

Этапы проектирования БД

Процесс проектирования БД состоит из **трех основных этапов**:

- 1) концептуальное проектирование;
- 2) логическое проектирование;
- 3) физическое проектирование.

1. Концептуальное проектирование БД – это процесс создания **высокоуровневой семантической модели** для данных, которые присутствуют в определенной предметной области.

Важно, что такая модель **никак не зависит** от любых аспектов ее **физической реализации**:

- тип СУБД и вычислительной платформы;
- набор прикладных программ;
- средства программирования приложений и др.

Концептуальная модель данных создается на основе информации, записанной в **требованиях пользователей**.

Дополнительно проводится специальный опрос (анкетирование) пользователей.

В процессе разработки этой модели она

2. Логическое проектирование БД – это процесс создания информационной модели на основе выбранной модели **структурной организации данных** при их хранении и обработке. Это делается **без выбора конкретной СУБД** и без учета остальных аспектов физической реализации БД.

Логическая модель данных создается путем **преобразования концептуальной модели** с учетом особенностей выбранной модели организации данных.

Важную роль логическая модель данных играет и при эксплуатации (сопровождении) уже готовой БД.

Эта модель, если ее постоянно поддерживать в актуальном состоянии, позволяет точно и наглядно представить любые изменения в структуре БД, а также оценить их влияние на прикладное ПО.

3. Физическое проектирование – это процесс принятия решений по реализации проекта разрабатываемой БД.

В случае реляционной БД это означает:

- ❖ выбор конкретной (целевой) СУБД;
- ❖ построение процедуры создания таблиц на жестком диске;
- ❖ определение методов доступа к данным, чтобы обеспечить высокую производительность СУБД:
 - выбор необходимой файловой структуры (т.е. типов файлов для хранения данных);
 - оценка целесообразности использования индексных файлов;

❖ планирование средств информации

Методология концептуального проектирования БД

1. Определение типов (классов) сущностей
2. Определение атрибутов для сущностей
3. Определение доменов для атрибутов
4. Определение потенциальных и первичных ключей
5. Определение типов связей между сущностями
6. Построение ER-диаграммы

При выборе первичного ключа среди нескольких потенциальных ключей наиболее важными являются следующие факторы:

- min набор атрибутов (наилучший вариант – простой ключ целого типа);
- значения min длины;
- высокая стабильность (т.е. min вероятность изменения значений);
- простота работы для пользователя.

Если нет возможности сделать удачный выбор первичного ключа среди собственных атрибутов сущности, то рекомендуется ввести вспомогательный **суррогатный**

Методология логического проектирования реляционной БД

Нужно выполнить следующую последовательность действий:

1. Исключение элементов, несовместимых с реляционной моделью данных.
2. Формирование набора таблиц для логической структуры реляционной БД.
3. Проверка полученных таблиц с учетом требований нормализации.
4. Определение ограничений целостности данных.

1. Исключение элементов, несовместимых с реляционной моделью данных

Концептуальная модель данных часто содержит конструкции, для которых *нет поддержки* в реляционных СУБД.

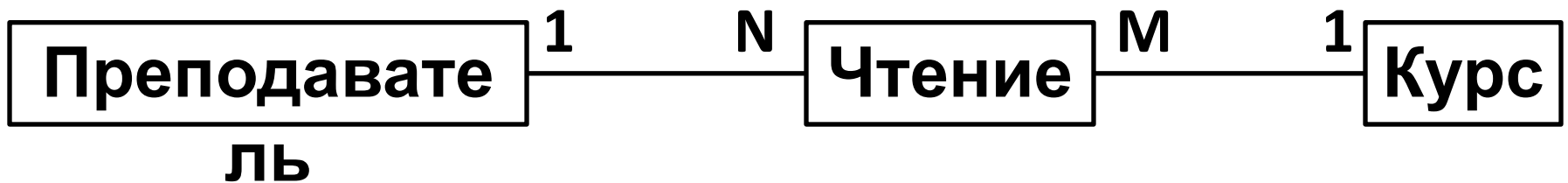
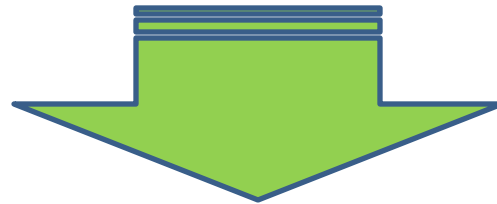
На этом этапе от таких конструкций *нужно избавиться* путем их преобразования.

