Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

Отчет по лабораторной работе №3

По дисциплине «Базы данных»

«SQL-DDL»

Работу выполнили студенты группы №43501/4 Н.С. Шаляпин\_\_\_\_\_\_\_\_\_

Работу принял преподаватель А.В. Мяснов\_\_\_\_\_\_\_\_\_\_

Санкт-Петербург

2015

**Цель работы**

Познакомить студентов 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**

Создание таблиц.

|  |
| --- |
| SET NAMES CYRL;  create database 'D:\!\43501\_4\!DB\my\_DB\AUTO\_SERVICES.sql'  user 'SYSDBA' password 'masterkey'  DEFAULT CHARACTER SET CYRL;  connect 'D:\!\43501\_4\!DB\my\_DB\AUTO\_SERVICES.sql'  user 'SYSDBA' password 'masterkey';  commit;  CREATE TABLE marks  (marks\_id INTEGER NOT NULL PRIMARY KEY, mark VARCHAR (16) NOT NULL);  commit;  CREATE TABLE motors  (motors\_id INTEGER NOT NULL PRIMARY KEY, motor VARCHAR (16) NOT NULL, type\_of\_motor VARCHAR (16), volume\_motor FLOAT, power FLOAT, torque FLOAT, consumption FLOAT);  commit;  CREATE TABLE complete\_set  (complete\_set\_id INTEGER NOT NULL PRIMARY KEY, equipment VARCHAR (16) NOT NULL);  commit;  CREATE TABLE model\_car  (model\_car\_id INTEGER NOT NULL PRIMARY KEY, model\_car VARCHAR (16) NOT NULL, marks\_id INTEGER REFERENCES marks(marks\_id),  motors\_id INTEGER REFERENCES motors(motors\_id), body\_type VARCHAR (16), complete\_set\_id INTEGER REFERENCES complete\_set(complete\_set\_id),  production\_year INTEGER, number\_of\_seats INTEGER, place\_vehicles VARCHAR (16));  commit;  CREATE TABLE customers  (customers\_id INTEGER NOT NULL PRIMARY KEY, customers VARCHAR (100) NOT NULL);  commit;  CREATE TABLE additional\_options  (additional\_options\_id INTEGER NOT NULL PRIMARY KEY, additional\_options VARCHAR (16) NOT NULL, price FLOAT);  commit;  CREATE TABLE trade  (trade\_id INTEGER NOT NULL PRIMARY KEY, customers\_id INTEGER REFERENCES customers(customers\_id),  model\_car\_id INTEGER REFERENCES model\_car(model\_car\_id),  additional\_options\_id INTEGER REFERENCES additional\_options(additional\_options\_id),  discount FLOAT, total\_price FLOAT);  commit;  CREATE TABLE service  (service\_id INTEGER NOT NULL PRIMARY KEY, model\_car\_id INTEGER REFERENCES model\_car(model\_car\_id),  type\_of\_fault VARCHAR (100) , trade\_id INTEGER REFERENCES trade(trade\_id));  commit; |

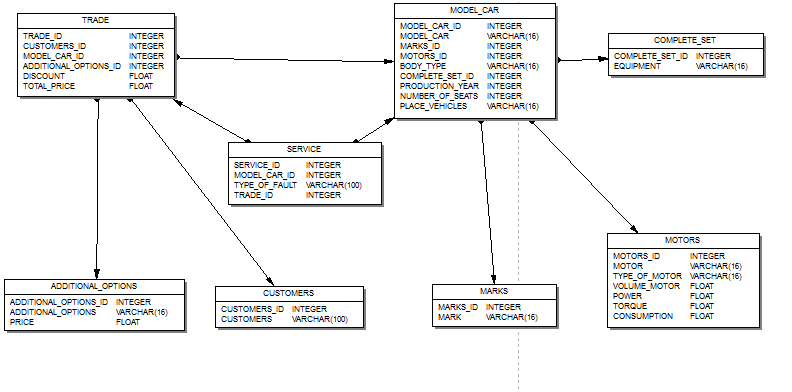
Авто инкремент.

