

# Проектирование БД

---

ЛЕКЦИЯ №2

# Вопросы

---

1. Общая схема создания базы данных
2. Концептуальное проектирование (модель сущность-атрибут-связь)
3. Логическое проектирование
4. Физические модели данных
5. CASE-технологии

# 1. Процесс создания базы данных

---

- Определение требований.
- Выбор СУБД и разработка структуры БД с учетом особенностей СУБД.
- Реализация.
- Тестирование, разработка документации, сопровождение.

# Определение требований – моделирование

---

- Изучения понятий и описаний.
- Описание информационных объектов или понятий предметной области и связей между ними.
- Описание ограничений целостности.

# Концептуальные модели

---

Семантическая модель (или концептуальная модель, инфологическая модель) — модель предметной области, предназначенная для представления семантики предметной области на самом высоком уровне абстракции.

# Уровни абстракции базы данных:

---

- внешняя схема;
- концептуальный уровень;
- внутренняя модель.

# Трехуровневая архитектура

---

Трехуровневая архитектура (инфологический, даталогический и физический уровни) позволяет обеспечить **независимость хранимых данных** от использующих их программ.

# Основные преимущества ER-моделей:

---

- наглядность;
- возможность проектировать базы данных с большим количеством объектов и атрибутов;
- реализация во многих системах автоматизированного проектирования баз данных.



## 2. Назначение диаграммы «сущность–связь»:

---


- проектирование баз данных;
- идентификация понятий предметной области и связей между ними;
- графическое представление логической структуры базы данных.

# Основные элементы диаграммы «сущность–связь»:

---

- сущности (прямоугольники);
- атрибуты (овалы);
- связи (ромбы).



Обозначение	Значение
Имя сущности	Независимая сущность
Имя сущности	Зависимая сущность
Имя атрибута	Атрибут
<u>Имя атрибута</u>	Ключевой атрибут
Имя связи	Связь
	Прямая линия указывает на связь между сущностями либо соединяет сущность и атрибут. Линия со стрелкой указывает направленную связь.

# Модель «сущность - СВЯЗЬ».

## Сущность

---

- Сущность — это абстрактный объект определенного вида.
- Уникальное имя.
- Набор экземпляров сущностей образует множество.
- Множества не содержат дубликатов.

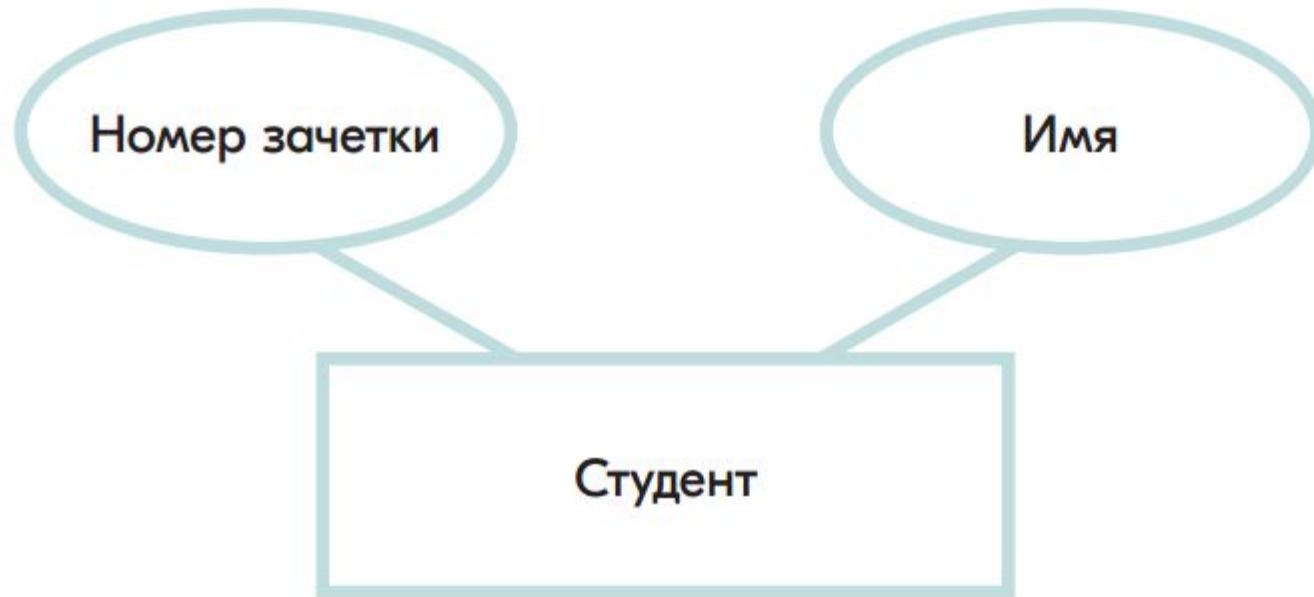
# Атрибуты

---

- Атрибуты — это отдельные характеристики объекта.
- Каждый атрибут имеет уникальное имя.
- Каждый атрибут имеет свой тип данных.
- Сущность может обладать любым количеством атрибутов.
- Значение атрибута атомарно.
- Сущность и ее атрибуты на диаграмме соединяются ненаправленными дугами.
- Значения атрибутов выбираются из соответствующего множества значений.

