Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчет по лабораторной работе №2**

по дисциплине «Базы данных»

«SQL-DDL»

Работу выполнил

студент гр. 43501/1

В.В. Сякаев

Проверил преподаватель

А.В. Мяснов

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2016г

Санкт - Петербург

2016

**Цель работы**

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

**Программа работы**

1. Создать скрипт БД в соответствии с согласованной схемой.
2. Создать скрипт, заполняющий все таблицы БД данными.
3. Осуществить выполнение SQL-запросов.
4. Получить ER-диаграмму БД.
5. Автоматически сгенерировать данные для таблиц.

**Ход работы**

**1. Создать скрипт для создания таблиц БД**

Вариант задания – музыкальная библиотека.

Музыкальная библиотека хранит информацию об альбомах, дисках, фестивалях, песнях и исполнителях.

Для составления схемы потребовались следующие таблицы:

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Название таблицы** | **Содержимое** | **Связи c таблицей album** |
| 1 | Album | Альбомы | - |
| 2 | Disc | Диски | Многие ко многим |
| 3 | Album\_disk | Связующая таблица | Один к одному |
| 4 | Album\_type | Связующая таблица | Один к одному |
| 5 | Group\_member | Члены группы | Многие ко многим |
| 6 | Album\_ group\_member | Связующая таблица | Один к одному |
| 7 | Fest | Фестивали | Многие ко многим |
| 8 | Fest\_albums | Связующая таблица | Один к одному |
| 9 | Song | Песни | Многие ко многим |
| 10 | Album\_song | Связующая таблица | Один ко многим |
| 11 | Fest\_song | Связующая таблица | - |

Связующие таблицы (хранящие id соответствующих таблиц) используются для организации связи «многие ко многим».

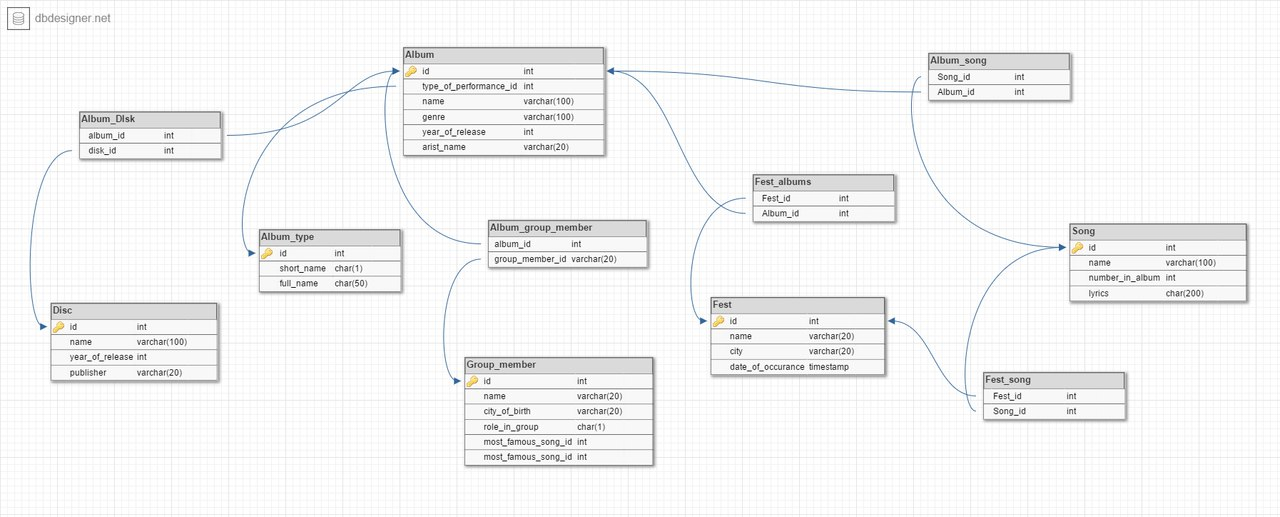


Рис. 1. Er-диаграмма исходной БД

CREATE TABLE 'Fest' (

'id' int NOT NULL ,

'name' varchar(20) NOT NULL,

'city' varchar(20) NOT NULL,

'date\_of\_occurance' DATE NOT NULL,

PRIMARY KEY ('id')

