

# ERM (Entity-Relation Model)

Анализ и проектирование  
структур данных с  
использованием CASE-средств

# Стандарт IDEF1x. Общие сведения

- ➔ Разработан в 1983 году в рамках проекта военного ведомства США "Интегрированные системы информационной поддержки" (ИСАМ).
- ➔ Как методология семантического моделирования данных.
- ➔ Стал расширением методологии IDEF1.

# *IDEF1x. Базовые определения*

➔ Логическая модель.

---

➔ Физическая модель.

---

➔ Сущность (Entity).

---

➔ Атрибут (Attribute).

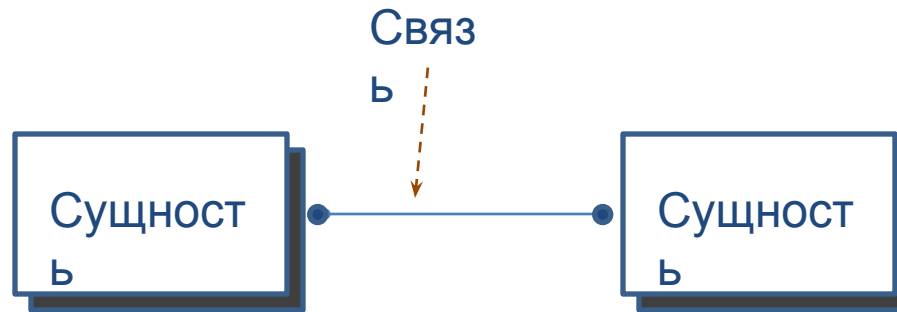
---

➔ Связь (Relationship).

---

➔ Ключ.

---



## ➔ Правила именованя связей (Verb Phrase)

Выражает некоторое бизнес-правило, либо ограничение, например –

МЕНЕДЖЕР оформляет ДОГОВОР,

Связи под названием *предметная надстройка* называются, например,

1..N (читается «один-ко-многим»),

Имя связи указывается в направлении родительской к дочерней

двустороннего отношения «многие-ко-многим» используют имена в обоих

направлениях

## ➔ Мощность связи (Cardinality)

Служит для обозначения отношения числа экземпляров родительской

сущности к числу экземпляров дочерней.

1..1 предположен. Символ

1..∞ или 1. Символ

1..∞ const. Обозначается соответствующим

## Идентифицирующая связь.

Связь между зависимой (дочерняя сторона связи) и независимой (родительская сторона связи) сущностями.

Сотрудник



имеется

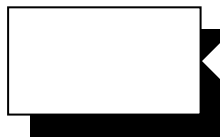
Ребёнок



## Неидентифицирующая связь.

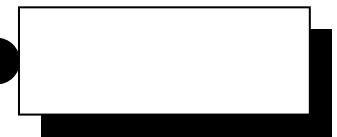
Необязательная

Отдел



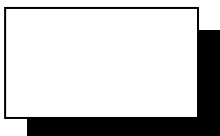
работает

Сотрудник



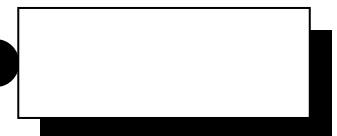
Обязательная

Отдел



работает

Сотрудник



## ➔ Виды ключей.

**Потенциальным ключом** - атрибут или группа атрибутов, однозначно

идентифицирующих экземпляр сущности.

В общем случае у сущности может быть несколько потенциальных ключей,

хотя для целей однозначной идентификации достаточно одного.

**Сложный ключ (составной)** – ключ, содержащий более одного атрибута.

**Первичный ключ** - один из потенциальных ключей.

**Альтернативный ключ** - потенциальный ключ, не ставший первичным.

**Суррогатный ключ** - создаётся на основе искусственно созданного атрибута,

не имеющего аналогов в предметной области, это – автоинкрементный столбец, уникальность значений которого поддерживается СУБД.

## ➔ Требования к первичному ключу.

Уникальност  
ь

Два экземпляра сущности не должны иметь одинаковых значений атрибутов, образующих ключ

Компактност  
ь

Сложный ключ не должен содержать ни одного атрибута, удаление которого не привело бы к потере уникальности.

Простот  
а

При выборе первичного ключа предпочтение отдаётся **более простым** ключам, т.е. ключам, содержащим меньшее количество атрибутов.

Полная  
определенность

Ни один из атрибутов, образующих ключ, не должен содержать значение с предопределённым значением NULL (пустой, неопределённый).