# Пример: представление сущности

---



# Ключи

---

- Ключ — это один или несколько атрибутов объекта, по которым объект можно однозначно идентифицировать.
- Если нет естественного ключа, придумывают искусственный ключ — «суррогатный».

# СВЯЗИ

---

- Связь — ассоциирование двух или более сущностей.
- Требование к организации базы данных — обеспечение возможности отыскания одних сущностей по значениям других.



# Как увидеть появление связи?

---

- Если хочется атрибутом какой-то сущности объявить другую сущность или список сущностей.
- Если хочется записать в одну сущность идентификатор другой.



Вам хочется сделать связь.

# Свойства связей

---

- Связи могут иметь собственные атрибуты.
- Подобные связи объединяются в множества.
- Связи не могут существовать без связываемых сущностей.
- Ключ связи состоит из ключей связываемых сущностей и, возможно, выделенных атрибутов связи.

# Характеристики связей:

---

- размерность;
- мощность;
- модальность.

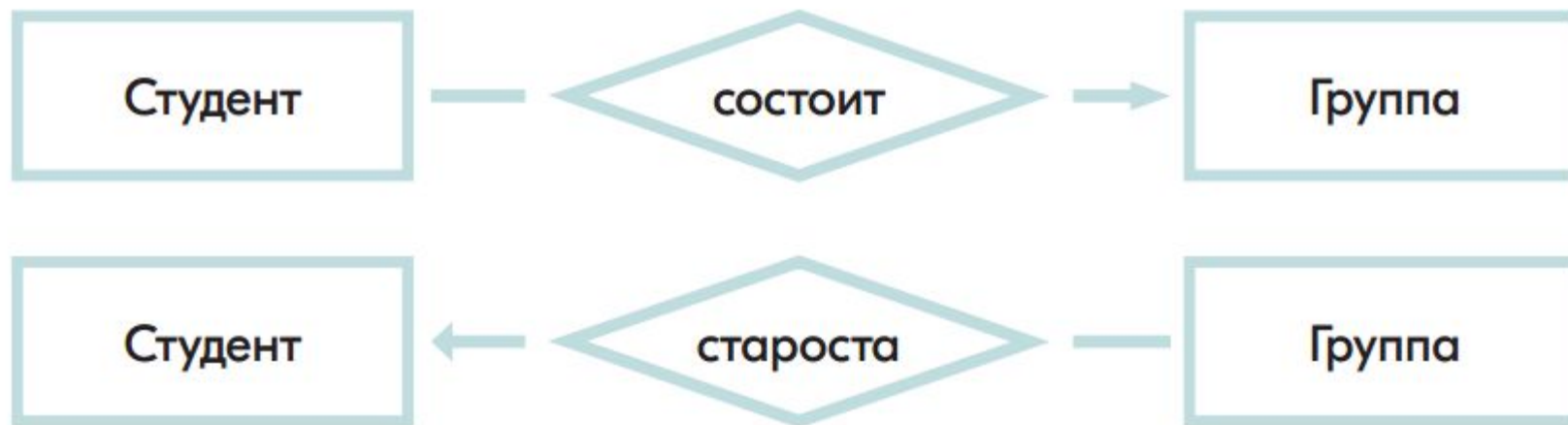
# Классификация связей: размерность

---

- Бинарные.
- Тернарные.
- N-арные.
- Рекурсивные.

