

Введение в базы данных

Белозубов Александр

Владимирович

belozubov@corp.ifmo.ru

Список литературы

- Базы данных. Проектирование, реализация и сопровождение. Теория и практика / Т. Коннолли, К. Бегг, А. Страчан ; [пер. с англ. канд. физ.-мат. наук Ю. Г. Гордиенко, А. В. Слепцов ; под ред. А. В. Слепцова] .— 2-е изд., испр. и доп. — М. [и др.] : Издательский дом "Вильямс", 2000 .— 1112 с. : ил.
- Введение в реляционные базы данных / В. В. Кириллов, Г. Ю. Громов ; [реценз. А. А. Бобцов] .— СПб. : БХВ-Петербург, 2012 .— 454 с. : ил. + 1 электрон. опт. диск (CD-ROM)
- Базы данных. Модели, разработка, реализация : [учебное пособие] / Т. С. Карпова .— СПб. [и др.] : Питер, 2001 .— 303, [1] с. : ил.

База данных

- База данных - это система специальным образом организованных данных (баз данных), программных, технических, языковых средств, предназначенных для обеспечения централизованного накопления и коллективного многоцелевого использования данных.
- База данных является общим корпоративным ресурсом и хранит не только данные, но и их описания. По этой причине базу данных еще называют *набором интегрированных записей с самоописанием*.
- Описание данных называется **системным каталогом** (system catalog), или **словарем данных** (data dictionary), а сами элементы описания принято называть **метаданными** (metadata), т.е. "данными о данных".
- Именно наличие самоописания данных в базе данных обеспечивает **независимость между программами и данными**.
- база данных — это совокупность описаний объектов реального мира и связей между ними, актуальных для конкретной прикладной области.

СУБД

СУБД - это совокупность языковых и программных средств, обеспечивающих для выполнение всех операций, связанных с организацией хранения данных, их корректирования и доступа к ним.

Позволяет определять базу данных, что осуществляется с помощью **языка определения данных** (DDL - Data Definition Language). Язык DDL предоставляет пользователям средства указания типа данных и их структуры, а также средства задания ограничений для информации, хранимой в базе данных.

Позволяет вставлять, обновлять и извлекать информацию из базы данных, что осуществляется с помощью **языка управления данными** (DML - Data Manipulation Language). Наличие централизованного хранилища всех данных и их описаний позволяет использовать язык DML как общий инструмент организации запросов, который иногда называют **языком запросов**.

Предоставляет контролируемый доступ к базе данных:

- системы обеспечения безопасности, предотвращающей несанкционированный доступ к базе данных со стороны пользователей;
- системы поддержки целостности данных, обеспечивающей непротиворечивое состояние хранимых данных;
- системы управления параллельной работой приложений, контролирующей процессы их совместного доступа к базе данных;
- системы восстановления, позволяющей восстановить базу данных до предыдущего непротиворечивого состояния, нарушенного в результате сбоя аппаратного или программного обеспечения;

Требования к современным СУБД

- функциональность
- производительность
- защищенность
- целостность
- масштабируемость
- надежность (катастрофоустойчивость),
- реактивность

Преимущества использования Баз Данных

- Независимость данных – сокращение размеров программной поддержки (внутри отдельных программ)
- Увеличение эффективности разработки приложений
- Возможность создания и использования стандартов
- Минимальная избыточность хранения данных
- Увеличение плотности данных и совместного доступа к данным
- Улучшенный доступ к данным и их соответствие конкретным решаемым задачам
- Увеличение качества данных
- Безопасность, сохранение и восстановление

Архитектура Баз Данных



Логическая и физическая независимость данных

- Основным назначением трехуровневой архитектуры является обеспечение **независимости от данных**, которая означает, что изменения на нижних уровнях никак не влияют на верхние уровни. Различают два типа независимости от данных: **логическую** и **физическую**.
- **Логическая независимость от данных** - означает полную защищенность внешних схем от изменений, вносимых в концептуальную схему
- **Физическая независимость от данных** - означает защищенность концептуальной схемы от изменений, вносимых во внутреннюю схему

