Концепции ER - модели

- Модель "сущность-связь" (Entity-Relationship model, или ER-модель). ER- моделирование представляет собой нисходящий подход к проектированию базы данных, который начинается с выявления наиболее важных данных, называемых сущностями (entities), и связей (relationships) между данными, которые должны быть представлены в модели.

- Затем в модель вносятся дополнительные сведения, например, указывается информация о сущностях и связях, называемая атрибутами (attributes), а также все ограничения, относящиеся к сущностям, связям и атрибутам. ER-моделирование — это важный метод, которым должен владеть любой проектировщик базы данных; он составляет основу

Основные концепции модели «сущность - связь» включают **типы сущностей, типы связей и атрибуты**.

Тип сущности. Группа объектов с одинаковыми свойствами, которая рассматривается в конкретной предметной области как имеющая независимое существование.

Типы сущностей

- Объект или концепция, которые характеризуются на данном предприятии как имеющие независимое существование.
- Основной концепцией ER –моделирования является тип сущности (entity type), который представляет множество объектов реального мира с одинаковыми свойствами.
- Тип сущности характеризуется независимым существованием и может быть объектом с физическими (реальным) существованием или объектом с концептуальным (абстрактным) существованием.
- В настоящее время можно дать только одно рабочее определение типа сущности, поскольку для них пока не существует строгого формального определения.
- Это значит, что разные разработчики могут выделять разные сущности.

Примеры сущностей с физическим и концептуальным существованием

Физическое существование	Концептуальное существование
Работник	Осмотр объекта недвижимости
Объект недвижимости	Инспекция объекта недвижимости
Клиент	Продажа объекта недвижимости
Деталь	Рабочий стаж
Поставщик	
Изделие	

- **Экземпляр сущности**. Однозначно идентифицируемый объект, который относится к сущности определенного типа.

- Каждый однозначно идентифицируемый объект типа сущности, который относится к сущности определенного типа, называется просто **экземпляром сущности (entity occurrence)**.

- Каждый тип сущности обозначается именем и характеризуется списком свойств. База данных, как правило, содержит много разных типов сущностей.

Сущность. Экземпляр типа сущности, который может быть идентифицирован уникальным образом.

Каждый тип сущности идентифицируется именем и списком свойств.

Слабый тип сущности. Тип сущности, существование которого зависит от какого-то другого типа сущности.

Сильный тип сущности. Тип сущности, существование которого не зависит от какого-то другого типа сущности.

Способы представления сущности на диаграмме

Каждый тип сущности изображается в виде прямоугольника с именем сущности внутри него; в качестве имен]! обычно применяется существительное в единственном числе. В языке UML принято использовать прописные буквы в начале каждого слова, составляющего имя сущности (например. Staff и PropertyForRent).

Пример схематического изображения типов сущностей Staff и Branch.

Рис. *11.2,* *Схематическое* ***изображение*** *типов сущностей* ***Staff*** *и* ***Branch***

Атрибуты

- **Атрибут.** Свойства типа сущности или типа связи.
- Отдельные свойства сущностей называют атрибутами. Например, сущность Branch (отделение компании) может быть описана номером отделения (Branch_No), адресом (Address), номером телефона (Tel_No).
- Атрибуты сущности содержат значения, описывающие каждую сущность.
- Значения атрибутов представляют основную часть сведений, сохраняемых в базе данных.

- **Домен атрибута.** Набор значений, которые могут быть присвоены атрибуту.
- Различные атрибуты могут совместно использовать один и тот же домен. Например, атрибуты адреса (Address) сотрудников компании (сущность Stuff) и владельцев объектов недвижимости (сущность Owner) могут совместно использовать один и тот же домен всех возможных адресов.
- Домены также могут представлять собой комбинацию, состоящую из нескольких других доменов. Например, домен даты рождения (DOB) сущности Stuff состоит из таких подчиненных доменов, как день, месяц, год.

- **Простой атрибут.** Атрибут, состоящий из одного компонента с независимым существованием.
- Простые атрибуты не могут быть разделены на более мелкие компоненты. Пример: атрибут пола (Sex), зарплаты (Salary) работника.
- Простые атрибуты иногда называют атомарными

- **Составной атрибут.** Атрибут, состоящий из нескольких компонентов, каждый из которых характеризуется независимым существованием.
- Некоторые атрибуты могут быть разделены на более мелкие компоненты. Например, атрибут адреса (Address) сущности, представляющий отдельные компании (Branch) со значением “163 Main st, Partick, Glasgow, G11 9QX”, может быть разбит на отдельные атрибуты улицы (Street) со значением “163 Main st”, района (Area) со значением “Partick” и т.д.

- **Однозначный атрибут**. Атрибут, который содержит одно значение для одной сущности.
- Большинство атрибутов типов сущности являются однозначными для каждого отдельного экземпляра этой сущности. Например, сущность Branch всегда имеет единственное значение в атрибуте номера отделения компании (Branch_No), например “B7”.

- **Многозначный атрибут.** Атрибут, который содержит несколько значений для одной сущности.
- Некоторые атрибуты могут иметь несколько значений для одной сущности. Например, сущность Branch может иметь несколько значений для атрибута номера телефона отделения компании (Tel_No), например “89161111112” и “89161111113”.
- Многозначный атрибут допускает присутствие определенного количества значений (возможно в заданных пределах - максимальном и минимальном количестве).

- **Производный атрибут.** Атрибут, который представляет значение, производное от значения связанного с ним атрибута или некоторого множества атрибутов, принадлежащих некоторому (не обязательно данному) типу сущности.

- Некоторые атрибуты могут быть связаны с определенной сущностью. Например, значение атрибута duration (Срок действия) сущности Lease (Договор аренды) вычисляется на основе атрибутов rentstart (Начало срока аренды) и rentFinish (Конец срока аренды), которые также относятся к типу сущности Lease. Атрибут duration является производным атрибутом, значение которого вычисляется на основании значений атрибутов rentstart и rentFinish.

