Основы проектирования баз данных

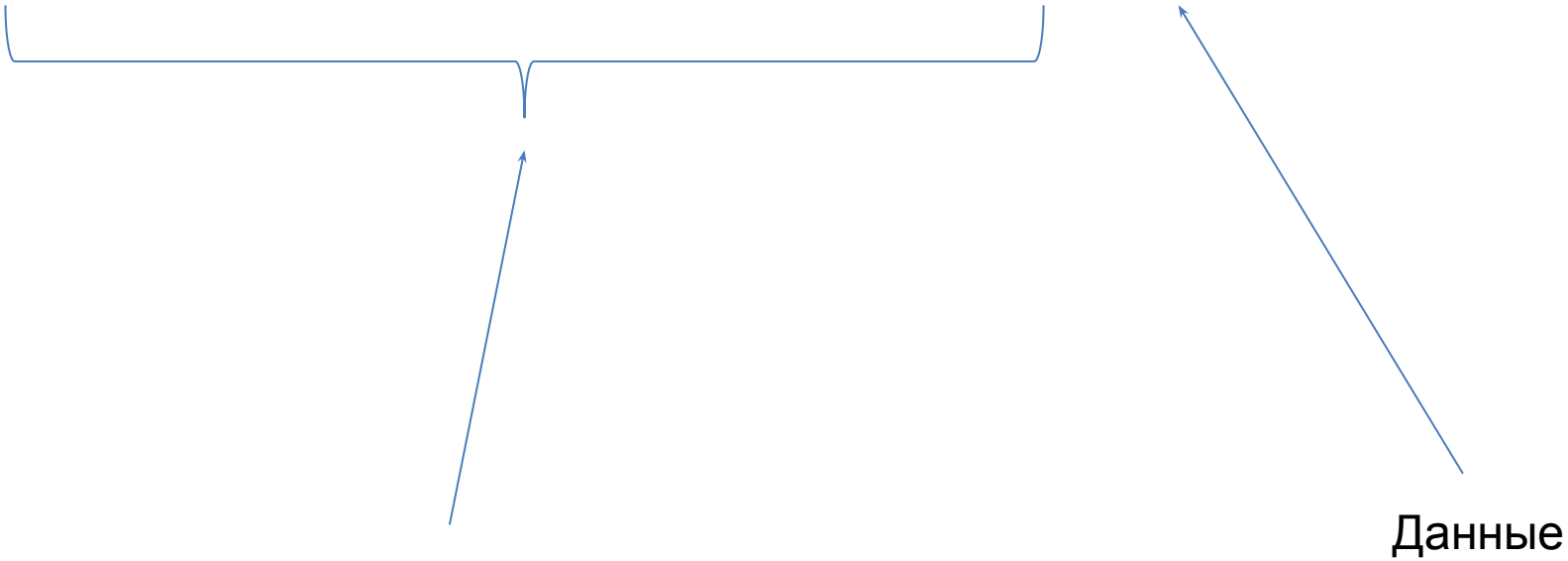
1.1 Понятие данных

Данные - это поддающееся многократной интерпретации представление информации в формализованном виде, пригодном для передачи, связи, или обработки.

Семантика данных-это их интерпретация или возможность формального описания смысла передаваемых данных.

Пример 1

Стоимость авиабилета 115



Пример 2

Интерпретация

Номер рейса	Дни недели	Пункт отправления	Время вылета	Пункт назначения	Время прибытия	Тип самолета	Стоимость билета

Данные

138	.2.4..7	Баку	21.12	Москва	0.52	ИЛ-86	115.00
57	..3..6.	Ереван	7.20	Киев	9.25	ТУ-154	92.00
1234	.2…6.	Казань	22.40	Баку	23.50	ТУ-134	73.50
242	1234567	Киев	14.10	Москва	16.15	Б-737	57.00
86	.23.5..	Минск	10.50	Сочи	13.06	ИЛ-86	78.50
137	1.3..6.	Москва	15.17	Баку	18.44	ИЛ-86	115.00
241	1234567	Москва	9.05	Киев	11.05	Б-737	57.00
577	1.3.5..	Рига	21.53	Таллинн	22.57	ЯК 42	21.50
78	..3..6.	Сочи	18.25	Баку	20.12	ТУ-134	44.00
578	.2.4.6.	Таллинн	6.30	Рига	7.37	ЯК 42	21.50

1.2. Концепция баз данных

Ба́за да́нных — совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных.

Система управления базами данных, сокр. СУБД (англ. Database Management System, сокр. DBMS) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Пример БД “Аэропорт”

```
СОЗДАТЬ ТАБЛИЦУ Расписание
  (Номер_рейса          Целое
   Дни_недели           Текст (8)
   Пункт_отправления    Текст (24)
   Время_вылета         Время
   Пункт_назначения     Текст (24)
   Время_прибытия       Время
   Тип_самолета         Текст (8)
   Стоимость_билета     Валюта);
```

Запросы к базе данных

Запрос – это обращение к данным для получения информации из базы данных или выполнения действий с данными.

Язык запросов-это искусственно созданный язык, на котором делаются запросы к базам данных и информационно-поисковым системам.

Пример запроса №1

```
ВЫБРАТЬ  Номер_рейса, Дни_недели, Время_вылета
ИЗ ТАБЛИЦЫ  Расписание
ГДЕ  Пункт_отправления = 'Москва'
  И  Пункт_назначения  = 'Киев'
  И  Время_вылета > 17;
```

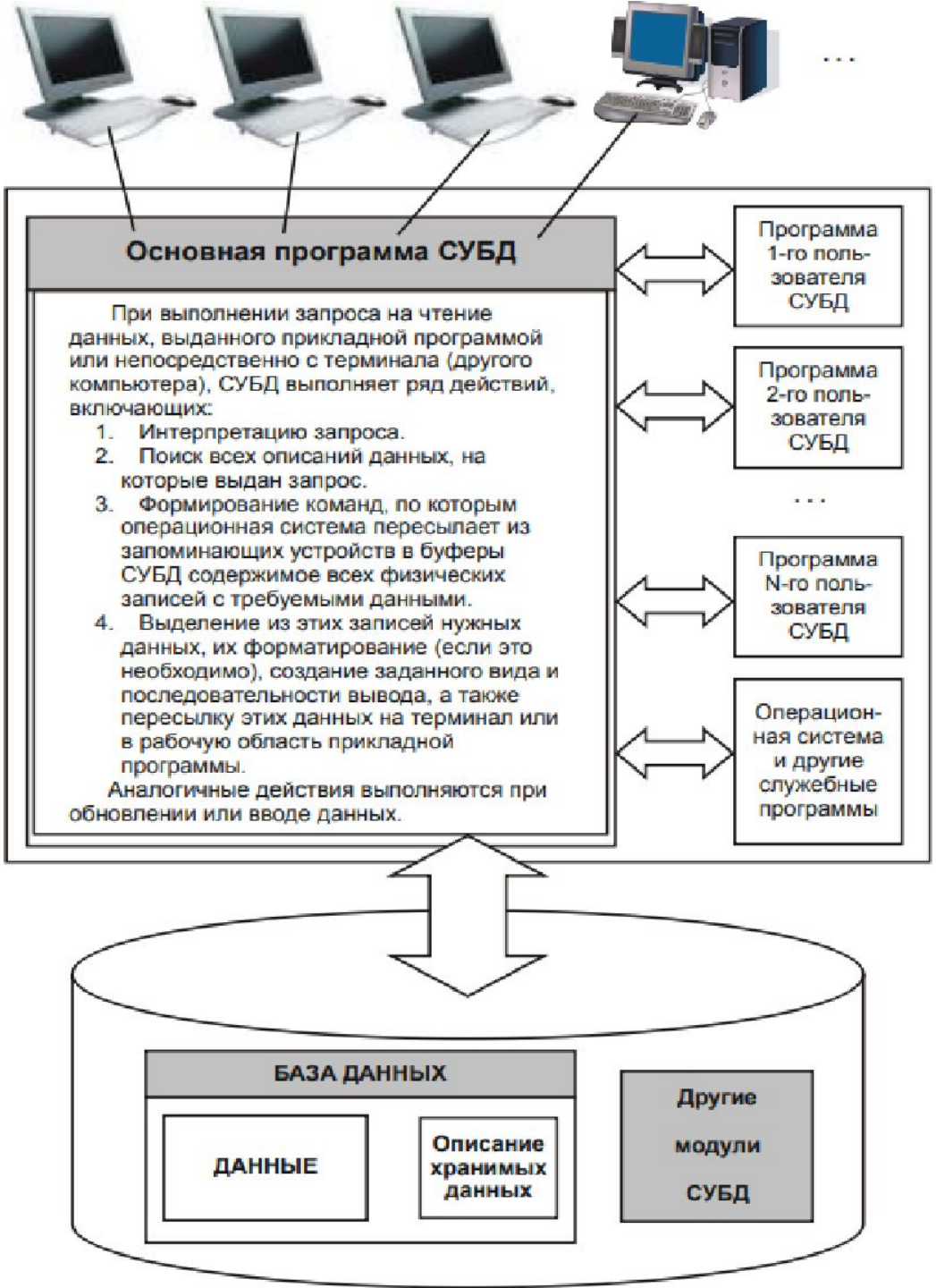
получим расписание "Москва—Киев" на вечернее время

Пример запроса №2

```
ВЫБРАТЬ  КОЛИЧЕСТВО(Номер_рейса)
ИЗ ТАБЛИЦЫ  Расписание
ГДЕ  Пункт_отправления = 'Москва'
  И  Пункт_назначения  = 'Минск';
```

получим количество рейсов "Москва—Минск".

