Санкт-Петербургский Политехнический университет Петра Великого

Институт компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

**Отчёт о лабораторной работе №4**

**Дисциплина**: Базы данных

**Тема**: Язык SQL-DDL

Выполнил студент гр. 43501/1 С.В. Смирнов

(подпись)

Руководитель А.В. Мяснов

(подпись)

“ ” 2016 г.

Санкт-Петербург

2016

# Цели работы

Познакомить студентов с языком создания запросов управления данными SQL-DML.

# Программа работы

1. Изучите SQL-DML
2. Выполните все запросы из списка стандартных запросов. Продемонстрируйте результаты преподавателю.
3. Получите у преподавателя и реализуйте SQL-запросы в соответствии с индивидуальным заданием. Продемонстрируйте результаты
4. Выполненные запросы SELECT сохраните в БД в виде представлений, запросы INSERT, UPDATE или DELETE -- в виде ХП. Выложите скрипт в Subversion.

# Ход работы

## Язык SQL

Data Manipulation Language (DML) (язык управления (манипулирования) данными) — это семейство компьютерных языков, используемых в компьютерных программах или пользователями [баз данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) для получения, вставки, удаления или изменения данных в [базах данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85).

## Выполнение стандартных запросов

* Выборка всех данных из всех таблиц

Скрипт:

connect 'D:\7\_sem\carservice.fdb' user 'SYSDBA' password 'masterkey';

commit;

CREATE VIEW v1 as SELECT \* FROM Car;

CREATE VIEW v2 as SELECT \* FROM Client;

CREATE VIEW v3 as SELECT \* FROM Bonus;

CREATE VIEW v4 as SELECT \* FROM Ser\_info;

CREATE VIEW v5 as SELECT \* FROM Work\_Parts;

CREATE VIEW v6 as SELECT \* FROM Offer;

CREATE VIEW v7 as SELECT \* FROM Id\_Works;

CREATE VIEW v8 as SELECT \* FROM Id\_Part;

CREATE VIEW v9 as SELECT \* FROM Works;

CREATE VIEW v10 as SELECT \* FROM Parts;

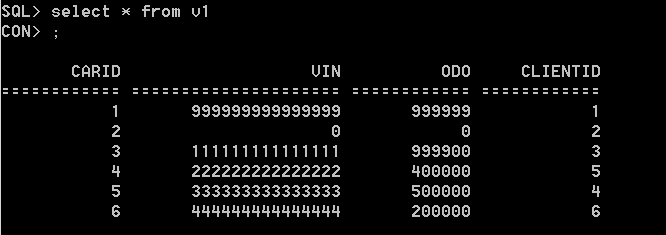
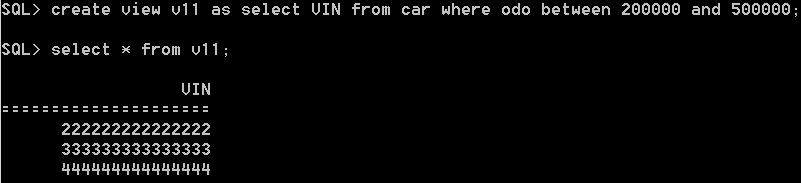


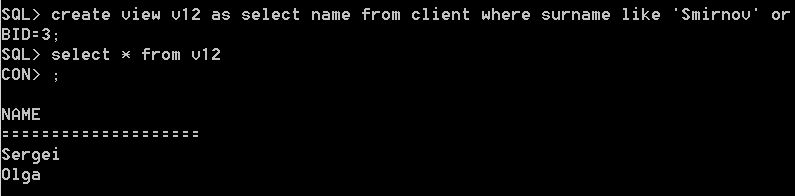
Рис.1. Пример выборки из табл. Car

* Выборка данных из 1 таблицы при нескольких условиях

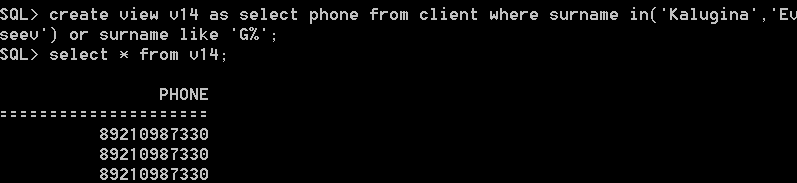
Выведем VIN машин с пробегом от 200 до 500 тысяч км :



