

# Информационные системы и сети

## Нормализация отношений

*Бывает, лежишь на диване, пьешь пиво,  
смотришь телевизор! А тут звонок:*

*- Ты сына забрал? Продукты купил?  
Завтра мама приезжает, не забыл? Что  
ты молчишь, Сережа?*

*А ты не Сережа, ты - Коля!!! И на душе  
праздник!*

*Чтобы на душе у админа всегда был  
праздник необходимо чтобы его БД была  
в НФБК*

**При разработке лекции использовались материалы сайта:**



**FORUM**

**<http://citforum.ru/database/dbguide/index.shtml>**

**И как всегда:**



**ВИКИПЕДИЯ**  
*Свободная энциклопедия*

Блюдо	Вид	Рецепт	Порций	Дата Р	Продукт	Калорийность	Вес (г)	Поставщик	Город	Страна	Вес (кг)	Цена (\$)	Дата П
Лобио	Закуска	Лом.	158	1/9/94	Фасоль	3070	200	"Хуанхэ"	Пекин	Китай	250	0.37	24/8/94
Лобио	Закуска	Лом	108	1/9/94	Лук	450	40	"Наталка"	Киев	Украина	100	0.52	27/8/94
Лобио	Закуска	Лом	108	1/9/94	Масло	7420	30	"Лайма"	Рига	Латвия	70	1.55	30/8/94
Лобио	Закуска	Лом	108	1/9/94	Зелень	180	10	"Даугава"	Рига	Латвия	15	0.99	30/8/94
Харчо	Суп	...	144	1/9/94	Мясо	1660	80	"Наталка"	Киев	Украина	100	2.18	27/8/94
Харчо	Суп	...	144	1/9/94	Лук	450	30	"Наталка"	Киев	Украина	100	0.52	27/8/94
Харчо	Суп	...	144	1/9/94	Томаты	240	40	"Полесье"	Киев	Украина	120	0.45	27/8/94
Харчо	Суп	...	144	1/9/94	Рис	3340	50	"Хуанхэ"	Пекин	Китай	75	0.44	24/8/94
Харчо	Суп	...	144	1/9/94	Масло	7420	15	"Полесье"	Киев	Украина	50	1.62	27/8/94
Харчо	Суп	...	144	1/9/94	Зелень	180	15	"Наталка"	Киев	Украина	10	0.88	27/8/94
Шашлык	Горячее	...	207	1/9/94	Мясо	1660	180	"Юрмала"	Рига	Латвия	200	2.05	30/8/94
Шашлык	Горячее	...	207	1/9/94	Лук	450	40	"Полесье"	Киев	Украина	50	0.61	27/8/94
Шашлык	Горячее	...	207	1/9/94	Томаты	240	100	"Полесье"	Киев	Украина	120	0.45	27/8/94
Шашлык	Горячее	...	207	1/9/94	Зелень	180	20	"Даугава"	Рига	Латвия	15	0.99	30/8/94
Кофе	Десерт	...	235	1/9/94	Кофе	2750	8	"Хуанхэ"	Пекин	Китай	40	2.87	24/8/94

*-Это ведь нехорошо, то, что мы с тобой сделали.  
- Попробуем еще раз... может быть, получится лучше.*

# Почему проект БД может быть плохим?

**1. Избыточность.** Данные практически всех столбцов многократно повторяются.

**2. Потенциальная противоречивость (аномалии обновления).**

Вследствие избыточности можно обновить адрес поставщика в одной строке, оставляя его неизменным в других.

**3. Аномалии включения.** В БД не может быть записан новый поставщик ("Няринга", Вильнюс, Литва), если поставляемый им продукт (Огурцы) не используется ни в одном блюде.

**4. Аномалии удаления.** Обратная проблема возникает при необходимости удаления всех продуктов, поставляемых данным поставщиком или всех блюд, использующих эти продукты. При таких удалениях будут утрачены сведения о таком поставщике.

# Как сделать хорошо?

Блюда

БЛ	Блюдо	Вид
1	Лобио	Закуска
2	Харчо	Суп
3	Шашлык	Горячее
4	Кофе	Десерт
...	...	...

Рецепты

Блюдо	Рецепт
Лобио	Ломаную очис
...	...

Расход

Блюдо	Порций	Дата_Р
Лобио	158	1/9/94
Харчо	144	1/9/94
Шашлык	207	1/9/94
Кофе	235	1/9/94
...	...	...

Продукты

ПР	Продукт	Калор.
1	Фасоль	3070
2	Лук	450
3	Масло	7420
4	Зелень	180
5	Мясо	1660
...	...	...