Связь программ и данных при использовании СУБД

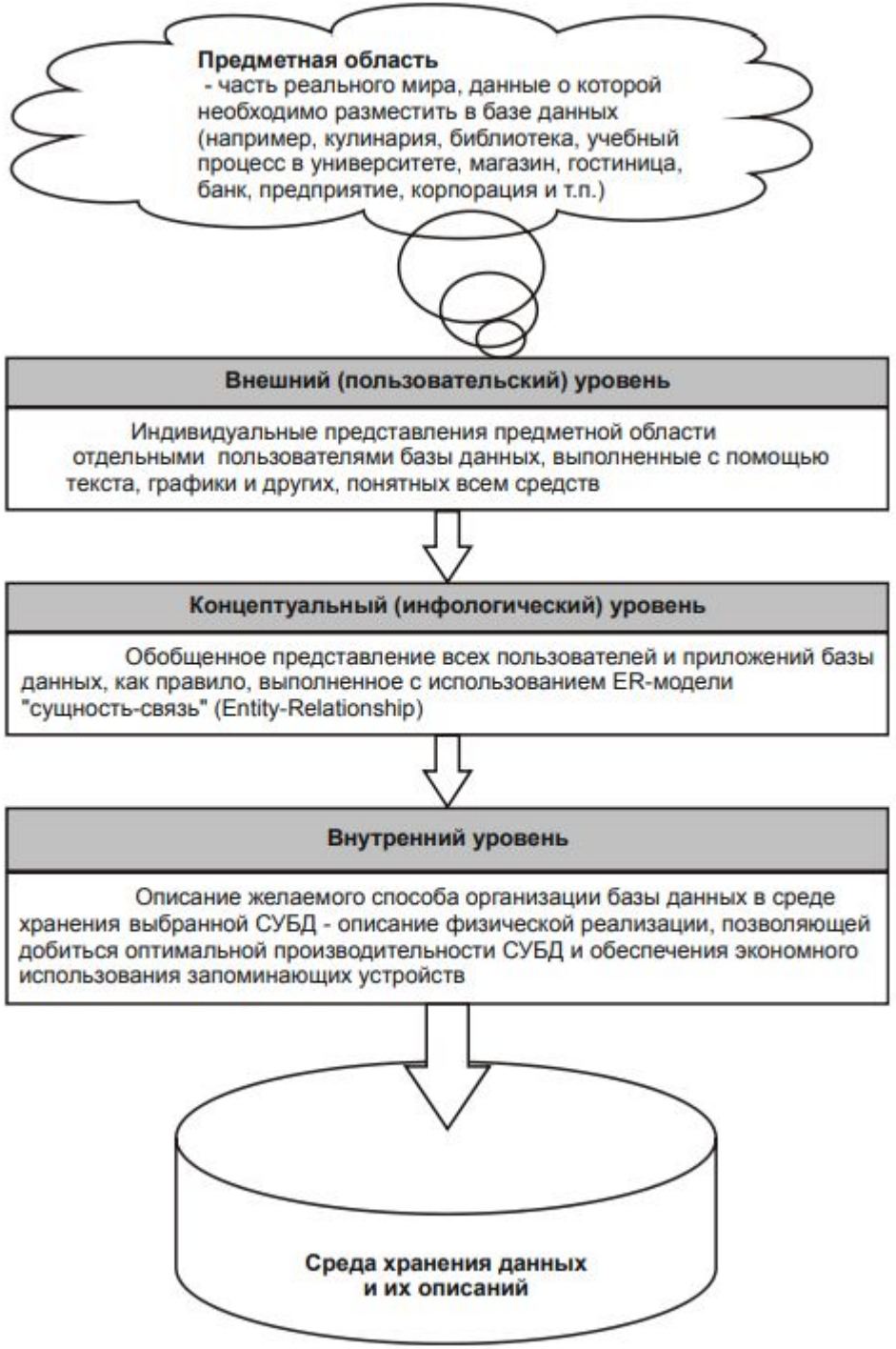


1.3. Архитектура СУБД

1975 году подкомитет SPARC (Standards Planning and Requirements Committee) американского национального института стандартов (American National Standards Institute, ANSI) выдвинул проект трехуровневой архитектуры СУБД :

1. Внешний (пользовательский).
2. Промежуточный (концептуальный).
3. Внутренний (физический).

Уровни моделей данных



Инфологическая модель данных “сущность-связь”

2.1. Основные понятия

Сущность — любой различимый объект, факт, явление, событие, идея или предмет, информацию о котором необходимо хранить в базе данных.

Атрибут — поименованная характеристика (свойство) сущности.

Ключ — минимальный набор атрибутов, по значениям которых можно однозначно найти требуемый экземпляр сущности.

Связь — ассоциирование двух или более сущностей.

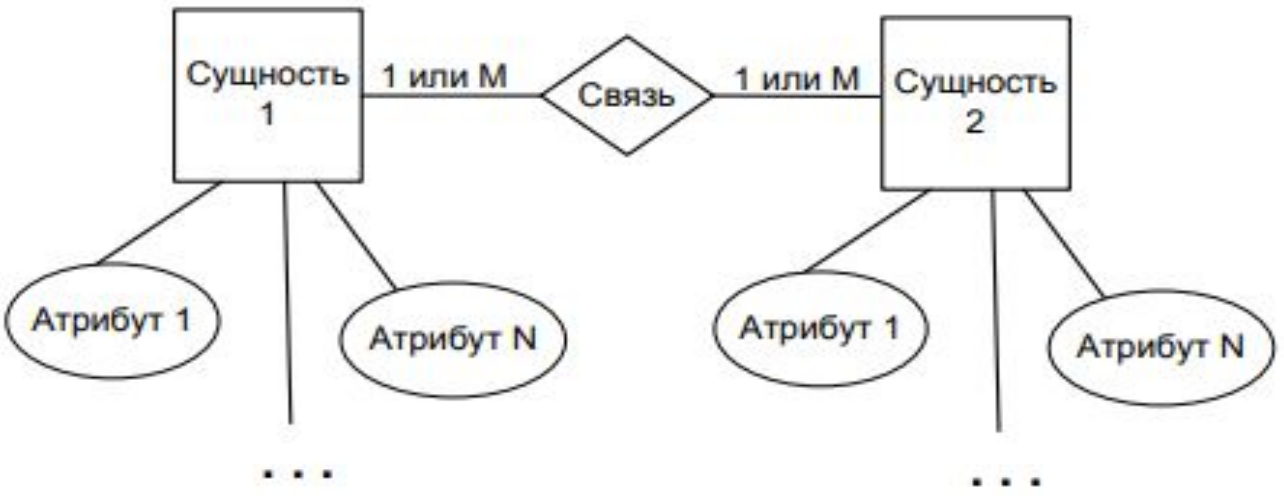
Вопрос 1

1. Данные-это…(Приведите примеры)
2. Семантика данных-это…(Приведите примеры)
3. Ба́за да́нных-это…(Приведите примеры)
4. Система управления базами данных-это…(Приведите примеры)

Вопрос 2

1. Запрос-это…
2. Язык запросов-это…
3. Архитектура СУБД
4. Уровни моделей данных

2.2. Характеристика связей и язык моделирования

- При построении инфологических моделей можно использовать язык ERдиаграмм — от англ. Entity-Relationship, т. е. сущность-связь.
- 

Между двумя сущностями, возможны четыре вида связей.

- связь "один-к-одному" (1:1): в каждый момент времени каждому представителю (экземпляру) сущности А соответствует 1 или 0 представителей сущности Б.
- Например, студент может не

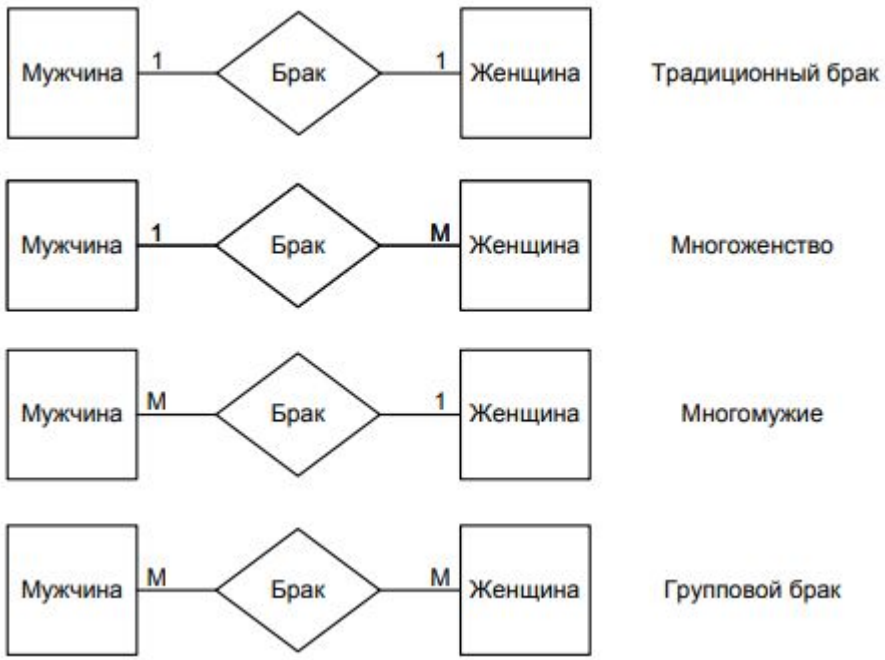
а)

- связь "один-ко-многим" (1:M): одному представителю сущности А соответствуют 0, 1 или несколько представителей сущности Б.
- Например, квартира может пустовать, в

б)

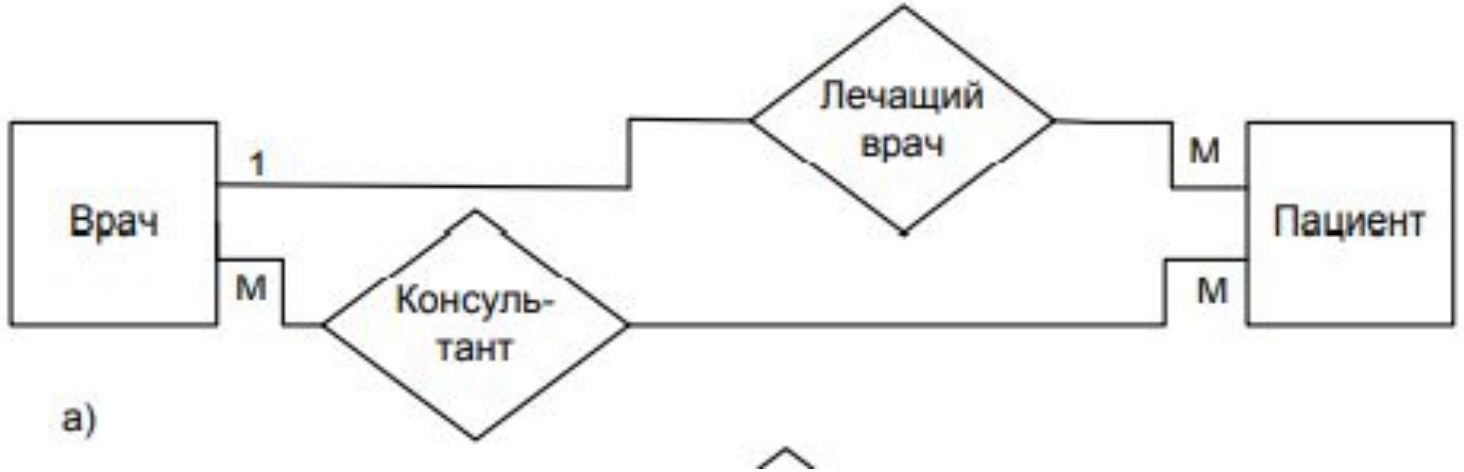
"многие-к-одному" (M:1) и "многие-ко многим" (M:M).