);

CREATE TABLE 'Song' (

'id' int NOT NULL ,

'name' varchar(100) NOT NULL,

'number\_in\_album' int NOT NULL,

'lyrics' char(200) NOT NULL,

PRIMARY KEY ('id')

);

CREATE TABLE 'Album' (

'id' int NOT NULL ,

'type\_of\_performance\_id' int NOT NULL,

'name' varchar(100) NOT NULL,

'genre' varchar(100) NOT NULL,

'year\_of\_release' int NOT NULL,

'arist\_name' varchar(20) NOT NULL,

PRIMARY KEY ('id')

);

CREATE TABLE 'Disc' (

'id' int NOT NULL ,

'name' varchar(100) NOT NULL,

'year\_of\_release' int NOT NULL,

'publisher' varchar(20) NOT NULL,

PRIMARY KEY ('id')

);

CREATE TABLE 'Fest\_albums' (

'Fest\_id' int NOT NULL,

'Album\_id' int NOT NULL

);

CREATE TABLE 'Fest\_song' (

'Fest\_id' int NOT NULL,

'Song\_id' int NOT NULL

);

CREATE TABLE 'Album\_song' (

'Song\_id' int NOT NULL,

'Album\_id' int NOT NULL

);

CREATE TABLE 'Album\_type' (

'id' int NOT NULL ,

'short\_name' char(1) NOT NULL,

'full\_name' char(50) NOT NULL,

PRIMARY KEY ('id')

);

CREATE TABLE 'Album\_DIsk' (

'album\_id' int NOT NULL,

'disk\_id' int NOT NULL

);

CREATE TABLE 'Album\_group\_member' (

'album\_id' int NOT NULL,

'group\_member\_id' varchar(20) NOT NULL

);

CREATE TABLE 'Group\_member' (

'id' int NOT NULL ,

'name' varchar(20) NOT NULL,

'city\_of\_birth' varchar(20) NOT NULL,

'role\_in\_group' char(1) NOT NULL,

'most\_famous\_song\_id' int NOT NULL,

'most\_famous\_song\_id' int NOT NULL,

PRIMARY KEY ('id')

);

ALTER TABLE 'Album' ADD CONSTRAINT 'Album\_fk0' FOREIGN KEY ('type\_of\_performance\_id') REFERENCES 'Album\_type'('id');

ALTER TABLE 'Fest\_albums' ADD CONSTRAINT 'Fest\_albums\_fk0' FOREIGN KEY ('Fest\_id') REFERENCES 'Fest'('id');

ALTER TABLE 'Fest\_albums' ADD CONSTRAINT 'Fest\_albums\_fk1' FOREIGN KEY ('Album\_id') REFERENCES 'Album'('id');

ALTER TABLE 'Fest\_song' ADD CONSTRAINT 'Fest\_song\_fk0' FOREIGN KEY ('Fest\_id') REFERENCES 'Fest'('id');

ALTER TABLE 'Fest\_song' ADD CONSTRAINT 'Fest\_song\_fk1' FOREIGN KEY ('Song\_id') REFERENCES 'Song'('id');

ALTER TABLE 'Album\_song' ADD CONSTRAINT 'Album\_song\_fk0' FOREIGN KEY ('Song\_id') REFERENCES 'Song'('id');

ALTER TABLE 'Album\_song' ADD CONSTRAINT 'Album\_song\_fk1' FOREIGN KEY ('Album\_id') REFERENCES 'Album'('id');

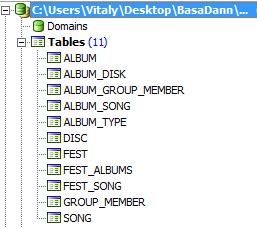
ALTER TABLE 'Album\_DIsk' ADD CONSTRAINT 'Album\_DIsk\_fk0' FOREIGN KEY ('album\_id') REFERENCES 'Album'('id');

ALTER TABLE 'Album\_DIsk' ADD CONSTRAINT 'Album\_DIsk\_fk1' FOREIGN KEY ('disk\_id') REFERENCES 'Disc'('id');

ALTER TABLE 'Album\_group\_member' ADD CONSTRAINT 'Album\_group\_member\_fk0' FOREIGN KEY ('album\_id') REFERENCES 'Album'('id');

