ИТМО ВТ

Введение в реляционные базы данных

Лектор — Цопа Е.А.
2015/16 уч. год

1. Зачем нужны СУБД

ИТМО ВТ

Современные информационные системы

Основные требования:

- Надёжность.
- Масштабируемость.
- Удобство разработки нового функционала (обычно разработкой занимается большая команда!).
- Удобство поддержки и сопровождения.

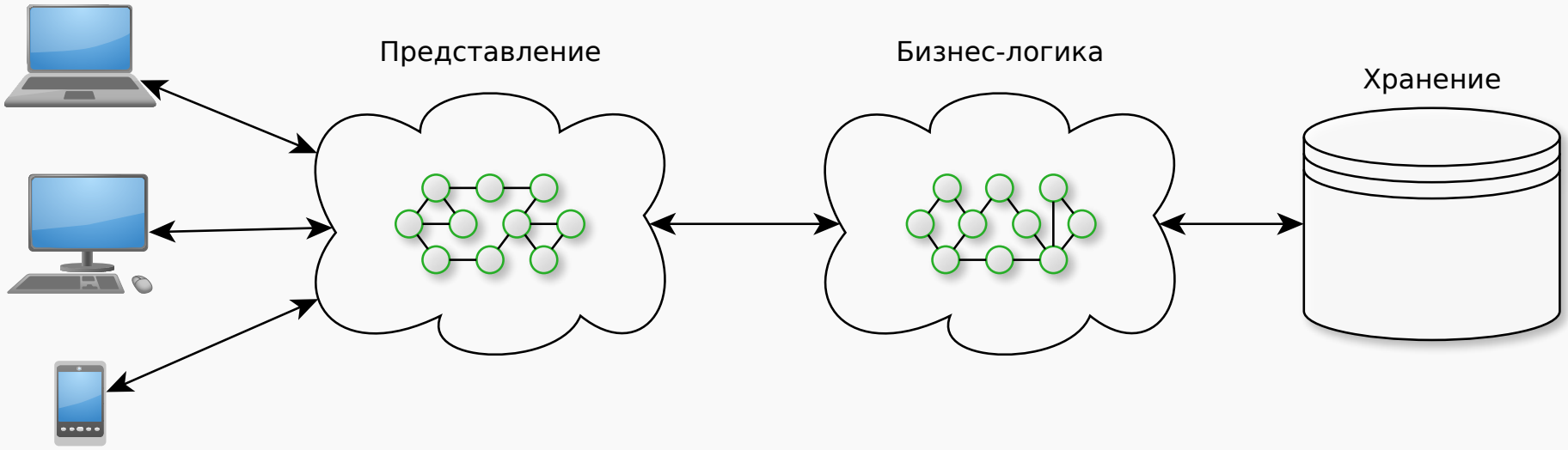
Следствия:

- ИС строится из отдельных «кубиков» (модулей, компонентов и т.д.).
- «Кубики» размещаются на нескольких уровнях.

ИТМО ВТ

Архитектура ИС

Типовая современная ИС:



ИТМО ВТ

Зачем нужны БД?

Просто хранить данные в файлах неудобно:

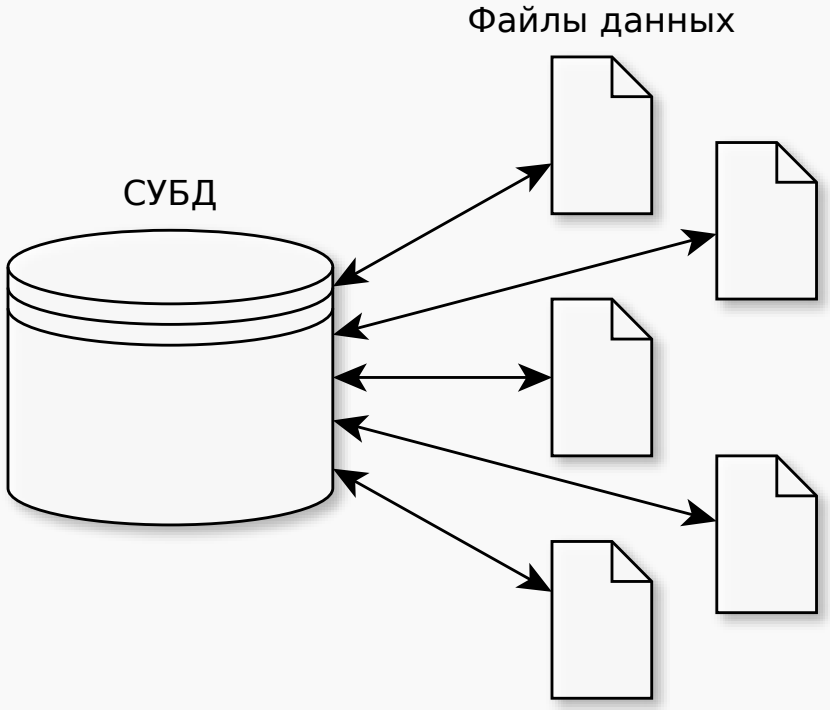
- Полное отсутствие гибкости (изменение структуры – изменение кода программы).
- Невозможность нормальной многопользовательской работы с данными.
- Вынужденная избыточность (проще создать еще одну копию данных, чем вносить изменения в десятки программ).