Постоянство во  
времени

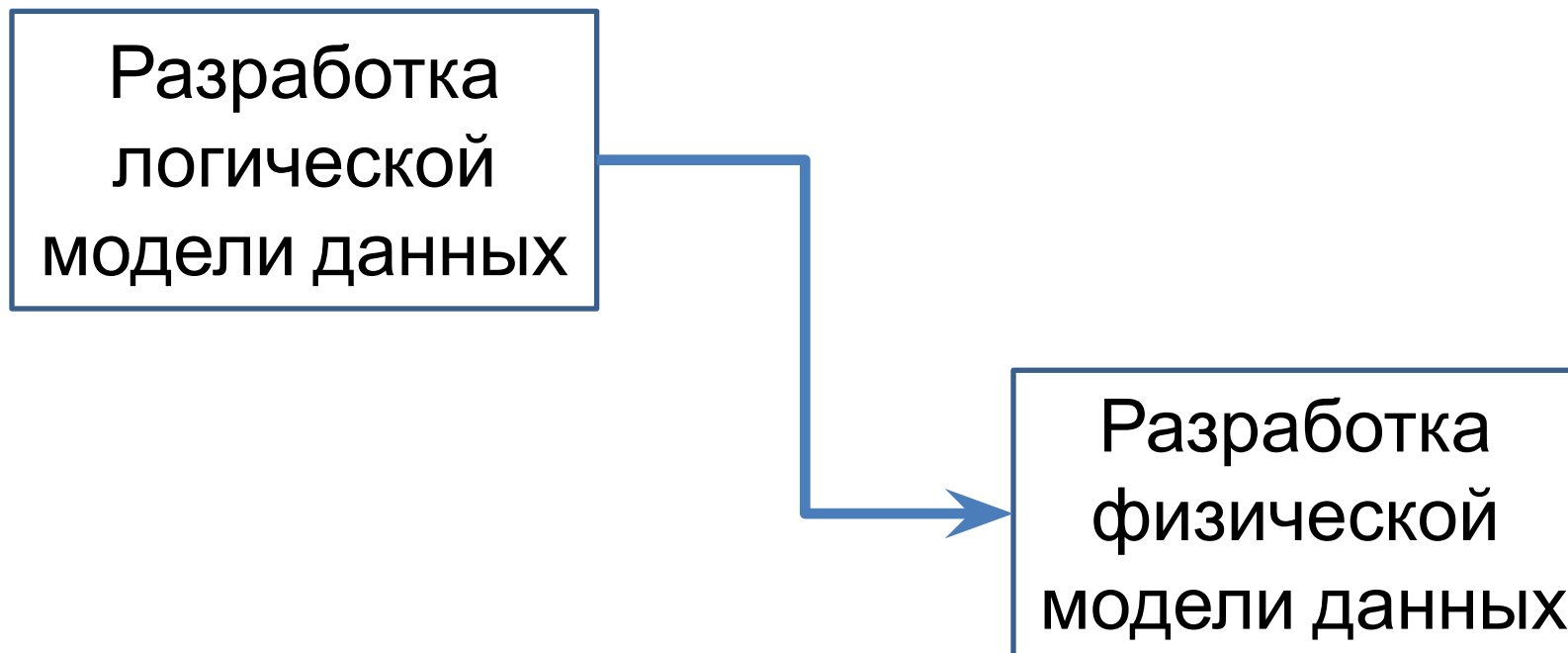
Значения атрибутов ключа не должны изменяться на протяжении всего времени его существования.