|  |
| --- |
| CREATE GENERATOR gen\_marks\_id;  SET GENERATOR gen\_marks\_id TO 0;  set term !! ;  CREATE TRIGGER marks\_bi FOR marks  ACTIVE BEFORE INSERT POSITION 0  AS  BEGIN  NEW.marks\_id = GEN\_ID(gen\_marks\_id, 1);  END!!  set term ; !!  CREATE GENERATOR gen\_motors\_id;  SET GENERATOR gen\_motors\_id TO 0;  set term !! ;  CREATE TRIGGER motors\_bi FOR motors  ACTIVE BEFORE INSERT POSITION 0  AS  BEGIN  NEW.motors\_id = GEN\_ID(gen\_motors\_id, 1);  END!!  set term ; !!  CREATE GENERATOR gen\_complete\_set\_id;  SET GENERATOR gen\_complete\_set\_id TO 0;  set term !! ;  CREATE TRIGGER complete\_set\_bi FOR complete\_set  ACTIVE BEFORE INSERT POSITION 0  AS  BEGIN  NEW.complete\_set\_id = GEN\_ID(gen\_complete\_set\_id, 1);  END!!  set term ; !!  CREATE GENERATOR gen\_model\_car\_id;  SET GENERATOR gen\_model\_car\_id TO 0;  set term !! ;  CREATE TRIGGER model\_car\_bi FOR model\_car  ACTIVE BEFORE INSERT POSITION 0  AS  BEGIN  NEW.model\_car\_id = GEN\_ID(gen\_model\_car\_id, 1);  END!!  set term ; !!  CREATE GENERATOR gen\_customers\_id;  SET GENERATOR gen\_customers\_id TO 0;  set term !! ;  CREATE TRIGGER customers\_bi FOR customers  ACTIVE BEFORE INSERT POSITION 0  AS  BEGIN  NEW.customers\_id = GEN\_ID(gen\_customers\_id, 1);  END!!  set term ; !!  CREATE GENERATOR gen\_additional\_options\_id;  SET GENERATOR gen\_additional\_options\_id TO 0;  set term !! ;  CREATE TRIGGER additional\_options\_bi FOR additional\_options  ACTIVE BEFORE INSERT POSITION 0  AS  BEGIN  NEW.additional\_options\_id = GEN\_ID(gen\_additional\_options\_id, 1);  END!!  set term ; !!  CREATE GENERATOR gen\_trade\_id;  SET GENERATOR gen\_trade\_id TO 0;  set term !! ;  CREATE TRIGGER trade\_bi FOR trade  ACTIVE BEFORE INSERT POSITION 0  AS  BEGIN  NEW.trade\_id = GEN\_ID(gen\_trade\_id, 1);  END!!  set term ; !!  CREATE GENERATOR gen\_service\_id;  SET GENERATOR gen\_service\_id TO 0;  set term !! ;  CREATE TRIGGER service\_bi FOR service  ACTIVE BEFORE INSERT POSITION 0  AS  BEGIN  NEW.service\_id = GEN\_ID(gen\_service\_id, 1);  END!!  set term ; !! |

Добавление записей в таблицу.

|  |
| --- |
| INSERT INTO marks (mark)  VALUES ('BMW');  INSERT INTO marks (mark)  VALUES ('Nissan');  INSERT INTO motors (motor, type\_of\_motor, volume\_motor, power, torque, consumption)  VALUES ('М50' ,'Бензиновый' ,1991 , 450, 6000, 155);  INSERT INTO complete\_set (equipment)  VALUES ('Базовый');  INSERT INTO model\_car (model\_car, marks\_id, motors\_id, body\_type, complete\_set\_id,  production\_year, number\_of\_seats, place\_vehicles )  VALUES ('М3' ,1 ,1 ,'F80' ,1 ,2015 ,4 ,'Слева');  INSERT INTO customers (customers)  VALUES ('Васильев Вася Васичкин');  INSERT INTO additional\_options (additional\_options, price)  VALUES ('ABS', 90000);  INSERT INTO trade (customers\_id, model\_car\_id, additional\_options\_id, discount, total\_price)  VALUES (1, 1, 1, 10, 3950000);  INSERT INTO service (model\_car\_id, type\_of\_fault, trade\_id)  VALUES (1,'Неисправна система выпуска отработавших газов' , 1);  commit; |

ER-диаграмма.

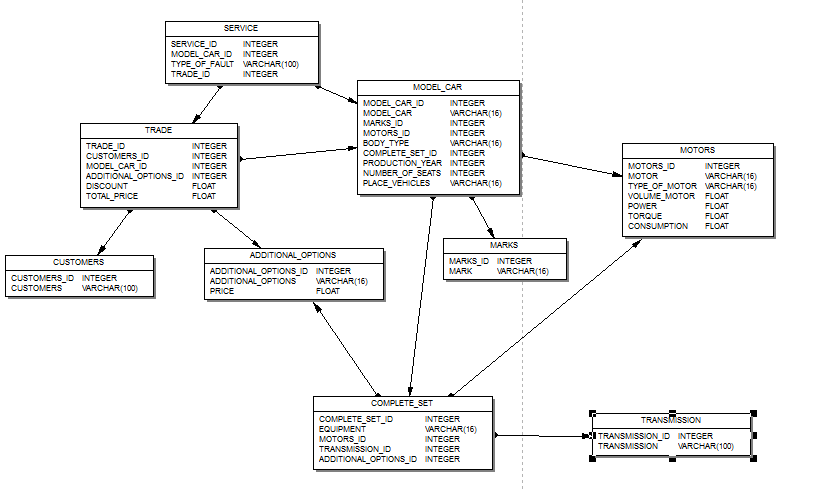


Скрипт изменения бд по заданию:

Ввести учет комплектаций автомобилей с множественными связями с опциями, трансмиссиями, двигателями.

|  |
| --- |
| CREATE TABLE transmission  (transmission\_id INTEGER NOT NULL PRIMARY KEY, transmission VARCHAR (100) NOT NULL);  commit;  CREATE GENERATOR gen\_transmission\_id;  SET GENERATOR gen\_transmission\_id TO 0;  set term !! ;  CREATE TRIGGER transmission\_bi FOR transmission  ACTIVE BEFORE INSERT POSITION 0  AS  BEGIN  NEW.transmission\_id = GEN\_ID(gen\_transmission\_id, 1);  END!!  set term ; !!  ALTER TABLE complete\_set  ADD motors\_id INTEGER REFERENCES motors(motors\_id);  ALTER TABLE complete\_set  ADD transmission\_id INTEGER REFERENCES transmission(transmission\_id);  ALTER TABLE complete\_set  ADD additional\_options\_id INTEGER REFERENCES additional\_options(additional\_options\_id);  INSERT INTO transmission(transmission)  VALUES ('S54B32');  commit;  INSERT INTO complete\_set(equipment, motors\_id, transmission\_id, additional\_options\_id)  VALUES ('Базовый', 1, 1, 1);  commit; |

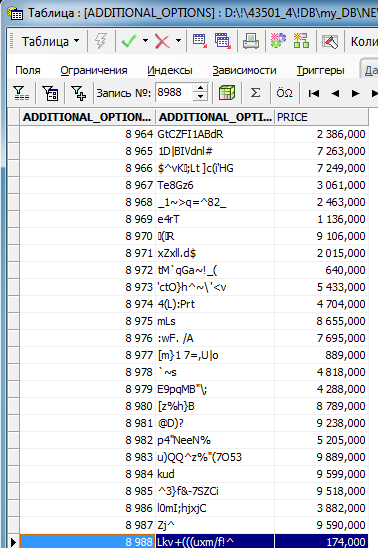
ЕR-диаграмма:

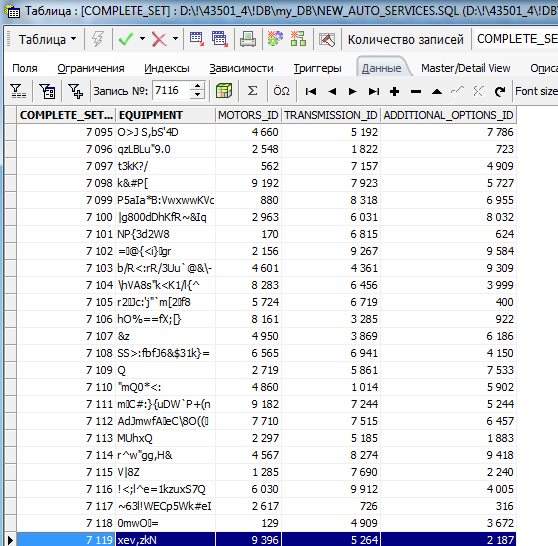


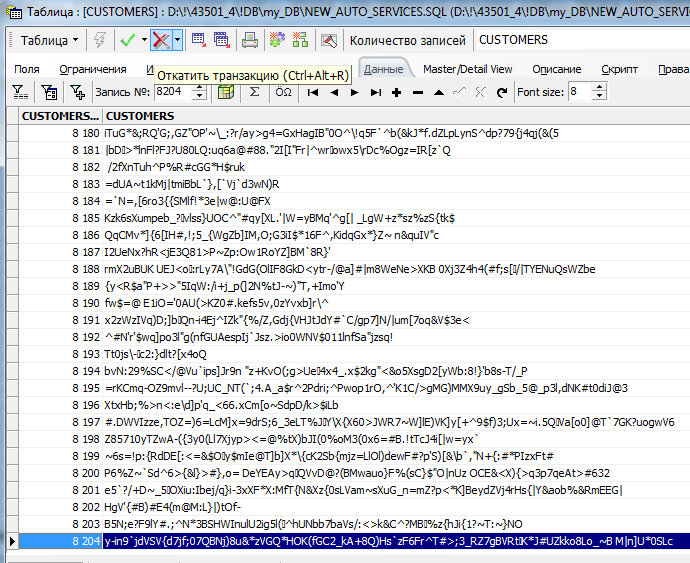
Автогенерация данных.