Решение — хранить данные и *метаданные* (данные о данных) вместе.

ИТМО ВТ

Формальное определение

База данных — это файлы, снабжённые описанием хранимых в них данных и находящиеся под управлением специальных программных комплексов, называемых "Системы управления базами данных" (СУБД).



2. Реляционные БД

ИТМО ВТ

Классификация СУБД

По степени распределённости:

- *локальные;*
- *распределённые.*

По способу доступа к БД.

- *Файл-серверные* — данные находятся на файл-сервере, СУБД — на каждом клиентском компьютере. Примеры — M$ Access, dBase, FoxPro.
- *Клиент-серверные* — СУБД находятся на сервере вместе с данными. Примеры — Oracle, M$ SQL Server, Caché.
- *Встраиваемые* — СУБД встраивается в приложение, хранит только его данные и не требует отдельной установки. Примеры — SQLite, BerkeleyDB.

ИТМО ВТ

Классификация СУБД (продолжение)

По модели данных:

- *Иерархические* — данные представляются в виде дерева. Пример — LDAP / AD, реестр Windows.
- *Сетевые* — используют сетевую модель данных. Частный случай — графовые СУБД. Примеры — HypergraphDB, OrientDB.
- *Объектно-ориентированные* — используют ОО-модель данных. Пример — InterSystems Caché.
- *Реляционные и объектно-реляционные* — используют реляционную модель данных (возможно, с частичной поддержкой ООП). Примеры — Oracle, MySQL, PostgreSQL.

ИТМО ВТ

Реляционные БД

Реляционная (relation — отношение, связь) или табличная модель данных была предложена в конце 60-х годов Эдгаром Коддом (IBM).

Ключевые особенности:

1) Данные воспринимаются пользователями как таблицы.
2) Каждая таблица состоит из однотипных строк и имеет уникальное имя.
3) Строки имеют фиксированное число полей (столбцов) и значений (множественные поля и повторяющиеся группы недопустимы).
4) В каждой позиции таблицы на пересечении строки и столбца всегда имеется в точности одно значение или ничего.

