**ВВЕДЕНИЕ**

Реляционная база данных появилась вследствие стремления сделать базу данных как можно более гибкой. РБД предоставила простой и эффективный механизм поддержания связей данных. Они представляются только в виде таблиц и обеспечивают единообразие представления данных. Реляционная база данных проста и доступна, имеет полную независимость данных, при изменении структуры таблицы в прикладных программах, для построения запросов и написания прикладных программ нет необходимости знания конкретной организации БД во внешней памяти.

1. **теоретические основы РЕЛЯЦИОННЫХ БАЗ ДАННЫХ**
   1. **Понятие реляционной базы данных**

Развитие реляционных баз данных началось в конце 1960-х гг., когда появились первые работы, в которых обсуждались возможности использования привычных для специалиста способов формализованного представления данных в виде таблиц. Некоторые специалисты такой способ представления информации называли таблицами решений, другие – табличными алгоритмами. Теоретики реляционных баз данных табличный способ представления информации называли даталогическими моделями.

* 1. **Нормализация таблиц реляционной базы данных**

Реляционная база данных представляет собой некоторое множество таблиц, связанных между собой. Число таблиц в одном файле или одной базе данных зависит от многих факторов, основными из которых являются:

* состав пользователей базы данных;
* обеспечение целостности информации (особенно важно в многопользовательских информационных системах);
* обеспечение наименьшего объем требуемой памяти и минимального времени обработки данных.
  1. **Преимущества** **и недостатки реляционной базы данных**

Достоинство реляционной модели данных заключается в простоте, по-нятности и удобстве физической реализации на ЭВМ. Именно простота и по-нятность для пользователя явились основной причиной их широкого использования. Проблемы же эффективности обработки данных этого типа оказались технически вполне разрешимыми. При проектировании реляционной БД применяются строгие правила, базирующие на математическом аппарате. Полная независимость данных. При изменении структуры реляционной изменения, которые требуют произвести в прикладных программах, минимальны.

* 1. **Основы реляционной алгебры**

*Реляционная алгебра* – замкнутая система операций над отношениями в реляционной модели данных. Операции реляционной алгебры также называют реляционными операциями [8].

*Унарные операции* служат для преобразования одного отношения в другое, являющееся подмножеством первого.

*Селекция* (горизонтальное подмножество) R создается из тех строк R, которые удовлетворяют заданным условиям, соединяемым логическими операторами AND, OR, NOT AND и NOT OR. Каждое условие может состоять из имени какого-либо столбца R, оператора сравнения (< , <= , = , <> , > , >=) и константы или имени другого столбца R. При этом сравниваемые столбцы должны быть определены на одном и том же домене (см. рис. 1.2).

* 1. **Проектирование связей между таблицами**

Процесс нормализации исходных таблиц баз данных позволяет создать оптимальную структуру информационной системы – разработать базу данных, требующую наименьших ресурсов памяти и, как следствие, обеспечивающую наименьшее время доступа к информации. В то же время разделение одной исходной таблицы на несколько требует выполнения одного из важнейших условий проектирования информационных систем – обеспечения целостности информации в процессе эксплуатации базы данных.

1. **ПРАКТИЧЕСКАЯ ЧАСТЬ**
   1. **Исследование предметной области**

*Цель создания БД:*

Требуется разработать базу данных для автоматизации учета получения информации о номерах телефонов организаций и предприятий города Троицка Челябинской области.

*Описание объектов предметной области*

Объект «Справочник» хранит информацию о наименованиях справочников, их идентификаторах, так как необходимо учитывать информацию о предприятиях в различных справочниках города Троицка.

idСправочник;

Наименование справочника.

