# Семантическая модель данных

**Модель ER - “сущность-связь”. Назначение ER-модели. Основные понятия ER-модели. Трансформация ER- модели в реляционную модель.**

Реляционная модель данных (РМД) достаточна для моделирования многих предметных областей. Но она имеет и свои недостатки:

1. В РМД нет достаточных средств для представления смысла (семантики) предметной области.
2. В РМД нет средств для представления зависимостей между данными (отношениями)
3. Для многих приложений нет достаточно представления данных в виде плоских таблиц
4. РМД не предлагает никакого аппарата для разделения сущностей и связей.

Указанные недостатки привели к созданию семантических моделей данных(СМД).

На практике СМД используются на первой стадии проектирования БД. При этом в терминах СМД описывается концептуальная (понятийная) схема БД, которая затем преобразуется к реляционной или другой схеме. Этот процесс описан соответствующими методиками.

## ER-модель предметной области

Одной из наиболее популярных СМД является модель **«сущность-связь» (entity-relationship) или ER-модель**, предложенная Ченом в 1976 г. Она основана на графических диаграммах, включающих небольшое число разнообразных компонентов. ER-модели получили распространение в CASE-системах, поддерживающих проектирование РБД.

CASE- системы - *computer-aided software engineering*) — набор инструментов и методов программной инженерии для проектирования программного обеспечения, который помогает обеспечить высокое качество программ, отсутствие ошибок и простоту в обслуживании программных продуктов

Примеры CASE-систем – ERWIN, BPWIN, IDEF, Rational Rose и т.д.

Основными понятиями ER-модели являются **сущность, связь, атрибут.**

**Сущность** это реальный или абстрактный объект, информация о котором должна сохраняться. В ER-модели задается в виде прямоугольника с именем сущности ( рис 5.1.).

# ЧИТАТЕЛЬ

Иванов

Петров

Сидоров

Рис. 5.1. Сущность ЧИТАТЕЛИ

Имя сущности это тип объекта. Для наглядности имя сопровождается конкретными примерами, **экземплярами сущности**. Экземпляры должны быть различаться друг от друга. Это требование аналогично требованию отсутствия кортежей дубликатов в реляционной модели данных.

**Связь** – графически изображаемая ассоциация между двумя сущностями. Связь всегда бинарная, т.е. связывает две сущности или сущность между собой. В последнем случае связь называется рекурсивной.

Связь имеет два конца. На каждом конце указывается:

* имя связи;
* степень связи, сколько экземпляров сущности связывается;
* обязательность связи, т.е. любой ли экземпляр сущности должен участвовать в связи.

Обязательный конец связи изображается сплошной линией, необязательный конец –прерывистой. Примеры ER-схем приведены на рис 5.2.

Билет

Пассажир

для

имеет

а)

Отдел

Сотрудник

Состоит из

Работает в

б)

Рис. 5.2. Примеры ER-схем

В месте стыковки связи с сущностью используется трехточечный вход в прямоугольник сущности, если в связи могут участвовать несколько экземпляров сущности или одиночный, если один экземпляр.

Лаконичная трактовка диаграммы а: каждый билет предназначен для одного и только одного пассажира, каждый пассажир может иметь 0 или более билетов. Трактовку диаграммы б предлагается сделать самостоятельно.

Пример рекурсивной связи приведен на рисунке 5.3.

ЧЕЛОВЕК

Сын

Отец

Рис. 5.3. Рекурсивная связь

Трактовка связи следующая: каждый человек является сыном другого человека. У одного отца может быть 0 или более сыновей.

**Атрибут** сущности – это свойство сущности, служит для уточнения, классификации, идентификации, числовой характеристики или выражения состояния сущности. Имена атрибутов пишутся под именем сущности строчными буквами.

Существует три типа свойств:

**Ключевые свойства** позволяют различать экземпляры сущности.

**Дифферен­циальные свойства** содержат смысл сущности, то, что отличает сущность от другой сущности.

**Валентные свойства** используются для связи с другими сущностями.

## Операции над сущностями

Различают три основных типа абстрагирования и соответственно три вида операций над сущностями: **агрегация, обобщение, ассоциация**. Соответственно обратные логические операции над сущностями - это **декомпозиция, специализация и индивидуализация.**

**Агрегация** – это метод абстрагирования, при котором сущность-агрегат связана с другими сущностями, как целое связано с частями. **Декомпозиция** – это деление сложной сущности на компоненты, операция обратная агрегации.

Свойства агрегата – это совокупность свойств компонентов. Обычно составные части агрегата имеют совершенно разный набор свойств. Например, сущности КАФЕДРЫ, СТУДЕНТЫ, АУДИТОРИИ, СОТРУДНИКИ являются компонентами сущности УНИВЕРСИТЕТ. УНИВЕРСИТЕТ представляет собой сущность-агрегат.

**Обобщение** – это метод абстрагирования, при котором обобщенная сущность связана с другими сущностями отношением “род-вид”. **Специализация** – операция обратная обобщению. Например, обобщенная сущность – УЧАЩИЕСЯ может быть разделено на категории СТУДЕНТЫ, ШКОЛЬНИКИ, АСПИРАНТЫ. Обобщенная сущность ПРЕДМЕТЫ может быть разделена на категории ОБЯЗАТЕЛЬНЫЕ, ФАКУЛЬ­ТАТИВЫ.

Категории обычно имеют много общих свойств, но некоторые свойства у них различны. Свойства обобщенной сущности представляют собой перечень общих свойств категорий, но могут включать и дополнительные свойства по сравнению с исходными, чаще всего это такие свойства, как Среднее значение, Сумма и т.п. На ER-диаграмме категории изображаются треугольниками, а обобщенное понятие – прямоугольником, который соединен с категориями связями типа "является видом…( is kind of..)" .

**Ассоциация** – это такая логическая операция, которая устанавливает связи между конкретными экземплярами сущностей. **Индивидуализация** – это независимое рассмотрение связанных ассоциацией сущностей. Обычно ассоциативные связи возникают в ходе реальных экономических процессов (бизнес - процессов). Например, ПОСТАВЩИКИ связаны с ПОСТАВКАМИ ТОВАРОВ, ПРОДАЖИ ТОВАРОВ связаны с ТОВАРАМИ, ПОКУПКИ ТОВАРОВ связаны с КЛИЕНТАМИ и т.д.

## Порядок построения ER-модели и построение реляционной схемы базы данных из ER-модели

ER - модели широко применяются как при ручном, так и при автоматизированном проектировании баз данных. При этом в некоторых системах, например, в системе IDEF, отсутствуют сложные понятия (агрегаты, ассоциации, обобщенные понятия), они заменяются связями. В разных системах применяются различные обозначения мощности связей и классов членства (обязательности связи) и т.д.

**Приведем рекомендуемый порядок построения ER-модели:**

1. В каждом внешнем представлении (пользовательском) выявить сущности и их свойства, исходя из анализа предметной области. Их должно быть не более 6-7 в каждом пользовательском представлении.
2. Обозначить сущности именами, которые должны быть краткими, отражать смысл и быть привычными для пользователя.
3. Выбрать ключевое свойство (свойства). Если такого нет, то вводят условное свойство “Порядковый №”.
4. Выявить связи между сущностями.
5. Найти общие свойства сущностей (обобщение);
6. Рассмотреть все пары сущностей и определить ассоциативные связи;
7. Определить степень каждой связи.
8. Записать формулы расчета, связывающие вычисляемые свойства.
9. Объединить модели, построенные для разных внешних (пользовательских) представлений. Объединение ведут по 2-4 модели за один шаг, получая более сложные объекты и устраняя противоречия (например, разные наименования одних свойств и одинаковые - для разных сущностей).