# Пример ER диаграммы

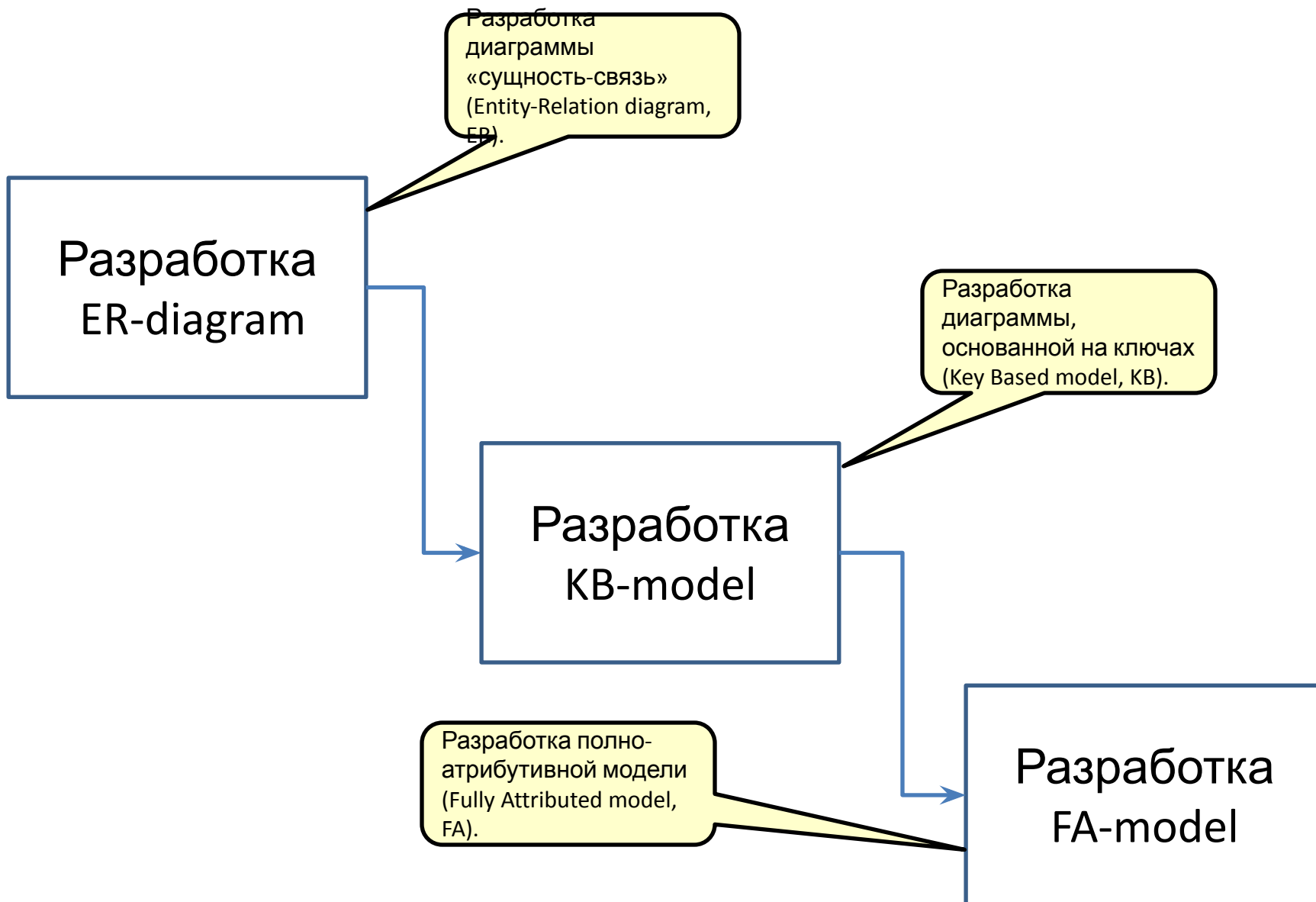
Проектирование модели данных  
на примере Интернет магазина



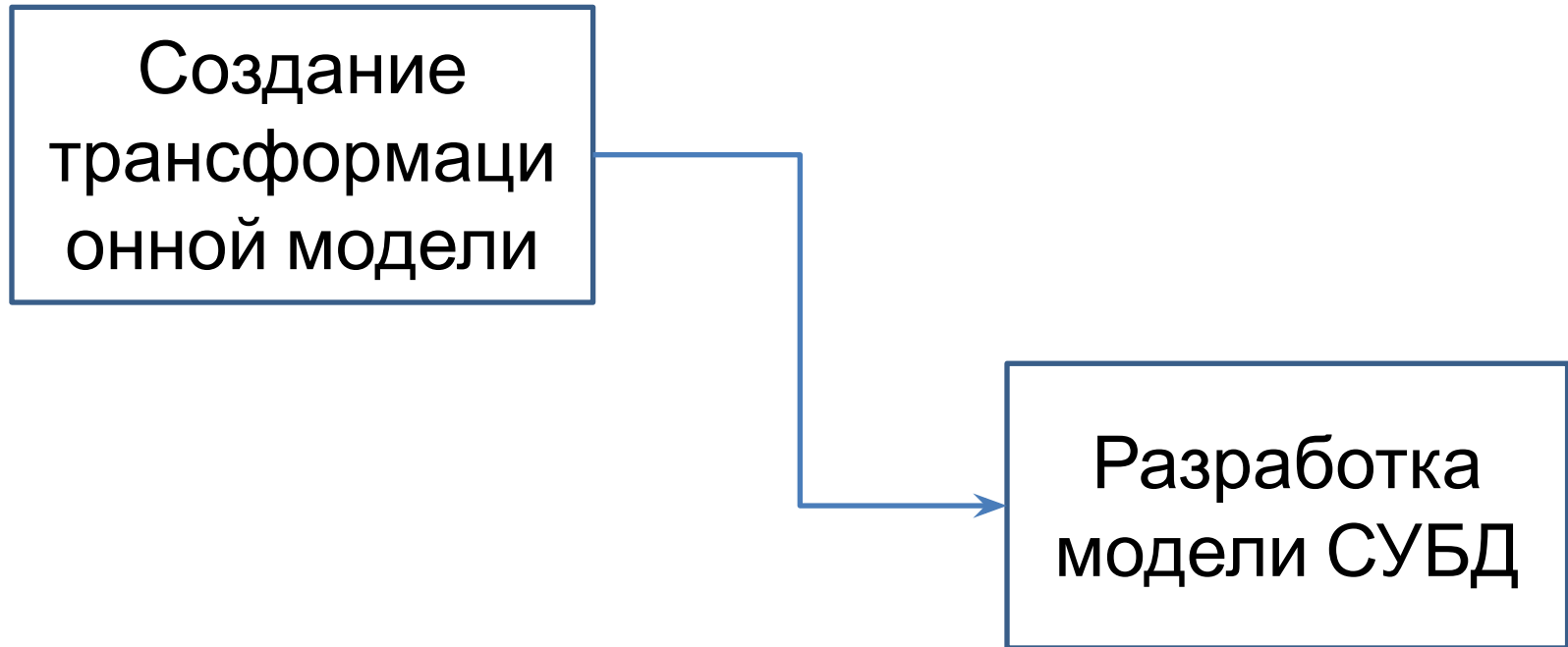
## *Фазы жизненного цикла проектирования*