- Некоторые атрибуты могут быть связаны с определенной сущностью. Например, значение атрибута duration (Срок действия) сущности Lease (Договор аренды) вычисляется на основе атрибутов rentstart (Начало срока аренды) и rentFinish (Конец срока аренды), которые также относятся к типу сущности Lease. Атрибут duration является производным атрибутом, значение которого вычисляется на основании значений атрибутов rentstart и rentFinish.
- В некоторых случаях значение атрибута является производным от многих экземпляров сущности одного и того же типа. Например, атрибут totalStaff (Общее количество сотрудников) сущности типа staff может быть вычислен на основе подсчета общего количества экземпляров сущности staff.
- Производные атрибуты могут также создаваться в форме ассоциаций атрибутов сущностей различных типов. Например, рассмотрим атрибут deposit (Задаток) сущности типа Lease. Значение атрибута deposit рассчитывается как удвоенное значение ежемесячной платы за аренду данного объекта недвижимо- сти. Следовательно, значение атрибута deposit сущности Lease является про- изводным от атрибута rent сущности типа PropertyForRent.

Ключи

- Под ключом подразумевается элемент данных, который позволяет, уникально идентифицировать отдельные экземпляры типа сущности.

- **Потенциальный ключ.** Атрибут или набор атрибутов, который уникально идентифицирует отдельные экземпляры типа сущности.

- Потенциальный ключ — это один или несколько атрибутов, значения которых однозначно идентифицируют каждый экземпляр сущности данного типа. Например, номер отделения компании (branchNo) является потенциальным ключом для сущности Branch, поскольку он содержит разные значения для каждого отдельного экземпляра сущности Branch. Потенциальный ключ должен содержать значения, которые уникальны для каждого отдельного экземпляра сущности данного типа. Это означает, что потенциальный ключ не может содержать значения NULL (см. раздел 3.2). В частности, каждое отделение компании имеет уникальный номер (например, ' ВООЗ '), и не существует отделений с одинаковыми номерами.

- **Первичный ключ.** Потенциальный ключ, который выбран для однозначной идентификации каждого экземпляра сущности определенного типа.

- Тип сущности может иметь несколько потенциальных ключей. Например, каждый сотрудник может иметь уникальный табельный номер staf fKo, а также уникальный номер карточки государственного социального страхования (National Insurance Number — NIN), который используется государственными органами. Таким образом, сущность Staff обладает двумя потенциальными ключами, каждый из которых может быть выбран в качестве первичного ключа.

- Выбор первичного ключа сущности осуществляется с учетом суммарной длины атрибутов, минимального количества необходимых атрибутов в ключе, а также наличия гарантий уникальности его значений в текущий момент времени и в обозримом будущем. В частности, табельный номер сотрудника (например, ' SG14 ') меньше по размеру, а потому удобнее в работе, чем номер карточки социального страхования (например, ' WL2206S8D1). Следовательно, первичным ключом сущности Staff целесообразно выбрать именно атрибут staff No, а не атрибут NIN, который в этом случае рассматривается как альтернативный ключ.

- **Составной ключ.** Потенциальный ключ, который состоит из двух или нескольких атрибутов. В некоторых случаях ключ сущности состоит из нескольких атрибутов, значения которых, взятые вместе, а не по отдельности, уникальны для каждого экземпляра сущности.

- Например, сущность Advert (Рекламное объявление) обладает атрибутами propertyNo, newspaperName, dateAdvert и cost.

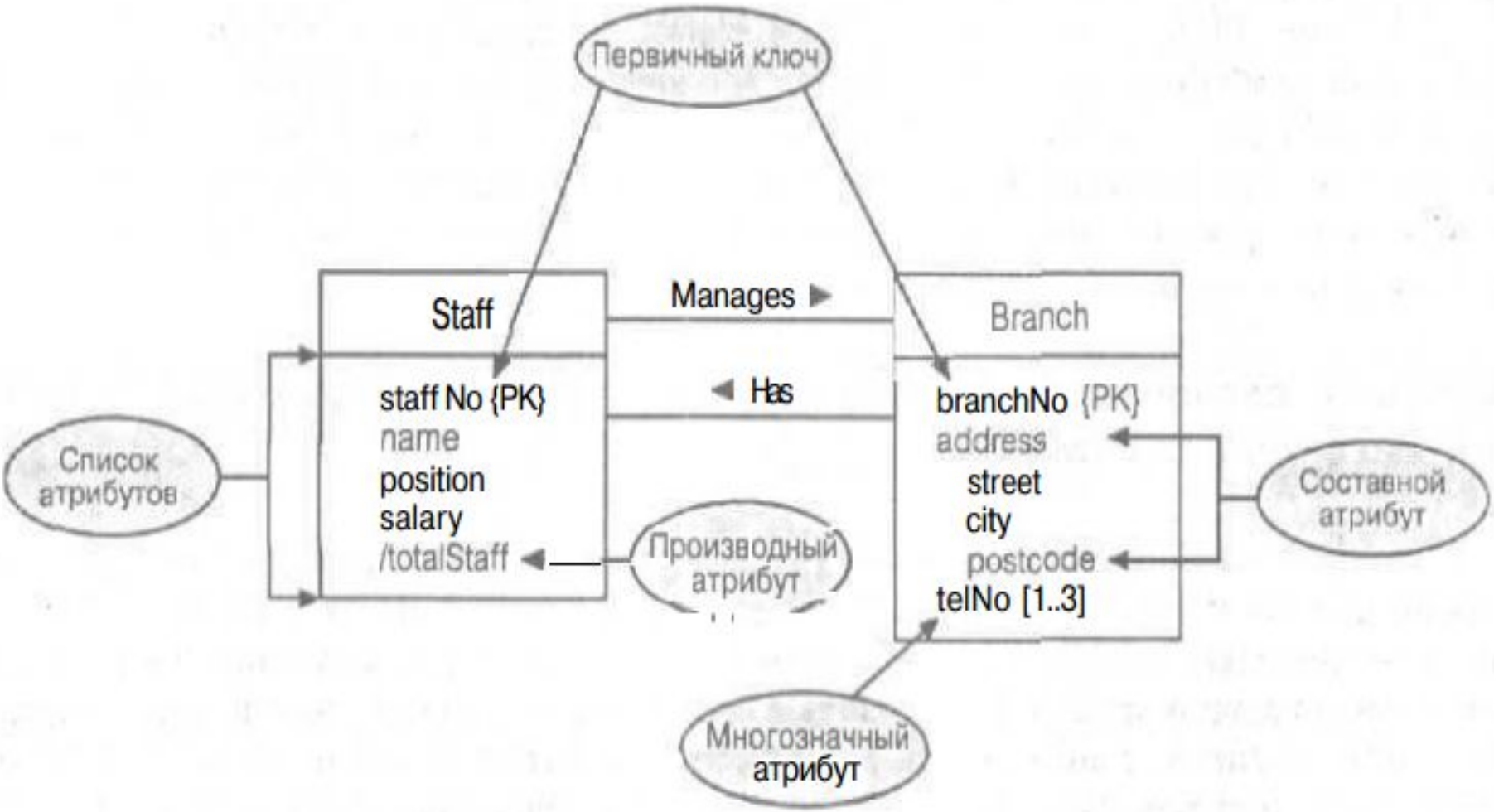
- Многие объекты недвижимости одновременно рекламируются в нескольких газетах за то же число. Для уникальной идентификации каждого рекламного объявления необходимо использовать значения propertyNo, newspaperName (Название газеты) и dateAdvert (Дата рекламного объявления). Таким образом, сущность Advert (Рекламное объявление) обладает составным первичным ключом, состоящим из атрибутов propertyNo,

