Лекция 3

Строгие определения

- **База данных (Коннолли, Бегг)** - совместно используемыый набор логически связанных данных и их описаниый, предназначенный для удволетворения информационных потребностей организаций
- **База данных (Дейт)** - набор постоянно хранимых данных, используемых прикладными системами предприятий
- **Базы данных (Хомоненко)** - совокупность специальбным образом организованных данных, хранимых в памяти вычислительной системы и отображающих состояние объектов и их взаимосвязей в рассматриваемой предметной области
- **СУБД (Коннолли, Бегг)** - программное обеспечение, с помощью которого пользователи могут определять, создавать и поддерживать базу данных, а также осуществлять к ней контролируемый доступ
- **СУДБ (Хомоненко)** - комлекс языковых и программных средств, предназначенный для создания, ведения и совместного использования базы данных многими пользователями

Логический уовень

- **Иерархическая модель данных**
  Идея в иерархической модели данных состоит в том, чтобы хранить данные в деревьях. В 60-70 годах вместо таблицы все данные представлялись в виде деревьев, например, на предприятиях продукт изготавливался из компонентов, которые делались из деталей, которые создавались из заготовок. Чаще всего деталей было относительно немного, поэтому иерархическая модель подходила лучше всего. В итоге появляются:
  > Поле данных - атомарная (неделимая) единица данных
  > Сегмент данных - совокупность полей данных
  Сегменты, состоящие из полей
тип - изделие	название - А1	...
тип - узел	название У1	изделие - pointer	...
тип - деталь	название - гайка М1	материал - сталь К	узел - pointer	...
тип - деталь	название - болт Т1	материал - сталь	узел - pointer to	...
тип - деталь	название - гайка М1	материал - сталь	узел - pointer
- **Сетевая модель данных** - добавили возможность ссылаться на несколько потомков. У любого сегмента может быть более одного родительского сегмента

    - **Хорошо**
        - Убрали дубликаты
        - Убрали проблему целостности (надо менять только у одного экземпляра (так как нет дубликатов))
    - **Плохо**
        - Циклы - контролировать их наличие невозможно (экспоненциальная ассимптотика)
- **Реляционная модель данных** (Эндер Кодд)
    - Каждую сущность отображаем на таблицу
        - **Таблица - Изделия** (Название, ...)
        - **Таблица - Узлы** (Название, Изделие, ...)
        - **Таблица - Детали** (Название, Изделие, Материал, ...)
    - Храним **названия отдельно**, а данные как **совокупность кортежей**
    - Как мы храним связь в такой модели?
        - **А никак нахуй**
        - Храним ID, сопоставление которому находим в другой абстрактной структуре, то есть по сути это интерпретация, а не связь. Отсюда и возникают проблемы - можем соединить два одинаковых ID, но которые будут иметь абсолютно разную семантику
        - (с)**Инженеры IBM'а поступили как правильные и хорошие разрабы - они переложили проблему на конечного пользователя**
- **Постреляционная модель**
    - **Разрешены составные поля** - поле перестает восприниматься как неделимое
- **Многомерная модель**
    - Таблица многомерная - многомерные таблицы
        - Проблемы:
            - Кубы могут быть весьма разреженными
            - Как все это записывать
        - Как делают реально:
            - Хранят реляционную модель, а раз в какое-то время перестраивает многомерную модель (менеджерам не нужны точности до секунд в своих таблицах)
- **ООП** -
    - Вспомним, что объект - совокупность полей. Тогда появились ORM-модели (Object Relation Model) - создается таблица с атрибутами класса, объект записывается как строка в таблицу.

- В этом случае целостность данных гарантируется тем, что данные изменяются только методами объекта, Но если изменить данные в табличке, в файле, получим нецелостный объект
- Можно хранить в объекте вместо данных ключ, который ссылается на данные в бинарном файле, а в методах объекта данные сразу же записываются в файл