

**Диаграммы
"сущность-связь"
(ER-модели) в нотации
IDEF1X**

Основные понятия ER-модели Entity-Relationship (Сущность-Связи)

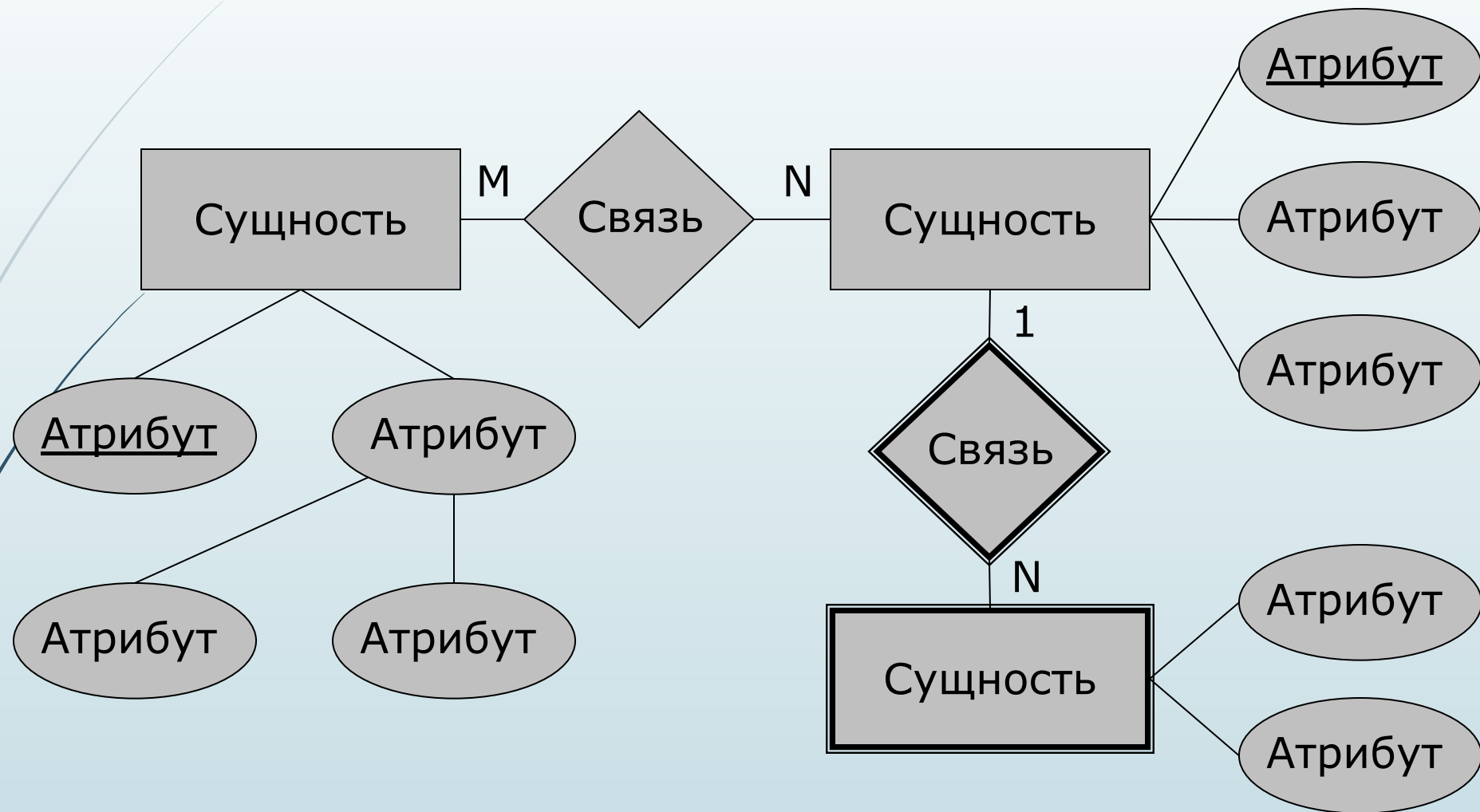


Большинство современных подходов к проектированию баз данных основано на использовании разновидностей ER-модели.

