

Лекция № 13.

Проектирование реляционных баз данных

Проектирование реляционных баз данных

При проектировании базы данных решаются две основные проблемы:

- 1) каким образом *отобразить объекты предметной области* (далее – ПО) в *абстрактные объекты модели данных*, чтобы это отображение не противоречило семантике ПО и было наиболее эффективным;
- 2) как *обеспечить эффективность хранения базы данных и выполнения запросов к ней*, т.е. как расположить данные во внешней памяти, какие дополнительные структуры (например индексные файлы) создать и т.п.

Первая задача известна как *проблема логического проектирования БД*, а вторая задача – как *проблема физического проектирования БД*.

Проектирование реляционных баз данных

Мы будем рассматривать только проблему **логического проектирования БД** и будем считать, что проектирование БД заключается в принятии следующих решений:

- 1) из каких отношений должна состоять БД,
- 2) какие атрибуты должны быть у этих отношений.

Проектирование баз данных будет выполняться с использованием *нормализации*.

Проектирование реляционных баз данных

1. Основные понятия

Так как РБД включает только нормализованные отношения, первый шаг перехода к РБД есть *представление данных в виде нормализованных отношений*.

Полученное таким образом отношение уже находится в **первой нормальной форме** (далее – НФ). Будем ее обозначать как НФ-1.

Дальнейший процесс нормализации состоит в том, что некоторые отношения, заданные в НФ-1, "расщепляются" на более простые. Причем сначала получается *вторая нормальная форма* (далее – НФ-2), а затем и *третья нормальная форма* (далее – НФ-3), обладающая лучшими свойствами, чем НФ-2.

Проектирование реляционных баз данных

Зачем нужна нормализация?

1. Нормализация нужна потому, что ни первая, ни вторая нормальные формы не обеспечивают сохранности отношений при изменениях БД.
2. Нормализация позволяет минимизировать избыточность данных и число выполняемых над ними операций.

Основные свойства нормальных форм:

- 1) каждая следующая НФ в некотором смысле лучше предыдущей,
- 2) при переходе к следующей НФ свойства предыдущей НФ сохраняются.

Понятие НФ основывается на фундаментальном для РБД понятии *функциональной зависимости*. Дадим несколько определений.

Проектирование реляционных баз данных

Определение 1. Функциональная зависимость.

В отношении R атрибут B функционально зависит от атрибута A (A и B могут быть составными), если каждому значению атрибута A соответствует в точности одно значение атрибута B , т.е. имеет место отображение $R.A \rightarrow R.B$.

(Если известно значение атрибута A , можно получить значение атрибута B .)

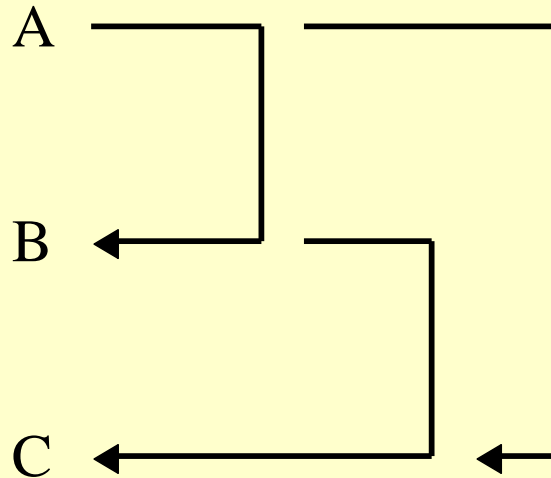
Определение 2. Полная функциональная зависимость.

Атрибут (набор атрибутов) B полностью зависит от другого набора атрибутов A отношения R , если B функционально зависит от всего множества A , но не зависит ни от какого подмножества A .

Проектирование реляционных баз данных

Определение 3. Транзитивная функциональная зависимость.

Функциональная зависимость $R.A \rightarrow R.C$ называется транзитивной, если в отношении R существует такой атрибут B , что имеют место зависимости $R.A \rightarrow R.B$ и $R.B \rightarrow R.C$.