Представление атрибутов на диаграммах

- Если сущность определенного типа должна отображаться на схеме вместе со своими атрибутами, то прямоугольник, представляющий эту сущность, делится на две части. В верхней части прямоугольника отображается имя сущности, а в нижней — список имен атрибутов.
- Первым атрибутом (атрибутами) в списке должен быть первичный ключ для сущности данного типа, если он известен. Имя (имена) атрибута (атрибутов) первичного ключа должно быть обозначено дескриптором {PK} (сокращение от primary key).

- В языке UML принято присваивать атрибуту имя, которое начинается со строчной буквы, а если оно состоит из нескольких слов, то первая буква каждого следующего слова пишется с прописной буквы (например, Address и Tel_No), кроме того, на схемах могут применяться дополнительные дескрипторы, в том числе дескриптор с обозначением компонента первичного ключа {PPK} (сокращение от partial primary key), если атрибут образует часть составного первичного ключа, и дескриптор с обозначением альтернативного ключа {АК} (сокращение от alternate key).

- Как показано, первичным ключом сущности типа Staff является атрибут Staff_No, а первичным ключом сущности типа Branch — атрибут Branch_No


Первичный ключ
Staff
Manages
Branch
staff No {PK}
name
position
salary
/totalStaff
Has
branchNo {PK}
address
street
city
postcode
telNo [1..3]
Список атрибутов
Производный атрибут
Составной атрибут
Многозначный атрибут

- В некоторых наиболее простых приложениях баз данных на ER-диаграмме могут быть показаны все атрибуты сущности каждого типа. Но в случае более сложных приложений баз данных на этих схемах отображается только один или несколько атрибутов, которые образуют первичный ключ сущности каждого типа. Если на ER-диаграмме отображаются только атрибуты первичного ключа, дескриптор {PK} на схеме можно не показывать.