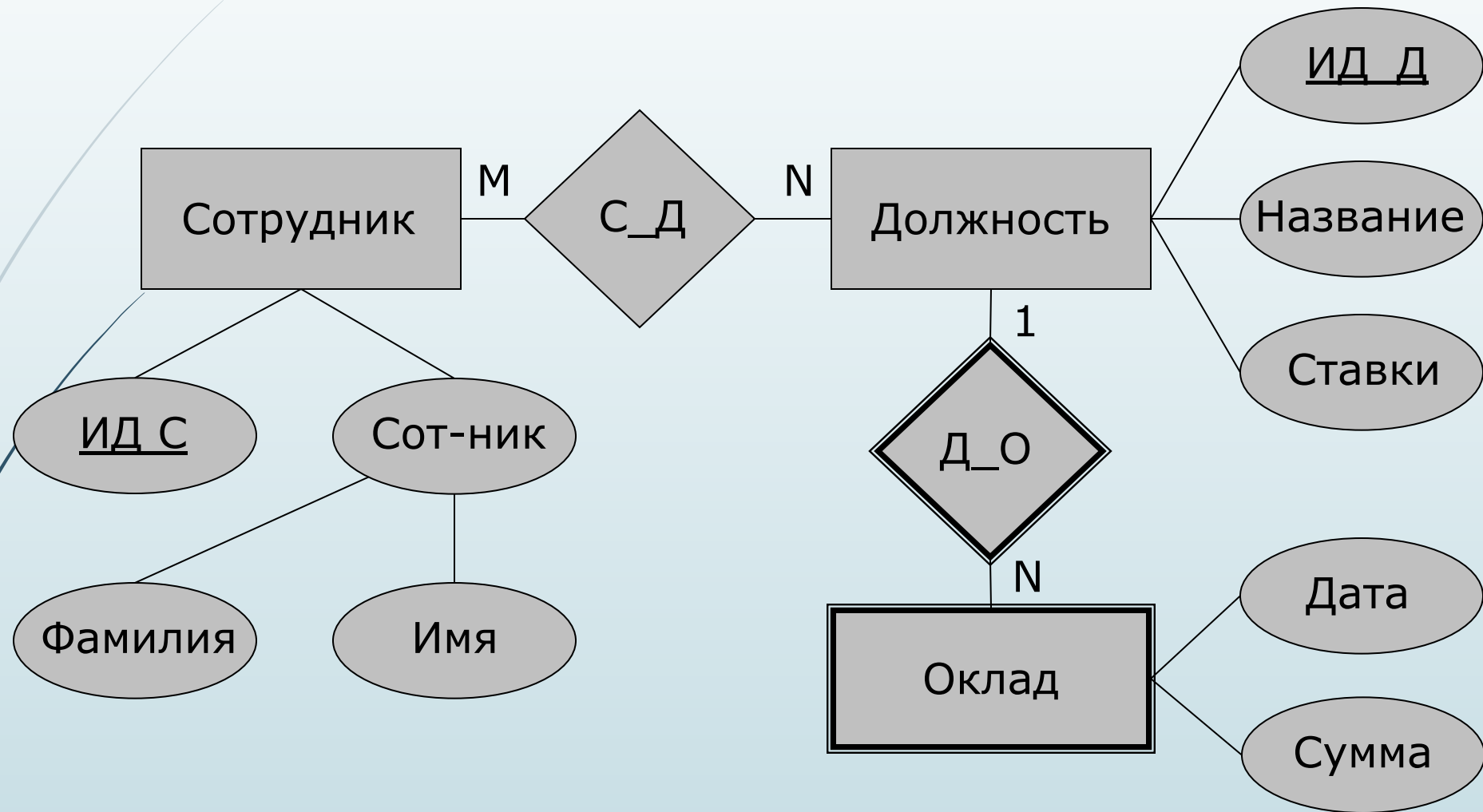


Модель была предложена Питером Ченом в 1976 г. Моделирование предметной области базируется на использовании **графических диаграмм**, включающих небольшое число разнородных компонентов. В связи с наглядностью представления концептуальных схем баз данных ER-модели получили широкое распространение в системах **CASE**, поддерживающих автоматизированное проектирование реляционных баз данных, в частности **IDEF (1x)**.

Диаграмма «Сущность-связь» в нотации Питера Чена



Пример



Основные понятия модели Entity-Relationship (Сущность-Связи)



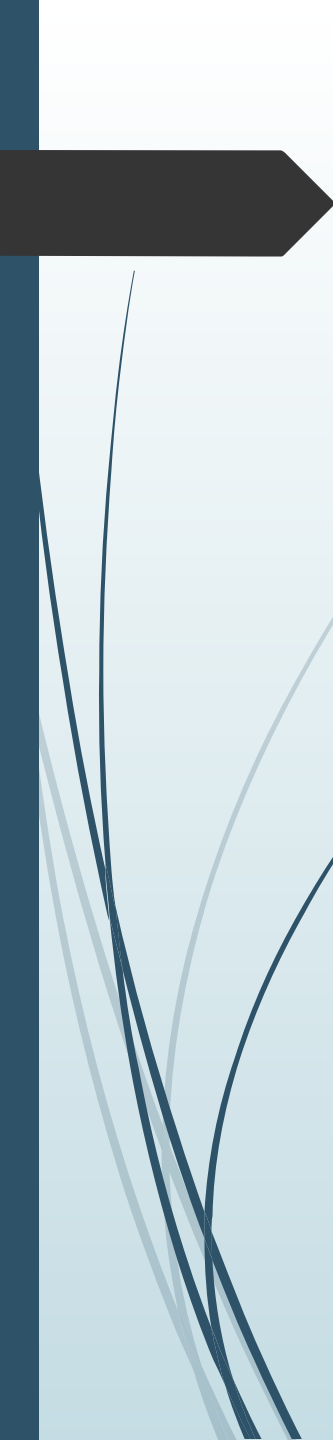
Основными понятиями ER-модели являются сущность, связь и атрибут.



Сущность - это реальный или представляемый объект, информация о котором должна сохраняться и быть доступна. В диаграммах ER-модели сущность представляется в виде прямоугольника, содержащего имя сущности. При этом имя сущности - это имя типа, а не некоторого конкретного экземпляра этого типа.

