Санкт-Петербургский Политехнический Университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

ОТЧЕТ
по лабораторной работе

«Язык SQL-DDL»

Базы данных

Работу выполнил студент

группа 43501/3 Крылов И.С.

Работу принял преподаватель

_____ Мяснов А.В.

Санкт-Петербург

2018

Содержание

Цель работы

Познакомиться 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. Демонстрация скрипта преподавателю.
3. Создание скрипта, заполняющего все таблицы БД данными.
4. Выполнение SQL-запросов, изменяющих схему созданной БД по заданию преподавателя. Демонстрация их работы преподавателю.

Теоретическая информация

Язык SQL (Structured Query Language) – язык структурированных запросов. Он позволяет формировать весьма сложные запросы к базам данных. В SQL определены два подмножества языка:

- **SQL-DDL** (Data Definition Language) – язык определения структур и ограничений целостности баз данных. Сюда относятся команды создания и удаления баз данных; создания, изменения и удаления таблиц; управления пользователями и т.д.
- **SQL-DML** (Data Manipulation Language) – язык манипулирования данными: добавление, изменение, удаление и извлечение данных, управления транзакциями. Функции SQL-DML определяются первым словом в предложении (часто называемом запросом), которое является глаголом: `SELECT` («выбрать»), `INSERT` («вставить»), `UPDATE` («обновить»), и `DELETE` («удалить»).