- *Пример.* Если связь между сущностями МУЖЧИНЫ и ЖЕНЩИНЫ называется БРАК, то существует четыре возможных предст



Существуют и более сложные связи:

- множество связей между одними и теми же сущностями : пациент, имея одного лечащего врача, может иметь также несколько врачей-консультантов; врач может быть лечащим врачом нескольких пациентов и может одновременно



Существуют и более сложные связи:

- тернарные связи : врач может назначить несколько пациентов на несколько анализов, анализ может быть назначен несколькими врачами нескольким пациентам и пациент мо анализов не

- связи более высоких порядков, семантика (смысл) которых иногда очень сложна.

В приведенных примерах для повышения иллюстративности рассматриваемых связей не показаны атрибуты сущностей ER-диаграмм.

Так, ввод атрибутов в описание брачных связей:

- фамилии, имена и отчества мужа и жены;
- даты и места рождения мужа и жены;
- даты и места регистрации брака;
- фамилий, присвоенных мужу и жене после регистрации;
- Даты выдачи и номера свидетельства о браке.

значительно усложнит ER-диаграмму и затруднит ее понимание.

- Поэтому к настоящему времени появилось множество более удобных графических нотаций описания ER-диаграмм, поддержанных CASE-средствами (от Computer Aided Software/System Engineering) разработки информационных систем и редакторами деловой графики.

Сущность

- В этих нотациях сущность изображается в виде прямоугольника, в верхней части которого располагается имя сущности.
- В прямоугольнике перечислены атрибуты сущности. Атрибуты, расположенные сверху и отделенные от остальных горизонтальной линией, являются ключевыми.

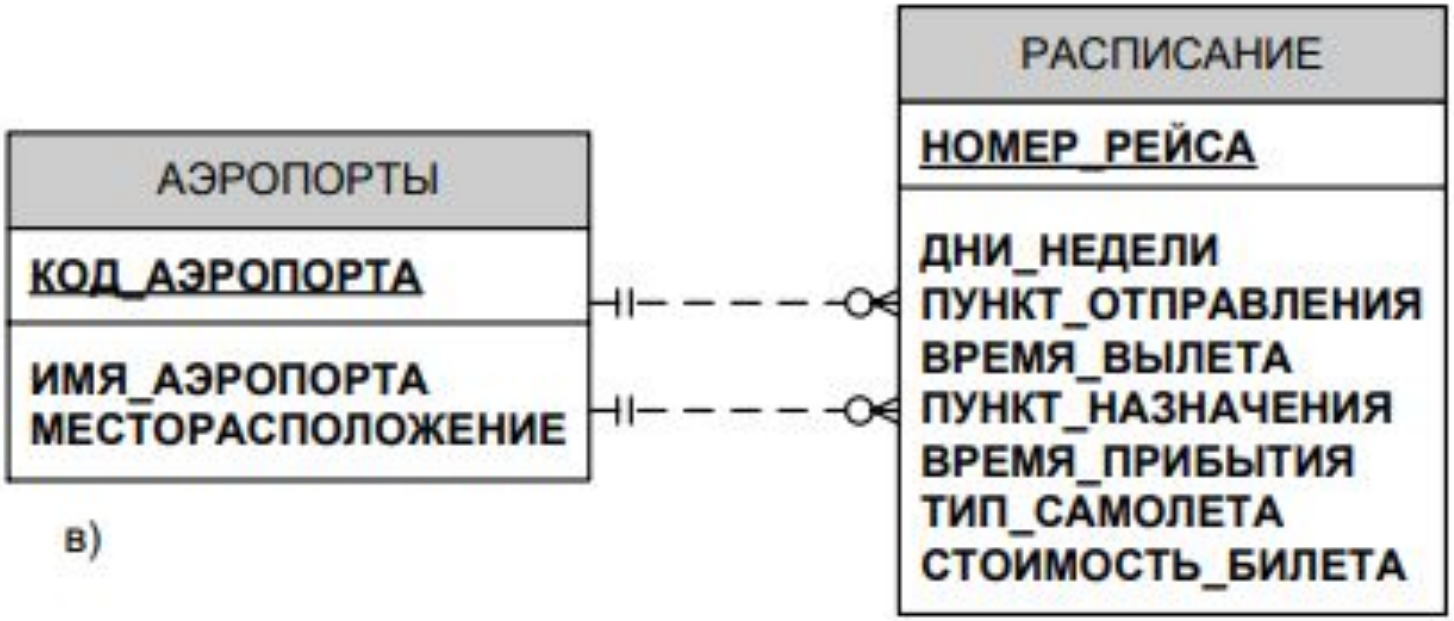
Связь

Связь изображается пунктирной линией между двумя сущностями.

На концах линий проставляются условные графические обозначения:

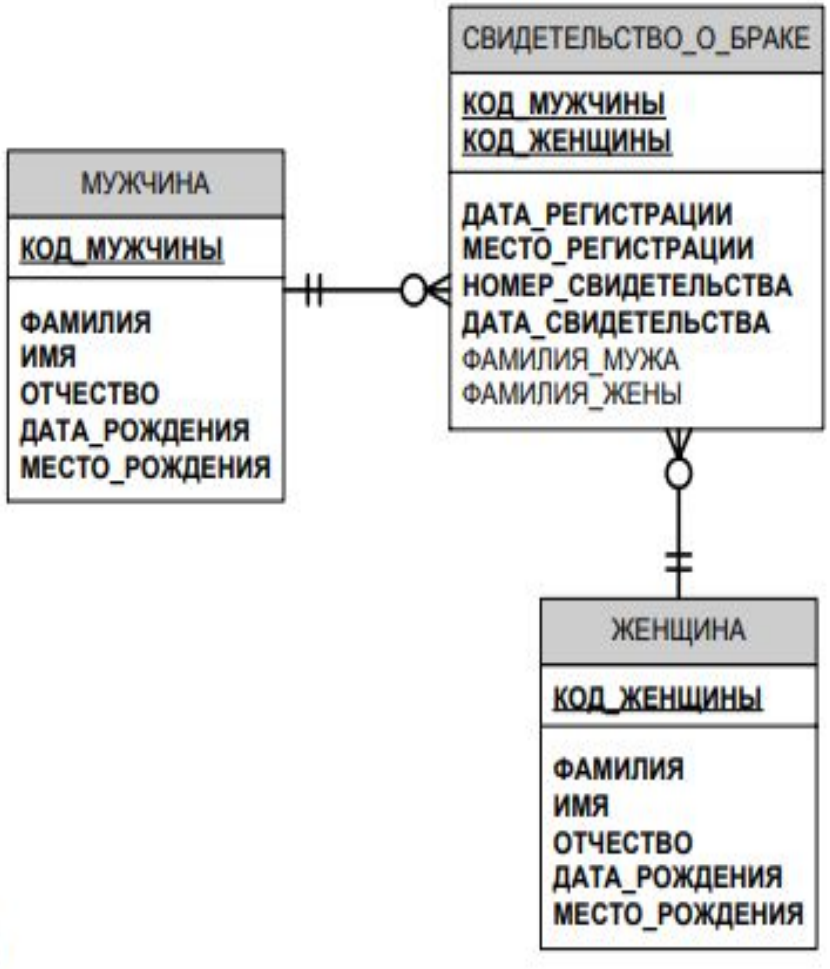
- вертикальная черта (один),
- кружок (ноль или "необязательно"),
- "воронья лапа" (много).

Пример ER-диаграммы



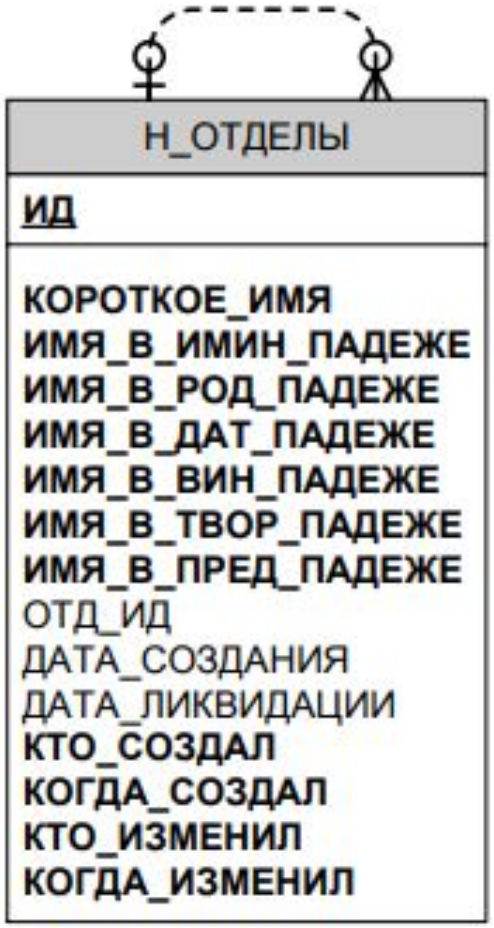
в)

- Если в ключ какой-либо сущности входит ключ другой сущности, то связь между такими сущностями изображается не пунктирной, а сплошной линией.



Рекурсивная связь

- Достаточно часто встречается еще один специфический вид связи: рекурсивная связь между атрибутами одной сущности "Один-ко-многим" ("Свиное ухо"), используемая для описания иерархий с любым числом уровней.
- Так кафедра вычислительной техники (ИД=102) входит в состав факультета компьютерных технологий и управления (ИД=703), который, в свою очередь, входит в состав университета (ИД=777). В данном случае рассмотрен пример с тремя уровнями иерархии, но ясно, что при использовании этого описания количество этих уровней ограничивается только предметной областью (здесь можно было бы представить, например, любые секции кафедры, подсекции и т. п.).



2.3. Классификация сущностей

Основоположник реляционной модели баз данных Эдгар Кодд определяет **три основных класса сущностей:**

- стержневые,
- ассоциативные
- характеристические.

Стержневая сущность

- **Стержневая сущность (стержень)** — это независимая сущность, которая не является ни характеристикой, ни ассоциацией.

- В рассмотренных ранее примерах стержни — это Студент, Квартира, Мужчины, Врач и другие, названия которых помещены в прямоугольники.

Ассоциативная сущность

Ассоциативная сущность (ассоциация) — это связь вида "многие-комногим") между двумя или более сущностями или экземплярами сущности.

Ассоциации рассматриваются как полноправные сущности:

- они могут участвовать в других ассоциациях точно так же, как стержневые сущности;
- могут обладать свойствами, т. е. иметь не только набор ключевых атрибутов, необходимых для указания связей, но и любое число других атрибутов, характеризующих связь.

- Например, ассоциация СВИДЕТЕЛЬСТВО_О_БРАКЕ содержит ключевые атрибуты КОД_МУЖЧИНЫ и КОД_ЖЕНЩИНЫ, а также уточняющие атрибуты ДАТА_РЕГИСТРАЦИИ, МЕСТО_РЕГИСТРАЦИИ, НОМЕР_СВИДЕТЕЛЬСТВА и т. д.