* 1. **Проектирование и создание БД**

Рисунок 1.10 Схема отношений базы данных «Телефонный справочник города»

* 1. **Хранимые процедуры и триггеры**
  2. **Реляционная алгебра в запросах к БД «Телефонный справочник города»**

*Селекция* (горизонтальное подмножество) таблицы `organizations` создается из строк с полями `NameOrg` и `MainNumber`, которые удовлетворяют условию idDirector=1, для вывода всех школ города Троицка Челябинской области (см. рис. 2.1).

select NameOrg,MainNumber

from `organizations`

where idDirector=1

group by MainNumber;

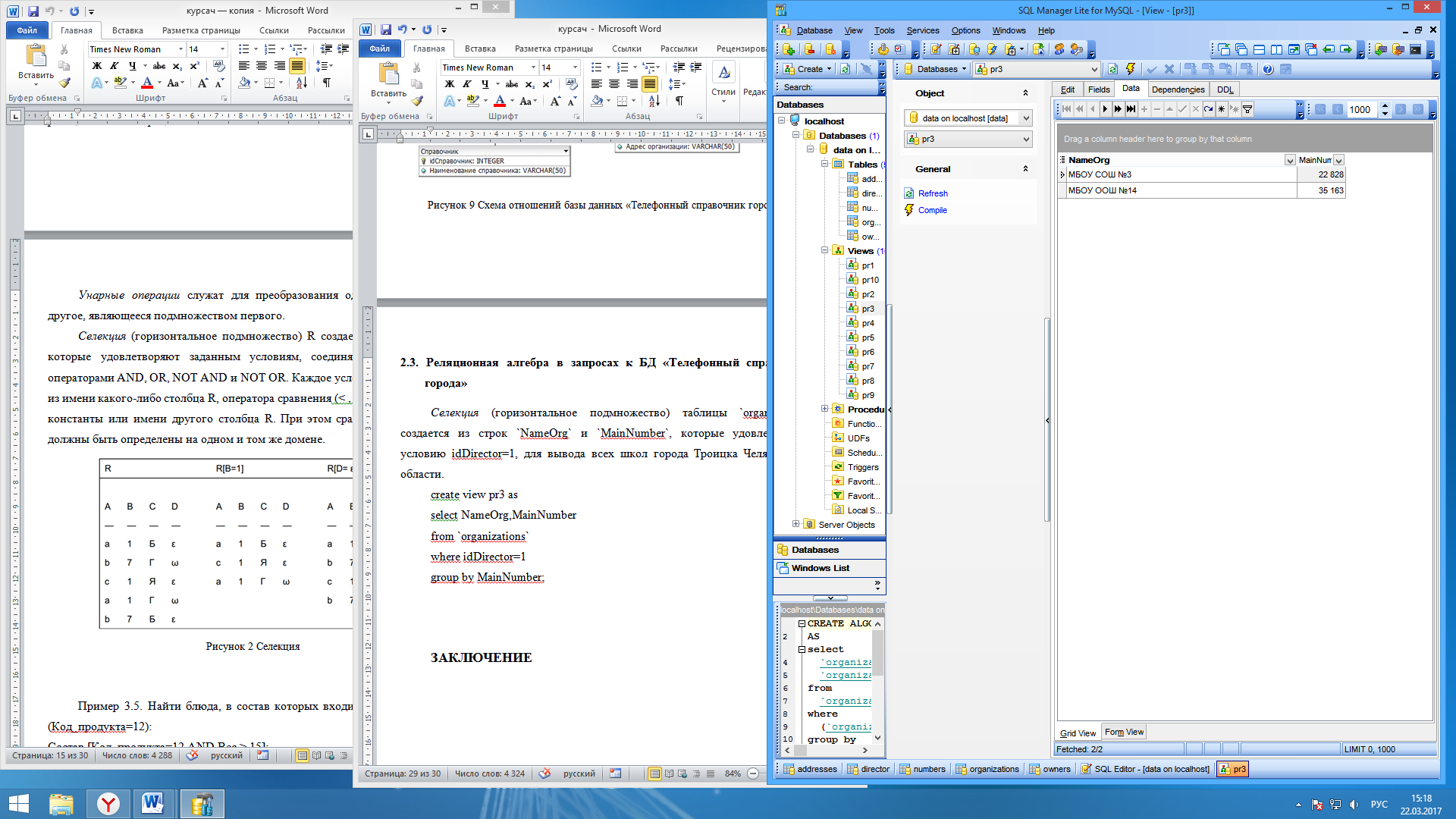


Рисунок 2.1 Селекция таблицы organizations

*Проекция* (вертикальное подмножество) таблицы `organizations` создается из полей `NameOrg` и `MainNumber`, с последующим исключением избыточных дубликатов строк (см. рис. 2.2).