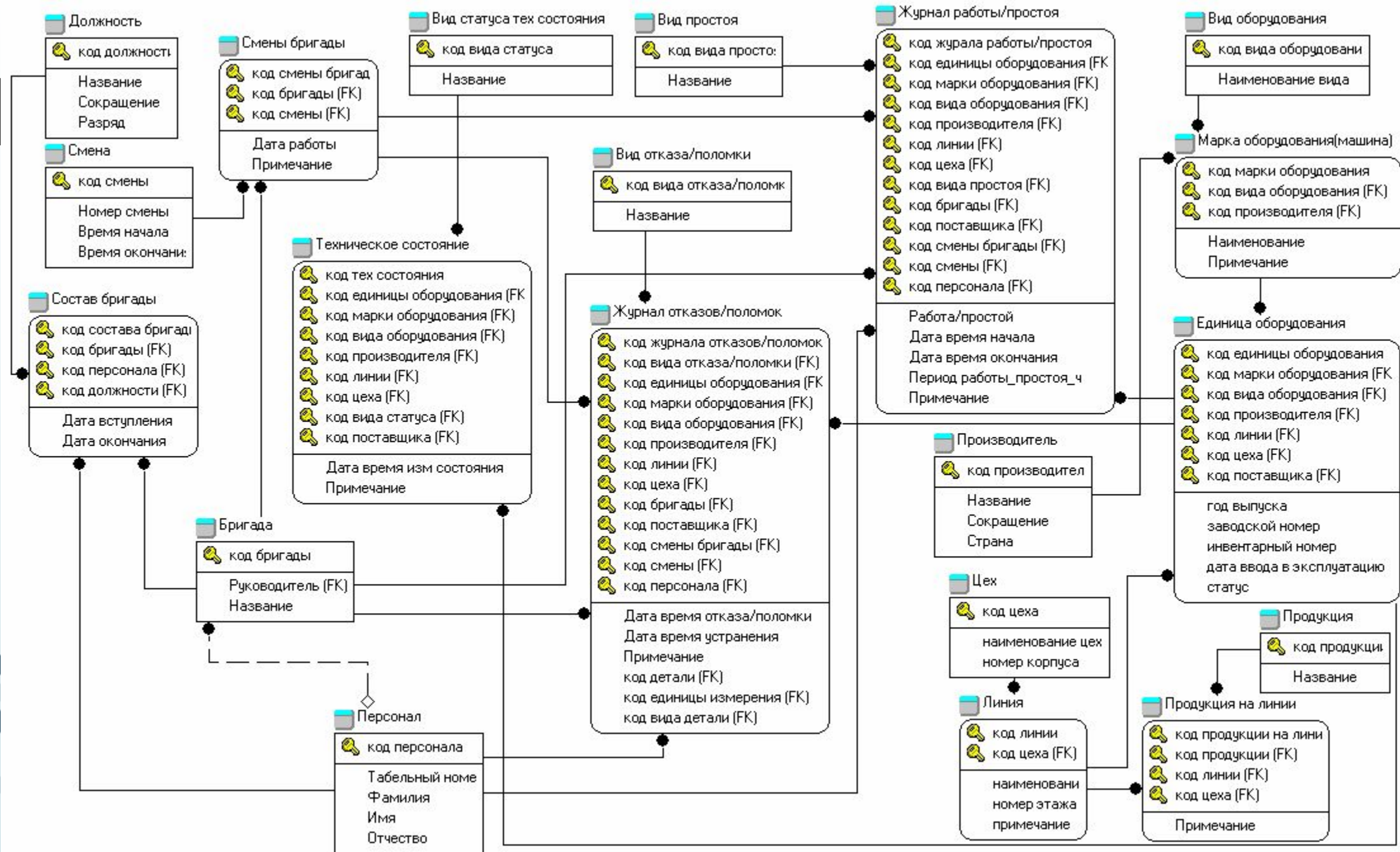


Каждый экземпляр сущности должен быть отличим от любого другого экземпляра той же сущности.



Метод IDEF1X (Data Modeling – метод моделирования баз данных) используется для разработки реляционных баз данных и использует условный синтаксис, специально разработанный для удобного построения концептуальной схемы.

- Концептуальной схемой мы называем универсальное представление структуры данных в рамках коммерческого предприятия, независимое от конечной реализации базы данных и аппаратной платформы. Будучи статическим методом разработки, IDEF1X изначально не предназначен для динамического анализа, тем не менее, он иногда применяется в этом качестве, как альтернатива методу IDEF1.
- Использование метода IDEF1X наиболее целесообразно для построения логической структуры базы данных после того, как все информационные ресурсы исследованы и решение о внедрении реляционной базы данных, как части корпоративной информационной системы, было принято. Однако не стоит забывать, что средства моделирования IDEF1X специально разработаны для построения реляционных информационных систем, и если существует необходимость проектирования другой системы, скажем объектно-ориентированной, то лучше избрать другие методы моделирования.