Характеристическая сущность

- **Характеристическая сущность (характеристика)** — это связь вида "многие-к-одной" или "одна-к-одной" между двумя сущностями (частный случай ассоциации).
- Единственная цель характеристики в рамках рассматриваемой предметной области состоит в описании или уточнении некоторой другой сущности.
- Необходимость в них возникает в связи с тем, что сущности реального мира имеют иногда многозначные свойства. Муж может иметь несколько жен; книга — несколько характеристик переиздания (исправленное, дополненное, переработанное и пр.) и т. д.
- Существование характеристики полностью зависит от характеризуемой сущности: женщины лишаются статуса жен, если умирает их муж.

Рассмотрим пример построения инфологической модели базы данных "СООК", предназначенной для использования в пансионатах, санаториях и других организациях, предоставляющих услуги по обеспечению отдыха.

- Информация из такой базы данных будет использоваться шеф-поваром пансионата для составления меню, учитывающего примерную стоимость и необходимую калорийности суточного рациона отдыхающих.

- В меню предлагается по несколько альтернативных блюд каждого вида (закуска, горячее, суп и т. п.) для каждой трапезы (завтрак, обед, ужин).

- Перед завтраком каждый отдыхающий выбирает в информационной системе номер закрепленного за ним места в столовой пансионата и желаемый набор блюд для каждой из трапез следующего дня (желаемый набор строк меню).
- Это позволяет определить, сколько порций того или иного блюда надо приготовить для каждой трапезы, а также набор и количество необходимых продуктов.

- Завхоз, связанный с поставщиками продуктов, определяет, что необходимо заказать для обеспечения работы столовой.
- При составлении меню шеф-повар

127 Лобио по грузински (закуска)

Ломаную очищенную фасоль, нашинкованный лук посолить, посыпать перцем и припустить в масле с небольшим количеством бульона; добавить кинзу, зелень петрушки, реган (базилик) и довести до готовности. Затем запечь в духовке.

Фасоль стручковая (свежая или консервированная) 200,

Лук зеленый 40, Масло сливочное 30, Зелень 10.

Выход 210. Калорий 725.

Исходя из потребностей указанных ранее лиц, можно определить объекты, необходимые для выявления сущностей и атрибутов проектируемой базы:

1. **Блюда**, для описания которых нужны данные, входящие в их кулинарные рецепты:

- код (номер) блюда (например, из книги кулинарных рецептов);
- название блюда;
- вид блюда (закуска, суп, горячее и т. п.);
- рецепт (технология приготовления блюда);
- выход (вес порции);
- калорийность блюда;
- название, вес и основные вещества (белки, жиры, углеводы, витамины и др.) каждого продукта, входящего в блюдо;
- приведенная стоимость приготовления одной порции блюда (трудоемкость).

2. Для каждого **поставщика** продуктов:

- код (номер) поставщика продукта;
- название поставщика и его статус (рынок, ферма, универсам и т. п.);
- данные о поставщике (город, адрес, телефон);
- название поставляемого продукта;
- дата поставки и цена на момент поставки.

3. Ежедневное потребление блюд (расход):

- блюдо,
- количество порций,
- дата.

4. Меню на следующий день, где на каждую из трапез (завтрак, обед, ужин) для каждого из видов блюд приводится несколько различных блюд.

5. Выбор каждым из отдыхающих конкретных блюд из меню (для упрощения схемы опущены сведения об отдыхающих и, следовательно, привязка их к местам в столовой пансионата).

Стержни:

- ВИДЫ_БЛЮД,
- ТРАПЕЗЫ,
- ПРОДУКТЫ
- ПОСТАВЩИКИ;

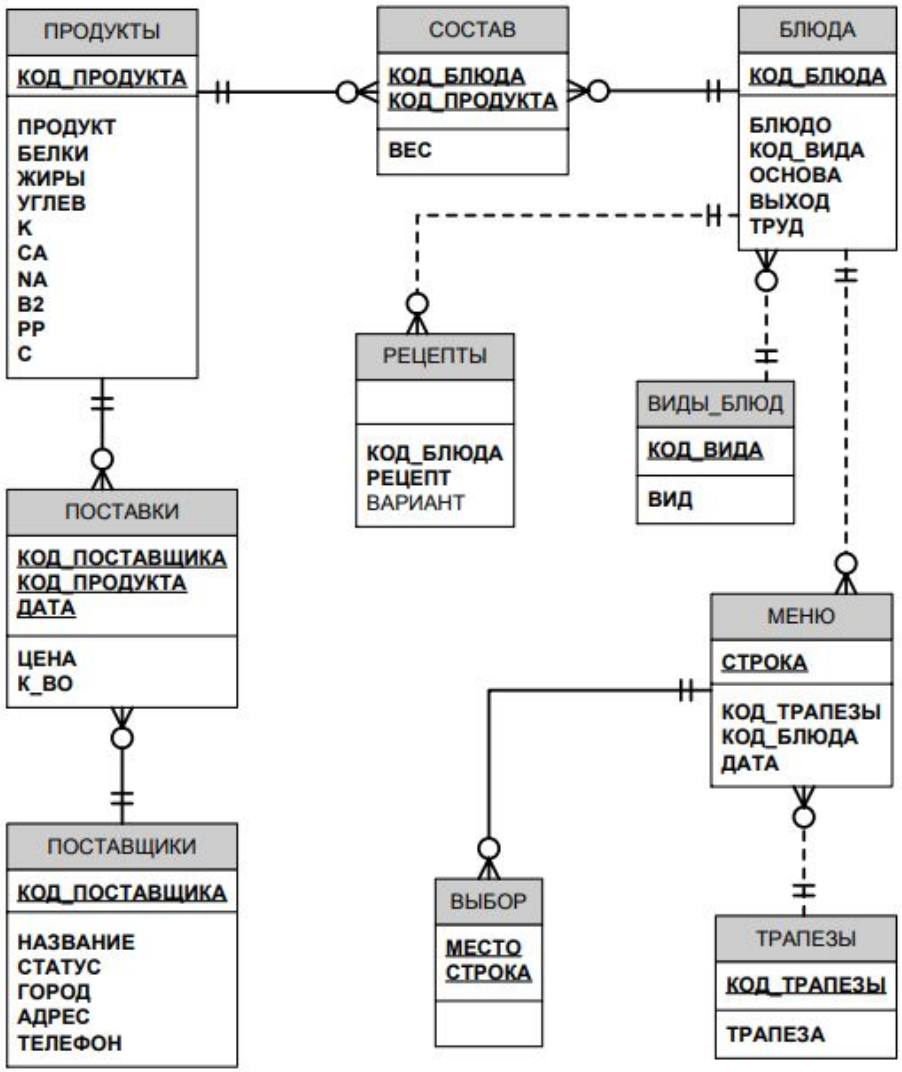
Ассоциации:

- СОСТАВ (связывает БЛЮДА с ПРОДУКТАМИ),
- ПОСТАВКИ (связывает ПОСТАВЩИКОВ с ПРОДУКТАМИ),
- МЕНЮ (связывает ТРАПЕЗЫ с БЛЮДАМИ)
- ВЫБОР (связывает МЕНЮ с МЕСТОМ в столовой)
- БЛЮДА, зависящие от единственной стержневой сущности ВИДЫ_БЛЮД;

Характеристика:

- РЕЦЕПТЫ (характеризует БЛЮДА).

ER-диаграмма модели



2.4. О первичных и внешних ключах

- **Ключ** — это минимальный набор атрибутов отношения, по значениям которых можно однозначно найти требуемый экземпляр сущности.

- Минимальность означает, что исключение из набора любого атрибута не позволяет идентифицировать сущность по оставшимся.

- Каждая сущность должна обладать хотя бы одним возможным ключом.

- Если же возникает ситуация, когда из состава атрибутов сущности не удается создать возможного ключа (естественного ключа), то создают, так называемый, **суррогатный ключ** — автоматически сгенерированное значение, никак не связанное с информационным содержанием сущности.

- https://www.youtube.com/watch?v=43yOFoEOKeI

О первичных и внешних ключах

- Один из возможных естественных ключей или суррогатный ключ принимается за первичный ключ.
- При выборе первичного ключа следует отдавать предпочтение несоставным ключам или ключам, составленным из минимального числа атрибутов.
- Нецелесообразно также использовать ключи с длинными текстовыми значениями (предпочтительнее использовать целочисленные атрибуты).
- Так, для идентификации студента можно использовать либо уникальный номер зачетной книжки, либо набор из фамилии, имени, отчества, номера группы и может быть дополнительных атрибутов, так как не исключено появление в группе двух студентов (а чаще студенток) с одинаковыми фамилиями, именами и отчествами.

О первичных и внешних ключах

- Плохо также использовать в качестве ключа не номер блюда, а его название, например, "Закуска из плавленых сырков "Дружба" с ветчиной и соленым огурцом" или "Заяц в сметане с картофельными крокетами и салатом из красной капусты".

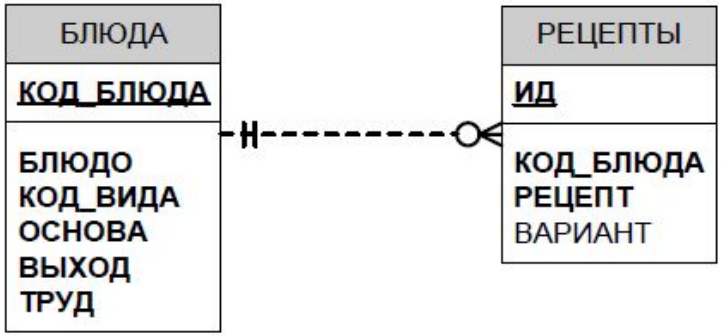
О первичных и внешних ключах

- Не допускается, чтобы первичный ключ стержневой сущности (любой атрибут, участвующий в первичном ключе) принимал неопределенное значение. Иначе возникнет противоречивая ситуация: появится не обладающий индивидуальностью и, следовательно, не существующий экземпляр стержневой сущности. По тем же причинам необходимо обеспечить уникальность первичного ключа.

Суррогатный ключ

- В сущности Рецепты можно было бы использовать в качестве ключа пару (КОД_БЛЮДА, РЕЦЕПТ), но значение РЕЦЕПТ-очень громоздкое.
- Или можно взять пару (КОД_БЛЮДА, ВАРИАНТ), где ВАРИАНТ-номер технологии приготовления блюда. Но, если будет всего один вариант приготовления, то все равно придется вводить номер варианта.

Поэтому введен суррогатный ключ ИД в качестве первичного ключа.



Внешние ключи

- Если сущность В связывает сущности А и Б, то она должна включать внешние ключи, соответствующие первичным ключам сущностей А и Б.

- Если сущность Б характеризует сущность А, то она должна включать внешний ключ, соответствующий первичному ключу сущности А.

3.1. Реляционная структура данных

- В конце 60-х годов исследователь фирмы IBM д-р Эдгар Кодд впервые применил термин "реляционная модель данных”.