Определение 4. Неключевой атрибут.

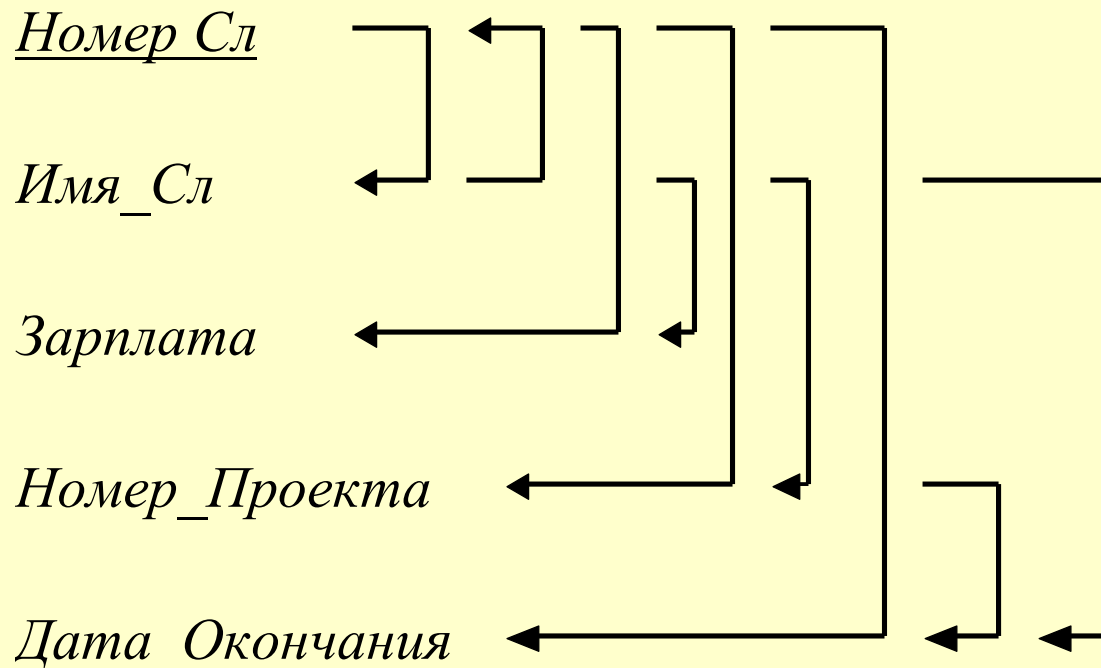
Неключевым (неосновным) называется атрибут, не входящий в состав первичного ключа.

Проектирование реляционных баз данных

Пример 1.

СЛУЖАЩИЙ (Номер Сл, Имя_Сл, Зарплата,
Номер_Проекта, Дата_Окончания).

Функциональные зависимости отношения СЛУЖАЩИЙ :



Проектирование реляционных баз данных

Пример 2.

Рассмотрим отношение *ДЕЯТЕЛЬНОСТЬ_ПРОГРАММИСТА*,
первичный ключ которого состоит из двух атрибутов.

ДЕЯТЕЛЬНОСТЬ_ПРОГРАММИСТА

(Номер Программиста, Номер Программы,
Имя_Программиста, *Имя_Программы*,
Количество_Рабочих_Часов).

Проектирование реляционных баз данных

Функциональные зависимости отношения

ДЕЯТЕЛЬНОСТЬ_ПРОГРАММИСТА заданы в предположении, что среди программистов нет однофамильцев и что две программы не могут иметь одинаковых имен.

