Санкт-Петербургский политехнический университет

Институт информационных технологий и управления

Кафедра компьютерных систем и программных технологий

ОТЧЕТ

**о лабораторной работе № 4**

Язык SQL-DML

Базы данных

**Выполнила студентка**: гр.43501/3 Емельянова А.В.

**Преподаватель:** Мяснов А.В.

Санкт-Петербург

2016

1. Цели работы:

Познакомиться с языком создания запросов управления данными SQL-DML.

2. Программа работы:

* Изучить SQL-DML.
* Выполнить все запросы из списка стандартных запросов. Продемонстрировать результаты преподавателю.
* Получить у преподавателя и реализовать SQL-запросы в соответствии с индивидуальным заданием. Продемонстрировать результаты преподавателю.
* Выполненные запросы SELECT сохранить в БД в виде представлений, запросы INSERT, UPDATE или DELETE -- в виде ХП. Выложить скрипт в Subversion.

3. Выполнение программы.

3.1. Выполнение всех запросов из списка стандартных запросов.

1) Сделать выборку всех данных из каждой таблицы:

select \* from ADMINN

select \* from blog

select \* from blog\_categories

select \* from blog\_post

select \* from categories1

select \* from categories2

select \* from enable\_blog

select \* from enable\_post

select \* from friendship

select \* from groupmembership

select \* from groupp

select \* from gr\_num

select \* from links

select \* from member

select \* from paid\_services

select \* from privacy\_blog

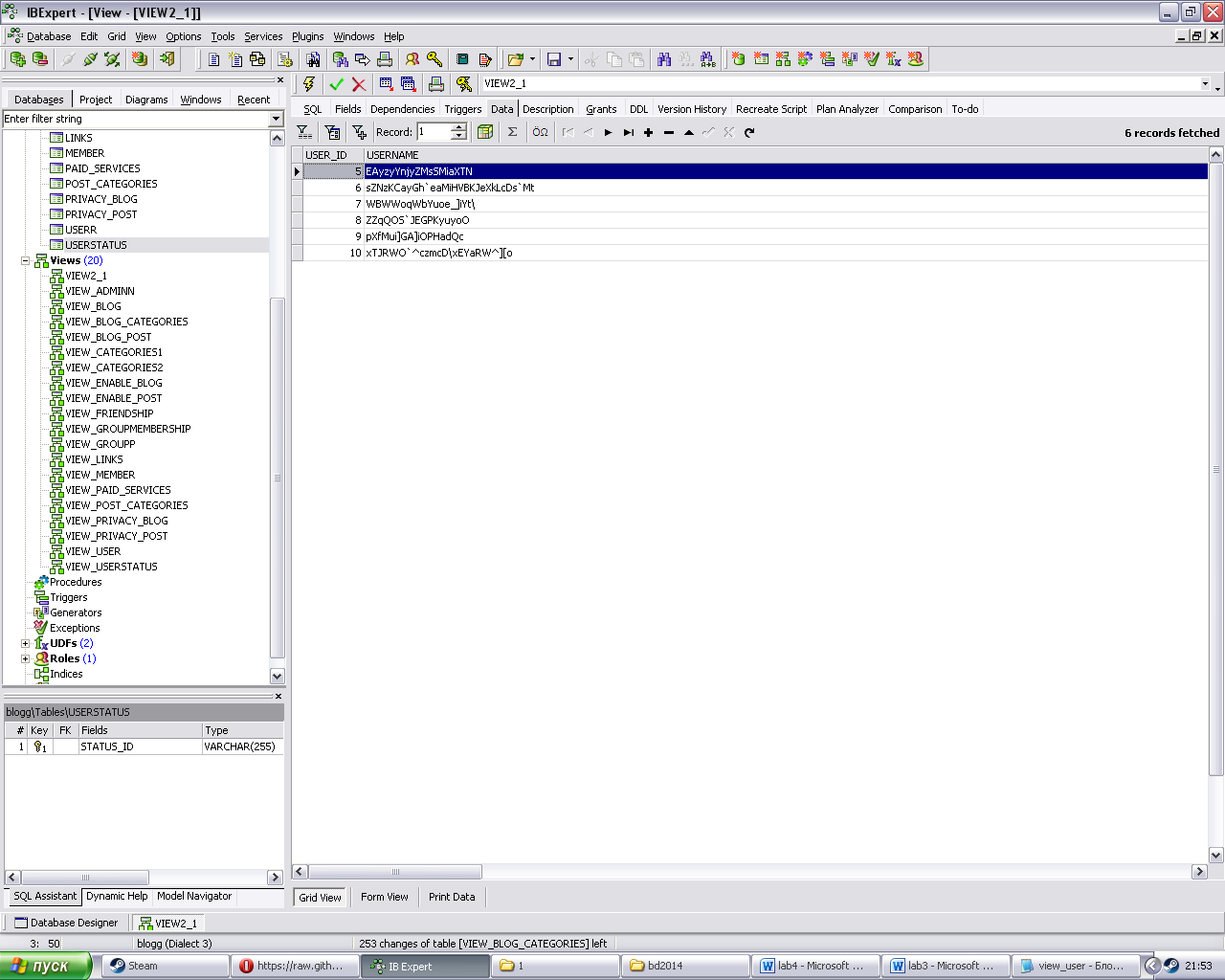
select \* from privacy\_post

select \* from userr

select \* from userstatus

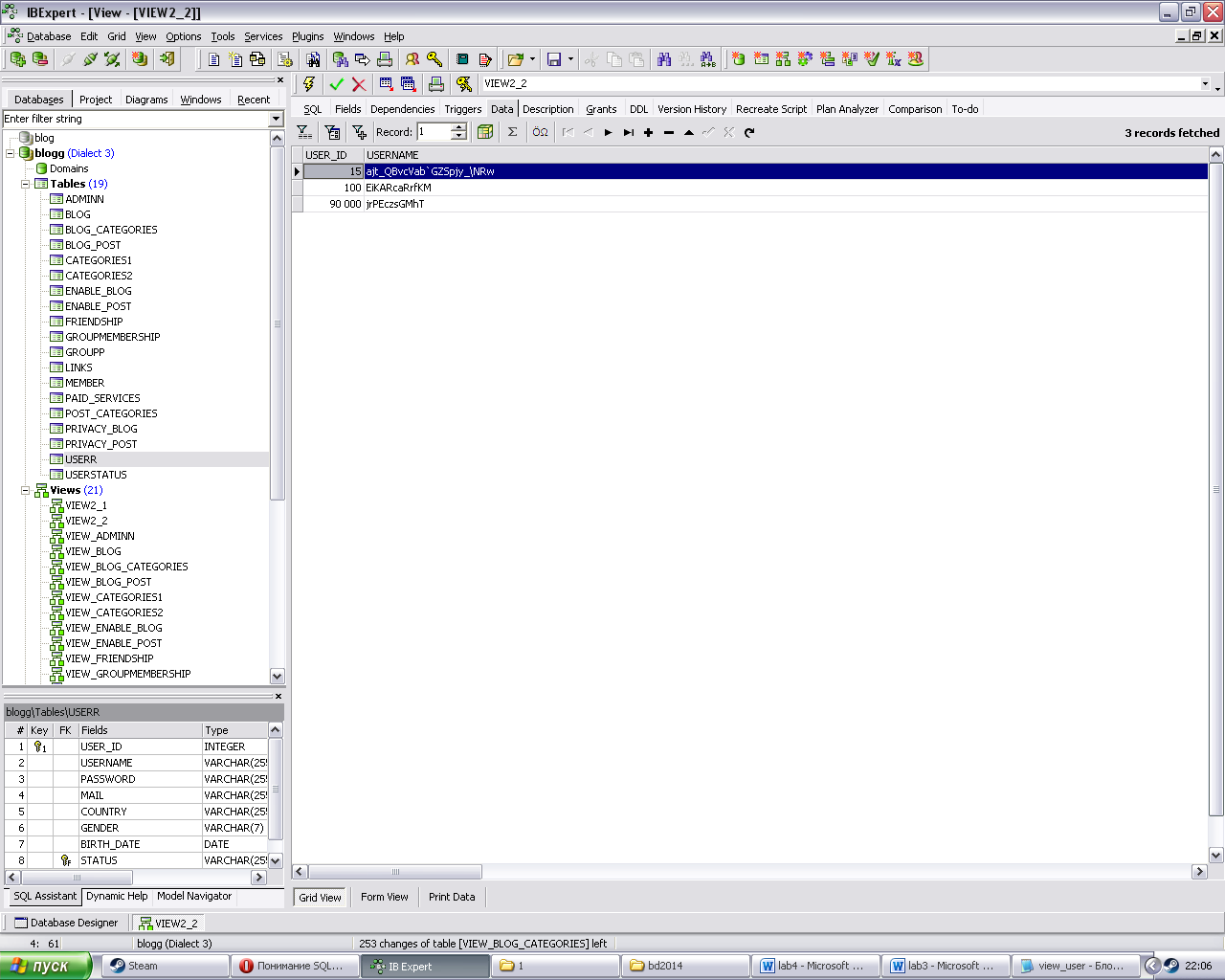
2) Сделать выборку данных из одной таблицы при нескольких условиях, с использованием логических операций, LIKE, BETWEEN, IN (не менее 3-х разных примеров):

