Базы данных

Гаврилов Александр Викторович
к.т.н., доцент

Тема 2. Логическая организация баз данных

Лекция 2

Вопросы лекции:

1. Модели данных
2. Реляционная модель данных
3. Информационно-логическая модель предметной области

1. Модели данных

Основные термины и определения

Модель данных – это формальная теория представления и обработки данных в системе управления базами данных (СУБД), которая включает, по меньшей мере, три аспекта: [ГОСТ 20886-85]. **Модель данных** включает три аспекта:

1) **аспект структуры:** методы описания типов и логических структур данных в базе данных;

2) **аспект манипуляции:** методы манипулирования данными;

3) **аспект целостности:** методы описания и поддержки целостности базы данных.

Э. Кодд. «Модели данных в управлении базами данных»

Модель данных ≠ Модель (схема) базы данных

аналогично тому, что

Язык программирования ≠ Программа

Модель данных — это абстрактное, самодостаточное, логическое определение объектов, операторов и прочих элементов, в совокупности составляющих абстрактную машину доступа к данным, с которой взаимодействует пользователь. Эти *объекты позволяют моделировать структуру данных*, а *операторы — поведение данных*.

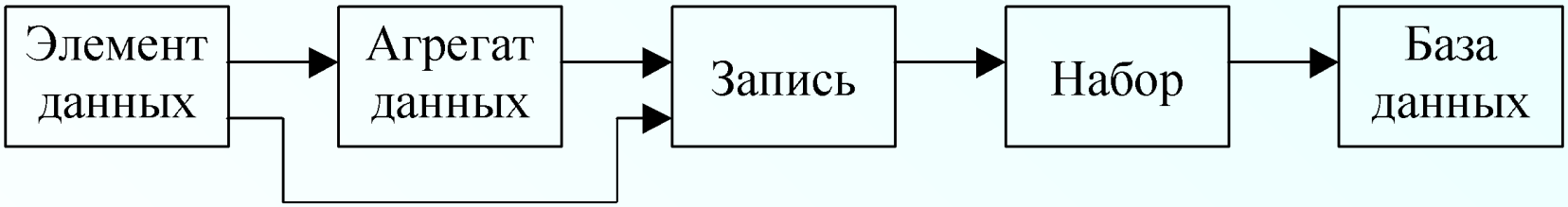
Каждая БД и СУБД строится на основе некоторой явной или неявной модели данных. Все СУБД, построенные на одной и той же модели данных, относят к одному типу. Например, основой реляционных СУБД является реляционная модель данных, иерархических СУБД — иерархическая модель данных и т. д.

Модель данных состоит из трёх частей:

1. **Набор типов структур данных**.
2. **Набор операторов** или правил вывода, которые могут быть применены к любым правильным примерам типов данных, перечисленных в (1), чтобы находить, выводить или преобразовывать информацию, содержащуюся в любых частях этих структур в любых комбинациях.
3. **Набор общих правил целостности**, которые прямо или косвенно определяют множество непротиворечивых состояний базы данных и/или множество изменений её состояния.

Типы структур данных. Версия CODASYL

Структуризация данных базируется на использовании концепций "агрегации" и "обобщения". Один из первых вариантов структуризации данных был предложен Ассоциацией по языкам обработки данных (Conference on Data Systems Languages, CODASYL):

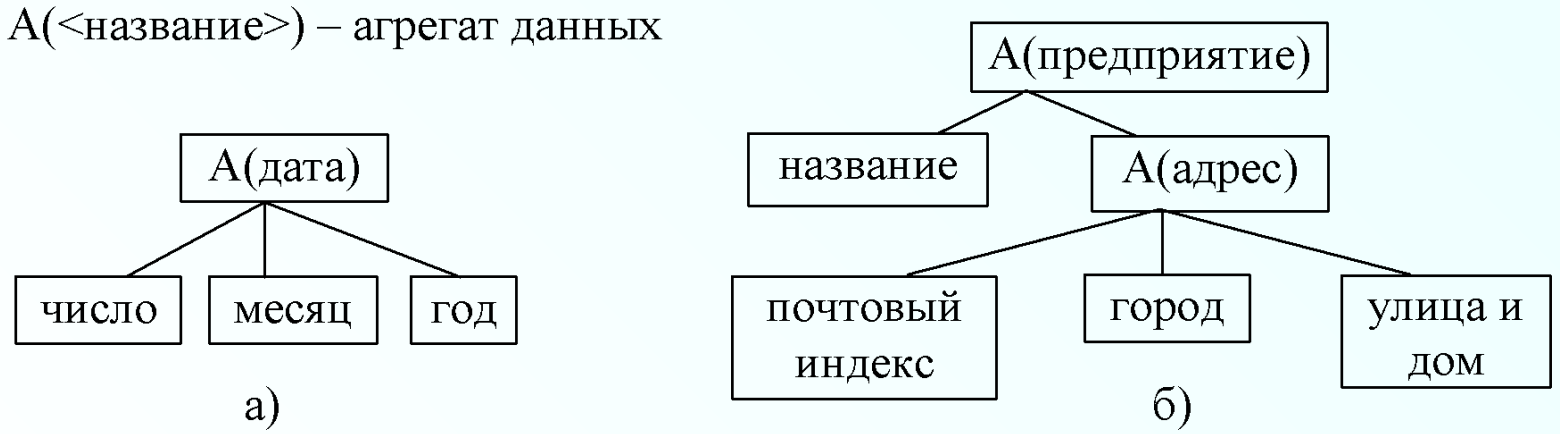


Элемент данных – наименьшая поименованная единица данных, к которой СУБД может обращаться непосредственно и с помощью которой выполняется построение всех остальных структур.

Для каждого элемента данных должны быть определены **название** и **тип**.

Версия CODASYL. Агрегаты

Агрегат данных – поименованная совокупность элементов данных внутри записи, которую можно рассматривать как единое целое. Агрегат может быть **простым** (включающим только элементы данных, рис. а) и **составным** (включающим наряду с элементами данных и другие агрегаты, рис. б).

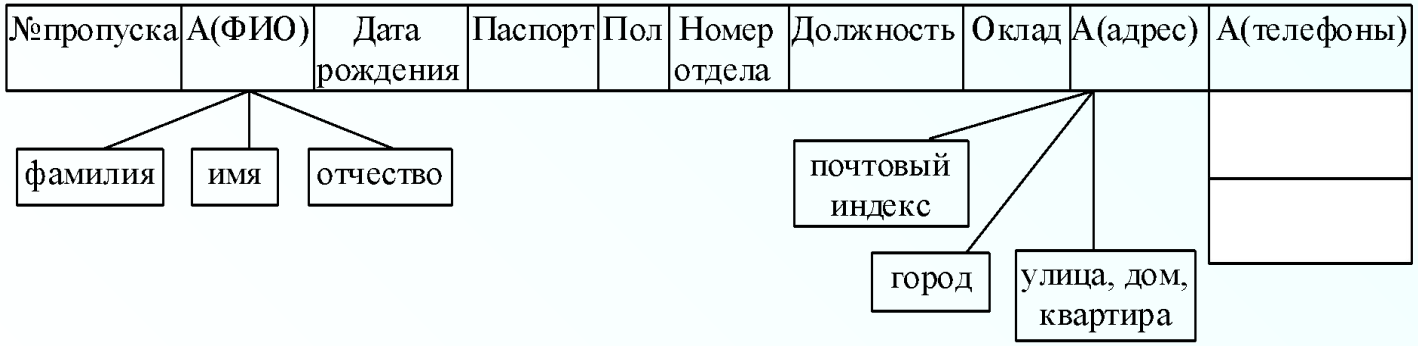


Для каждого агрегата должны быть определены название и структура.

Версия CODASYL. Запись

Запись – поименованная совокупность элементов данных или элементов данных и агрегатов. Запись – это агрегат, не входящий в состав никакого другого агрегата; она может иметь сложную иерархическую структуру, поскольку допускается многократное применение агрегации. Различают **тип записи** (её структуру) и **экземпляр записи**, т.е. запись с конкретными значениями элементов данных. Одна запись описывает свойства одной сущности ПО (экземпляра). Иногда термин "запись" заменяют термином "группа".

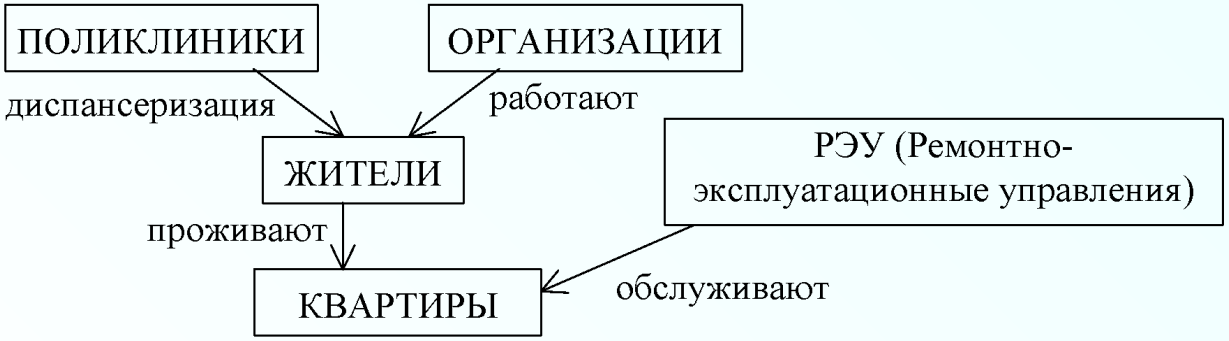
Пример записи типа "Сотрудник":



Версия CODASYL. Набор. База данных

Набор (или групповое отношение) – поименованная совокупность записей, образующих двухуровневую иерархическую структуру. Каждый тип набора представляет собой связь между двумя или несколькими типами записей. Для каждого типа набора один тип записи объявляется владельцем набора, остальные типы записи объявляются членами набора. Для группового отношения также различают тип и экземпляр.

Фрагмент диаграммы Бахмана для БД "Город":



База данных – поименованная совокупность экземпляров групп и групповых отношений.

