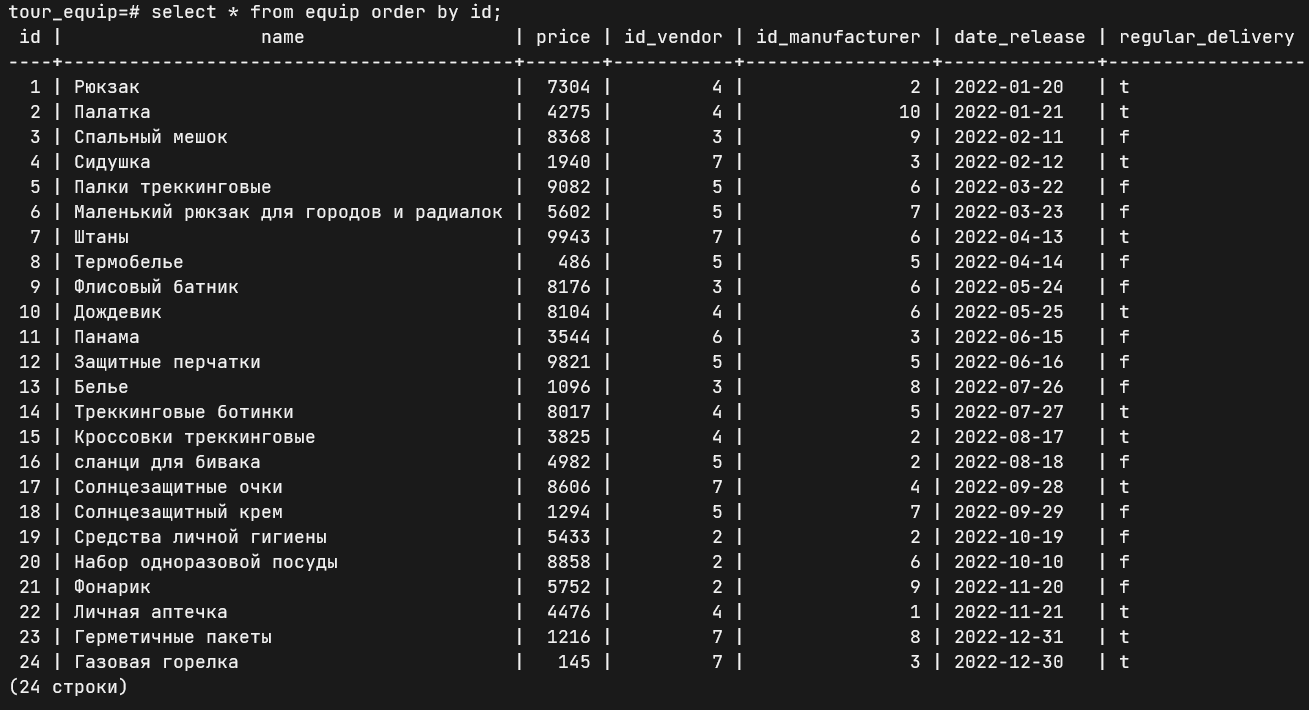


Рисунок 7. Содержание таблицы снаряжения

* orders



Рисунок 8. Содержание таблицы заказов

# Руководство пользователя

Таблица 1. Список функций

|  |  |  |
| --- | --- | --- |
| Функция | Входные параметры | Результат |
| get\_data() | - | Таблица со сведениями о снаряжении |
| sort\_date\_release() | - | Таблица всех товаров, отсортированных по дате выпуска |
| sort\_vendor() | - | Таблица всех товаров, отсортированных по названию поставщика |
| sort\_price() | - | Таблица всех товаров, отсортированных по стоимости |
| sort\_date\_sold() | - | Таблица всех товаров, отсортированных по дате продажи |
| max\_price() | - | Информация о самом дорогом снаряжении |
| min\_price() | - | Информация о самом дешевом снаряжении |
| avg\_price() | - | Средняя стоимость снаряжения |
| between\_price(numeric, numeric) | Нижняя и верхняя граница цены | Таблица снаряжения с ценой в заданных пределах |
| get\_by\_vendor(varchar) | Производитель | Таблица снаряжения заданного производителя |
| get\_less\_than(numeric) | Заданная стоимость | Таблица снаряжения, чья стоимость меньше заданной |
| get\_by\_release(date) | Дата выпуска | Таблица снаряжения с заданной датой выпуска |
| by\_date(date, date) | Нижняя и верхняя граница даты продажи | Таблица снаряжения, проданного за определенный период |
| by\_interval(interval) | Интервал | Таблица снаряжения, чья дата продажи находится в заданном интервале в целом |
| by\_date\_and\_manuf(interval, varchar) | Интервал, производитель | Таблица снаряжения заданного производителя, чья дата продажи находится в заданном интервале |
| most\_popular() | - | Самое популярное снаряжение (больше всего продано) |
| by\_vendor\_with\_price(varchar, text) | Поставщик, название страны | Таблица снаряжения, поступившее от заданного поставщика, чья стоимость больше, чем средняя стоимость снаряжения, поступившего из заданной страны |
| greater\_than(numeric) | Заданная стоимость | Таблица снаряжения, чья стоимость больше заданной |
| greater\_than(numeric, varchar) | Заданная стоимость, поставщик | Таблица снаряжения, поступившего от заданного поставщика, чья стоимость больше заданной |
| avg\_price\_by\_date(date, date) | Нижняя и верхняя граница даты продажи | Средняя стоимость снаряжения, проданного за определенный период времени |
| price\_greater\_than\_avg\_manufac  (text) | Производитель | Таблица снаряжения, чья стоимость выше, чем средняя стоимость снаряжения заданного производителя |
| is\_regular\_delivery() | - | Таблица снаряжения с регулярными поставками |
| count\_with\_price\_status(integer) | Число месяцев | Количество и средняя стоимость снаряжения, проданного за указанное число месяцев |
| equip\_with\_price\_by\_period  (integer, text) | Число месяцев, параметр (max, min) | Самое дорогое/дешевое снаряжение, проданное за указанное число месяцев |

# Тексты запросов и примеры работы

Для каждого снаряжения для горного туризма указать сведения о нем (наименование, дату выпуска, поставщик, цена, дата продажи и т.п.).