select o.NameOrg, o.MainNumber

from `organizations` o

where idDirector=1 or idDirector=5 or idDirector=9

group by MainNumber

order by o.NameOrg ASC;

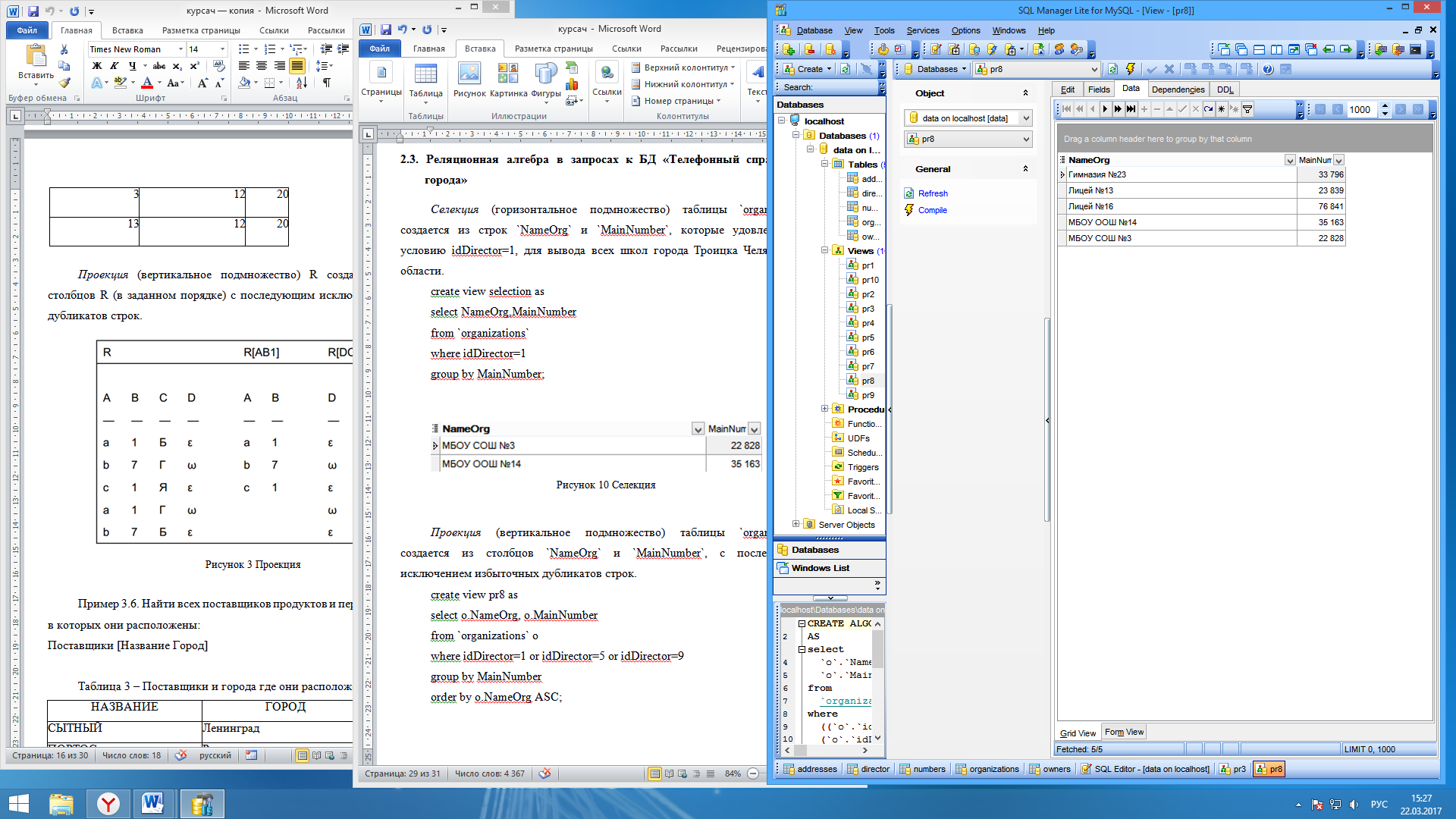


Рисунок 2.2 Проекция таблицы organizations

Бинарное отношение объединение, разность, пересечение в базе данных «Телефонный справочник города» не реализуется, в виду того, что отсутствует отношение с одинаковыми значениями

*Соединение* таблиц `organizations`, `addresses` и `owners`, позволяет выделить его определенное подмножество строк `NameOrg`, `MainNumber`, `AddressOrg`, `FIOowner`(см. рис. 2.3).

select o.NameOrg, o.MainNumber, w.AddressOrg, v.FIOowner

from `organizations` o, `addresses` w, `owners` v

where (v.FIOowner like ('%Ижевская%')) and (o.idAddress = w.idAddress) and (o.idOwners = v.idOwners);