Преобразованию подлежат *следующие элементы* концептуальной модели данных:

- a) связи типа «многие ко многим»;
- b) сложные связи;
- c) многозначные атрибуты;
- d) связи с атрибутами;

1a) Исключение связи «многие ко многим»

Вместо связи N:M нужно ввести еще одну промежуточную сущность и две



1b) Преобразование сложных связей

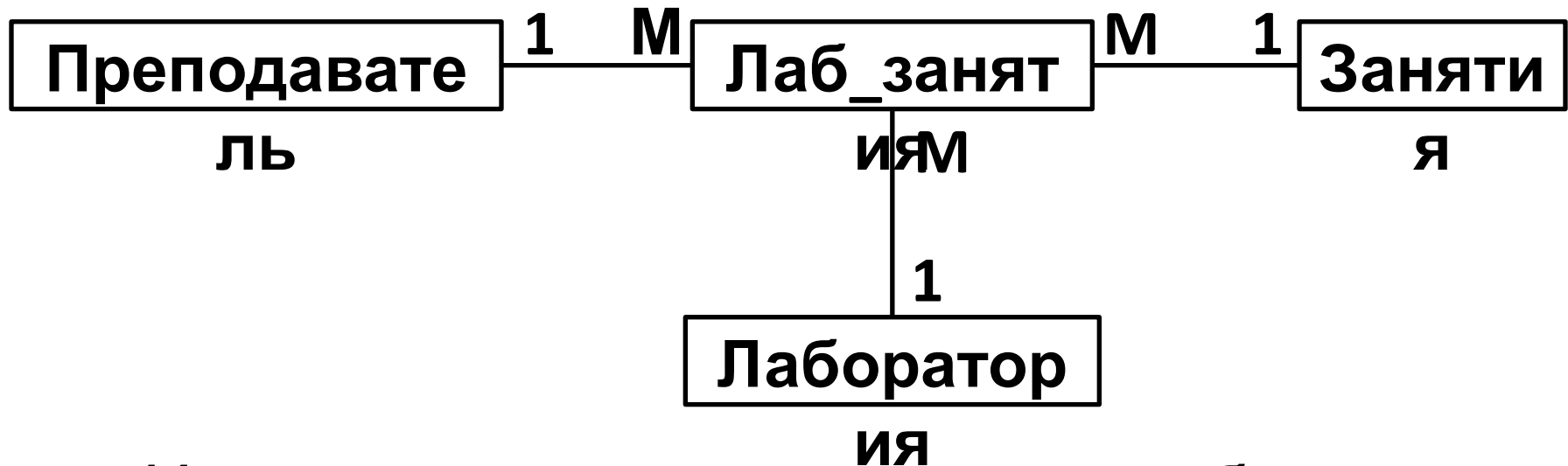
Исключение сложной связи (степень > 2) идет по следующему сценарию:

- в модель добавляется новая сущность;
- вводятся бинарные связи типа 1:M для связи этой сущности с исходными



**Пример
сложной
связи**

Сложная связь после преобразования:



1с) Исключение многозначных атрибутов

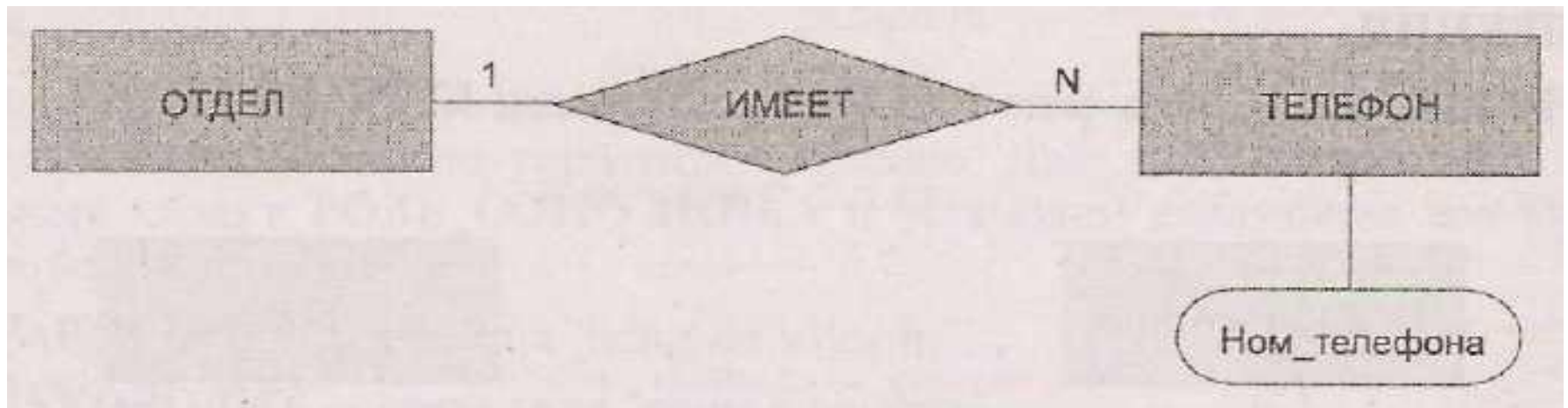
Вместо такого атрибута вводится новая сущность с соответствующим однозначным атрибутом, который становится первичным ключом.