create or replace function get\_data() returns table (

id integer,

name text,

release date,

vendor varchar(255),

manufacturer varchar(255),

price numeric,

sold date

) as $$ BEGIN RETURN QUERY

select e.id,

e.name,

e.date\_release as release,

v.name as vendor,

m.name as manufacturer,

e.price,

o.date\_sold as sold

from equip e

join vendors v on v.id = e.id\_vendor

join manufacturers m on m.id = e.id\_manufacturer

join orders o on o.equip\_name = e.name;

END;

$$ language plpgsql;

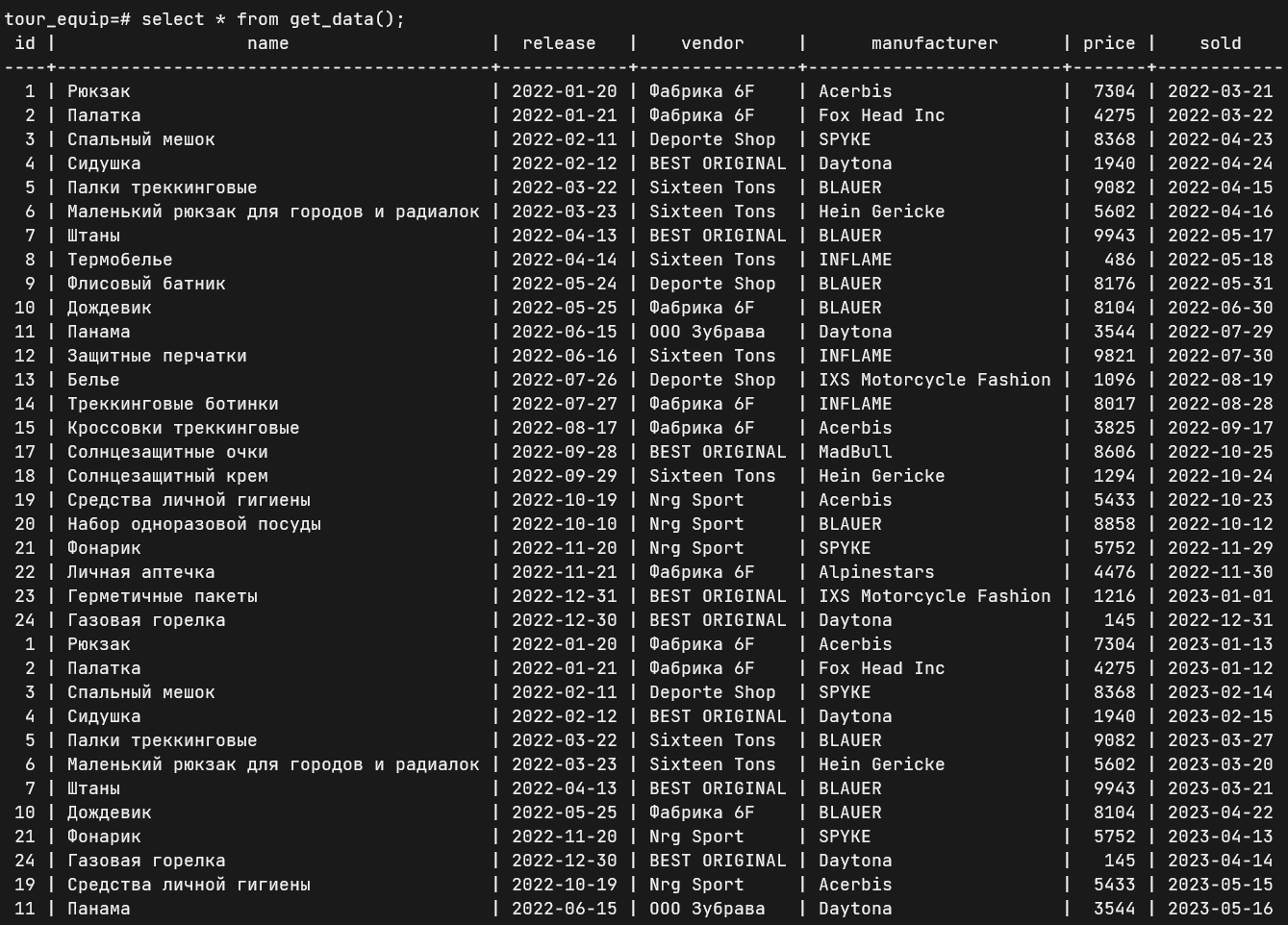


Рисунок 9. Демонстрация работы функции get\_data()

Получить список, отсортированный:

* по дате выпуска,
* в алфавитном порядке по поставщику,
* по стоимости,
* по дате продажи.
* по дате выпуска:

create or replace function sort\_date\_release() returns table (

id integer,

name text,

price numeric,

id\_vendor integer,

id\_manufacturer integer,

date\_release date

) as $$ begin return query

select \*

from equip

order by date\_release;

end;

$$ language plpgsql;