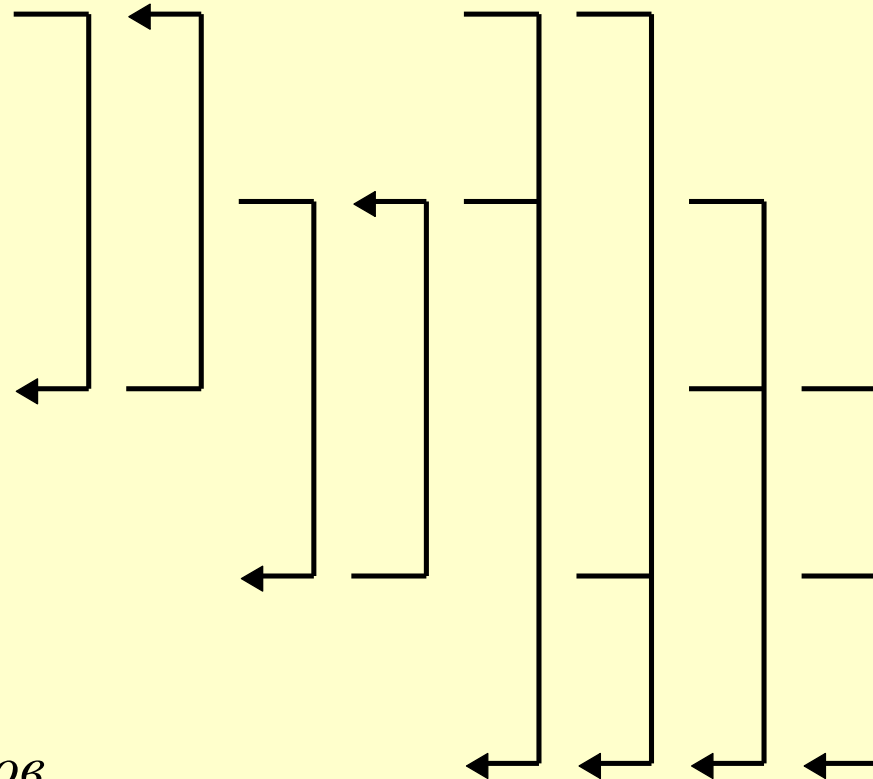
Номер Программиста

Номер Программы

Имя_Программиста

Имя_Программы

Количество_Рабочих_Часов



Проектирование реляционных баз данных

Комментарий к примерам.

1. В примере 1 все неключевые атрибуты полностью зависят от ключевых (элементов возможного ключа).
2. В примере 2 неключевой атрибут *Количество_Рабочих_Часов* полностью зависит от нескольких пар атрибутов (от первичного ключа и других пар).
3. В примере 1 атрибут *Дата_Окончания* транзитивно зависит от атрибута *Номер_Сл.*

Проектирование реляционных баз данных

2. Вторая нормальная форма

Определение 5. Вторая нормальная форма.

Отношение R находится во второй нормальной форме, если оно находится в первой нормальной форме и каждый неключевой атрибут полностью зависит от первичного ключа.

Если в отношении R ключ содержит один атрибут, то это отношение уже задано в НФ-2. (См. отношение *СЛУЖАЩИЙ* в примере 1.)
Нарушение условий НФ-2 приводит к ряду неудобств.

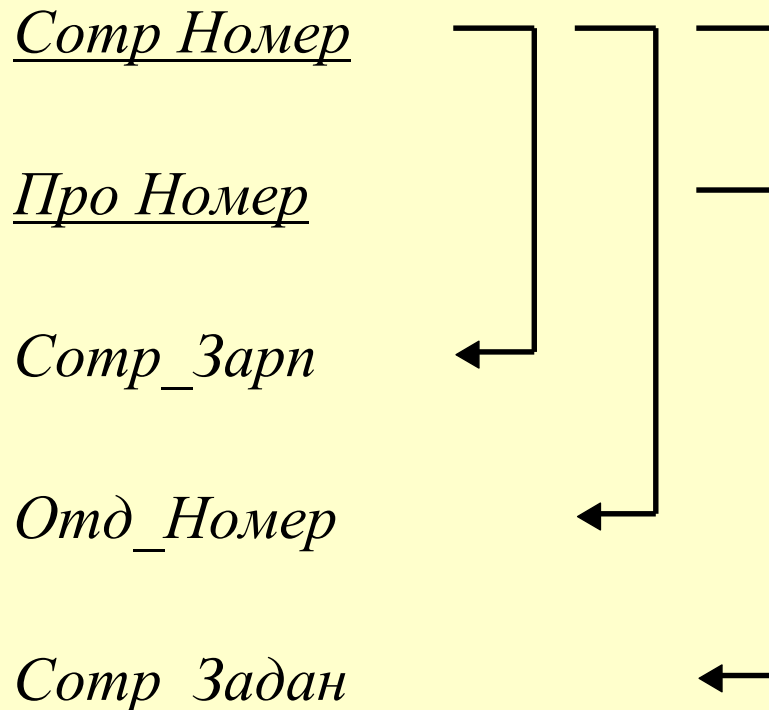
Пример 3.

Рассмотрим отношение

СОТРУДНИКИ-ОТДЕЛЫ-ПРОЕКТЫ (Сотр Номер, Проект Номер,
Сотр_Зарп, Отд_Номер, Сотр_Задан)

Проектирование реляционных баз данных

Функциональные зависимости отношения
СОТРУДНИКИ-ОТДЕЛЫ-ПРОЕКТЫ



Проектирование реляционных баз данных

В отношении *СОТРУДНИКИ-ОТДЕЛЫ-ПРОЕКТЫ* неключевые атрибуты *Сотр_Зарп* и *Отд_Номер* функционально зависят от части ключа, а именно от атрибута *Сотр_Номер*.

В результате при выполнении операций возникают следующие сложности:

- 1) нельзя вставить кортеж, описывающий сотрудника, еще не включенного ни в один проект, так как первичный ключ не может содержать неопределенное значение;
- 2) при удалении кортежа из отношения мы не только уничтожаем связь сотрудника с проектом, но и теряем информацию о его отделе и зарплате;
- 3) при переводе сотрудника в другой отдел потребуется модифицировать все кортежи, описывающие этого сотрудника.

Проектирование реляционных баз данных

Эти аномалии устраняются путем нормализации. Можно, например, произвести следующую декомпозицию отношения *СОТРУДНИКИ-ОТДЕЛЫ-ПРОЕКТЫ* на два отношения:

СОТРУДНИКИ-ОТДЕЛЫ (Сотр Номер, Сотр_Зарп, Отд_Номер)

СОТРУДНИКИ-ПРОЕКТЫ (Сотр Номер, Про Номер, Сотр_Задан)

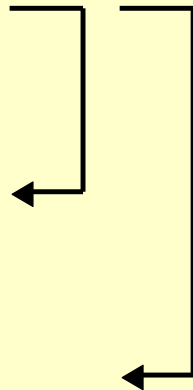
Функциональные зависимости этих отношений:

СОТРУДНИКИ-ОТДЕЛЫ

Сотр Номер

Сотр_Зарп

Отд_Номер

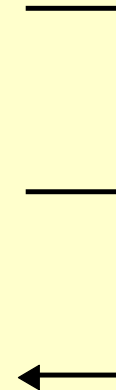


СОТРУДНИКИ-ПРОЕКТЫ

Сотр Номер

Про Номер

Сотр_Задан



Проектирование реляционных баз данных

3. Третья нормальная форма

Определение 6. *Третья нормальная форма.*

Отношение R находится в третьей нормальной форме, если оно находится во второй нормальной форме и каждый неключевой атрибут нетранзитивно зависит от первичного ключа.