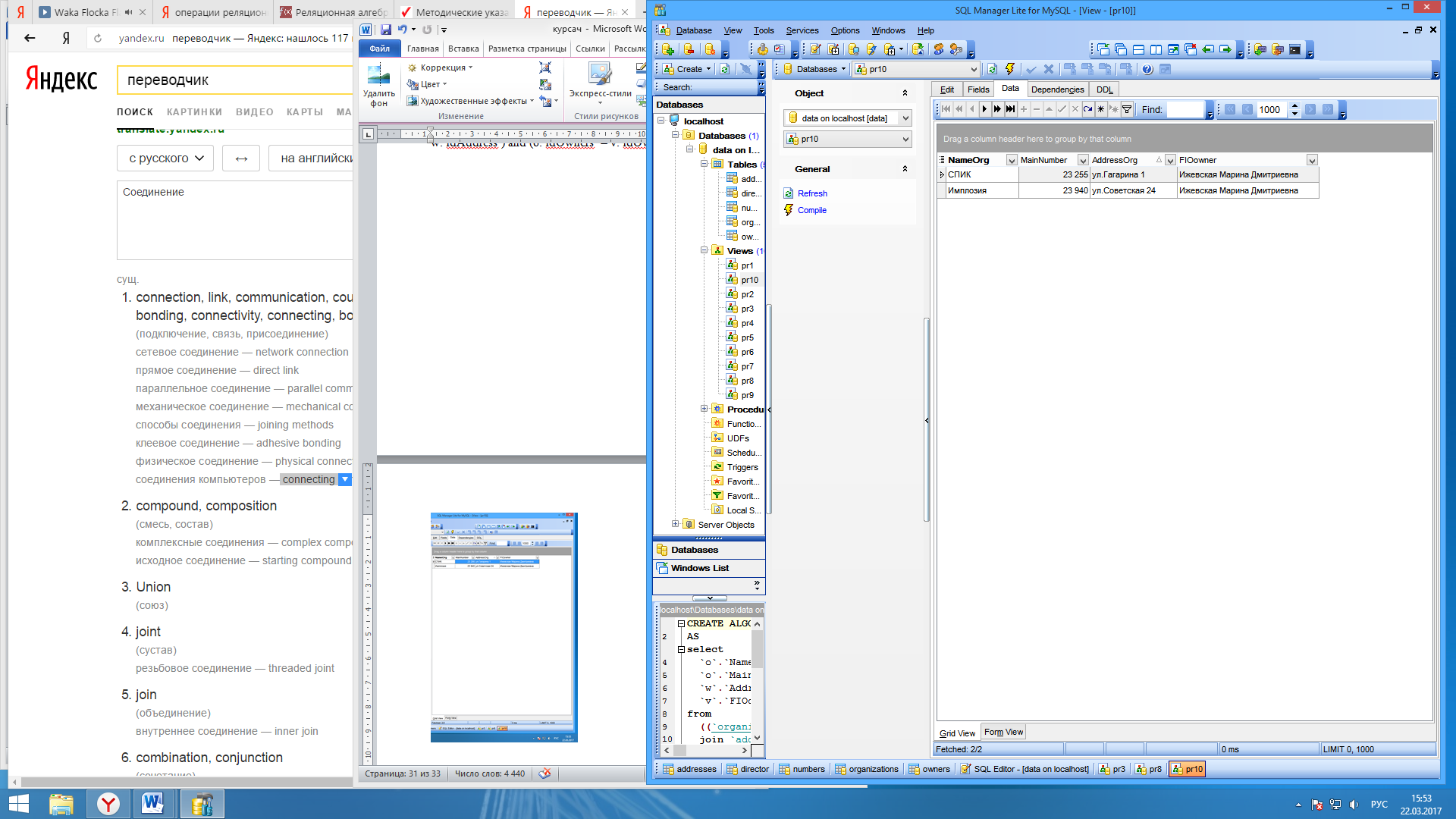


Рисунок 2.3 Соединение таблиц organizations, addresses и owners

*Декартово произведение* двух таблиц `director`, `addresses` – это множество упорядоченных пар из кортежей. Оно содержит все возможные строки, составленные сцеплением строки из таблицы `director` со строкой из таблицы `addresses` (см. рис. 2.4).

select \* from `director`, `addresses`