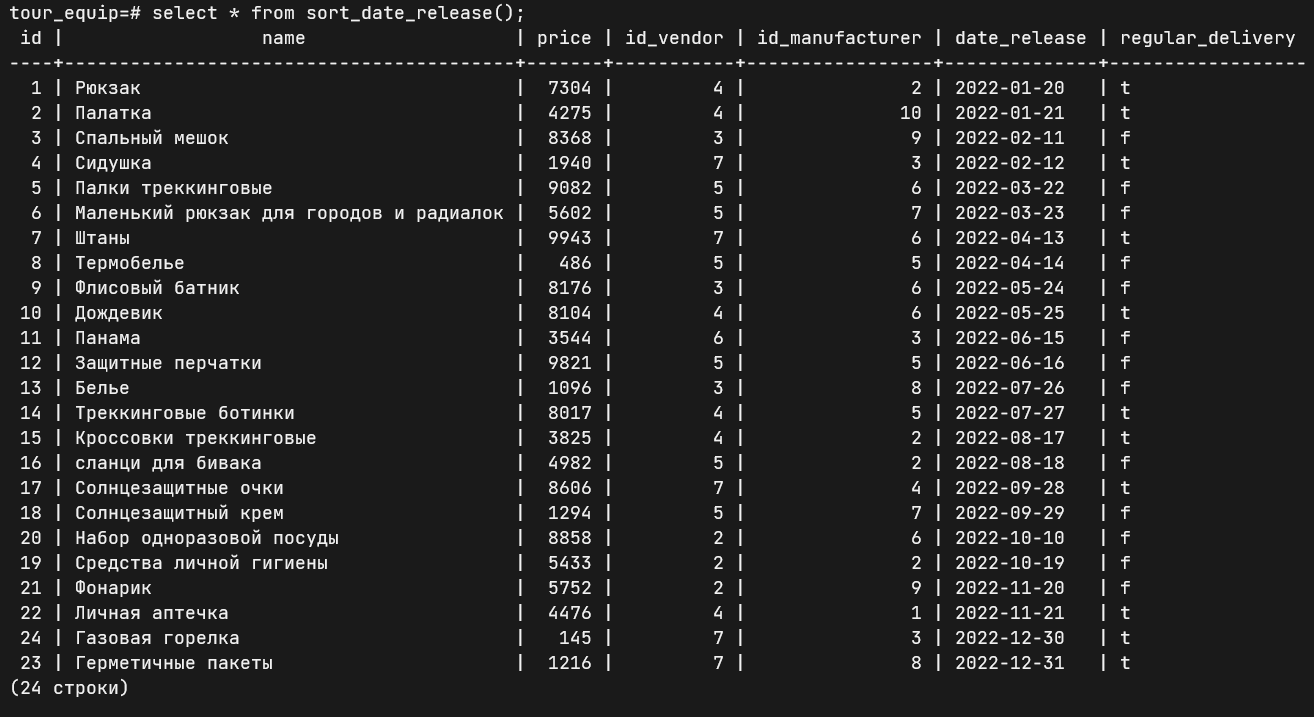


Рисунок 10. Демонстрация работы функции sort\_date\_release()

* в алфавитном порядке по поставщику:

create or replace function sort\_vendor() returns table (

id integer,

name text,

price numeric,

vendor varchar(255)

) as $$ begin return query

select e.id,

e.name,

e.price,

v.name

from equip e

join vendors v on e.id\_vendor = v.id

order by v.name;

end;

$$ language plpgsql;

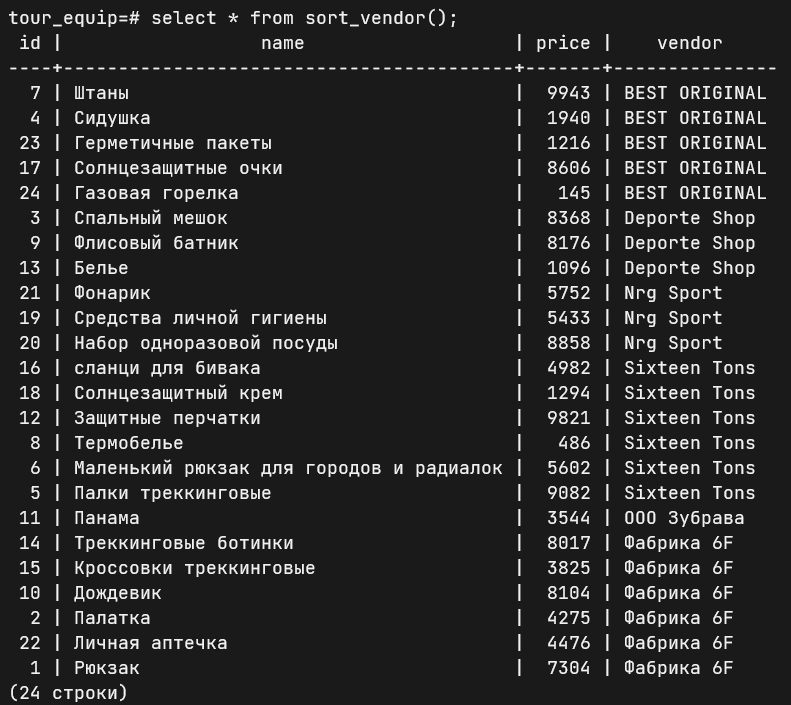


Рисунок 11. Демонстрация работы функции sort\_vendor()

* по стоимости:

create or replace function sort\_price() returns table (

id integer,

name text,

price numeric,

id\_vendor integer,

id\_manufacturer integer,

date\_release date

) as $$ begin return query

select \*

from equip

order by price;

end;

$$ language plpgsql;