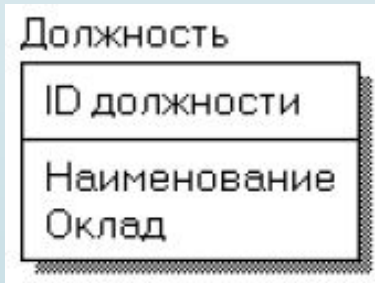


Сущность, атрибут

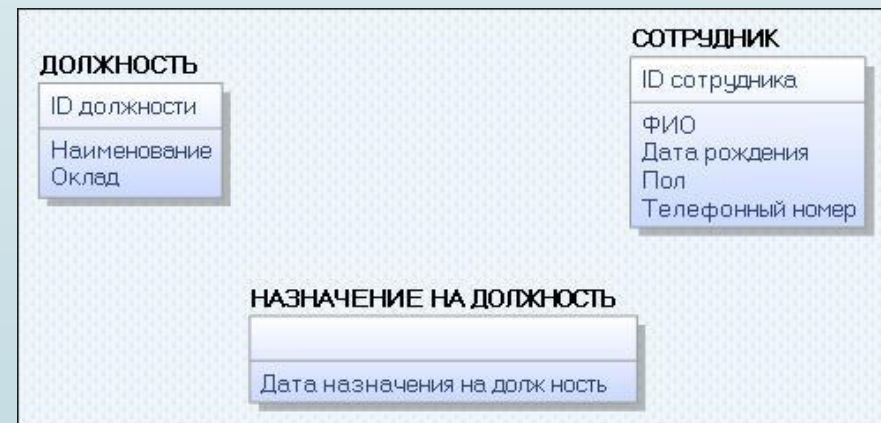
Сущность – это объект, который может быть идентифицирован некоторым способом, отличающим его от других объектов. Каждая сущность обладает набором атрибутов.

Атрибут – отдельная характеристика сущности.

Сущность состоит из экземпляров, каждый из которых должен отличаться от другого экземпляра. Пример: сущность – «Город», экземпляры сущности «Город» – Москва, Пенза,



Сущности с атрибутами



Ключ



Ключ - минимальный набор атрибутов, по значениям которых можно однозначно найти требуемый экземпляр сущности.

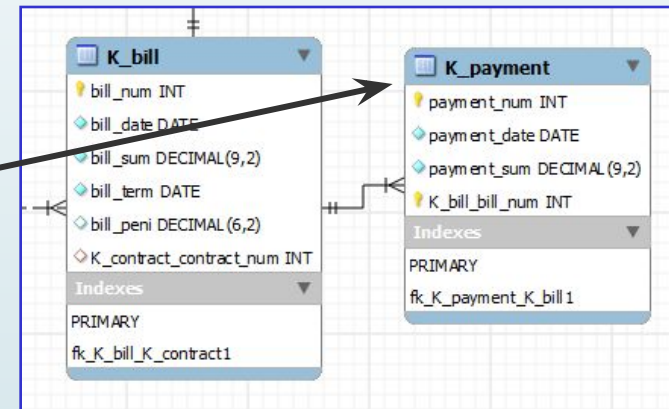
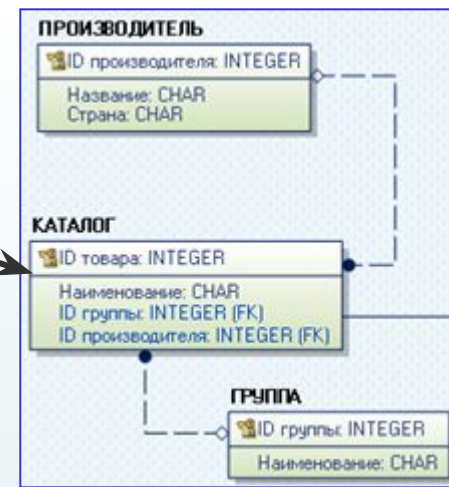


Первичный ключ сущности позволяет идентифицировать ее экземпляры, а внешний - экземпляры сущности, которая находится в связи с данной сущностью.

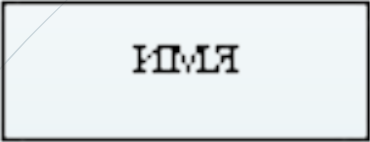
Типы сущностей

Независимая сущность Для определения экземпляра сущности нет необходимости ссылаться на другие сущности.

Зависимая сущность Для определения экземпляра такой сущности необходимо сослаться на экземпляр независимой сущности, с которой связана зависимая сущность.

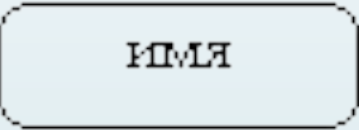


Обозначение сущностей в нотации IDEF1X



ИМЯ

Независимая сущность



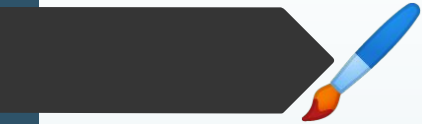
ИМЯ

Зависимая сущность

Сущность является "**независимой**", если каждый экземпляр сущности может быть однозначно идентифицирован без определения его отношений с другими сущностями. Пример: «Сотрудник».

Сущность называется "**зависимой**", если однозначная идентификация экземпляра сущности зависит от его отношения к другой сущности. Пример: «Участие в проектах».