where idDirector=1 or idDirector=5 or idDirector=9;

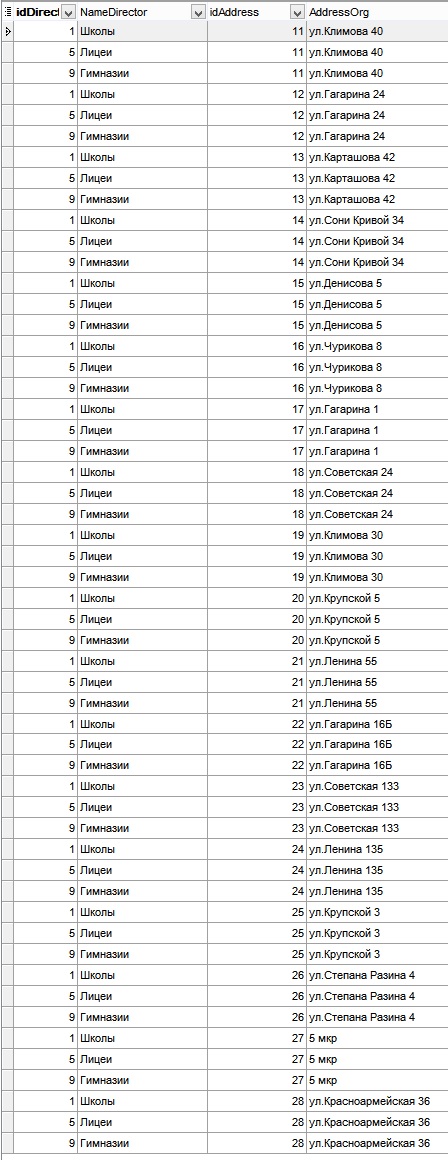


Рисунок 2.4 Декартово произведение таблиц director и addresses

*Делением* двух таблиц `director` и `numbers` является набор строк таблицы `numbers`, которые соответствуют комбинации всех строк таблицы `director`. В результирующем отношении присутствуют только те атрибуты таблицы `numbers`, которых нет в таблице `director` (см. рис. 2.5).