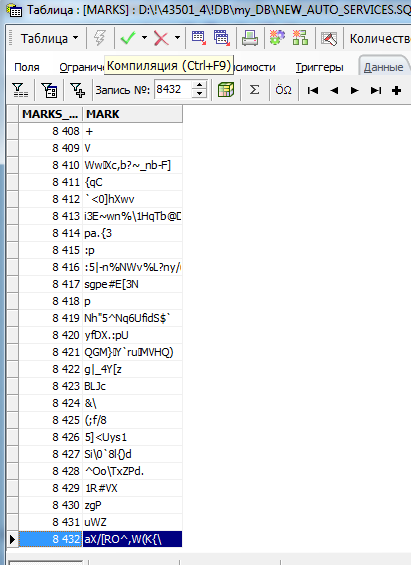
Для таблиц marks, motors, complete\_set, model\_car, customers, additional\_options, trade, service, transmission было автоматически добавлено по 100000 записей.

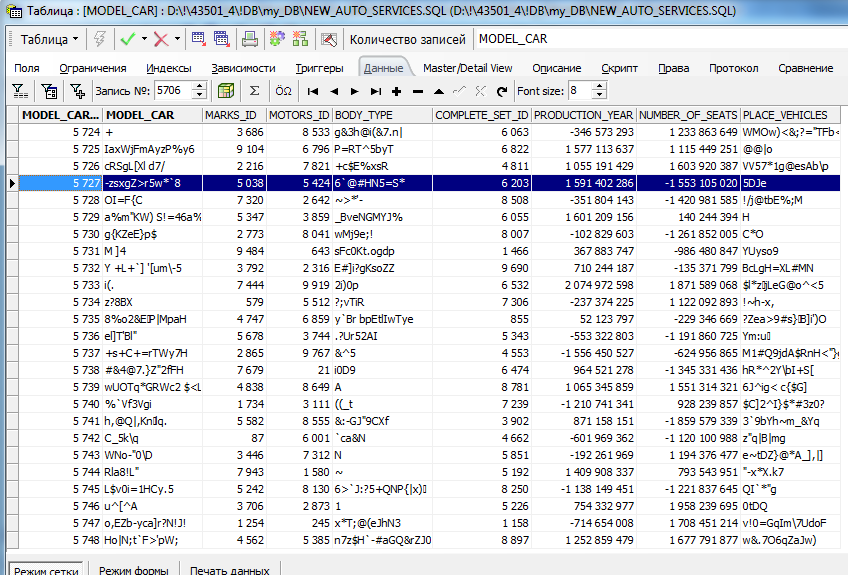
Фрагменты сгенерированных записей приведены ниже.