# Пример: бинарная связь

---



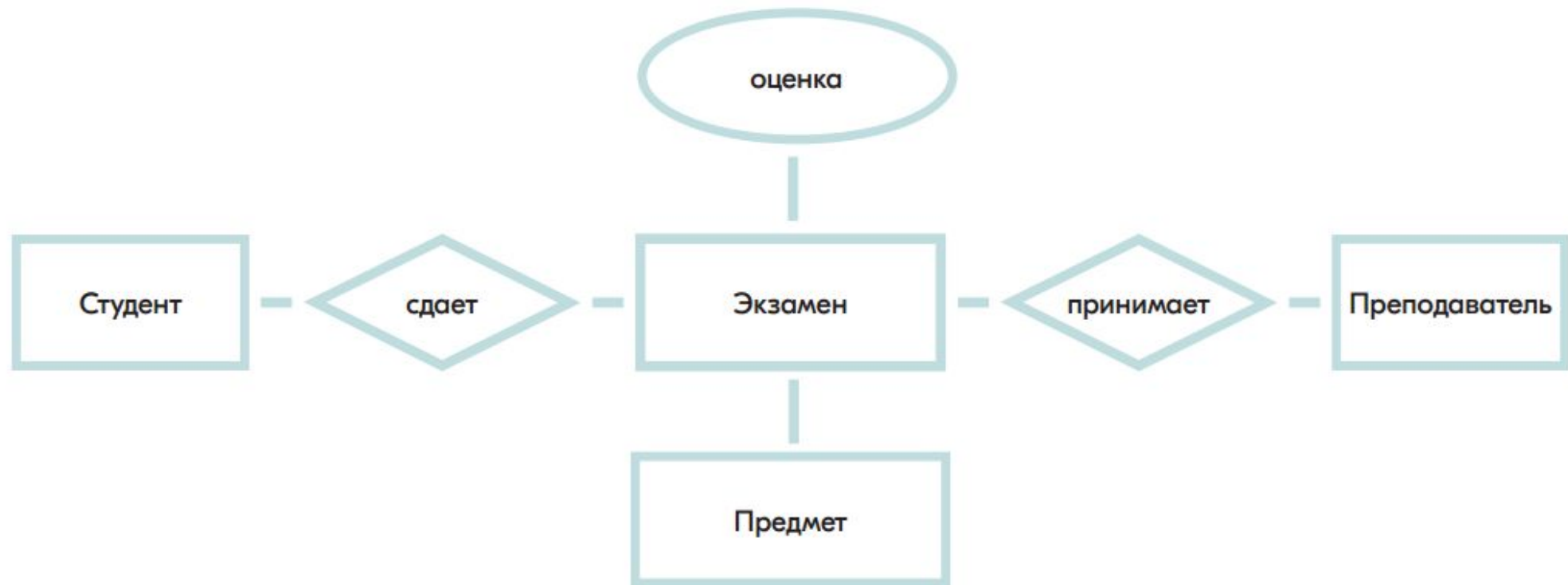
# Пример: тернарная связь

---



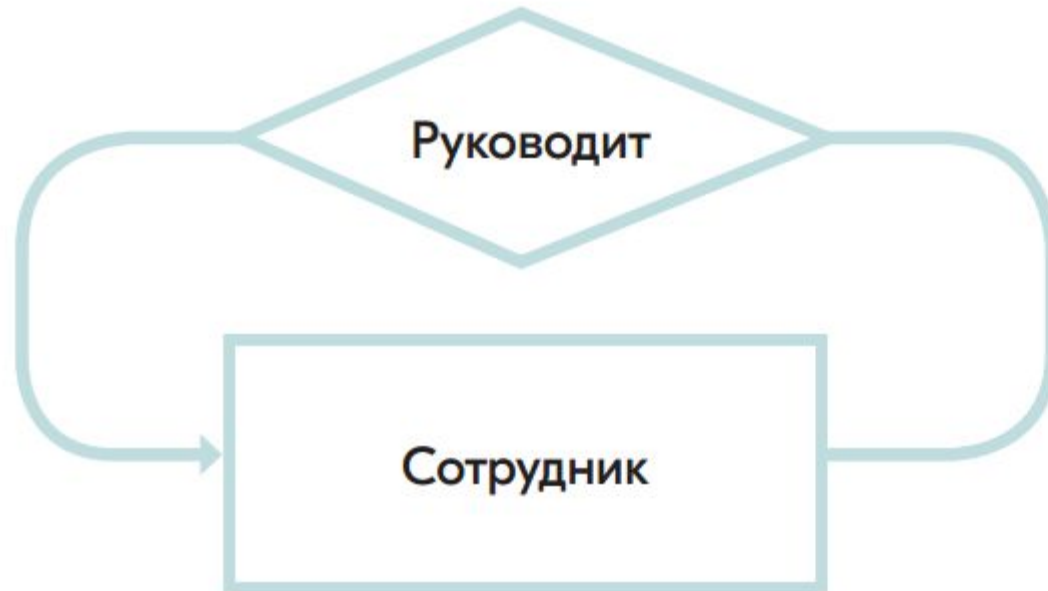
# Пример: преобразование тернарной связи в бинарную