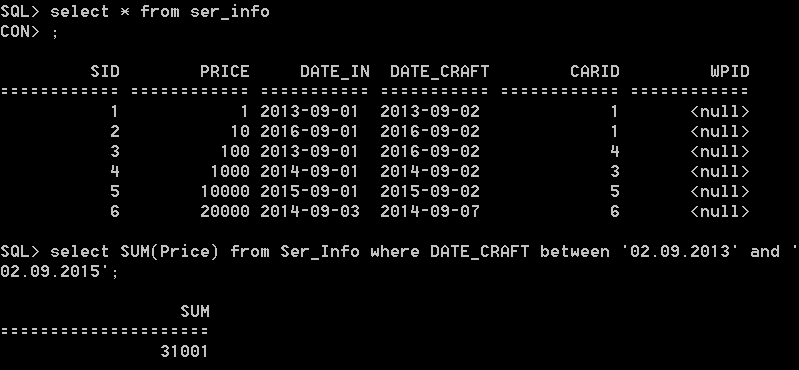
Выведем имена клиентов с фамилией «Смирнов» или с бонуснымID=3:



Выведем номера телефонов клиентов с фамилией Калугина или Евсеев или с фамилией начинающейся на букву Г:

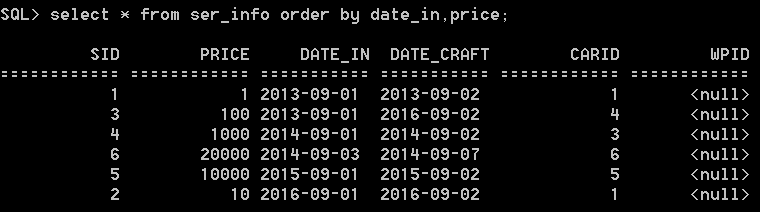


* Вычисляемое поле в запросе: вычислим сумму всех заказов с датой исполнения между 02.09.2013 и 02.09.2015



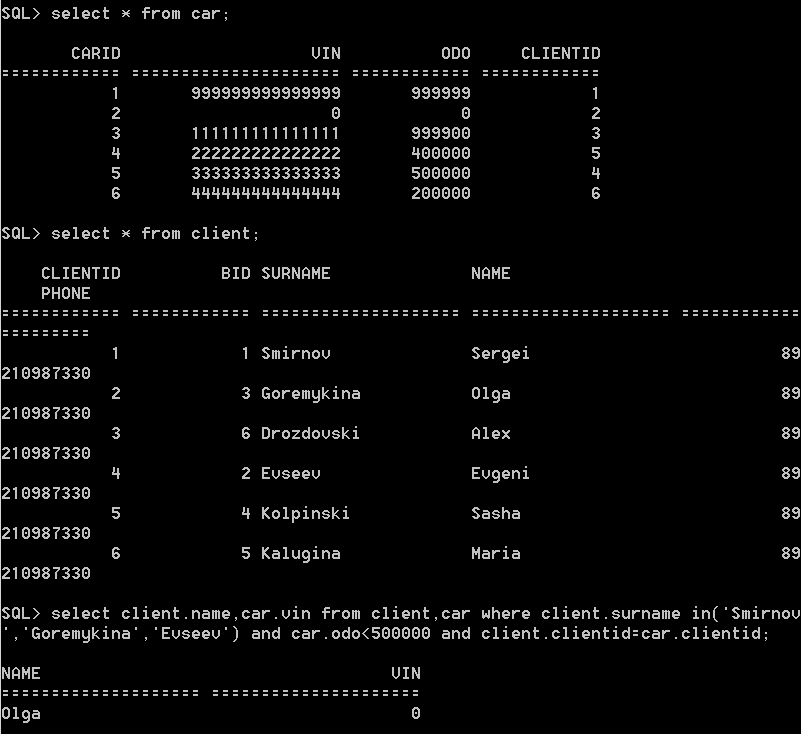
* Выборка всех данных с сортировкой по нескольким полям:

Выберем все данные с сортировкой по дате поступления машины и цене ремонта:

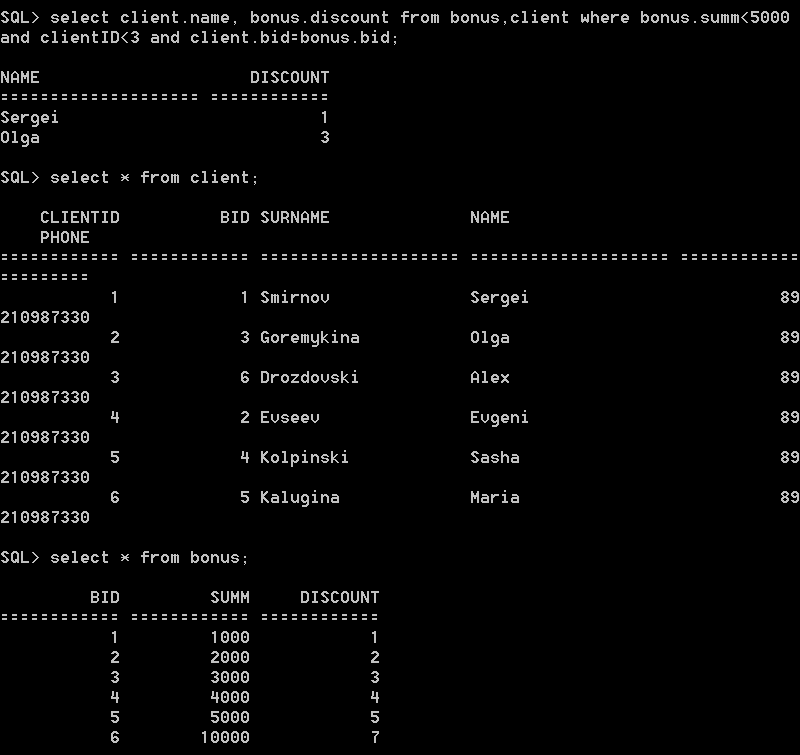


* Выборка данных из связанных таблиц:

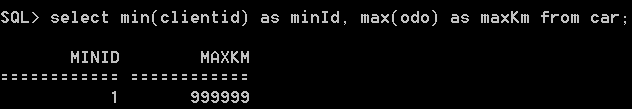
Выведем Имя владельца авто с соответствующим VIN-номером:



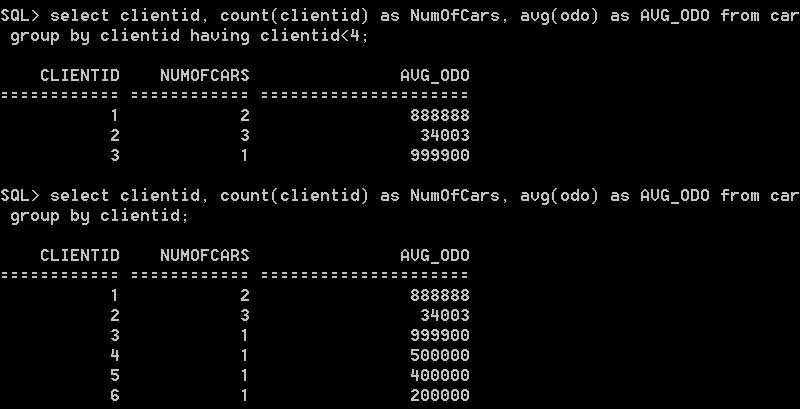
Выведем имя клиента и соответствующую ему скидку при условии, что сумма покупок у клиентов не более 5000, ид клиента не более 3:



* Запрос, рассчитывающий несколько совокупных характеристик таблиц

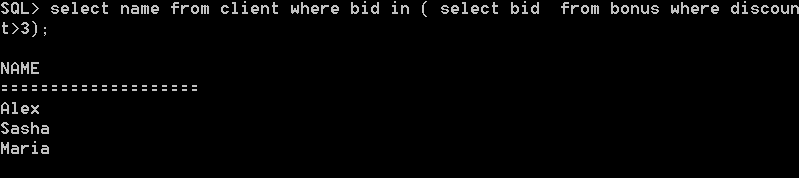


* Запрос, рассчитывающий совокупную характеристику с использованием группировки, наложите ограничение на результат группировки. Выберем всех клиентов ИД которых меньше 4, посчитаем количество машин для каждого клиента и средний пробег по всем машинам:



* Придумайте и реализуйте пример использования вложенного запроса

Выведем имена клиентов со скидкой больше 3%:



* Использование оператора INSERT



















* С помощью оператора UPDATE измените значения нескольких полей у всех записей, отвечающих заданному условию

Увеличим процент скидки для посетителей с суммарным чеком больше 5000:



Увеличился процент скидки для 6 и 7 ценовой категории.