ALTER TABLE 'Album\_group\_member' ADD CONSTRAINT 'Album\_group\_member\_fk1' FOREIGN KEY ('group\_member\_id') REFERENCES 'Group\_member'('id');

В результате запуска скрипта создаются все таблицы БД:



**2. Создать скрипт для заполнения таблиц БД**

insert into Album\_type values (0,'B','Bbb');

insert into Album\_type values (1,'C','Ccc');

insert into Album\_type values (2,'D','Ddd');

insert into Album\_type values (3,'E','Eee');

insert into Album\_type values (4,'F','Fff');

insert into Album\_type values (5,'G','Ggg');

insert into Album\_type values (6,'H','Hhh');

insert into Album\_type values (7,'I','Iii');

insert into Album\_type values (8,'J','Jjj');

insert into Album\_type values (9,'K','Kkk');

insert into Group\_member values (0,'Lars Urlich','Los Angeles','v',9);

insert into Group\_member values (1,'Kirk Hammet','Los Angeles','s',8);

insert into Group\_member values (2,'James Hetfield','Los Angeles','a',7);

insert into Group\_member values (3,'Robert Truhil'o','Los Angeles','q',6);

insert into Group\_member values (4,'Robert Truhil'o','Los Angeles','e',5);

insert into Group\_member values (5,'James Hetfield','Los Angeles','y',4);

insert into Group\_member values (6,'Lars Urlich','Los Angeles','u',3);

insert into Group\_member values (7,'Klif Bernton','Los Angeles','o',2);

insert into Group\_member values (8,'Jason Nusted','Los Angeles','s',1);

insert into Group\_member values (9,'Dave Mastein','Los Angeles','g',0);

insert into Album values (0,8,'Black\_album','Heavy metall',1984,'James');

insert into Album values (1,2,'Death Magnetic','Heavy metall',1984,'Kirk');

insert into Album values (2,4,'Lulu','Heavy metall',1984,'James');

insert into Album values (3,6,'Master of Puppets','Heavy metall',1984,'Lars');

insert into Album values (4,9,'Metallica','Heavy metall',1984,'James');

insert into Album values (5,5,'Ride the Lightning','Heavy metall',1984,'Robert');

insert into Album values (6,7,'Load','Heavy metall',1984,'James');

insert into Album values (7,3,'Reload','Heavy metall',1984,'Kirk');

insert into Album values (8,0,'St. Anger','Heavy metall',1984,'James');

insert into Album values (9,1,'Kill 'em All','Heavy metall',1984,'Lars');

insert into Album\_group\_member values (0,6);

insert into Album\_group\_member values (1,8);

insert into Album\_group\_member values (2,5);

insert into Album\_group\_member values (3,9);

insert into Album\_group\_member values (4,7);

insert into Album\_group\_member values (5,4);

insert into Album\_group\_member values (6,2);

insert into Album\_group\_member values (7,3);

insert into Album\_group\_member values (8,1);

insert into Album\_group\_member values (9,0);

insert into Disc values (0,'The Call of Ktulu',1989,'Lars Urlisch');

insert into Disc values (1,'Sad But True',1989,'Lars Urlisch');

insert into Disc values (2,'The Unforgiven',1989,'Lars Urlisch');

insert into Disc values (3,'Through The Never',1989,'Lars Urlisch');

insert into Disc values (4,'Golden Hits',1989,'Lars Urlisch');

insert into Disc values (5,'Golden Hits',1989,'Lars Urlisch');

insert into Disc values (6,'Golden Hits',1989,'Lars Urlisch');

insert into Disc values (7,'Golden Hits',1989,'Lars Urlisch');

insert into Disc values (8,'Golden Hits',1989,'Lars Urlisch');

insert into Disc values (9,'Golden Hits',1989,'Lars Urlisch');

insert into Album\_Disk values (0,4);

insert into Album\_Disk values (1,6);

insert into Album\_Disk values (2,7);

insert into Album\_Disk values (3,8);

insert into Album\_Disk values (4,5);

insert into Album\_Disk values (5,9);

insert into Album\_Disk values (6,1);

insert into Album\_Disk values (7,2);

insert into Album\_Disk values (8,3);

