

KARPOV.COURSES >>> КОНСПЕКТ



> Конспект > 2 урок > Нормальные формы

> Оглавление

> Оглавление

> Реляционная модель данных

Аспекты (составляющие) РМД

Основные понятия

Отношение

> Первая нормальная форма

> Потенциальный/первичный ключи и их выбор

Потенциальный ключ

Пример

Ключи-идентификаторы

> Вторая и третья нормальные формы

Функциональная зависимость

Аномалии

2НФ

Аномалии

3НФ

> Сравнение первых трёх нормальных форм

Слабая/сильная нормализация

Определения **1НФ**, **2НФ** и **3НФ**

> Темпоральные (хронологические) базы данных

> [Шестая нормальная форма](#)

> [Entity-Relationship](#)

[Entity](#)

[Relation](#)

[Примеры](#)

[Примеры](#)

> [Дополнительные материалы](#)

> **Реляционная модель данных**

Реляционная модель данных (РМД) — логическая модель данных, прикладная теория построения баз данных, которая является приложением к задачам обработки данных таких разделов математики, как теория множеств и логика первого порядка.

Аспекты (составляющие) РМД

1. **Структурный** — данные в базе данных представляют собой набор отношений.
2. **Целостности** — отношения (таблицы) отвечают определенным условиям целостности. РМД поддерживает декларативные ограничения целостности уровня домена (типа данных), уровня отношения и уровня базы данных.
3. **Обработки (манипулирования)** — РМД поддерживает операторы манипулирования отношениями (реляционная алгебра, реляционное исчисление).

Мы будем рассматривать именно **структурный** аспект.

Основные понятия

- **Атрибут** — свойство некоторой сущности. Часто называется полем таблицы.
- **Домен атрибута** — множество допустимых значений, которые может принимать атрибут.
- **Кортеж** — конечное множество взаимосвязанных допустимых значений атрибутов, которые вместе описывают некоторую сущность (строка таблицы).
- **Отношение** — конечное множество кортежей (таблица).



Основные понятия с табличной точки зрения

Чтобы проще понимать реляционные термины, можно посмотреть таблицу соответствующих «табличных» терминов:

Таблица соответствия «табличных» терминов реляционным

Реляционный термин	«Табличный термин»
<u>База данных</u>	Набор таблиц
<u>Схема базы данных</u>	Набор заголовков таблиц
<u>Отношение</u>	Таблица
<u>Заголовок отношения</u>	Заголовок таблицы
<u>Тело отношения</u>	Тело таблицы
<u>Атрибут отношения</u>	Наименование столбца таблицы
<u>Кортеж отношения</u>	Строка таблицы
<u>Степень (-арность) отношения</u>	Количество столбцов таблицы
<u>Мощность отношения</u>	Количество строк таблицы
<u>Домены и типы данных</u>	Типы данных в ячейках таблицы

Отношение

В отношении **нет** одинаковых кортежей. Тело отношения есть множество кортежей и, как всякое множество, не может содержать неразличимые элементы. Таблицы же, напротив, могут содержать одинаковые строки.

Кортежи не упорядочены (сверху вниз). Причина та же - тело отношения есть *множество*, а множество не упорядочено. Одно и то же отношение может быть изображено разными таблицами, в которых строки идут в различном порядке. При этом в таблицах у нас в любом случае **есть какой-то порядок** – вне зависимости от того, указываем мы явную сортировку строк или нет, физически данные будут расположены в определенном порядке.

Атрибуты отношения не упорядочены (слева направо). Т.к. каждый атрибут имеет уникальное имя в пределах отношения, то порядок атрибутов не имеет значения. Одно и то же отношение может быть изображено разными таблицами, в которых столбцы идут в различном порядке.

Все значения атрибутов атомарны. Это следует из того, что лежащие в их основе атрибуты имеют атомарные значения.

> Первая нормальная форма

Отношение находится в **первой нормальной форме (1НФ)** тогда и только тогда, когда в любом допустимом значении отношения каждый его кортеж содержит только одно значение для каждого из атрибутов.

Если говорить проще: все кортежи в отдельном атрибуте должны иметь значения одного домена, и эти значения не должны являться группой/массивом данных.