Выполнение работы

Структура базы данных

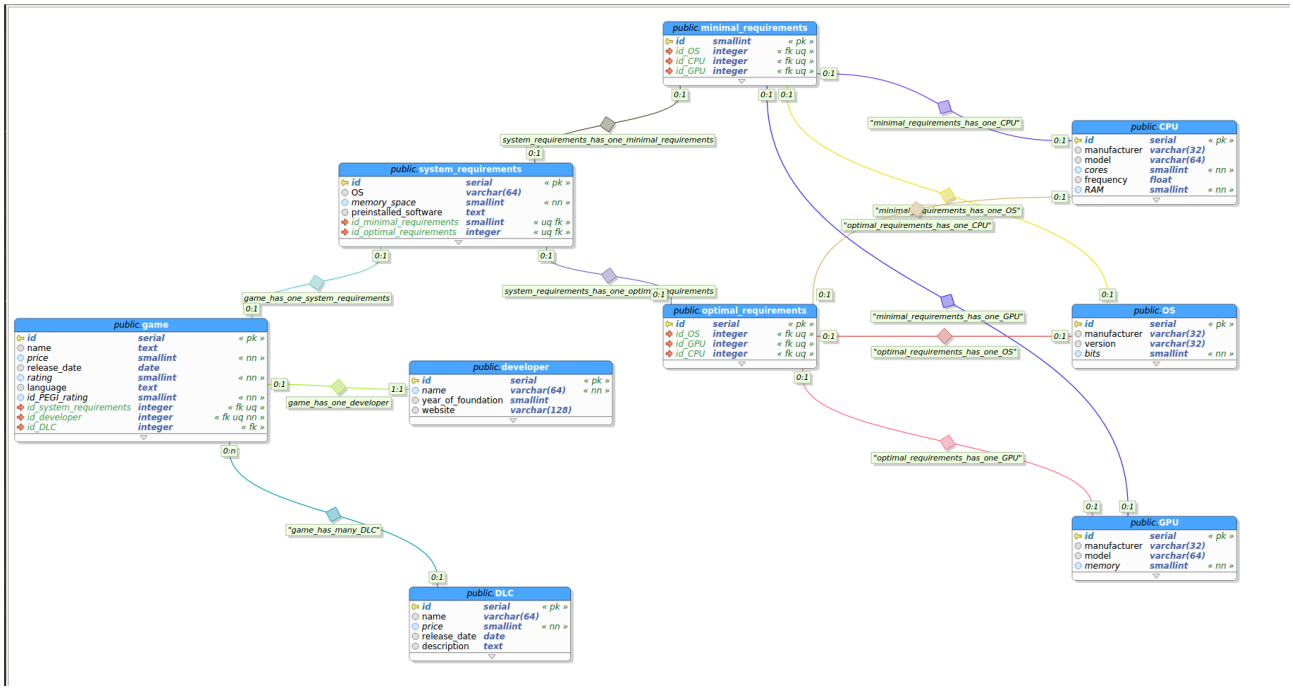

Рис. 4.1: Структура базы данных

Скрипт создания структуры базы данных

```
-- object: public.game | type: TABLE -
-- DROP TABLE IF EXISTS public.game CASCADE;
CREATE TABLE public.game(
    id serial NOT NULL,
    name text,
    price smallint NOT NULL GENERATED ALWAYS AS IDENTITY ,
    release_date date,
    rating smallint NOT NULL,
    language text,
    id_developer varchar(64),
    "id_PEGI_rating" smallint NOT NULL GENERATED ALWAYS AS
   IDENTITY ,
    "id_DLC" varchar(64),
    id_system_requirements smallint NOT NULL GENERATED ALWAYS AS
   IDENTITY ,
    id_system_requirements1 integer,
    id_developer1 integer NOT NULL,
    CONSTRAINT game_pk PRIMARY KEY (id)

);
-- ddl-end -
ALTER TABLE public.game OWNER TO postgres;
-- ddl-end -
```

```
22
23 -- object: public."DLC type: TABLE -
24 -- DROP TABLE IF EXISTS public."DLC"CASCADE;
25 CREATE TABLE public."DLC"(
26     id serial NOT NULL,
27     price smallint NOT NULL GENERATED ALWAYS AS IDENTITY ,
28     release_date date,
29     description text,
30     id_game integer,
31     CONSTRAINT "DLC_pk" PRIMARY KEY (id)
32
33 );
34 -- ddl-end -
35 ALTER TABLE public."DLC" OWNER TO postgres;
36 -- ddl-end -
37
38 -- object: public.system_requirements | type: TABLE -
39 -- DROP TABLE IF EXISTS public.system_requirements CASCADE;
40 CREATE TABLE public.system_requirements(
41     id serial NOT NULL,
42     "OS" varchar(64),
43     memory_space smallint NOT NULL GENERATED ALWAYS AS IDENTITY ,
44     preinstalled_software text,
45     id_minimal_requirements smallint,
46     id_optimal_requirements integer,
47     CONSTRAINT system_requirements_pk PRIMARY KEY (id)
48
49 );
50 -- ddl-end -
51 ALTER TABLE public.system_requirements OWNER TO postgres;
52 -- ddl-end -
53
54 -- object: public.developer | type: TABLE -
55 -- DROP TABLE IF EXISTS public.developer CASCADE;
56 CREATE TABLE public.developer(
57     id serial NOT NULL,
58     rating smallint NOT NULL GENERATED ALWAYS AS IDENTITY ,
59     year_of_foundation smallint,
60     website varchar(128),
61     CONSTRAINT developer_pk PRIMARY KEY (id)
62
63 );
64 -- ddl-end -
65 ALTER TABLE public.developer OWNER TO postgres;
66 -- ddl-end -
67
68 -- object: public."OS type: TABLE -
69 -- DROP TABLE IF EXISTS public."OS"CASCADE;
70 CREATE TABLE public."OS"(
71     id serial NOT NULL,
72     manufacturer varchar(32),
73     version varchar(32),
```

```
    bits smallint NOT NULL GENERATED ALWAYS AS IDENTITY ,
    CONSTRAINT "OS_pk" PRIMARY KEY (id)

);
-- ddl-end -
ALTER TABLE public."OS" OWNER TO postgres;
-- ddl-end -

-- object: public."CPU type: TABLE -
-- DROP TABLE IF EXISTS public."CPU"CASCADE;
CREATE TABLE public."CPU"(
    id serial NOT NULL,
    manufacturer varchar(32),
    model varchar(64),
    cores smallint NOT NULL GENERATED ALWAYS AS IDENTITY ,
    frequency smallint,
    "RAM" smallint NOT NULL GENERATED ALWAYS AS IDENTITY ,
    CONSTRAINT "CPU_pk" PRIMARY KEY (id)

);
-- ddl-end -
ALTER TABLE public."CPU" OWNER TO postgres;
-- ddl-end -

-- object: public."GPU type: TABLE -
-- DROP TABLE IF EXISTS public."GPU"CASCADE;
CREATE TABLE public."GPU"(
    id serial NOT NULL,
    manufacturer varchar(32),
    model varchar(64),
    memory smallint NOT NULL GENERATED ALWAYS AS IDENTITY ,
    CONSTRAINT "GPU_pk" PRIMARY KEY (id)

);
-- ddl-end -
ALTER TABLE public."GPU" OWNER TO postgres;
-- ddl-end -

-- object: public.minimal_requirements | type: TABLE -
-- DROP TABLE IF EXISTS public.minimal_requirements CASCADE;
CREATE TABLE public.minimal_requirements(
    id smallint NOT NULL,
    "id_CPU" integer,
    "id_OS" integer,
    "id_GPU" integer,
    CONSTRAINT minimal_requirements_pk PRIMARY KEY (id)

);
-- ddl-end -
ALTER TABLE public.minimal_requirements OWNER TO postgres;
-- ddl-end -