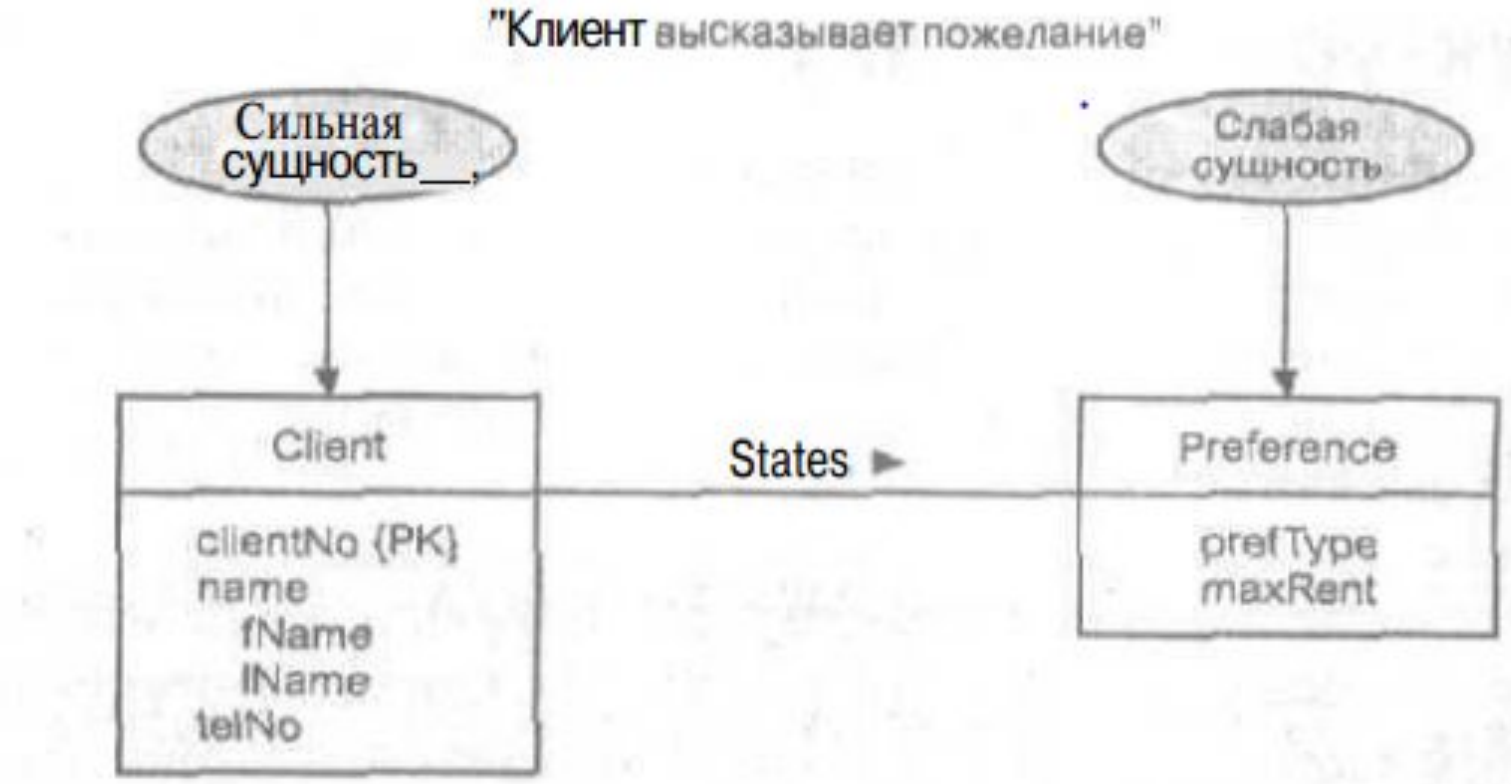
- Для простых однозначных атрибутов дескрипторы не требуются, и поэтому их имена просто отображаются в виде списка под именем сущности. А за сложным атрибутом следует список составляющих его простых атрибутов, обозначенный отступом вправо.

- Например, на схеме показан составной атрибут address сущности Branch, за которым следуют имена атрибутов, входящих в его состав, а именно street, city и postcode. Рядом с именами многозначных атрибутов ставится обозначение диапазона возможных значений этого атрибута. Например, если атрибут Tel_No обозначен диапазоном [1. .*] , это означает, что атрибут Tel_No может принимать одно или несколько значений.
- Если точно известно максимальное количество значений, можно обозначить атрибут с указанием точного диапазона. Например, если атрибут Tel_No не может содержать более трех значений, то он обозначается с помощью диапазона [1, . 3]. Производные атрибуты отмечаются префиксом в виде косой черты (/).

Атрибуты связей

- Рассмотрим в качестве примера связь Advertises между сущностями Newspaper и PropertyForRent.
- Допустим, что нужно зафиксировать в базе данных дату публикации рекламы арендуемой недвижимости и стоимость аренды, указанную в этой рекламе. Для этого лучше всего ассоциировать такую информацию со связью Advertises с помощью атрибутов dateAdvert и cost, а не вводить эти атрибуты в состав определений сущностей Newspaper или PropertyForRent.