Студенты

Аа ФИО	☰ Должность	☰ Курсы
<u>Иванов</u> <u>Иван</u> <u>Иванович</u>	Программист	Название: Аналитик данных; Дата сдачи: 30.09.2021 Название: Start ML; Дата сдачи: 30.11.2021 Название: Hard ML; Дата сдачи: 01.12.2021

В таблице выше можно заметить, что атрибут «Курсы» не атомарен. Если нам потребуется проанализировать данный атрибут, то мы будем вынуждены разобрать

строку. Осуществим переход в первую нормальную форму:

Студенты

<u>Аа</u> ФИО	<u>≡</u> Должность	<u>≡</u> Название	<u>≡</u> Дата сдачи
<u>Иванов Иван Иванович</u>	Программист	Аналитик данных	30.09.2021
<u>Иванов Иван Иванович</u>	Программист	Start ML	30.11.2021
<u>Иванов Иван Иванович</u>	Программист	Hard ML	01.12.2021

Теперь каждое значение атомарно, строки не повторяются.

> Потенциальный/первичный ключи и их выбор

Потенциальный ключ

Подмножество атрибутов отношения будем называть **потенциальным ключом**, если оно обладает следующими свойствами:

- Свойство **уникальности** – в отношении не может быть двух различных кортежей с одинаковым значением ключа.
- Свойство **неизбыточности** - никакое подмножество не обладает свойством уникальности.

Потенциальный ключ, состоящий из одного атрибута, называется **простым**.

Потенциальный ключ, состоящий из нескольких атрибутов, называется **составным**.

Отношение может иметь несколько потенциальных ключей. Традиционно, один из потенциальных ключей объявляется **первичным**, а остальные - **альтернативными**.

Пример

Рассмотрим таблицу, находящуюся в 1НФ:

Студенты

<u>Аа</u> ФИО	<u>≡</u> Должность	<u>≡</u> Название	<u>≡</u> Дата сдачи
<u>Иванов Иван Иванович</u>	Программист	Аналитик данных	30.09.2021

Аа ФИО	≡ Должность	≡ Название	≡ Дата сдачи
<u>Иванов Иван Иванович</u>	Программист	Start ML	30.11.2021
<u>Иванов Иван Иванович</u>	Программист	Hard ML	01.12.2021

Глядя на имеющиеся в этой таблице строки, мы можем попробовать определить потенциальные и первичный ключи:

1. **ФИО** и **Должность** повторяются, поэтому не могут быть потенциальными ключами.
2. **Название** и **Дата сдачи** уникальны, поэтому являются потенциальными ключами. При этом, определить составной ключ вида **{Название, Дата сдачи}** нельзя, так как это нарушает свойство **неизбыточности**.

Однако если добавить ещё одну строчку вида:

Студенты

Аа ФИО	≡ Должность	≡ Название	≡ Дата сдачи
<u>Петров Петр Петрович</u>	Аналитик	Hard ML	01.12.2021

То становится понятно, что **Название** и **Дата сдачи** не могут быть потенциальными ключами – они могут повторяться. Теперь нам подходит только составной тип ключа, например: **{ФИО, Название}**, **{Должность, Название}**, **{ФИО, Дата сдачи}**, **{Должность, Дата сдачи}**.

Последовательность определения ключей в примере выше также показывает нам, что выбирая ключ **по** некоторому небольшому **набору данных**, легко допустить **ошибку**. Поэтому, лучше всего подходить к выбору ключа – не снизу-вверх (от данных), а **сверху-вниз**, принимая во внимание **природу данных**.

Ключи-идентификаторы

Также, в современных СУБД повсеместно используются **ключи-идентификаторы** – некоторый специальный, суррогатный (искусственный) уникальный ключ.

Распространенный пример такого ключа – инкрементальный, т.е. содержащий числовой идентификатор, который монотонно увеличивается для каждой новой строки.