Состав

БЛ	ПР	Вес (г)
1	1	200
1	2	40
1	3	30
1	4	10
2	5	80
...	...	...

Поставщики

ПОС	Поставщик	Город	Страна
1	"Полесье"	Киев	Украина
2	"Наталка"	Киев	Украина
3	"Хуанхэ"	Пекин	Китай
4	"Лайма"	Рига	Латвия
5	"Юрмала"	Рига	Латвия
...	...	...	...

Поставки

ПОС	ПР	Вес (кг)	Цена (\$)	Дата_П
1	6	120	0.45	27/8/94
1	3	50	1.62	27/8/94
1	2	50	0.61	27/8/94
2	2	100	0.52	27/8/94
...	...	...	...	...

# Какой механизм позволяет улучшать структуру БД?

***Нормализация*** – это разбиение таблицы на две или более, обладающих лучшими свойствами при включении, изменении и удалении данных.

Окончательная цель нормализации сводится к получению такого проекта базы данных, в котором каждый факт появляется лишь в одном месте, т.е. исключена избыточность информации. Это делается не столько с целью экономии памяти, сколько для исключения возможной противоречивости хранимых данных.

Различают ***интуитивную нормализацию*** (основана на работе 6-го чувства ☺) и ***декомпозицию***, выполняемую по строго определенному алгоритму.



# Функциональные зависимости

Всякая *нормализованная* таблица (содержащая только атомарные значения) автоматически считается таблицей в первой нормальной форме, сокращенно 1НФ.

Таблица находится в 2НФ, если она находится в 1НФ и удовлетворяет, кроме того, некоторому дополнительному условию и т.д. Таким образом, каждая следующая нормальная форма является в некотором смысле более ограниченной, но и более желательной, чем предшествующая.

Теория нормализации основывается на наличии той или иной зависимости между полями таблицы. Определены два вида таких зависимостей: *функциональные* и *многозначные*.

***Функциональная зависимость.*** Поле В таблицы функционально зависит от поля А той же таблицы в том и только в том случае, когда в любой заданный момент времени для каждого из различных значений поля А обязательно существует только одно из различных значений поля В. Отметим, что здесь допускается, что поля А и В могут быть составными.

# Функциональные зависимости

Например, в таблице **Блюда** поля Блюдо и Вид функционально зависят от ключа БЛ, а в таблице **Поставщики** поле Страна функционально зависит от составного ключа (Поставщик, Город). Однако последняя зависимость не является функционально полной, так как Страна функционально зависит и от части ключа – поля Город.

Блюда

БЛ	Блюдо	Вид
1	Лобио	Закуска
2	Харчо	Суп
3	Шашлык	Горячее
4	Кофе	Десерт
---	---	---

Поставщики

Поставщик	Город	Страна
"Полесье"	Киев	Украина
"Наталка"	Киев	Украина
"Хуанхэ"	Пекин	Китай
"Лайма"	Рига	Латвия
"Юрмала"	Рига	Латвия
---	---	---

**Полная функциональная зависимость.** Поле В находится в полной функциональной зависимости от составного поля А, если оно функционально зависит от А и не зависит функционально от любого подмножества поля А.



# Многозначные зависимости

**Многозначная зависимость.** Поле А многозначно определяет поле В той же таблицы, если для каждого значения поля А существует хорошо определенное множество соответствующих значений В.

Обучение

Дисциплина	Преподаватель	Учебник
Информатика	Шипилов П.А.	Форсайт Р. Паскаль для всех
Информатика	Шипилов П.А.	Уэйт М. и др. Язык Си
Информатика	Голованевский Г.Л.	Форсайт Р. Паскаль для всех
Информатика	Голованевский Г.Л.	Уэйт М. и др. Язык Си
...	...	...

Многозначная зависимость "*Дисциплина-Преподаватель*": дисциплина может читаться несколькими преподавателями.

Другая многозначная зависимость "*Дисциплина-Учебник*": при изучении Информатики используются учебники "Паскаль для всех" и "Язык Си". При этом Преподаватель и Учебник не связаны функциональной зависимостью, что приводит к появлению избыточности (для добавление еще одного учебника придется ввести в таблицу две новых строки). Дело улучшается при замене этой таблицы на две: (Дисциплина-Преподаватель и Дисциплина-Учебник).

# Нормальные формы

Таблица находится в *первой нормальной форме (1НФ)* тогда и только тогда, когда ни одна из ее строк не содержит в любом своем поле более одного значения и ни одно из ее ключевых полей не пусто.