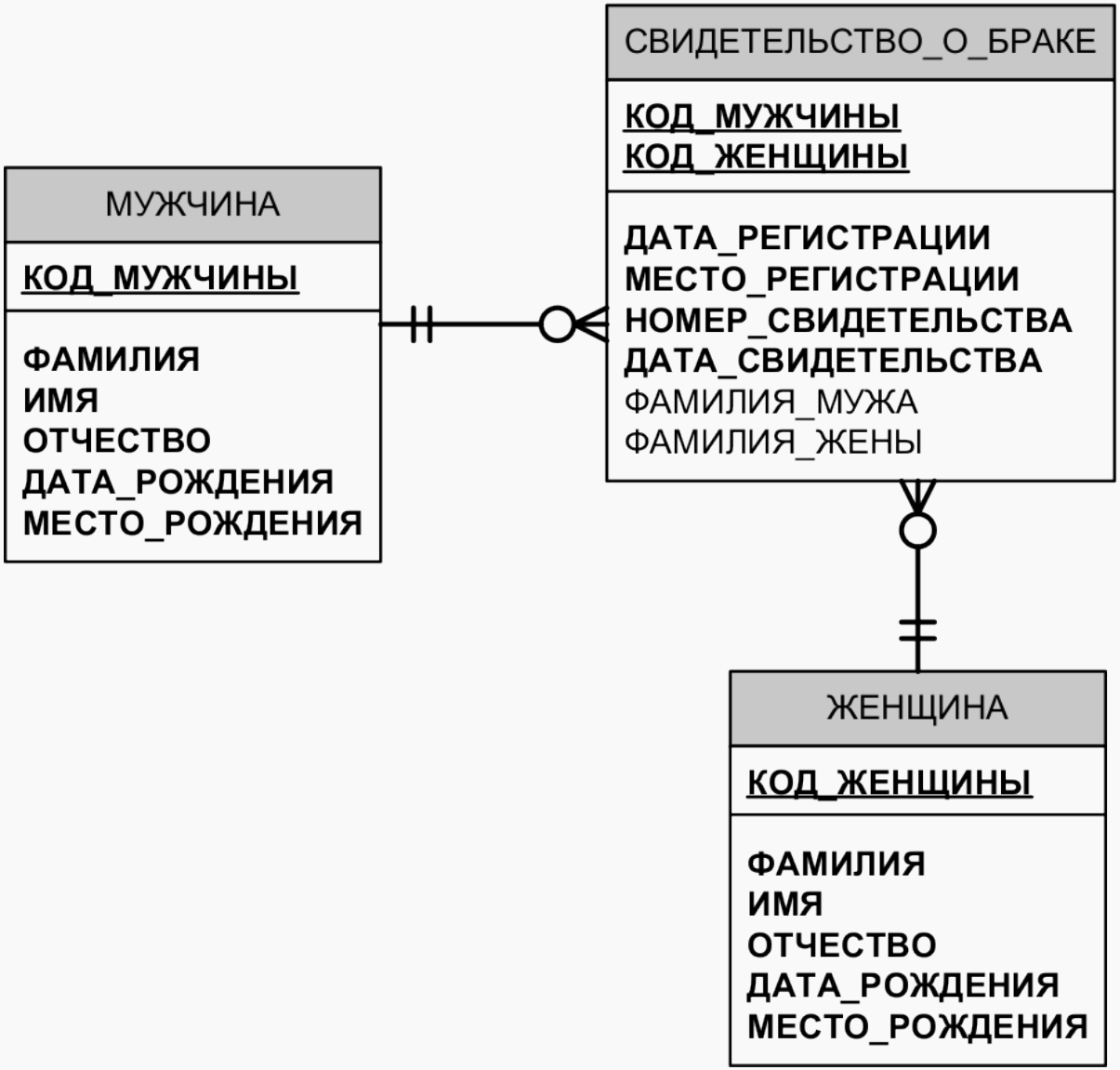
ИТМО ВТ

Реляционные БД (продолжение)

5) Строки таблицы обязательно отличаются друг от друга хотя бы единственным значением.
6) Столбцам таблицы однозначно присваиваются имена.
7) В каждом из столбцов размещаются однородные значения данных (даты, фамилии, целые числа, денежные суммы и т.д.).
8) Содержание базы данных представляется в виде явных значений данных (не существует каких-либо специальных "связей" или указателей, соединяющих одну таблицу с другой).
9) При выполнении операций с таблицей ее строки и столбцы можно обрабатывать в любом порядке безотносительно к их содержанию.

ИТМО ВТ

Пример реляционной БД



ИТМО ВТ

Таблица МУЖЧИНА

КОД_МУЖЧИНЫ	ФАМИЛИЯ	ИМЯ	ОТЧЕСТВО	ДАТА_РОЖДЕНИЯ	МЕСТО_РОЖДЕНИЯ
1	Колокольцев	Георгий	Макарович	1980-12-01	г. Москва
2	Эрдниев	Сергей	Фомич	1992-03-12	г. Санкт-Петербург
3	Кузнецов	Андрей	Гаврилович	1987-10-14	пос. Упёртовка
4	Дебилко	Анатолий	Вячеславович	1983-07-12	Село Верхние Мандроги
5	Кабицин	Аким	Тарасович	1991-12-01	NULL

ИТМО ВТ

Таблица ЖЕНЩИНА

КОД_ЖЕНЩИНЫ	ФАМИЛИЯ	ИМЯ	ОТЧЕСТВО	ДАТА_РОЖДЕНИЯ	МЕСТО_РОЖДЕНИЯ
1	Семёнова	Владислава	Ивановна	1988-09-24	г. Саратов
2	Ядова	Любовь	Трофимовна	1990-06-30	г. Москва
3	Сулимова	Александра	Филипповна	1989-12-28	Село Верхние Мандроги
4	Мисалова	Алина	Борисовна	1977-04-11	г. Москва
5	Ельцова	Изабелла	Дормидонтовна	1967-08-22	NULL