Связь



Связь – это логическая ассоциация, устанавливаемая между сущностями.



Связь определяет количество экземпляров данной сущности, которое могут быть связаны с одним экземпляром другой сущности.



Связи бывают следующих типов:

- один к одному
- один ко многим



Пример 1:

«Страны» - «Столицы»



Пример 2:

«Международные Союзы» -
«Страны»



Пример 3:

«Сотрудник» - «Проект»

Связь «один-ко-многим»: Отделы – Сотрудники

Таблица
«Сотрудники» (Emp)

Табельный номер	ФИО сотрудника	<u>Отдел</u>
023	Волкова Елена Павловна	2
113	Белов Сергей Юрьевич	1
101	Рогов Сергей Михайлович	2
056	Панина Анна Алексеевна	1
...	...	...
098	Фролов Юрий Вадимович	9

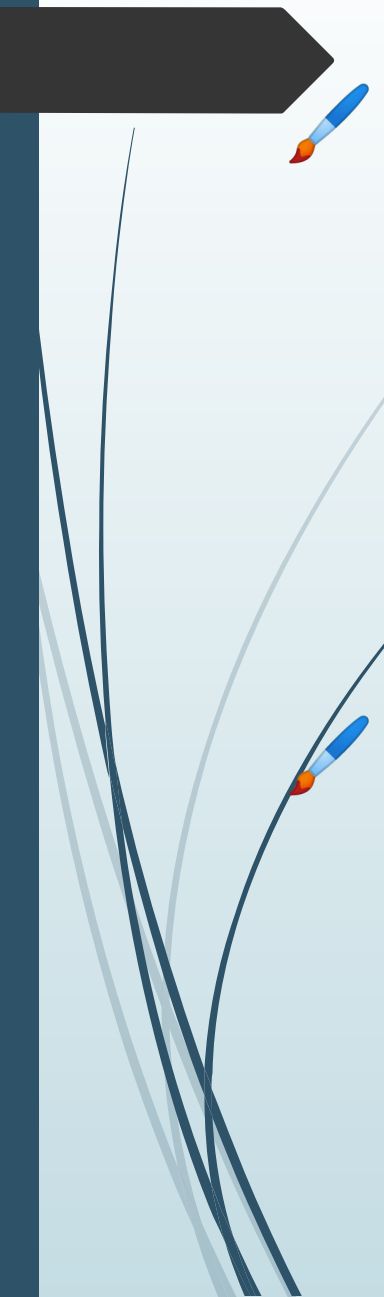
Таблица «Отделы»
(Departs)

Номер отдела	Название отдела
1	Информационный отдел
2	Администрация
3	Отдел кадров
...	...
9	Проектный отдел

«Номер отдела» –
первичный ключ в
таблице «Отделы»

«Отдел» – внешний ключ в таблице
«Сотрудники» к таблице «Отделы»

Связь «многие-со-многими»: Сотрудники- Проекты



Связи "many-to-many". Иногда бывает необходимо связывать сущности таким образом, что с обоих концов связи могут присутствовать несколько экземпляров сущности. Например, сотрудники консалтинговой компании участвуют в проектах. При этом один сотрудник может участвовать в нескольких проектах и в одном проекте могут участвовать несколько сотрудников. Для этого вводится разновидность связи "многие-со-многими". Оформляются через «развязочные таблицы», например: «участие в проектах» (сущность из двух атрибутов: код сотрудника (FK), код проекта (FK)).

Связь «многие-со-многими»: Сотрудники- Проекты

Таблица
«Сотрудники»

ФИО	Номер
Волкова Е.П.	023
Белов С.Ю.	113
Рогов С.М.	101
Панина А.А.	056
Фролов Ю.В.	098
...	...

Таблица
«Участие»

Участник	Роль	Пр
113	исполнитель	23/Н
101	руководитель	18-К
056	исполнитель	09/Р
101	консультант	18-К
098	руководитель	09/Р
...	...	23/Н
...	...	...

Таблица
«Проекты»

Шифр	Название проекта
23/Н	АИС "Налог"-2
18-К	ИПС "Жители"
09/Р	ГИС "Город"

В таблице «Участие»:

«Участник» – внешний ключ к таблице «Сотрудники»

«Проект» – внешний ключ к таблице «Проекты»

Виды связей

Отношение дополнительно определяется с помощью указания мощности: какое количество экземпляров сущности-потомка может существовать для сущности-родителя.

	1, 1
	0, M
	0, 1
	1, M
	точно N (N-произвольное число)

Виды связей



Связь – это графически изображаемая ассоциация, устанавливаемая между двумя сущностями. Эта ассоциация всегда является бинарной и может существовать между двумя разными сущностями или между сущностью и ей же самой (рекурсивная связь).