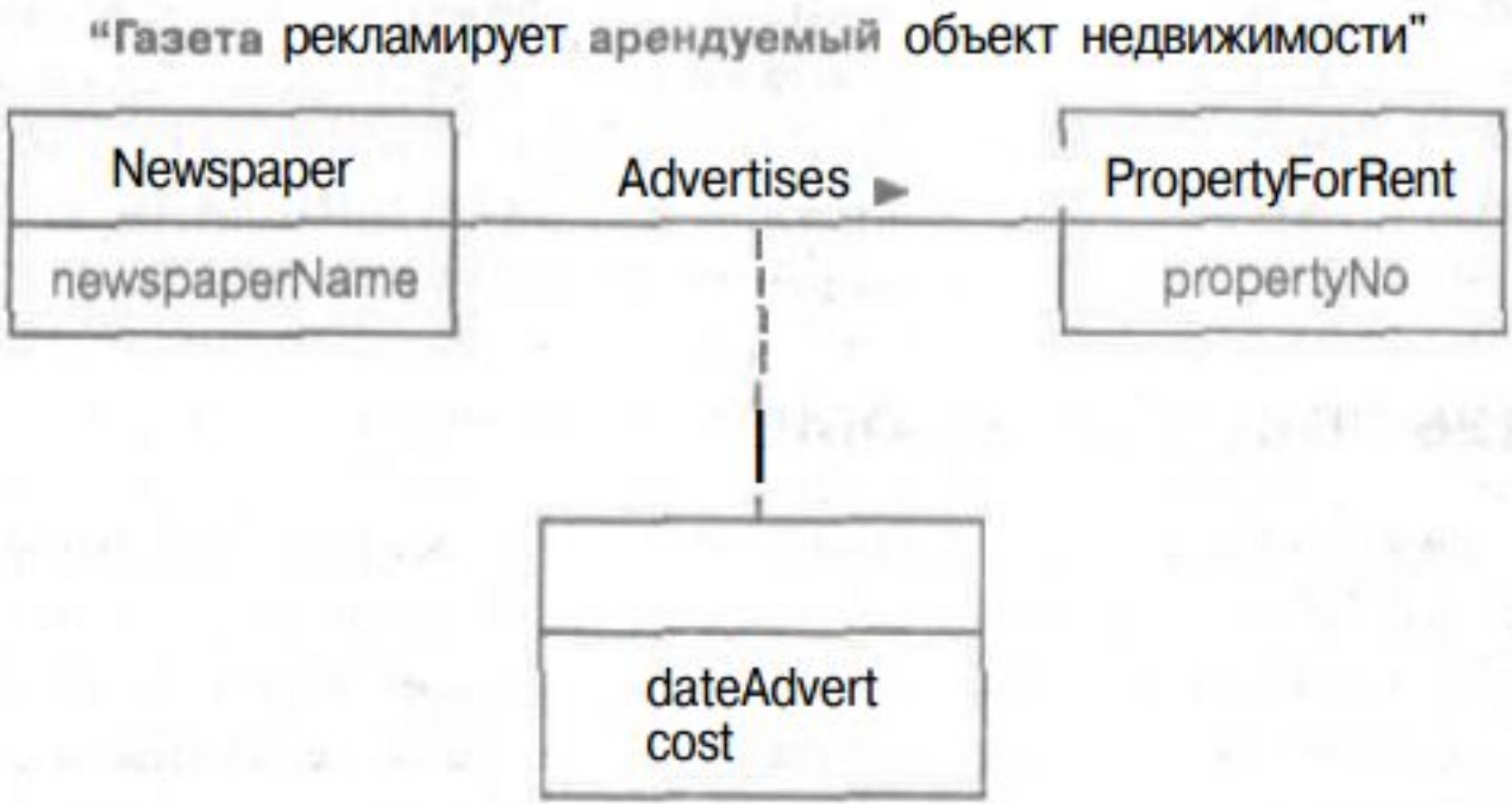
Рисунок 1.1


"Клиент высказывает пожелание"
Сильная сущность
Слабая сущность
Client
clientNo {PK}
name
fName
lName
telNo
States
Preference
prefType
maxRent

Схематическое представление атрибутов связей

- Для отображения атрибутов, относящихся к типу связи, применяется такое же условное обозначение, как и для типа сущности. Тем не менее, чтобы подчеркнуть различие между сущностью и связью, обладающей атрибутом, линия, которая соединяет прямоугольник с именем атрибута (атрибутов) и саму связь, отображается как штриховая.
- Например, на рис. 1.1 показана связь Advertises с атрибутами dateAdvert и cost. В качестве еще одного примера можно указать показанную на рис. 1.2 связь Manages с атрибутами mgrStartDate и bonus

Рисунок 1.2


"Газета рекламирует арендуемый объект недвижимости"
Newspaper
newspaperName
Advertises
PropertyForRent
propertyNo
dateAdvert
cost

- Если на схеме появляется связь с одним или несколькими атрибутами, это может свидетельствовать о том, что за этой связью скрывается не выявленный тип сущности. Например, наличие атрибутов dateAdvert и cost у связи Advertises свидетельствует о наличии скрытой сущности Advert (Рекламное объявление).