Операции над данными

Модель данных определяет множество действий, которые допустимо производить над некоторой реализацией БД для её перевода из одного состояния в другое. Это множество соотносят с языком манипулирования данными (**Data Manipulation Language, DML**).

Любая операция над данными включает в себя **селекцию данных** (select). Условие селекции – это некоторый критерий отбора данных, в котором могут быть использованы логическая позиция элемента данных, его значение и связи между данными.

По типу производимых действий различают следующие операции:

- **идентификация данных и нахождение их позиции в БД;**
- **выборка (чтение) данных из БД;**
- **включение (запись) данных в БД;**
- **удаление данных из БД;**
- **модификация (изменение) данных БД.**

Обработка данных в БД осуществляется с помощью процедур базы данных – транзакций. Транзакцией называют упорядоченное множество операций, переводящих БД из одного согласованного состояния в другое.

Ограничения целостности

Ограничения целостности – это правила, которым должны удовлетворять значения элементов данных. Ограничения целостности делятся на:

- **явные** (включаются в структуру базы данных с помощью средств языка контроля данных (DCL, Data Control Language))
- **неявные** (определяются самой структурой данных).

Также различают **статические** и **динамические** ограничения целостности. Статические ограничения присущи всем состояниям ПО, а динамические определяют возможность перехода ПО из одного состояния в другое.

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.

Таким образом, **ограничения целостности обеспечивают логическую непротиворечивость данных при переводе БД из одного состояния в другое.**

Иерархическая модель данных

— это модель данных, где используется представление базы данных в виде древовидной (иерархической) структуры, состоящей из объектов (данных) различных уровней.

Между объектами существуют связи, каждый объект (**предок**) **может** включать в себя несколько объектов более низкого уровня (**потомков**).

В иерархической модели автоматически поддерживается **целостность ссылок** между предками и потомками. Основное правило: **никакой потомок не может существовать без своего родителя**.

Типичным представителем (наиболее известным и распространенным) является СУБД **IMS (Information Management System)** компании **IBM**. Первая версия системы появилась в 1968 г.

Иерархическая модель данных

К основным понятиям иерархической структуры относятся: **элемент данных (атрибут), запись, групповое отношение.**

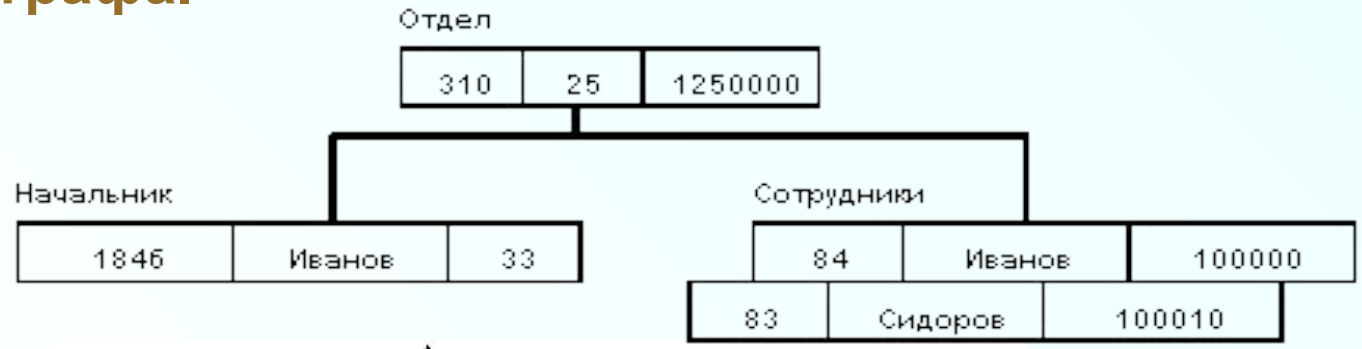
Атрибут (элемент данных) – наименьшая единица структуры данных. Обычно каждому элементу при описании базы данных присваивается уникальное имя. По этому имени к нему обращаются при обработке. Элемент данных также часто называют полем.

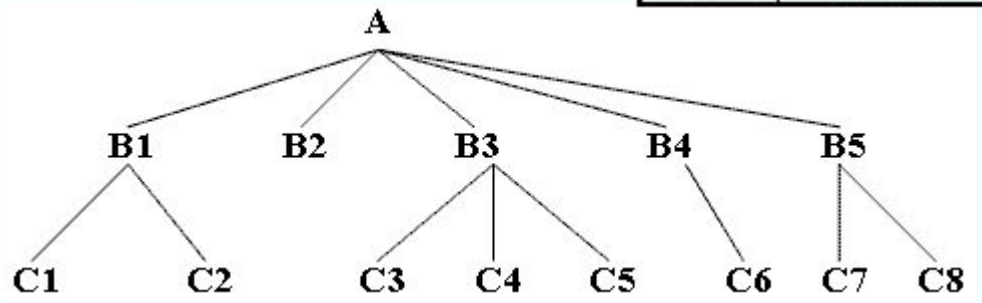
Групповое отношение – иерархическое отношение между записями двух типов. Родительская запись (владелец группового отношения) называется исходной записью, а дочерние записи (члены группового отношения) - подчиненными. Иерархическая база данных может хранить только такие древовидные структуры.

Иерархическая модель данных

Запись – именованная совокупность атрибутов. Использование записей позволяет за одно обращение к базе получить некоторую логически связанную совокупность данных. Именно записи изменяются, добавляются и удаляются. **Тип записи** определяется составом ее атрибутов. **Экземпляр записи** – конкретная запись с конкретным значением элементов.

На схеме иерархического дерева записи представляются вершинами графа.





Иерархическая модель данных

Корневая запись каждого дерева обязательно должна содержать **ключ** с уникальным значением. Ключи некорневых записей должны иметь уникальное значение только в рамках группового отношения. Каждая запись идентифицируется полным сцепленным ключом, под которым понимается совокупность ключей всех записей от корневой, по иерархическому пути.

При графическом изображении групповые отношения изображают дугами ориентированного графа, а типы записей - вершинами (диаграмма Бахмана).

Для групповых отношений в иерархической модели обеспечивается автоматический режим включения и **фиксированное членство**. Это означает, что для запоминания любой некорневой записи в БД должна существовать ее родительская запись.

Пример иерархической модели данных

Пример

Рассмотрим следующую модель данных предприятия: предприятие состоит из отделов, в которых работают сотрудники. В каждом отделе может работать несколько сотрудников, но сотрудник не может работать более чем в одном отделе.

Поэтому, для информационной системы управления персоналом необходимо создать групповое отношение, состоящее из родительской записи
ОТДЕЛ (НАИМЕНОВАНИЕ_ОТДЕЛА, ЧИСЛО_РАБОТНИКОВ)
и дочерней записи
СОТРУДНИК (ФАМИЛИЯ, ДОЛЖНОСТЬ, ОКЛАД).

Пример

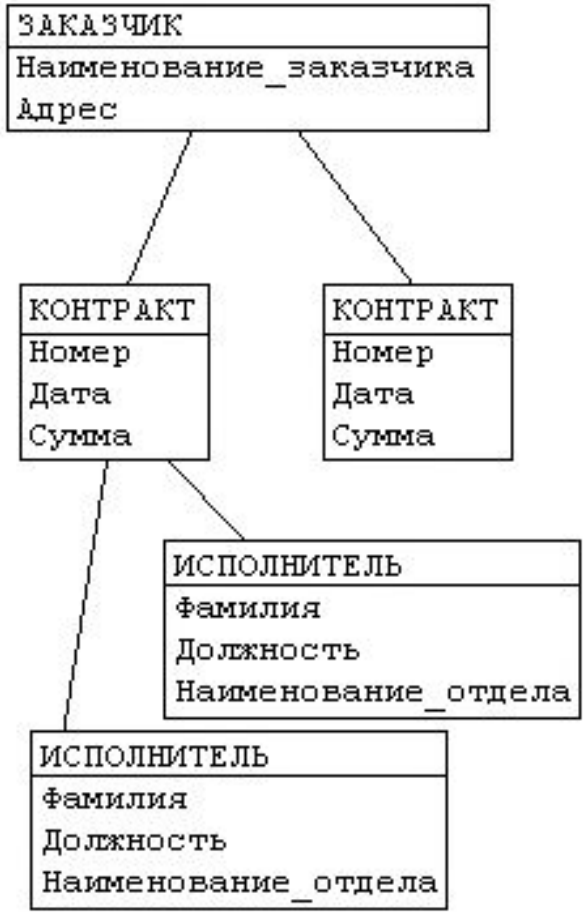
Для автоматизации учета контрактов с заказчиками необходимо создание еще одной иерархической структуры: заказчик - контракты с ним - сотрудники, задействованные в работе над контрактом.