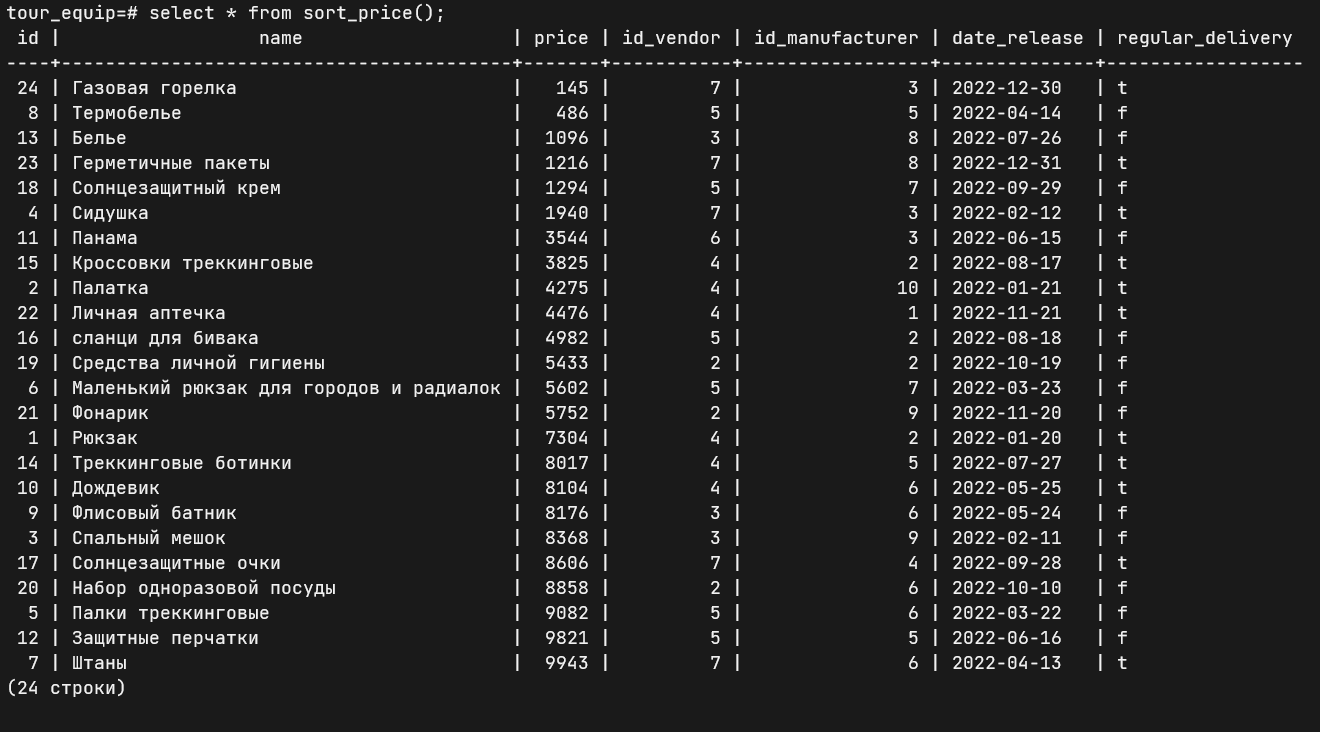


Рисунок 12. Демонстрация работы функции sort\_price()

* по дате продажи:

create or replace function sort\_date\_sold() returns table (

id integer,

name text,

price numeric,

release date

) as $$ begin return query

select e.id,

e.name,

e.price,

o.date\_sold

from equip e

join orders o on e.name = o.equip\_name

order by o.date\_sold;

end;

$$ language plpgsql;



Рисунок 13. Демонстрация работы функции sort\_date\_sold()

Найти самое дорогое снаряжение для горного туризма, самое дешевое, среднюю стоимость.

* самое дорогое:

create or replace function max\_price() returns table (

id integer,

name text,

price numeric,

id\_vendor integer,

id\_manufacturer integer,

date\_release date,

regular\_delivery boolean

) as $$ begin return query

select \*

from equip e

where e.price = (

select MAX(e.price)

from equip e

);

end;

$$ language plpgsql;

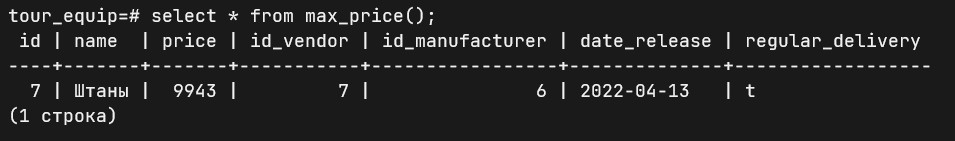


Рисунок 14. Демонстрация работы функции max\_price()

* самое дешевое:

create or replace function min\_price() returns table (

id integer,

name text,

price numeric,

id\_vendor integer,

id\_manufacturer integer,

date\_release date,

regular\_delivery boolean

) as $$ begin return query

select \*

from equip e

where e.price = (

select MIN(e.price)

from equip e

);

end;

$$ language plpgsql;

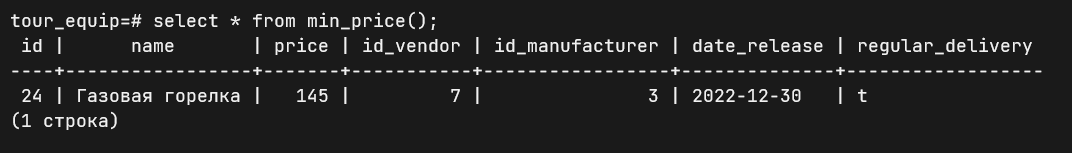


Рисунок 15. Демонстрация работы функции min\_price()

* среднюю стоимость:

create or replace function avg\_price() returns numeric language sql as $$

select AVG(e.price)

from equip e; $$;

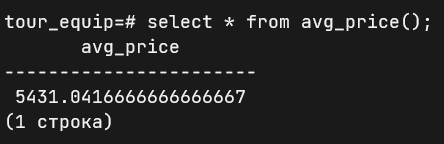


Рисунок 16. Демонстрация работы функции avg\_price()

Найти снаряжение для горного туризма с ценой в заданных пределах (предусмотреть ввод цены с клавиатуры).

create or replace function between\_price(min numeric, max numeric) returns table (

id integer,

name text,

price numeric,

id\_vendor integer,

id\_manufacturer integer,

date\_release date,

regular\_delivery boolean

) as $$ begin return query

select \*

from equip e

where e.price between min and max;

end;

$$ language plpgsql;

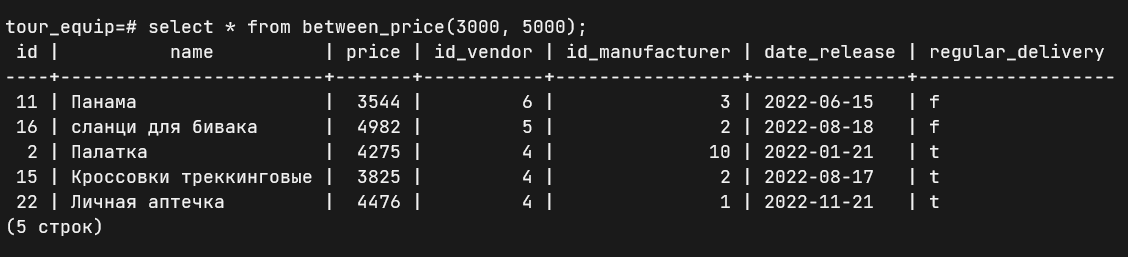


Рисунок 17. Демонстрация работы функции between\_price()

Найти все снаряжение для горного туризма заданного производителя (выбор производителя).

create or replace function get\_by\_vendor(\_vendor varchar(255)) returns table (

id integer,

name text,

price numeric,

vendor varchar(255)

) as $$ begin return query

select e.id,

e.name,

e.price,

v.name as vendor

from equip e

join vendors v on v.id = e.id\_vendor

where upper(v.name) = upper(\_vendor);

end;

$$ language plpgsql;