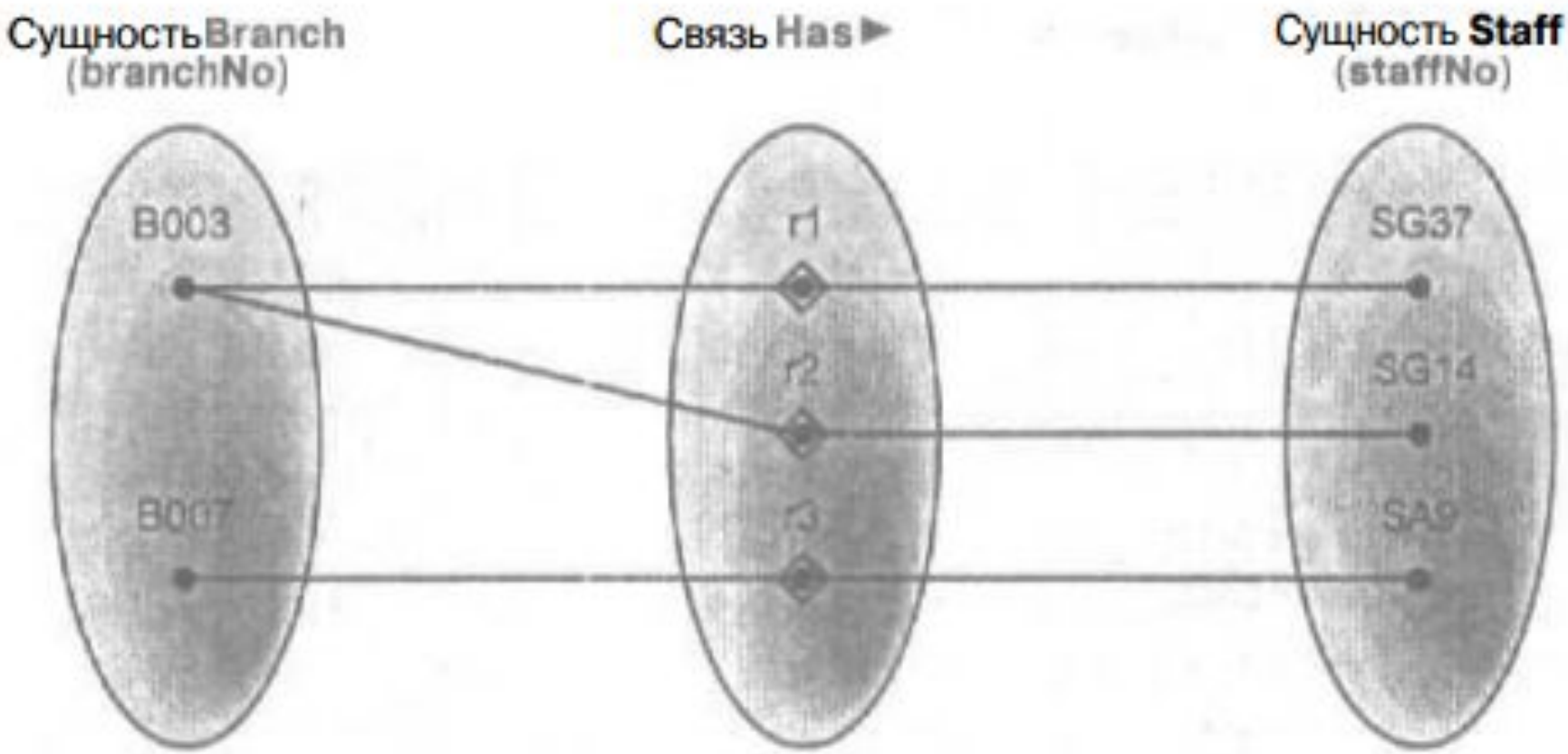
Типы связей

- **Тип связи**. Набор осмысленных ассоциаций между сущностями разных типов. Тип связи (relationship type) является набором ассоциаций между одним (или несколькими) типами сущностей, участвующими в этой связи.
- Каждому типу связи присваивается имя, которое должно описывать его назначение. В качестве примера типа связи можно указать связь POwns (Владеет недвижимостью) между сущностями PrivateOwner (Владелец недвижимости) и Proper tyForRent (Объект недвижимости).
- Как и при использовании понятий сущности и типа сущности, необходимо различать понятия "экземпляр связи" и "тип связи"

- **Экземпляр связи.** Однозначно идентифицируемая ассоциация, которая включает по одному экземпляру сущности из каждого участвующего в связи типа сущности – участников.

- Рассмотрим тип связи Has, который представляет ассоциацию между сущностями Branch (Отделение) и Staff (Персонал), иными словами, ассоциацию Branch Has Staff (Отделение имеет персонал). Каждый экземпляр связи Has устанавливает соответствие одного экземпляра сущности Branch с одним экземпляром сущности Staff. Для изучения примеров отдельных экземпляров связи Has может применяться так называемая семантическая сеть.
- Семантическая сеть представляет собой модель объектного уровня, в которой применяются символ • для изображения сущностей и символ Ф для изображения связей.

Семантическая сеть с изображением отдельных экземпляров связи, типа Has



Схематическое изображение связи, типа Branch Has Staff

"В отделении имеется персонал"

- Каждая связь описывает соответствие между одним экземпляром сущности Branch и одним экземпляром сущности Staff. Связи отображаются линиями, соединяющими каждый рассматриваемый объект Branch с соответствующим ему объектом Staff. Например, связь rl показывает соответствие между сущностью BOO3 типа Branch и сущностью SG37 типа Staff.

Схематическое изображение типов связей

- Каждый тип связи изображается в виде линии, соединяющей соответствующие типы сущностей и обозначенные именем этой связи. Обычно для обозначения имени связи принято использовать глагол (например, Supervises (Контролирует) или Manages (Управляет)) или короткую фразу, содержащую глагол (например, LeasedBy (Взят в аренду)). И в этом случае первая буква каждого слова в имени связи являются прописной. По возможности в каждой конкретной ER-модели все имена связей должны быть уникальными.
- На схеме должно быть показано направление действия каждой связи, поскольку обычно имеет смысл только одно направление этой связи (например, связь Branch Has Staff (Отделение имеет персонал) имеет больше смысла, чем Staff Has Branch (Персонал имеет отделение)). Поэтому после выбора имени связи рядом с этим именем на схеме размещается стрелка, которая показывает направление ее действия, чтобы читатель мог правильно интерпретировать имя связи (например. Branch Has > Staff).

Степень типа связи

- **Степень типа связи**. Количество типов сущностей, которые охвачены данной связью. - Сущности, охваченные некоторой связью, называются участниками этой связи.
- Количество участников связи определенного типа называется степенью (degree) этой связи. Следовательно, степень связи указывает количество типов сущностей, охваченных данной связью.
- Связь со степенью два называется двух- сторонней (binary). Примером двухсторонней связи является связь Has, показанная на предидущем рисунке, в которой участвуют сущности двух типов, а именно Staff и Branch.
- Вторым примером двухсторонней связи является показанная на рис. 11.5 связь POwns с двумя участвующими типами сущностей — PrivateOwner и

Пример двухсторонней связи POwns

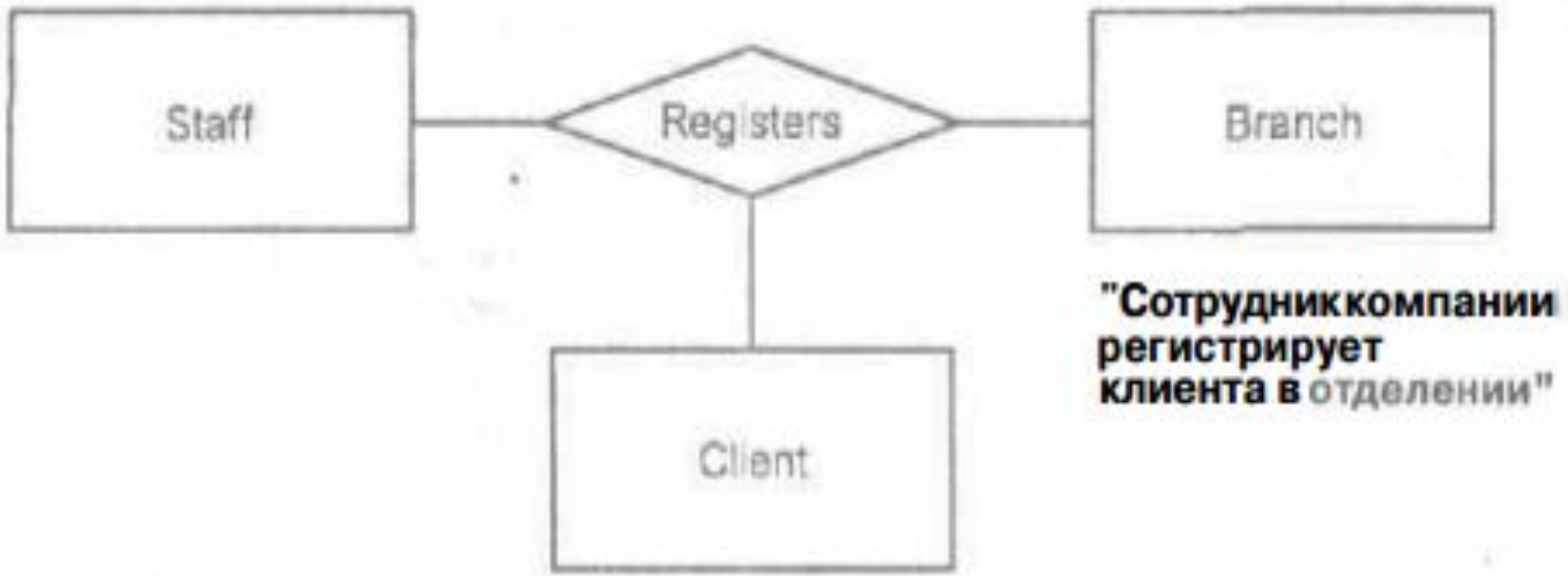
"Владелец недвижимости владеет арендуемым объектом недвижимости"