Это дерево будет включать записи:

ЗАКАЗЧИК (НАИМЕНОВАНИЕ_ЗАКАЗЧИКА, АДРЕС),

КОНТРАКТ(НОМЕР, ДАТА,СУММА),

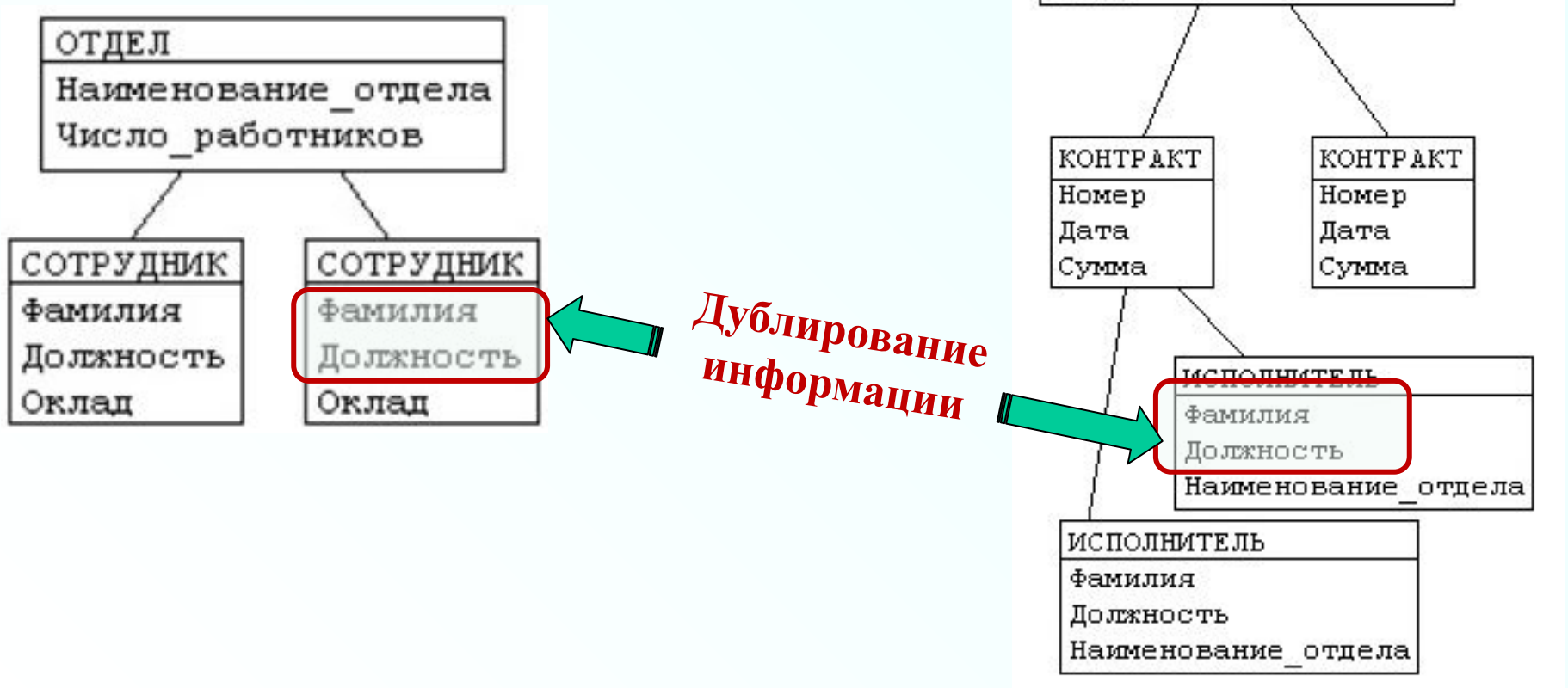
ИСПОЛНИТЕЛЬ (ФАМИЛИЯ, ДОЛЖНОСТЬ, НАИМЕНОВАНИЕ_ОТДЕЛА).



Пример

Из примера видны недостатки иерархических БД:

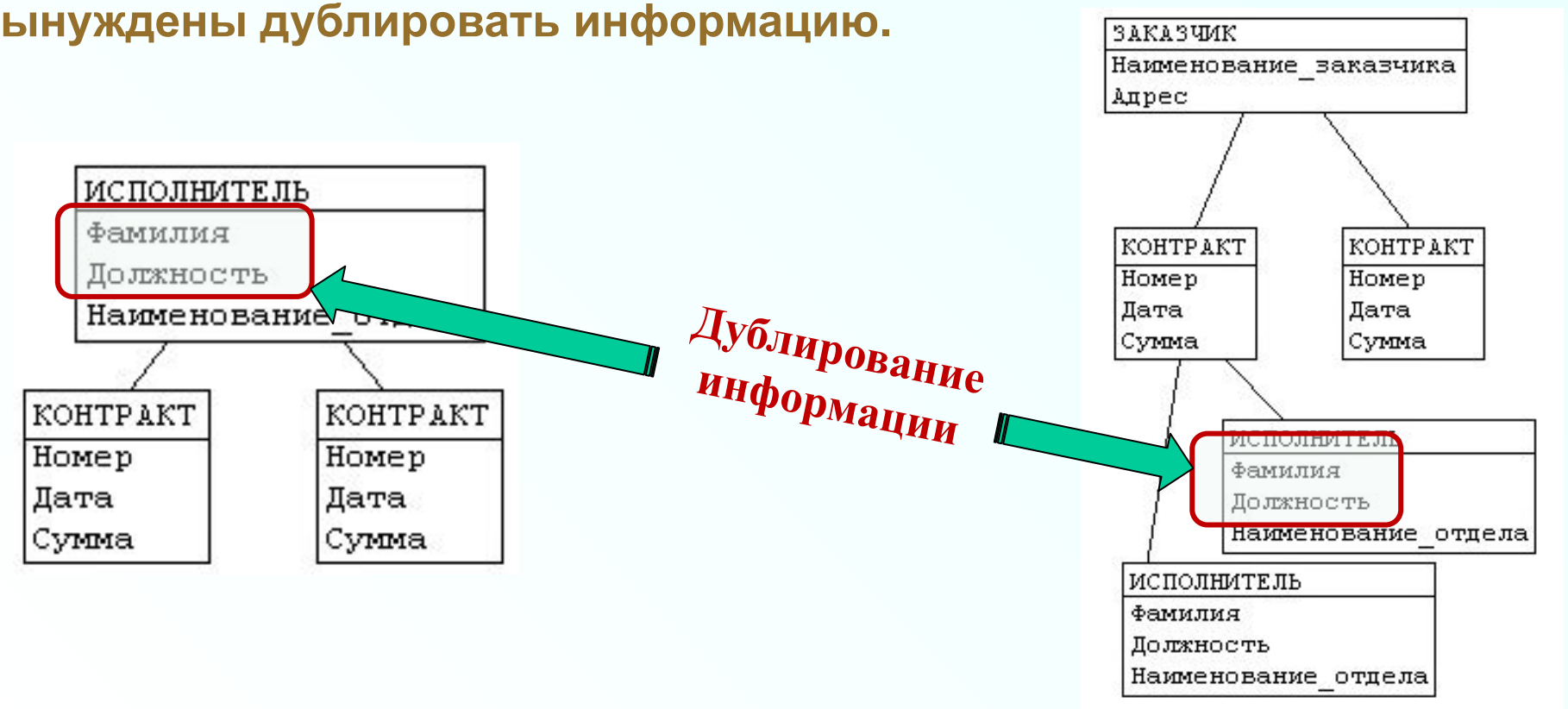
Частично дублируется информация между записями СОТРУДНИК и ИСПОЛНИТЕЛЬ (такие записи называют парными), причем в иерархической модели данных не предусмотрена поддержка соответствия между парными записями.



Пример

Иерархическая модель реализует отношение между исходной и дочерней записью по схеме 1:N, то есть одной родительской записи может соответствовать любое число дочерних.

Допустим теперь, что исполнитель может принимать участие более чем в одном контракте (т.е. возникает связь типа M:N). В этом случае в базу данных необходимо ввести еще одно групповое отношение, в котором ИСПОЛНИТЕЛЬ будет являться исходной записью, а КОНТРАКТ - дочерней Таким образом, мы опять вынуждены дублировать информацию.



Иерархические СУБД

Операции над данными, определенные в иерархической модели:

- ✔ **найти** указанный экземпляр типа дерева БД (например, отдел 310)
- ✔ **удалить** некоторую запись и все подчиненные ей записи.
- ✔ **перейти** от одного экземпляра типа дерева к другому
- ✔ **перейти** от экземпляра одного типа записи к экземпляру другого типа
- ✔ **перейти** от одной записи к другой в порядке обхода иерархии
- ✔ **вставить** новую запись в указанную позицию;
- ✔ **удалить** текущую запись
- ✔ **изменить** значение данных предварительно извлеченной записи. Ключевые данные не должны

Иерархические СУБД

Ограничения целостности

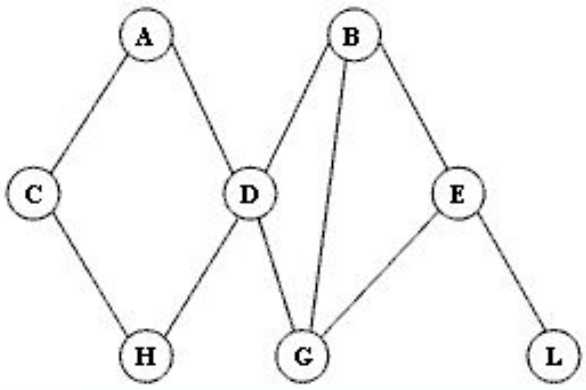
Поддерживается только целостность связей между владельцами и членами группового отношения (никакой потомок не может существовать без предка).

Не обеспечивается автоматическое поддержание соответствия парных записей, входящих в разные иерархии.

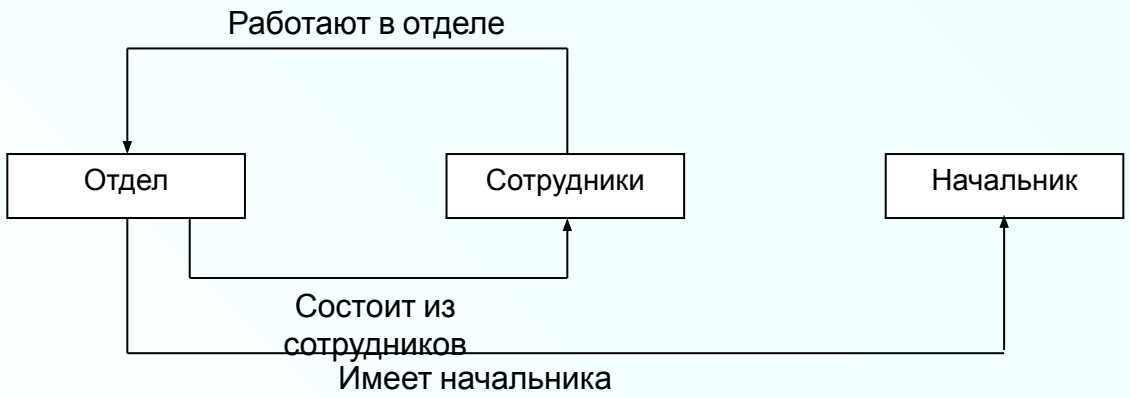
Сетевая модель данных

Сетевая модель данных позволяет отображать разнообразные взаимосвязи элементов данных в виде произвольного графа, обобщая тем самым иерархическую модель данных.