```

```
-- object: public.optimal_requirements | type: TABLE -
-- DROP TABLE IF EXISTS public.optimal_requirements CASCADE;
CREATE TABLE public.optimal_requirements(
    id serial NOT NULL,
    "id_GPU" integer,
    "id_OS" integer,
    "id_CPU" integer,
    CONSTRAINT optimal_requirements_pk PRIMARY KEY (id)

);
-- ddl-end -
ALTER TABLE public.optimal_requirements OWNER TO postgres;
-- ddl-end -

-- object: "CPU_fk type: CONSTRAINT -
-- ALTER TABLE public.minimal_requirements DROP CONSTRAINT IF EXISTS
   "CPU_fk"CASCADE;
ALTER TABLE public.minimal_requirements ADD CONSTRAINT "CPU_fk"
   FOREIGN KEY ("id_CPU")
REFERENCES public."CPU" (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end -

-- object: minimal_requirements_uq | type: CONSTRAINT -
-- ALTER TABLE public.minimal_requirements DROP CONSTRAINT IF EXISTS
   minimal_requirements_uq CASCADE;
ALTER TABLE public.minimal_requirements ADD CONSTRAINT
   minimal_requirements_uq UNIQUE ("id_CPU");
-- ddl-end -

-- object: "OS_fk type: CONSTRAINT -
-- ALTER TABLE public.minimal_requirements DROP CONSTRAINT IF EXISTS
   "OS_fk"CASCADE;
ALTER TABLE public.minimal_requirements ADD CONSTRAINT "OS_fk"
   FOREIGN KEY ("id_OS")
REFERENCES public."OS" (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
-- ddl-end -

-- object: minimal_requirements_uq2 | type: CONSTRAINT -
-- ALTER TABLE public.minimal_requirements DROP CONSTRAINT IF EXISTS
   minimal_requirements_uq2 CASCADE;
ALTER TABLE public.minimal_requirements ADD CONSTRAINT
   minimal_requirements_uq2 UNIQUE ("id_OS");
-- ddl-end -

-- object: "GPU_fk type: CONSTRAINT -
-- ALTER TABLE public.minimal_requirements DROP CONSTRAINT IF EXISTS
   "GPU_fk"CASCADE;
ALTER TABLE public.minimal_requirements ADD CONSTRAINT "GPU_fk"
   FOREIGN KEY ("id_GPU")
REFERENCES public."GPU" (id) MATCH FULL
ON DELETE SET NULL ON UPDATE CASCADE;
```

```
169 -- ddl-end -
170
171 -- object: minimal_requirements_uq1 | type: CONSTRAINT -
172 -- ALTER TABLE public.minimal_requirements DROP CONSTRAINT IF EXISTS
    minimal_requirements_uq1 CASCADE;
173 ALTER TABLE public.minimal_requirements ADD CONSTRAINT
    minimal_requirements_uq1 UNIQUE ("id_GPU");
174 -- ddl-end -
175
176 -- object: "GPU_fk type: CONSTRAINT -
177 -- ALTER TABLE public.optimal_requirements DROP CONSTRAINT IF EXISTS
    "GPU_fk"CASCADE;
178 ALTER TABLE public.optimal_requirements ADD CONSTRAINT "GPU_fk"
    FOREIGN KEY ("id_GPU")
179 REFERENCES public."GPU" (id) MATCH FULL
180 ON DELETE SET NULL ON UPDATE CASCADE;
181 -- ddl-end -
182
183 -- object: optimal_requirements_uq | type: CONSTRAINT -
184 -- ALTER TABLE public.optimal_requirements DROP CONSTRAINT IF EXISTS
    optimal_requirements_uq CASCADE;
185 ALTER TABLE public.optimal_requirements ADD CONSTRAINT
    optimal_requirements_uq UNIQUE ("id_GPU");
186 -- ddl-end -
187
188 -- object: "OS_fk type: CONSTRAINT -
189 -- ALTER TABLE public.optimal_requirements DROP CONSTRAINT IF EXISTS
    "OS_fk"CASCADE;
190 ALTER TABLE public.optimal_requirements ADD CONSTRAINT "OS_fk"
    FOREIGN KEY ("id_OS")
191 REFERENCES public."OS" (id) MATCH FULL
192 ON DELETE SET NULL ON UPDATE CASCADE;
193 -- ddl-end -
194
195 -- object: optimal_requirements_uq2 | type: CONSTRAINT -
196 -- ALTER TABLE public.optimal_requirements DROP CONSTRAINT IF EXISTS
    optimal_requirements_uq2 CASCADE;
197 ALTER TABLE public.optimal_requirements ADD CONSTRAINT
    optimal_requirements_uq2 UNIQUE ("id_OS");
198 -- ddl-end -
199
200 -- object: "CPU_fk type: CONSTRAINT -