select \*from userr where user\_id between 5 and 10



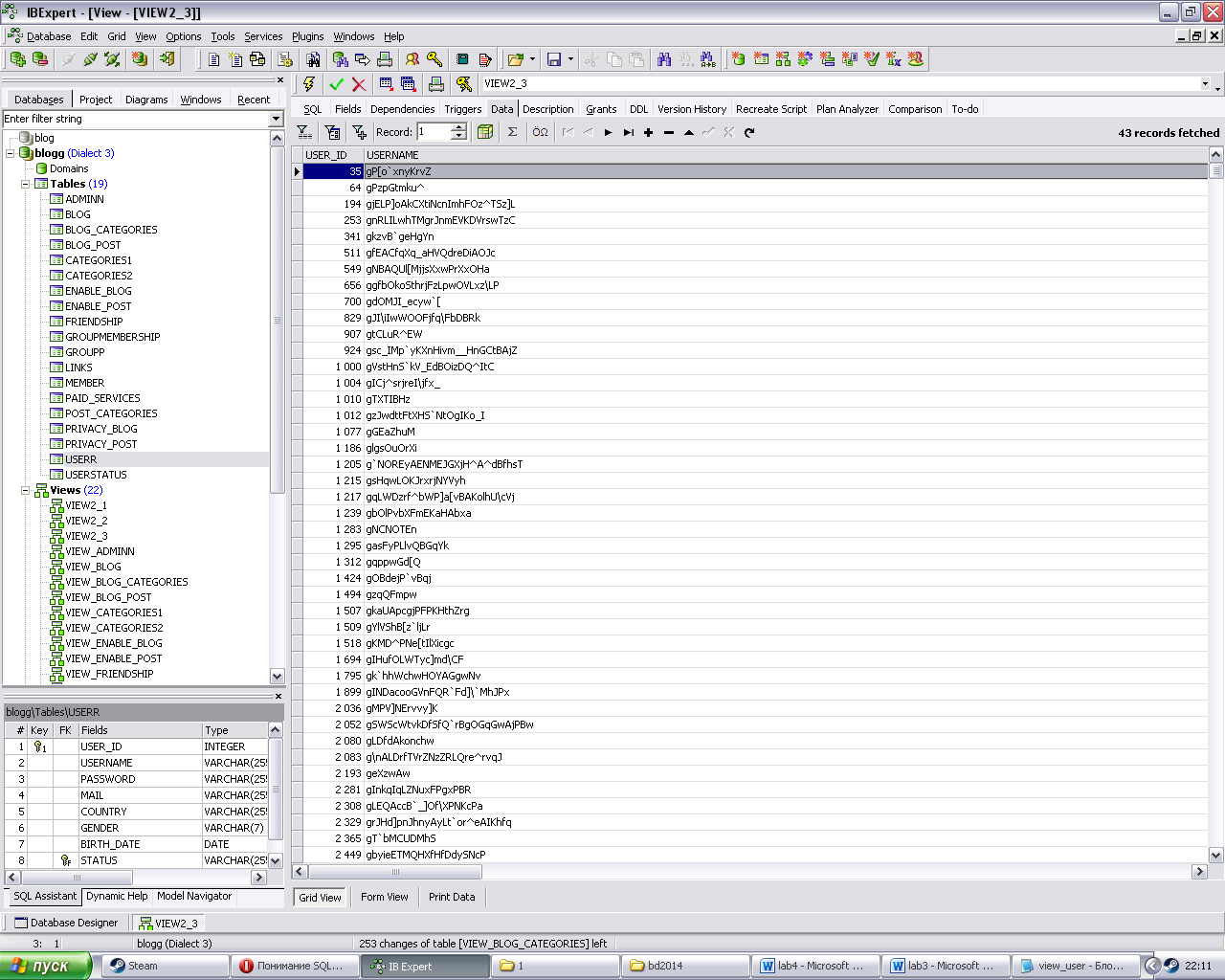
*Рис.1. Результат запроса.*

select \* from userr where user\_id in (15, 100,90000);



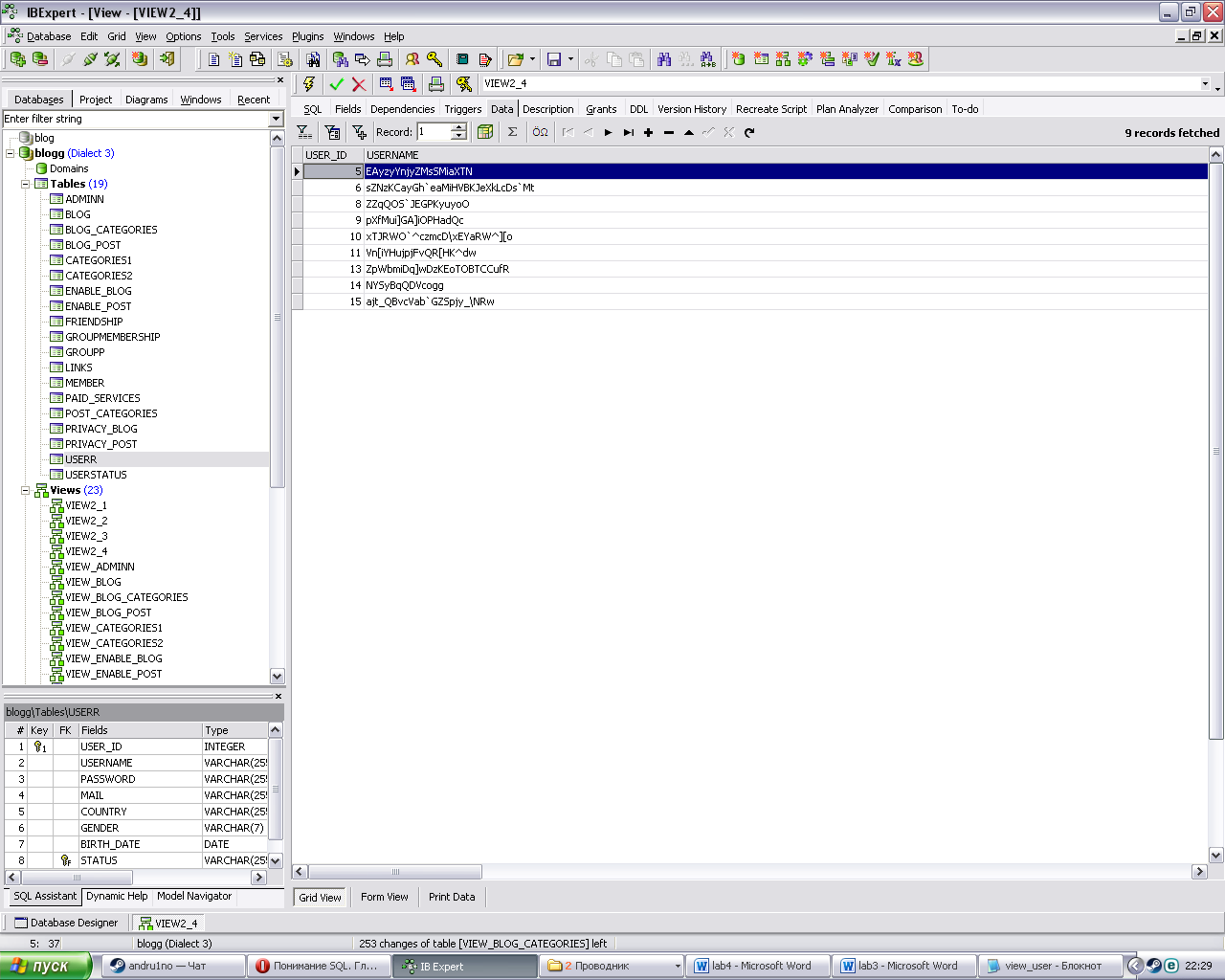
*Рис.2. Результат запроса.*

select \* from userr where username like ('g%');



*Рис.3. Результат запроса.*

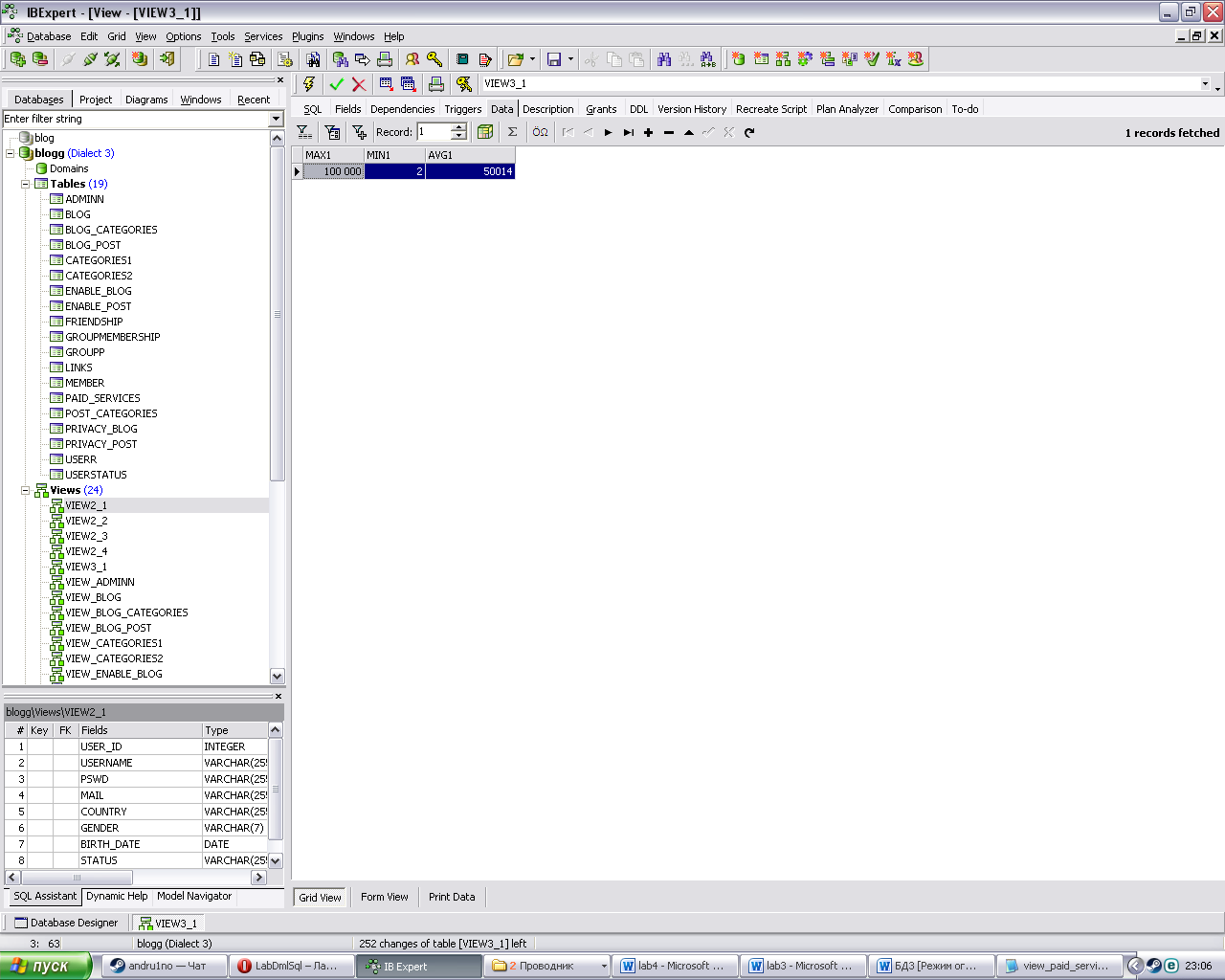
where (user\_id between 5 and 15)and not user\_id in (7, 12);