Сетевая БД состоит из набора записей и набора соответствующих связей. На формирование связи особых ограничений не накладывается. Если в иерархических структурах запись-потомок могла иметь только одну запись-предка, то в сетевой модели данных запись-потомок может иметь произвольное число записей-предков.



Представление связей в сетевой модели



Пример схемы простейшей сетевой БД

СМД. Реализации. Достоинства и недостатки

Наиболее распространенной и стандартизованной из реализаций СМД является модель CODASYL. В соответствии с ней описание схемы БД осуществляется на языке COBOL, а манипулирование данными – с помощью включающего языка программирования высокого уровня.
Примером сетевой СУБД является система **Integrated Database Management System (IDMS).**

Достоинства СМД:

- ✔ СМД является наиболее полной с точки зрения реализации различных типов связей и ограничений целостности.

СМД. Реализации. Достоинства и недостатки

Недостатки СМД:

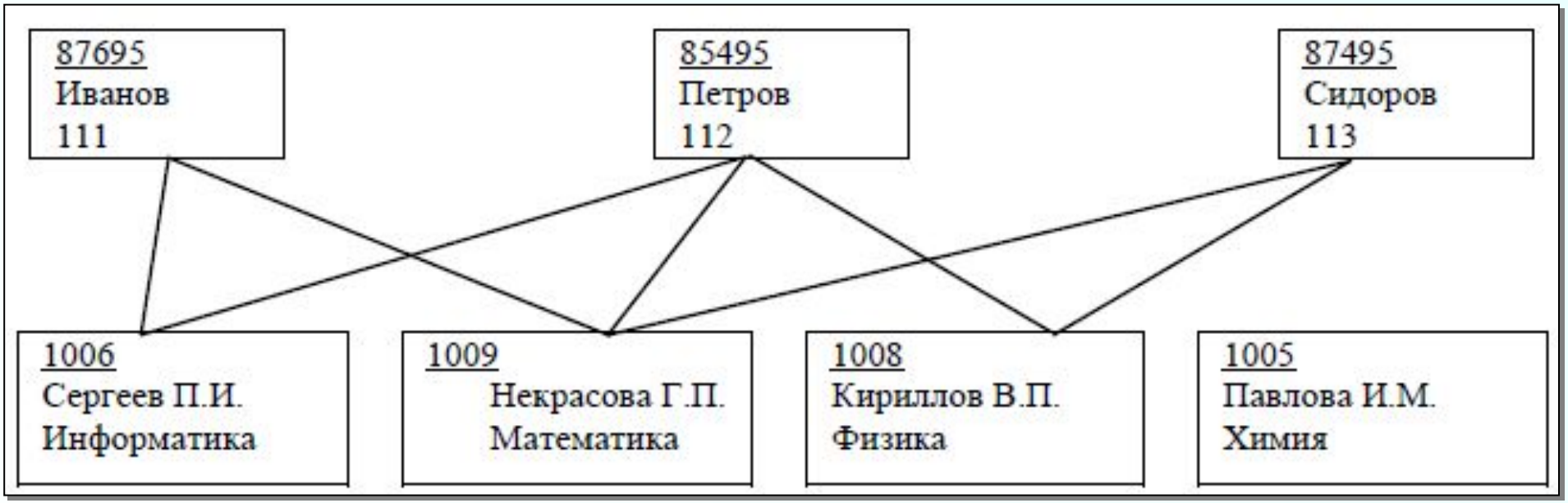
- ✔ является достаточно сложной для проектирования и поддержки.
- ✔ в СМДО не обеспечивается физическая независимость данных, т.к. наборы организованы с помощью физических ссылок.
- ✔ в СМД не обеспечивается независимость данных от программ.

Из-за указанных недостатков эта модель не получила широкого распространения.
Наиболее известной сетевой СУБД является **CA IDMS (Integrated Database Management System).**

Примерный набор операций СМД:

- ✔ **Найти** конкретную запись в наборе однотипных записей (инженера Сидорова)
- ✔ **Перейти** от предка к первому потомку по некоторой связи (к первому сотруднику отдела 310)
- ✔ **Перейти** к следующему потомку в некоторой связи (от Сидорова к Иванову)
- ✔ **Перейти** от потомка к предку по некоторой связи (найти отдел Сидорова)
- ✔ **Создать** новую запись
- ✔ **Удалить** запись
- ✔ **Модифицировать** запись
- ✔ **Включить** в связь
- ✔ **Исключить** из связи

Пример сетевой модели данных



Постреляционная модель

Классическая **реляционная модель** предполагает неделимость данных, хранящихся в полях записей таблиц. Это означает, что информация в таблице представляется в первой нормальной форме. Существует ряд случаев, когда это ограничение мешает эффективной реализации приложений.

Постреляционная модель данных представляет собой расширенную реляционную модель, **снимающую ограничение неделимости данных, хранящихся в записях таблиц. Постреляционная модель данных допускает многозначные поля – поля, значения которых состоят из подзначений.** Набор значений многозначных полей считается самостоятельной таблицей, встроенной в основную таблицу.

Структуры данных реляционной и постреляционной моделей

Реляционная модель данных

НАКЛАДНЫЕ

Номер накладной	Номер покупателя
0373	8723
8374	8232
7364	8723

ТОВАРЫ В НАКЛАДНОЙ

Номер накладной	Название товара	Количество товара
0373	Сыр	3
0373	Рыба	2
8374	Лимонад	1
8374	Сок	6
8374	Печенье	2
7364	Йогурт	1

Таблица НАКЛАДНЫЕ связана с таблицей ТОВАРЫ В НАКЛАДНОЙ по полю *Номер накладной*.

Постреляционная модель данных

Номер накладной	Номер покупателя	Название товара	Количество товара
0373	8723	Сыр	3
		Рыба	2
8374	8232	Лимонад	1
		Сок	6
		Печенье	2
7364	8723	Йогурт	1

Примеры запросов выбора данных из всех полей базы данных

Реляционная модель данных

При обработке данных в **реляционной модели** требуется выполнять операцию соединения данных из двух таблиц.

```
SELECT
НАКЛАДНЫЕ.Номер накладной,
Номер покупателя, Название товара,
Количество товара
FROM
НАКЛАДНЫЕ,
ТОВАРЫ В НАКЛАДНОЙ
WHERE
НАКЛАДНЫЕ.Номер накладной =
ТОВАРЫ В НАКЛАДНОЙ. Номер
накладной;
```

Постреляционная модель данных

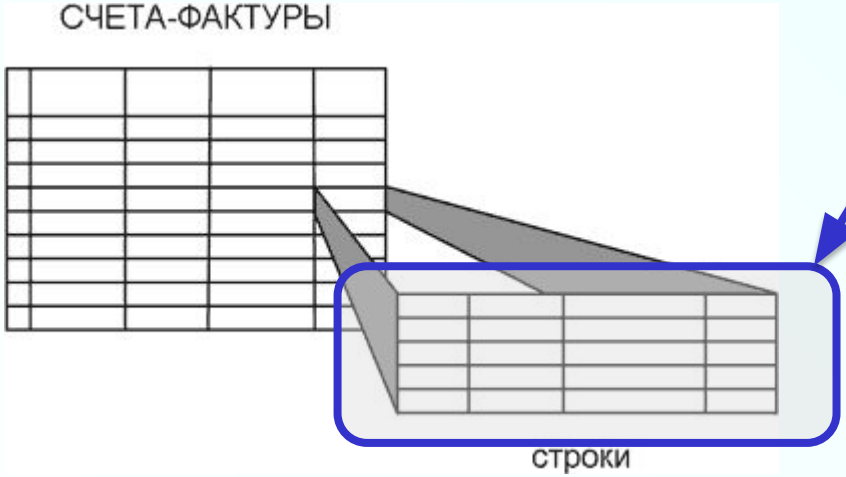
При обработке данных в **постреляционной модели** не требуется выполнять операцию соединения данных из двух таблиц.

```
SELECT
Номер накладной,
Номер покупателя,
Название товара, Количество товара
FROM
НАКЛАДНЫЕ;
```

Особенности постреляционной модели данных

Номер накладной	Номер покупателя	Название товара	Количество товара
0373	8723	Сыр	3
		Рыба	2
8374	8232	Лимонад	1
		Сок	6
		Печенье	2
7364	8723	Йогурт	1

✔ обеспечивает возможность вложенности полей



✔ поддерживает ассоциированные многозначные поля (ассоциации)

✔ На длину полей и количество полей в записях таблицы не накладывается требование постоянства.

✔ Допускает хранение в таблицах ненормализованных данных.

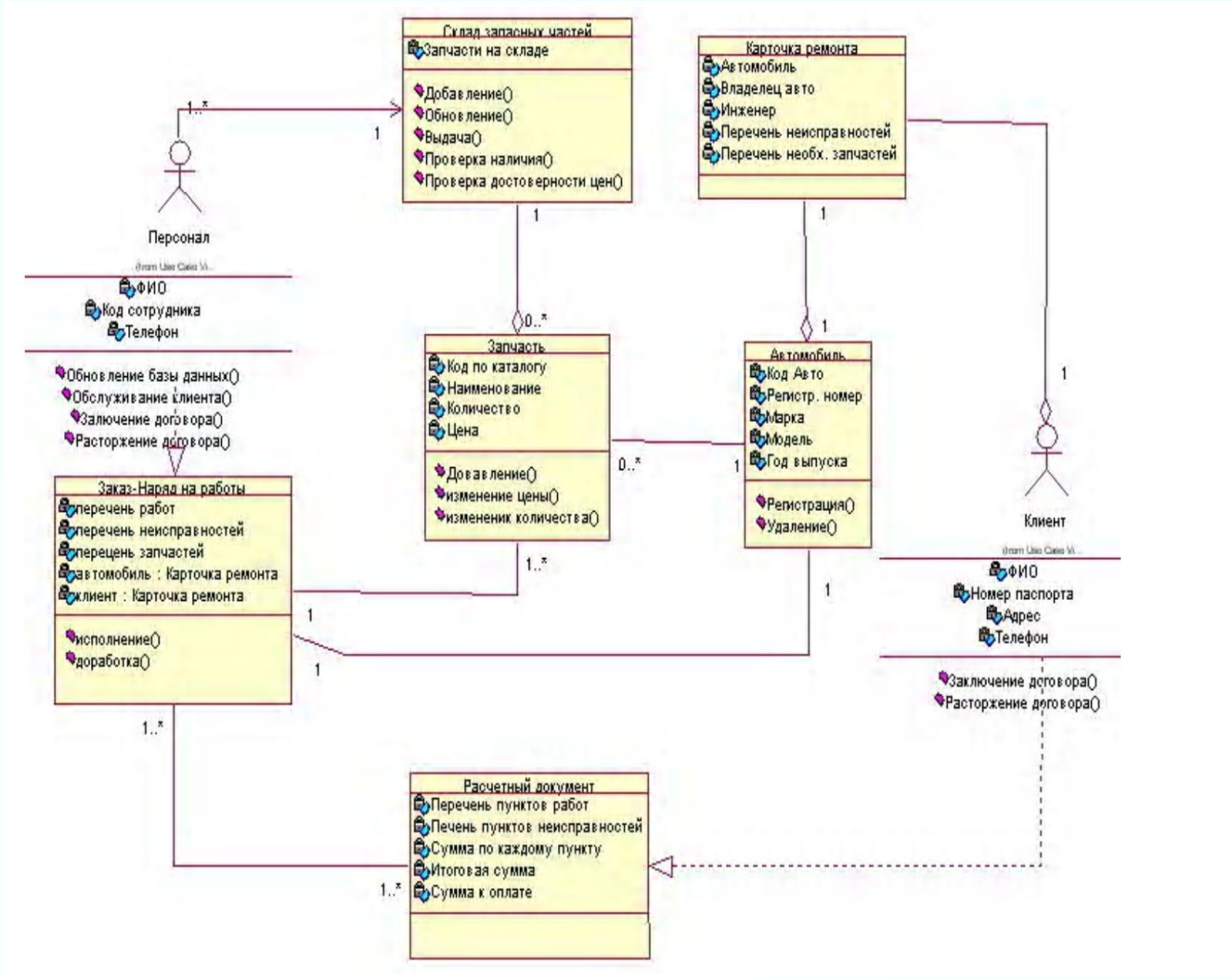
Достоинства и недостатки постреляционной модели

- ✔ **Достоинством постреляционной модели** является возможность представления совокупности связанных реляционных таблиц одной постреля¬ционной таблицей. Это обеспечивает высокую наглядность представления информации и повышение эффективности ее обработки.
- ✔ **Недостатком постреляционной модели** является сложность решения проблемы обеспечения целостности и непротиворечивости хранимых данных.

Постреляционная модель данных поддерживается СУБД uniVers, Bubba и Dasdb.

Объектно-ориентированная модель данных (ООМД)

Моделирование данных в ООМД базируется на понятии **объекта**. ООМД обычно применяется в сложных предметных областях, для моделирования которых не хватает функциональности реляционной модели (например, для САПР, издательских систем и т.п.).



Объектно-ориентированная модель данных (ООМД)

Моделирование данных в ООМД базируется на понятии **объекта**. ООМД обычно применяется в сложных предметных областях, для моделирования которых не хватает функциональности реляционной модели (например, для САПР, издательских систем и т.п.).

При создании объектно-ориентированных СУБД (ООСУБД) используются разные методы, а именно:

- ✔ встраивание в объектно-ориентированный язык средств, предназначенных для работы с БД;
- ✔ расширение существующего языка работы с базами данных объектно-ориентированными функциями;
- ✔ создание объектно-ориентированных библиотек функций для работы с БД;
- ✔ создание нового языка и новой объектно-ориентированной модели данных.

Достоинства и недостатки ООМД

К **достоинствам** ООМД можно отнести **широкие возможности моделирования предметной области, выразительный язык запросов и высокую производительность**. Т.к. каждый объект в ООМД имеет уникальный идентификатор (OID – object identifier), обращение по OID происходит существенно быстрее, чем поиск в реляционной таблице.

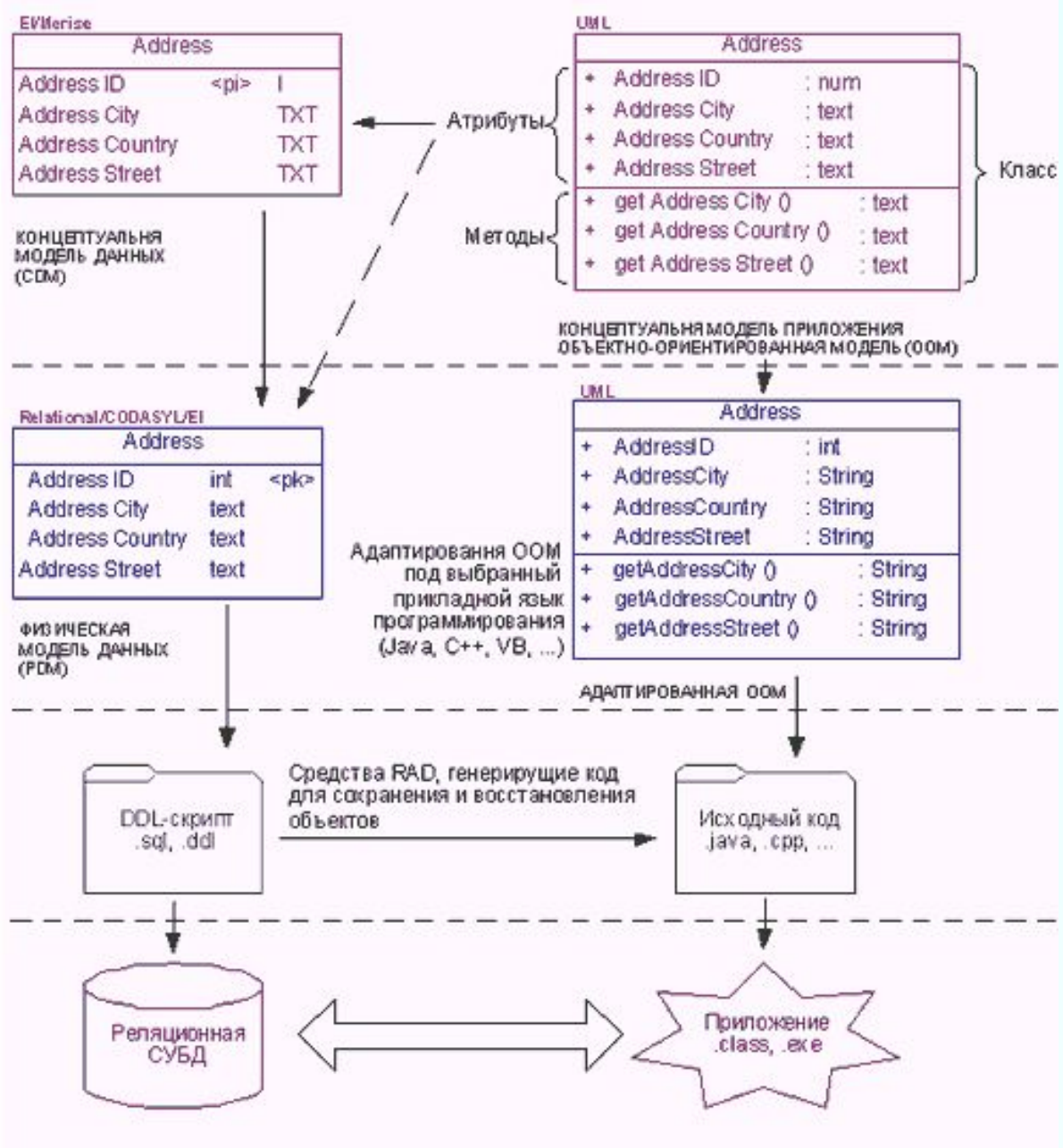
Среди **недостатков** ООМД следует отметить **отсутствие общепринятой модели, недостаток опыта создания и эксплуатации ООБД, сложность использования и недостаточность средств защиты данных**.

В 2000 г. рабочая группа **ODMG** (Object Database Management Group), образованная фирмами-производителями ООСУБД, выпустила стандарт (ODMG 3.0) для ООСУБД, в котором описана объектная модель, язык определения запросов, язык объектных запросов и связующие языки C++, Smalltalk и Java. Стандарты ODMG не являются официальными.

К числу ООСУБД СУБД относятся системы **O2, ORION, GemStone** и **Iris**.

Объектно-реляционная модель данных

Объектно-реляционная СУБД (ОРСУБД) - реляционная система управления базами данных, использующая в своей работе заимствования и методы свойственные объектно-ориентированному подходу.



2. Реляционная модель данных

Реляционная модель данных

Реляционная модель данных (РМД) включает следующие компоненты:

- **Структурный аспект** — данные в базе данных представляют собой набор отношений.
- **Аспект целостности** — отношения (таблицы) отвечают определенным условиям целостности. РМД поддерживает декларативные ограничения целостности **уровня домена (типа данных), уровня отношения и уровня базы данных.**
- **Аспект обработки (манипулирования)** — РМД поддерживает операторы манипулирования отношениями (реляционная алгебра, реляционное исчисление).

Реляционные БД

Реляционная база данных — база данных, основанная на реляционной модели данных.

Реляционные СУБД – системы для работы с реляционными БД

Определение (неформальное)

Реляционная база данных (от англ. Relation – отношение) – набор таблиц, связанных между собой по значениям определенных столбцов.

Схема БД – графическое описание логической структуры БД.

При создании информационной системы совокупность отношений позволяет хранить данные об объектах предметной области и моделировать связи между ними.