Если отношение содержит транзитивные зависимости, их необходимо устранить. Так, в *примере 1* в отношении

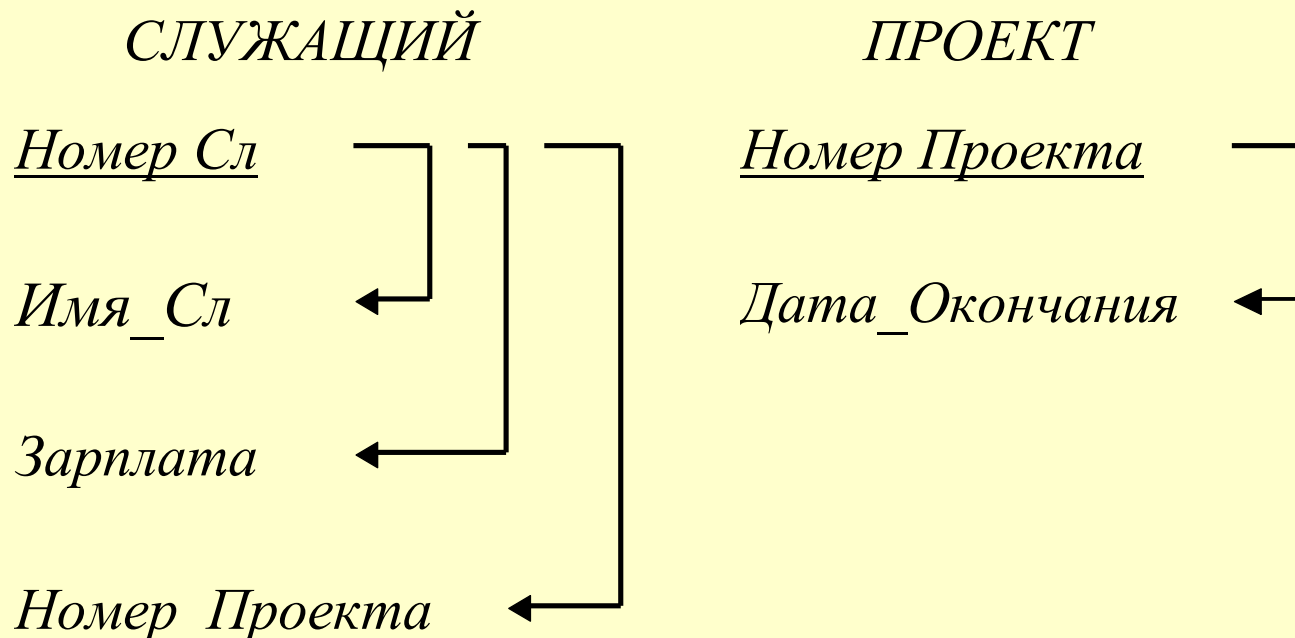
СЛУЖАЩИЙ (Номер_Сл, Имя_Сл, Зарплата,
Номер_Проекта, Дата_Окончания)

атрибут *Дата_Окончания* зависит от атрибута *Номер_Проекта*, который в свою очередь зависит от атрибута *Номер_Сл*. Таким образом, атрибут *Дата_Окончания* транзитивно зависит от атрибута *Номер_Сл*.

Проектирование реляционных баз данных

Отношение *СЛУЖАЩИЙ* приводится к НФ-3 путем его расщепления на два отношения

СЛУЖАЩИЙ (Номер Сл, Имя_Сл, Зарплата, Номер_Проекта),
ПРОЕКТ (Номер Проекта, Дата_Окончания).



Проектирование реляционных баз данных

Отношение *СЛУЖАЩИЙ* в примере 1 имело следующие недостатки:

- 1) до момента привлечения хотя бы одного сотрудника в проект дату окончания проекта некуда было записать;
- 2) если бы все сотрудники прекратили работу над проектом и он был бы приостановлен до привлечения новых сотрудников, в БД были бы уничтожены все кортежи и пропала бы дата окончания проекта;
- 3) изменение даты окончания проекта потребовало бы изменений в нескольких кортежах.

Отношения в НФ-3 лишены таких недостатков, так как не содержат транзитивных и неполных зависимостей.