Таблица находится во *второй нормальной форме (2НФ)*, если она удовлетворяет определению 1НФ и все ее поля, не входящие в первичный ключ, связаны полной функциональной зависимостью с первичным ключом.

Таблица находится в *третьей нормальной форме (3НФ)*, если она удовлетворяет определению 2НФ и не одно из ее неключевых полей не зависит функционально от любого другого неключевого поля.

Таблица находится в *нормальной форме Бойса-Кодда (НФБК)*, если и только если любая функциональная зависимость между его полями сводится к полной функциональной зависимости от возможного ключа.

# Нормальные формы более высоких порядков

*Полной декомпозицией* таблицы называют такую совокупность произвольного числа ее проекций, соединение которых полностью совпадает с содержимым таблицы.

Естественным соединением\* таблиц, полученных в результате полной декомпозиции, можно образовать исходную таблицу

Таблица находится в *пятой нормальной форме (5НФ)* тогда и только тогда, когда в каждой ее полной декомпозиции все проекции содержат возможный ключ. Таблица, не имеющая ни одной полной декомпозиции, также находится в 5НФ.

*Четвертая нормальная форма (4НФ)* является частным случаем 5НФ, когда полная декомпозиция должна быть соединением ровно двух проекций. Весьма не просто подобрать реальную таблицу, которая находилась бы в 4НФ, но не была бы в 5НФ!!!

\* - будет рассмотрено в рамках курса «Математические основы реляционных БД» (1 курс магистратуры)

# Процедура нормализации

***Нормализация*** – это процесс последовательной замены таблицы ее полными декомпозициями до тех пор, пока все они не будут находиться в 5НФ. На практике же достаточно привести таблицы к НФБК.

Процедура нормализации основывается на том, что единственными функциональными зависимостями в любой таблице должны быть зависимости вида  $K \rightarrow F$ , где  $K$  – первичный ключ, а  $F$  – некоторое другое поле. Заметим, что это следует из определения первичного ключа таблицы, в соответствии с которым  $K \rightarrow F$  всегда имеет место для всех полей данной таблицы. "Один факт в одном месте" говорит о том, что не имеют силы никакие другие функциональные зависимости. Цель нормализации состоит именно в том, чтобы избавиться от всех этих "других" функциональных зависимостей, т.е. таких, которые имеют иной вид, чем  $K \rightarrow F$ .

# Процедура нормализации

Если подменить на время нормализации коды первичных (внешних) ключей на исходные ключи, то, по существу, следует рассмотреть лишь два случая:

1. Таблица имеет составной первичный ключ вида, скажем,  $(K1, K2)$ , и включает также поле  $F$ , которое функционально зависит от части этого ключа, например, от  $K2$ , но не от полного ключа. В этом случае рекомендуется сформировать другую таблицу, содержащую  $K2$  и  $F$  (первичный ключ –  $K2$ ), и удалить  $F$  из первоначальной таблицы:

Заменить	$T(K1, K2, F)$ ,	первичный ключ $(K1, K2)$ ,	ФЗ $K2 \rightarrow F$
на	$T1(K1, K2)$ ,	первичный ключ $(K1, K2)$ ,	
и	$T2(K2, F)$ ,	первичный ключ $K2$ .	

# Процедура нормализации

2. Таблица имеет первичный (возможный) ключ  $K$ , не являющееся возможным ключом поле  $F1$ , которое, конечно, функционально зависит от  $K$ , и другое неключевое поле  $F2$ , которое функционально зависит от  $F1$ . Решение здесь, по существу, то же самое, что и прежде – формируется другая таблица, содержащая  $F1$  и  $F2$ , с первичным ключом  $F1$ , и  $F2$  удаляется из первоначальной таблицы:

Заменить	$T(K, F1, F2)$ ,	первичный ключ $K$ ,	ФЗ $F1 \rightarrow F2$
на	$T1(K, F1)$ ,	первичный ключ $K$ ,	
и	$T2(F1, F2)$ ,	первичный ключ $F1$ .	

Для любой заданной таблицы, повторяя применение двух рассмотренных правил, почти во всех практических ситуациях можно получить в конечном счете множество таблиц, которые находятся в "окончательной" нормальной форме и, таким образом, не содержат каких-либо функциональных зависимостей вида, отличного от  $K \rightarrow F$ .



# Процедура нормализации

## Пример!!!

Блюдо, Дата\_Р, Продукт, Поставщик, Город, Дата\_П.

Блюдо->Вид.

