Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт по лабораторной работе**

**Дисциплина**: Базы данных

**Тема**: SQL-DDL

Выполнила студентка гр. 43501/4 Хрусталева М.

(подпись)

Руководитель Мяснов А.В.

(подпись)

“\_ ” 2016 г.

Санкт - Петербург

2016

**Цель работы:**

Знакомство с основами проектирования схемы БД, языком описания сущностей и ограничений БД SQL-DDL.

**Программа работы**

1. Самостоятельное изучение SQL-DDL
2. Создание скрипта БД в соответствии с согласованной схемой (должны присутствовать первичные и внешние ключи, ограничения на диапазоны значений). Продемонстрировать скрипт преподавателю.
3. Создайте скрипт, заполняющий все таблицы БД данными
4. Выполнение SQL-запросов, изменяющих схему созданной БД по заданию преподавателя. Продемонстрировать их работу преподавателю.
5. Изучите основные возможности IBExpert. Получите ER-диаграмму созданной БД с помощью Database Designer.
6. Автоматически сгенерируйте данные при помощи IBExpert (для трех или большего числа таблиц, не менее 100000 записей в каждой из выбранных таблиц)

**Выполнение работы**

Data Definition Language (DDL) — язык описания данных. С помощью этого подмножества языка создаются, модифицируются и удаляются объекты базы данных.

Скрипт, создающий базу данных, имеет следующий вид:

create table Animals (

idAnimals integer not null,

name varchar(128) not null,

gender varchar(45) not null,

birth date not null,

cost numeric not null,

date\_animal date not null,

ill varchar(128),

pregnat varchar(128),

sp\_name varchar(128),

sp\_money\_min numeric,

sp\_money\_max numeric,

detachment\_id int not null,

kind\_id integer not null,

country\_id integer not null

);

create table Country (

idCountry integer not null,

name varchar(128) not null

);

create table Kind (

idKind integer not null,

name varchar(128) not null,

location varchar(128),

unique\_kind integer,

detachment\_id integer

);

create table Detachment (

idDetachment integer not null,

name varchar(128) not null

);

create table Eat (

idEat integer not null,

amount numeric not null,

deliver\_id integer not null,

animal\_id integer not null

);

create table Suppliers (

idSuppliers integer not null,

name varchar(128) not null

);

create table Deliveries (

idDeliveries integer not null,

cost numeric not null,

date\_del date not null,

product\_id integer not null,

supplier\_id integer not null

);

create table Products (

idProducts integer not null,

name varchar(128) not null

);

Для работы с БД необходимо после создания заполнить её данными, для этого был написал следующий скрипт, который записывает несколько значений во все таблицы базы данных:

insert into Animals (idAnimals, name, gender, birth, cost, date\_animal, ill, pregnat, sp\_name, sp\_money\_min, sp\_money\_max, detachment\_id, kind\_id, country\_id) values (0, 'Перси', 'М', '2007-04-02', 2300, '2008-09-27', 'Нет', 'Нет', 'Стройтрест', 120, 230, 0, 0, 0);

insert into Animals (idAnimals, name, gender, birth, cost, date\_animal, ill, pregnat, sp\_name, sp\_money\_min, sp\_money\_max, detachment\_id, kind\_id, country\_id) values (1, 'Тоша', 'М', '2013-08-12', 1250, '2013-12-21', 'Нет', 'Нет', 'Жилфонд', 80, 130, 0, 1, 2);

insert into Country (idCountry, name) values (0, 'Россия');

insert into Country (idCountry, name) values (1, 'Украина');

insert into Country (idCountry, name) values (2, 'Белорусия');

insert into Kind (idKind, name, location, unique\_kind, detachment\_id) values (0, 'Выхухоль', 'Лесная и лесостепная', 2, 0);

insert into Kind (idKind, name, location, unique\_kind, detachment\_id) values (1, 'Даурский ёж', 'Степная зона', 2, 0);

insert into Detachment (idDetachment, name) values (0, 'Насекомоядные');

insert into Detachment (idDetachment, name) values (1, 'Рукокрылые');

insert into Detachment (idDetachment, name) values (2, 'Парнокопытные');

insert into Eat (idEat, amount, deliver\_id, animal\_id) values (0, 2, 0, 0);

insert into Eat (idEat, amount, deliver\_id, animal\_id) values (1, 3, 8, 2);

insert into Suppliers (idSuppliers, name) values (0, 'ЧП Мясо');

insert into Suppliers (idSuppliers, name) values (1, 'ООО Еда');

insert into Suppliers (idSuppliers, name) values (2, 'Питание для животных');

insert into Deliveries (idDeliveries, cost, date\_del, product\_id, supplier\_id) values (2, 70, '2015-04-02', 0, 2);

insert into Deliveries (idDeliveries, cost, date\_del, product\_id, supplier\_id) values (3, 75, '2015-03-05', 0, 3);

insert into Deliveries (idDeliveries, cost, date\_del, product\_id, supplier\_id) values (4, 68, '2015-02-12', 1, 1);

insert into Products (idProducts, name) values (0, 'Мясо');

insert into Products (idProducts, name) values (1, 'Рыба');

insert into Products (idProducts, name) values (2, 'Молоко');

Индивидуальное задание – SQL-запрос, изменяющий структуру базы данных имеет следующий вид:

1. Реализовать иерархическую структуру видов, родов, семейств животных.
2. Реализовать учет представлений с животными.
3. Добавить учет продукта в таблицу потребления (Eat).

Для выполнения данного задания был написан скрипт:

alter table animals drop detachment\_id;

alter table kind

alter column kind.detachment\_id set detachment\_id int

references detachment(iddetachment)

alter table detachment add population int;

create table animalClass(

classID int not null primary key,

className varchar(30),

popul int);

alter table detachment add classID int references animalClass (classid)

create table showView(

showvID int not null primary key,

vie varchar(100));

create table animalShow(

showID int not null primary key,

animalID int not null references animals(idanimals),

showViewID int not null references showview(showvid),

showDate date);

alter table eat drop deliver\_id

alter table eat add product\_id int references products(idproducts)

alter table products add deliver\_id int references deliveries(iddeliveries)

alter table deliveries drop product\_id

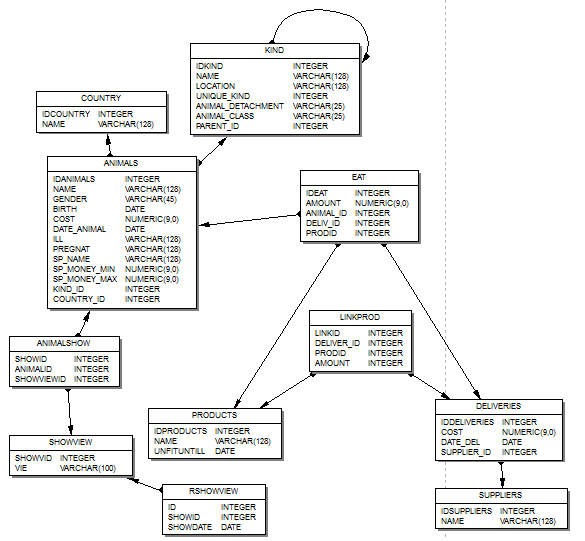
create table linkProd(

linkID int,

prodID int,

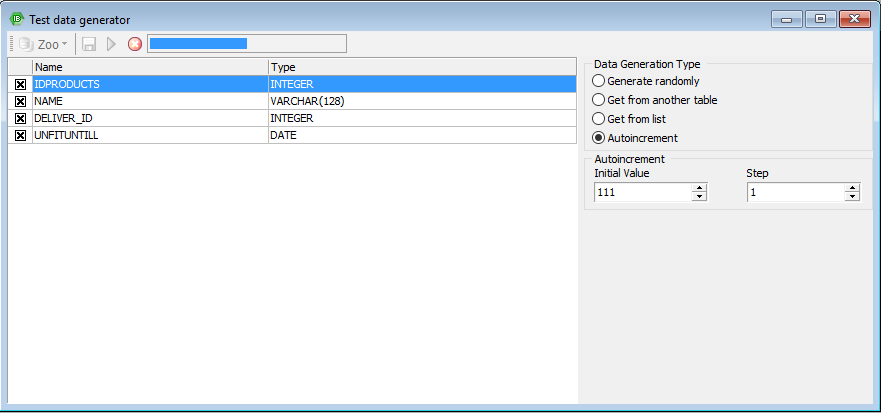
deliver\_ID int references deliveries(iddeliveries))

После изменения структуры данных была получена её блок-схема с помощью встроенного средства IBExpert Database Designer:

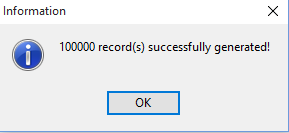


Генерировать данные будем с помощью встроенного средства Test Data Generator.

Test Data Generator предоставляет большой инструментарий для генерации тестовых данных для таблиц. Сгенерируем 100000 строк для таблицы Products:



При успешном завершении генерация выдаётся соответствующее сообщение:



**Вывод**

В данной работе мы познакомились с операторами DDL, изучили средство работы с базами данных IBExpert, научились писать скрипты, создающие таблицы и заполняющие их данными.

Запрос «create» используется для создания базы данных, таблицы, индекса, представления или хранимой процедуры. Запрос «alter» используется для изменения существующего объекта базы данных (таблицы, индекса, представления или хранимой процедуры) или самой базы данных. Запрос «drop» используется для удаления существующего объекта базы данных (таблицы, индекса, представления или хранимой процедуры) или самой базы данных. И наконец, в DDL существуют понятия первичного и внешнего ключа, которые осуществляют соблюдение целостности данных. Команды "первичный ключ" primary key, "внешний ключ" foreign key включаются в запросы «create table", «alter table».

Для создания БД были применены операторы DDL create table, alter table. Для хранения дат был использован тип данных date. Для задания значений из определенного списка был использован тип данных domain.

Функции языков DDL определяются первым словом в предложении (часто называемом запросом), которое почти всегда является глаголом. В случае с SQL это глаголы — «create» («создать»), «alter» («изменить»), «drop» («удалить»). Эти запросы или команды часто смешиваются с другими командами SQL, в связи с чем DDL не является отдельным компьютерным языком.