**Задача построения модели** предметной области **осложняется** тем обстоятельством, что **деление** информации **на свойства, сущности и связи условно**, и один и тот же элемент информации можно описать по-разному. Например, информацию о преподавателе можно рассматривать как сущность, а можно и как свойство сущности ДИСЦИПЛИНА; любую ассоциированную сущность можно рассматривать как связь между несколькими сущностями. Принятие конкретного решения зависит от специфики предметной области и применяемых средств автоматизации проектирования.

На основе созданной ER-схемы строится реляционная модель базы данных. **Порядок построения реляционной модели из ER-схемы следующий:**

* 1. Каждая простая сущность отображается в таблицу, при **этом имя сущности становится именем таблицы.**
  2. Каждый **атрибут сущности** становится столбцом таблицы с тем же именем, при этом выбирается более точный формат значения атрибута. Ключевые свойства преобразуются в первичные ключи таблицы.
  3. Столбцы, соответствующие необязательным атрибутам могут содержать неопределенные значения, обязательные столбцы не могут.
  4. Если в соответствующие ключевые свойства входят связи, к числу столбцов первичного ключа добавляется копия ключевого свойства сущности, находящейся на дальнем конце связи.
  5. Связи один-ко-одному, один-ко-многим преобразуются во внешние ключи. Делается **копия уникального идентификатора с конца связи «один», которая мигрирует в таблицу на конце связи «много», где и становится внешним ключом.**

## Пример построения ER- модели

Например, требуется разработать информационную систему «**Справочное аптек города**», которая позволяет получать справки о наличии лекарств в аптеках города. Система должна:

* предоставлять информацию обо всех аптеках города: номер, телефон, адрес, ФИО ответственного лица и др.
* предоставлять хотя бы краткую информацию о лекарствах: название, дозировка, применимость, какие заболевания лечит и др.
* пользователь должен получить информацию о том, в каких аптеках находится искомое лекарство и по какой цене.

Сначала создадим ER-модель данной предметной области, затем отобразим созданную модель в реляционную базу данных.

Выделим сущности. Их две, это АПТЕКА и ЛЕКАРСТВО.

Выделим свойства каждой сущности. У сущности АПТЕКИ имеются следующие свойства: номер аптеки, адрес, телефон. У сущности ЛЕКАРСТВА свойства: название лекарства, описание лекарства.

Сущности связаны между собой, так как это показано на рис. 2. Приведены свойства каждой сущности и звездочкой помечено ключе­вое свойство у каждой сущности.

###### АПТЕКА

\* Номер аптеки

Адрес

Телефон

**ЛЕКАРСТВО**

\*Номер лекарства

Название

Описание

имеет

находится в

Рис. 2

**Трактовка ER-схемы:** в АПТЕКЕ имеется 0 или более лекарств. В свою очередь ЛЕКАРСТВО находится в одной или нескольких аптеках. Иными словами между данными сущностями связь типа много-ко-многим.

**Отобразим данную ER модель в реляционную базу данных.**

* Каждая сущность отображается в реляционную таблицу. Таким образом, получаем две таблицы – АПТЕКИ и ЛЕКАРСТВА.
* Связь много-ко-многим в реляционной модели должна быть преобразована в связь один-ко-многим путем добавления еще одной таблицы, которую назовем РАЗМЕЩЕНИЕ ЛЕКАРСТВ В АПТЕКАХ.
* Копия первичного ключа таблицы АПТЕКИ мигрирует в таблицу РАЗМЕЩЕНИЕ ЛЕКАРСТВ В АПТЕКАХ и становится там внешним ключом, ссылающимся на первичный ключ таблицы АПТЕКИ.
* Копия первичного ключа таблицы ЛЕКАРСТВО мигрирует в таблицу РАЗМЕЩЕНИЕ ЛЕКАРСТВ В АПТЕКАХ и становится там внешним ключом, ссылающимся на первичный ключ таблицы ЛЕКАРСТВА.

Таким образом, получаем следующую реляционную схему базы данных:

**РАЗМЕЩЕНИЕ ЛЕКАРСТВ**

\*Номер аптеки

\*Номер лекарства

Количество

Цена

###### АПТЕКА

\*Номер аптеки

Адрес

телефон

###### ЛЕКАРСТВО

\*Номер лекарства

Название

Описание

## 

1

∞

∞

1

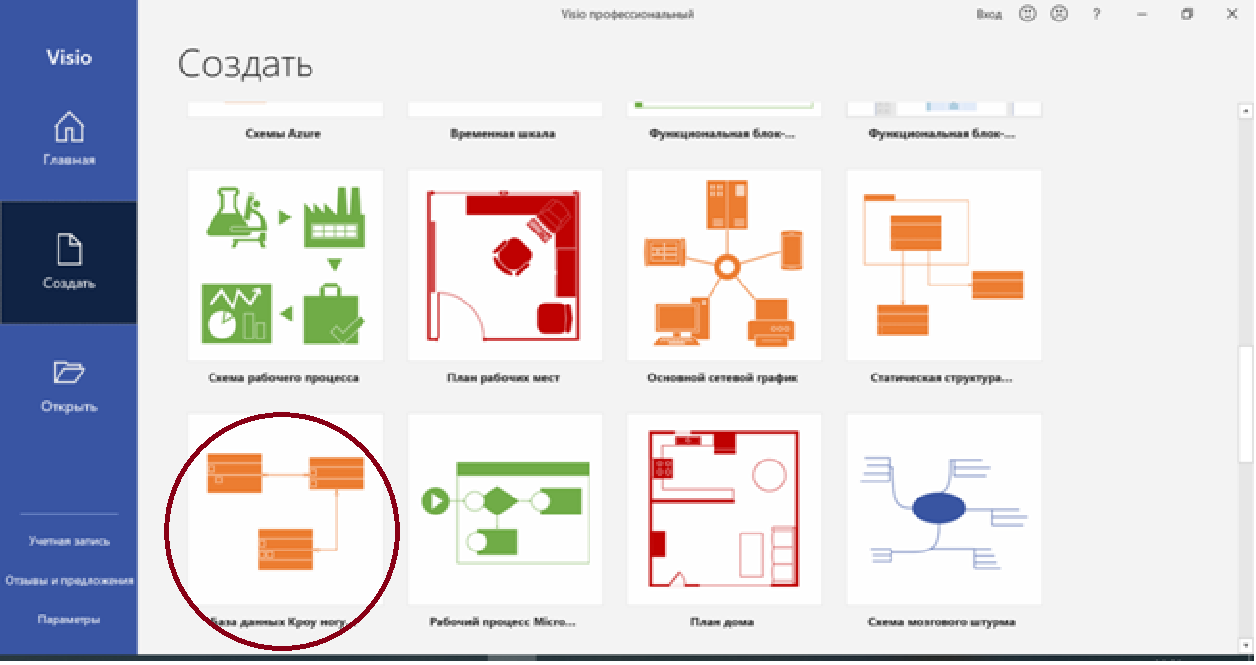
**Основные понятия и ключевые слова:** семантическая модель, ER-модель, сущность, экземпляр сущности, связь, атрибут сущности, агрегация, обобщение, ассоциация, агрегация, обобщение.