Между новой и исходной сущностями

Пример исключения многозначного атрибута

Пусть концептуальная модель содержит сущность *ОТДЕЛ* с атрибутом **Номер_телефона**.

Если некоторые отделы имеют несколько контактных телефонов, то этот атрибут относится к типу **многозначного**.



2. Формирование набора таблиц для логической структуры реляционной БД

Для каждой сущности создается таблица и в нее включают все *простые атрибуты* этой сущности.

В случае *составного атрибута* в таблицу включают отдельные *простые части* этого атрибута.

Связи между разными таблицами реализуются по схеме «*первичный ключ/внешний ключ*».

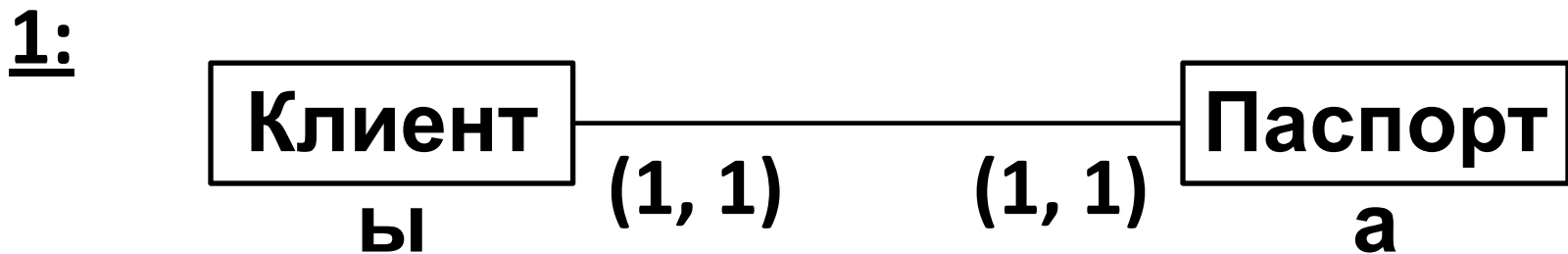
Суть этой схемы: из родительской сущности копия первичного ключа передается в

При реализации бинарных связей **типа 1:1** возможны следующие варианты:

1. Для обеих сторон участие в связи **полное** (т.е. связь **обязательная**)

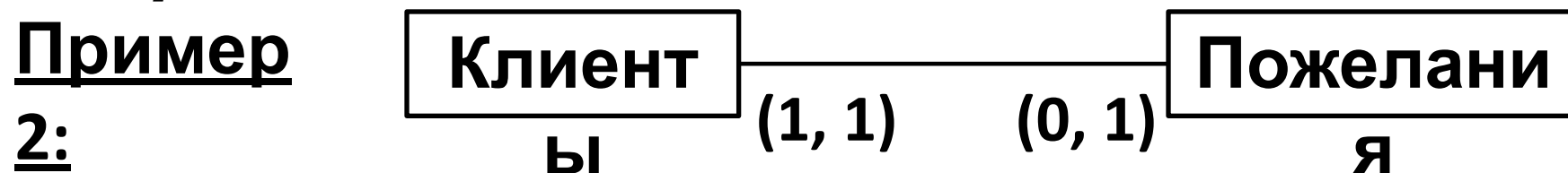
Такие сущности целесообразно

Пример **привести**.



В этом случае целесообразно паспортные данные включить в таблицу **КЛИЕНТЫ**.

2. Для одной из сторон участие в связи **неполное** (т.е. связь **необязательная**) Сущность, для которой имеет место неполное участие в связи, объявляется родительской. После этого можно применять схему **«первичный ключ/внешний ключ»**.



Для связи между таблицами *КЛИЕНТЫ* и *ПОЖЕЛАНИЯ* копия первичного ключа таблицы *КЛИЕНТЫ* передается в таблицу *ПОЖЕЛАНИЯ* и становится там внешним ключом

3. **Обе стороны** характеризуются **неполным** участием в связи

Наилучшее решение – дополнительно ввести промежуточную таблицу для связи конкретных экземпляров исходных

Пример
сущностей

3:

