

База данных «Торговля»



Петров Михаил Иванович

201 группа

Содержание

1. Описание базы данных
2. Схема
3. Таблицы
4. Легкие запросы
5. Средние запросы
6. Сложные запросы

Описание базы данных

Объекты:

- a. Товары
- b. Производители
- c. Покупатели
- d. Позиции

Отношения:

1. Товар m:1 Покупатель
(Покупатель может купить несколько товаров)
2. Товар m:m Производитель
(У товара могут быть разные производители)
(У производителя могут быть разные товары)
3. Позиция 1:1 Товар
(Позиция на складе у товара одна и товар один у позиции на складе)

Cxema

positions			
	id	integer	
	position	varchar	
	name_loader	varchar	
Add field			

buyers			
	id	integer	
	name	varchar	
	surname	varchar	
	patronymic	varchar	
	age	integer	
	type	integer	
Add field			
















goods			
	id	integer	
	name	varchar	
	type	varchar	
	price	integer	
	position_id	integer	
	buyer_id	integer	
Add field			

goods_manufacturers_relations			
	good_id	integer	
	manufacturer_id	integer	
Add field			

manufacturers			
	id	integer	
	name	varchar	
	country	varchar	
	phone_number	integer	
Add field			










Таблицы sql

```
CREATE TABLE goods (  
    id int PRIMARY KEY DEFAULT nextval('goods_id_seq'),  
    name varchar(30) NOT NULL,  
    type varchar(30) NOT NULL,  
    price int NOT NULL,  
    position_id int UNIQUE,  
    buyer_id int NOT NULL  
);
```

goods 				
	id	integer		
	name	varchar		
	type	varchar		
	price	integer		
	position_id	integer		
	buyer_id	integer		
 Add field				















Таблицы sql

```
CREATE TABLE positions (  
    id int PRIMARY KEY DEFAULT nextval('positions_id_seq'),  
    position varchar(10) NOT NULL,  
    name_loader varchar(30) NOT NULL  
);
```

positions			
	id	integer	 
	position	varchar	 
	name_loader	varchar	 
 Add field			











Таблицы sql

```
CREATE TABLE buyers (  
    id int PRIMARY KEY DEFAULT nextval('buyers_id_seq'),  
    name varchar(30) NOT NULL,  
    surname varchar(30) NOT NULL,  
    patronymic varchar(30),  
    age int NOT NULL,  
    type varchar(30) NOT NULL  
);
```

buyers			
	id	integer	 
	name	varchar	 
	surname	varchar	 
	patronymic	varchar	 
	age	integer	 
	type	integer	 
 Add field			








Таблицы sql

```
CREATE TABLE manufacturers (  
    id int PRIMARY KEY DEFAULT  
    nextval('manufacturers_id_seq'),  
    name varchar(30) UNIQUE,  
    country varchar(30) NOT NULL,  
    phone_number bigint UNIQUE  
);
```

manufacturers			
	id	integer	 
	name	varchar	 
	country	varchar	 
	phone_number	integer	 
 Add field			

Таблицы sql

```
CREATE TABLE goods_manufacturers_relations (  
    good_id int,  
    manufacturer_id int,  
    PRIMARY KEY(good_id, manufacturer_id)  
);
```

goods_manufacturers_relations			
	good_id	integer	 
	manufacturer_id	integer	 
 Add field			

Лёгкие запросы

1. Поиск товара по имени, просмотр его позиции на складе.

```
SELECT s.name, s.position_id FROM goods AS s WHERE lower(s.name) = __NAME__;
```

Допустимые параметры [__NAME__]:

- 'pen'
- 'bread'
- 'apple'

Оптимизация: Был добавлен индекс goods_name_idx_lower для осуществления фильтрации по нижнему регистру поля name.

```
CREATE INDEX goods_name_idx_lower ON goods((lower(name)));
```

Лёгкие запросы

2. Поиск количества покупателей по типу.

```
SELECT s.type, count(*) FROM buyers s GROUP BY s.type ORDER BY s.type;
```

Допустимые параметры: -

Оптимизация: Был добавлен индекс buyer_type_idx для осуществления фильтрации по типу.

```
CREATE INDEX buyer_type_idx ON buyers(type);
```

Лёгкие запросы

3. Поиск покупателя по Имени и Фамилии для определения его типа и возраста.

```
SELECT s.name, s.surname, s.patronymic, s.age, s.type FROM buyers AS s WHERE lower(s.name) = __NAME__ AND lower(s.surname) = __SURNAME__;
```

Допустимые параметры [__NAME__] AND [__SURNAME__]:

- 'podolskii' AND 'roman'
- 'filippov' AND 'roman'
- 'grey' AND 'alexandr'

Оптимизация: Был добавлен индекс buyers_name_surname_idx_lower для осуществления фильтрации по Имени и Фамилии.

```
CREATE INDEX buyers_name_surname_idx_lower ON buyers((lower(name)),(lower(surname)));
```