- В основе реляционной модели лежит математическое понятие теоретико-множественное отношения.

- Поэтому модель и получила название реляционной (от английского relation — отношение).

основные понятия теории отношений

- Элементы отношения называют **кортежами** (или **записями**).

- Каждый кортеж отношения соответствует одному экземпляру объекта, информацию о котором требуется хранить в базе данных.

- Элементы кортежа принято называть **атрибутами** (или **полями**).

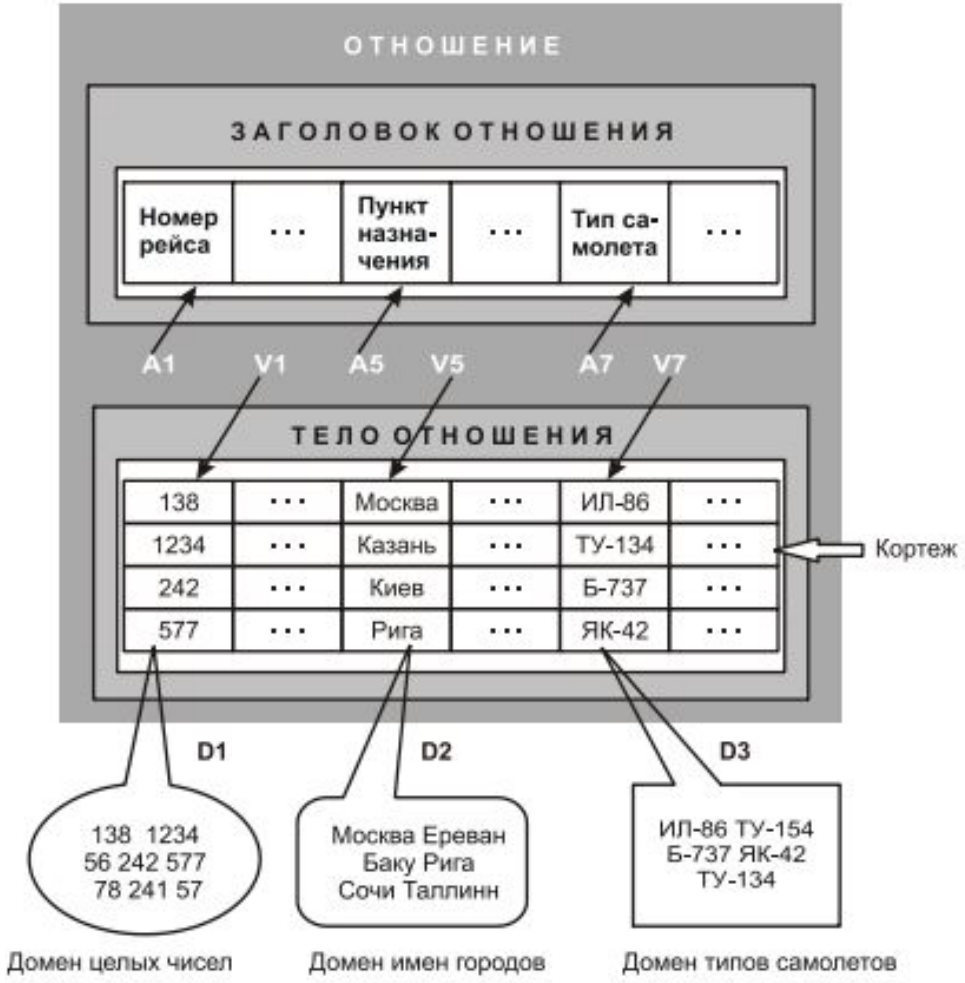
основные понятия теории отношений

Домен-это множество атомарных значений одного и того же типа.

Пусть $A_1, A_2, \ldots, A_n$ имена атрибутов. Каждому имени атрибута A_i соответствует допустимое множество значений, которые может принимать атрибут A_i.

Это множество значений D_i называется **доменом** атрибута A_i =1,n

Отношение на доменах



Домен

- Каждый домен образует значения одного простого типа данных, например, числовые, символьные и др..

- Домен может задаваться перечислением элементов, указанием диапазона значений, функцией и т.д

Декартовое произведением доменов

- ***Декартовым произведением k*** доменов D_1, D_2, D_3,. . . D_k называется множество всех кортежей длины k, т.е. состоящих из k элементов - по одному из каждого домена и определяется следующим образом

$$D = D_1 \times D_2 \times D_3 \times \ldots \times D_k$$

Пример

D_1 = {A,1},
D_2= {B, C},
D_3 = {2,3,D}

$$D = D_1 \times D_2 \times D_3 = \{(A, B, 2), (A, B, 3), (A, B, D), (A, C, 2), (A, C, 3), (A, C, D), (1, B, 2), (1, B, 3), (1, B, D), (1, C, 2), (1, C, 3), (1, C, D)\}.$$

декартово произведение позволяет получить все возможные комбинации элементов исходных множеств - элементов рассматриваемых доменов.

Отношение

- **Отношением** называют некоторое подмножество декартова произведения одного или более доменов.

Например, если построить произведение трёх доменов Должности ('директор', 'бухгалтер', 'водитель', 'продавец'), Оклады (20000 <= x <= 80000), Надбавки (1.1, 1.2, 1.3), то мы получим 4*60001*3=720012 комбинаций. Но реально отношение «Штатное расписание» содержит по одной строке на каждую должность, т.е. является именно подмножеством декартова произведения доменов.

Схема отношений

- **Схемой отношения R** называется перечень имен атрибутов отношения с указанием доменов этих атрибутов и обозначается

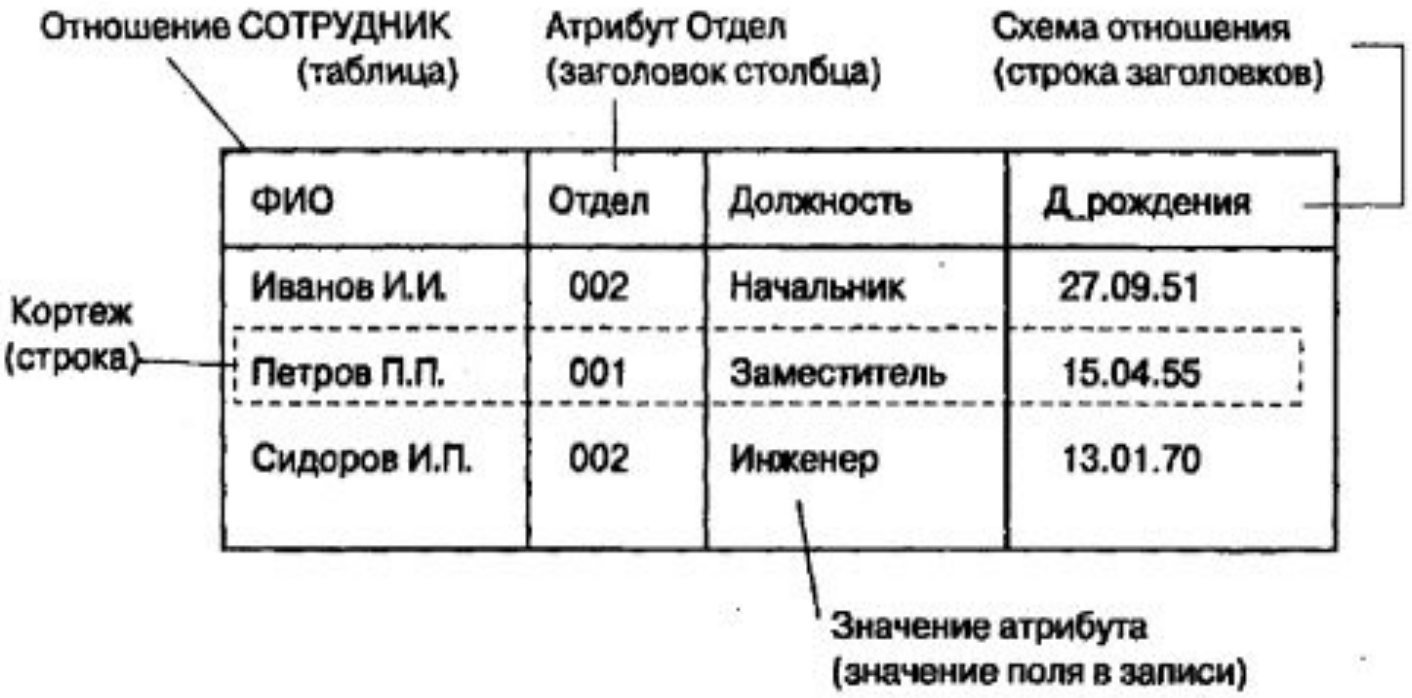
$$R(A_1, A_2, \ldots, A_n);\ \{A_i\} \subseteq D_i,$$

где $\{D_i\}$ – множество значений, принимаемых атрибутом A_i ($i = 1, n$).

Свойства отношения

1. Отношение имеет имя, которое отличается от имен всех других отношений.
2. Каждый атрибут имеет уникальное имя.
3. В отношении нет повторяющихся кортежей.
4. Значения всех атрибутов являются атомарными (неделимыми).
5. Порядок рассмотрения атрибутов в схеме отношения не имеет значения, т.к. для ссылки на значение атрибута в кортеже отношения всегда используется имя атрибута.
6. Порядок рассмотрения кортежей в отношении не имеет значения, т.к. отношение представляет собой множество кортежей, а элементы множества, по определению теории множеств, не упорядочены.

Графическая интерпретация отношения



Реляционная база данных

- Таким образом, реляционная база данных с логической точки зрения может быть представлена множеством двумерных таблиц самого различного предметного наполнения.
- Совокупность схем отношений, используемых для представления предметной области, называется схемой реляционной базы данных.
- **Реляционная схема базы данных** - список, в котором даются имена реляционных таблиц с перечислением их атрибутов (ключи подчеркнуты) и определений внешних ключей.

Описание таблицы (отношения)

<ИМЯ ТАБЛИЦЫ> (<СПИСОК АТРИБУТОВ>),

где список атрибутов - это множество неповторяющихся имен атрибутов, в котором ключевые атрибуты будут выделены подчеркиванием.

Например

СОТРУДНИК(ФИО, Отдел, Должность, Д_рождения)

Определения в реляционной алгебре

Ключ - это минимальный набор атрибутов отношения, однозначно идентифицирующий каждый из его кортежей.

Каждое отношение обладает хотя бы одним ключом. Это гарантируется тем, что в отношении нет повторяющихся кортежей, а это значит, что, по крайней мере, вся совокупность атрибутов обладает свойством однозначной идентификации кортежей отношения.

Определения в реляционной алгебре

Ключ, содержащий более одного атрибута, называется **составным ключом**.