# Этапы разработки логической модели данных



# Этапы разработки физической модели данных

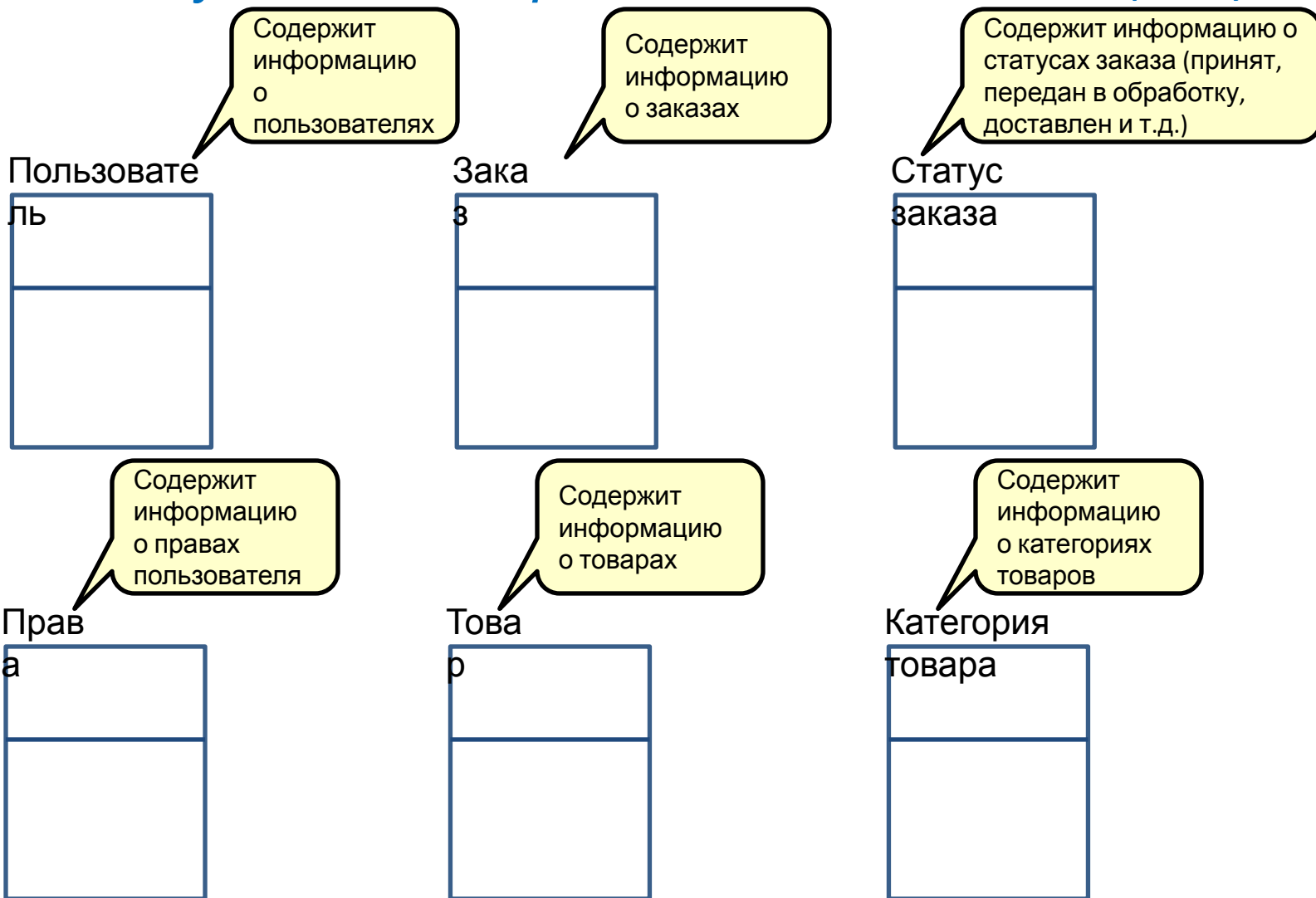


# Разработка диаграммы «сущность-связь» (ER)

## ➔ **Последовательность разработки**

- Выделение сущностей предметной области (ПО)
- Описание сущностей
- Формирование тематических областей данных
- Определение ключей и основных атрибутов сущностей
- Определение связей
- Описание связей

# Выделение сущностей предметной области (ПО)



# Выделение тематических областей данных

Данные,  
описывающие  
пользователей.