Элемент диаграммы	Обозначает
-----	неидентифицирующая связь
_____	идентифицирующая связь

Виды связей



При идентифицирующей связи миграция ключевых атрибутов родительской сущности (сущности со стороны связи 1 осуществляется всегда в область ключевых атрибутов дочерней сущности (сущности со стороны связи М). Мигрирующие ключи помечены символами (FK) – это аббревиатура слов **Foreign Key** (чуждый, посторонний ключ). Идентифицирующая связь между сущностями указывается тогда, когда экземпляр дочерней сущности не может существовать без экземпляра родительской сущности. При этом ключ связи FK (внешний ключ, ссылающийся на родительскую сущность) в составе дочерней сущности не может принимать значение **Null**. Можно сказать, что идентифицирующая связь является особым случаем **обязательной связи**.

Виды связей



При неидентифицирующей связи миграция ключевых атрибутов родительской сущности всегда осуществляется в область **неключевых атрибутов дочерней сущности**. При этом ключ связи **FK** (внешний ключ, ссылающийся на родительскую сущность) в составе дочерней сущности может принимать любые значения (в пределах заданных типов данных), в том числе и значение **Null**.

Нотация IDEF1X



Сущность обладает одним или несколькими **атрибутами**, которые являются либо **собственными** для сущности, либо наследуются через другое отношение (от **РК «родителя»** передается **FK** в **«сущность-потомок»**).



Атрибуты однозначно идентифицируют каждый экземпляр сущности.



Каждый атрибут идентифицируется уникальным именем.

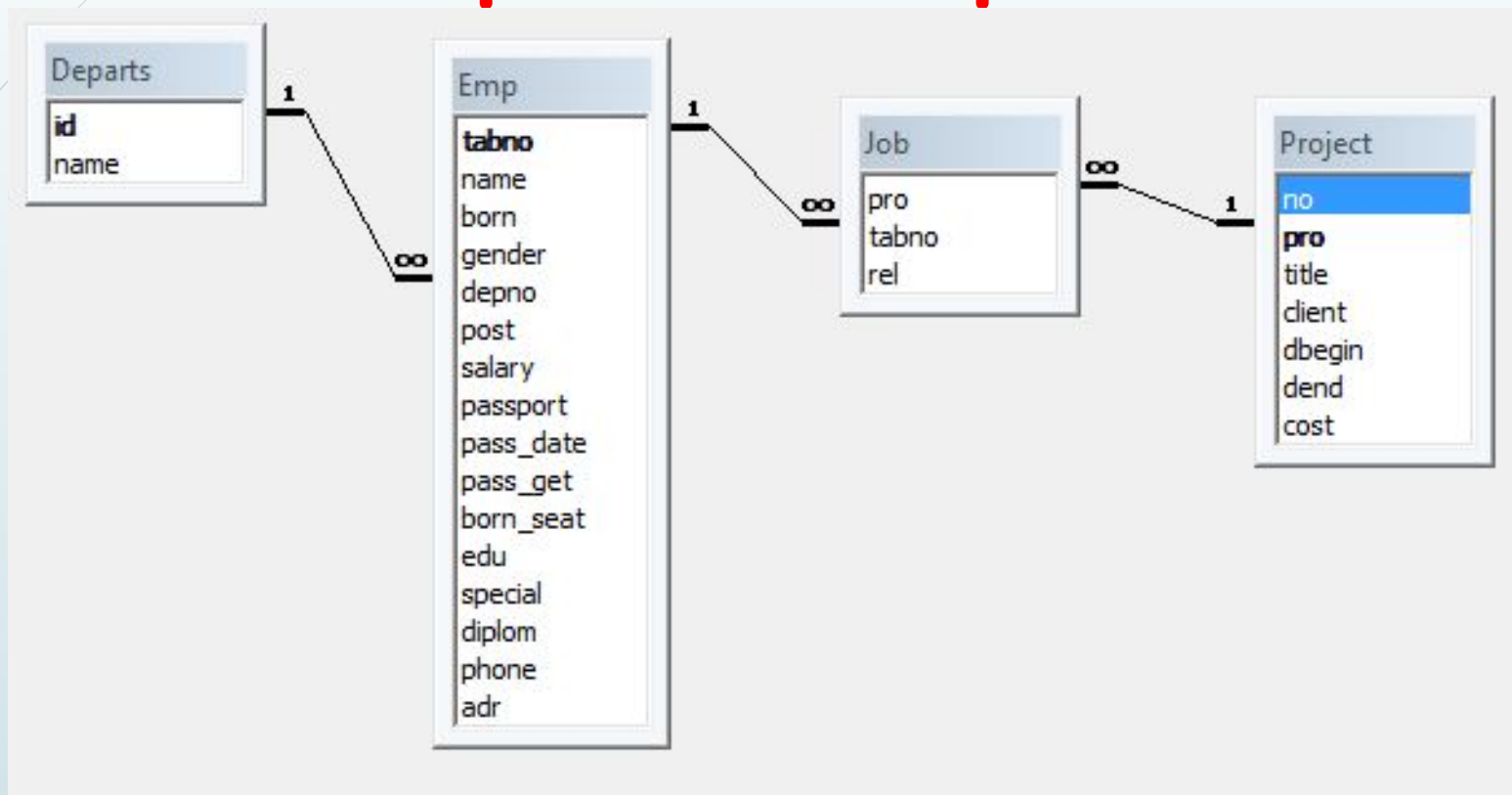


Атрибуты изображаются в виде списка их имен внутри блока ассоциированной сущности, причем каждый атрибут занимает отдельную строку.