201 -- ALTER TABLE public.optimal_requirements DROP CONSTRAINT IF EXISTS
    "CPU_fk"CASCADE;
202 ALTER TABLE public.optimal_requirements ADD CONSTRAINT "CPU_fk"
    FOREIGN KEY ("id_CPU")
203 REFERENCES public."CPU" (id) MATCH FULL
204 ON DELETE SET NULL ON UPDATE CASCADE;
205 -- ddl-end -
206
207 -- object: optimal_requirements_uq1 | type: CONSTRAINT -
208 -- ALTER TABLE public.optimal_requirements DROP CONSTRAINT IF EXISTS
    optimal_requirements_uq1 CASCADE;
```

```
209 ALTER TABLE public.optimal_requirements ADD CONSTRAINT
    optimal_requirements_uq1 UNIQUE ("id_CPU");
210 -- ddl-end -
211
212 -- object: minimal_requirements_fk | type: CONSTRAINT -
213 -- ALTER TABLE public.system_requirements DROP CONSTRAINT IF EXISTS
    minimal_requirements_fk CASCADE;
214 ALTER TABLE public.system_requirements ADD CONSTRAINT
    minimal_requirements_fk FOREIGN KEY (id_minimal_requirements)
215 REFERENCES public.minimal_requirements (id) MATCH FULL
216 ON DELETE SET NULL ON UPDATE CASCADE;
217 -- ddl-end -
218
219 -- object: system_requirements_uq | type: CONSTRAINT -
220 -- ALTER TABLE public.system_requirements DROP CONSTRAINT IF EXISTS
    system_requirements_uq CASCADE;
221 ALTER TABLE public.system_requirements ADD CONSTRAINT
    system_requirements_uq UNIQUE (id_minimal_requirements);
222 -- ddl-end -
223
224 -- object: optimal_requirements_fk | type: CONSTRAINT -
225 -- ALTER TABLE public.system_requirements DROP CONSTRAINT IF EXISTS
    optimal_requirements_fk CASCADE;
226 ALTER TABLE public.system_requirements ADD CONSTRAINT
    optimal_requirements_fk FOREIGN KEY (id_optimal_requirements)
227 REFERENCES public.optimal_requirements (id) MATCH FULL
228 ON DELETE SET NULL ON UPDATE CASCADE;
229 -- ddl-end -
230
231 -- object: system_requirements_uq1 | type: CONSTRAINT -
232 -- ALTER TABLE public.system_requirements DROP CONSTRAINT IF EXISTS
    system_requirements_uq1 CASCADE;
233 ALTER TABLE public.system_requirements ADD CONSTRAINT
    system_requirements_uq1 UNIQUE (id_optimal_requirements);
234 -- ddl-end -
235
236 -- object: system_requirements_fk | type: CONSTRAINT -
237 -- ALTER TABLE public.game DROP CONSTRAINT IF EXISTS system_requirements_fk
    CASCADE;
238 ALTER TABLE public.game ADD CONSTRAINT system_requirements_fk
    FOREIGN KEY (id_system_requirements1)
239 REFERENCES public.system_requirements (id) MATCH FULL
240 ON DELETE SET NULL ON UPDATE CASCADE;
241 -- ddl-end -
242
243 -- object: game_uq | type: CONSTRAINT -
244 -- ALTER TABLE public.game DROP CONSTRAINT IF EXISTS game_uq CASCADE;
245 ALTER TABLE public.game ADD CONSTRAINT game_uq UNIQUE (
    id_system_requirements1);
246 -- ddl-end -
247
248 -- object: developer_fk | type: CONSTRAINT -
249 -- ALTER TABLE public.game DROP CONSTRAINT IF EXISTS developer_fk CASCADE;
```

```
250 ALTER TABLE public.game ADD CONSTRAINT developer_fk FOREIGN KEY (
    id_developer1)
251 REFERENCES public.developer (id) MATCH FULL
252 ON DELETE RESTRICT ON UPDATE CASCADE;
253 -- ddl-end -
254
255 -- object: game_uq1 | type: CONSTRAINT -
256 -- ALTER TABLE public.game DROP CONSTRAINT IF EXISTS game_uq1 CASCADE;
257 ALTER TABLE public.game ADD CONSTRAINT game_uq1 UNIQUE (
    id_developer1);
258 -- ddl-end -
259
260 -- object: game_fk | type: CONSTRAINT -
261 -- ALTER TABLE public."DLC"DROP CONSTRAINT IF EXISTS game_fk CASCADE;
262 ALTER TABLE public."DLC" ADD CONSTRAINT game_fk FOREIGN KEY (
    id_game)
263 REFERENCES public.game (id) MATCH FULL
264 ON DELETE SET NULL ON UPDATE CASCADE;
265 -- ddl-end -
```