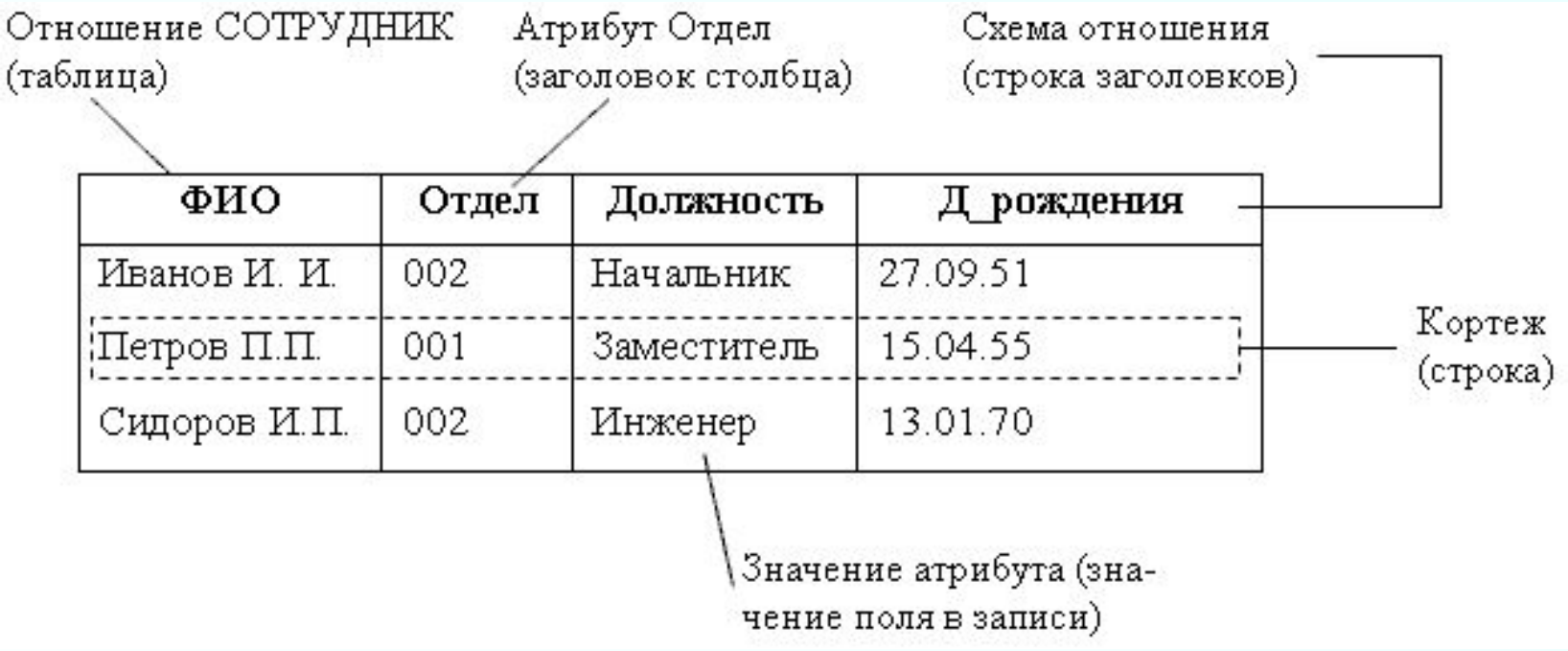
Реляционные БД

- **Отношение** – двухмерная таблица не содержащая строк-дубликатов
- **Сущность** есть объект любой природы, данные о котором хранятся в базе данных. Данные о сущности хранятся в отношении
- **Запись** – строка (ряд, запись, row, кортеж) таблицы
- **Отношение** – множество кортежей
- **Атрибут** (столбец). Атрибуты представляют собой свойства, характеризующие сущность. В структуре таблицы каждый атрибут именуется и ему соответствует заголовок некоторого столбца таблицы
- **Домен** – множество значений атрибута

Элементы реляционной модели

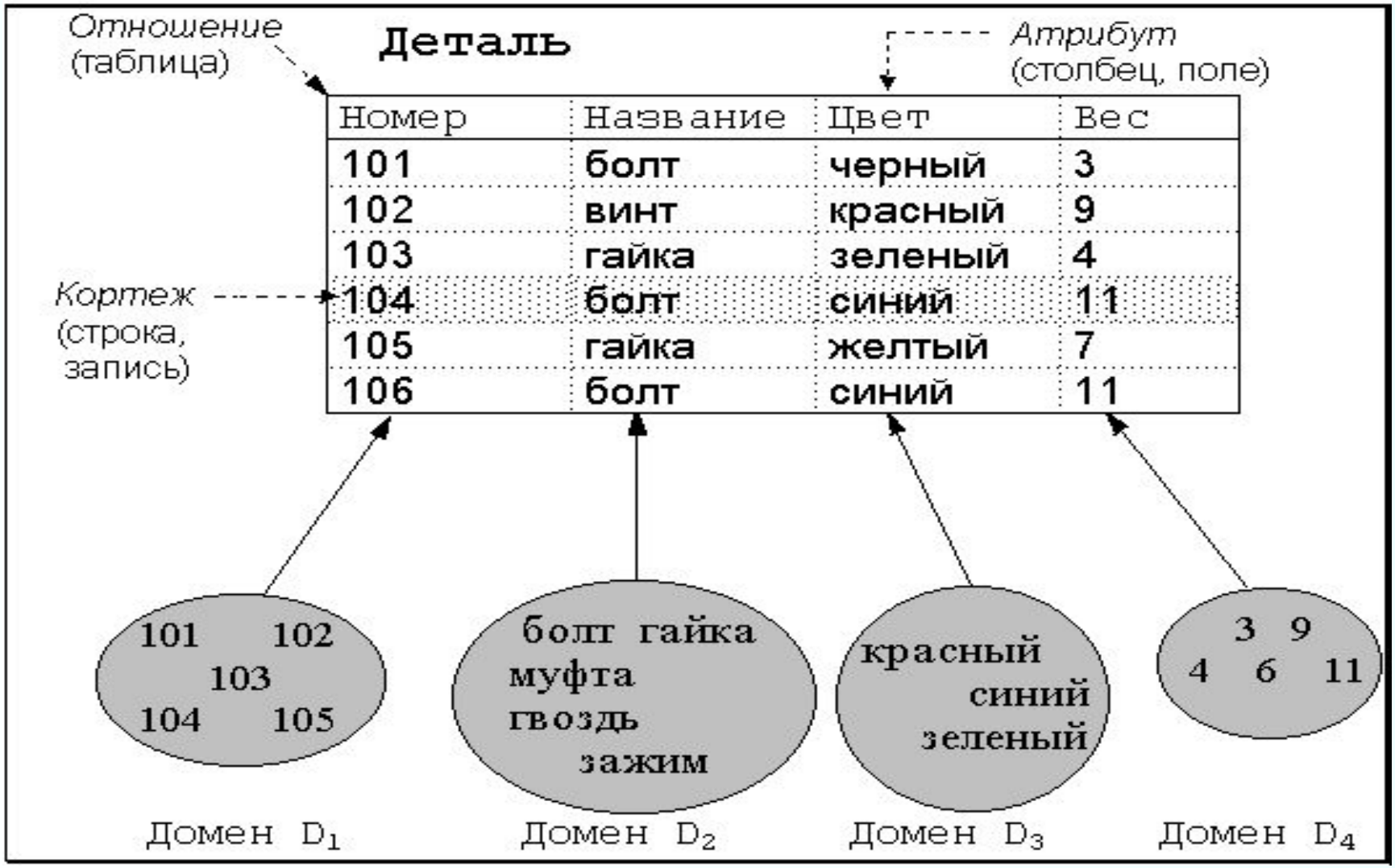
Элемент реляционной модели	Форма представления
Отношение	Таблица
Схема отношения	Строка заголовков столбцов таблицы
Кортеж (запись)	Строка таблицы
Сущность	Объект
Атрибут	Заголовок столбца таблицы
Домен	Множество допустимых значений атрибута
Значение атрибута (реквизит)	Значение поля в записи
Первичный ключ	Один или несколько атрибутов
Тип данных	Тип значений элементов таблицы

Пример реляционной модели данных



Представление отношения СОТРУДНИК

Пример реляционной модель данных



Представление отношения ДЕТАЛЬ

Свойства отношений

Отсутствие кортежей-дубликатов

Данное свойство следует из определения отношения как множества кортежей. В классической теории множеств по определению каждое множество состоит из различных элементов.

Свойства отношений

Отсутствие упорядоченности кортежей

Свойство отсутствия упорядоченности кортежей отношения также является следствием определения отношения-экземпляра как множества кортежей. Отсутствие требования к поддержанию порядка на множестве кортежей отношения дает дополнительную гибкость СУБД при хранении баз данных во внешней памяти и при выполнении запросов к базе данных. Это не противоречит тому, что при формулировании запроса к БД, например, на языке SQL можно потребовать сортировки результирующей таблицы в соответствии со значениями некоторых столбцов. Такой результат, вообще говоря, не отношение, а некоторый упорядоченный список кортежей.

Свойства отношений

Атомарность значений атрибутов

Значения всех атрибутов являются атомарными. Это следует из определения домена как потенциального множества значений простого типа данных, т.е. среди значений домена не могут содержаться множества значений (отношения). Принято говорить, что в реляционных базах данных допускаются только нормализованные отношения или отношения, представленные в первой нормальной форме.

Понятие ключа

- **Ключ** – атрибут или совокупность атрибутов однозначно идентифицирующих строку отношения;
- Ключ, состоящий из одного атрибута, называется **простым**.
- Ключ, состоящий из нескольких атрибутов, называется **составным**.

Свойства ключа:

- ✔ Уникальность
- ✔ Неизбыточность
- ✔ Не может содержать пустых значений

Ключи обычно используют для достижения следующих целей:

- ✔ **исключения дублирования** значений в ключевых атрибутах (остальные атрибуты в расчет не принимаются)
- ✔ **упорядочения кортежей**
- ✔ **ускорения работы** с кортежами отношения;
- ✔ **организации связывания таблиц**.

Пусть в отношении R1 имеется не ключевой атрибут A, значения которого являются значениями ключевого атрибута B другого отношения R2. Тогда говорят, что

Понятие ключа

Каждое отношение обязательно имеет комбинацию атрибутов, которая может служить ключом. Ее существование гарантируется тем, что отношение – это множество, которое не содержит одинаковых элементов – кортежей.