insert into Album\_Disk values (9,0);

insert into Song values (0,'fade to black',9,'Live is seemes to fade away, drifting further every day');

insert into Song values (1,'Nothing Else Matters',8,'And nothing else matters');

insert into Song values (2,'Of Wolf And Man',7,'Off through the new day's mist I run');

insert into Song values (3,'Holier That Thou',6,'The crap rolls out your mouth again Haven't changed, your brain is still gelatin');

insert into Song values (4,'one',5,'I can't remember anything Can't tell if this is true or dream');

insert into Song values (5,'master of puppets',4,'Come crawling faster, obey your Master');

insert into Song values (6,'blackened',3,'Live is seemes to fade away, drifting further every day');

insert into Song values (7,'creeping death',2,'So let it be written So let it be done');

insert into Song values (8,'broken beat',1,'You rise, you fall, you're down then you rise again What don't kill you make you more strong');

insert into Song values (9,'welcome home',0,'Welcome to where time stands still No one leaves and no one will');

insert into Album\_song values (0,9);

insert into Album\_song values (1,0);

insert into Album\_song values (2,7);

insert into Album\_song values (3,5);

insert into Album\_song values (4,6);

insert into Album\_song values (5,4);

insert into Album\_song values (6,2);

insert into Album\_song values (7,3);

insert into Album\_song values (8,1);

insert into Album\_song values (9,8);

insert into Fest values (0,'Fest1','Moscow','01.01.1984');

insert into Fest values (1,'Fest1','Saint-Petersburg','01.01.1984');

insert into Fest values (2,'Fest1','Kazan','01.01.1984');

insert into Fest values (3,'Fest1','Bugulma','01.01.1984');

insert into Fest values (4,'Fest1','Bryansk','01.01.1984');

insert into Fest values (5,'Fest1','Surgut','01.01.1984');

insert into Fest values (6,'Fest1','Pscow','01.01.1984');

insert into Fest values (7,'Fest1','Rostov','01.01.1984');

insert into Fest values (8,'Fest1','Volgograd','01.01.1984');

insert into Fest values (9,'Fest1','Vladivostok','01.01.1984');

insert into Fest\_albums values (0,9);

insert into Fest\_albums values (1,8);

insert into Fest\_albums values (2,7);

insert into Fest\_albums values (3,6);

insert into Fest\_albums values (4,5);

insert into Fest\_albums values (5,4);

insert into Fest\_albums values (6,3);

insert into Fest\_albums values (7,2);

insert into Fest\_albums values (8,1);

insert into Fest\_albums values (9,0);

insert into Fest\_song values (0,1);

insert into Fest\_song values (1,2);

insert into Fest\_song values (2,3);

insert into Fest\_song values (3,4);

insert into Fest\_song values (4,5);

insert into Fest\_song values (5,6);

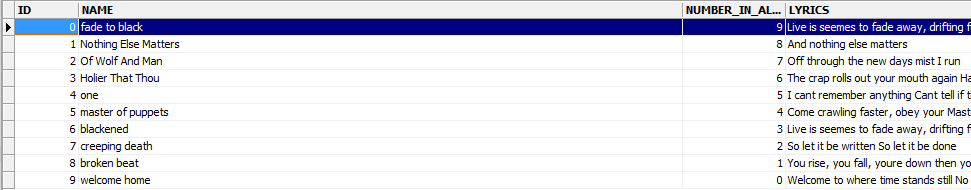
insert into Fest\_song values (6,7);

insert into Fest\_song values (7,8);

insert into Fest\_song values (8,9);

insert into Fest\_song values (9,0);

Пример заполненной с помощью скрипта таблицы:



**3. Осуществить выполнение SQL-запросов**

Модифицировать схему БД для удовлетворения следующим требованиям:

1. Ввести учет групп.
2. Ввести консистентный учет авторства альбомов: необходима возможность однозначного определения исполнителя альбома.
3. Ввести учет авторов текста и музыки композиций.