Студенты

Аа ИД Студента	≡ ФИО	≡ Должность	# ИД Курса	≡ Название	≡ Дата сдачи
----------------	-------	-------------	------------	------------	--------------

ИД Студента	ФИО	Должность	ИД Курса	Название	Дата сдачи
1	Иванов Иван Иванович	Программист	1	Аналитик данных	30.09.2021
1	Иванов Иван Иванович	Программист	2	Start ML	30.11.2021
1	Иванов Иван Иванович	Программист	3	Hard ML	01.12.2021
2	Петров Петр Петрович	Аналитик	3	Hard ML	01.12.2021

В таблице мы ввели ключи идентификаторы `ИД Студента` и `ИД Курса`, которые могут вместе образовать первичный ключ вида `{ИД Студента, ИД Курса}`.

> Вторая и третья нормальные формы

Функциональная зависимость

Функциональная зависимость между атрибутами (множествами атрибутов) X и Y означает, что для любого допустимого набора кортежей в данном отношении если два кортежа совпадают по значению X , то они совпадают по значению Y .

Например, если значение атрибута «Название компании» — *Canonical Ltd*, то значением атрибута «Штаб-квартира» в таком кортеже всегда будет *Millbank Tower, London, United Kingdom*.

Обозначение: `{X} -> {Y}`.

В целом, проще всего понять этот термин, вспомнив школьную математику и зависимость `y = f(x)`.

Преподаватели, курсы, лекции

ID_Преподавателя	ФИО	ID_Категории	Тип	ID_Курса	Курс	ID_Лекции
1	Иванов	1	Full Staff	1	Hard ML	1
1	Иванов	1	Full Staff	2	Аналитика	1
2	Петров	1	Full Staff	1	Hard ML	2

Aa ID_Преподавателя	≡ ФИО	# ID_Категории	▼ Тип	# ID_Курса	≡ Курс	# ID_Лекции
3	Сидоров	2	Part Staff	1	Hard ML	3
3	Сидоров	2	Part Staff	2	Аналитика	2

Первичный ключ в данном отношении это {ID_Преподавателя, ID_Курса, ID_Лекции}. ID_Категории не является частью ключа для выполнения свойства избыточности.

В этом отношении мы можем наблюдать следующие функциональные зависимости:

- ID_Преподавателя → ФИО
- ID_Преподавателя → ID_Категории
- ID_Преподавателя → Тип
- ID_Категории → Тип
- ID_Курса → Курс
- {ID_Преподавателя, ID_Курса} → ID_Лекции

Аномалии

Однако, таблице выше имеет следующие **аномалии**:

- Аномалия **вставки**. В отношение нельзя вставить данные о преподавателе, который пока не участвует ни в одном блоке.
- Аномалия **удаления**. Если по курсу временно прекращены лекции, то при удалении данных о лекции по этому курсу будут удалены и данные о самом курсе. При этом, если был преподаватель, который работал только над этим курсом, то будут потеряны и данные об этом сотруднике.
- Аномалия **обновления**. Если необходимо изменить какую-либо информацию о преподавателе, то придётся изменять значения атрибутов во всех записях.

Часть этих аномалий нам поможет решить **вторая нормальная форма**.

2НФ

Отношение находится во **второй нормальной форме (сокращённо 2НФ)** тогда и только тогда, когда оно находится в **первой нормальной форме** и каждый его неключевой

атрибут *неприводимо* зависим от первичного ключа.

Неприводимость означает, что в составе потенциального ключа отсутствует меньшее подмножество атрибутов, от которого можно также вывести данную функциональную зависимость. Для неприводимой функциональной зависимости часто используется эквивалентное понятие «*полная функциональная зависимость*».

Перефразируя,

Отношение находится во **второй нормальной форме** тогда и только тогда, когда оно находится в **первой нормальной форме** и нет неключевых атрибутов, зависящих от части сложного ключа.

Разделим таблицу из примера выше на три других.