Очень часто отношение может содержать несколько ключей, которые называют возможными (потенциальными) ключами. Один из них принимается за первичный ключ. При выборе первичного ключа следует отдавать предпочтение несоставным ключам. Так, для идентификации студента можно использовать либо уникальный номер зачетной книжки, либо набор из фамилии, имени, отчества, номера группы и может быть дополнительных атрибутов, так как не исключено появление в группе двух студентов с одинаковыми фамилиями, именами и отчествами.

Ключи используются для:

1) исключения дублирования значений в ключевых атрибутах (остальные атрибуты в расчет не принимаются);
2) упорядочения кортежей. Возможно упорядочение по возрастанию или убыванию значений всех ключевых атрибутов;
3) ускорения работы с кортежами отношения;
4) организации связывания отношений.

Определения в реляционной алгебре

Внешние ключи предназначены для установления связей между отношениями.

Столбец одной таблицы, значения в котором совпадают со значениями столбца, являющегося первичным ключом другой таблицы, называется **внешним ключом**.

Пример

- имеются две таблицы

СТУДЕНТ(ФИО, Номер_зач_кн, Группа)
ЭКЗАМЕН(Номер_зач_кн, Дисциплина, Оценка),

В первой таблице первичным ключом является атрибут Номер_зач_кн. Так как каждый студент сдает несколько дисциплин в процессе сессии, то первичным ключом в таблице ЭКЗАМЕН будет набор атрибутов Номер_зач_кн, Дисциплина, т.е. ключ является **составным**.

Оба отношения связаны через атрибуты Номер_зач_кн, которые представляют собой **внешние ключи**.

Ссылочная целостность

- **Ссылочная целостность**-каждому значению внешнего ключа должны соответствовать строки в связываемых отношениях.

Суррогатный ключ

- Кортежи отношения часто могут быть представлены внутренними номерами, которые не имеют смысла вне системы. Внутренний номер часто называют **суррогатным ключом**.

Типы связи таблиц

- При связывании двух таблиц выделяют основную (родительскую) и подчиненную (дочернюю) таблицы.
- Это означает, что одна запись родительской таблицы может быть связана с одной или несколькими записями дочерней таблицы.
- Для поддержки этих связей обе таблицы должны содержать наборы атрибутов, по которым они связаны, т.е. внешние ключи.

Типы связи таблиц

- один - к - одному (1:1);
- один - ко - многим (1 : М);
- многие - ко - многим (М : М).

Тип связи определяется тем, как соотносятся первичные ключи с внешними ключами в обеих таблицах

Связь типа 1:1

Связь один - к - одному означает, что каждая запись одной таблицы соответствует одной записи в другой таблице.

Связь типа 1:1

- ЛИЧНОСТЬ (Код_личности, ФИО, Адрес, Телефон, Дата_рожд.)
- СЛУЖАЩИЙ (Код служащего, Должность, Квалификация (Разряд), Зарплата, Да- та_поступления, Дата_уволнения).

- Связь между этими таблицами поддерживается при помощи внешних ключей, в качестве которых используются совпадающие поля: Код_личности (таблица ЛИЧНОСТЬ) и Код служащего (таблица СЛУЖАЩИЙ).
- Связь между таблицами устанавливается на основании значений совпадающих полей, но не их наименований.

Дополнительно

- На практике связи вида 1:1 используются сравнительно редко, так как хранимую в двух таблицах информацию легко объединить в одну таблицу, которая занимает гораздо меньше места в памяти ЭВМ.
- Возможны случаи, когда удобнее иметь не одну, а две и более таблицы. Причинами этого может быть необходимость ускорить обработку, повысить удобство работы нескольких пользователей с общей информацией, обеспечить более высокую степень защиты информации и т. д.

- Приведем пример, иллюстрирующий последнюю из приведенных причин.
- Пусть имеются сведения о выполняемых в некоторой организации научно-исследовательских работах.
- Эти данные включают в себя следующую информацию по каждой из работ: тему (полное наименование работ), шифр (код), даты начала и завершения работы, количество этапов, головного исполнителя и другую дополнительную информацию.
- Все работы имеют гриф «Для служебного пользования» или «секретно».
- В такой ситуации всю информацию целесообразно хранить в двух таблицах: в одной из них — всю секретную информацию (например, шифр, полное наименование работы и головной исполнитель), а в другой — всю оставшуюся несекретную информацию. Обе таблицы можно связать по шифру работы.
- Первую из таблиц целесообразно защитить от несанкционированного доступа.

Связь типа 1 : M

Связь 1 : M имеет место в случае, когда одной записи родительской таблицы соответствует несколько записей дочерней таблицы.

Пример

- надо описать карьеру некоторого индивидуума. Каждый человек в своей трудовой деятельности сменяет несколько мест работы в разных организациях, где он работает в разных должностях. Кроме того, важно не только отследить переход работника из одной организации в другую, но и прохождение его по служебной лестнице в рамках одной организации. Тогда мы должны создать две таблицы:

СОТРУДНИК (Паспорт, Фамилия, Имя, Отчество)

для моделирования всех работающих людей

КАРЬЕРА (Дата, Место работы, Должность, Номер приказа, Паспорт)

для моделирования записей в их трудовых книжках.

Внешним ключом является поле Паспорт, причем в таблице КАРЬЕРА это поле не является ключевым.

Связь типа M : M

Возникает когда:

- одна запись из первой таблицы может быть связана более чем с одной записью второй таблицы;
- одна запись из второй таблицы может быть связана более чем с одной записью первой таблицы.

Пример

- Преподаватель может вести занятия по разным предметам у разных студентов, а студенты, изучая разные предметы, учатся у нескольких преподавателях.
- Между этими двумя таблицами имеется связь типа много - ко - многим.
- Современные СУБД связь М : М не поддерживают.
- Для того, чтобы организовать такую связь необходимо использовать дополнительную таблицу - таблицу связи, которая связана с каждой таблицей, а тип связи один-ко-многим.

Манипулирование данными в реляционной модели

- Основной единицей обработки данных в реляционной модели является отношение (таблица), а не отдельные его кортежи (записи) БД, и результатом обработки также является отношение.

Манипулирование данными в реляционной модели

Для манипулирования данными в реляционной модели используются два формальных аппарата:

- реляционная алгебра, основанная на теории множеств;

- реляционное исчисление, базирующееся на исчислении предикатов первого порядка.

Оба этих аппарата лежат в основе языков манипулирования данными (ЯМД), предназначенных выполнять соответствующие операции над отношениями (таблицами) для получения информации из баз данных. Наиболее важной частью ЯМД является его раздел для формулировки запросов.

Реляционная алгебра Операторы и операции

- •традиционные операции над множествами: ***объединение, пересечение, вычитание и декартово произведение***
- специальные реляционные операторы: **выборка, проекция, соединение и деление**.
- **операция присваивания**

Основное понятие

- два отношения являются совместимыми по объединению, если имеют одинаковое число атрибутов и i-атрибут одного отношения должен быть определен на том же домене, что и i-атрибут второго отношения.

Объединение (union) (R:=R1 U R2)

- **Объединением** двух совместимых по объединению отношений R1 и R2 является отношение R, включающее в себя все кортежи отношений R1 и R2 без повторов.

Пример

- пусть имеются отношения R1(A,B,C) и R2(

R1

A	B	C
1	3	4
2	5	7
6	8	1
9	3	2
4	5	7

R2

A	B	C
2	4	6
6	8	1
5	7	3
4	5	7

R

A	B	C
1	3	4
2	5	7
6	8	1
9	3	2
4	5	7
2	4	6
5	7	3

- Тогда объединение R:=R1 union R2

Дополнение

- Этот пример, впрочем, как и другие примеры, подтверждает тезис о том, что с помощью одной операции реляционной алгебры решается простейшая информационная задача.
- Операция объединение используется для решения практической задачи - слияние файлов однотипных записей.
- Заметим, что с помощью операции объединения может быть реализовано добавление новой записи к имеющемуся отношению R: =R union R2.
- В этом случае R - исходное отношение, R2- отношение, содержащее одну добавляемую запись.

Пересечение (intersect) (R:=R1 ⋂ R2)

- **Пересечением** двух совместимых по объединению отношений R1 и R2 является отношение R, включающее в себя все кортежи как отношения R1, так и отношения R2.

Пример

- пусть имеются отношение R1(A,B,C) и отношение R2(A,B,C)
- Пересечение R:=R1 intersect R2

R1

A	B	C
1	3	4
2	5	7
6	8	1
9	3	2
4	5	7

R2

A	B	C
2	4	6
6	8	1
5	7	3
4	5	7

R

A	B	C
6	8	1
4	5	7

Разность (except) (R :=R1 - R2)

- Разностью двух совместимых по объединению отношений R1 и R2 является отношение R, кортежи которого принадлежат отношению R1 и не принадлежат отношению R2, т.е. кортежи отношения R1, которых нет в отношении R2.

Пример

- Например, пусть имеются отношение R1(A,B,C) и отношение R2{A,B,C)
- Тогда разность R:=R1 except R2

R1

A	B	C
1	3	4
2	5	7
6	8	1
9	3	2
4	5	7

R2

A	B	C
2	4	6
6	8	1
5	7	3
4	5	7

R

A	B	C
1	3	4
2	5	7
9	3	2

Произведение (product) (R := R1 * R2)

- Это бинарная операция над разносхемными отношениями.
  Если отношение R1(A1,A2, ..., An) имеет m атрибутов, а отношение R2(B1, B2, ..., Bn) - n атрибутов, то произведением является от
  ношение R(A1, A2, ..., Am, B1, B2, ..., Bn), имеющее (m + n) атрибутов.

Пример

Отношение *R1(A,B,C)* и отношение *R2(D,E)*

R:=R1 product *R2*

R1

A	B	C
1	4	7
2	5	8
3	6	9

R2

D	E
a	c
b	f

R

A	B	C	D	E
1	4	7	a	c
1	4	7	b	f
2	5	8	a	c
2	5	8	b	f
3	6	9	a	c
3	6	9	b	f

Селекция (select) (Выборка)

Селекция - унарная операция реляционной алгебры, производящая отбор кортежей из отношения на основании некоторых условий, которые накладываются на значения определённых атрибутов.

Условия

Условия задаются логическим выражением вида:

«Имя атрибута - Знак сравнения - Значение»

Они могут соединяться логическими операторами AND (И), OR (ИЛИ) и иметь оператор NOT (ОТРИЦАНИЕ).

Знаки сравнения: =, <>, >,>=< <=

Пример
отношение СКЛАД

НОМЕНКЛАТУРНЫЙ НОМЕР	НАИМЕНОВАНИЕ	МАРКА	КОЛИЧЕСТВО	ЦЕНА
7120027	Диод	Д18	205	0.75
7120381	Диод	2Д522Б	487	1.80
7121301	Диод	КД521А	803	2.35
7509118	Микросхема	К555ЛА3	64	4.34
7509069	Микросхема	К555ТЛ2	4551	2.17

Найти информацию о диодах, хранящихся на складе, по цене меньше 2 руб.