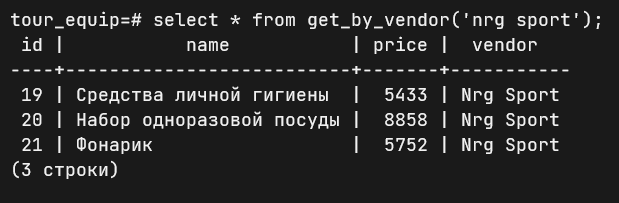


Рисунок 18. Демонстрация работы функции get\_by\_vendor()

Найти долю дешевого снаряжения для горного туризма (меньше заданного, ввод ограничения) от общего числа снаряжения для горного туризма.

create or replace function get\_less\_than(\_price numeric) returns table (

id integer,

name text,

price numeric,

id\_vendor integer,

id\_manufacturer integer,

date\_release date,

regular\_delivery boolean

) as $$ begin return query

select \*

from equip e

where e.price < \_price;

end;

$$ language plpgsql;

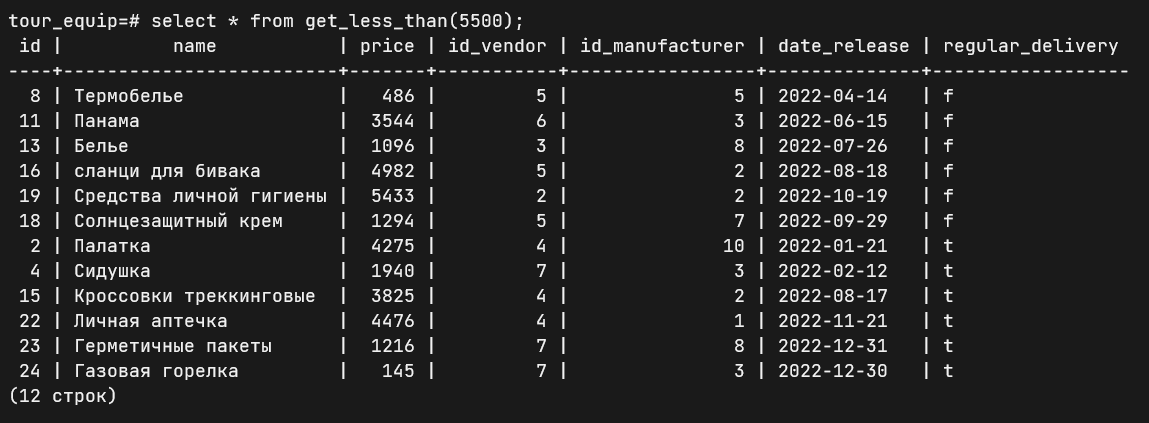


Рисунок 19. Демонстрация работы функции get\_less\_than()

Найти все снаряжение для горного туризма с заданной датой выпуска (ввод даты).

create or replace function get\_by\_release(\_release date) returns table (

id integer,

name text,

price numeric,

id\_vendor integer,

id\_manufacturer integer,

date\_release date,

regular\_delivery boolean

) as $$ begin return query

select \*

from equip e

where e.date\_release = \_release;

end;

$$ language plpgsql;

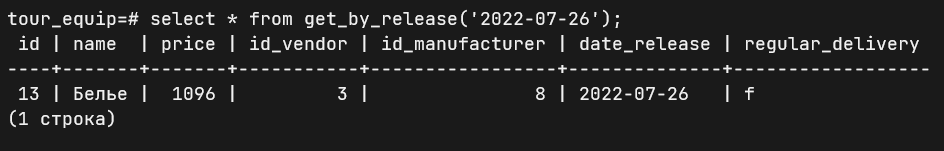


Рисунок 20. Демонстрация работы функции get\_by\_release()

Найти все снаряжение для горного туризма, чья дата продажи находится в заданных пределах (ввод интервала) для заданного производителя (ввод производителя) и в целом.

* для заданного производителя:

create or replace function by\_date\_and\_manuf(\_interval interval, \_manuf varchar(255)) returns table (

id integer,

name text,

date\_sold date,

manufacturer varchar(255)

) as $$ begin return query

select e.id,

e.name,

o.date\_sold,

m.name as manufacturer

from equip e

join orders o on e.name = o.equip\_name

join manufacturers m on e.id\_manufacturer = m.id

where now() - o.date\_sold <= \_interval

and now() - o.date\_sold >= '1 days'

and upper(m.name) = upper(\_manuf);

end;

$$ language plpgsql;

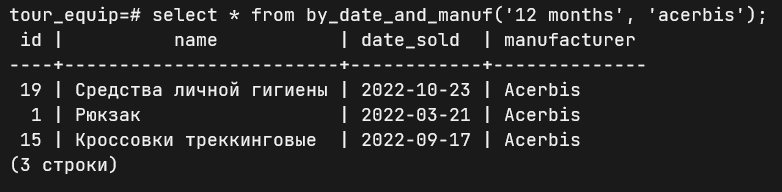


Рисунок 21. Демонстрация работы функции by\_date\_and\_manuf()

* в целом:

create or replace function by\_interval(\_interval interval) returns table (

id integer,

name text,

date\_sold date,

vendor varchar(255)

) as $$ begin return query

select e.id,

e.name,

o.date\_sold,

v.name as vendor

from equip e

join orders o on e.name = o.equip\_name

join vendors v on e.id\_vendor = v.id

where now() - o.date\_sold <= \_interval

and now() - o.date\_sold >= '1 days';

end;

$$ language plpgsql;

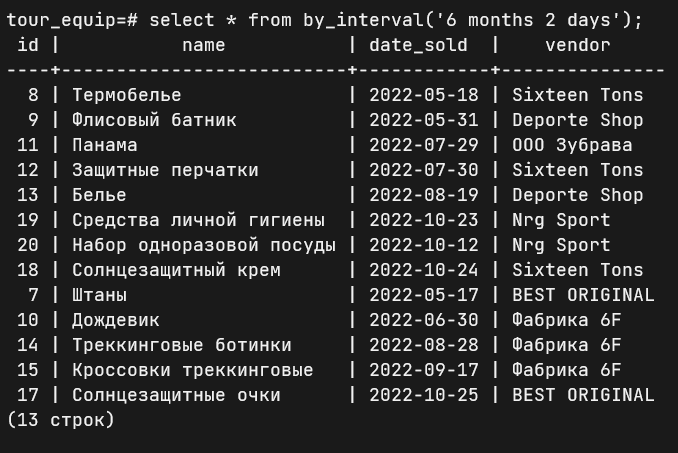


Рисунок 22. Демонстрация работы функции by\_interval()