---



# Пример: рекурсивная связь

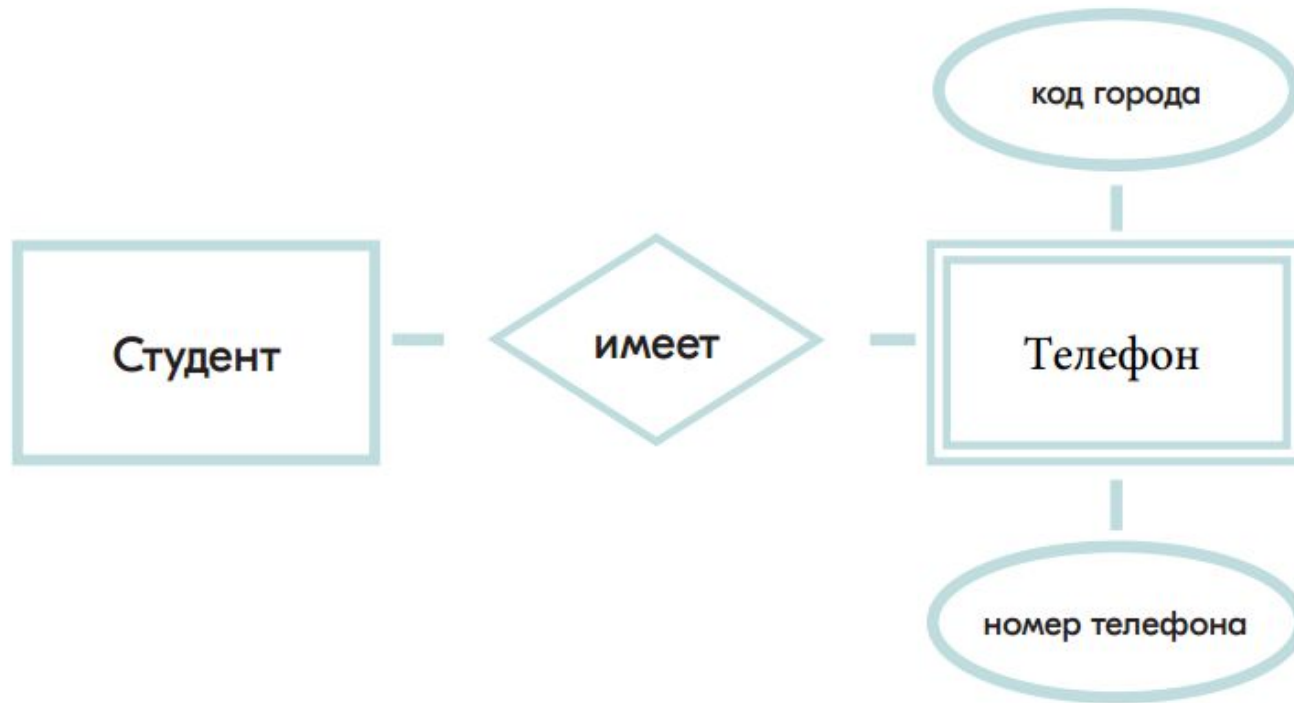
---





# Слабые (зависимые) сущности и СВЯЗИ

---



# Мощность бинарной связи

---

Делятся на три вида в зависимости от количества участвующих в них сущностей:

- «один-к-одному»      1:1;
- «один-ко-многим»      1:N;
- «многие-ко-многим»    M:N.

# СВЯЗИ «ОДИН-К-ОДНОМУ»

---

Каждому экземпляру первой сущности может соответствовать ровно один экземпляр другой сущности и наоборот.



# СВЯЗИ «ОДИН-КО-МНОГИМ»

---

Каждому экземпляру первой сущности может соответствовать несколько экземпляров другой сущности, но каждому экземпляру второй сущности соответствует не более одного экземпляра первой сущности.



# СВЯЗИ «МНОГИЕ-КО-МНОГИМ»

---

Каждому экземпляру первой сущности может соответствовать несколько экземпляров другой сущности и наоборот.



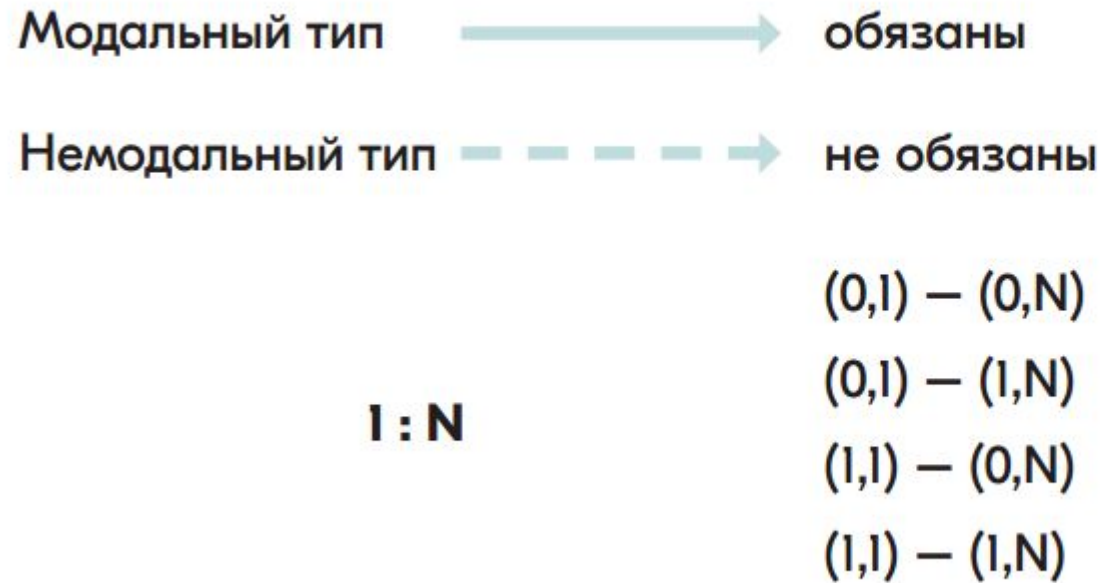
# Модальность связей

---

- **«Может»**: экземпляр одной сущности может быть связан с одним или несколькими экземплярами другой сущности, а может быть и не связан ни с одним экземпляром.
- **«Должен»**: экземпляр одной сущности обязан быть связанным не менее чем с одним экземпляром другой сущности.