Пользовате  
ль


Данные,  
описывающие заказы  
пользователей.

Зака  
з


Статус  
заказа

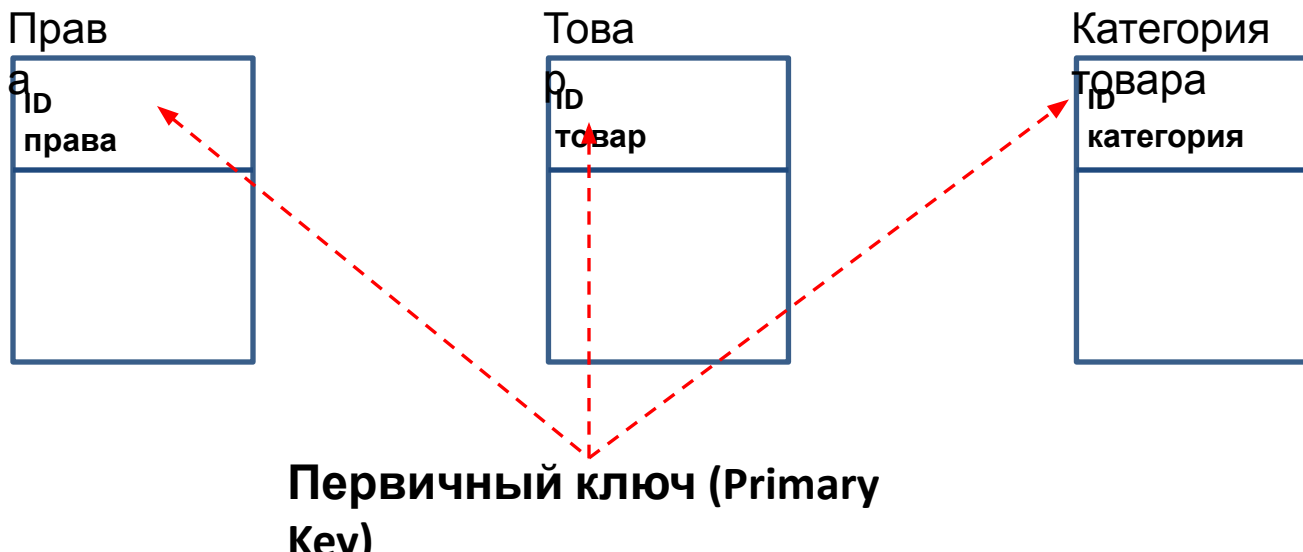
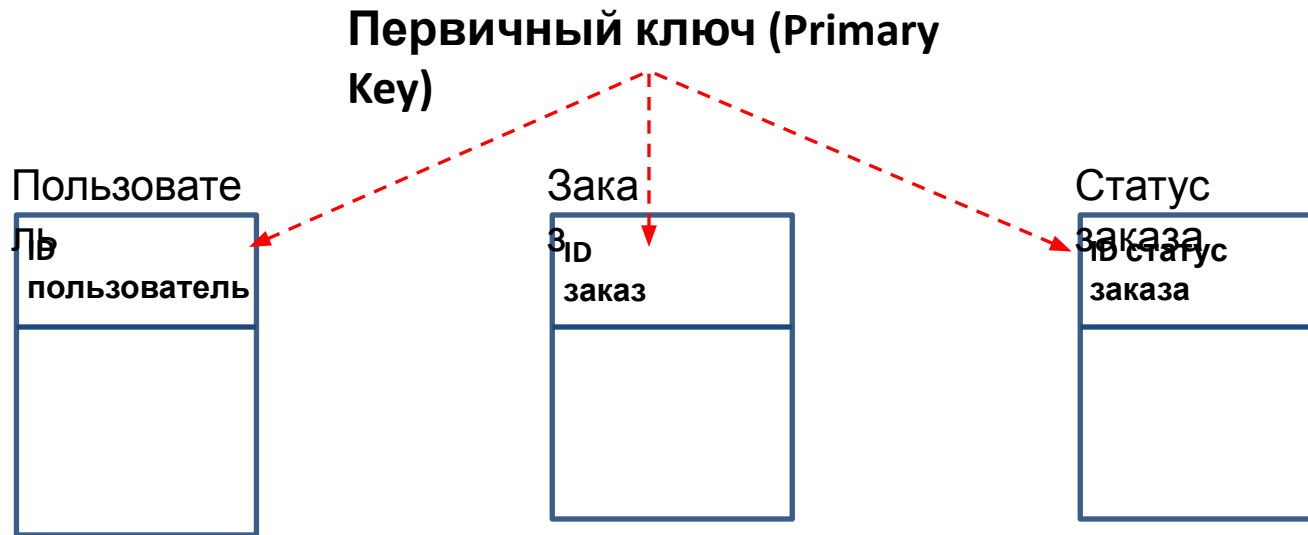

Прав  
а


Това  
р


Категория  
товара


Данные,  
описывающие товары  
интернет магазина.

