**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Институт информационных технологий и управления**

**Кафедра компьютерных систем и программных технологий**

**ОТЧЕТ**

**по лабораторной работе №3**

**«Язык SQL-DDL»**

**Информационное обеспечение систем управления**

Студент гр. 43501/1 Васильев А.М.

Преподаватель Мяснов А.В.

**Санкт-Петербург**

**2014**

# 1.Цель работы

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

## Индивидуальное задание:

БД хранит информацию об абонентах мобильного оператора, услугах (с фиксированной и посекундной тарификацией), использованию абонентами услуг, счетах, списаниях и зачислениях на счета. ИС дает возможность управлять различными акциями и начислением бонусов.

## 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).

3. Создать скрипт, заполняющий все таблицы БД данными

4. Выполнение SQL-запросов, изменяющих схему созданной БД. Получить ER-диаграмму созданной БД с помощью Database Designer.

5. Автоматически сгенерировать данные при помощи IBExpert (для трех или большего числа таблиц, не менее 100000 записей в каждой из выбранных таблиц)

# 2. Создание БД

Скрипт для создания БД:

### SET NAMES WIN1251;

### create database 'C:/db/nanofon.fdb' user 'SYSDBA' password 'masterkey' DEFAULT CHARACTER SET WIN1251;

### connect 'C:/db/nanofon.fdb' user 'SYSDBA' password 'masterkey';

### create table persons (

### id int primary key,

### passport decimal(10,0) not null unique check (passport >= 1000000000),

### name varchar(20) not null,

### sec\_name varchar(20),

### surname varchar(20) not null,

### birth date not null);

### create table tariffs (

### id int primary key,

### tariff\_name varchar(20));

### create table numbers (

### id int primary key,

### number decimal(10,0) not null unique check (number >= 1000000000));

### create table subs (

### id int primary key,

### person int references persons(id) not null,

### number int references numbers(id) not null,

### balance decimal(10,2) not null,

### tariff int references tariffs(id) not null);

### create table payments (

### id int primary key,

### sub int references subs(id) not null,

### amount decimal(8,2) not null,

### ptime timestamp not null);

### create table sms\_prices (

### id int primary key,

### price decimal(6,2) not null);

### create table call\_prices (

### id int primary key,

### price decimal(6,2) not null);

### create table actions (

### id int primary key);

### create table transactions (

### id int primary key,

### sub int references subs(id) not null,

### action int references actions(id) not null,

### amount int not null,

### actime timestamp not null);

### create table bonuses (

### id int primary key,

### name varchar (20) not null unique,

### minutes\_amount int,

### sms\_amount int,

### payments\_amount decimal(10,2),

### init\_amount\_sms int,

### init\_amount\_calls int,

### days int);

### create table actions\_bonuses(

### id int primary key,

### action int references actions(id) not null,

### bonus int references bonuses(id) not null);

### create table actions\_tariffs(

### id int primary key,

### action int references actions(id) not null,

### tariff int references tariffs(id) not null);

### create table subs\_bonuses(

### id int primary key,

### sub int references subs(id) not null,

### bonus int references bonuses(id) not null,

### amount\_left\_sms int,

### amount\_left\_calls int,

### start\_date date);

### 

### commit;

3 типа бонусов:

1. «Болтун» – необходимо говорить 100 минут в месяц, награда – скидка на звонки на следующий месяц.

2. «Телеграфист» - необходимо отправить 300 сообщений за месяц, награда – 10 бесплатных sms на месяц.

3. «Меценат» - необходимо внести на счет 1000 рублей за месяц, награда – 100 минут и 100 sms бесплатно и бессрочно.

### commit;

### drop database;

### commit;

### SET NAMES WIN1251;

### create database 'C:/db/nanofon.fdb' user 'SYSDBA' password 'masterkey' DEFAULT CHARACTER SET WIN1251;

### connect 'C:/db/nanofon.fdb' user 'SYSDBA' password 'masterkey';

### create table persons (

### id int primary key,

### passport decimal(10,0) not null unique check (passport >= 1000000000),

### name varchar(20) not null,

### sec\_name varchar(20),

### surname varchar(20) not null,

### birth date not null);

### create table tariffs (

### id int primary key,

### tariff\_name varchar(20));

### create table numbers (

### id int primary key,

### number decimal(10,0) not null unique check (number >= 1000000000));

### create table subs (

### id int primary key,

### person int references persons(id) not null,

### number int references numbers(id) not null,

### balance decimal(10,2) not null,

### tariff int references tariffs(id) not null);

### create table payments (

### id int primary key,

### sub int references subs(id) not null,

### amount decimal(8,2) not null,

### ptime timestamp not null);

### create table sms\_prices (

### id int primary key,

### price decimal(6,2) not null);

### create table call\_prices (

### id int primary key,

### price decimal(6,2) not null);

### create table actions (

### id int primary key);

### create table transactions (

### id int primary key,

### sub int references subs(id) not null,

### action int references actions(id) not null,

### amount int not null,

### actime timestamp not null);

### create table bonuses (

### id int primary key,

### name varchar (20) not null unique,

### minutes\_amount int,

### sms\_amount int,

### payments\_amount decimal(10,2),

### init\_amount\_calls int,

### init\_amount\_sms int,

### days int);

### create table actions\_bonuses(

### id int primary key,

### action int references actions(id) not null,

### bonus int references bonuses(id) not null);

### create table actions\_tariffs(

### id int primary key,

### action int references actions(id) not null,

### tariff int references tariffs(id) not null);

### create table subs\_bonuses(

### id int primary key,

### sub int references subs(id) not null,

### bonus int references bonuses(id) not null,

### amount\_left\_sms int,

### amount\_left\_calls int,

### start\_date date);

### 

### commit;

Скрипт для заполнения таблиц данными:

### insert into tariffs values

### (1,'Простецкий');

### insert into tariffs values

### (2,'SMS-зависимый');

### insert into tariffs values

### (3,'Звонить маме');

### insert into bonuses values

### (1,'Болтун',100,NULL,NULL,NULL,NULL,30);

### insert into bonuses values

### (2,'Телеграфист',NULL,300,NULL,NULL,10,30);

### insert into bonuses values

### (3,'Меценат',NULL,NULL,1000,100,100,NULL);

### insert into sms\_prices values (1,0.0);

### insert into sms\_prices values (2,2.0);

### insert into sms\_prices values (3,5.0);

### insert into call\_prices values (4,0.0);

### insert into call\_prices values (5,1.0);

### insert into call\_prices values (6,2.0);

### insert into call\_prices values (7,3.0);

### insert into actions values (1);

### insert into actions values (2);

### insert into actions values (3);

### insert into actions values (4);

### insert into actions values (5);

### insert into actions values (6);

### insert into actions values (7);

### insert into actions\_tariffs values (1,3,1);

### insert into actions\_tariffs values (2,6,1);

### insert into actions\_tariffs values (3,2,2);

### insert into actions\_tariffs values (4,7,2);

### insert into actions\_tariffs values (5,3,3);

### insert into actions\_tariffs values (6,5,3);

### insert into actions\_bonuses values (1,5,1);

### insert into actions\_bonuses values (2,1,2);

### insert into actions\_bonuses values (3,1,3);

### insert into actions\_bonuses values (4,4,3);

### insert into persons values

### (1,4001111222,'ИВАН','ИВАНОВИЧ','АНДРЕЕВ','1990-10-04');

### insert into persons values

### (2,4001113222,'ИВАН','ОЛЕГОВИЧ','ВИЙ','1990-10-04');

### insert into persons values

### (3,4002342422,'ИГОРЬ','ПЕТРОВИЧ','ИВАНОВ','1956-11-14');

### insert into persons values

### (4,4023456422,'ИВАН','ИВАНОВИЧ','ИВАНОВ','1978-12-24');

### insert into persons values

### (5,4111116422,'АЛЕКСАНДР','АРТЕМОВИЧ','СЕЧИН','1989-01-24');

### insert into numbers values (

### 1, 9216563548);

### insert into numbers values (

### 2, 9215346454);

### insert into numbers values (

### 3, 9213455344);

### insert into numbers values (

### 4, 9215455566);

### insert into numbers values (

### 5, 9215433566);

### 

### insert into subs values (

### 1,1,1,100.0,1);

### insert into subs values (

### 2,2,2,150.0,2);

### insert into subs values (

### 3,3,3,-50.0,3);

### insert into subs values (

### 4,4,4,152.34,2);

### insert into subs values (

### 5,5,5,12.22,1);

### insert into payments values

### (1,1,100.0,current\_timestamp (0));

### insert into payments values

### (2,2,10.0,current\_timestamp (0));

### insert into payments values

### (3,3,10.0,current\_timestamp (0));

### insert into payments values

### (4,4,50.0,current\_timestamp (0));

### insert into payments values

### (5,5,25.50,current\_timestamp (0));

Скрипт для модификации таблицы в соответствии с заданием:

 Модифицировать схему БД, чтобы выполнялось следующее:  
  1. Ввести возможность учета интернета.  
  2. Ввести учет порогов отключения для тарифов.

### create table inet\_prices (

### id int primary key,

### price decimal(6,2) not null);

### insert into inet\_prices values (8,9.0);

### insert into inet\_prices values (9,15.0);

### insert into actions values (8);

### insert into actions values (9);

### insert into actions\_tariffs values (7,8,1);

### insert into actions\_tariffs values (8,8,2);

### insert into actions\_tariffs values (9,9,3);

### alter table tariffs add

### min\_balance decimal(10,2) default 0.0;

### update tariffs set min\_balance = -10.0 where id = 3;

### update tariffs set min\_balance = 0.0 where id = 1;

### update tariffs set min\_balance = 0.0 where id = 2;

### commit;

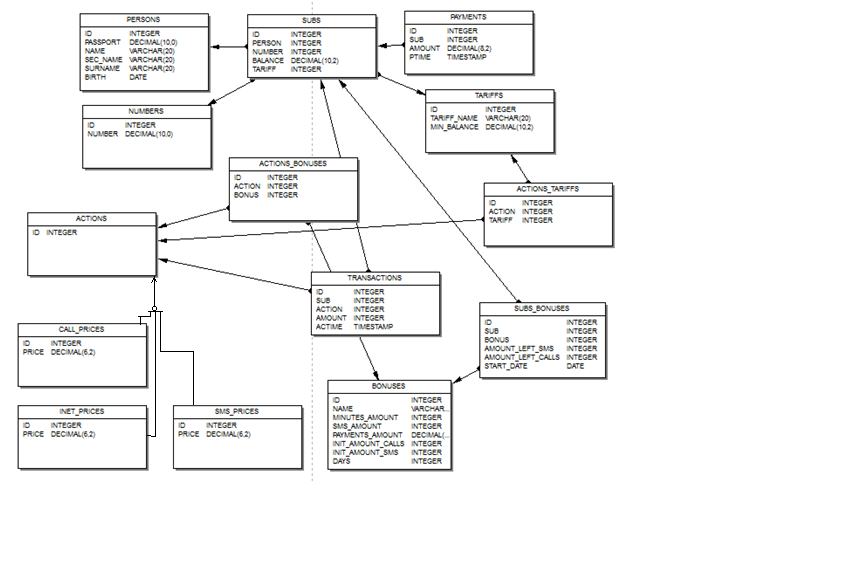


Рис 1. ER-диаграмма, построенная в IBExpert.

## Заполнение таблиц тестовыми данными

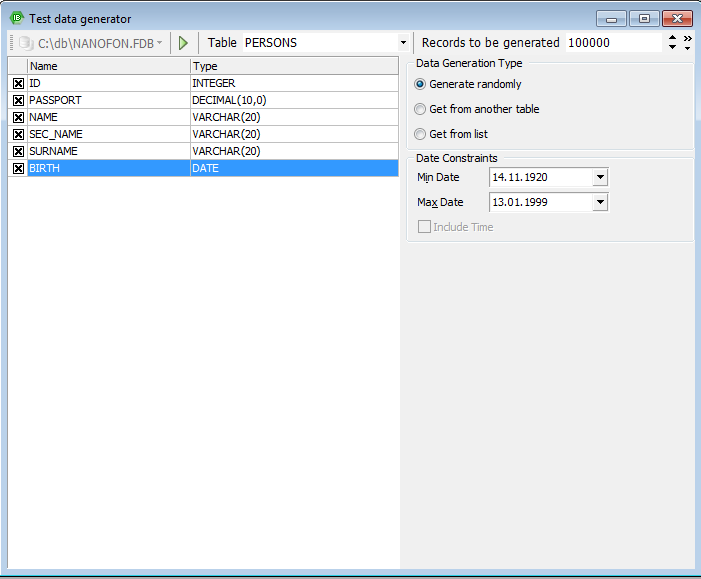


Рис 2. Окно IBExpert. Test data generator.

Для заполнения 3 таблиц тестовыми данными использовался IBExpert. Поскольку он не может генерировать целочисленные данные больше предела 32bit integer, использовался дополнительный запрос типа

### SQL> update persons set persons.passport = persons.passport+4008000000;

Результат генерации:

### SQL> select persons.id,persons.name,persons.sec\_name,persons.surname,persons.passport,persons.birth from persons where p

### ersons.id<10;

### ID NAME SEC\_NAME SURNAME PASSPORT BIRTH

### ============ ==================== ==================== ==================== ===================== ===========

### 1 Михаил Иванович Петров 4008631245 1971-12-30

### 2 Александр Иванович Петров 4008964695 1997-07-25

### 3 Иван Александрович Петров 4008351937 1994-06-17

### 4 Артем Петрович Петров 4008626701 1936-06-28

### 5 Артем Иванович Иванов 4008193154 1966-02-09

### 6 Иван Александрович Сидоров 4008191344 1977-09-12

### 7 Михаил Иванович Сидоров 4008411854 1945-06-03

### 8 Артем Михайлович Сидоров 4008510158 1924-09-24

### 9 Александр Иванович Сидоров 4008470619 1928-05-20

# 3. Вывод

В ходе работы были изучены основы языка SQL-DDL – языка определения структур и ограничений целостности баз данных. Были изучены команды создания и удаления баз данных; создания, изменения и удаления таблиц. Для создания большого количества тестовых данных использовалась программа IBExpert, предоставляющая графический интерфейс работы с БД.

Была создана база данных в соответствии с заданием, затем она была заполнена данными и модифицирована с помощью запросов INSERT, UPDATE и ALTER TABLE. Запросы такого типа будут использоваться в лабораторных работах и в дальнейшем.