Лёгкие запросы

4. Просмотр самых дорогих товаров магазина.

```
SELECT name, type, price FROM goods ORDER BY price DESC LIMIT 10;
```

Допустимые параметры: -

Оптимизация: Был добавлен индекс goods_price_idx для осуществления фильтрации по цене.

```
CREATE INDEX goods_price_idx ON goods(price);
```

Средние запросы

1. Поиск Имени товаров, которые купил конкретный покупатель.

```
SELECT g.name, b.name, b.surname FROM goods g JOIN buyers b ON (g.buyer_id = b.id) WHERE  
lower(b.name) = __NAME__ AND lower(b.surname) = __SURNAME__;
```

Допустимые параметры [__NAME__] AND [__SURNAME__]:

- 'podolskii' AND 'roman'
- 'filippov' AND 'roman'
- 'grey' AND 'alexandr'

Оптимизация: Был добавлен индекс goods_buyers_name_idx_lower для осуществления фильтрации по нижнему регистру Имени и Фамилии покупателя.

```
CREATE INDEX goods_buyers_name_idx_lower ON buyers((lower(name)),(lower(surname)));
```

Средние запросы

2. Просмотр Имени, Фамилии покупателей которые покупают самые дорогие товары в магазине чтобы предоставить им скидку.

```
SELECT g.name, g.price, b.name, b.surname FROM goods g JOIN buyers b ON (g.buyer_id = b.id)  
ORDER BY price DESC LIMIT 10;
```

Допустимые параметры: -

Оптимизация: Был добавлен индекс goods_buyers_price_idx для осуществления фильтрации по цене.

```
CREATE INDEX goods_buyers_price_idx ON goods(price);
```

Средние запросы

3. Поиск Имени товаров на складе по его Позиции.

```
SELECT g.name FROM goods g JOIN positions p ON (g.position_id = p.id) WHERE p.position =  
__POSITION__;
```

Допустимые параметры [__POSITION__]:

- 'A1B1'
- 'A9B6'
- 'A1B4'

Оптимизация: Был добавлен индекс goods_positions_name_idx_lower для осуществления фильтрации по нижнему регистру поля position.

```
CREATE INDEX goods_positions_name_idx ON positions(position);
```


Сложные запросы

1. Поиск товаров которые купил конкретный покупатель.

```
WITH buyer_goods AS (SELECT g.name, g.price, g.id FROM goods g JOIN buyers b ON (g.buyer_id = b.id) WHERE lower(b.name) = __NAME__ AND lower(b.surname) = __SURNAME__) SELECT bg.name, bg.price, m.country FROM buyer_goods bg JOIN goods_manufacturers_relations rel ON (bg.id = rel.good_id) JOIN manufacturers m ON (rel.manufacturer_id = m.id);
```

Допустимые параметры [__NAME__] AND [__SURNAME__]:

- 'podolskii' AND 'roman'
- 'filippov' AND 'roman'
- 'grey' AND 'alexandr'

Оптимизация: Был добавлен индекс buyers_name_surname_idx_lower для осуществления фильтрации по нижнему регистру Имени и Фамилии покупателя.

```
CREATE INDEX buyers_name_surname_idx_lower ON buyers((lower(name)),(lower(surname)));
```

Сложные запросы

2. Поиск информации о товаре по его Позиции на складе.

```
WITH position_good AS (SELECT p.id, p.position, p.name_loader FROM positions p JOIN goods g ON (p.id = g.position_id) WHERE p.position = __POSITION__) SELECT pg.position, pg.name_loader, m.name, m.country, m.phone_number FROM position_good pg JOIN goods_manufacturers_relations rel ON (pg.id = rel.good_id) JOIN manufacturers m ON (rel.manufacturer_id = m.id);
```

Допустимые параметры [__POSITION__]:

- 'A1B1'
- 'A3B3'
- 'A1B4'

Оптимизация: Был добавлен индекс goods_positions_name_idx для осуществления фильтрации поля position.

```
CREATE INDEX goods_positions_name_idx ON positions(position);
```

Сложные запросы

3. Просмотр людей которые больше всего потратили денег в магазине кроме покупателя 'Петрова' - это хозяин магазина.

```
WITH all_buyers AS (SELECT g.buyer_id, g.price, b.name, b.surname, b.patronymic, b.age, b.type FROM goods g JOIN buyers b ON (g.buyer_id = b.id) WHERE b.name!='Petrov') SELECT ab.name, ab.surname, ab.patronymic, ab.age, ab.type, sum(ab.price) FROM all_buyers ab GROUP BY ab.name, ab.surname, ab.buyer_id, ab.patronymic, ab.age, ab.type ORDER BY ab.buyer_id;
```

Допустимые параметры: -

Оптимизация: Был добавлен индекс buyers_name_price_idx для осуществления фильтрации поля position.

```
CREATE INDEX buyers_name_price_idx ON buyers(name);
```

3/2

ОЦЕНКА ПРЕЗЕНТАЦИИ

ТРЕБУЮ ОСКАРА