# Разработка диаграммы, основанной на ключах (Key Based model, KB).



# Разработка полно-атрибутивной модели (Fully Attributed model, FA)

## ➔ Порядок разработки FA Model

Формирование полных наборов атрибутов каждой из сущностей.

Использование

доменов

Имя связи в направлении предок – потомок (только для связей типа

N..N)

Нормализация

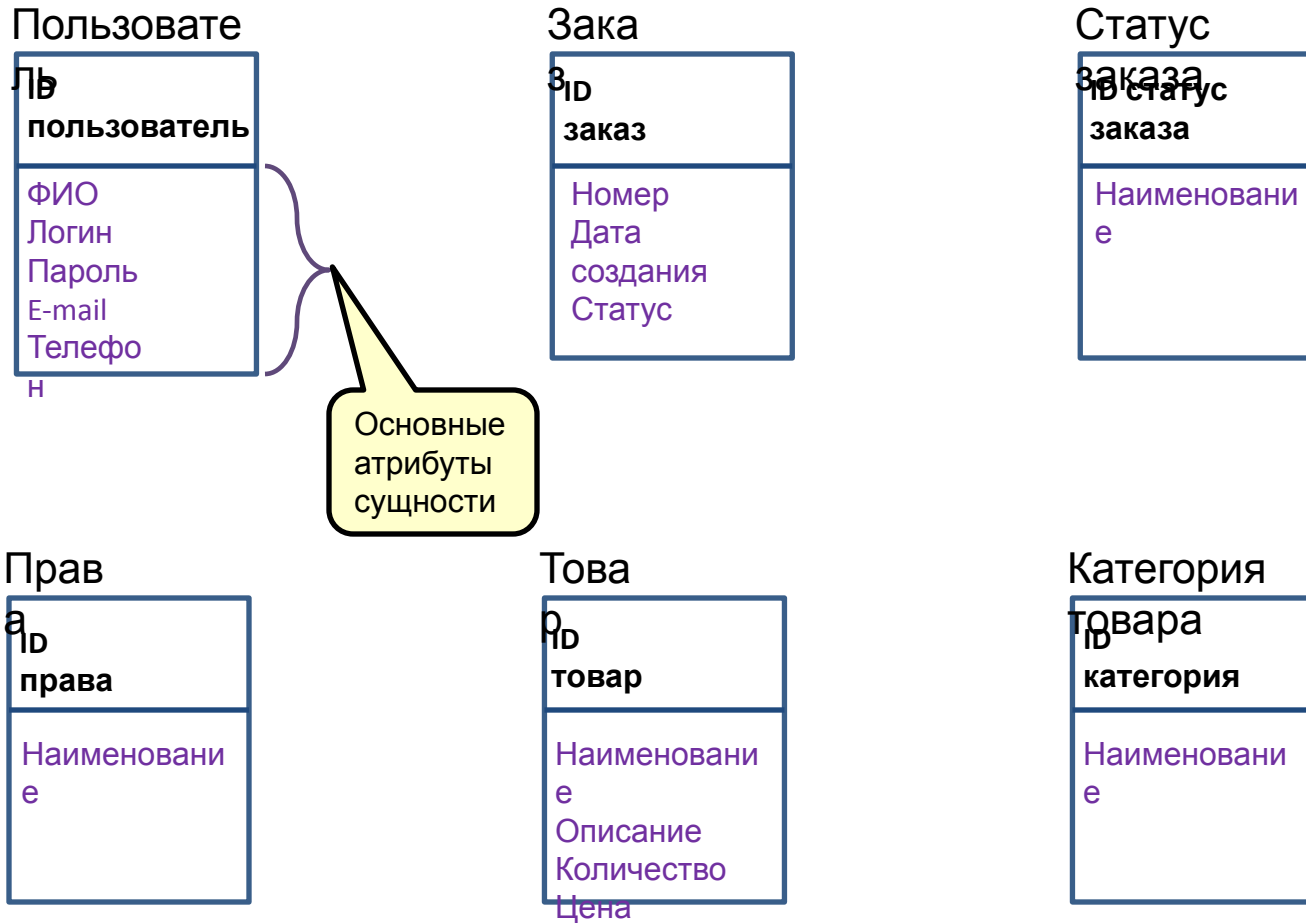
я

### Рекомендации по приведению модели данных к нормальному виду

- ❑ Для приведения модели к **первой нормальной форме** удалите повторяющиеся атрибуты, либо группы атрибутов сущности
- ❑ Для приведения модели к **второй нормальной форме** удалите атрибуты, зависящие только от части уникального идентификатора
- ❑ Для приведения модели к **третьей нормальной форме** удалите атрибуты, значения которых зависят от атрибутов, не входящих в уникальный идентификатор.