SELECT (СКЛАД: НАИМЕНОВАНИЕ = 'Диод' AND ЦЕНА < 2.00)

НОМЕНКЛАТУРНЫЙ НОМЕР	НАИМЕНОВАНИЕ	МАРКА	КОЛИЧЕСТВО	ЦЕНА
7120381	Диод	2Д522Б	487	1.80
7121301	Диод	КД521А	803	2.35

Проекция (project)

- Проекция - это унарная операция реляционной алгебры, создающая новое отношение путем исключения атрибутов из существующего отношения.

Пример

Пусть имеется отношение R1 (A,B,C)

Выполним операцию проекция по атрибутам C и A.

В результате получаем отношение *R*

*R*1

A	B	C
1	3	4
2	5	7
6	8	1
9	3	2
4	5	7

R

C	A
4	1
7	2
1	6
2	9
7	4

Соединение (join)

Операция соединения позволяет строить новое отношение посредством конкатенации (сцепления) кортежей двух исходных отношений.

Однако конкатенация производится лишь при выполнении заданного логического условия.

Если условием является равенство значений двух атрибутов исходных отношений, такая операция называется **эквисоединением.**

Естественным называется ***эквисоединение*** по одинаковым атрибутам исходных отношений.

Пример

Пусть имеются отношения *R1(A,B,C)* и *R2(D,E)*

Выполним операцию соединения этих двух отношений при условии, что значения атрибута *B* отношения *R1* меньше значения атрибута *D* таблицы *R2*

R1

A	B	C
1	2	3
4	5	6
7	8	9

R2

D	E
3	1
6	2

R

A	B	C	D	E
1	2	3	3	1
1	2	3	6	2
4	5	6	6	2

Деление (division)

- *Операция деление* в определенном смысле обратна операции произведения.
- Пусть отношение *R1* содержит атрибуты (A1,A2,..., *Am*, *B1, B2,...,Bn*), а отношение *R2*- атрибуты *(B*1, *B2, ...,B*n).
- Тогда результирующее отношение *R* содержит атрибуты *(A1,A2, ..., Am).*
- Кортеж отношения *R1* включается в результирующее отношение, если его декартово произведение с отношением *R2* входит в R1.

Пример

Пример: пусть имеются отношения R1(A,B,C,D) и R2(C,D)

R1

A	B	C	D
a	b	c	d
a	b	e	f
b	c	e	f
e	d	c	d
e	d	e	f
a	b	d	e

R2

C	D
c	d
e	f

R

A	B
a	b
e	d

Свойства операций реляционной алгебры

- ***Коммутативность***

объединение, пересечение, произведение и соединение коммутативны

R1 union R2 = R2 union R1 ;
R1 intersect R2 = R2 intersect R1;
R1 product R2 = R2 product R1 ;
R1 join R2 = R2 join R1.

Ассоциативность

Операции пересечения, объединения, произведения и соединения ассоциативны

R1 intersect (R2 intersect R3) = (R1 intersects R2) intersects R3;

R1 union (R2 union R3) = (R1 union R2) union R3;

R1 product (R2 product R3) = (R1 product R2) product R3;

R1 join (R2 join R3) = (R1 join R2) join R3.

Дистрибутивность

Пересечение дистрибутивно относительно объединения:

R1 intersect (R2 union R3) = (R1 intersect R2) union (R1 intersect R3).

Объединение дистрибутивно относительно пересечения:

R1 union (R2 intersect R3) = (R1 union R1) intersect (R1 union R3).

Произведение и соединение дистрибутивны относительно операций объединения и пересечения

R1 product (R2 union R3) = (R1 product R2) union (R1 product R3);

R1 product (R2 intersect R3) = (R1 product R2) intersect (R1 productR3).

Достоинства реляционной модели данных

Достоинство реляционной модели данных заключается в простоте, понятности и удобстве физической реализации на ЭВМ.

Именно простота и понятность для пользователя явились основной причиной их широкого использования.

Проблемы же эффективности обработки данных этого типа оказались технически вполне разрешимыми.

Недостатки реляционной модели данных

В качестве основного недостатка реляционной модели можно указать дублирование информации для организации связей между таблицами.

Например, база данных имеет следующую схему:

R1(A1, A2, A3, A4)

R2(A5,A6)

R3(A7, A8)

R4(A1, A5)

R5(A1,A7)

R6(A5, A7)

где R1, R2, R3, R4, R5, R6 - это таблицы,

AI, A2, A3, A4, A5, A6, A7 - это атрибуты таблиц.

Не трудно заметить, атрибуты A1, A5, A8 присутствуют в нескольких таблицах.

Ограничения целостности базы данных

- В любой момент реляционная база данных содержит некоторую определенную конфигурацию данных, и эта конфигурация должна отражать реальную действительность (целостность данных).

- Следовательно, определение базы данных нуждается в расширении, включающем правила целостности данных, назначение которых в том, чтобы информировать СУБД о разного рода ограничениях реального мира.

Примеры правил целостности

- количество продаваемого товара должно быть больше 0;
- номер паспорта является уникальным значением;
- возраст принимаемого сотрудника на работу обязательно не меньше 14 лет.

Ограничения целостности

Ограничения целостности - это правила, которым должны удовлетворять значения данных в БД.

Ограничения целостности включаются в структуру базы данных с помощью средств языка SQL.

Ограничения целостности

За выполнением ограничений целостности следит СУБД в процессе своего функционирования.

Она проверяет ограничения целостности каждый раз, когда они могут быть нарушены (например, при добавлении данных, при удалении данных и т.п.), и гарантирует их соблюдение.

Если какая-либо команда нарушает ограничение целостности, она не будет выполнена и система выдаст соответствующее сообщение об ошибке.

Например

Если задать в качестве ограничения правило «**Остаток денежных средств на счёте не может быть отрицательным**», то при попытке снять со счёта денег больше, чем там есть, система выдаст сообщение об ошибке и не позволит выполнить эту операцию.

Ограничения целостности

- Таким образом, ограничения целостности обеспечивают логическую непротиворечивость данных при переводе БД из одного состояния в другое.
- Более сложные правила целостности реализуются с помощью хранимых процедур и триггеров, представляющие определенные последовательности команд языка SQL при изменении или добавлении данных в БД.

Выбор СУБД

Выбор СУБД является одним из важнейших моментов в разработке проекта БД, так как он принципиальным образом влияет на процесс проектирования БД и реализации информационной системы.

Теоретически при осуществлении этого выбора нужно принимать во внимание десятки факторов.

Но на практике разработчики руководствуются лишь собственной интуицией и несколькими наиболее важными критериями.

Выбор СУБД

Наиболее важные критерии выбора СУБД:

- тип модели данных, которую поддерживает данная СУБД, адекватность модели данных структуре рассматриваемой предметной области;
- характеристики производительности СУБД;
- запас функциональных возможностей для дальнейшего развития информационной системы;
- степень оснащённости СУБД инструментарием для персонала администрирования данными;
- удобство и надежность СУБД в эксплуатации;
- наличие специалистов по работе с конкретной СУБД;
- стоимость СУБД и дополнительного программного обеспечения.

Нормализация

Для поддержания БД в устойчивом состоянии используется ряд механизмов, которые называются средствами поддержки целостности.

Приведение структуры БД в соответствие этим ограничениям - это и есть **нормализация.**

Суть этих ограничений : каждый факт, хранимый в БД, должен храниться один-единственный раз, поскольку дублирование может привести к несогласованности между копиями одной и той же информации.

Следует избегать любых неоднозначностей, а также избыточности хранимой информации.

Нормализация

Выделяются шесть нормальных форм, пять из которых так и называются: *первая, вторая, третья, четвертая, пятая нормальная форма*, а также *нормальная форма Бойса-Кодда*, лежащая между третьей и четвертой.

Для реляционной модели данных разработано несколько нормализованных форм, три из которых являются основными.

База данных считается ***нормализованной***, если ее таблицы представлены как минимум в третьей нормальной форме.

Часто многие таблицы нормализуются до четвертой нормальной формы, иногда, наоборот, производится *денормализация*. Использования таблиц в пятой нормальной форме в реальных базах данных встречается редко.

Нормализация

При использовании универсального отношения возникают две проблемы:

- избыточность данных;
- потенциальная противоречивость (аномалии).

Под **избыточностью** понимают повторение данных в разных строках одной таблицы или в разных таблицах БД.

Аномалии

Аномалии – это проблемы, возникающие в данных из-за дефектов проектирования БД.

Аномалия приводит к противоречию в БД либо существенно усложняет обработку БД.

Причиной является излишнее дублирование данных в таблице, которое вызывается наличием функциональных зависимостей от не ключевых атрибутов.

Существуют три вида аномалий:

- вставки;
- удаления;
- модификации.

Аномалия вставки

Аномалии вставки (добавления) проявляются при вводе данных в дефектную таблицу.

Такие аномалии возникают, когда информацию в таблицу нельзя поместить, пока она не полная, либо вставка записи требует дополнительного просмотра таблицы. Для вставки данных нужно добавлять (выгадывать) лишние (несуществующие) данные.

При аномалии вставки часть составного ключа оказывается неопределенной (нарушается принцип целостности).

Аномалия вставки

Пример. Задана следующая база данных, которая основана на одной таблице. В таблице определяется информация о преподавателях (Преподаватель, Дисциплина, Кафедра), студентах (Студент, Номер зачетки, Адрес), успеваемость студентов (Оценка).

Номер	Студент	Номер зачетки	Адрес	Дисциплина	Преподаватель	Кафедра	Оценка
1	Иванов И.И.	2535	г. Москва	Физика	Петренко М.М.	Природоведческие дисциплины	4
2	Иванов И.И.	2535	г. Москва	Химия	Бауман М.В.	Природоведческие дисциплины	3
3	Петров П.П.	2580	г. Киев	Физика	Петренко М.М.	Природоведческие дисциплины	4
4	Сидоров С.С.	2676	г. Харьков	Физика	Петренко М.М.	Природоведческие дисциплины	5
5	Сидоров С.С.	2676	г. Харьков	Химия	Бауман М.В.	Природоведческие дисциплины	3
6	Иванов И.И.	2535	г. Москва	Информатика	Лев М.К.	Математические дисциплины	4

Аномалия вставки

Пусть в эту базу данных нужно добавить нового преподавателя математики (столбцы Преподаватель, Дисциплина), который недавно принят на работу.

Для этого необходимо, чтобы новый преподаватель обязательно оценил хотя бы одного студента. Иначе, в таком представлении базы данных, добавить данные будет невозможно. Значит, при добавлении преподавателя, нужно выгадывать несуществующие данные оценивания студента.

Это и есть *аномалия вставки*.