Преподаватели

<u>Aa</u> ID_Преподавателя	 ФИО	 ID_Категории	 Тип
<u>1</u>	Иванов	1	Full Staff
<u>2</u>	Петров	1	Full Staff
<u>3</u>	Сидоров	2	Part Staff

Функциональные зависимости:

- ID_Преподавателя → ФИО
- ID_Преподавателя → ID_Категории
- ID_Преподавателя → Тип
- ID_Категории → Тип

Курсы

<u>Aa</u> ID_Курса	 Курс
<u>1</u>	Hard ML
<u>2</u>	Аналитика

Функциональные зависимости:

- `ID_Курса` → `Курс`

Соответствие преподавателя-курса-лекции

<u>Aa</u> ID_Преподавателя	# ID_Курса	# ID_Лекции
<u>1</u>	1	1
<u>1</u>	2	1
<u>2</u>	1	2
<u>3</u>	1	3
<u>3</u>	2	2

Функциональные зависимости:

- `{ID_Преподавателя, ID_Курса}` → `ID_Лекции`

Аномалии

Однако, если внимательно посмотреть на таблицу преподавателей, то можно заметить, что мы все еще имеем две аномалии:

- Аномалия **удаления**. Если удалить данные о сотруднике, то можно потерять и данные о категориях.
- Аномалия **обновления**. Если необходимо изменить информацию о категории, то придется изменить много записей.

ЗНФ

Отношение находится в **третьей нормальной форме (ЗНФ)**, когда находится во **2НФ** и каждый неключевой атрибут нетранзитивно (*напрямую*) зависит от первичного ключа.

Отношение находится в **третьей нормальной форме (ЗНФ)** тогда и только тогда, когда отношение находится в **2НФ** и все неключевые атрибуты взаимно независимы.

Вспомним функциональные зависимости таблицы преподавателей:

- `ID_Преподавателя` → `ФИО`
- `ID_Преподавателя` → `ID_Категории`

- ID_Преподавателя → Тип
- ID_Категории → Тип

Здесь мы как раз можем наблюдать **транзитивную зависимость**, так как Тип зависит от ID_Категории и ID_Преподавателя, а ID_Категории в свою очередь зависит от ID_Преподавателя.

Разрешить её можно, разделив эту таблицу ещё на две.

Преподаватели

<u>Aa</u> ID_Преподавателя	≡ ФИО	# ID_Категории
<u>1</u>	Иванов	1
<u>2</u>	Петров	1
<u>3</u>	Сидоров	2

Функциональные зависимости:

- ID_Преподавателя → ФИО
- ID_Преподавателя → ID_Категории

Категории

<u>Aa</u> ID_Категории	▼ Тип
<u>1</u>	Full Staff
<u>2</u>	Part Staff

Функциональные зависимости:

- ID_Категории → Тип

> Сравнение первых трёх нормальных форм

Слабая/сильная нормализация

Сравнение степеней нормализации

Аа Критерий	☰ Отношения слабо нормализованы (1НФ, 2НФ)	☰ Отношения сильно нормализованы (3НФ)
<u>Адекватность базы данных предметной области</u>	ХУЖЕ (-)	ЛУЧШЕ (+)
<u>Легкость разработки и сопровождения базы данных</u>	СЛОЖНЕЕ (-)	ЛЕГЧЕ (+)
<u>Скорость выполнения вставки, обновления, удаления</u>	МЕДЛЕННЕЕ (-)	БЫСТРЕЕ (+)
<u>Скорость выполнения выборки данных</u>	БЫСТРЕЕ (+)	МЕДЛЕННЕЕ (-)

По большинству критериев более сильная нормализация превосходит слабую. Однако, если нам необходимо производить **выборку**, то менее нормализованные данные в этом случае подходят **лучше**, так как требуют меньше ресурсов для выполнения запроса.

Слабо нормализованные данные используются в последних слоях, куда обращаются аналитики данных.

Определения 1НФ, 2НФ и 3НФ

Первая нормальная форма (1НФ) – это обычное отношение. Отношение в 1НФ обладает следующими свойствами: в отношении нет одинаковых кортежей; кортежи не упорядочены; атрибуты не упорядочены; все значения атрибутов атомарны.

Отношение находится во **второй нормальной форме (2НФ)** тогда и только тогда, когда отношение находится в 1НФ и нет неключевых атрибутов, зависящих от части сложного ключа.

Отношение находится в **третьей нормальной форме (3НФ)** тогда и только тогда, когда отношение находится в 2НФ и все неключевые атрибуты взаимно независимы.

Однако, чтобы не запоминать определения, можно воспользоваться следующим «шпаргалкой»:

Данные зависят от ключа [1НФ], всего ключа [2НФ] и ничего, кроме ключа [3НФ]. Как проще запомнить суть первых трёх нормальных форм

> Темпоральные (хронологические) базы данных

Хронологическая база данных может быть неформально определена как база, которая содержит исторические данные наряду с текущими данными или вместо них.

Обычные, или нехронологические, базы данных содержат только текущие данные; актуальность таких баз поддерживается путем обновления данных сразу же после того, как представленные в них высказывания становятся ложным.

Рассмотрим две таблицы зарплат преподавателей.

Первая таблица – таблица из обычной базы данных. В ней мы храним только актуальные данные по зарплатам преподавателей:

Зарплаты преподавателей

Преподаватель	Договор	Зарплата
Прохоров А.И.	Full	150
Соловьева М.Е.	Part	150
Амосова Е.С.	Part	120

Во второй таблице добавляется столбец **Время фиксации факта**, и тем самым таблица становится темпоральной:

Зарплаты преподавателей

Преподаватель	Договор	Зарплата	Время фиксации факта
Прохоров А.И.	Full	100	с 1 января 2018
Соловьева М.Е.	Part	150	с 1 января 2018

Аа Преподаватель	▼ Договор	# Зарплата	≡ Время фиксации факта
<u>Амосова Е.С.</u>	Full	60	с 1 января 2018
<u>Амосова Е.С.</u>	Full	120	с 1 июня 2018
<u>Амосова Е.С.</u>	Part	120	с 1 сентября 2018

> Шестая нормальная форма

Переменная отношения находится в **6НФ** тогда и только тогда, когда она удовлетворяет всем нетривиальным зависимостям соединения. Из определения следует, что переменная находится в 6НФ тогда и только тогда, когда она **неприводима**, то есть не может быть подвергнута дальнейшей декомпозиции без потерь.

Для достижения 6НФ, преобразуем таблицу с зарплатами из предыдущего шага. и получаем две новые таблицы.

История типа договора

Аа Преподаватель	▼ Договор	≡ Время фиксации акта
<u>Прохоров А.И.</u>	Full	с 1 января 2018
<u>Соловьева М.Е.</u>	Part	с 1 января 2018
<u>Амосова Е.С.</u>	Full	с 1 января 2018
<u>Амосова Е.С.</u>	Part	с 1 сентября 2018

История зарплат

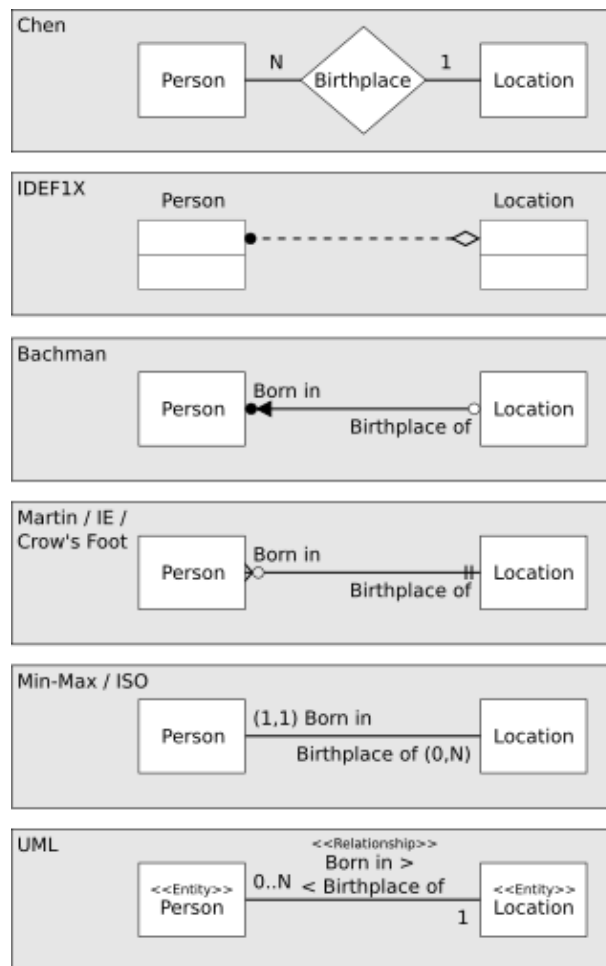
Аа Преподаватель	# Зарплата	≡ Время фиксации акта
<u>Прохоров А.И.</u>	150	с 1 января 2018
<u>Соловьева М.Е.</u>	150	с 1 января 2018
<u>Амосова Е.С.</u>	60	с 1 января 2018
<u>Амосова Е.С.</u>	120	с 1 июня 2018

> Entity-Relationship

Схема «сущность-связь» (также ERD или ER-диаграмма) — это разновидность блок-схемы, где показано, как разные «сущности» (люди, объекты, концепции и так далее) связаны между собой внутри системы

Эти диаграммы устроены по тому же принципу, что и грамматические структуры: сущности выполняют роль существительных, а связи — глаголов.

Существует множество нотаций этой диаграммы:

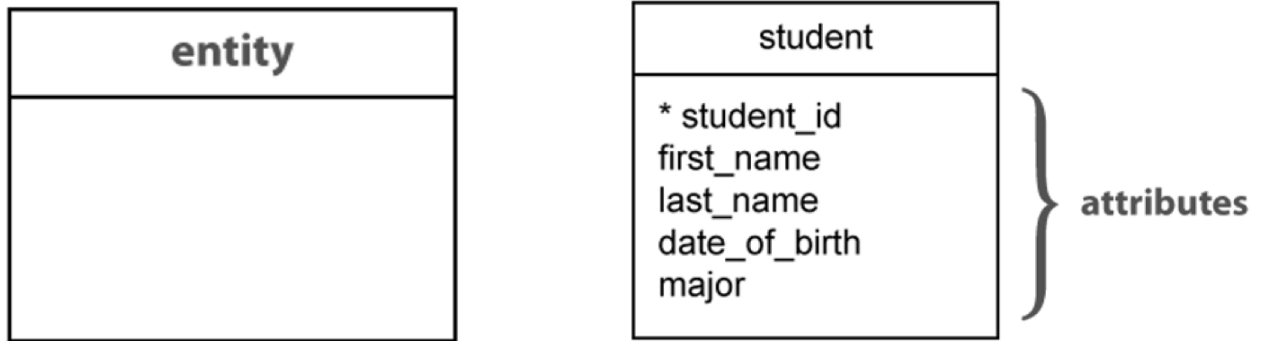


Нотации ER-диаграммы

Чаще всего используется нотация Мартина.

Entity

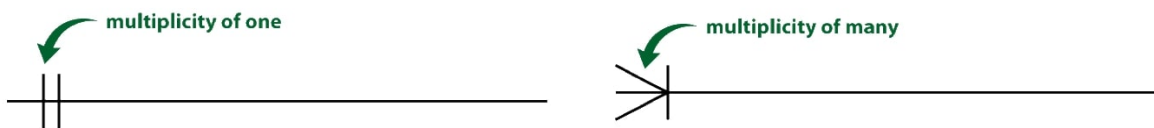
Сущность – это предмет, который существует независимо от другого предмета.



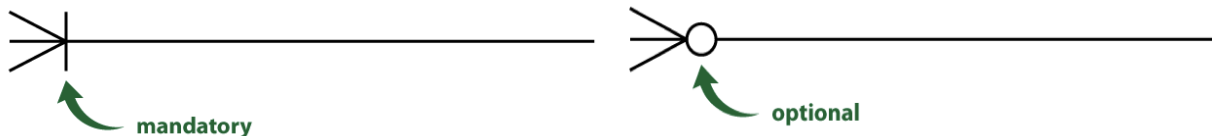
Пример сущности в нотации Мартина

Relation

Связь – это отношение между экземплярами двух (и более) разных сущностей.

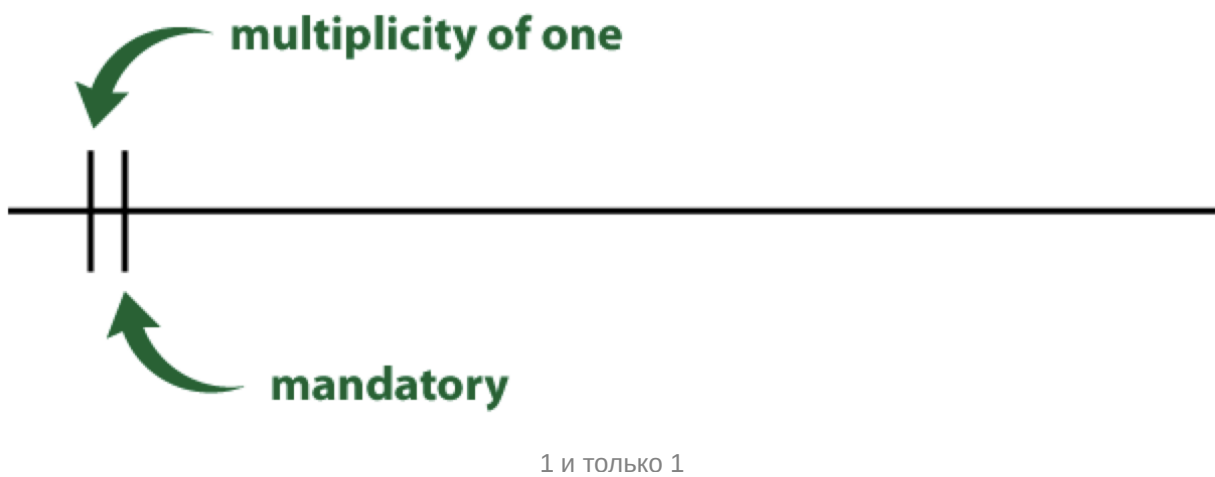


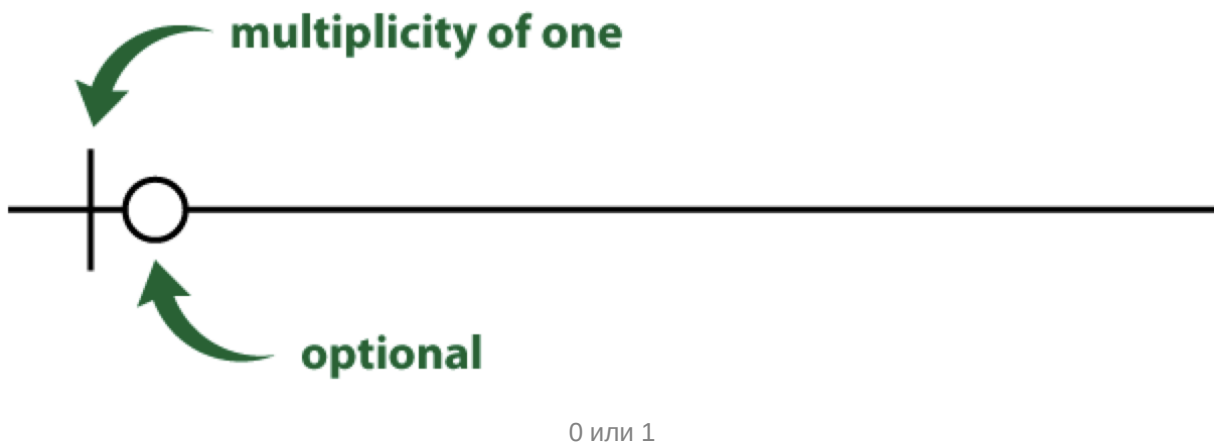
Кардинальность связи – количество (одна/много).



Минимальное количество связей – хотя бы одна/ни одной.

Комбинируя все эти варианты, мы получаем четыре типа связей:





Примеры

Примеры

Один студент занимает одно место:



Один преподаватель может вести несколько курсов, а может не вести ни одного.



Студент может быть записан на много курсов или не быть записан вообще. На курс могут быть записаны много студентов, а может быть никто не записан.



> **Дополнительные материалы**

- Нормальная форма Бойса-Кодда
- Четвертая нормальная форма
- Пятая нормальная форма
- Нестеров С.А. – Базы данных, 2016 – Глава 5 – Нормализация реляционных баз данных.