# Разработка полно-атрибутивной модели (Fully Attributed model, FA)



# Определение связей между сущностями ПО

## ➔ Порядок определения связей

Классификация

СВЯЗИ

Имя связи в направлении потомок –

предок

Имя связи в направлении предок – потомок (только для связей типа

N..N)

Мощность

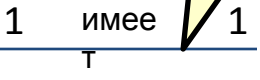
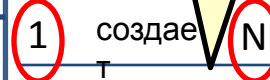
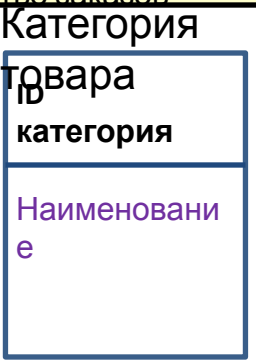
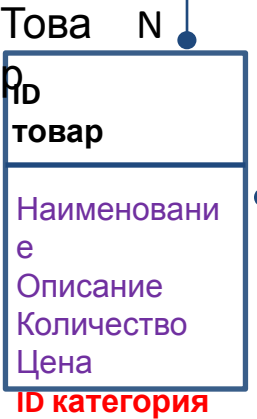
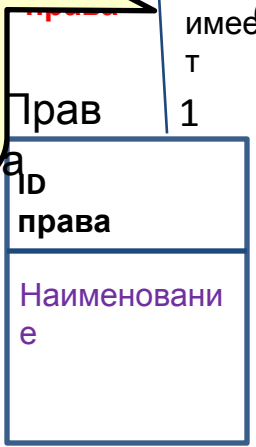
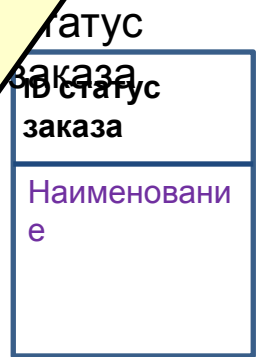
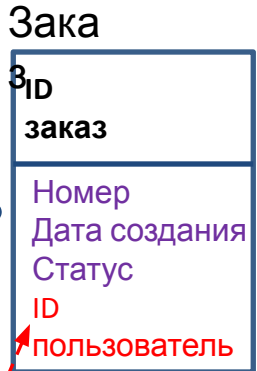
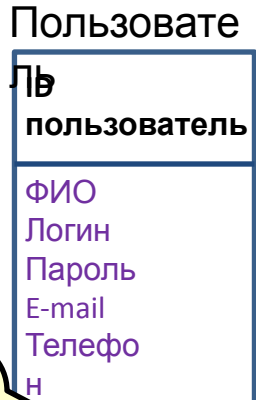
СВЯЗИ

Определение связи (по аналогии с определением сущности)

# Определение связей между сущностями ПО

**Связь «один ко многим»**  
Один пользователь может создать неограниченное количество заказов.

**Связь «один к одному»**  
Заказ имеет один определенный статус в определенный период времени.



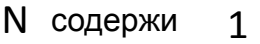
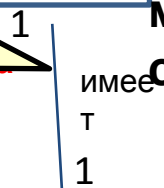
**Связь «один к одному»**  
Пользователь имеет права, закрепленные за одним идентификатором права.

**МОЩНОСТЬ СВЯЗИ**

**Связь «многие ко многим»**  
Заказ содержит неограниченное количество товара. Единица товара может входить в неограниченное количество заказов.

Миграция атрибутов.  
Внешний ключ.