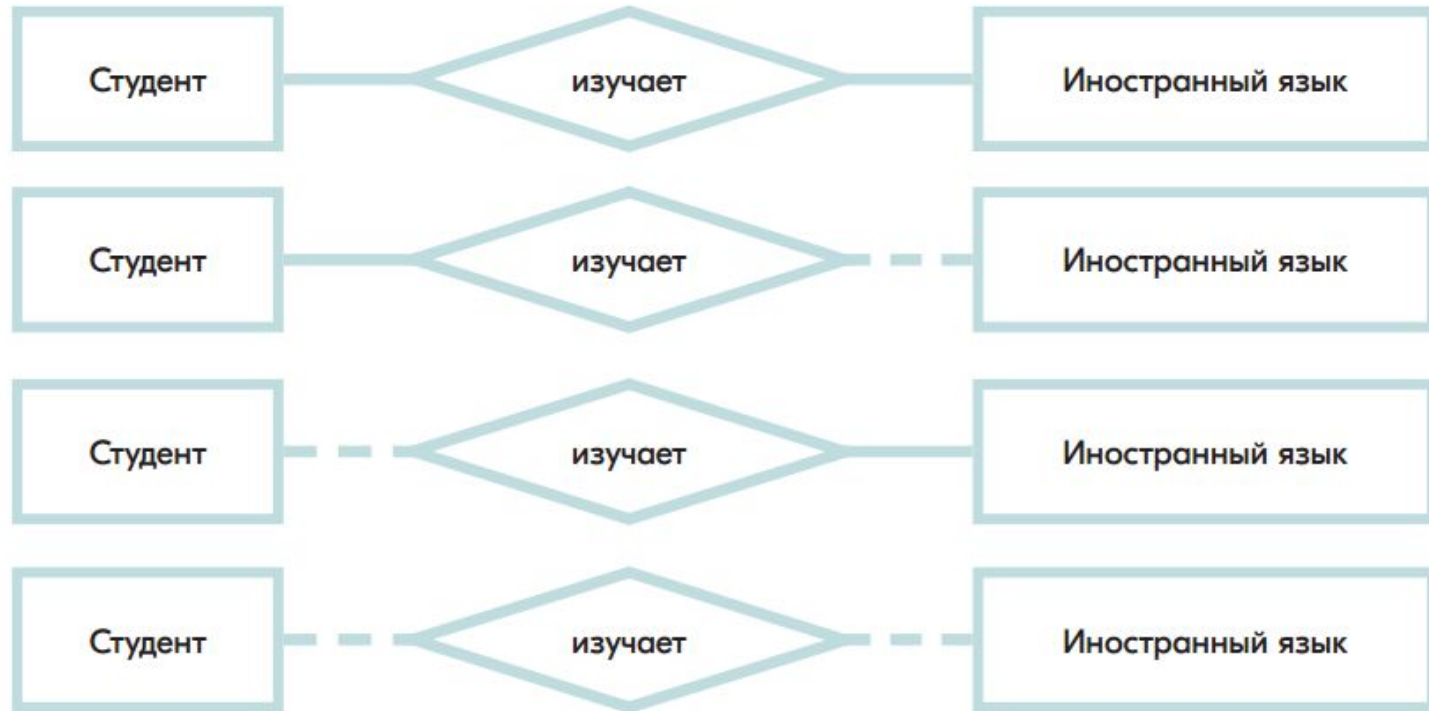
# Бинарные связи – модальность (обязательность связи)

---



# Пример: варианты типов связей

---





# Шаги при создании ERD:

---

- определить сущности;
- определить атрибуты сущностей;
- определить первичные ключи;
- определить связи между сущностями;
- определить кардинальность связей;
- нарисовать ERD;
- проверить ERD.

# Пример ER-диаграммы:



## 3. Логическое проектирование

---

*Логическая модель БД* называется модель логического уровня, построенная в рамках конкретной СУБД, в среде которой проектируется БД.

Описание логической структуры БД в терминологии данной СУБД называется *схемой БД*.