Аномалия вставки

Номер	Студент	Номер зачетки	Адрес	Дисциплина	Преподаватель	Кафедра	Оценка
1	Иванов И.И.	2535	г. Москва	Физика	Петренко М.М.	Природоведческие дисциплины	4
2	Иванов И.И.	2535	г. Москва	Химия	Бауман М.В.	Природоведческие дисциплины	3
3	Петров П.П.	2580	г. Киев	Физика	Петренко М.М.	Природоведческие дисциплины	4
4	Сидоров С.С.	2676	г. Харьков	Физика	Петренко М.М.	Природоведческие дисциплины	5
5	Сидоров С.С.	2676	г. Харьков	Химия	Бауман М.В.	Природоведческие дисциплины	3
6	Иванов И.И.	2535	г. Москва	Информатика	Левитан М.К.	Математические дисциплины	4
7	???	???	???	Математика	Савченко М.И.	Математические дисциплины	???

Дисциплина	Преподаватель	Кафедра
Математика	Савченко М.И.	Математические дисципліни

Аномалия удаления

Аномалии удаления возникают при удалении данных из дефектной схемы.

Аномалия удаления – это потеря одних данных в таблице при удалении других данных в таблице.

Аномалия удаления

Пример. Пусть в таблице базы данных по ошибке было введено оценивание по дисциплине Информатика, которую перенесли на следующие семестры обучения. Автоматически, при удалении строки с дисциплиной «Информатика», будет потеряна строка с данными о преподавателе (Левитан М.К), который преподает эту дисциплину и наз

Номер	Студент	Номер зачетки	Адрес	Дисциплина	Преподаватель	Кафедра	Оценка
1	Иванов И.И.	2535	г. Москва	Физика	Петренко М.М.	Природоведческие дисциплины	4
2	Иванов И.И.	2535	г. Москва	Химия	Бауман М.В.	Природоведческие дисциплины	3
3	Петров П.П.	2580	г. Киев	Физика	Петренко М.М.	Природоведческие дисциплины	4
4	Сидоров С.С.	2676	г. Харьков	Физика	Петренко М.М.	Природоведческие дисциплины	5
5	Сидоров С.С.	2676	г. Харьков	Химия	Бауман М.В.	Природоведческие дисциплины	3

удаление

6	Иванов И.И.	2535	г. Москва	Информатика	Левитан М.К.	Математические дисциплины	4

Аномалия корректировки

Аномалии корректировки (модификации, редактирования) возникают при изменении данных дефектной схемы. Изменения, вносимые в одну запись требуют внесения изменений в другие записи.

Аномалия редактирования возникает в случаях, когда в таблице базы данных существуют повторяющиеся данные. Такие данные тяжело обновлять при их редактировании, поскольку нужно вносить изменения во все ячейки таблицы, в которых эти данные фигурируют. Если при изменении повторяемых данных в одной ячейке не изменить так же эти данные в других ячейках, то компьютер будет воспринимать эти данные как разные (в отличие от человека).

Аномалия корректировки – это вынужденная необходимость изменения (обновления) данных во всей таблице в случае их изменения (обновления) в одной ячейке таблицы с целью избежание их двузначного трактования.

Аномалия корректировки

Пример. Задана таблица базы данных учета успеваемости в учебном заведении. Пусть преподаватель физики Петренко М.М. вышла замуж и изменила фамилию на Маркевич. Теперь во всех ячейках столбца (атрибута) Преподаватель нужно изменить имя преподавателя Петренко М.М. на Маркевич М.М.

Петренко М.М. => Маркевич М.М.

Номер	Студент	Номер зачетки	Адрес	Дисциплина	Препод	Кафедра	Оценка
1	Иванов И.И.	2535	г. Москва	Физика	~~Петренко М.М.~~ Маркевич М.М.	Природоведческие дисциплины	4
2	Іванов І.І.	2535	г. Москва	Химия	Бауман М.В.	Природоведческие дисциплины	3
3	Петров П.П.	2580	г. Киев	Физика	~~Петренко М.М.~~ Маркевич М.М.	Природоведческие дисциплины	4
4	Сідоров С.С.	2676	г. Харьков	Физика	~~Петренко М.М.~~ Маркевич М.М.	Природоведческие дисциплины	5
5	Сідоров С.С.	2676	г. Харьков	Химия	Бауман М.В.	Природоведческие дисциплины	3
6	Іванов І.І.	2535	г. Москва	Информатика	Левитан М.К.	Математические дисциплины	4

Декомпозиция

Существует научно-обоснованные приемы для того, что бы избежать таких аномалий. Это делается с помощью декомпозиции.

Декомпозиция – это разделение одной таблицы на несколько, но так, что бы информация не терялась и чтобы можно было собрать информацию в таблицу.

Процедура декомпозиции называется **нормализацией.**

https://www.youtube.com/watch?v=zqQxWdTpSIA

Нормальные формы

Различают:

- 1НФ – первая нормальная форма
- 2НФ – вторая нормальная форма
- 3НФ – третья нормальная форма
- НФБК – нормальная форма Бойса-Кодда
- 4НФ – четвертая нормальная форма
- 5НФ – пятая нормальная форма

Нормальные формы

Каждая нормальная форма налагает определенные ограничения на данные. Каждая нормальная форма более высокого уровня предполагает, что анализируемая таблица уже находится в нормальной форме на уровень ниже рассматриваемой.

В ходе нормализации схема базы данных становится все более строгой, а ее таблицы все менее подвержены различного рода аномалиям. Для реляционных баз данных необходимо, чтобы ее таблицы находились в 1НФ. Нормальные формы более высоких уровней могут использоваться разработчиками по своему усмотрению. Однако грамотный специалист стремится к тому, чтобы довести уровень нормализации базы данных хотя бы до 3НФ, тем самым исключив избыточность данных и аномалии обновления.

НФБК, 4НФ и 5НФ используются крайне редко. Поэтому рассмотрим только первые три.

Первая нормальная форма

Таблица находится в **первой нормальной форме**, если все ее поля имеют простые (атомарные) значения, т. е. каждый её атрибут атомарен.

Под выражением «атрибут атомарен» понимается, что атрибут может содержать только одно значение.

Так же можно сказать, что таблица находится в **первой нормальной форме**, если она является реляционной таблицей.

Таблица, находящаяся в 1НФ должна отвечать следующим требованиям:

- таблица не должна иметь повторяющихся записей;
- в таблице должны отсутствовать повторяющиеся группы полей.

Первая нормальная форма

Для приведения к 1НФ можно использовать следующий алгоритм:

1. Определить поле, которое можно назначить первичным ключом. Если такого поля нет, то добавить новое уникальное ключевое поле.
2. Определить группы повторяющихся полей.
3. Вынести группы повторяющихся полей в отдельные таблицы, в основной таблице остается одно поле для организации связи между таблицами.
4. Назначить первичные ключи в новых таблицах. (В качестве ключевых полей можно использовать поля таблицы или добавить новое поле. Если ключевое поле имеет большой размер, предпочтительней добавлять новое поле.)
5. Определить тип отношения между таблицами.

Вторая нормальная форма

Таблица, находящаяся во **второй нормальной форме** должна отвечать всем требованиям 1НФ, а также любое не ключевое поле однозначно идентифицируется полным набором ключевых полей, т.е. все ее ключевые поля зависят от всего ключа и не существует неключевых полей, которые зависят от части всего ключа.

Неключевое поле - поле, не входящее в состав никакого ключа.

2НФ применяется к таблицам, которые имеют составной ключ. Если ключ является простым, то таблица автоматически находится во 2НФ.

Третья нормальная форма

Таблица, находящаяся в **третьей нормальной форме** должна отвечать всем требованиям 2НФ, а также ни одно из не ключевых полей не идентифицируется при помощи другого не ключевого поля.

Другими словами, в таблице нет полей, которые не зависят от ключа. Среди неключевых полей нет функциональной зависимости, т. е. все они взаимонезависемы.

3.1. Общие сведения о языке SQL

- **Язык SQL (Structured Query Language)** - Структурированный Язык Запросов - предназначен для работы с реляционными базами данных.

- Он дает возможность выполнять операции над таблицами (создание, удаление, изменение структуры) и над данными таблиц (выборка, изменение, добавление, удаление).

История SQL

- История возникновения языка SQL начинается в 1970 году, когда доктор Э.Ф. Кодд предложил реляционную модель в качестве новой модели базы данных.

- Для доказательства жизнеспособности новой модели данных в компании IBM был создан мощный исследовательский проект, получивший название System/R.

- Проект включал разработку собственно реляционной СУБД и специального языка запросов к базе данных. Первоначально язык получил название SEQUEL (Structured English Query Language) - структурированный английский язык запросов и произносился как «сиквэл».

- Позже по юридическим соображениям («SEQUEL» был товарной маркой британской авиастроительной группы компаний «Hawker Siddeley») язык SEQUEL был переименован в язык SQL (Structured Query Language) и официальное произношение стало побуквенным «эс-кью-эль».

История SQL

- В дальнейшем этот язык применялся во многих коммерческих СУБД и в силу своего широкого распространения постепенно стал стандартом “де-факто” для языков манипулирования данными в реляционных СУБД.
- С момента создания и до наших дней язык SQL претерпел массу изменений, но идеология осталась неизменной.
- Элегантность и независимость от специфики компьютерных технологий, а также его поддержка лидерами промышленности в области технологии реляционных баз данных, сделало SQL основным стандартным языком. По этой причине, любой, кто хочет работать с базами данных, должен знать SQL.

Особенности SQL

- Язык SQL является непроцедурным языком, с помощью которого программист определяет только требуемый результат, не указывая алгоритм его достижения.
- Поэтому первоначально он не содержал команды управления ходом вычислительного процесса, организации подпрограмм, ввода-вывода, описания типов и многое другое, что присуще традиционным языкам программирования.
- В связи с этим язык SQL автономно не используется. Обычно команды SQL встраиваются в язык программирования СУБД.
- Кроме того, команды SQL могут выполняться непосредственно в интерактивном режиме.

Особенности SQL

- В архитектуре «клиент-сервер» язык SQL занимает очень важное место.
- Именно он используется как язык общения клиентского программного обеспечения с серверной СУБД, расположенной на удаленном компьютере.
- Так, клиент посылает серверу запрос на языке SQL, а сервер разбирает его, интерпретирует, выбирает план выполнения, выполняет запрос и отсылает клиенту результат.

Особенности SQL

- Несмотря на то, что стандарты обозначают некоторое общее понимание того, каким должен быть язык взаимодействия с базой данных, различные производители реализуют его в своих программных продуктах (СУБД) по-разному.
- Связано это с тем, что для расширения функциональных возможностей и повышения эффективности разработчики конкретной СУБД добавляют к стандартному языку SQL дополнительные команды и функции, исходя из собственного понимания их необходимости, сохраняя при этом некоторые особенности предыдущих версий.
- Поскольку сферы интересов пользователей различных СУБД отличаются друг от друга, различаются и создаваемые расширения.
- Таким образом, существует несколько диалектов языка SQL.