Проектирование реляционных баз данных

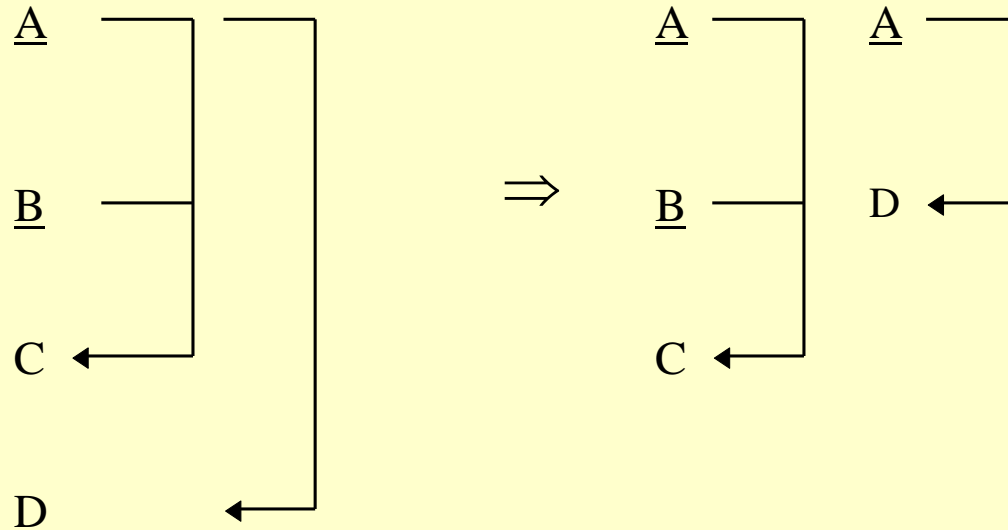
4. Схема нормализации

Нормализация отношений базы данных состоит из трех основных этапов:

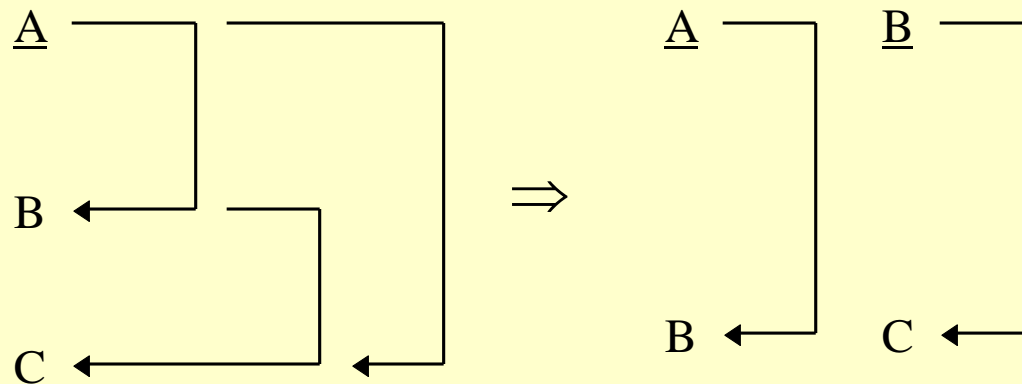
- 1) переход от произвольных структур данных (ненормализованной формы) к двумерным отношениям (приведение к НФ-1);
- 2) устранение неполных зависимостей (приведение к НФ-2);
- 3) устранение транзитивных зависимостей (приведение к НФ-3).

Схема нормализации

Приведение к НФ-2:



Приведение к НФ-3:



Проектирование реляционных баз данных

В определении взаимозависимости данных у пользователя имеется некоторая свобода. Можно выбирать разные схемы отношений и разные ключи.

Однако, рекомендуется выбирать такие атрибуты и отношения, которые **наиболее устойчивы к изменениям и пополнениям БД**.

При выборе первичного ключа надо учитывать, что при изменении базы данных атрибуты, составляющие первичный ключ, могут перестать быть таковыми.

Например, если в качестве первичного ключа отношения *СЛУЖАЩИЙ* выбрать атрибут *Имя_Сл*, то нужно быть готовым к тому, что при появлении в БД однофамильцев он уже не будет первичным ключом.