ИТМО ВТ

Таблица СВИДЕТЕЛЬСТВО_О_БРАКЕ

КОД_МУЖЧИНЫ	КОД_ЖЕНЩИНЫ	ДАТА_РЕГИСТРАЦИИ	МЕСТО_РЕГИСТРАЦИИ	НОМЕР_СВИДЕТЕЛЬСТВА	ДАТА_СВИДЕТЕЛЬСТВА	ФАМИЛИЯ_МУЖА	ФАМИЛИЯ_ЖЕНЫ
1	4	2010-01-02	Дворец Бракосочетания №1 города Москвы	7954161	2010-01-02	Колокольцев	Колокольцева
2	5	2012-02-05	NULL	2314689	2012-02-05	Эрдниев	Эрдниева
3	1	2015-10-01	NULL	8653678	2015-10-01	Кузнецов	Семёнова
4	3	2001-01-01	Дворец Бракосочетания №1 города Москвы	3567813	2001-01-01	Сулимов	Сулимова
5	2	NULL	NULL	9306001	NULL	Ядов	Кабицина

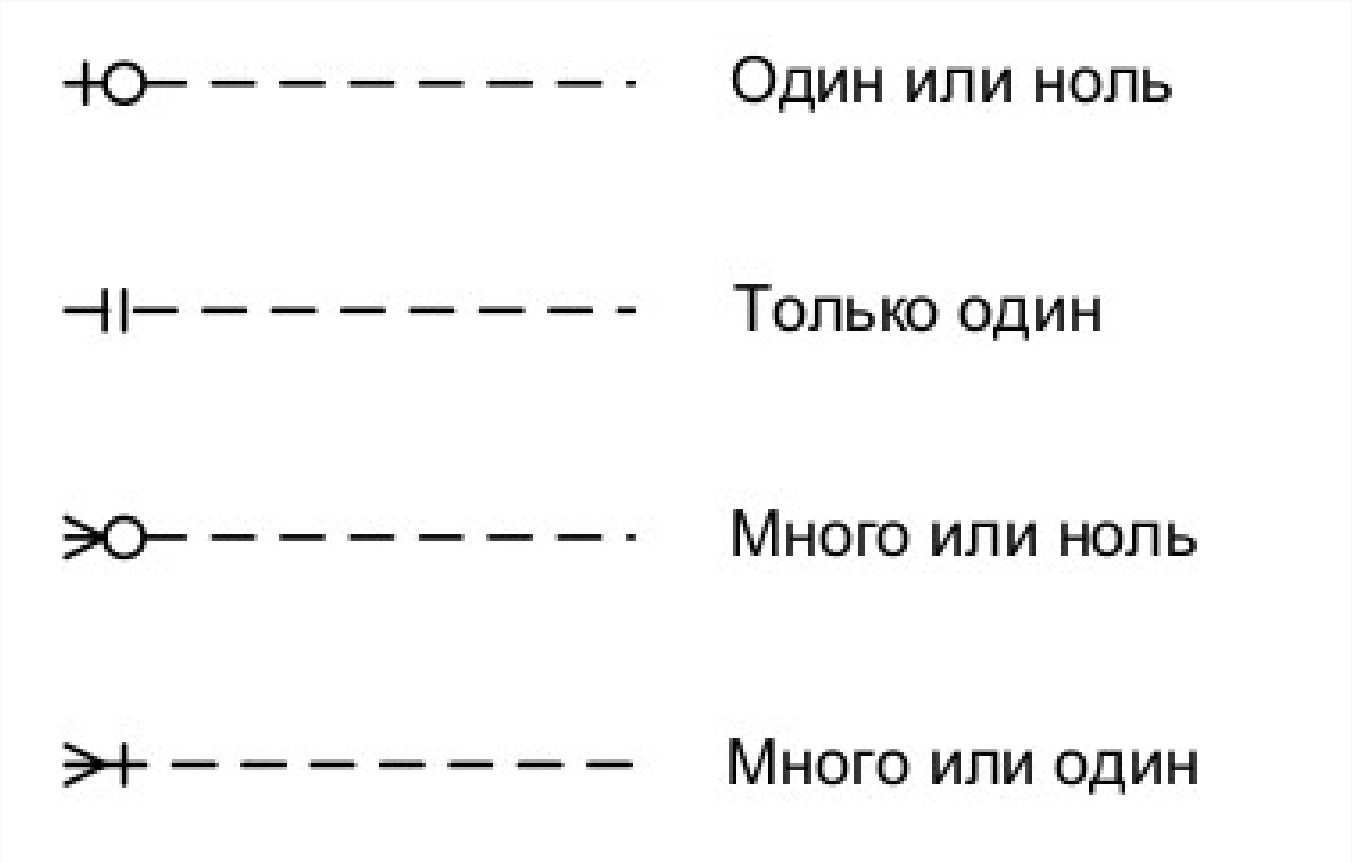
ИТМО ВТ

Терминология

- *Сущность* — любой различимый объект, факт, явление, событие, идея или предмет, информацию о котором необходимо хранить в базе данных.
- *Экземпляр сущности* относится к конкретной вещи в наборе. Например, типом сущности может быть ГОРОД , а экземпляром — Москва , Киев и т.д.
- *Атрибут* — поименованная характеристика (свойство) сущности.
- *Ключ* — минимальный набор атрибутов, по значениям которых можно однозначно найти требуемый экземпляр сущности.
- *Связь* — ассоциирование двух или более сущностей.

ИТМО ВТ

Типы связей



ИТМО ВТ

Целостность данных

Целостность (от англ. integrity) — понимается как правильность данных в любой момент времени.

Выделяют три группы правил целостности.

- *Целостность по сущностям*. Не допускается, чтобы какой-либо атрибут, участвующий в первичном ключе, принимал неопределенное значение.
- *Целостность по ссылкам*. Значение внешнего ключа должно либо:
  - быть равным значению первичного ключа ассоциируемой сущности;
  - быть полностью неопределенным.
- *Целостность, определяемая пользователем*:
  - уникальность тех или иных атрибутов;
  - диапазон значений (экзаменационная оценка от 2 до 5);
  - принадлежность набору значений (пол "М" или "Ж").