Найти долю снаряжения для горного туризма, проданного за определенный период (ввод периода) от общего времени продажи.

create or replace function by\_date(\_start date, \_end date) returns table (

id integer,

name text,

date\_sold date

) as $$ begin return query

select e.id,

e.name,

o.date\_sold

from equip e

join orders o on e.name = o.equip\_name

where o.date\_sold >= \_start

and o.date\_sold <= \_end;

end;

$$ language plpgsql;

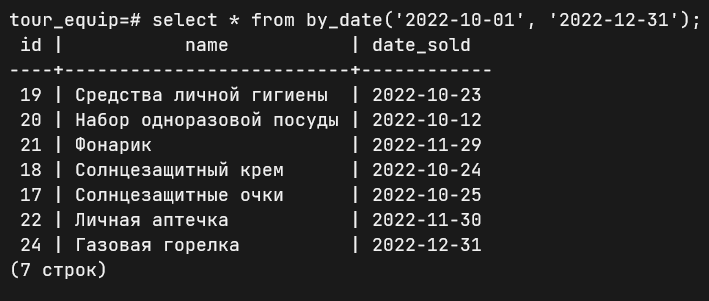


Рисунок 23. Демонстрация работы функции by\_date()

Найти самое популярное снаряжение для горного туризма (продано наибольшее количество).

create or replace function most\_popular() returns table (name text, quantity bigint) as $$ begin return query

select e.name,

sum(o.quantity) as quantity

from orders o

left join equip e on o.equip\_name = e.name

group by e.name

order by quantity desc

limit 1;

end;

$$ language plpgsql;

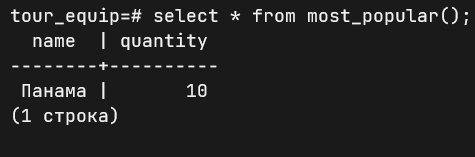


Рисунок 24. Демонстрация работы функции most\_popular()

Найти все снаряжение для горного туризма, поступившие от заданного поставщика (выбор поставщика), чья стоимость больше, чем средняя стоимость снаряжения для горного туризма, поступившего из заданной страны (выбор страны).

create or replace function by\_vendor\_with\_price(\_vendor varchar(255), \_country text) returns table (

id integer,

name text,

avg\_price numeric,

price numeric,

vendor varchar(255)

) as $$

declare \_avg numeric;

begin

select AVG(e.price) into \_avg

from equip e

join vendors v on v.id = e.id\_vendor

join countries c on c.id = v.id\_country

where upper(c.name) = upper(\_country);

return query

select e.id,

e.name,

round(\_avg, 2),

e.price,

v.name as vendor

from equip e

join vendors v on v.id = e.id\_vendor

where upper(v.name) = upper(\_vendor)

and e.price > \_avg;

end;

$$ language plpgsql;

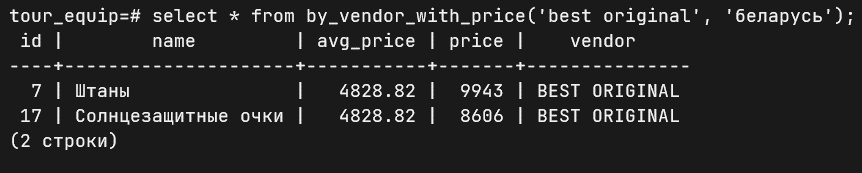


Рисунок 25. Демонстрация работы функции by\_vendor\_with\_price()

Найти долю дорогого снаряжения для горного туризма (чья стоимость больше заданной стоимости), поступившего от заданного поставщика (выбор поставщика) и в целом.

* от заданного поставщика

create or replace function greater\_than(\_price numeric, \_vendor varchar(255)) returns table (

id integer,

name text,

price numeric,

vendor varchar(255)

) as $$ begin return query

select e.id,

e.name,

e.price,

v.name

from equip e

join vendors v on v.id = e.id\_vendor

where e.price > \_price

and v.name = \_vendor;

end;

$$ language plpgsql;

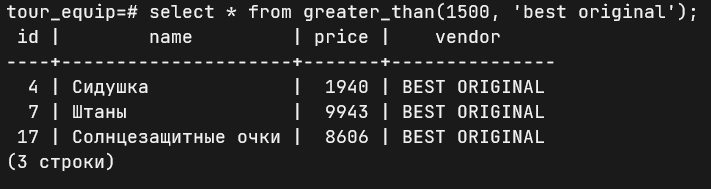


Рисунок 26. Демонстрация работы функции greater\_than()

* в целом

create or replace function greater\_than(\_price numeric) returns table (

id integer,

name text,

price numeric

) as $$ begin return query

select e.id,

e.name,

e.price

from equip e

where e.price > \_price;

end;

$$ language plpgsql;

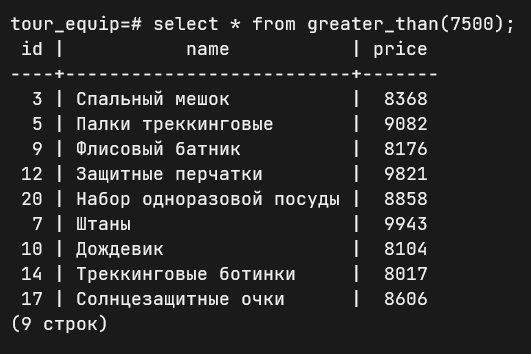


Рисунок 27. Демонстрация работы функции greater\_than()

Найти среднюю стоимость снаряжения для горного туризма, проданного за определенный промежуток времени (ввод интервала).

create or replace function avg\_price\_by\_date(\_start date, \_end date) returns real language sql as $$

select AVG(e.price)

from equip e

join orders o on e.name = o.equip\_name

where o.date\_sold >= \_start

and o.date\_sold <= \_end; $$;

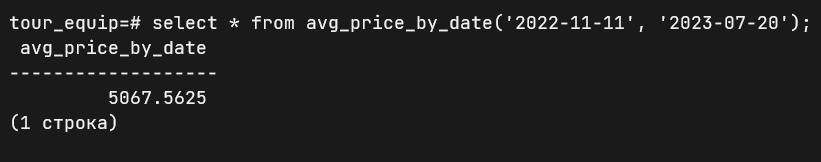


Рисунок 28. Демонстрация работы функции avg\_price\_by\_date()

