**3 Создание запросов на языке SQL**

**3.1 Основные команды SQL для создания базы данных**

Для работы с данным программным продуктом необходимо реализовать базу данных, её таблицы, представления, хранимые процедуры и триггеры.

В SQL представлены специальные команды, зарезервированы специальные имена использующиеся для создания базы данных, а также её таблиц.

Запросы, реализующие создание базы данных, а также ее таблиц представлены в листингах 3.1-3.6.

**Листинг 3.1 – Создание базы данных**

CREATE DATABASE BOOM;

**Листинг 3.2 – Создание таблицы Клиенты**

CREATE table Клиенты(

id\_Kl int AUTO\_INCREMENT Primary Key,

Login varchar(40) NOT NULL,

Password varchar(40) NOT NULL,

Admin bit not null default 0,

Surname varchar(40) NOT NULL,

Name varchar(40) NOT NULL,

Lastname varchar(40) NOT NULL,

Adress varchar(40) NOT NULL,

Indexx int NOT NULL,

Numberr varchar(40),

Data\_r date NOT NULL,

CONSTRAINT Numbar check (Numberr LIKE '+375([0-9][0-9])[0-9][0-9][0-9]-[0-9][0-9]-[0-9][0-9]'));

**Листинг 3.3 – Создание таблицы Тип\_товара**

CREATE table Тип\_товара(

Id\_type int AUTO\_INCREMENT Primary Key,

Type varchar(40));

**Листинг 3.4 – Создание таблицы Товары**

CREATE table Товары(

Id\_tov int AUTO\_INCREMENT Primary Key,

Name varchar(40) NOT NULL,

Id\_type int not null ,

Price int NOT NULL,

Kol\_vo int NOT NULL check (Kol\_vo > 0),

Marka varchar(40) NOT NULL,

Desk varchar(255) Not null,

Imgg varchar(60) not null,

CONSTRAINT fk\_type FOREIGN KEY (Id\_type) REFERENCES Тип\_товара(Id\_type) on delete cascade);

**Листинг 3.5 – Создание таблицы Заказы**

CREATE table Заказы(

Id\_Ord int AUTO\_INCREMENT Primary Key,

Id\_Kl int not null,

Id\_dost int not null,

Name varchar(40) not null,

Surname varchar(40) not null,

Lastname varchar(40) not null,

Adress varchar(40) not null,

CONSTRAINT fk\_Ord foreign key (Id\_Kl) REFERENCES Клиенты(Id\_Kl) ON DELETE CASCADE,

CONSTRAINT fk\_Ord1 foreign key (Id\_dost) REFERENCES Доставка(Id\_dost) ON DELETE CASCADE);

**Листинг 3.6 – Создание таблицы Корзина**

CREATE table Корзина(

Id\_Kor int AUTO\_INCREMENT Primary Key,

Id\_Ord int NOT NULL,

Id\_Tov int NOT NULL,

Price int not null check (Price > 0.5),

Kol\_vo int not null check (Kol\_vo > 0.5),

CONSTRAINT fk\_Kor FOREIGN KEY (Id\_Ord) references Заказы(Id\_Ord) ON DELETE CASCADE,

CONSTRAINT fk\_Kor1 foreign key (Id\_Tov) REFERENCES Товары(Id\_Tov) ON DELETE CASCADE);

**3.2 Создание представлений и функций**

Представление – это виртуальная таблица, содержимое которой определяется запросом, который динамически извлекает используемые данные.

Создание представлений открывает ряд преимуществ:

* они упрощают комплексные SQL-операции,
* они защищают данные, так как представления могут дать доступ к части таблицы, а не ко всей таблице,
* представления также позволяют возвращать отформатированные значения из таблиц в нужной и удобной форме.

Реализация представлений базы данных указана в листингах 3.7-3.8.

**Листинг 3.7 – Создание представления «tovar»**

CREATE VIEW tovar AS

SELECT Id\_Tov, Товары.Name AS Name, Тип\_товара.Type as Type, Price, Kol\_vo, Marka, Desk, Imgg FROM Товары INNER JOIN Тип\_товара ON Тип\_товара.Id\_type=Товары.Id\_type;

В данном представлении хранится информация о товарах магазина, где каждый товар связан с категорией, к которой он принадлежит. Это представление будет использовано для вывода каталога товаров.

**Листинг 3.8 – Создание представления «zak»**

CREATE VIEW zak AS

SELECT Id\_Ord, Клиенты.Name as Kname, Клиенты.Adress as adress, Клиенты.Nubmerr as Numberr, Товары.Name as Tname, Kol\_vo, oplata FROM Заказы INNER JOIN Клиенты ON Клиенты.Id\_Kl = Заказы.Id\_Kl inner join Товары on Товары.Id\_Tov=Заказы.Id\_Tov;

Данное представление хранит информацию о заказах. Оно будет использовано для вывода информации о заказах в удобной для администратора форме.

**3.3 Создание триггеров и хранимых процедур**

Триггер – это откомпилированная SQL-процедура, исполнение которой обусловлено наступлением определенных событий внутри реляционной базы данных. В отличие от процедур, триггеры нельзя вызывать из приложения, а также передавать им параметры.

Реализация триггера представлена в листинге 3.9.

**Листинг 3.9 – Создание триггера «tovZak»**

CREATE TRIGGER tovZak AFTER INSERT ON Заказы FOR EACH ROW

BEGIN UPDATE Товары SET

Kol\_vo = Kol\_vo - NEW.Kol\_vo WHERE Товары.Id\_tov = NEW.Id\_Tov END;

Данный триггер уменьшит количество товара, в случае его заказа на заказанное количество.

Хранимые процедуры представляют набор инструкций, которые выполняются как единое целое. Тем самым хранимые процедуры позволяют упростить комплексные операции и вынести их в единый объект.

Реализация хранимой процедуры, отвечающей за добавление, данных в базе представлена в листинге 3.10.

**Листинг 3.10 – Создание хранимой процедуры «Tov»**

CREATE PROCEDURE `Tov`(IN `n` VARCHAR(50), IN `id` INT(11), IN `p` INT(11), IN `k` INT(11), IN `m` VARCHAR(20), IN `d` VARCHAR(200), IN `i` VARCHAR(60)) BEGIN Insert Into `Товары`(Name, Id\_type, Price, Kol\_vo, Marka, Desk, Imgg) Values (n,id,p,k,m,d,i); END

Процедуры добавления, данных для других таблиц реализованы аналогичным образом.

Таким образом разработаны все необходимые составляющие базы данных.