Дальнейшие пункты будут выполнены в виде процедур. Так же в скрипте ниже будет указаны предыдущие запросы в виде представлений/процедур в соответствии с заданием:

connect 'D:\7\_sem\carservice.fdb' user 'SYSDBA' password 'masterkey';

commit

drop view v1;

drop view v2;

drop view v3;

drop view v4;

drop view v5;

drop view v6;

drop view v7;

drop view v8;

drop view v9;

drop view v10;

drop view v11;

drop view v12;

drop view v13;

drop view v14;

drop view v15;

drop view v16;

drop view v17;

drop view v18;

drop view v19;

drop view v20;

drop procedure insert\_value\_in\_car;

drop procedure up\_bonus;

drop procedure rem\_Offer;

drop procedure del\_unused\_parts;

CREATE VIEW v1 as SELECT \* FROM Car;

CREATE VIEW v2 as SELECT \* FROM Client;

CREATE VIEW v3 as SELECT \* FROM Bonus;

CREATE VIEW v4 as SELECT \* FROM Ser\_info;

CREATE VIEW v5 as SELECT \* FROM Work\_Parts;

CREATE VIEW v6 as SELECT \* FROM Offer;

CREATE VIEW v7 as SELECT \* FROM Id\_Works;

CREATE VIEW v8 as SELECT \* FROM Id\_Part;

CREATE VIEW v9 as SELECT \* FROM Works;

CREATE VIEW v10 as SELECT \* FROM Parts;

CREATE VIEW v11 AS SELECT VIN FROM car where odo between 200000 and 500000;

CREATE VIEW v12 AS SELECT name FROM client where surname like 'Smirnov' or BID = 3;

CREATE VIEW v13 AS SELECT phone FROM client where surname in ('Kalugina','Evseev') or surname like 'G%';

CREATE VIEW v14 AS SELECT SUM(Price) as summa FROM Ser\_Info where DATE\_CRAFT between '02.09.2013' and '02.09.2015';

CREATE VIEW v15 AS SELECT \* FROM ser\_info order by date\_in,price;

CREATE VIEW v16 AS SELECT client.name, car.vin from client,car where client.surname in ('Smirnov','Goremykina','Evseev') and car.odo<500000 and client.clientid = car.clientid;

CREATE VIEW v17 AS SELECT client.name, bonus.discount FROM bonus,client where bonus.summ < 5000 and clientid<3 and client.bid = bonus.bid;

CREATE VIEW v18 AS SELECT min(clientid) as minid,max(odo) as maxKm from car;

CREATE VIEW v19 AS SELECT clientid,count(clientid) as numofcars, avg(odo) as avg\_odo from car group by clientid having clientid<4;

CREATE VIEW v20 AS SELECT name FROM client where bid in ( select bid from bonus where discount>3);

SET TERM ^ ;

CREATE PROCEDURE insert\_value\_in\_car ( n1 INT, n2 DECIMAL(15), n3 DECIMAL(6), n4 INT)

AS

BEGIN

INSERT INTO Car VALUES( :n1, :n2, :n3, :n4 );

END

^

CREATE PROCEDURE up\_bonus(summ int)

AS BEGIN

UPDATE Bonus

SET discount = discount + 1

WHERE summ > :summ;

END

^

--execute up\_bonus(5000)

CREATE PROCEDURE rem\_Offer(price INT)

AS BEGIN

DELETE FROM offer

WHERE price >

(select avg(:price)

from Offer);

END

^

CREATE PROCEDURE del\_unused\_parts

AS BEGIN

DELETE FROM Parts

WHERE PartID

NOT IN

(SELECT PartID

FROM Id\_Part);

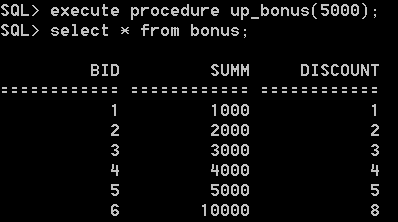
END

^

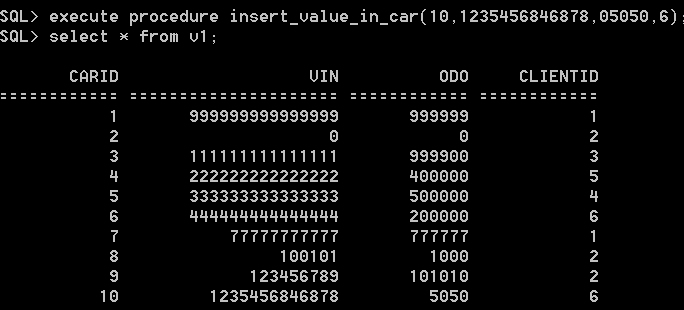
SET TERM ; ^

SELECT \* FROM v2;