CREATE TABLE GROUPS (

ID INTEGER,

NAME CHAR(20),

CITY CHAR(20),

DATE\_BASEMENT DATE);

CREATE TABLE ARTIST (

ID INTEGER,

GROUP\_ID INTEGER,

NAME CHAR(20),

CITY CHAR(20),

ROLE CHAR(50));

ALTER TABLE GROUP ADD CONSTRAINT PK\_GROUP PRIMARY KEY (ID);

ALTER TABLE ALBUM ADD "GROUP\_ID" INTEGER;

ALTER TABLE ALBUM ADD CONSTRAINT FK\_GROUP\_ID FOREIGN KEY (GROUP\_ID) REFERENCES GROUP(ID);

ALTER TABLE ARTIST ADD CONSTRAINT PK\_ARTIST\_ID PRIMARY KEY (ID);

ALTER TABLE ALBUM ADD "ARTIST\_ID" INTEGER;

ALTER TABLE ALBUM ADD CONSTRAINT FK\_ARTIST\_ID FOREIGN KEY (ARTIST\_ID) REFERENCES ARTIST(ID);

DROP TABLE album\_disc;

DROP TABLE group\_member;

DROP TABLE album\_group\_member;

ALTER TABLE DISC ADD "ALBUM\_ID" INTEGER;

ALTER TABLE disc CONSTRAINT FK\_ALBUM\_ID FOREIGN KEY (ALBUM\_ID) REFERENCES ALBUM(ID);

ALTER TABLE SONG ADD "COMPOSER\_ID" INTEGER;

ALTER TABLE song CONSTRAINT FK\_COMPOSER\_ID FOREIGN KEY (COMPOSER\_ID) REFERENCES ARTIST(ID);

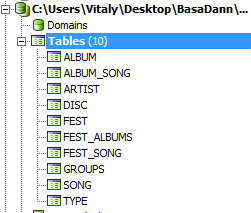
ALTER TABLE SONG ADD "AUTHOR\_ID" INTEGER;

ALTER TABLE song CONSTRAINT FK\_AUTHOR\_ID FOREIGN KEY (AUTHOR\_ID) REFERENCES ARTIST(ID);

ALTER TABLE FEST ADD "DATE\_OF\_OCCURANCE" DATE;

ALTER TABLE SONG ADD "CR\_DATE” DATE;

Результат работы скрипта:



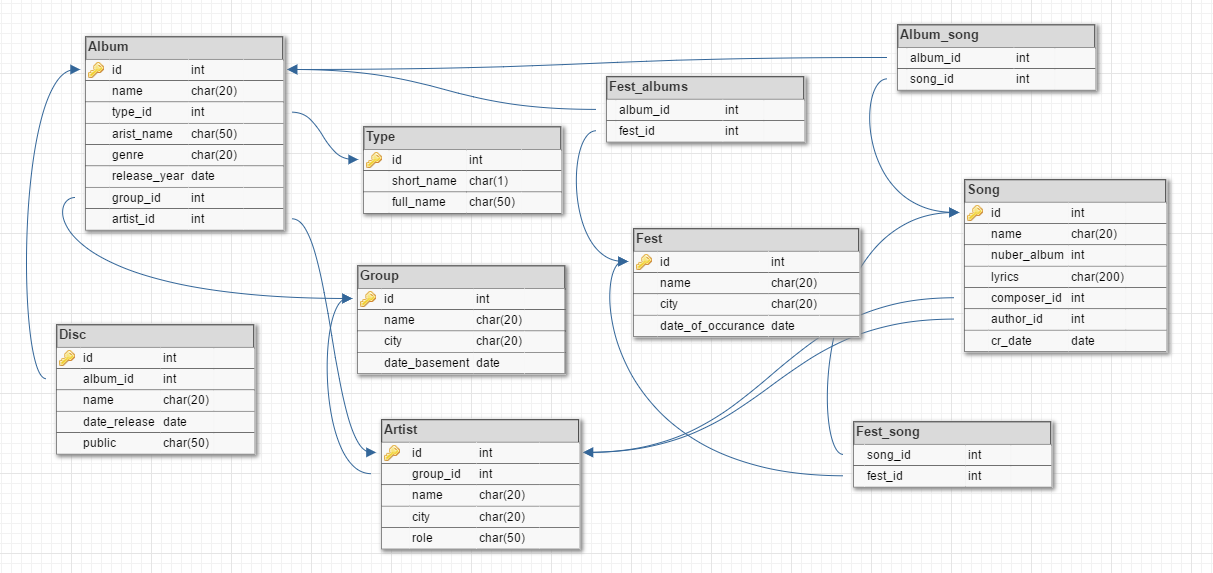
В базу данных добавлены таблицы:

GROUPS,ARTIST

Удалены таблицы:

album\_disc; group\_member; album\_group\_member;

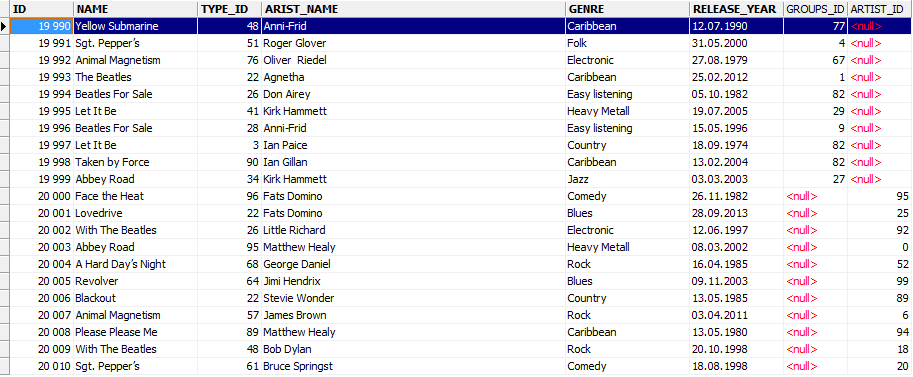
**4. Получить диаграмму БД**



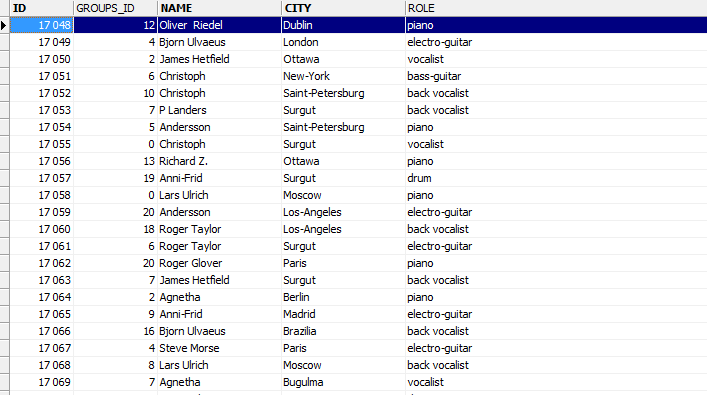
**5. Автоматически сгенерировать данные для таблиц**

Для генерации тестовых данных использовано средство «Test data generator» среды IBExpert.

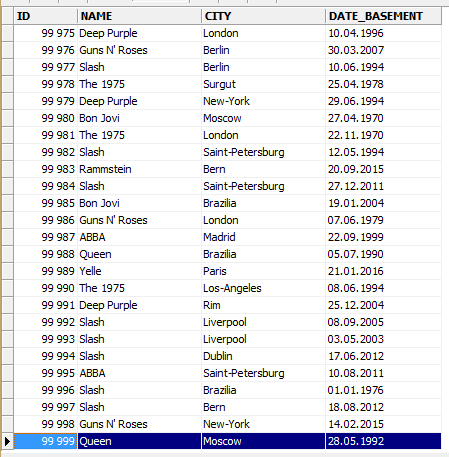
Часть данных для таблицы «album»:



Часть данных для таблицы «artist»:



Часть данных для таблицы «groups»:



**Выводы**

В данной работе использованы элементы языка SQL, предназначенные для оперирования таблицами: создание, удаление, добавление полей, ключей, внесение данных.

В данной работе также была изучена оболочка СУБД для проектирования и администрирования – IBExpert. Она существенно упрощает работу с БД Firebird за счет наличия графического интерфейса для работы со всеми объектами БД и возможности автоматической генерации данных:

1. Базу данных можно удобно создавать путем написания скрипта (это упрощает работу по сравнению с ручным вводом в Firebird);
2. С помощью «Database designer» создаются ER-диаграммы на основе существующих БД;
3. С помощью «Test data generator» базы данных можно заполнять большими объемами данных для тестирования.