*Рис.4. Результат запроса.*

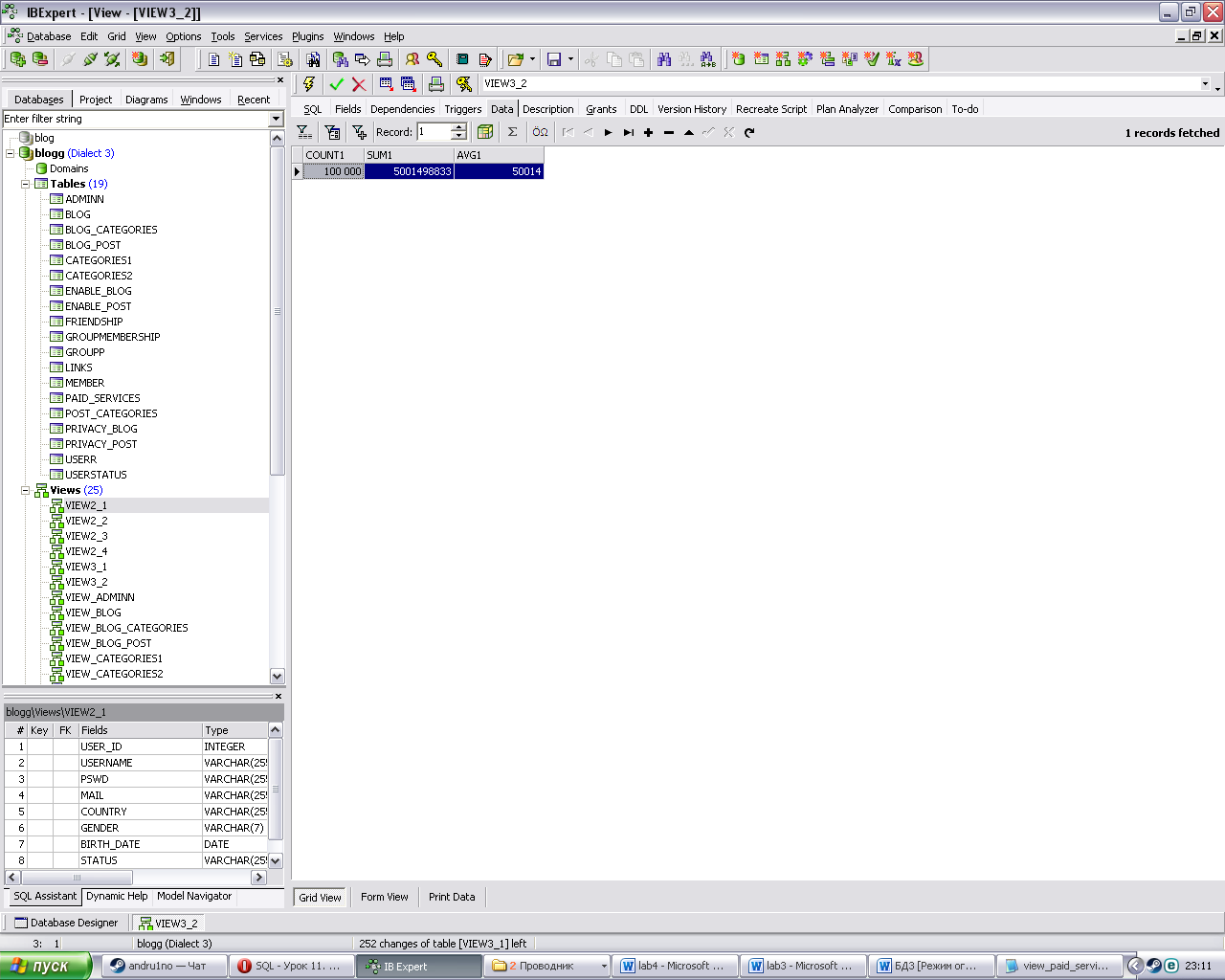
3) Создать в запросе вычисляемое поле:

select max(costs), min(costs), avg(costs) from paid\_services



*Рис.5. Результат запроса.*

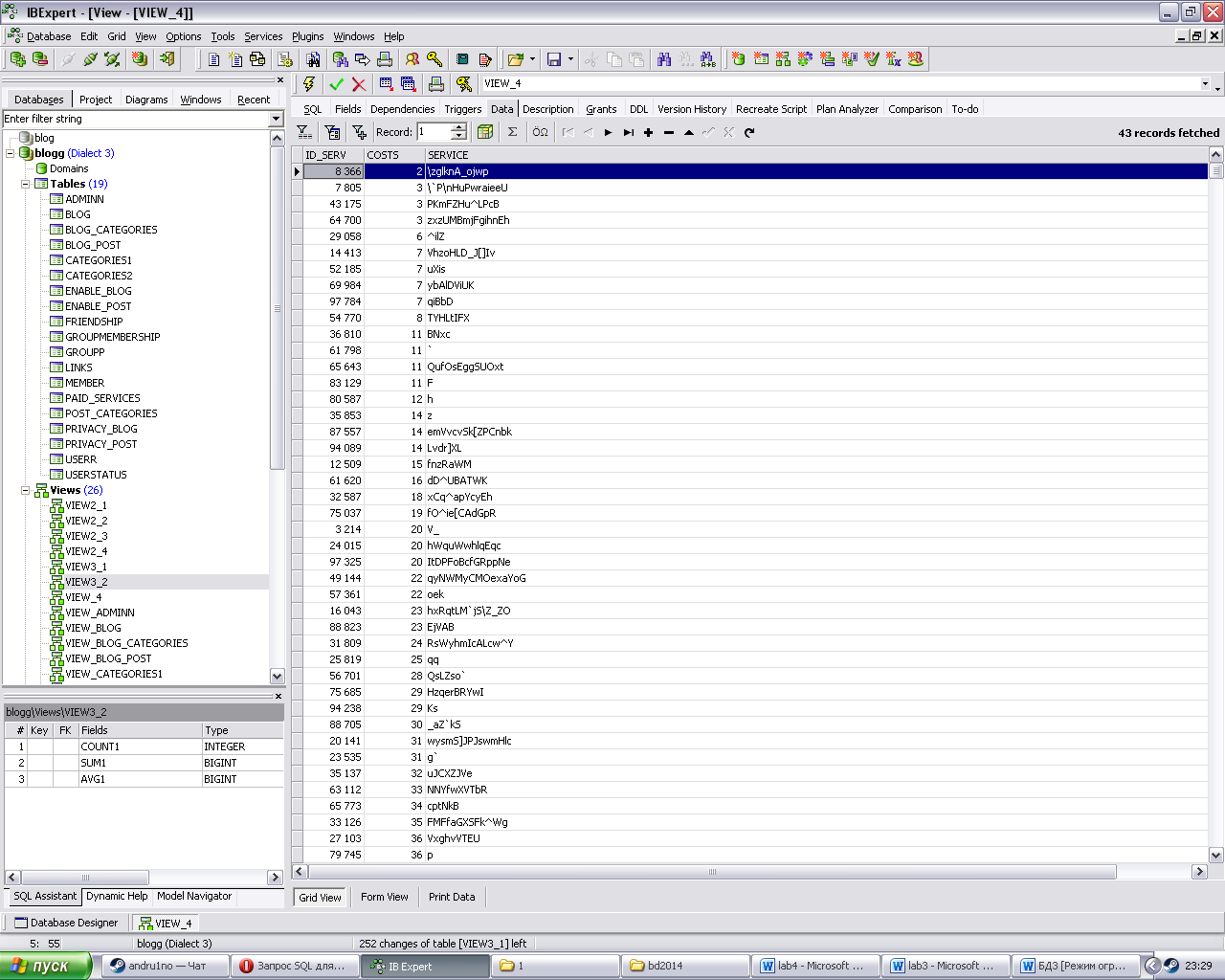
select count(costs), sum(costs), sum(costs)/count(costs) from paid\_services



*Рис.6. Результат запроса.*

4) Сделать выборку всех данных с сортировкой по нескольким полям:

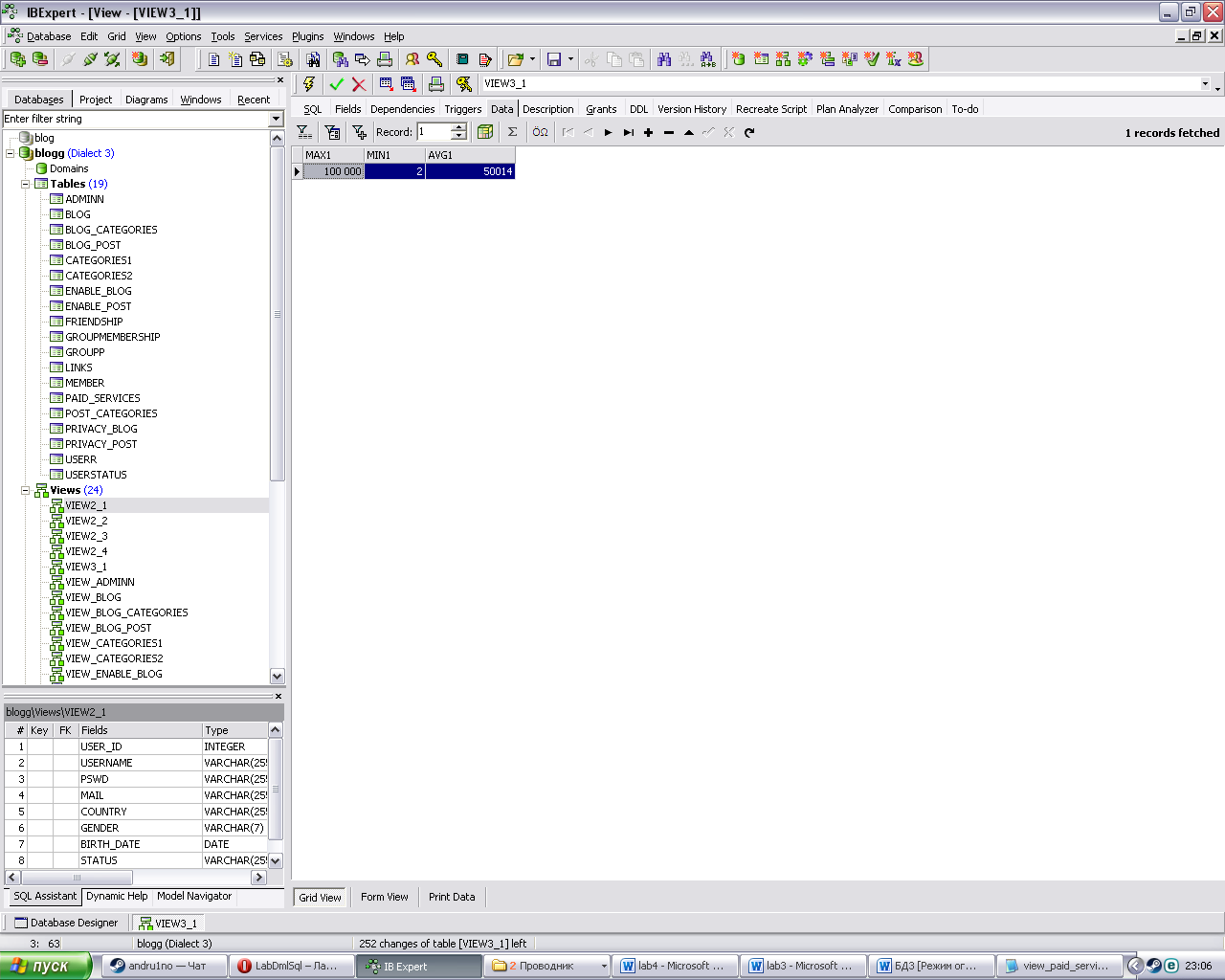
select \* from paid\_services order by costs, id\_serv



*Рис.7. Результат запроса.*

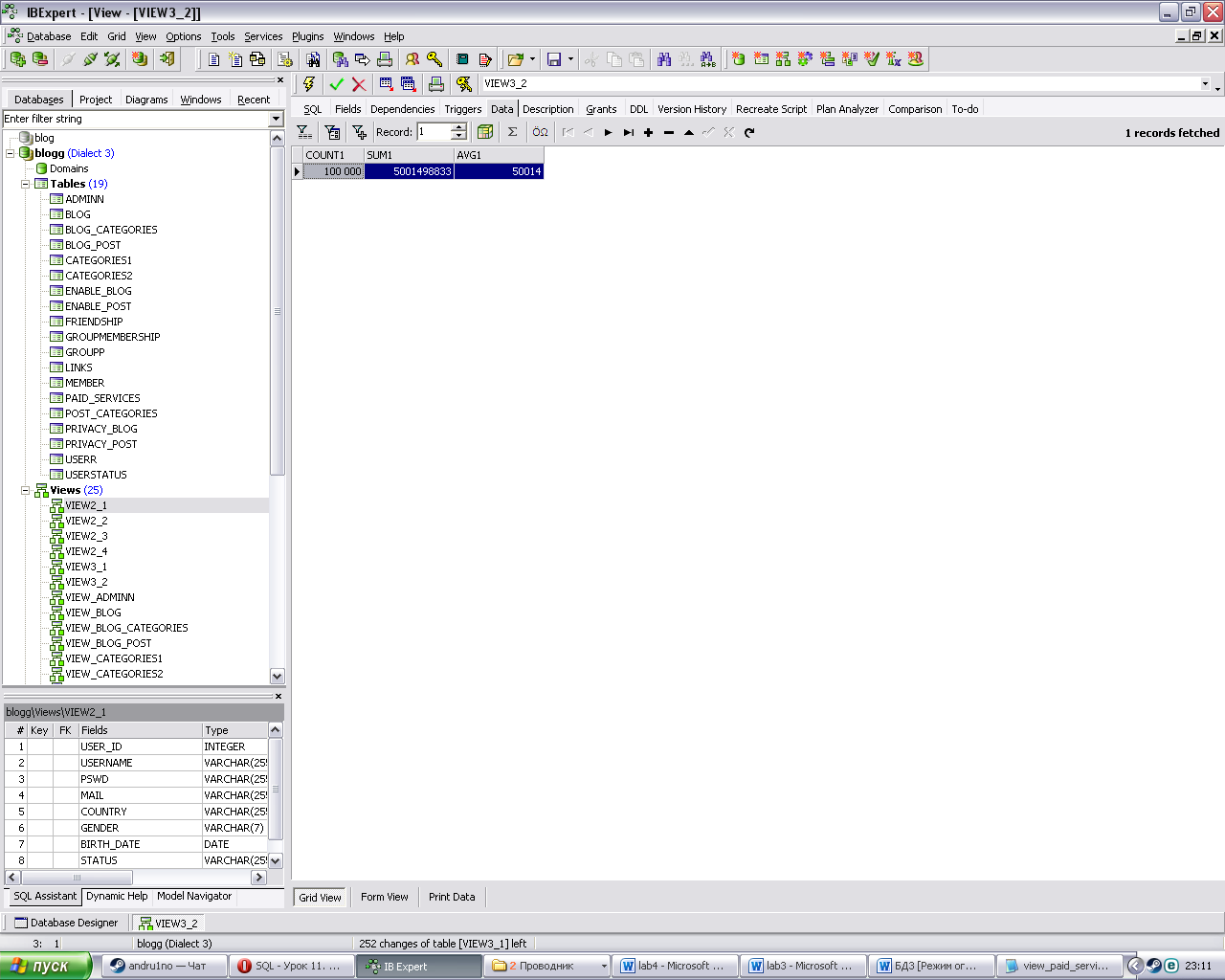
5) Создать запрос, вычисляющий несколько совокупных характеристик таблиц:

select max(costs), min(costs), avg(costs) from paid\_services



*Рис.8. Результат запроса.*

select count(costs), sum(costs), sum(costs)/count(costs) from paid\_services

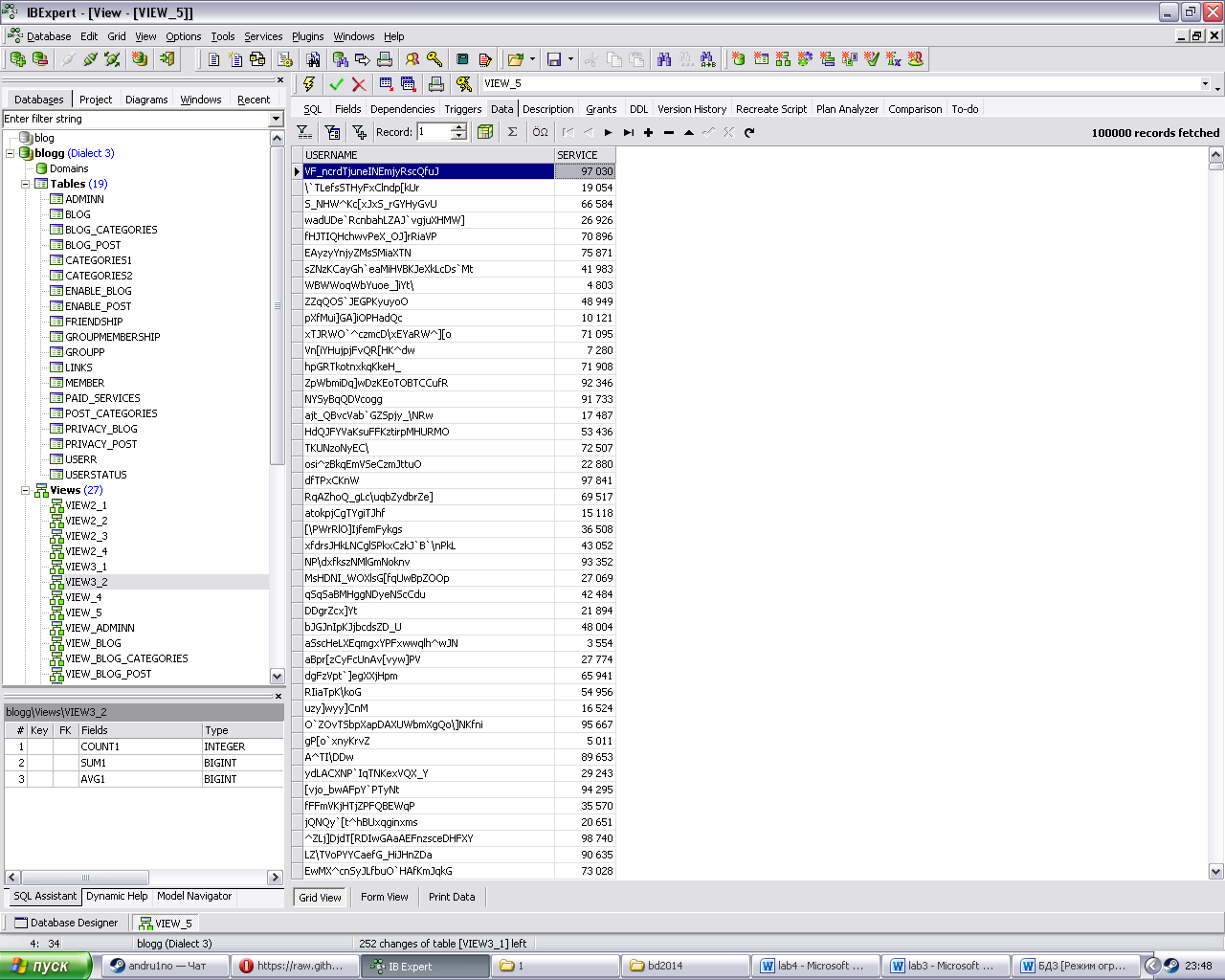


*Рис.9. Результат запроса.*

6) Сделать выборку данных из связанных таблиц (не менее двух примеров):

select userr.username, paid\_services.costs from userr, paid\_services

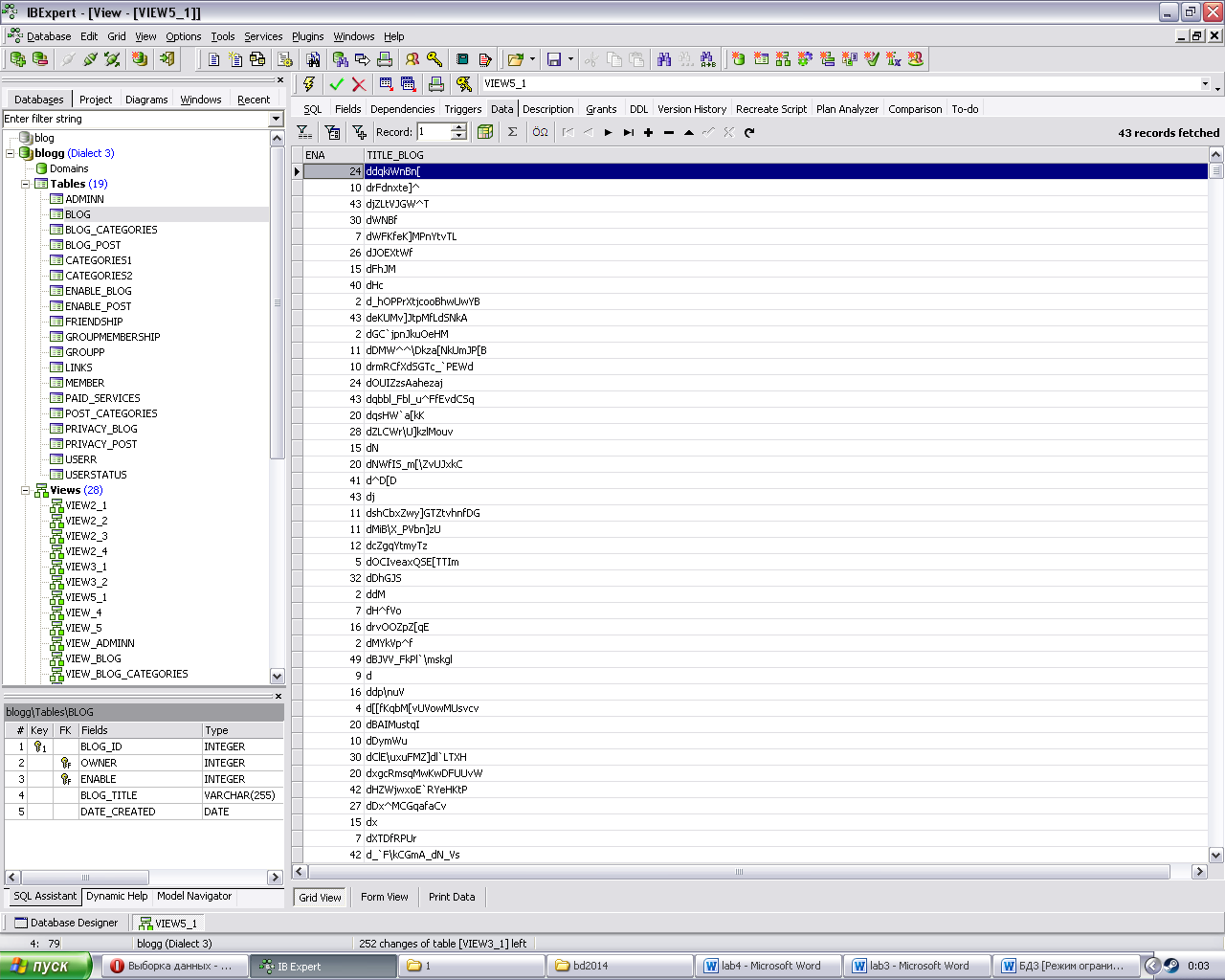
where userr.paid\_service=paid\_services.id\_serv;



*Рис.10. Результат запроса.*

select blog.blog\_title, privacy\_blog.ena\_blog from blog, privacy\_blog

where blog.enable=privacy\_blog.priv\_id and blog.blog\_title like 'd%';



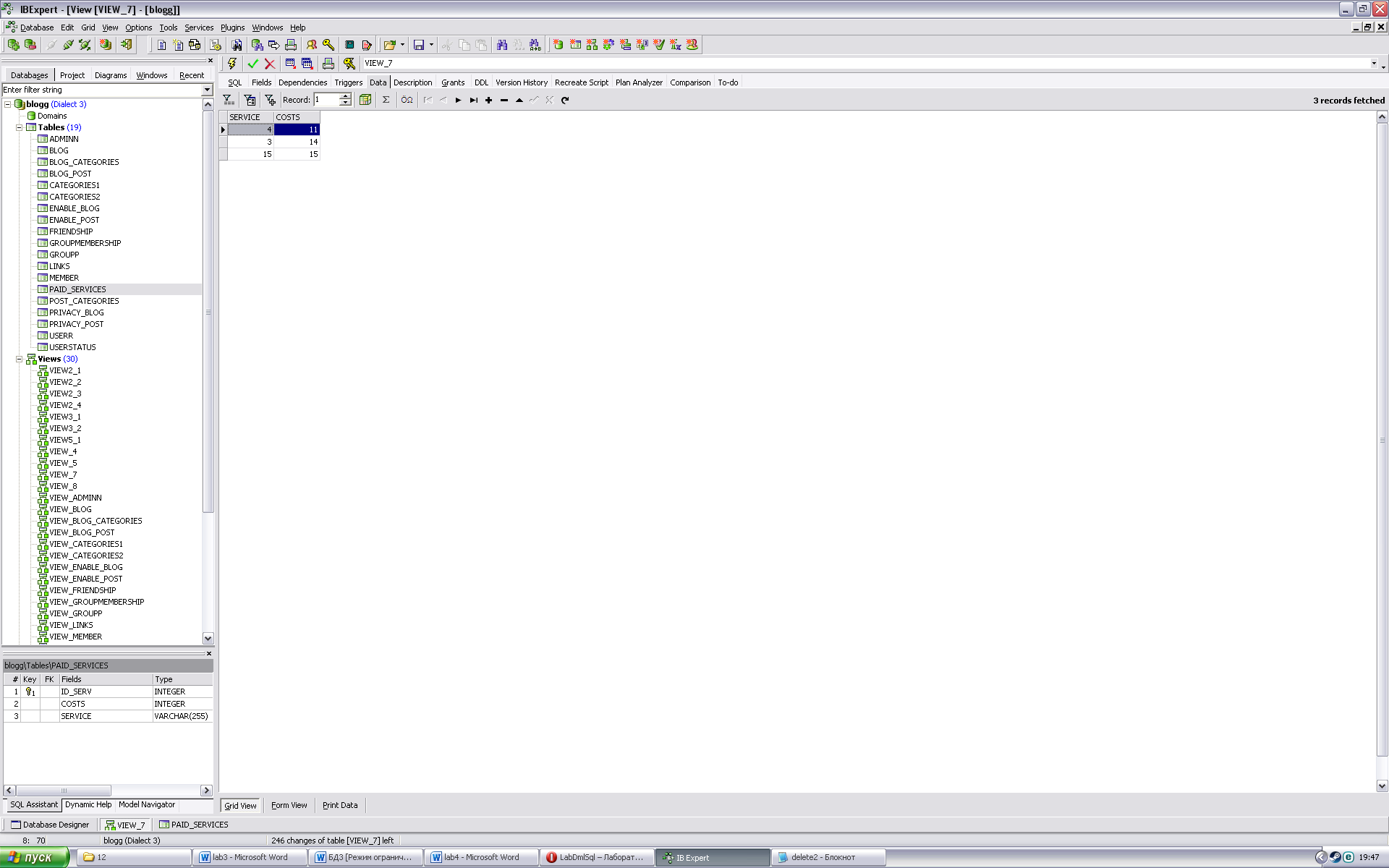
*Рис.11. Результат запроса.*

7) Создать запрос, рассчитывающий совокупную характеристику с использованием группировки, наложить ограничение на результат группировки:

select paid\_services.costs, count(paid\_services.service) from paid\_services

where paid\_services.costs<20

group by paid\_services.costs having count(paid\_services.service)>2



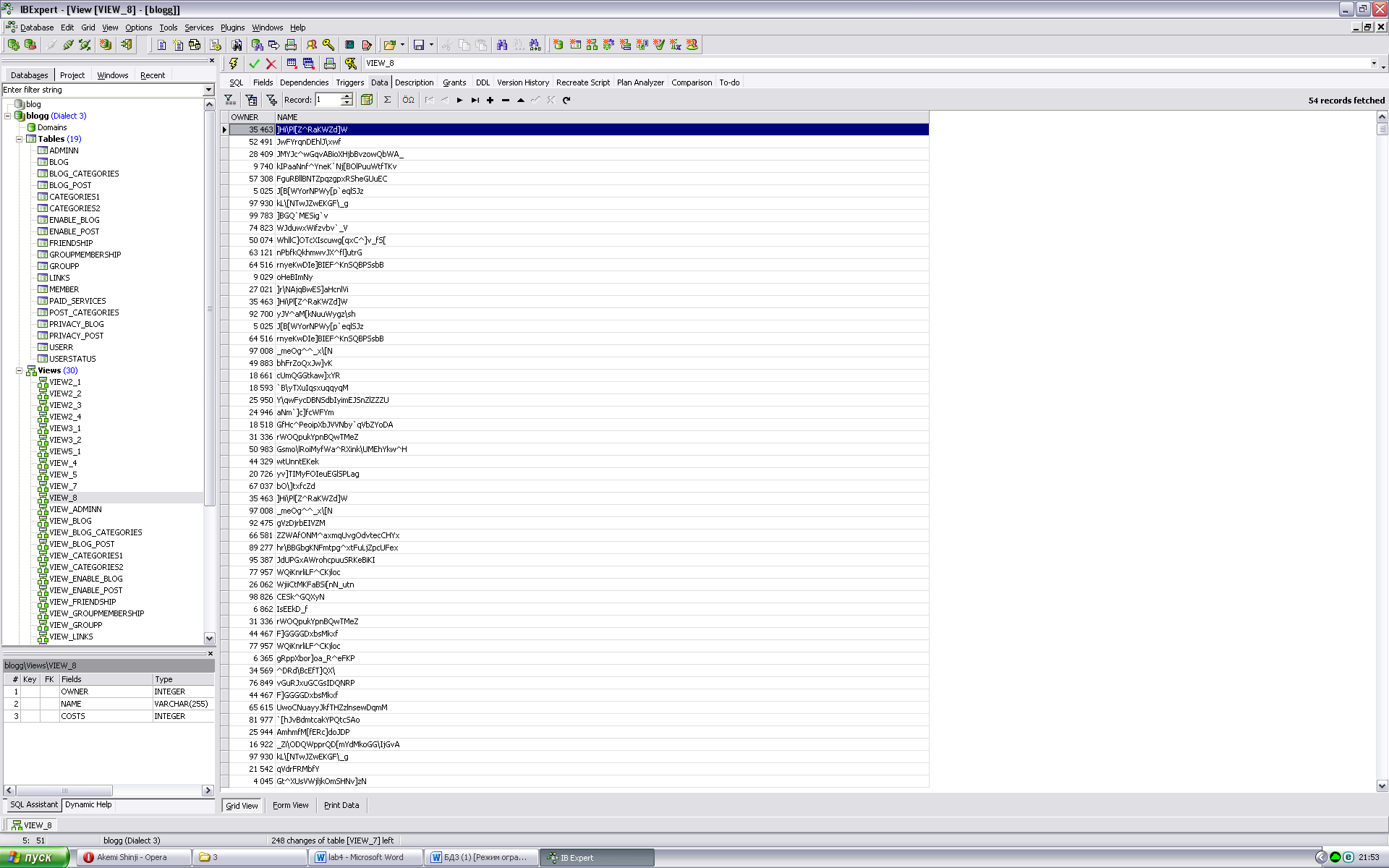
*Рис.12. Результат запроса.*

8) Придумать и реализовать пример использования вложенного запроса:

select blog.owner, userr.username from blog, userr

where blog.owner=userr.user\_id and userr.paid\_service in (select paid\_services.id\_serv from paid\_services

where paid\_services.id\_serv=userr.paid\_service and costs<100)



*Рис.13. Результат запроса.*

9) С помощью оператора INSERT добавить в каждую таблицу по одной записи:

Пример для таблицы userstatus:

begin

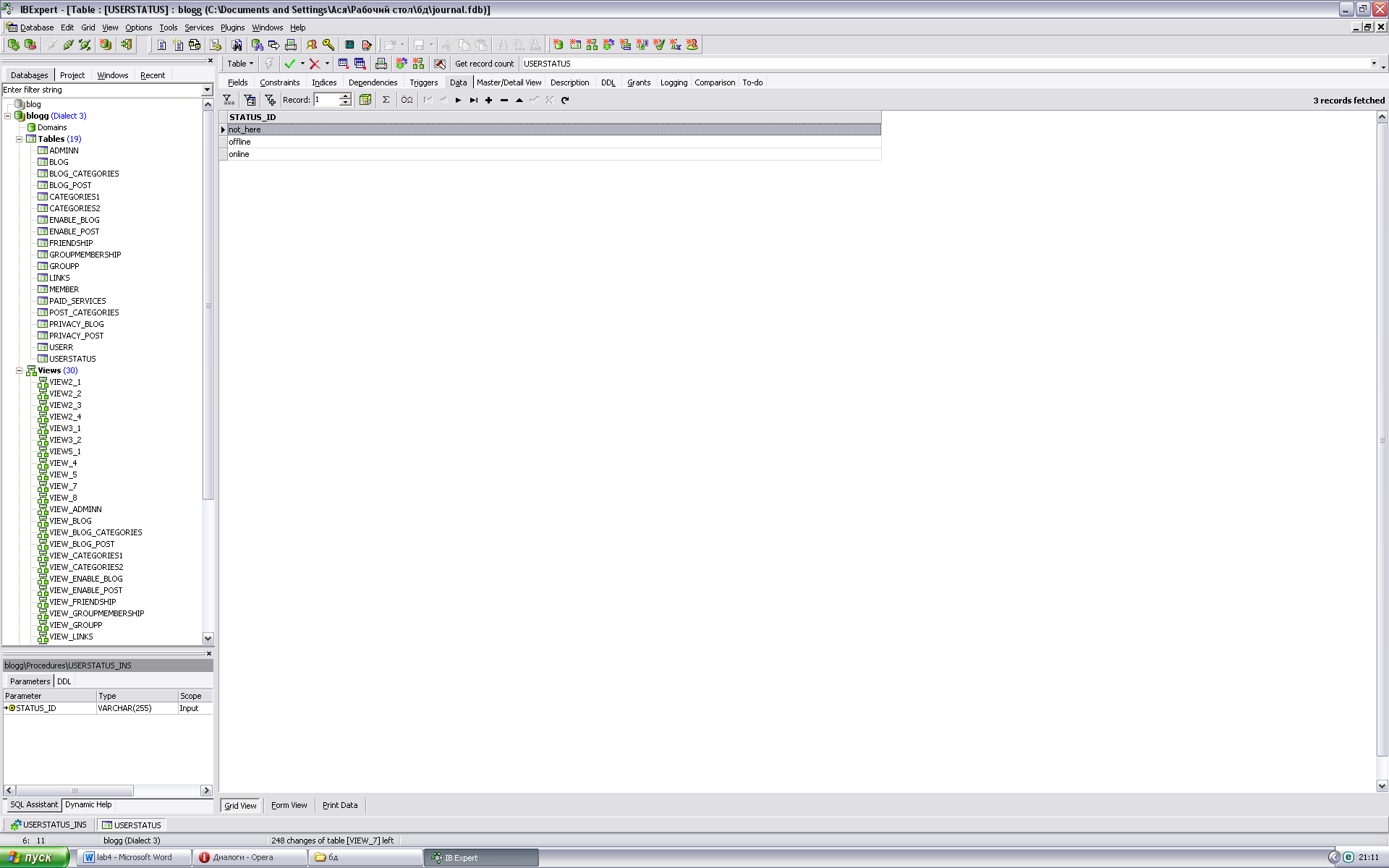
insert into userstatus (

status\_id)

values (

:status\_id);

end



*Рис.14. Результат выполнения.*

10) С помощью оператора UPDATE изменить значения нескольких полей у всех записей, отвечающих заданному условию:

begin

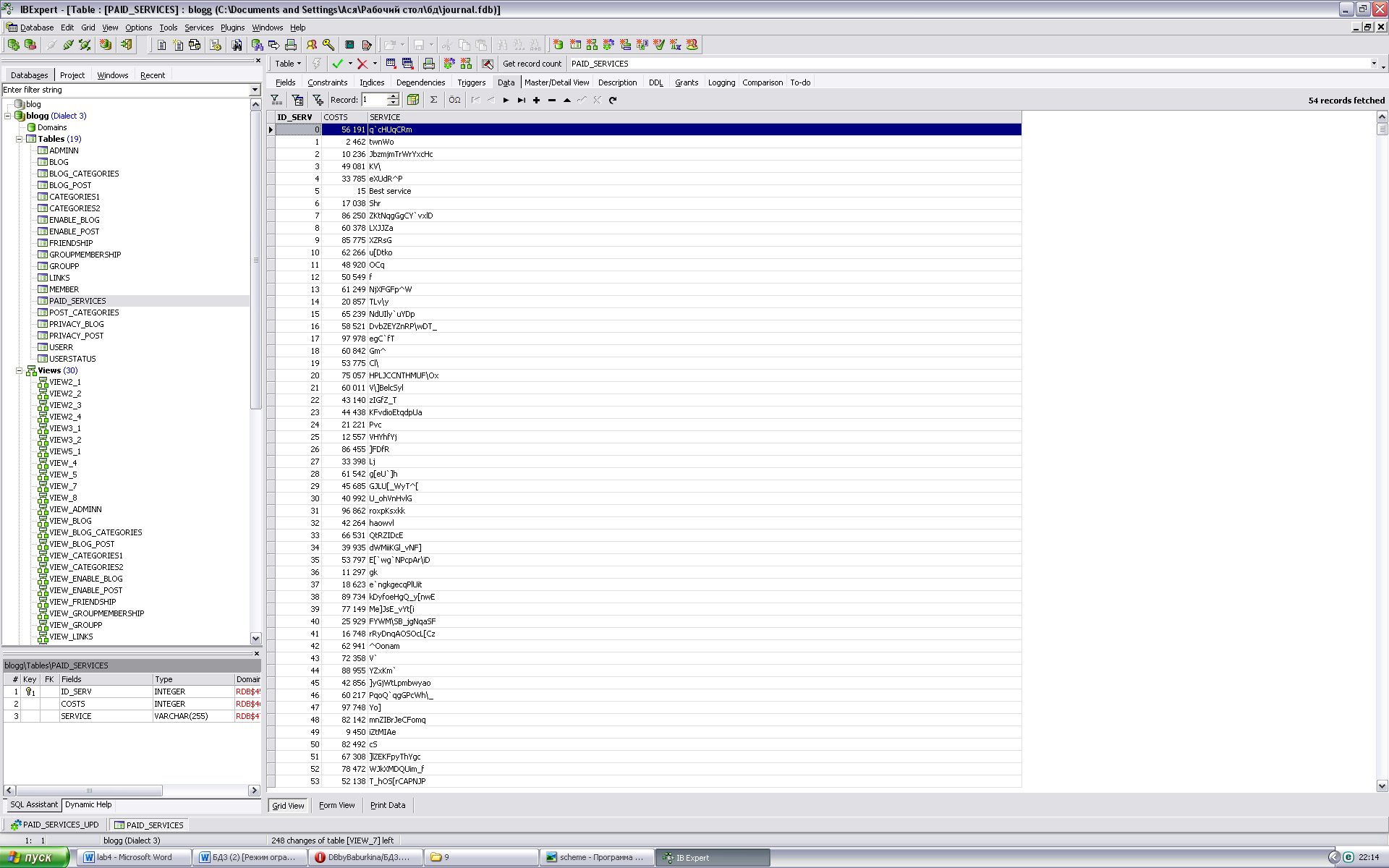
update paid\_services

set costs = 15,

service = ‘Best service’

where (id\_serv = 5);

end



*Рис.15. Результат выполнения.*

11) С помощью оператора DELETE удалить запись, имеющую максимальное (минимальное) значение некоторой совокупной характеристики:

declare variable min\_cost integer;

begin

select min(costs) from paid\_services into :min\_cost;

delete from paid\_services where costs=:min\_cost;

end

12) С помощью оператора DELETE удалить записи в главной таблице, на которые не ссылается подчиненная таблица (используя вложенный запрос):

begin

delete from groupmembership where groupmembership.userid not in (select member.usernumber from member where member.usernumber=groupmembership.userid);

end

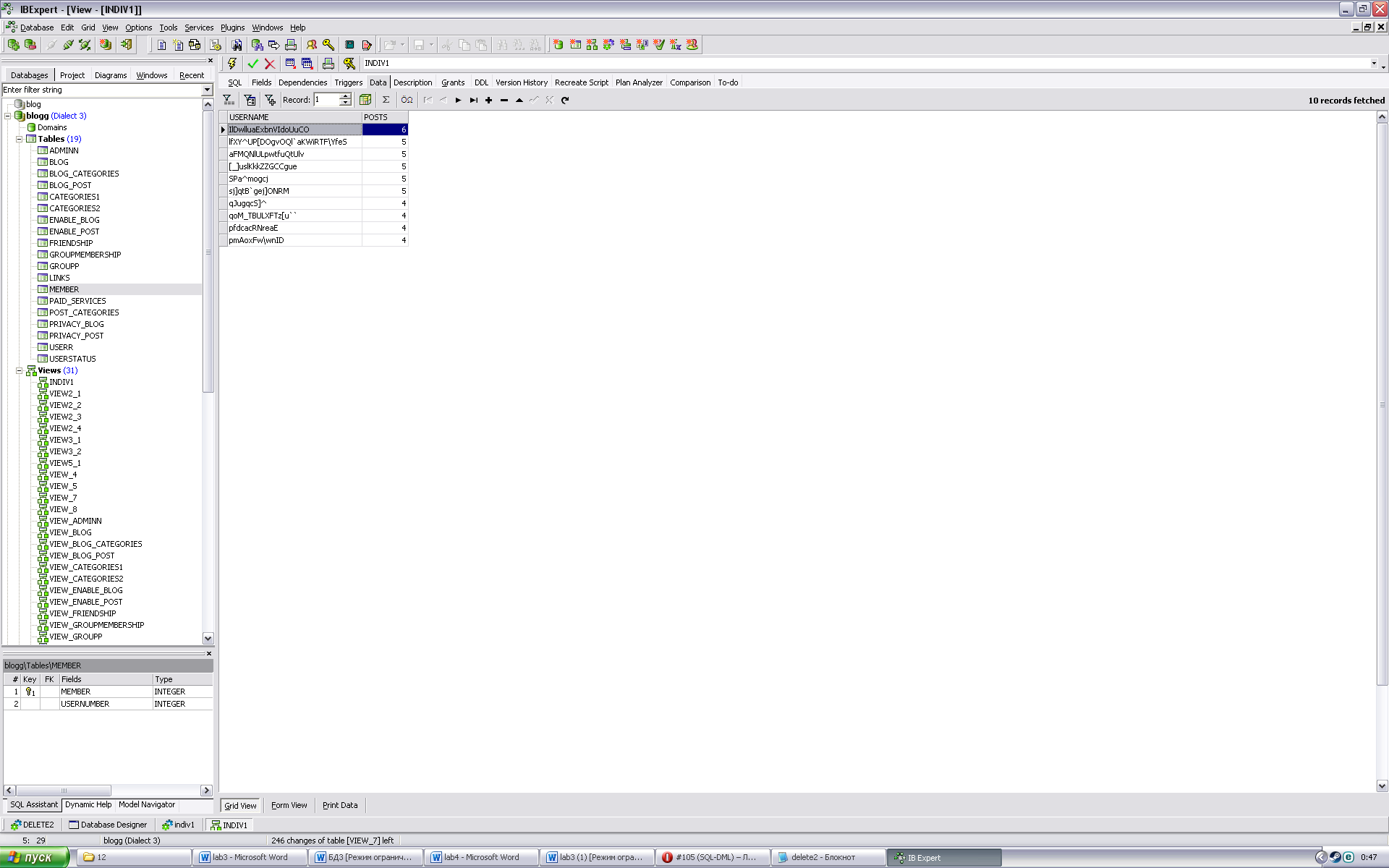
3.2. Индивидуальные запросы.

1) Вывести наиболее активных пользователей за выбранный период с количеством постов:

select first 10 u.username,count(b.blog\_id) from blog b, userr u

where b.owner=u.user\_id and b.date\_created between '2014-11-01' and '2014-12-31'

group by u.username order by 2 desc;



*Рис.16. Результат выполнения.*

2) Вывести наиболее оживленные разделы с количеством постов и комментариев.

select first 10 sum(c2.postnum), pc.categories, sum(cm.id\_com)

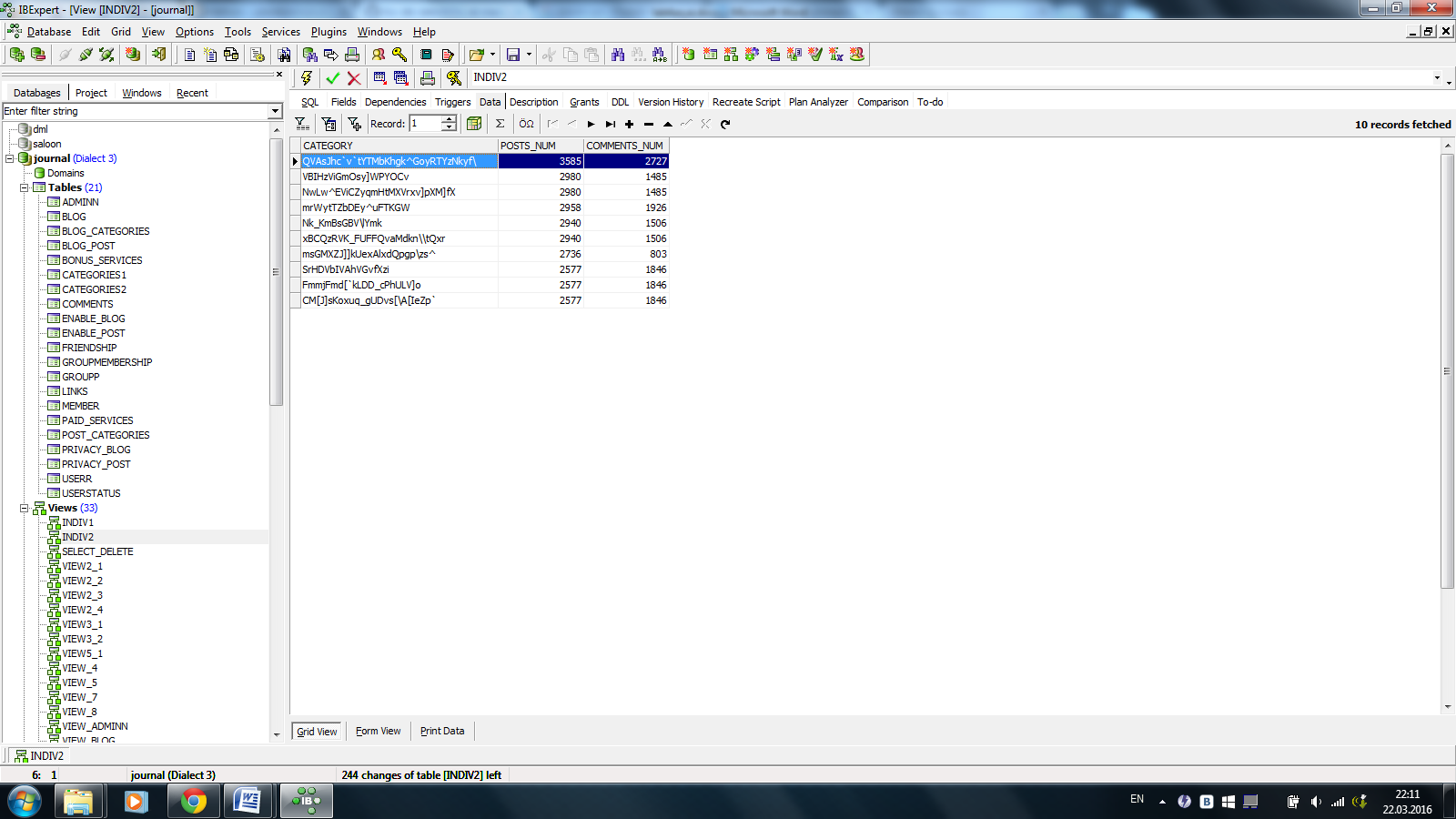
from categories2 c2, post\_categories pc, comments cm, blog\_post bp

where c2.postnum=bp.post\_id and cm.post\_num=bp.post\_id and pc.id\_cat=c2.catnum

group by pc.categories

order by 1 desc

;



*Рис.17.Результата выполнения.*

3) Удалить неиспользуемые разделы.