Определяющие первичный ключ атрибуты размещаются наверху списка и отделяются от других атрибутов горизонтальной чертой.

Пример БД: «Проектная организация»



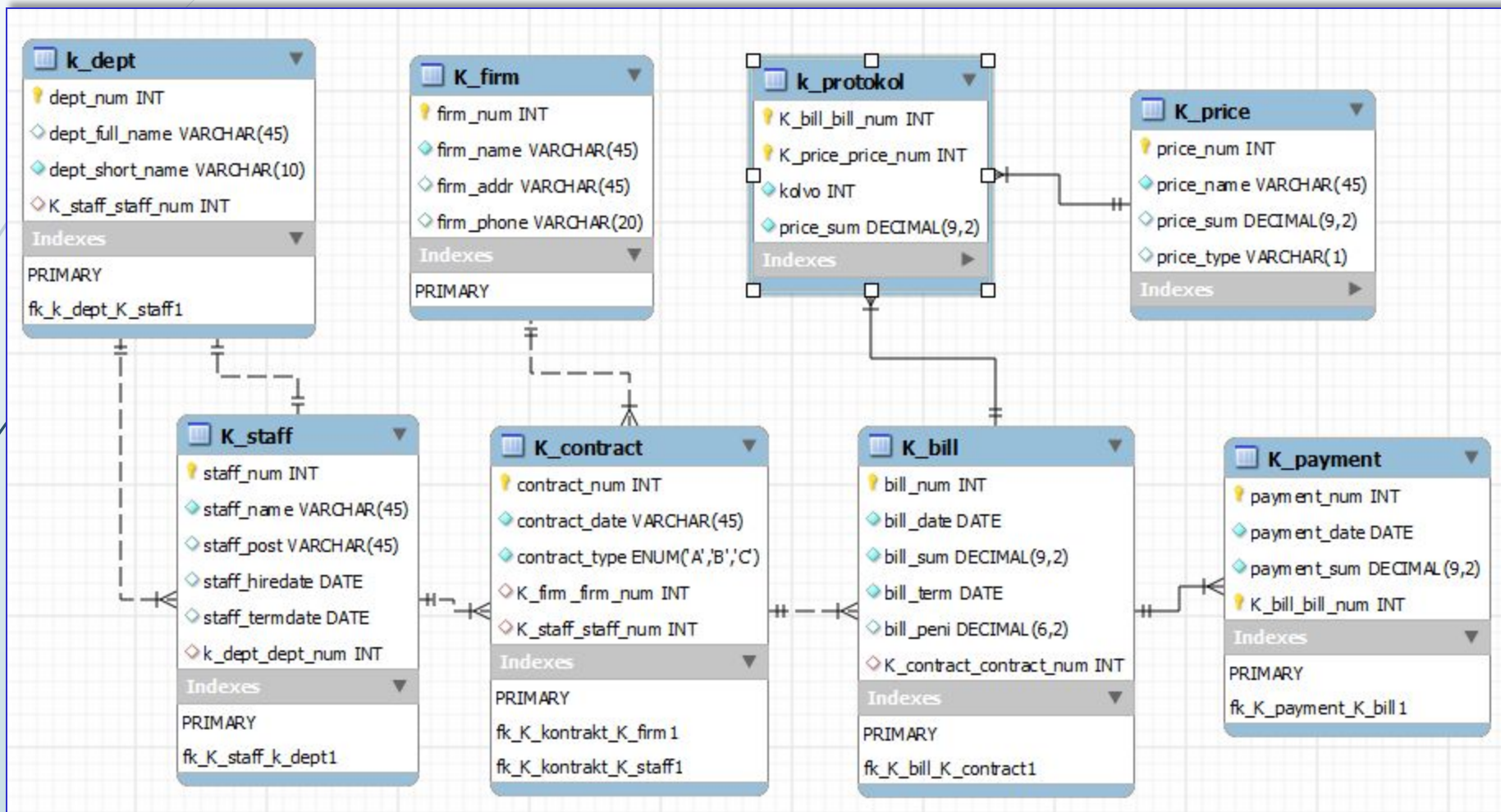
Departs – отделы,

Emp – сотрудники,

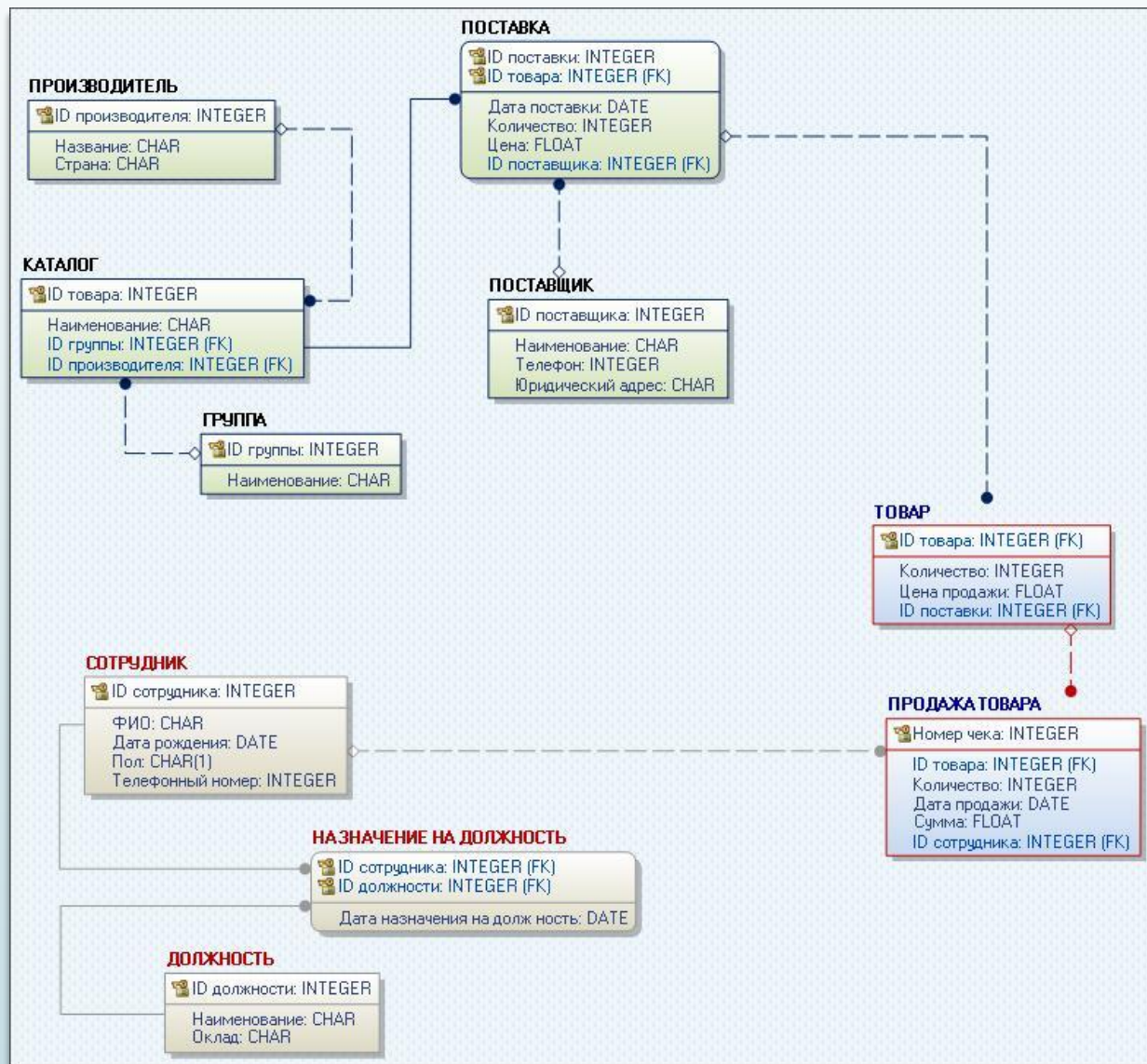
Project – проекты,

Job – участие в проектах.

Пример ER-диаграммы в MySQL WORKBENCH



Пример ER-диаграммы в СА ERwin Data Modeler (ERwin)



Нормальные формы ER-схем



Как и в реляционных схемах баз данных, в ER-схемах вводится понятие **нормальных форм**, причем их смысл соответствует смыслу реляционных нормальных форм.



Заметим, что формулировки нормальных форм ER-схем делают более понятным смысл нормализации реляционных схем. Приведем краткие и неформальные определения трех первых нормальных форм.



В первой нормальной форме ER-схемы устраняются повторяющиеся атрибуты или группы атрибутов, т.е. производится выявление неявных сущностей, "замаскированных" под атрибуты.

Нормальные формы ER-схем



Во второй нормальной форме устраняются атрибуты, зависящие только от части уникального идентификатора. Эти атрибуты должны определять отдельную сущность.