## Контрольные вопросы

1. Перечислить недостатки реляционной модели данных.
2. Основные понятия ER-модели. Привести пример ER-схемы.
3. Какие Вы знаете операции над сущностями. Привести примеры.
4. Каков порядок построения ER-модели.
5. Каков порядок преобразования ER-модели в реляционную базу данных.

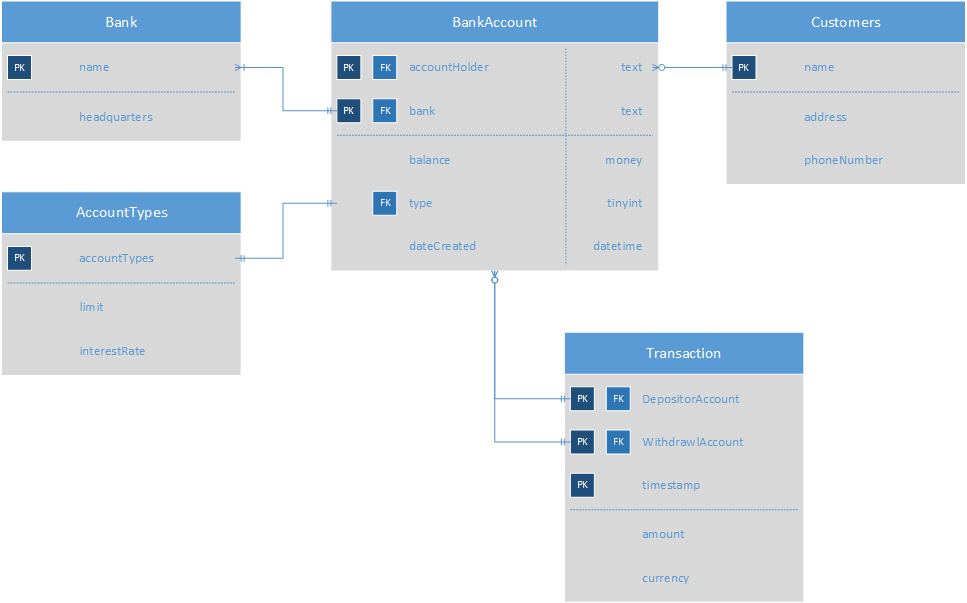
**MS VISIO**

ER-модели будем строить в MS VISIO. Среда разработки MS VISIO. Выбираем модель: база данных Crow Foots.



**Нотация Crow's Foot**

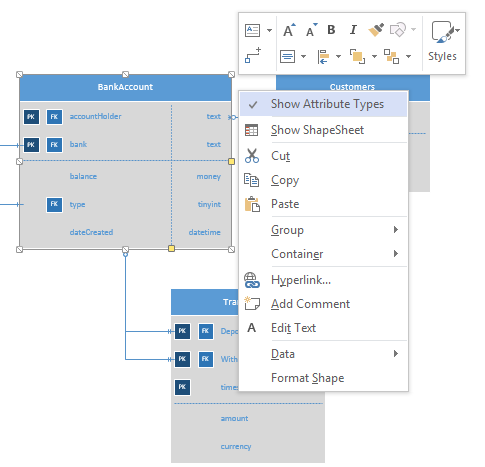
Нотация Crow's Foot используется наиболее часто. С ее помощью можно создавать четкие и понятные схемы при наличии большого числа атрибутов (как, например, в случае модели комплексной системы баз данных). Атрибуты заключаются в фигуры сущностей.

[](https://www.microsoft.com/en-us/microsoft-365/blog/wp-content/uploads/sites/2/migrated-images/40/1464.CrowsFoot.png)

Для отображения связей между сущностями служат соединительные линии отношений. **Наконечники показывают количество элементов в отношениях.**

Чтобы настроить это количество, можно щелкнуть отношение правой кнопкой мыши и **задать начальные и конечные символы**. Для большей точности количество элементов можно указать вручную. Отношения можно присоединить ко всей сущности или к отдельному атрибуту (к примеру, указывая, как присоединение будет работать с внешним ключом). Первичный и внешний ключи отображаются для каждого атрибута с помощью контекстного меню.

*Совет. Чтобы отобразить тип атрибута в нотации Crow's Foot и IDEF1X, в контекстном меню фигуры сущности нужно выбрать команду Show Type (Показать тип).*

*[](https://www.microsoft.com/en-us/microsoft-365/blog/wp-content/uploads/sites/2/migrated-images/40/7711.ShowType.png)*

В нотации Crow's Foot для упорядочения атрибутов в рамках сущности используются фигуры списков Visio. Данный метод также применяется в диаграммах баз данных IDEF1X и UML. Советы по использованию этих фигур см. в разделе о схемах классов нашей [статьи, посвященной созданию прекрасных гибких схем UML профессионального вида](https://www.microsoft.com/en-us/microsoft-365/blog/2012/10/22/professional-flexible-beautiful-uml-content/).

**Сущности и Связи**

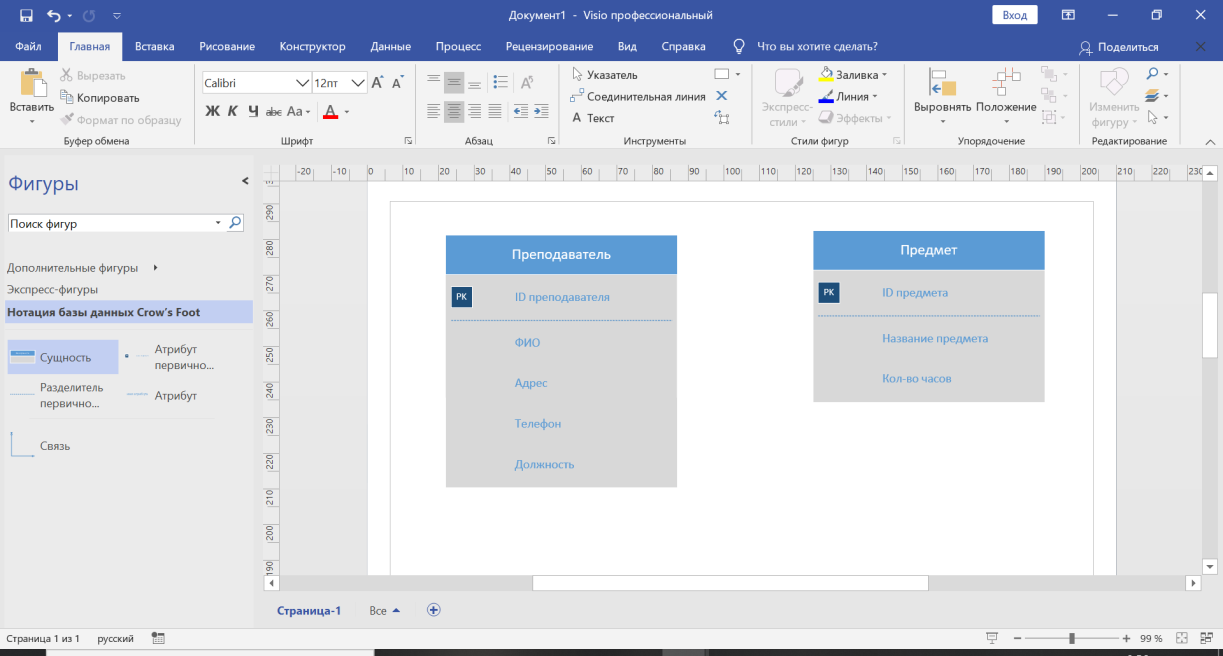
Создадим модель для предложения: *преподаватель может вести несколько предметов, и каждый предмет могут вести несколько преподавателей*.