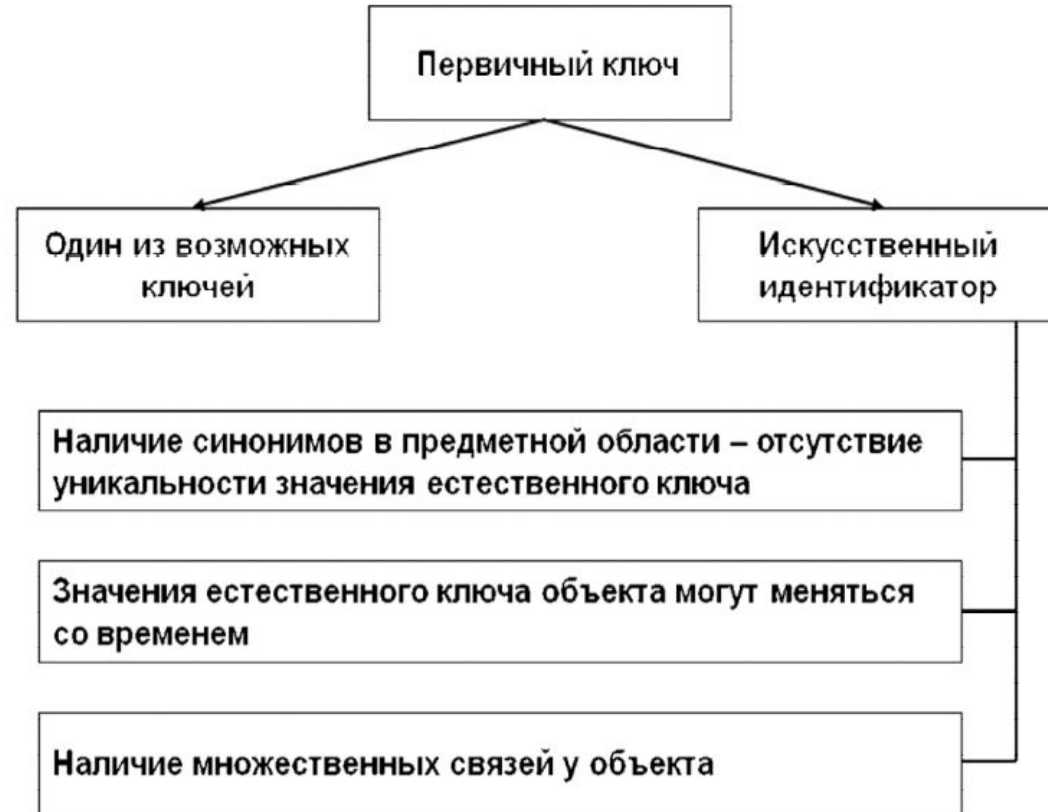
# Алгоритм перехода к реляционной модели

---



# Алгоритм выбора первичного ключа

---



# Отображение свойств простых объектов

---



# Отображение свойств сложных объектов

---



# Определение свойств атрибутов отношения

---





# Критерии анализа БД

---

- Адекватность схемы БД
  - Полнота схемы БД
  - Сложность структуры
  - Адаптируемость
  - Дублирование данных
- 
- Объем необходимой памяти

## 4. Физические модели данных

---

**Физическая модель данных** - способ размещения данных на устройствах внешней памяти и способ доступа к этим данным

# Способы хранения данных.

---



# Способы хранения данных

---



# Организация файловой структуры

---



# Способы поиска записей в файле

---



# Метод хэширования

---

1. Выбор хэш-функции
2. Выбор стратегии разрешений коллизий.