В третьей нормальной форме устраняются атрибуты, зависящие от атрибутов, не входящих в уникальный идентификатор. Эти атрибуты являются основой отдельной сущности.



При правильном проектировании все СУЩНОСТИ должны быть по крайней мере в третьей нормальной форме.

Получение реляционной схемы из ER-схемы

-  **Шаг 1.** Каждая простая сущность превращается в таблицу. Имя сущности становится именем таблицы.
-  **Шаг 2.** Каждый атрибут становится возможным столбцом с тем же именем; может выбираться более точный формат. Столбцы, соответствующие необязательным атрибутам, могут содержать неопределенные значения; столбцы, соответствующие обязательным атрибутам, - не могут.
-  **Шаг 3.** Компоненты уникального идентификатора сущности превращаются в первичный ключ таблицы.

Получение реляционной схемы из ER-схемы



Шаг 4. Связи многие-к-одному (и один-к-одному) становятся внешними ключами. Т.е. делается копия уникального идентификатора с конца связи "один", и соответствующие столбцы составляют внешний ключ. Необязательные связи соответствуют столбцам, допускающим неопределенные значения; обязательные связи - столбцам, не допускающим неопределенные значения.



Шаг 5. Индексы создаются для первичного ключа (уникальный индекс) и внешних ключей.