Классификация Баз Данных

- ❑ ***По модели данных***
 - На основе инвертированных списков
 - Иерархические
 - Сетевые
 - Реляционные
 - Объектно-ориентированные и мультимедийные (постреляционные)
- ❑ ***По количеству пользователей***
 - Персональные (настольные)
 - Уровня рабочей группы
 - Масштаба предприятия
 - Корпоративные
- ❑ ***По организации системы***
 - Распределённые
 - Централизованные
- ❑ ***По характеру хранимой информации***
 - Фактографические
 - Полнотекстовые

Классификация БД по *характеру хранимой информации*

- Фактографические БД – содержат краткие сведения об описываемых объектах, представленные в строго определенном формате(картотеки);**
- Документальные БД – содержат обширную информацию самого разного типа: текст, графику, видео и звук(архив).**

Классификация БД по способу хранения данных

- Централизованные - вся информация хранится на одном компьютере. Это может быть автономный ПК или сервер сети, к которому имеют доступ пользователи - клиенты;
- Распределенные - используются в локальных и глобальных компьютерных сетях. В таком случае разные части базы хранятся на разных компьютерах.

Классификация БД по *структуре организации данных*

- Реляционные (табличные БД)
- Иерархические.
- Сетевые.

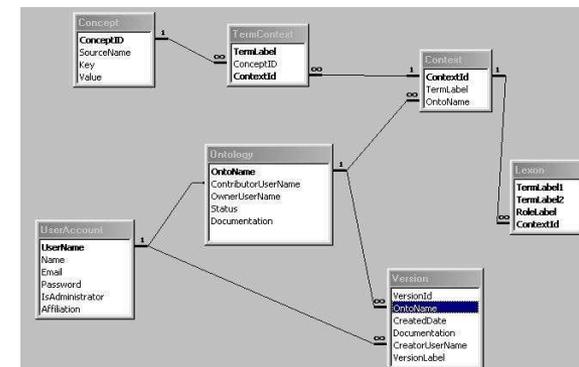
Реляционные БД

1970 г. Эдгар Кодд – «A Relational Model of Data for Large Shared Data Banks» – первая работа по реляционной модели данных.

Codd E.F. A Relational Model of Data for Large Shared Data Banks // CACM. – June 1970. – 13, #6.

Англ. **relation** – отношение.

Реляционная база данных – это набор простых таблиц, между которыми установлены связи (отношения) с помощью числовых кодов.



Сущность

- **Сущность** – это объект, который может быть идентифицирован некоторым способом, отличающим его от других объектов. Каждая сущность обладает набором атрибутов. **Атрибут** - отдельная характеристика сущности.
- Сущность состоит из **экземпляров**, каждый из которых должен отличаться от другого экземпляра. **Пример:** сущность – «Город», экземпляры сущности «Город» – Пушкин, Павловск, Колпино.

Типы сущностей

- **Независимая сущность.** Для определения экземпляра сущности нет необходимости ссылаться на другие сущности.
- **Зависимая сущность.** Для определения экземпляра такой сущности необходимо сослаться на экземпляр независимой сущности, с которой связана зависимая сущность.

СВЯЗЬ

- **СВЯЗЬ** - это логическая ассоциация, устанавливаемая между сущностями.
- Связь определяет количество экземпляров данной сущности, которое могут быть связаны с одним экземпляром другой сущности.
- Связи бывают следующих типов:
 - один к одному;
 - один ко многим;
 - многие ко многим.

Типы связей

Один-к-одному (1:1)

Используется редко, в случаях, когда часть информации об объекте либо редко используется, либо является конфиденциальной.

Типы связей

"Один-ко-многим" (1:M)

Наиболее распространенный тип связей.

Пример: информация о студентах и результатах сдачи ими экзаменов.

Типы связей

"Многие-ко-многим" (М:М)

Для реализации такая связь разбивается на две связи типа один-ко многим.

Ключ

- **Ключ** - минимальный набор атрибутов, по значениям которых можно однозначно найти требуемый экземпляр сущности.
- **Первичный ключ** сущности позволяет идентифицировать ее экземпляры, а **внешний** – экземпляры сущности, которая находится в связи с данной сущностью.

Отношение, кортеж, атрибут

$$\begin{aligned} R &\subseteq A_1 \times A_2 \times \dots \times A_n = \\ &= \{(a_1, a_2, \dots, a_n) : a_1 \in A_1, a_2 \in A_2, \dots, \\ & a_n \in A_n\} \end{aligned}$$

где:

- n – степень отношения;
- A_1, A_2, \dots, A_n – домены;
- (a_1, a_2, \dots, a_n) – кортеж;
- a_1, a_2, \dots, a_n – атрибуты.

Пример 5:

$$A_1 = \{1, 2, 3\}, A_2 = \{1, 2, 3, 4\}$$

$$R = \{(a_1, a_2) : a_1 \in A_1, a_2 \in A_2, a_1 > a_2\}$$

a_1	a_2
2	1
3	1
3	2

Основные достоинства реляционной модели

- 1) Наличие небольшого набора абстракций, которые позволяют моделировать предметную область и допускают точные формальные определения.
- 2) Наличие простого и достаточно мощного математического аппарата, опирающегося на теорию множеств и математическую логику и обеспечивающего теоретический базис реляционного подхода к организации баз данных.
- 3) Возможность ненавигационного манипулирования данными без необходимости знания конкретной физической организации баз данных во внешней памяти.

- **Фундаментальные свойства отношений**

- нет одинаковых кортежей
- кортежи не упорядочены
- атрибуты не упорядочены
- все значения атрибутов атомарные

Соответствие формальных реляционных терминов и их неформальных эквивалентов

Формальный реляционный термин	Неформальный эквивалент
Отношение	Таблица
Кортеж	Строка
Кардинальное число	Количество строк
Атрибут	Столбец
Степень	Количество столбцов
Первичный ключ	Уникальный идентификатор
Домен	Совокупность допустимых значений

Таблица, строка, столбец

- данные в ячейках таблицы структурно неделимы;
- данные в одном столбце одного типа;
- имена столбцов уникальны;
- каждая строка таблицы уникальна;
- строки и столбцы таблицы размещаются в произвольном порядке.

Реляционная алгебра

Реляционная алгебра – это коллекция операций, которые принимают таблицы в качестве операндов и возвращают таблицы в качестве результата.

Язык SQL, его структура, стандарты, история развития.

Доступ к данным осуществляется в виде запросов, которые формулируются на стандартном языке запросов. Сегодня для большинства СУБД таким языком является SQL.

Появление и развития этого языка как средства описания доступа к базе данных связано с созданием теории реляционных баз данных. Пробраз языка SQL возник в 1970 году в рамках научно-исследовательского проекта System/R (IBM). Ныне SQL — это стандарт интерфейса с реляционными СУБД.

SQL не является языком программирования в традиционном представлении.

На нем пишутся не программы, а запросы к базе данных. Поэтому SQL — декларативный или непроцедурный язык. Это означает, что с его помощью можно сформулировать, что необходимо получить, но нельзя указать, как это следует сделать.

Первый международный стандарт языка SQL был принят в 1989 г. (SQL/89 или SQL1), в 1992 г. был принят стандарт языка SQL (SQL/92 или SQL2). В 1999 г. появился стандарт SQL3. В SQL3 введены новые типы данных, при этом предоставляется возможность задания сложных структурированных типов данных, которые в большей степени соответствуют объектной ориентации. Появились стандарты на события и триггеры, которые раньше не затрагивались в стандартах.

История развития SQL

SQL

- не относится к традиционным языкам программирования;
- не содержит традиционные операторы, управляющие ходом выполнения программы, операторы описания типов и т. д.;
- содержит только набор стандартных операторов доступа к данным, хранящимся в базе данных;
- операторы SQL встраиваются в базовый язык программирования.

Язык SQL делится на подмножества.

- 1) **Язык определения данных (DDL - Data Definition Language)** предоставляет пользователям средства указания типа данных и их структуры, а также средства задания ограничений для информации, хранимой в базе данных.
Операторы – CREATE, ALTER, DROP.
- 2) **Язык манипулирования данными (DML - Data Manipulation Language)** позволяет вставлять, обновлять и извлекать информацию из базы данных.
Операторы – SELECT, INSERT, DELETE, UPDATE.
- 3) **Язык управления данными (DCL - Data Control Language)** состоит из управляющих операторов.
Операторы – GRANT, REVOKE.
- 4) **Язык управления транзакциями.**
Операторы – COMMIT, ROLLBACK, SAVEPOINT.
Запрос на языке SQL состоит из одного или нескольких операторов, следующих один за другим и разделенных точкой с запятой.

Основные операторы языка SQL

SELECT – выбрать строку (группу строк) из таблицы базы данных;

INSERT – добавить строку (группу) в таблицу базы данных;

UPDATE – изменить строку (группу) таблицы БД;

DELETE – удалить строку (группу) из таблицы БД.

Основные операторы языка SQL

Примеры запросов:

Определить количество деталей на складе для всех типов деталей.

```
SELECT Название_детали, Количество  
FROM Деталь .
```

Какие детали, изготовленные из стали, хранятся на складе?

```
SELECT *  
FROM Деталь  
WHERE Материал = 'Сталь' .
```

Основные операторы языка SQL

Примеры запросов:

Определить название и количество деталей на складе, которые изготовлены из пластмассы и весят менее 5 килограммов.

SELECT Название_детали, Количество

FROM Деталь

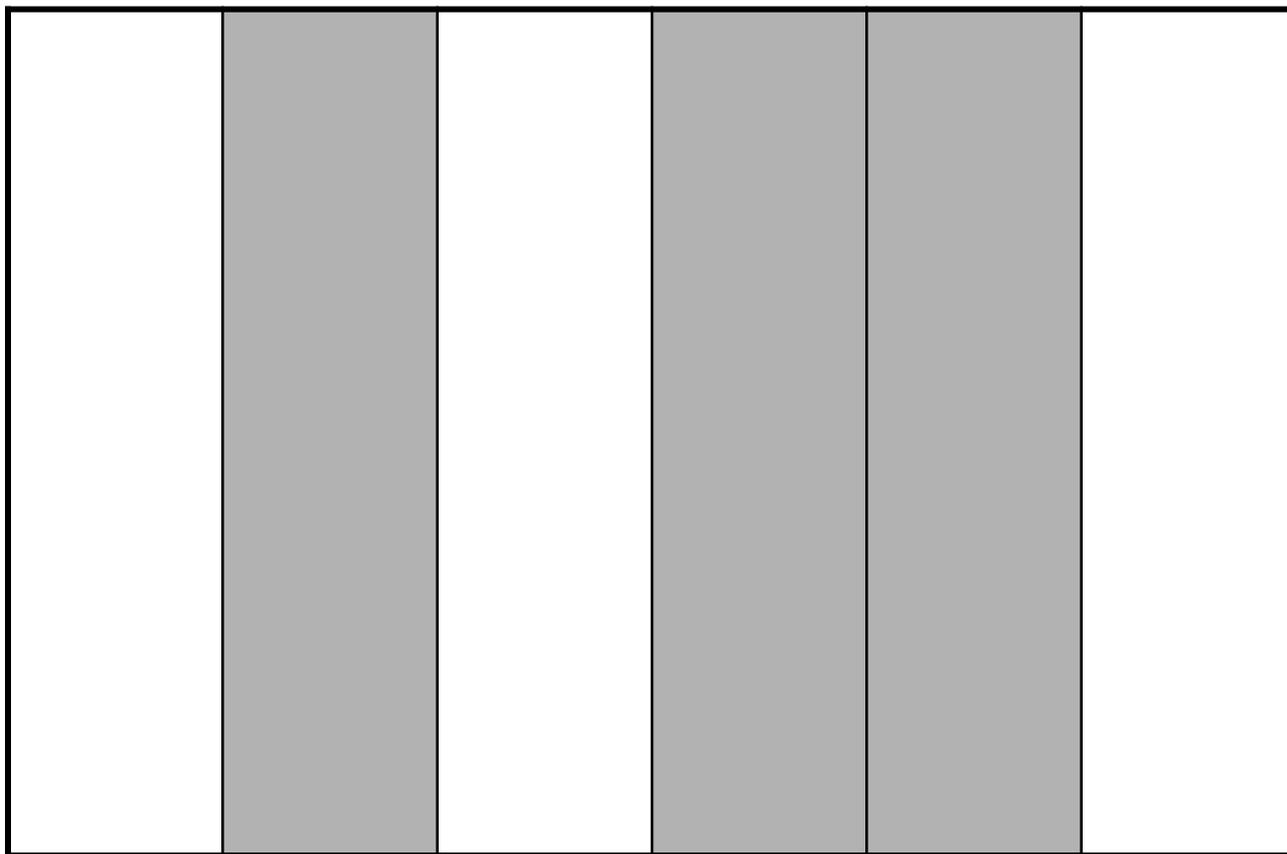
WHERE Материал = 'Пластмасса'

AND Вес < 5 .

Пример 6:

```
SELECT * FROM A WHERE A.a > 10;
```

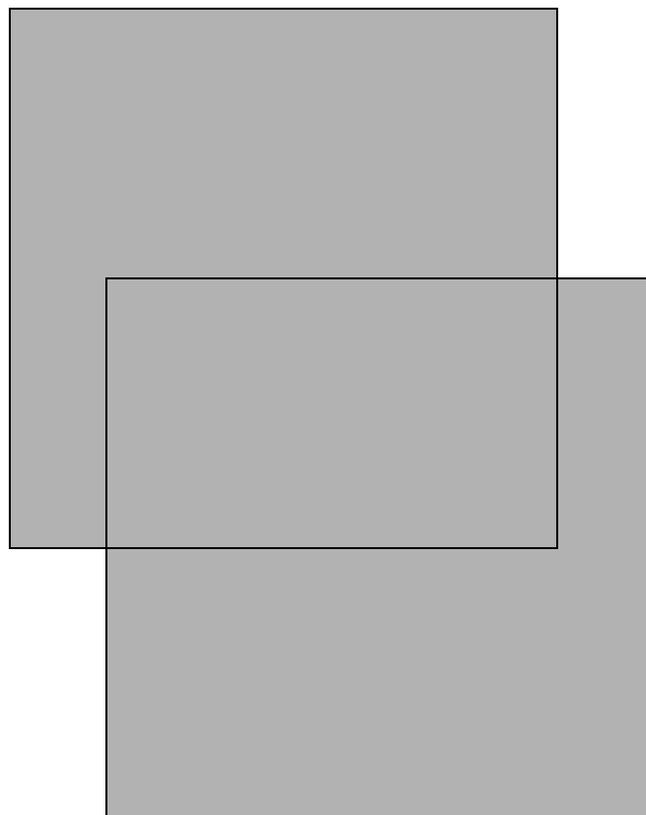
Проекция



Пример 7:

```
SELECT A.a, A.c, A.f FROM A;
```

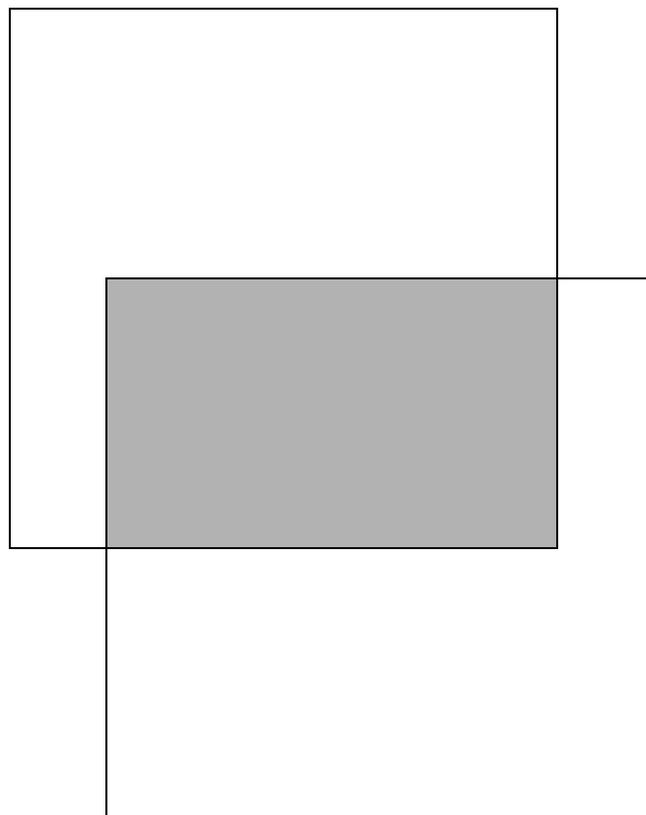
Объединение



Пример 8:

```
SELECT * FROM A  
UNION  
SELECT * FROM B;
```

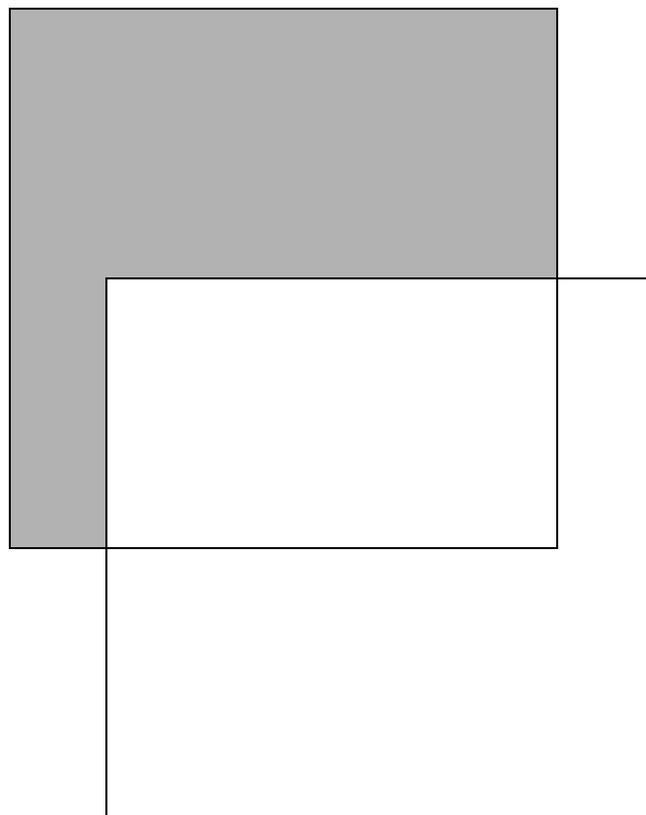
Пересечение



Пример 9:

```
SELECT * FROM A  
INTERSECT  
SELECT * FROM B;
```

Разность



Пример 10:

```
SELECT * FROM A  
MINUS  
SELECT * FROM B;
```

Соединение

a1	b1
a2	b2
a3	b3

b1	c1
b2	c2
b3	c3

a1	b1	c1
a2	b2	c2
a3	b3	c3

Пример 11:

```
SELECT A.a, A.b, B.c  
FROM A, B  
WHERE A.b = B.b;
```

Произведение

a
b
c

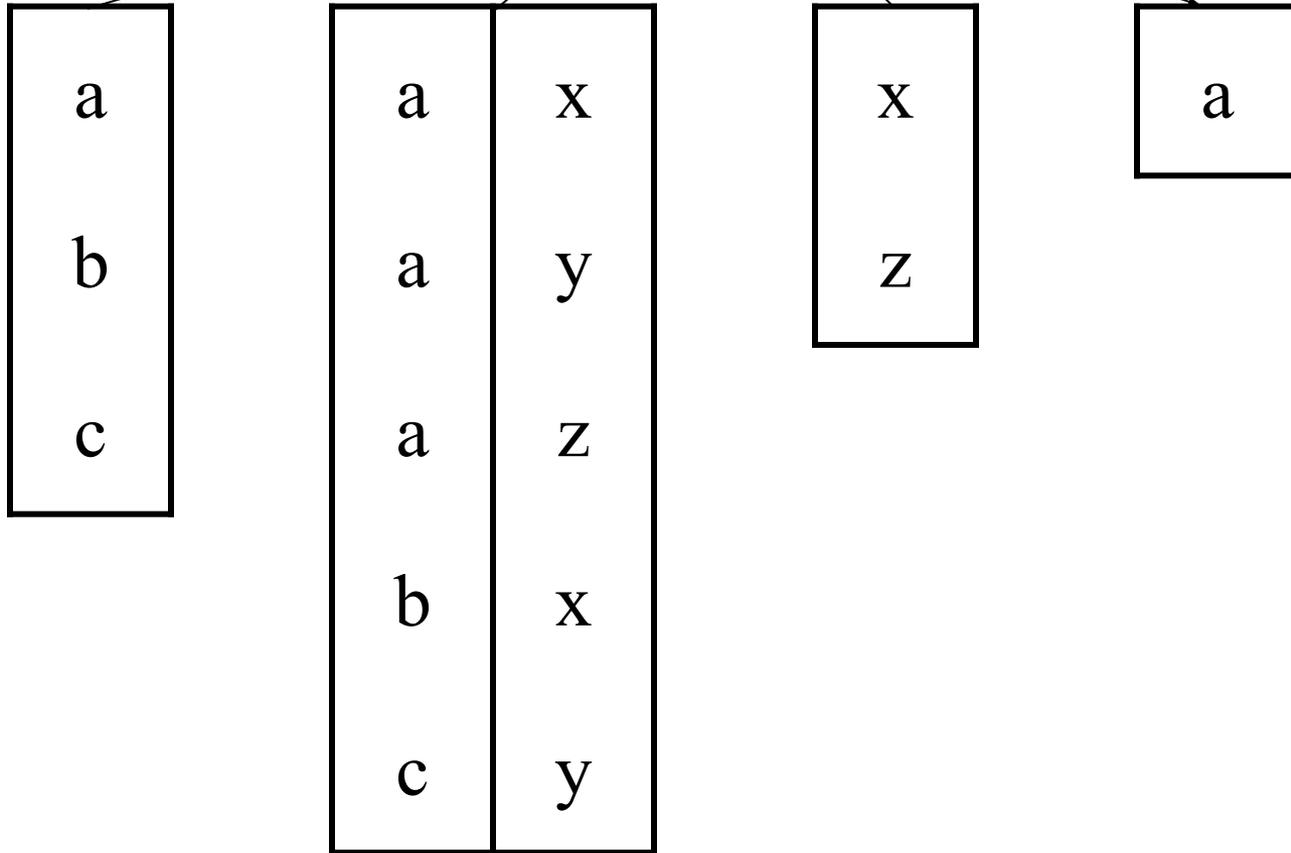
x
y

a	x
a	y
b	x
b	y
c	x
c	y

Пример 12:

```
SELECT A.*, B.* FROM A, B;
```

Деление



Пример 13:

```
SELECT DISTINCT A.a
FROM A
WHERE NOT EXISTS
  (SELECT X.x
   FROM X
   WHERE NOT EXISTS
     (SELECT AX.*
      FROM AX
      WHERE
        AX.a=A.a AND
        AX.x=X.x) ) ;
```

Жизненный цикл Базы Данных

Процедуры, выполняемые на этапах жизненного цикла БД