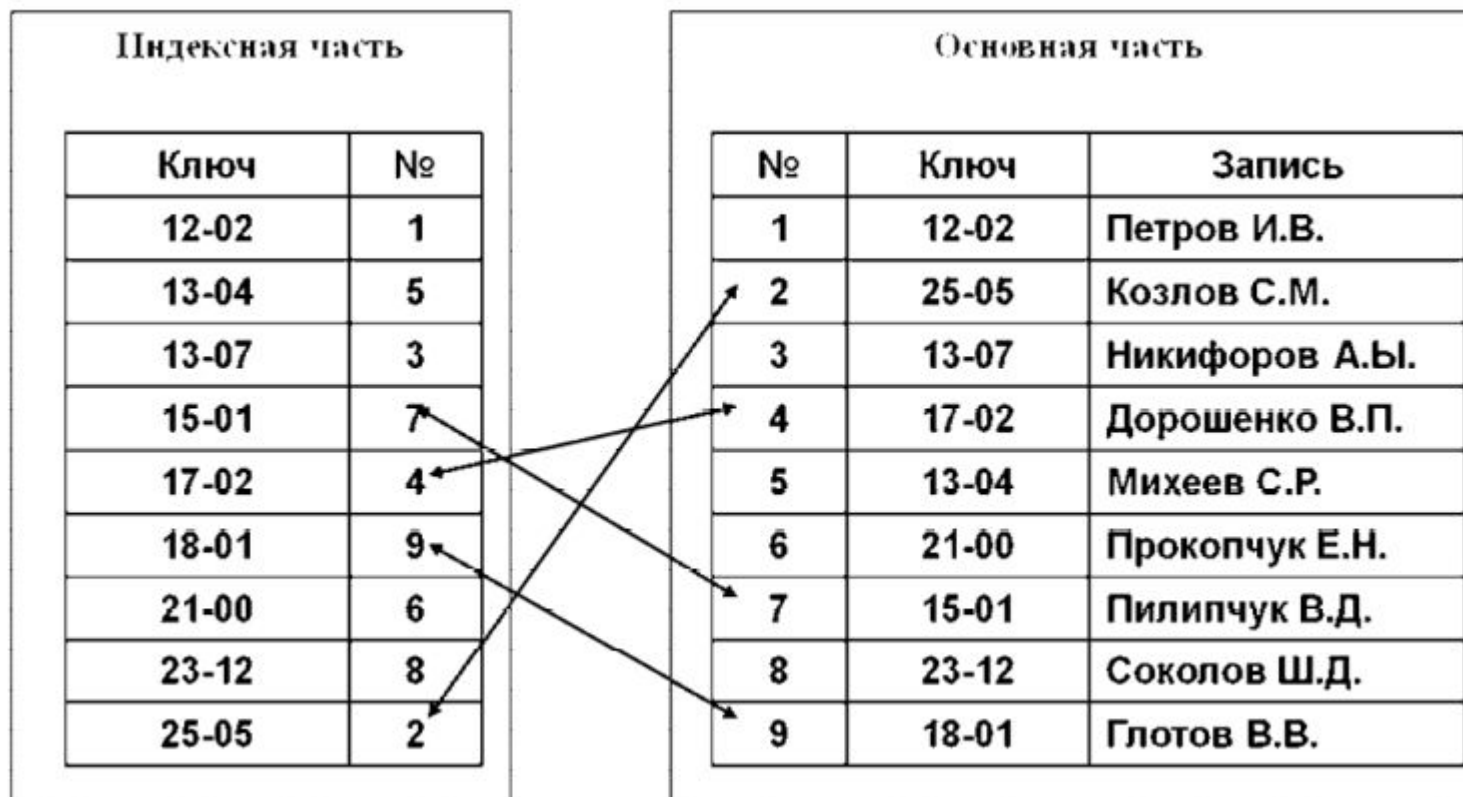
# Типы индексных файлов

---

- *файлы с плотным индексом (индексно-прямые файлы)*
- *файлы с неплотным индексом (индексно-последовательные файлы)*
- *B-деревья.*