Блюдо->Рецепт

(Блюдо, Дата\_Р)->Порций

Продукт->Калорийность

(Блюдо, Продукт)->Вес

Город->Страна

(Поставщик, Город, Дата\_П)->Цена

Блюда (Блюдо, Вид)

Рецепты (Блюдо, Рецепт)

Расход (Блюдо, Дата Р, Порций)

Продукты (Продукт, Калорийность)

Состав (Блюдо, Продукт, Вес (г))

Города (Город, Страна)

Поставки (Поставщик, Город, Дата П, Вес (кг), Цена).

# Требования к СУБД

## (правила Кодда)

*правило 0: Основное правило (Foundation Rule): Реляционная СУБД должна быть способна полностью управлять базой данных, используя связи между данными.*

*правило 1: Явное представление данных (The Information Rule):*

*Информация должна быть представлена в виде данных, хранящихся в ячейках. Данные, хранящиеся в ячейках, должны быть атомарны. Порядок строк в реляционной таблице не должен влиять на смысл данных.*

*правило 2: Гарантированный доступ к данным (Guaranteed Access Rule):*

*Доступ к данным должен быть свободен от двусмысленности. К каждому элементу данных должен быть гарантирован доступ с помощью комбинации имени таблицы, первичного ключа строки и имени столбца.*

# Требования к СУБД

## (правила Кодда)

*правило 3: Полная обработка неизвестных значений (Systematic Treatment of Null Values):*

*Неизвестные значения NULL, отличные от любого известного значения, должны поддерживаться для всех типов данных при выполнении любых операций. Например, для числовых данных неизвестные значения не должны рассматриваться как нули, а для символьных данных — как пустые строки.*

*правило 4: Доступ к словарю данных в терминах реляционной модели (Active On-Line Catalog Based on the Relational Model):*

*Словарь данных должен сохраняться в форме реляционных таблиц, и СУБД должна поддерживать доступ к нему при помощи стандартных языковых средств, тех же самых, которые используются для работы с реляционными таблицами, содержащими пользовательские данные.*

# Требования к СУБД (правила Кодда)

**правило 5: Полнота подмножества языка (Comprehensive Data Sublanguage Rule):**

*Система управления реляционными базами данных должна поддерживать хотя бы один реляционный язык, который*

*(а) имеет линейный синтаксис,*

*(б) может использоваться как интерактивно, так и в прикладных программах,*

*(в) поддерживает операции определения данных, определения представлений, манипулирования данными (интерактивные и программные), ограничители целостности, управления доступом и операции управления транзакциями (begin, commit и rollback).*

**правило 6: Возможность модификации представлений (View Updating Rule):**

*Каждое представление должно поддерживать все операции манипулирования данными, которые поддерживают реляционные таблицы: операции выборки, вставки, модификации и удаления данных.*

# Требования к СУБД

## (правила Кодда)

**правило 7: Наличие высокоуровневых операций управления данными (High-Level Insert, Update, and Delete):**

*Операции вставки, модификации и удаления данных должны поддерживаться не только по отношению к одной строке реляционной таблицы, но по отношению к любому множеству строк.*

**правило 8: Физическая независимость данных (Physical Data Independence):**

*Приложения не должны зависеть от используемых способов хранения данных на носителях, от аппаратного обеспечения компьютеров, на которых находится реляционная база данных.*

**правило 9: Логическая независимость данных (Logical Data Independence):** *Представление данных в приложении не должно зависеть от структуры реляционных таблиц. Если в процессе нормализации одна реляционная таблица разделяется на две, представление должно обеспечить объединение этих данных, чтобы изменение структуры реляционных таблиц не сказывалось на работе приложений.*

# Требования к СУБД

## (правила Кодда)

***правило 10: Независимость контроля целостности (Integrity Independence):***

*Вся информация, необходимая для поддержания целостности, должна находиться в словаре данных. Язык для работы с данными должен выполнять проверку входных данных и автоматически поддерживать целостность данных.*

***правило 11: Дистрибутивная независимость (Distribution Independence):***

*База данных может быть распределённой, может находиться на нескольких компьютерах, и это не должно оказывать влияние на приложения. Перенос базы данных на другой компьютер не должен оказывать влияния на приложения.*

***правило 12: Согласование языковых уровней (The Nonsubversion Rule):***

*Если используется низкоуровневый язык доступа к данным, он не должен игнорировать правила безопасности и правила целостности, которые поддерживаются языком более высокого уровня.*



# Требования к СУБД (правила Кодда)

*НЕТ ни одной СУБД, которая бы удовлетворяла ВСЕМ правилам Кодда*

*Предлагайте свои варианты СУБД – поле деятельности открыто*

*Зачет в этом случае – автоматом ☺*