3. Основы реляционной алгебры

ИТМО ВТ

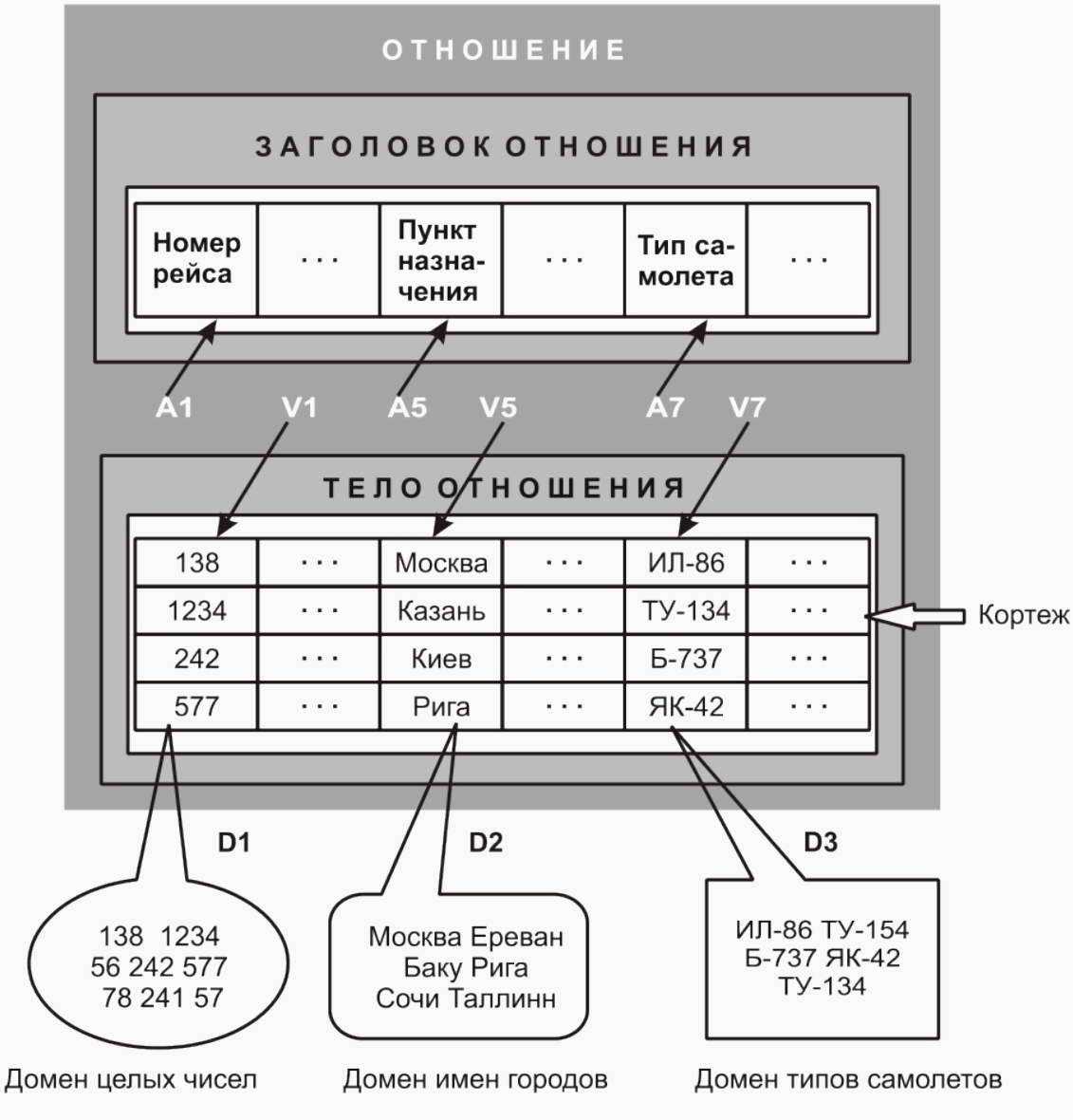
Реляционная алгебра

(C) Wikipedia:

- *Реляционная алгебра* — замкнутая система операций над отношениями в реляционной модели данных.
- Операции реляционной алгебры также называют реляционными операциями.
- Первоначальный набор из 8 операций был предложен Э. Коддом в 1970-е годы.

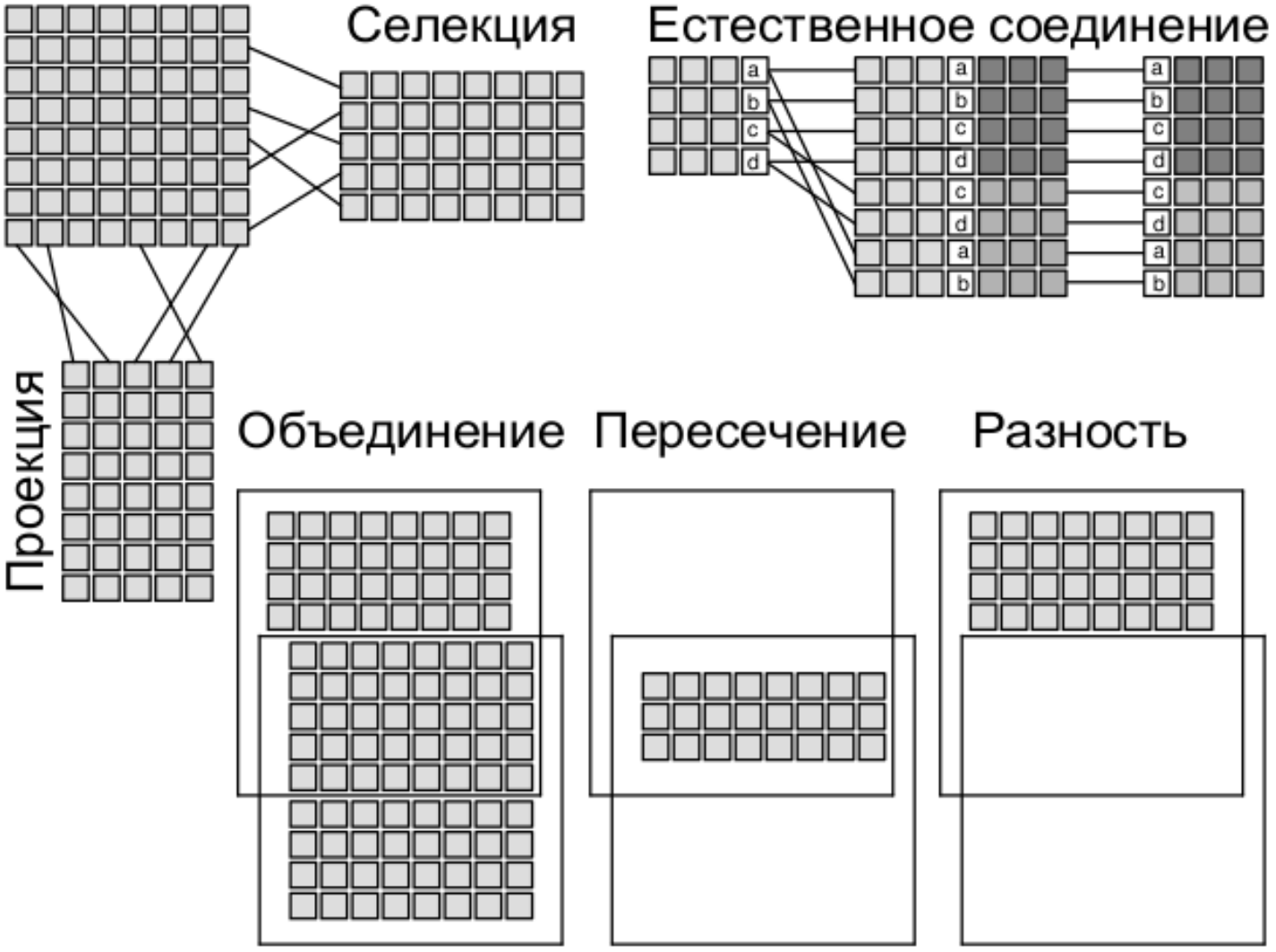
Основные понятия

- *Домен* — множество атомарных значений одного и того же типа.
- *Заголовок* состоит из такого фиксированного множества атрибутов *A1*, *A2*, ..., *An*, что существует взаимно однозначное соответствие между этими атрибутами Ai и определяющими их доменами *Di (i = 1, 2, ..., n)*.
- *Тело* состоит из меняющегося во времени множества *кортежей*, где каждый кортеж состоит в свою очередь из множества пар атрибут-значение *(Ai:Vi), (i =1, 2, ..., n)*, по одной такой паре для каждого атрибута *Ai* в заголовке.



ИТМО ВТ

Операции реляционной алгебры



ИТМО ВТ

Переименование

В результате применения операции переименования получаем новое отношение, с измененными именами атрибутов.

Синтаксис:

```
R RENAME Atr1, Atr2, … AS NewAtr1, NewAtr2, …
```

где

`R` — отношение

`Atr1, Atr2, …` — исходные имена атрибутов

`NewAtr1, NewAtr2, …` — новые имена атрибутов

ИТМО ВТ

Объединение, пересечение и разность

Объединение

Отношение с тем же заголовком, что и у совместимых по типу отношений A и B, и телом, состоящим из кортежей, принадлежащих или A, или B, или обоим отношениям.

Синтаксис:

```
A UNION B
```

Пересечение

Отношение с тем же заголовком, что и у отношений A и B, и телом, состоящим из кортежей, принадлежащих одновременно обоим отношениям A и B.

Синтаксис:

```
A INTERSECT B
```

Разность

Отношение с тем же заголовком, что и у совместимых по типу отношений A и B, и телом, состоящим из кортежей, принадлежащих отношению A и не принадлежащих отношению B.

Синтаксис:

```
A MINUS B
```

ИТМО ВТ

Декартово произведение

Отношение (A1, A2, …, Am, B1, B2, …, Bm), заголовок которого является сцеплением заголовков отношений A(A1, A2, …, Am) и B(B1, B2, …, Bm), а тело состоит из кортежей, являющихся сцеплением кортежей отношений A и B:

(a1, a2, …, am, b1, b2, …, bm)

таких, что

(a1, a2, …, am)∈ A,

(b1, b2, …, bm)∈ B.

Синтаксис:

```
A TIMES B.
```

ИТМО ВТ

Выборка и проекция

Выборка (ограничение)

Отношение с тем же заголовком, что и у отношения A, и телом, состоящим из кортежей, значения атрибутов которых при подстановке в условие c дают значение ИСТИНА. c представляет собой логическое выражение, в которое могут входить атрибуты отношения A и/или скалярные выражения.

Синтаксис:

```
A WHERE c
```

Проекция

При выполнении проекции выделяется «вертикальная» вырезка отношения-операнда с естественным уничтожением потенциально возникающих кортежей-дубликатов.

Синтаксис:

```
A[X, Y, …, Z]
```

или

```
PROJECT A {x, y, …, z}
```

ИТМО ВТ

Соединение и деление

Соединение

Операция соединения отношений A и B по предикату P логически эквивалентна последовательному применению операций декартового произведения A и B и выборки по предикату P. Если в отношениях имеются атрибуты с одинаковыми наименованиями, то перед выполнением соединения такие атрибуты необходимо переименовать.