Найти все снаряжение для горного туризма, чья стоимость выше, чем средняя стоимость снаряжения для горного туризма заданного производителя (выбор производителя).

create or replace function price\_greater\_than\_avg\_manufac(\_manufacturer varchar(255)) returns table (

id integer,

name text,

avg\_price numeric,

price numeric,

manufacturer varchar(255)

) as $$

declare \_avg numeric;

begin

select avg(e.price) into \_avg

from equip e

join manufacturers m on e.id\_manufacturer = m.id

where upper(m.name) = upper(\_manufacturer);

return query

select e.id,

e.name,

round(\_avg, 2),

e.price,

m.name as manufacturer

from equip e

join manufacturers m on e.id\_manufacturer = m.id

where e.price > \_avg;

end;

$$ language plpgsql;

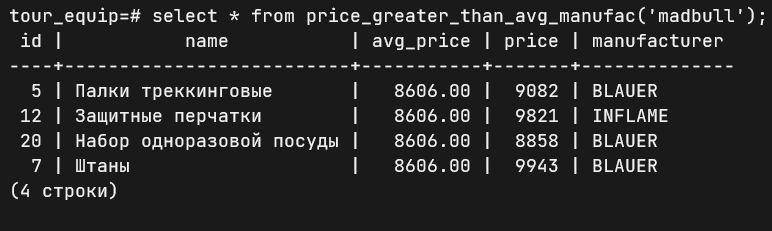


Рисунок 29. Демонстрация работы функции price\_greater\_than\_avg\_manufac()

Определить долю регулярных поставок снаряжения для горного туризма от общего числа снаряжения для горного туризма.

create or replace function is\_regular\_delivery() returns table (

id integer,

name text,

regular\_delivery boolean

) as $$ begin return query

select e.id,

e.name,

e.regular\_delivery

from equip e

where e.regular\_delivery = true;

end;

$$ language plpgsql;

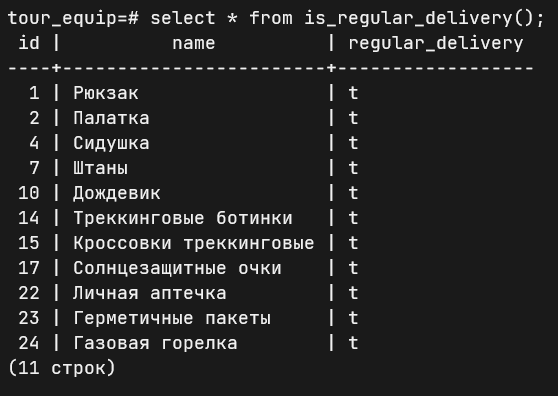


Рисунок 30. Демонстрация работы функции is\_regular\_delivery()

Найти объем продаж снаряжения для горного туризма за месяц, за квартал, за год; за этот же период: найти среднюю цену, самое дорогое снаряжение для горного туризма, самое дешевое снаряжение для горного туризма.

* объем продаж и средняя стоимость

create or replace function count\_with\_price\_status(\_period integer) returns table (

count bigint,

avg\_price numeric

) as $$ begin return query

select COUNT(e.id) as volume,

AVG(e.price) as avg\_price

from equip e

join orders o on e.name = o.equip\_name

where now() - o.date\_sold <= \_period \* interval '1 month'

and now() - o.date\_sold >= '1 days';

end;

$$ language plpgsql;

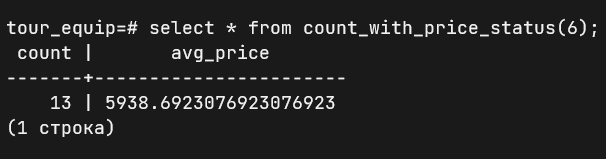


Рисунок 31. Демонстрация работы функции count\_with\_price\_status();

* самое дорогое и самое дешевое

create or replace function equip\_with\_price\_by\_period(\_period integer, \_property text) returns table (

id integer,

name text,

price numeric,

date\_sold date

) as $$

declare \_max numeric;

declare \_min numeric;

begin

select MAX(e.price) into \_max

from equip e

join orders o on e.name = o.equip\_name

where now() - o.date\_sold <= \_period \* interval '1 month'

and now() - o.date\_sold >= '1 days';

select MIN(e.price) into \_min

from equip e

join orders o on e.name = o.equip\_name

where now() - o.date\_sold <= \_period \* interval '1 month'

and now() - o.date\_sold >= '1 days';

if (upper(\_property) like upper('max')) then return query

select e.id,

e.name,

e.price,

o.date\_sold

from equip e

join orders o on e.name = o.equip\_name

where e.price = \_max

and now() - o.date\_sold <= \_period \* interval '1 month'

and now() - o.date\_sold >= '1 days';

end if;

if (upper(\_property) like upper('min')) then return query

select e.id,

e.name,

e.price,

o.date\_sold

from equip e

join orders o on e.name = o.equip\_name

where e.price = \_min

and now() - o.date\_sold <= \_period \* interval '1 month'

and now() - o.date\_sold >= '1 days';

end if;

end;

$$ language plpgsql;

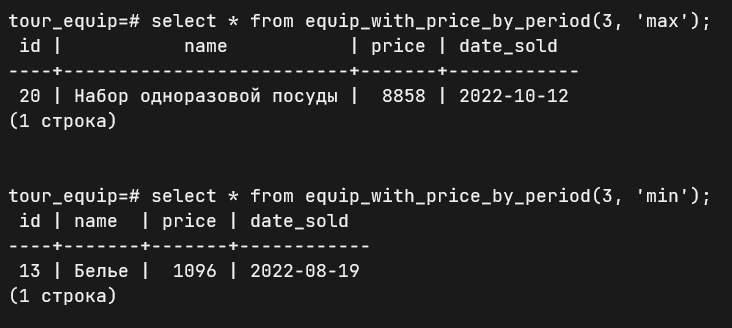


Рисунок 32. Демонстрация работы функции equip\_with\_price\_by\_period()

Создание ролей «оператор», «пользователь», «администратор».

«администратор» - CREATE ROLE admin WITH PASSWORD 'admin' LOGIN NOSUPERUSER CREATEROLE CREATEDB;

Для администратора предоставлена возможность изменять целостность любых таблиц.

GRANT SELECT, INSERT, UPDATE, DELETE ON countries, customers, equip, manufacturers, orders, vendors TO admin;

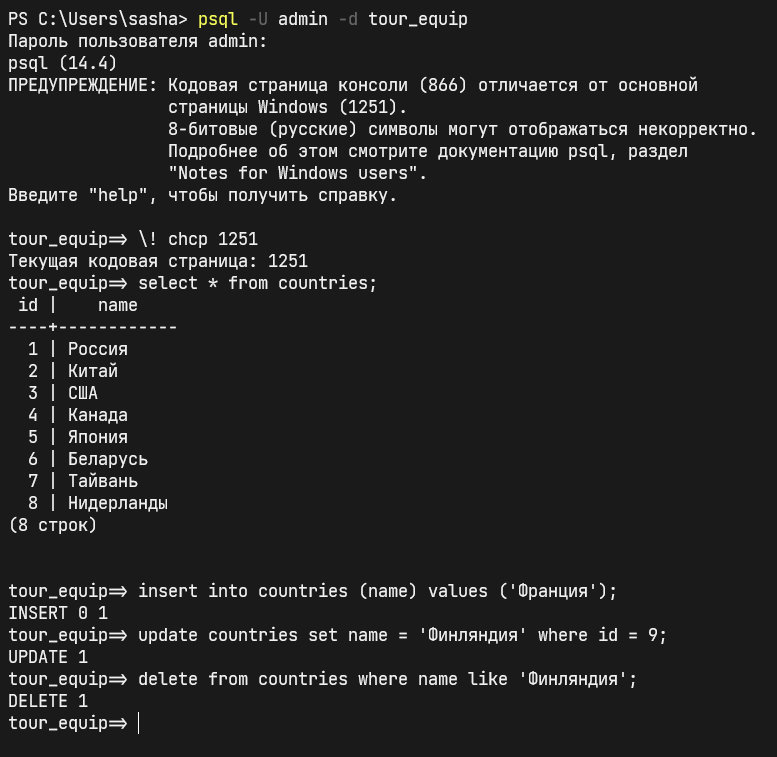


Рисунок 33. Демонстрация операций в роли админа

«оператор» - CREATE ROLE operator WITH PASSWORD 'operator' LOGIN;

Оператору запрещается удалять данные из таблиц. Разрешен просмотр, пополнение и обновление таблиц и таблиц-справочников.

GRANT SELECT, UPDATE, INSERT ON countries, customers, equip, manufacturers, orders, vendors TO operator;

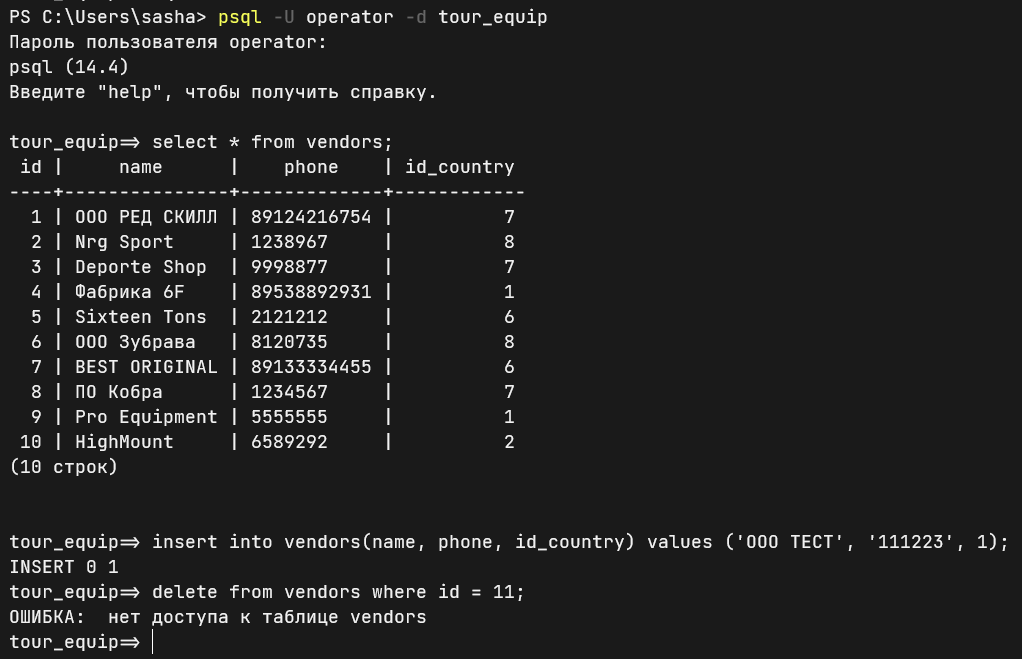


Рисунок 34. Демонстрация команд в роли оператора

«пользователь» - CREATE ROLE user\_db WITH PASSWORD 'user' LOGIN;

Пользователю базы данных разрешен просмотр, обновление, вставка и удаление данных из таблиц.

GRANT SELECT, INSERT, UPDATE, DELETE ON countries, customers, equip, manufacturers, orders, vendors TO user\_db;

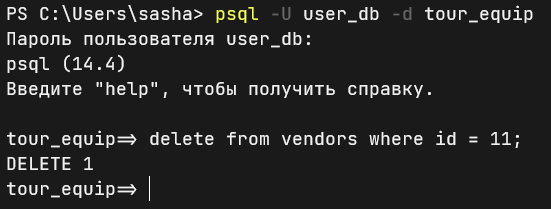


Рисунок 35. Демонстрация команд в роли пользователя

# Анализ результатов и выводы

В ходе выполнения работы была разработана структура базы данных «Снаряжение для горного туризма» с использованием СУБД PostgreSQL, реализованы типовые запросы с использованием нескольких таблиц, которые были превращены в функции на языке plpgsql для простоты воспроизведения. В базе данных присутствуют таблицы-справочники, а также таблицы, использующие справочники. Графически представлена связь между таблицами. Были предусмотрены роли – оператор, администратор, пользователь.

В результате работы были обобщены и структурированы знания, полученные в рамках дисциплины «Базы данных» и применены современные технологии создания баз данных.

# Список использованных источников

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: 01.10.2022)
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: 17.10.2022)