Информационный объект → **Студент**

Ключевое поле

Номер	Фамилия	Имя	Отчество	Дата
16493	Сергеев	Петр	Михайлович	01.01.76
16593	Петрова	Анна	Владимировна	15.03.75
16693	Анохин	Андрей	Борисович	14.04.76

Свойства реляционной таблицы:

- ✔ Все **строки** таблицы должны быть **уникальны**, т. е. не может быть строк с одинаковыми первичными ключами.
- ✔ Имена **столбцов** таблицы должны быть различны, а **значения** их **простыми**, т. е. недопустима группа значений в одном столбце одной строки.
- ✔ Все **строки** одной таблицы **должны иметь одну структуру**, соответствующую именам и типам столбцов.
- ✔ **Порядок** размещения строк в таблице может быть **произвольным**.

Связывание таблиц

- При проектировании БД информацию обычно размещают в **нескольких таблицах**. Таблицы при этом связаны семантикой информации. В реляционных СУБД для указания связей таблиц производят операцию их связывания.
- Многие СУБД при связывании таблиц автоматически выполняют **контроль целостности** вводимых в базу **данных** в соответствии с установленными связями. В конечном итоге это повышает достоверность хранимой в БД информации.
- Кроме того, установление связи между таблицами облегчает **доступ к данным**. Связывание таблиц при выполнении таких операций, как поиск, просмотр, редактирование, выборка и подготовка отчетов, обычно обеспечивает возможность обращения к произвольным полям связанных записей. Это уменьшает количество явных обращений к таблицам данных и число манипуляций в каждой из них.

Основные виды связи таблиц

- При связывании двух таблиц выделяют **основную (родительскую) и дополнительную (подчиненную, дочернюю) таблицы**. Логическое связывание таблиц производится с помощью **ключа связи**.
- Ключ связи, по аналогии с обычным ключом таблицы, состоит из одного или нескольких полей, которые в данном случае называют **полями связи** (ПС).
- Суть связывания состоит в установлении соответствия полей связи основной и дополнительной таблиц.
- В зависимости от того, как определены поля связи основной и дополнительной таблиц (как соотносятся ключевые поля с полями связи), между двумя таблицами в общем случае могут устанавливаться следующие четыре основных вида связи: **один — один (1:1); один — много (1:М); много — один (М:1); много — много (М:М или М:N)**.

Связь

Связь - это логическая ассоциация, устанавливаемая между таблицами БД.

Связь определяет **количество записей данной таблицы, которые могут быть связаны с одной записью другой таблицы.**

Связи бывают следующих типов:

- **один к одному**
- **один ко многим**

- Пример 1:

  «Страны» - «Столицы»
- Пример 2:

  «Группа» - «Студент»
- Пример 3:

  «Сотрудники» - «Проекты»

Характеристика видов связей таблиц

Характеристика полей связи по видам	1:1	1:М	М:1	М:М
Поля связи основной таблицы	являются ключом	являются ключом	не являются ключом	не являются ключом
Поля связи дополнительной таблицы	являются ключом	не являются ключом	являются ключом	не являются ключом

Характеристика связей:

Связь вида 1:1

Связь вида 1:1 образуется в случае, когда все поля связи основной и дополнительной таблиц являются ключевыми. Поскольку значения в ключевых полях обеих таблиц не повторяются, обеспечивается взаимно-однозначное соответствие записей из этих таблиц. Сами таблицы, по сути, здесь становятся равноправными.

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.

Связь вида 1:М

Связь 1:М имеет место в случае, когда одной записи основной таблицы ответствует несколько записей вспомогательной таблицы.

Характеристика связей:

Связь вида M:1

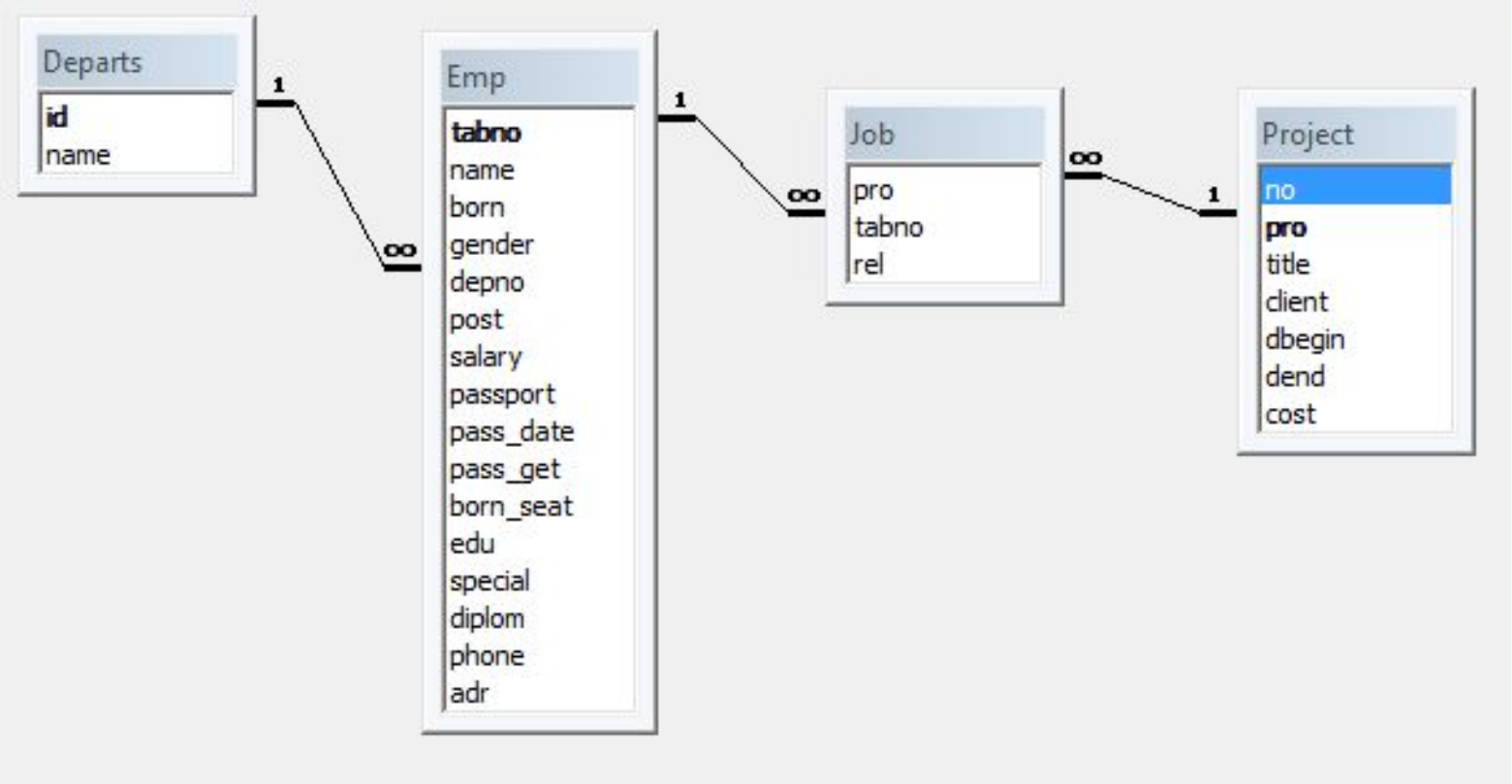
Связь M:1, по сути, является «зеркальным отображением» связи 1:M.

Связь вида M:M

Самый общий вид связи M:M возникает в случаях, когда нескольким записям основной таблицы соответствует несколько записей дополнительной таблицы.

На практике в связь обычно вовлекается сразу несколько таблиц. При этом одна из таблиц может иметь различного рода связи с несколькими таблицами. В случаях, когда связанные таблицы, в свою очередь, имеют связи с другими таблицами, образуется иерархия или дерево связей.

Пример БД: «Проектная организация»



Departs – отделы, **Project – проекты,**

Emp – сотрудники, **Job – участие в проектах.**

Связь «один-ко-многим»: Отделы - Сотрудники

Таблица «Сотрудники» (Emp)

Табельный номер	*ФИО сотрудника*	*Отдел*
023	Волкова Елена Павловна	2
113	Белов Сергей Юрьевич	1
101	Рогов Сергей Михайлович	2
056	Панина Анна Алексеевна	1
…	…	…
098	Фролов Юрий Вадимович	9

Таблица «Отделы» (Departs)

Номер отдела	*Название отдела*
1	Информационный отдел
2	Администрация
3	Отдел кадров
…	…
9	Проектный отдел

«Номер отдела» – первичный ключ в таблице «Отделы»

«Отдел» – внешний ключ в таблице «Сотрудники» к таблице «Отделы»

Связь «многие-со-многими»: Сотрудники- Проекты

- Связи "many-to-many". Иногда бывает необходимо связывать таблицы БД таким образом, что с обоих концов связи могут присутствовать несколько записей. Например, сотрудники консалтинговой компании участвуют в проектах. При этом один сотрудник может участвовать в нескольких проектах и в одном проекте могут участвовать несколько сотрудников. Для этого вводится разновидность связи **"многие-со-многими"**.
- Оформляются через «развязочные таблицы», например: «участие в проектах» (таблица из двух полей: код сотрудника (FK), код проекта (FK)).

Связь «многие-со-многими»: Сотрудники- Проекты

Таблица «Сотрудники»

ФИО	*Номер*
Волкова Е.П.	023
Белов С.Ю.	113
Рогов С.М.	101
Панина А.А.	056
Фролов Ю.В.	098
...	...

Таблица «Участие»

Участник	*Роль*	*Проект*
113	исполнитель	23/Н
101	руководитель	18-К
056	исполнитель	18-К
101	консультант	09/Р
098	руководитель	23/Н
...	...	...

Таблица «Проекты»

Шифр	*Название проекта*
23/Н	АИС "Налог"-2
18-К	ИПС "Жители"
09/Р	ГИС "Город"

В таблице «Участие»:

«Участник» – внешний ключ к таблице «Сотрудники»

«Проект» – внешний ключ к таблице «Проекты»

Контроль целостности связей

- Из перечисленных видов связи наиболее широко используется связь вида **1:M.** Связь вида **1:1** можно считать частным случаем связи **1:M**, когда одной записи главной таблицы соответствует одна запись вспомогательной таблицы. Вид связи **M:M** характеризуется как слабый вид связи или даже как отсутствие связи. Рассмотрим связь вида **1:M**.
- При образовании связи вида **1:M** одна запись главной таблицы (главная, родительская запись) оказывается связанной с несколькими записями дополнительной (дополнительные, подчиненные записи).