SELECT DISTINCT NameDirector, idDirector from `director`

where NOT EXISTS (Select \* from `numbers`

where `director`.`idDirector`=`numbers`.`idDirector`);

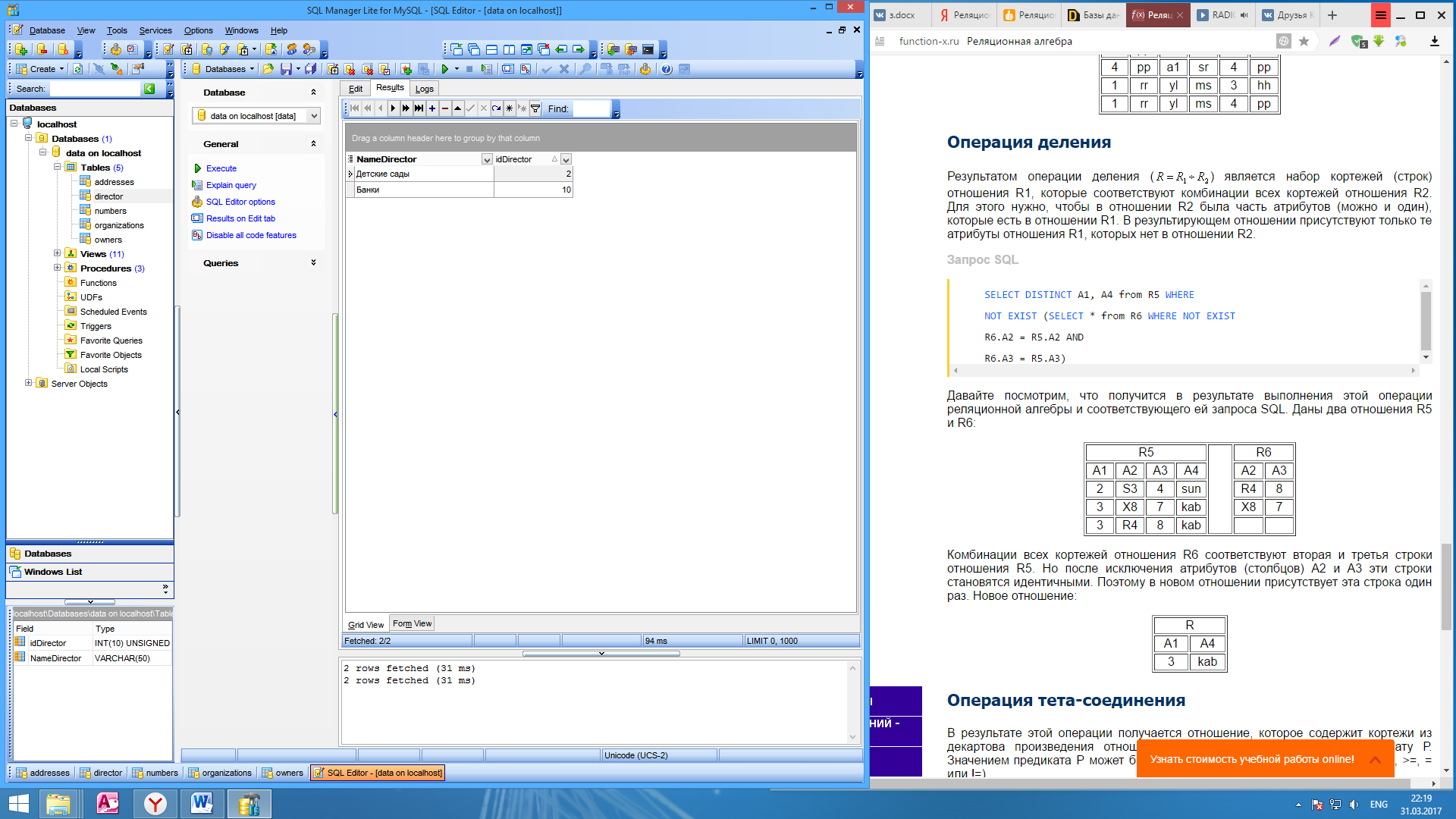


Рисунок 2.5 Деление таблиц director и numbers

**ЗАКЛЮЧЕНИЕ**

**СПИСОК ЛИТЕРАТУРЫ**

1. Дейт К.Дж. Введение в системы баз данных, 8-е издание / К.Дж.Дейт – М.: ИНФРА, 2009. – 400 с.