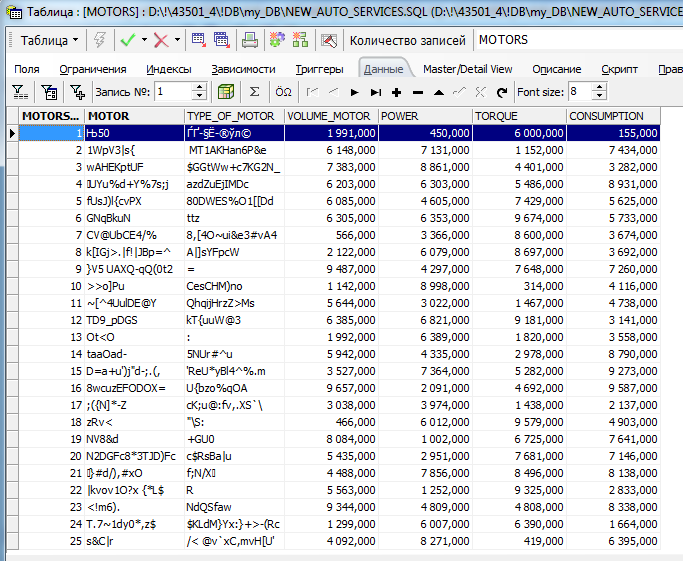


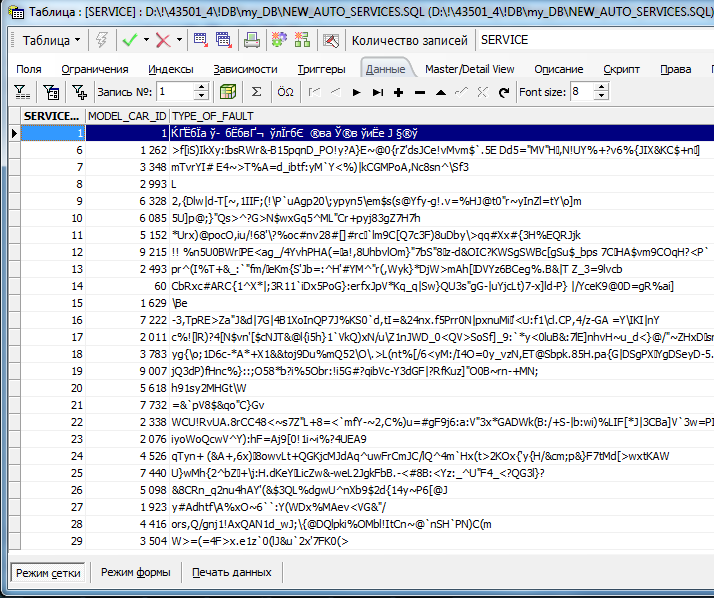


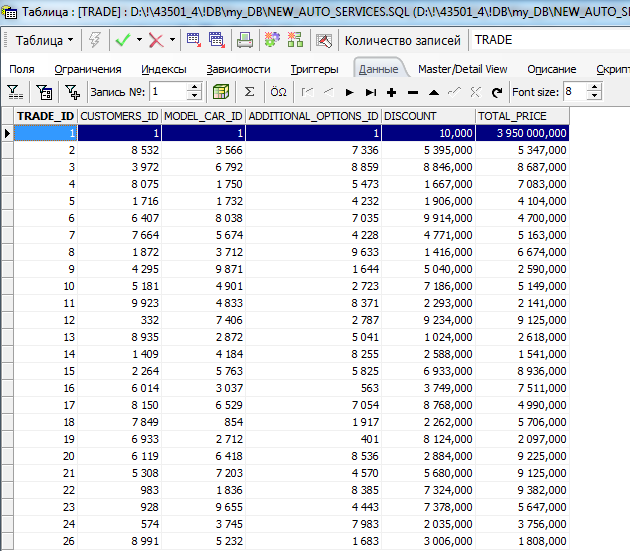


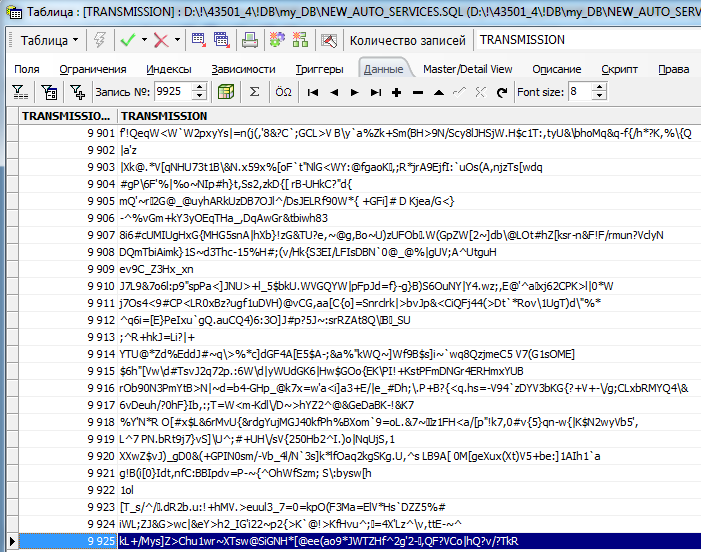












Заключение.

В результате работы был изучен язык SQL-DDL, Были написаны скрипты для создания, заполнения и модификации БД,

SQL-DDL (Data Definition Language) — язык определения структур и ограничений це­лостности баз данных. Использовались такие инструменты языка DDL как <Create> (со­здать), <Alter> (изменить), < Drop> (удалить).

Основой языка SQL является язык реляционных БД. Табличная структура реляци­онных БД хорошо понятна, поэтому язык SQL так же прост для изучения. Язык SQL позволяет управлять структурой БД, это обеспечивает гибкость БД к изменяющимся тре­бованиям предметной области.

Были изучены некоторые возможности среды IBExpert: генерирование ER-диаграммы созданной БД и заполнение таблиц тестовыми данными. В ходе работы проблем не возникло.