Контроль целостности связей

- Контроль целостности связей обычно означает анализ содержимого двух таблиц на соблюдение следующих правил:

✔ каждой записи основной таблицы соответствует ноль или более записей дополнительной таблицы;

✔ в дополнительной таблице нет записей, которые не имеют родительских записей в основной таблице;

✔ каждая запись дополнительной таблицы имеет только одну родительскую запись основной таблицы.

Контроль целостности связей

Опишем действие контроля целостности при манипулировании данными в таблицах. Рассмотрим три основные операции над данными двух таблиц:

- ✔ ввод новых записей
- ✔ модификацию записей
- ✔ удаление записей

Контроль целостности связей

При вводе новых записей данные сначала вводятся в основную таблицу, а потом — в дополнительную. Очередность ввода может быть установлена на уровне целых таблиц или отдельных записей (случай одновременного ввода в несколько открытых таблиц). В процессе заполнения основной таблицы контроль значений полей связи ведется как контроль обычного ключа (на совпадение со значениями тех же полей других записей). Заполнение полей связи дополнительной таблицы контролируется на предмет совпадения со значениями полей связи основой таблицы. Если вновь вводимое значение в поле связи дополнительной таблицы не совпадет ни с одним соответствующим значением в записях основной таблицы, то ввод такого значения должен блокироваться.

Контроль целостности связей

Модификация записей. Изменение содержимого полей связанных записей не относящихся к полям связи происходит обычным образом. Рассмотрим механизм изменения полей связи.

При редактировании полей связи дополнительной таблицы очевидным требованием является то, чтобы новое значение поля связи совпадало с соответствующим значением какой-либо записи основной таблицы. Т. е. дополнительная запись может сменить родителя, но остаться без него не должна.

Редактирование поля связи основной таблицы разумно подчинить одному из следующих правил:

- редактировать записи, у которых нет подчиненных записей. Если есть подчиненные записи, то блокировать модификацию полей связи;
- изменения в полях связи основной записи мгновенно передавать во все поля связи всех записей дополнительной таблицы (каскадное обновление).

Контроль целостности связей

В операциях удаления записей связанных таблиц большую свободу имеют записи дополнительной таблицы. Удаление их должно происходить практически бесконтрольно. Удаление записей основной таблицы логично подчинить одному из следующих правил:

- удалять можно запись, которая не имеет подчиненных записей;
- запретить (блокировать) удаление записи при наличии подчиненных записей, либо удалять ее вместе со всеми подчиненными записями (каскадное удаление).

3. Информационно-логическая модель предметной области

Проектирование базы данных

Проектирование базы данных состоит в построении комплекса взаимосвязанных объектов данных. Проект базы данных надо начинать с анализа предметной области и выявления требований к ней отдельных пользователей.

Разработчик базы данных сначала создает обобщенное описание создаваемой базы данных. Это описание, выполненное с использованием средств, понятных всем людям, работающих над проектированием базы данных, называют концептуальной (инфологической) моделью данных. В концептуальной модели средствами структур данных в интегрированном виде отражают состав и структуру данных, а также информационные потребности приложений.

Такая модель полностью независима от физических параметров среды хранения данных.

Моделирование БД

Предметная область – это часть реального мира, данные о которой мы хотим отразить в базе данных. Например, в качестве предметной области можно выбрать бухгалтерию какого-либо предприятия, отдел кадров, магазин и т.д. Предметная область бесконечна и содержит как существенно важные понятия и данные, так и малозначащие или вообще не значащие данные. Так, если в качестве предметной области выбрать учет товаров на складе, то понятия "накладная" и "счет-фактура" являются существенно важными понятиями, а то, что сотрудница, принимающая накладные, имеет двоих детей – это для учета товаров неважно. Однако, с точки зрения отдела кадров данные о наличии детей являются существенно важными. Таким образом, важность данных зависит от выбора предметной области.

Модель предметной области

- Имеется большое количество **методик описания предметной области**. Из наиболее известных можно назвать методику структурного анализа **SADT** и основанную на нем **IDEF0**, **диаграммы потоков данных Гейна-Сарсона**, **методику объектно-ориентированного анализа UML** и др. Модель предметной области описывает скорее процессы, происходящие в предметной области и данные, используемые этими процессами. От того, насколько правильно смоделирована предметная область, зависит успех дальнейшей разработки приложений.

Концептуальная (информационно-логическая) модель предметной области

Информационно-логическая модель (ИЛМ) является моделью данных, отображающих предметную область в виде совокупности информационных объектов и структурных связей между ними.

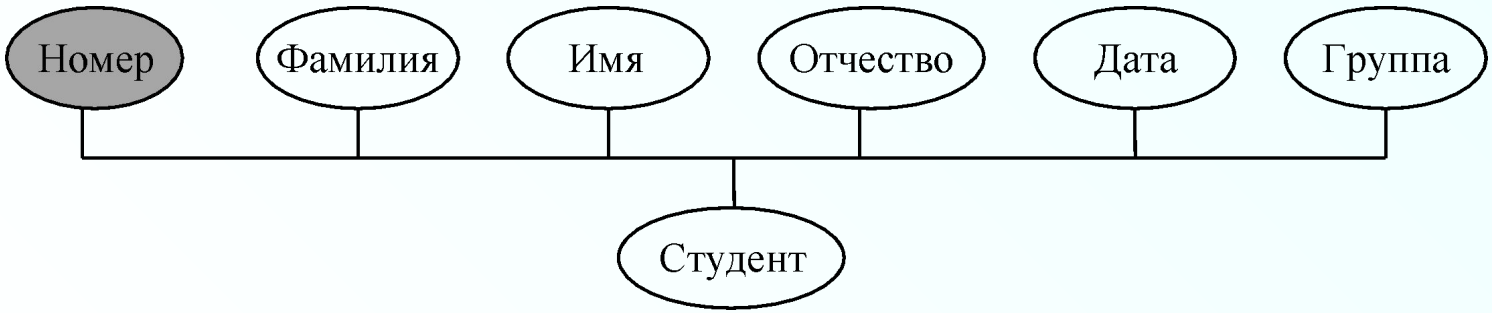
Объектом называется любой элемент некоторой системы. Принято называть отдельный предмет экземпляром объекта, а различные множества объектов, образованные по какому-либо принципу – типами объектов.

Свойством объекта называется некоторая величина, характеризующая состояние объекта в любой момент времени. Отдельный экземпляр объекта может быть точно описан, если указаны значения его свойств. Два экземпляра будут различными, если они отличаются по значению хотя бы одного свойства.

Информационно-логическая модель предметной области

Представление объекта или процесса в БД сводится к указанию его свойств. Информационным отображением свойств служат атрибуты, следовательно экземпляр объекта может быть представлен в базе данных как набор имени атрибута и его значения (имена атрибутов соответствуют названиям свойств объекта).

Информационный объект предметной области – это информационное отображение некоторой сущности, то есть реального объекта, явления или процесса, информация о котором должна быть представлена в виде базы данных или информационной системы.



Пример представления информационного объекта Студент в виде графа

Информационно-логическая модель предметной области

Информационный объект имеет множество реализаций – **экземпляров**, каждый из которых представлен совокупностью конкретных значений свойств (**реквизитов**) и идентифицируется значением ключа (простого – один реквизит или составного – несколько реквизитов). Остальные реквизиты информационного объекта являются описательными.

Атрибуты	Номер	Фамилия	Имя	Отчество	Дата	Группа
Экземпляры инф. объекта Студент	16493	Сергеев	Петр	Михайлович	01.01.76	ДКБ-111
	16593	Петрова	Анна	Владимировна	15.03.75	ДКБ-112
	16693	Анохин	Андрей	Борисович	14.04.76	ДКБ-114

Пример структуры и экземпляров информационного объекта

При проектировании БД между информационными объектами устанавливаются **структурные связи**, отражающие отношения между реальными объектами, процессами или явлениями.

CASE-средства построения концептуальных моделей

- Одной из наиболее популярных форм представления концептуальных моделей данных является модель **«сущность-связь»** (ER-модель Entity (сущность) и Relation (связь)).
- Модель была предложена Ченом в 1976 г. Кроме модели Чена известны другие модели:
  - семантическая модель данных Хаммера и Маклеода;
  - функциональная модель Шипмана и др.
- Моделирование предметной области базируется на использовании графических диаграмм, включающих небольшое число разнородных компонентов. В связи с наглядностью представления концептуальных моделей, ER-диаграммы получили широкое распространение в **CASE-средствах** (Computer-Aided System Engineering), предназначенных для автоматизированного проектирования реляционных баз данных.

ER-диаграммы

- **ER-диаграммы** в наглядной форме представляют связи между сущностями. Наиболее популярными являются CASE-системы ERwin, Design/IDEF, Power Designer, в которых диаграммы создаются в соответствии со стандартом **IDEF1X**.
- Основные понятия ER-диаграммы:
  - ✔ сущность;
  - ✔ атрибут;
  - ✔ связь.
- Построение ER-диаграммы предполагает определение сущностей предметной области, атрибутов этих сущностей и установление связей между ними.

Спасибо за внимание!

CODASYL

CODASYL (англ. COnference on DAta SYstems Language — Конференция по языкам систем обработки данных) — организация (название произносится «кодасил»), принимавшая активное участие в эволюции информационных технологий в 60-80-е годы XX века. Основана в 1959 для разработки стандартного языка программирования для коммерческих систем [1]. Этот язык получил название COBOL. Помимо разработки языка в рамках консорциума была сформирована группа Data Base Task Group, которой было поручено разработать универсальный язык для баз данных, встроенный в COBOL. В 1969 году группа опубликовала спецификацию языка для сетевой модели данных, которая получила название "CODASYL Data Model".

В настоящее время конференция CODASYL расформирована, архив был передан Институту имени Чарльза Бэббиджа.