Сущности: **Преподаватель** и **Предмет**.

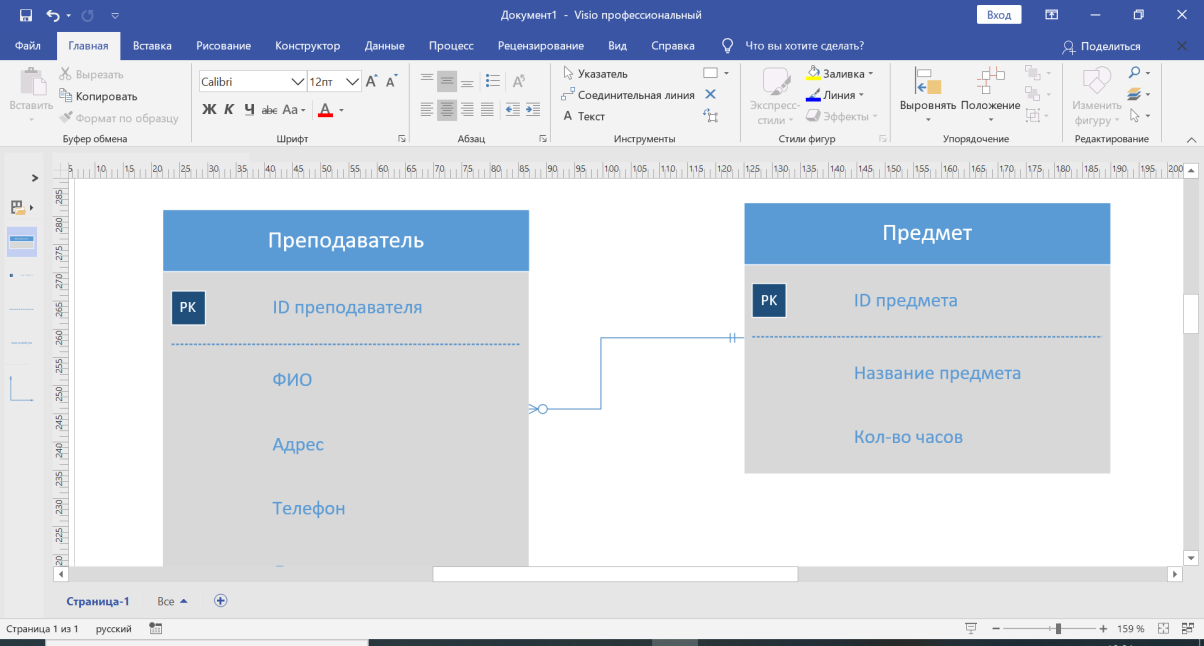
**Атрибуты** сущности **Преподаватель**: ФИО, Адрес Телефон,\*ID преподавателя (это первичный ключ).

**Атрибуты** сущности **Предмет: \***ID предмета, Название, кол-во часов.

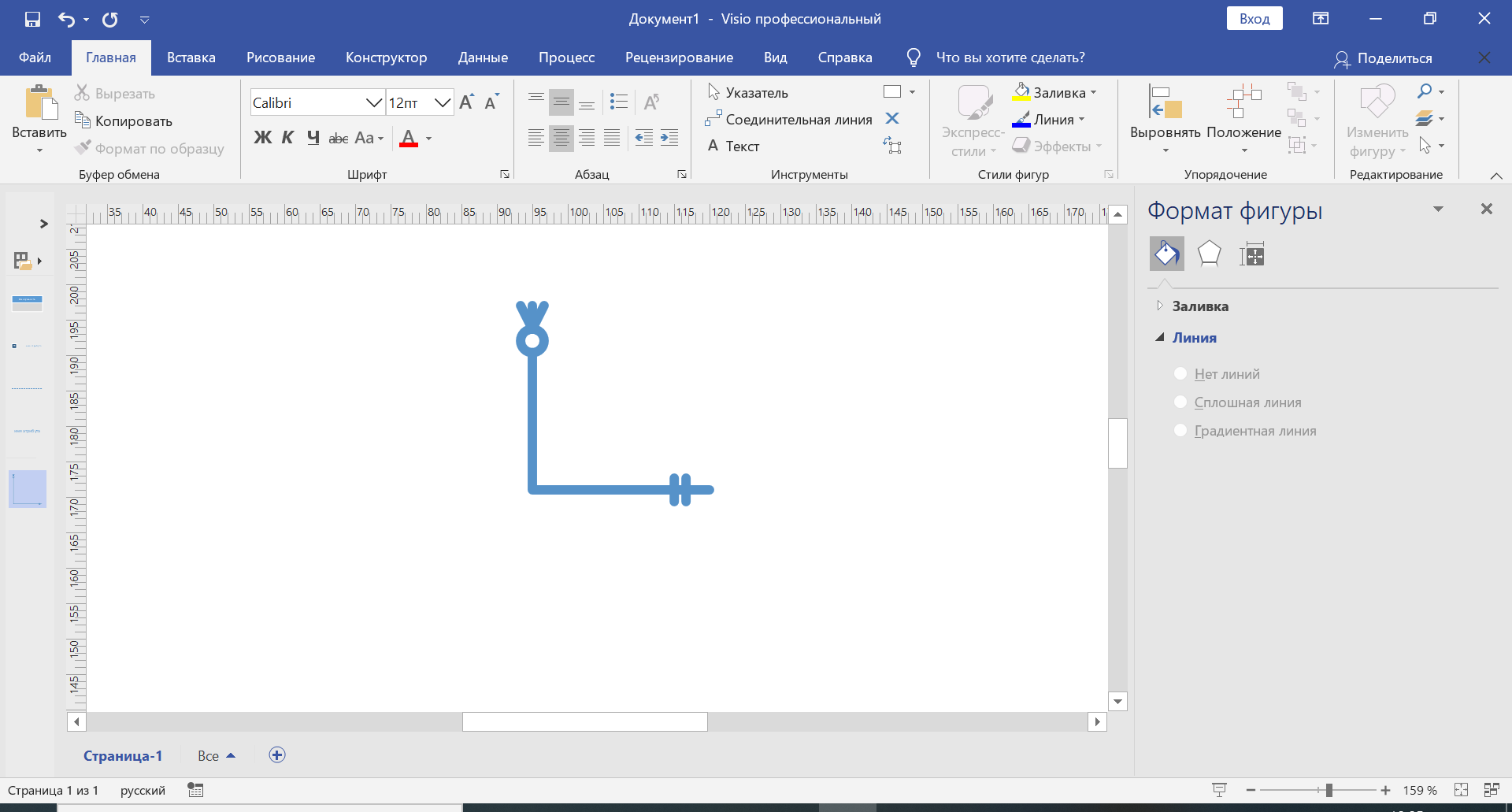
Создадим сущности. Для этого выделяем сущность в области Фигуры и перетаскиваем ее в область документа. Создаем атрибуты сущностей. Выделяем первичные ключи.



**Создадим связи.** Для этого выделяем связь в области Фигура и перетаскиваем ее в область документа. Прикрепляем каждый конец связи к нужной сущности. Не важно в каком месте прикрепляется связь, важно , что она связывает две сущности.

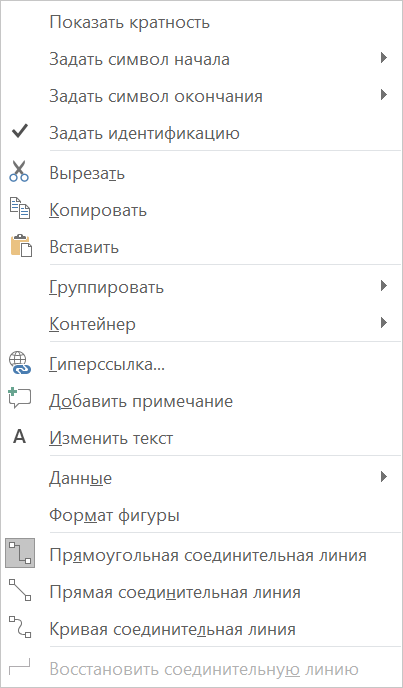


На каждом конце связи должны быть обозначены **степень связи и обязательность связи.** Пример связи:

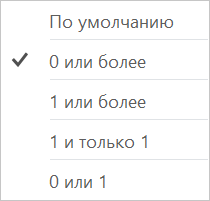


На одном конце **много и необязательно** на другом **1(один)**.

Для изменения степени связи и обязательности выделяем конец связи и по правой мыши открываем контекстное меню и выбираем пункты **Задать символ начала**, **Задать символ окончания.**



Пункт **Задать символ:**



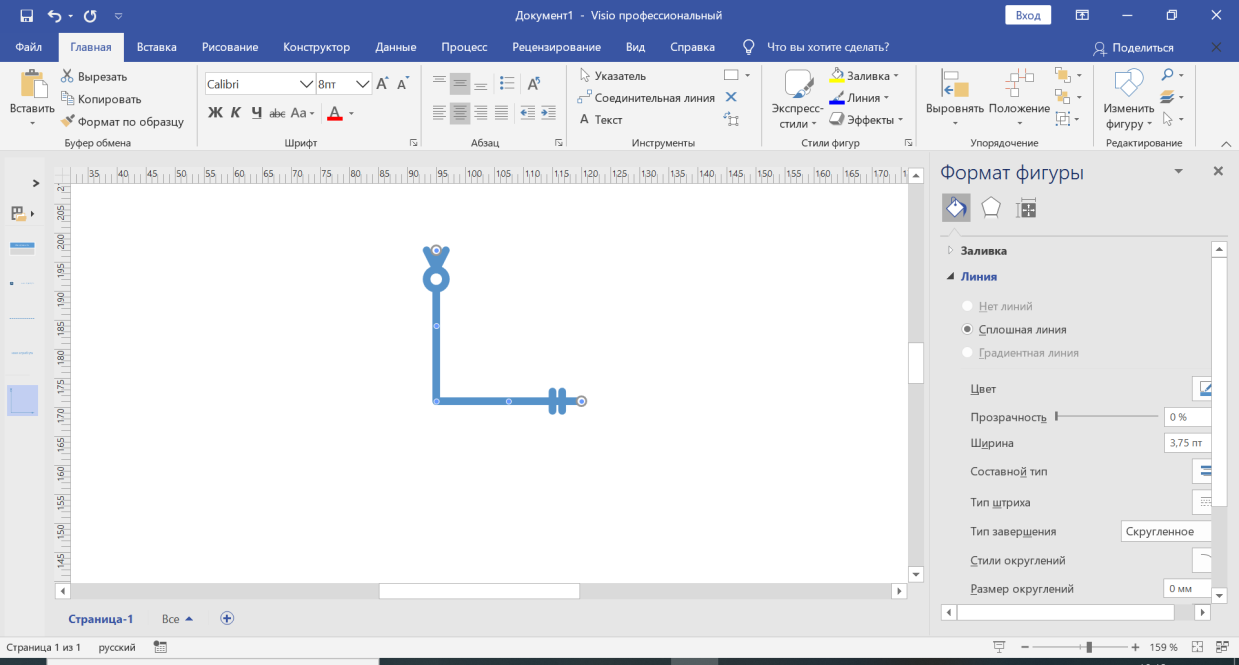
- **0 или более** - означает **необязательную связь** – **много**.

- **1 или более** - означает **обязательную связь – много**.

- **1 и только 1** – означает **обязательную связь один к одному.**

**- 0 или 1** - означает **необязательную связь один к одному.**

Для изменения формы связи, толщины линии и т.д. нужно выделить связь и открыть подменю **Формат фигуры:**



**Связь:**



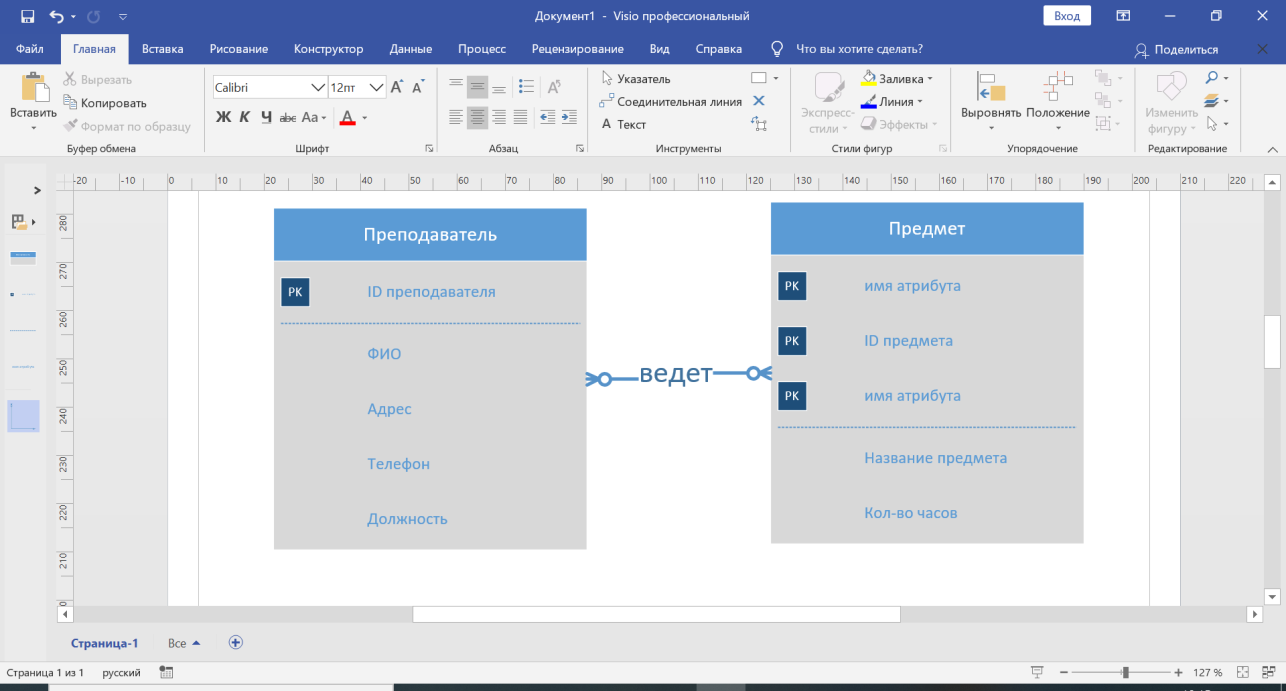
**Читается**: на одном конце **много и необязательно** ( об этом говорит 0 ), на другом конце **много и обязательно**. Т.е. **связь много-к-многим.**