# Структура файла с плотным индексом

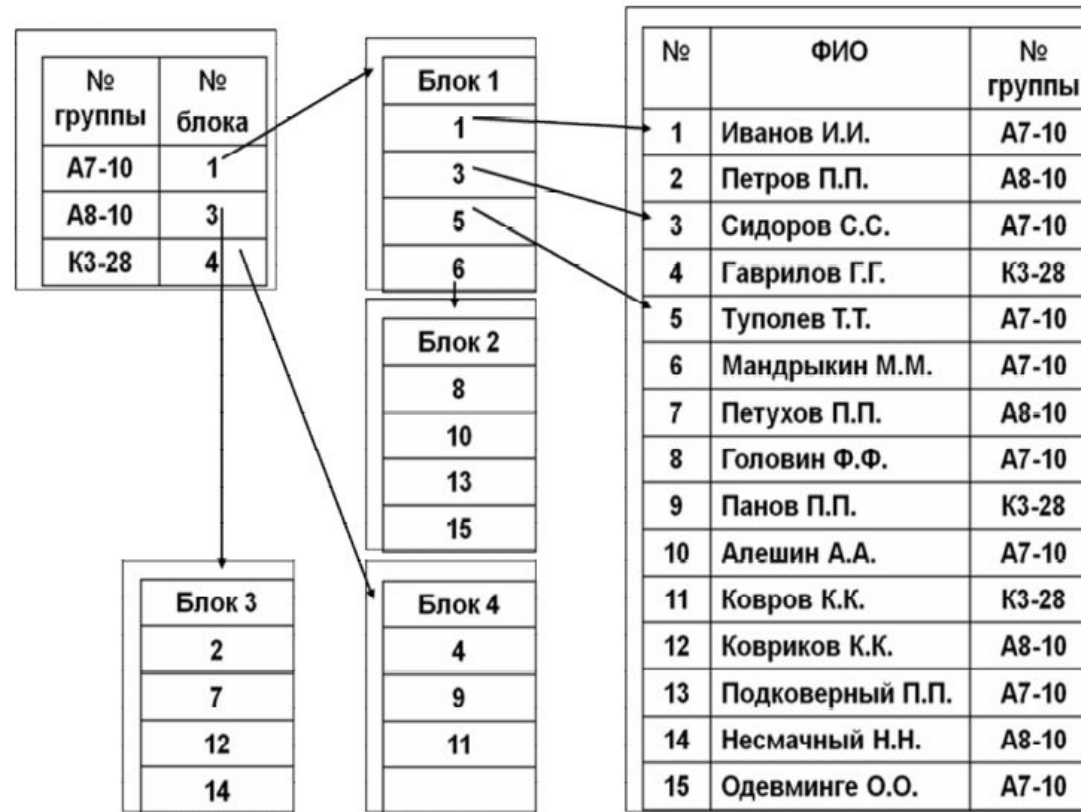


# Структура файла с неплотным индексом





# Структура инвестированного списка



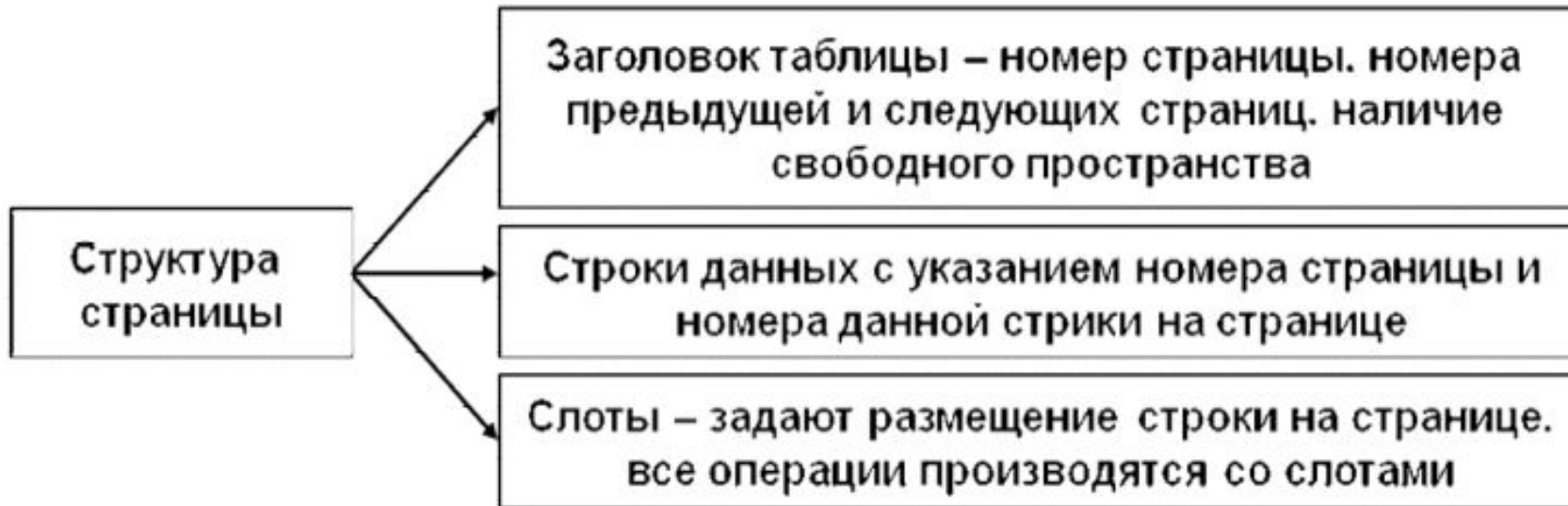
# Страничная организация данных

---



# Страничная организация данных

---



## 5. CASE-технологии

---

Понятие CASE (**Computer Aided System Engineering** - автоматизированное проектирование информационных систем) включает в себя совокупность регламентно-методических материалов, автоматизированных методов и инструментальных средств разработки, поддерживающих все этапы Жизненного Цикла Системы (ЖЦС, Business System Life Cycle) начиная от первоначального формирования технических требований и спецификаций до получения и сопровождения готового программного продукта.

# Состав CASE-систем

---





# Компоненты CASE-систем

---



# Задачи CASE-систем

---



# Характеристики CASE-систем

---



# Виды CASE-систем

---



# Методологии CASE-систем

---



# Этапы жизненного цикла систем

---



# Модели жизненного цикла систем



# Возможности CASE-систем

---





# Используемые CASE-технологии

---

1. Системы Design/IDEF и VPwin используются как средства анализа и предназначены для исследования всевозможных моделей предметной области.
2. Основными средствами проектирования баз данных являются такие технологии, как ERwin, S-Designor, DataBase Designor.
3. Технологии PRO-IV, Vantage Team Builder применяются для решения задач создания различных проектных спецификаций в качестве средств анализа и проектирования.
4. Технологии Oracle, JAM, Delphi, C++ Duilder, Uniface, Power Builder, SQL являются основными средствами для разработки приложений. Эти средства могут быть предназначены как для решения задач на одном или нескольких этапах ЖЦС (такие как Erwin, SDesignor), так и являться интегрированными, поддерживающими весь ЖЦС (Vantage Team Builder, Oracle).