- Связь со степенью три называется трехсторонней (ternary). Примером трех-сторонней связи является связь Registers (Регистрирует) с тремя участвующими типами сущностей, а именно Staff, Branch и Client. Эта связь представляет процесс регистрации клиента представителем персонала отделения. Для описания связей со степенью больше двух принято применять термин сложная связь.

Схематическое изображение сложных связей

- В системе обозначений UML для обозначения связей со степенями больше двух применяются ромбы. Имя связи записывается внутри ромба, и в этом случае направленная стрелка, которая обычно применяется вместе с именем, не предусмотрена.

Пример трехсторонней связи Registers



- Связь со степенью четыре называется четырехсторонней (quaternary). Поскольку на одном рисунке нельзя найти пример такой связи, в качестве иллюстрации на следующем рисунке показана четырехсторонняя связь Arranges с четырьмя участвующими типами сущностей, а именно Buyer (Покупатель), Solicitor (Доверенное лицо), FinancialInstitution (Финансовый -орган) и Bid (Сделка). Эта связь представляет ситуацию, в которой покупатель, консультируемый доверенным лицом и поддерживаемый финансовым органом, заключает сделку.

Рекурсивная связь

- **Рекурсивная связь**. Связь, в которой одни и те же сущности участвуют несколько, раз в разных ролях.
- Рассмотрим рекурсивную связь Supervises, которая представляет взаимосвязь персонала с инспектором, также входящим в состав персонала.
- Иначе говоря, сущность Staff участвует в связи Supervises дважды: первый раз — в качестве инспектора, а второй — в качестве сотрудника, которым управляют. Рекурсивные связи иногда называются односторонними (unary).

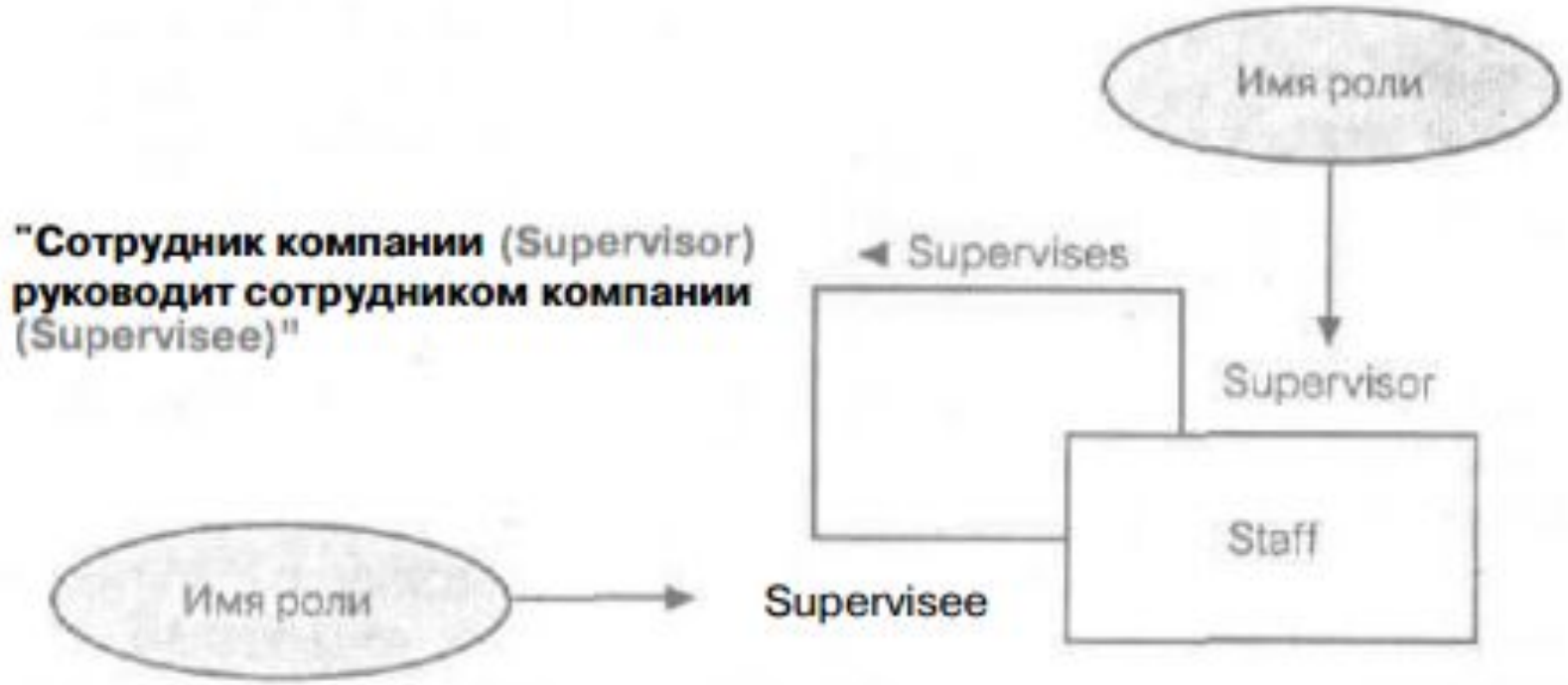
- Связям могут присваиваться ролевые имена для указания назначения каждой сущности, участвующей в данной связи. Ролевые имена имеют большое значение в рекурсивных связях, поскольку позволяют определить функции каждого участника. На рисунке показан пример использования ролевых имен для описания рекурсивной связи Supervises.
- Первый участник связи Supervises с типом сущности Staff получил имя Supervisor (Инспектор), а второй — Supervisee (Контролируемый сотрудник).