**Связь**:



**Читается**: на одном конце **много и необязательно** ( об этом говорит 0 ), на другом конце **один** **и обязательно**. Т.е. связь **один-ко-многим.**

**Для задания имени связи** щелкаем на связи и вводим имя. На рис. приведен тип связи **много-ко-многим**:



**ВАЖНО.** Для ER-модели не важно где крепится связь, важно, что она соединяет две сущности. Для реляционной модели важно как выполняется соединение, связь как правило связывает первичный и внешний ключи.

**Задания для самостоятельной работы:**

1. Один жокей может участвовать в нескольких состязаниях и в каждом состязании могут участвовать несколько жокеев. Атрибуты задать самостоятельно.
2. Для предметной области СКАЧКИ создать ER-модель предметной области. Сохранить модель. На основе ER-модели создать в Access таблицы со связями и схему базы данных.
3. Для предметной области КИНОЛОГ создать ER-модель предметной области. Сохранить модель. На основе ER-модели создать в Access таблицы со связями и схему базы данных.
4. Построить ER-модель для следующего фрагмента: на каждом факультете учится несколько студентов, но каждый студент учится на одном факультете. Выделить сущности, определить атрибуты сущностей, установить связи.
5. Построить ER-модель для следующего фрагмента: каждый владелец имеет несколько собак и у каждой собаки только один владелец. Выделить сущности, определить атрибуты сущностей, установить связи.
6. Построить ER-модель для следующего фрагмента: предмет могут вести несколько преподавателей, и каждый преподаватель может вести несколько предметов. Выделить сущности, определить атрибуты сущностей, установить связи.
7. Построить ER-схему и на ее основании реляционную базу данных для предметной области поставки товаров в магазин.
8. Построить ER-схему и на ее основании реляционную базу данных для предметной области ШКОЛА, при этом требуется хранить информацию о школьниках, преподавателях, предметах, оценках.
9. Предложить свою предметную область и для нее построить ER-схему и на ее основании реляционную базу данных.

ПРИМЕРЫ

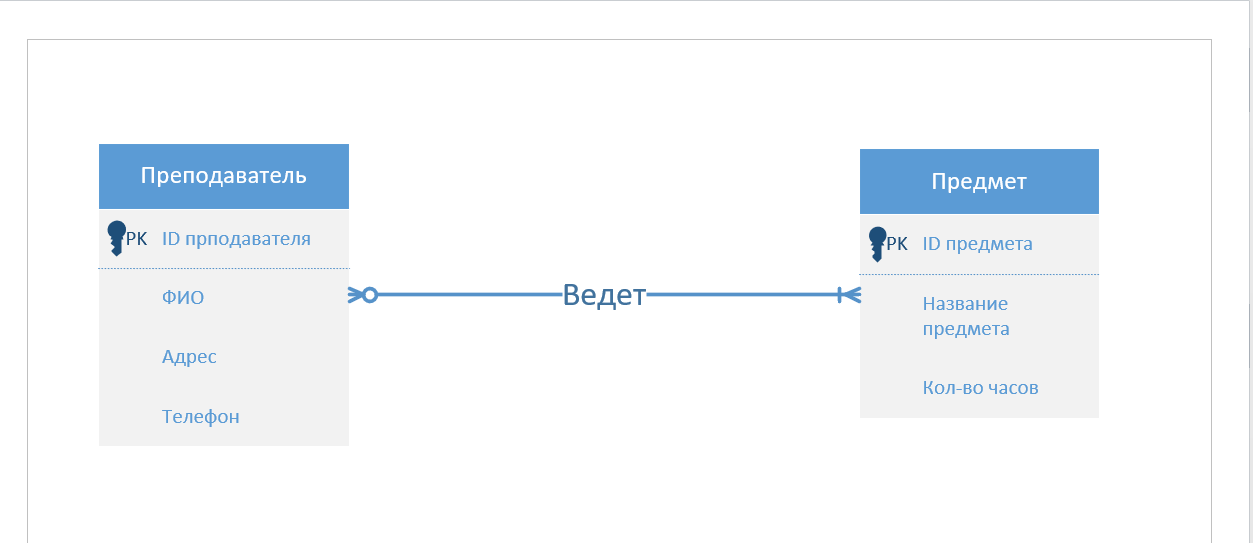
Разберем некоторые примеры

**Пример 1.**

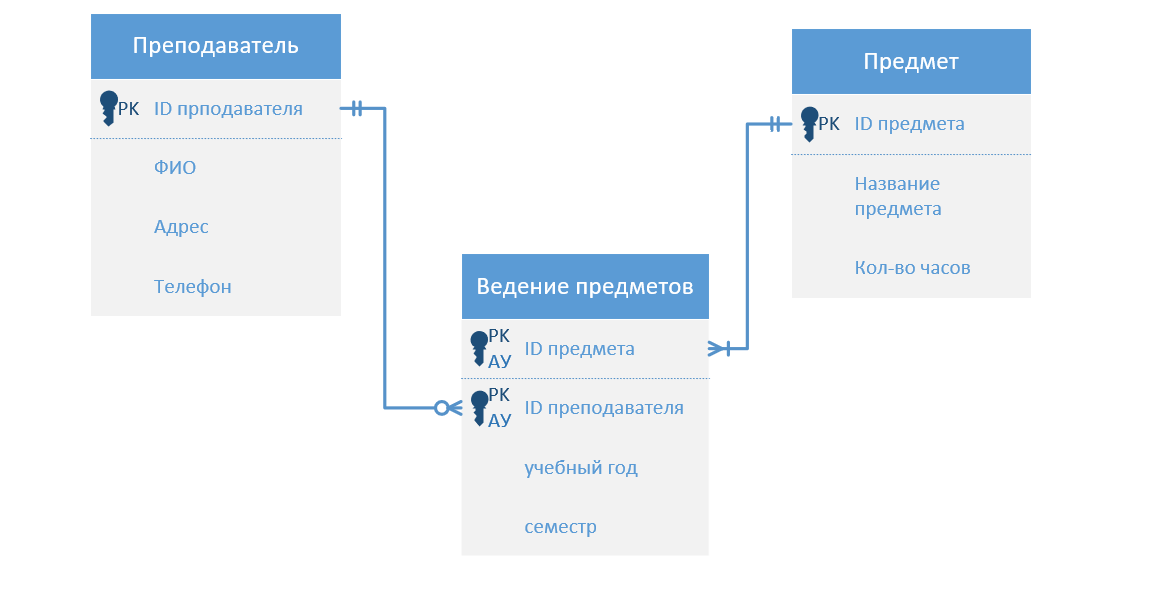
Построить ER-модель для следующего фрагмента: **предмет** могут вести несколько **преподавателей**, и каждый преподаватель может вести несколько предметов. Выделить сущности, определить атрибуты сущностей, установить связи.