Скрипт заполнения таблиц тестовыми данными

```
1  INSERT INTO game
2  VALUES (DEFAULT, 'Red Dead Redemption', 1999, '01-01-2018', 8, '
       english', 18, 1, 1, 1);
3
4  INSERT INTO system_requirements
5  VALUES (DEFAULT, 'Windows', 73, 'DirectX 13', 1, 1);
6
7  INSERT INTO developer
8  VALUES (DEFAULT, 'rockstar games' , '1990', 'rockstargames.com');
9
10 INSERT INTO DLC
11 VALUES (DEFAULT, 'Rancho shooting', 395, '02-02-2018', '
       Дополнительные карты с сайдквестами-');
12
13 INSERT INTO minimal_requirements
14 VALUES (DEFAULT, 1, 1, 1);
15
16 INSERT INTO optimal_requirements
17 VALUES (DEFAULT, 1, 2, 2);
18
19 INSERT INTO CPU
20 VALUES (DEFAULT, 'Intel', 'i3', 4, 1.7, 1),
21        (DEFAULT, 'Intel', 'i7', 8, 2.4, 2);
22
23 INSERT INTO OS
24 VALUES (DEFAULT, 'Windows', '7', 64);
25
26 INSERT INTO GPU
27 VALUES (DEFAULT, 'Nvidia', 'GeForce GTX1080', 2),
```

```
28        (DEFAULT, 'Nvidia', 'RTX 2070', 8);
```

Выводы

В ходе выполнения лабораторной работы были изучены основы создания скриптов на языке SQL. С помощью SQL-DDL были описаны структуры хранимой в баз данных информации. С использованием SQL-DML созданные структуры были заполнены конкретными данными.