- Ролевые имена могут также использоваться, когда две сущности связаны несколькими связями. Например, сущности Staff и Branch связаны двумя раз- личными связями — Manages и Has. Как показано на рисунке, использование ролевых имен существенно проясняет назначение каждой связи. Например, что касается связи Staff Manages Branch, то в ней один из представителей персонала (сущности Staff) с ролевым именем Manager (Менеджер) управляет отделением компании (сущность Branch), которому присвоено ролевое имя Branch Offic e (Отделение компании).
- В случае связи Branch Has Staff отделение с ролевым именем Branch Office имеет персонал, представителям которого присвоено ролевое имя Member of Staff (Член персонала).

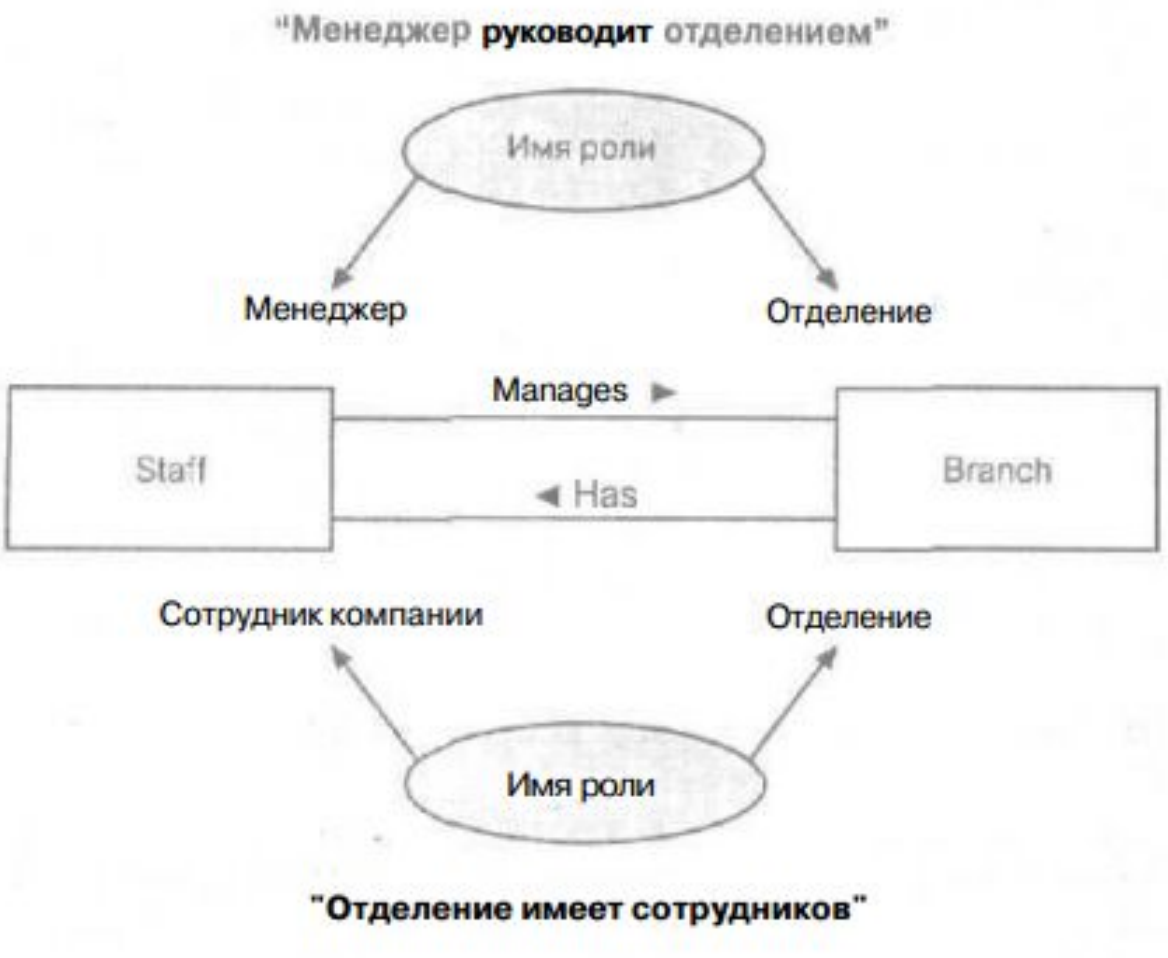
Пример четырехсторонней связи Arranges

"Покупатель, **консультируемый доверенным лицом и поддерживаемый финансовым органом,** заключает сделку"

Пример рекурсивной связи Supervises с ролевыми именами Supervisor и Supervisee



Пример сущностей, связанных двумя различными связями Manages и Has, с указанием ролевых имен



СПАСИБО ЗА ВНИМАНИНЕ!