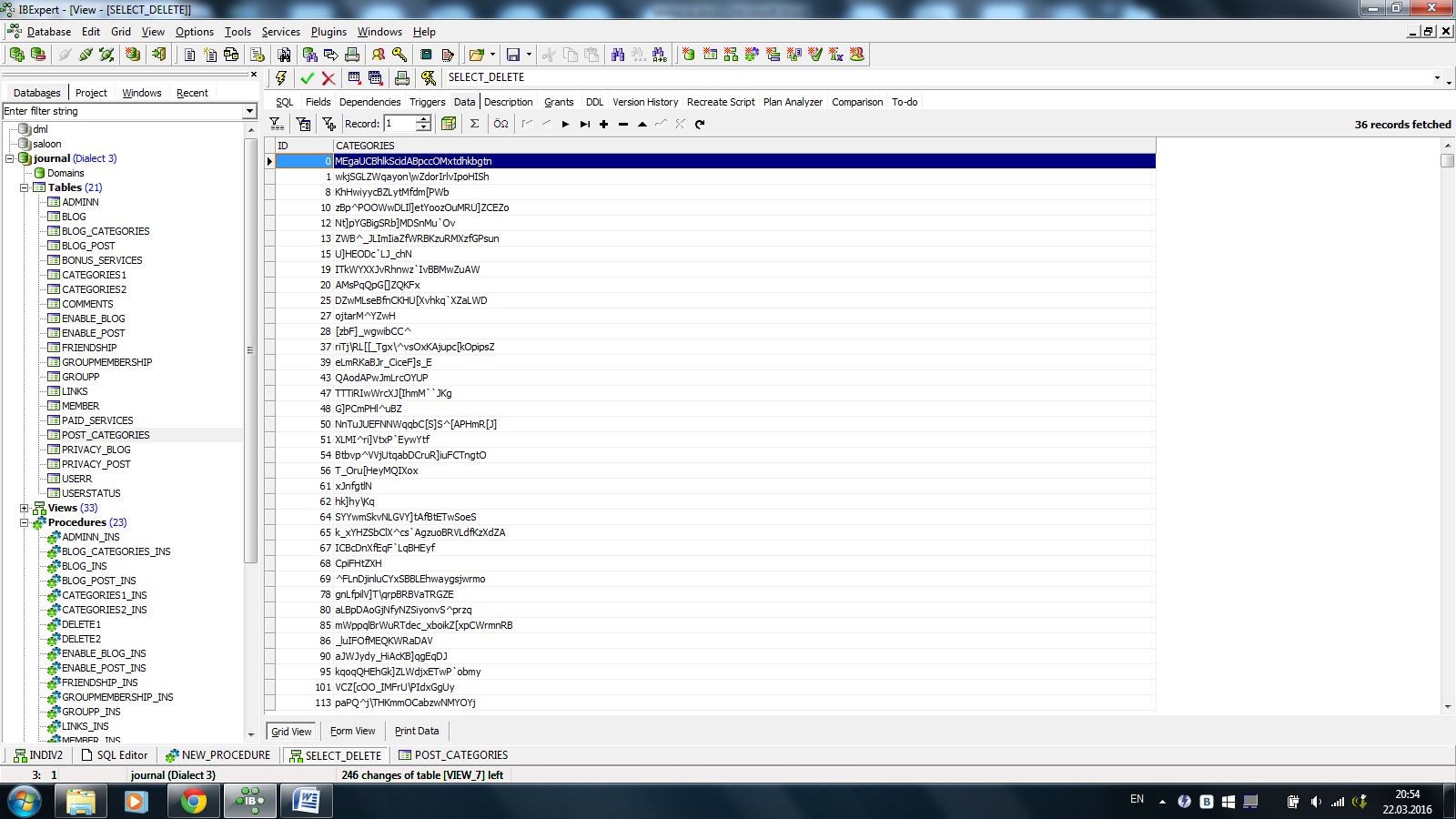
delete \* from post\_categories pc where not exists (select 1 from categories2 c2

where c2.catnum=pc.id\_cat)

Для проверки работоспособности запроса сначала выберем из таблицы Post\_categories неиспользующиеся разделы с помощью запроса:

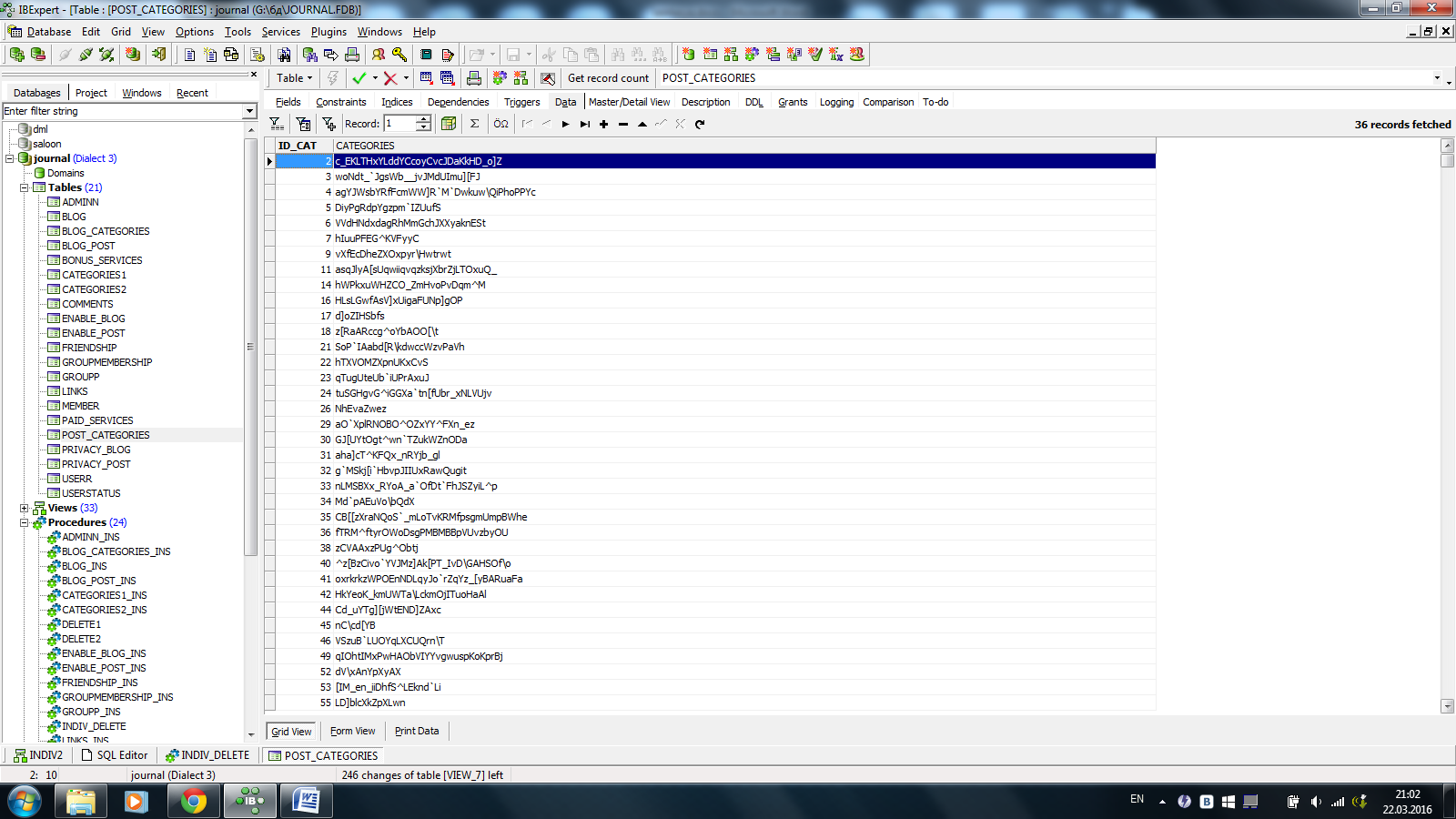
select \* from post\_categories pc where not exists (select 1 from categories2 c2

where c2.catnum=pc.id\_cat)



*Рис.18.Резульатат выполнения.*

Результат удаления:



*Рис.19.Результат выполнения.*

Выводы.

SQL-DML (Data Manipulation Language) представляет собой язык манипулирования данными, который позволяет добавлять, изменять, удалять и извлекать данные. Он позволяет произвести все необходимые действия с данными, в том числе с помощью select получить необходимые строки, которые в программе IBExpert выводятся в удобном формате.

С помощью данного языка можно выполнять различные запросы, но для сложных запросов ситуация усложняется. Например, если будет меняться какой-то параметр, то нужно будет выполнять запрос много раз – будет передача больших объемов данных.

В этом случае целесообразнее использовать хранимые процедуры.