**Связь «один ко многим»**  
Категория товара содержит неограниченное количество товара



## Миграция атрибутов

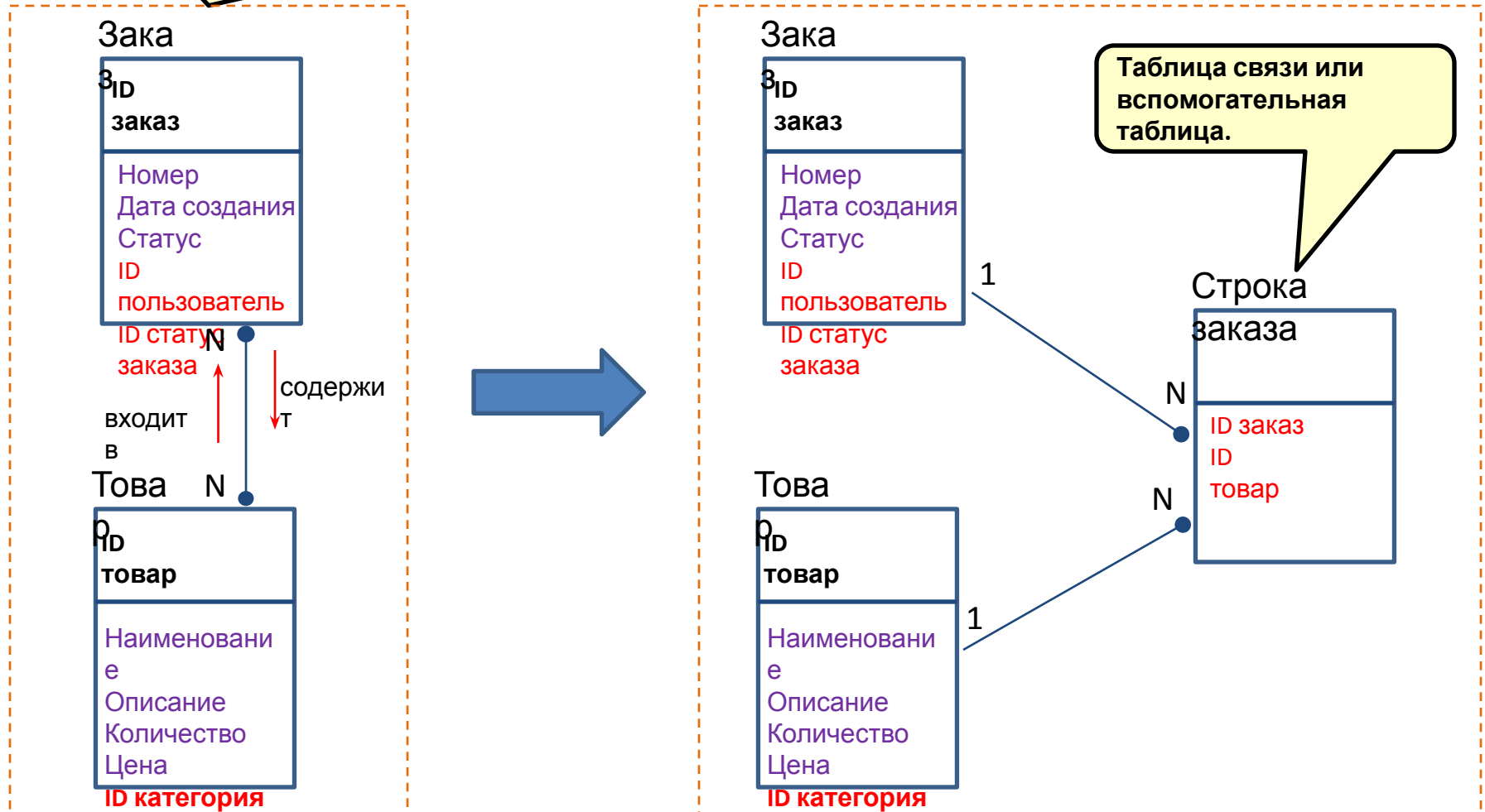
- ➔ При формировании связи между двумя сущностями атрибуты ПК родительской сущности мигрируют (копируются) в дочернюю сущность.
- ➔ Процесс миграции характерен как для идентифицирующей, так и к неидентифицирующей связи, однако его реализация различна.
- ➔ Атрибуты, мигрировавшие по идентифицирующей связи, входят в ПК дочерней сущности, а атрибуты, мигрировавшие по неидентифицирующей связи – нет.
- ➔ В любом случае, совокупность атрибутов, появившаяся в дочерней сущности путём миграции, образует так называемый **внешний ключ** (ЕК)

# Связь «многие ко многим» в схеме СУБД

Рассмотрим связь «многие ко многим» в приведенном примере.

Заказ может содержать большое количество товара и каждый товар, в свою очередь, может входить в несколько заказов.

Для практической реализации данной связи в схеме СУБД необходимо ввести таблицу **связи**, где каждому ID заказа будет соответствовать несколько ID товара.



# Создание трансформационной модели

- ➔ Создается CASE-средством автоматически.
- ➔ Содержит всю информацию, необходимую для генерации схемы данных в реляционной СУБД, поддерживающей стандарт SQL.
- ➔ Инвариантна к конкретной реализации.

- ➔ Для перехода к **модели СУБД** необходимо выбрать в интерфейсе CASE-средства одну из доступных СУБД.
- ➔ Переход к физической модели не является необратимым: её можно будет трансформировать в другую физическую модель.
- ➔ Однако при этом все настройки, специфичные для данной СУБД, будут потеряны.