Синтаксис:

```
(A TIMES B) WHERE P
```

Деление

Отношение с заголовком (X1, X2, …, Xn) и телом, содержащим множество кортежей (x1, x2, …, xn), таких, что для всех кортежей (y1, y2, …, ym) ∈ B в отношении A(X1, X2, …, Xn, Y1, Y2, …, Ym) найдется кортеж (x1, x2, …, xn, y1, y2, …, ym).

Синтаксис:

```
A DIVIDEBY B
```

4. Язык SQL

ИТМО ВТ

О языке

- *SQL (structured query language)* — формальный непроцедурный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных.
- Основывается на исчислении кортежей.
- Разработан в 1970-е гг. в компании IBM.
- Первая СУБД, поддерживающая SQL — Oracle V2 (1979 г.).

ИТМО ВТ

Особенности языка

Каждое предложение SQL — это либо запрос данных из базы, либо обращение к базе данных, которое приводит к изменению данных в базе.

Различают следующие типы запросов:

- запросы на создание или изменение в базе данных новых или существующих объектов;
- запросы на получение данных;
- запросы на добавление новых данных (записей);
- запросы на удаление данных;
- обращения к СУБД.

ИТМО ВТ

Операторы SQL

Операторы определения данных *(Data Definition Language, DDL)*:

- `CREATE` создает объект БД (саму базу, таблицу, представление, пользователя и т. д.);
- `ALTER` изменяет объект;
- `DROP` удаляет объект.

Операторы манипуляции данными *(Data Manipulation Language, DML)*:

- `SELECT` считывает данные, удовлетворяющие заданным условиям;
- `INSERT` добавляет новые данные;
- `UPDATE` изменяет существующие данные;
- `DELETE` удаляет данные.

ИТМО ВТ

Операторы SQL (продолжение)

Операторы определения доступа к данным *(Data Control Language, DCL)*:

- `GRANT` предоставляет пользователю (группе) разрешения на определенные операции с объектом;
- `REVOKE` отзывает ранее выданные разрешения;
- `DENY` задает запрет, имеющий приоритет над разрешением.

Операторы управления транзакциями *(Transaction Control Language, TCL)*:

- `COMMIT` применяет транзакцию;
- `ROLLBACK` откатывает все изменения, сделанные в контексте текущей транзакции;
- `SAVEPOINT` делит транзакцию на более мелкие участки.

ИТМО ВТ

Оператор SELECT

Возвращает набор данных из БД, удовлетворяющих заданному условию.

Формат запроса:

```
SELECT список полей FROM список таблиц WHERE условия…
```

Основные ключевые слова:

- `WHERE` — используется для определения, какие строки должны быть выбраны или включены в `GROUP BY`.
- `GROUP BY` — используется для объединения строк с общими значениями в элементы меньшего набора строк.
- `HAVING` — используется для определения, какие строки после `GROUP BY` должны быть выбраны.
- `ORDER BY` — используется для определения, какие столбцы используются для сортировки результирующего набора данных.

ИТМО ВТ

Параметр GROUP BY

- Позволяет группировать строк ипо результатам агрегатных функций (MAX, SUM, AVG, …).
- Необходимо, чтобы в SELECT были заданы только требуемые в выходном потоке столбцы, перечисленные в GROUP BY и/или агрегированные значения.
- Пример использования:

  Запрос возвращает список партнеров с общей суммой продажи с 1 января 2000 года:

```
SELECT Partner, SUM(SaleAmount) FROM Sales
  WHERE SaleDate > '01-Jan-2000'
  GROUP BY Partner;
```

ИТМО ВТ

Параметр HAVING

- Позволяет указывать условия на результат агрегатных функций (MAX, SUM, AVG, …).
- Аналогичен WHERE за исключением того, что строки отбираются не по значениям столбцов, а строятся из значений столбцов, указанных в GROUP BY, и значений агрегатных функций, вычисленных для каждой группы, образованной GROUP BY.
- Необходимо, чтобы в SELECT были заданы только требуемые в выходном потоке столбцы, перечисленные в GROUP BY и/или агрегированные значения.
- Если параметр GROUP BY в SELECT не задан, HAVING применяется к «группе» всех строк таблицы, полностью дублируя WHERE.
- Пример использования:

  Возвращает список идентификаторов отделов, продажи которых превысили 1000 долларов за 1 января 2000 года, вместе с суммами продаж за этот день:

```
SELECT DeptID, SUM(SaleAmount) FROM Sales
  WHERE SaleDate = '01-Jan-2000'
  GROUP BY DeptID
  HAVING SUM(SaleAmount) > 1000;
```

ИТМО ВТ

Пример SQL-запроса