**Решение:**

1. Сущности: Преподаватель, Предмет.
2. Атрибуты:
   1. Преподаватель(\*Номер преподавателя, ФИО, Должность, Адрес, Телефон)
   2. Предмет(\*Номер предмета, Название предмета, Кол-во часов)
3. Связи: т.к. один преподаватель может вести несколько предметов и каждый предмет могут вести несколько преподавателей, то *связь много-ко-многим.*
4. Построим ER-модель.



1. Строим реляционную модель:

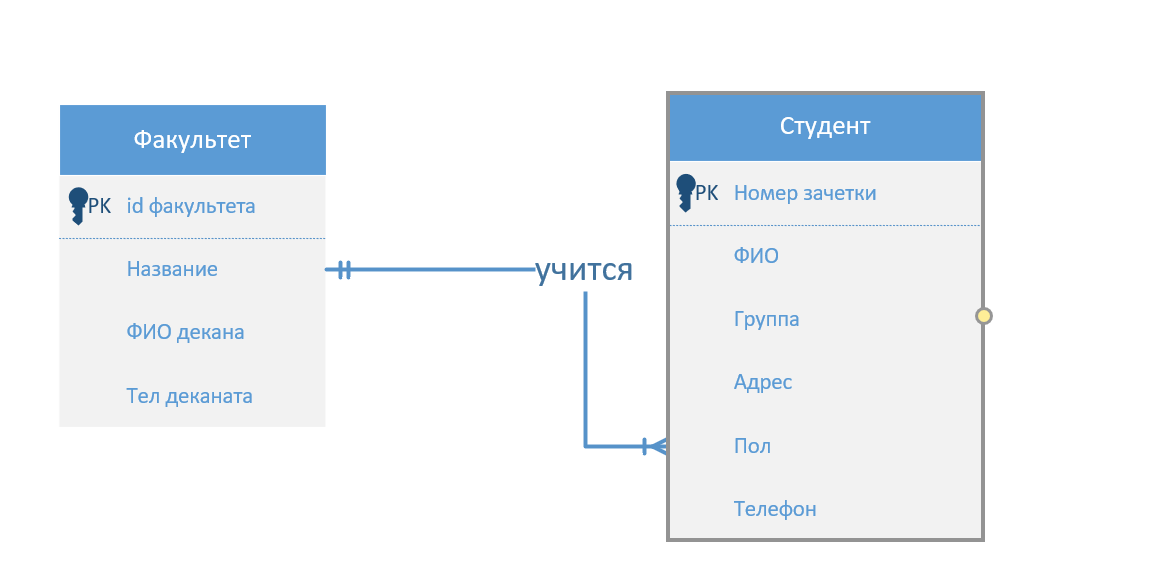


**Пример 2**

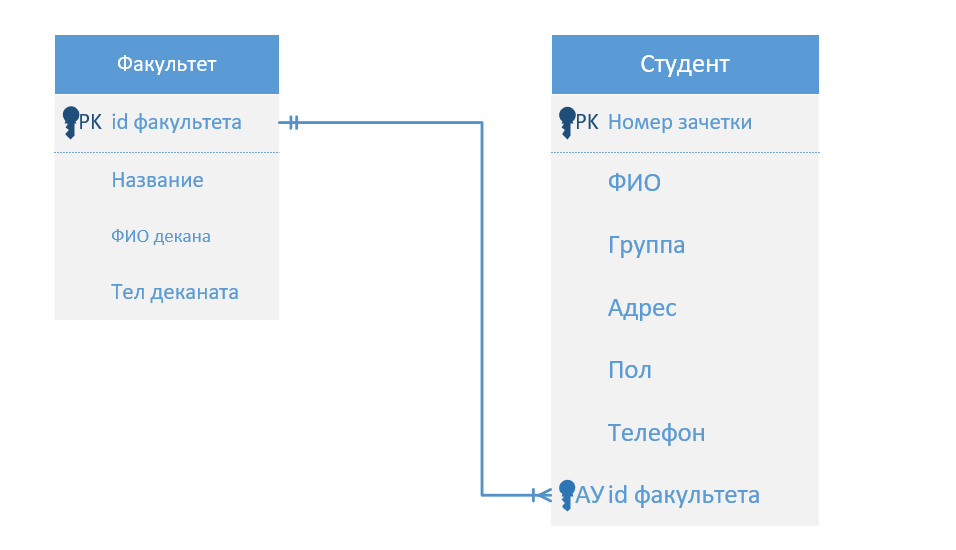
Построить ER-модель для следующего фрагмента: на каждом факультете учится несколько студентов, но каждый студент учится на одном факультете. Выделить сущности, определить атрибуты сущностей, установить связи.

**Решение:**

1. Сущности: Факультет, Студент
2. Атрибуты сущностей:
   1. Факультет(\*Номер факультета, Название факультета, ФИО декана, Тел деканата)
   2. Студент(\*Номер зач книжки, ФИО студента, Группа, Адрес, Тел)
3. Связь: со стороны факультета связь «обязательная один», со стороны студента связь «обязательная много».
4. Строим ER-модель с помощью нотации Crow Foots:



1. Строим реляционную модель:



**Пример 3.**

Дано описание предметной области (Это *Описание заказчика)*

В информационной системе клуба любителей скачек должна быть представлена информация:

1. об ипподромах, на которых проводятся скачки
2. об участвующих в скачках лошадях (кличка, пол, возраст),
3. их владельцах (имя, адрес, телефон) и жокеях (имя, адрес, возраст, рейтинг).

Необходимо сформировать таблицы для хранения информации по каждому состязанию: дата, время и место проведения скачек (ипподром), название состязаний (если таковое имеется), клички участвующих в заездах лошадей и имена жокеев, занятые ими места и показанное в заезде время.

**Требуется:**

1. Разработать ER-модель предметной области;
2. Преобразовать ER-модель в реляционную базу данных. Разработать концептуальную схему базы данных.
3. Какие операции реляционной алгебры нужно выполнить, чтобы получить следующую информацию:

а) показать список лошадей, участвующих в скачках в заданном интервале дат;