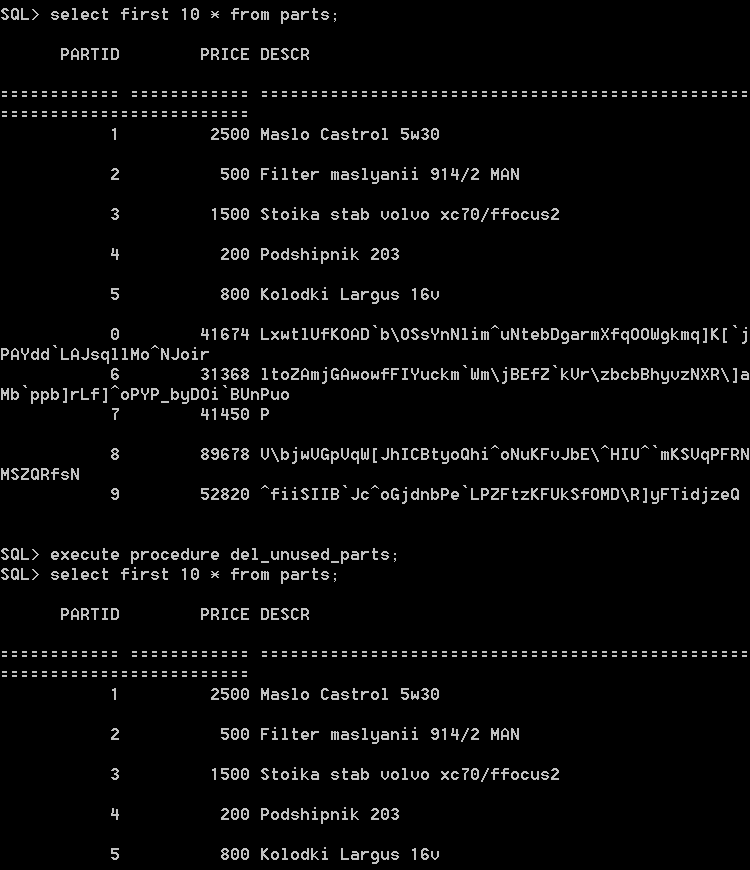
Так как запросы аналогичны выполненным вручную- результаты их исполнения не изменились. Результаты выполнения процедур:



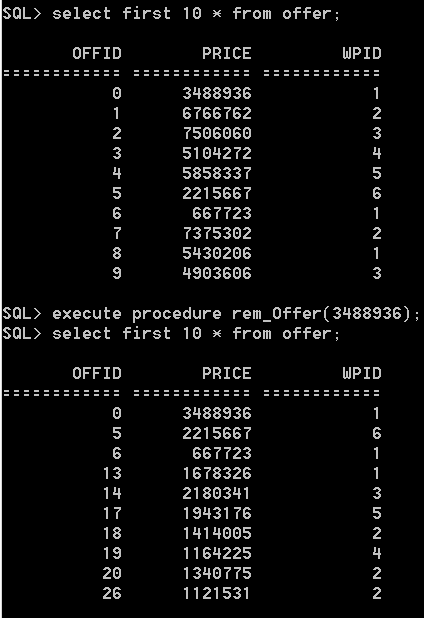
Процент скидки для программы 6 увеличился на 1.



Добавился новый автомобиль соответствующий запросу.



Удалили неиспользуемые записи о деталях, осталось только 5 записей.



Удалили акционные предложения со средней стоимостью больше указанной.

**Выводы:**

В ходе лабораторной работы мы научились эффективно использовать конструкции языка sql-DML. Так же были изучены хранимые процедуры и представления, которые позволяют хранить наиболее используемые запросы не вводя их каждый раз заново.

В ходе лабораторной работы возникла необходимость переназначения оператора конца строки при создании процедур, однако эта проблема была успешно решена.