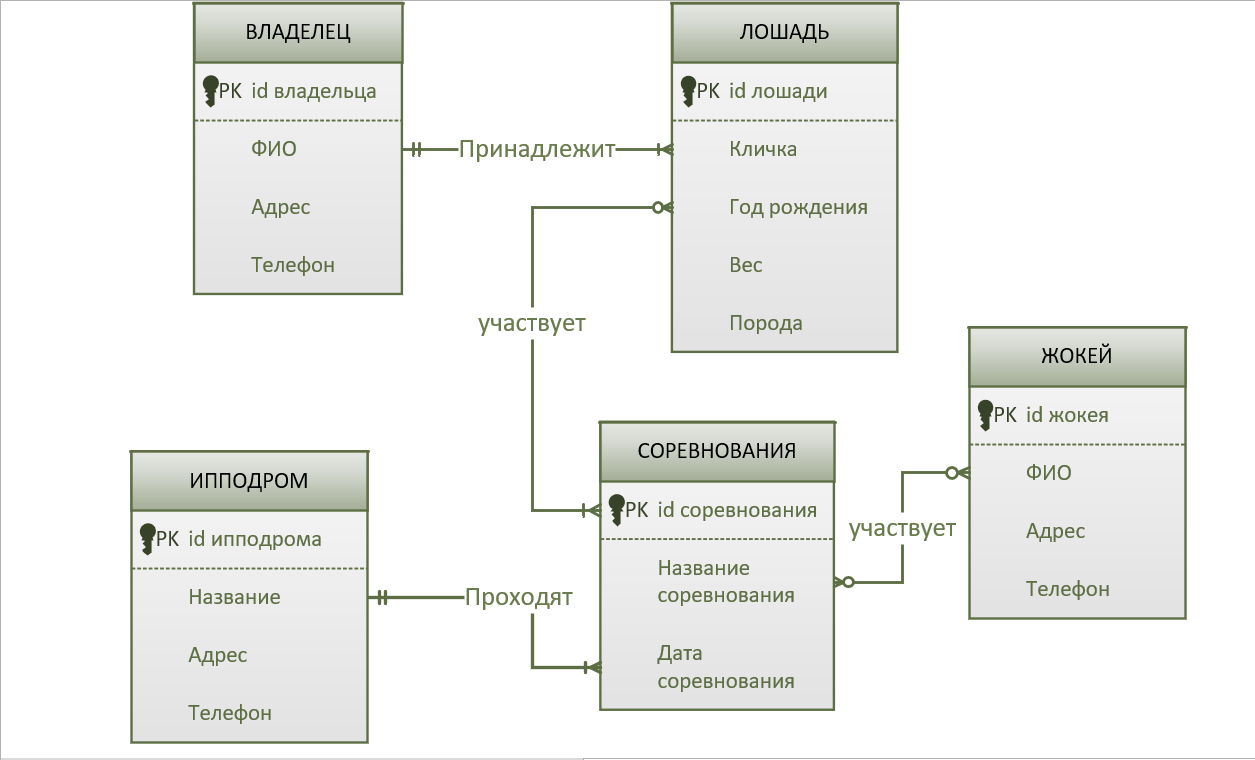
б) показать список жокеев, участвующих в скачках в заданном интервале дат;

в) вывести информацию о победителях заданного состязания: жокей и лошадь;

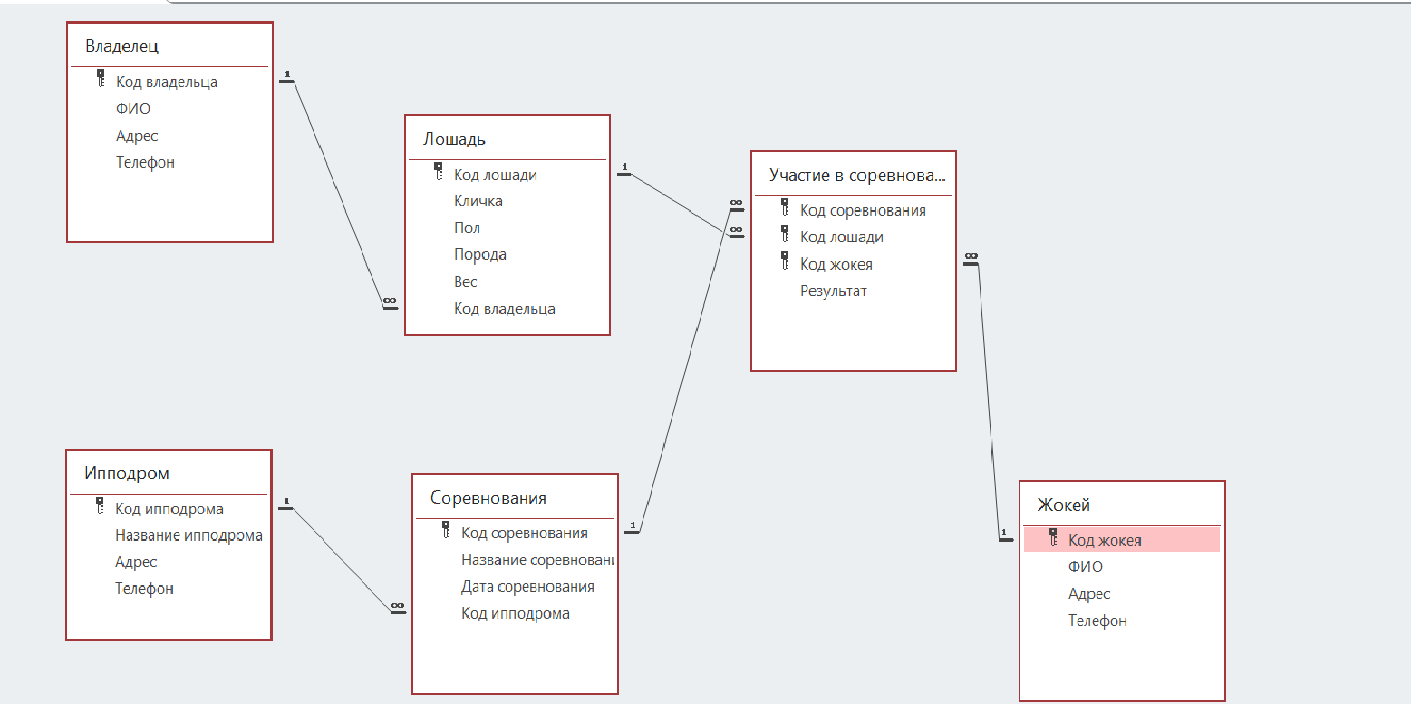
г) вывести информацию о количестве лошадей у каждого владельца.

**Решение:**

1. Выделяем сущности:
   1. Ипподром
   2. Лошадь
   3. Скачки(состязация)
   4. Владелец
   5. Жокей
2. Атрибуты сущностей:
   1. Ипподром(\*ID ипподрома, Название, Адрес, Тел)
   2. Лошадь(\*ID лошали, Кличка, Год рождения, Вес)
   3. Владелец(\*ID владельца, ФИО, Адрес, Тел)
   4. Скачки(ID состязания, Название, Дата начала)
   5. Жокей(\*ID жокея, ФИО, Адрес, Тел)
3. Связи:
   1. Владелец может иметь несколько Лошадей, но каждая Лошадь имеет только одного Владельца, связь со стороны Владельца «необязательная один». Со стороны Лошади – «обязательная много».
   2. На одном Ипподроме может быть несколько Состязаний(Скачек), но каждое Состязание проводится только на одном Ипподроме. Со стороны Ипподрома связь «обязательная один», со стороны Состязания «обязательная много».
   3. Каждая Лошадь может участвовать в нескольких Состязаниях и в одном Состязании может участвовать несколько Лошадей. Связь «много-к-многим».
   4. Каждый Жокей может участвовать в нескольких Состязаниях и в каждом Состязании могут участвовать несколько Жокеев. Связь «много-к-многим».
4. Строим ER-модель:



1. Строим реляционную модель



Задание:

1. Какие операции реляционной алгебры нужно выполнить, чтобы получить следующую информацию:

а) показать список лошадей, участвующих в скачках в заданном интервале дат;

б) показать список жокеев, участвующих в скачках в заданном интервале дат;

в) вывести информацию о победителях заданного состязания: жокей и лошадь;

г) вывести информацию о количестве лошадей у каждого владельца.