

## Лабораторная работа №3.

### Проектирование баз данных.

#### Проектирование даталогической модели предметной области.

#### Даталогическая модель реляционной модели данных

##### Цель работы:

Познакомиться с основами проектирования даталогической модели реляционной модели данных

##### Теория

**Логическое (даталогическое) проектирование** — создание схемы базы данных на основе конкретной модели данных (реляционной модели данных, сетевой модели данных и т.д.).

На этапе логического проектирования учитывается специфика конкретной модели данных, но может не учитываться специфика конкретной СУБД.

##### Реляционная модель данных

Реляционная модель данных предоставляет средства описания данных на основе только их естественной структуры, т.е. без потребности введения какой-либо дополнительной структуры для целей машинного представления. Другими словами, представление данных не зависит от способа их физической организации. Это обеспечивается за счет использования математической теории отношений (само название «реляционная» происходит от английского relation –«отношение»).

Базовое понятие реляционной модели данных «отношение» (R), определенное на множествах  $D_1, D_2, \dots, D_n$ , определяется как подмножество декартова произведения  $D_1 * D_2 * \dots * D_n$ . При этом: 1) множества  $D_1, D_2, \dots, D_n$  называются **доменами** отношения; 2) элементы декартова произведения

$d_1 * d_2 * \dots * d_n$  называются **кортежами**; 3) число  $n$  определяет **степень отношения** ( $n=1$  - унарное,  $n=2$  - бинарное, ...,  $n$ -арное); количество кортежей называется **мощностью отношения**

Понятие **тип данных** в реляционной модели данных полностью адекватно понятию типа данных в языках программирования (обычно в современных реляционных БД допускается хранение символьных, числовых данных, битовых строк, специализированных числовых данных (таких как "деньги"), а также специальных "темпоральных" данных (дата, время, временной интервал)).

В самом общем виде **домен** определяется заданием некоторого базового типа данных, к которому относятся элементы домена, и произвольного логического выражения, применяемого к элементу типа данных. Если вычисление этого логического выражения дает результат "истина", то элемент данных является элементом домена.

Наиболее правильной интуитивной трактовкой понятия домена является понимание домена как допустимого потенциального множества значений данного типа (например, домен для числового значения оценки студента может быть записан в виде интервала  $[1,5]$ ).

Схема отношения – это именованное множество пар {имя атрибута, имя домена}. Схема базы данных (в структурном смысле) – это набор именованных схем отношений.

Кортеж, соответствующий данной схеме отношения, - это множество пар {имя атрибута, значение}, которое содержит одно вхождение каждого имени атрибута, принадлежащего схеме отношения. "Значение" является допустимым значением домена данного атрибута. Попросту говоря, кортеж - это набор именованных значений заданного типа.

Потенциальный ключ – в реляционной модели данных — подмножество атрибутов отношения, удовлетворяющее требованиям уникальности и минимальности.

Уникальность означает, что нет и не может быть двух кортежей данного отношения, в которых значения этого подмножества атрибутов совпадают (равны).

Свойство уникальности определяется не для конкретного значения переменной отношения в тот или иной момент времени, а по всем возможным значениям, то есть следует из внешнего знания о природе и закономерностях данных, которые могут находиться в переменной отношения.

Минимальность (несократимость) означает, что в составе потенциального ключа отсутствует меньшее подмножество атрибутов, удовлетворяющее условию уникальности. Иными словами, если из потенциального ключа убрать любой атрибут, он утратит свойство уникальности.

Поскольку все кортежи в отношении по определению уникальны, в нём всегда существует хотя бы один потенциальный ключ (например, включающий все атрибуты отношения).

В отношении может быть одновременно несколько потенциальных ключей. Один из них может быть выбран в качестве первичного ключа отношения, тогда другие потенциальные ключи называют альтернативными ключами.

Первичный ключ (англ. primary key) – в реляционной модели данных один из потенциальных ключей отношения, выбранный в качестве основного ключа (или ключа по умолчанию). Все другие потенциальные ключи отношения (если они есть) называют альтернативными ключами

Теоретически, все потенциальные ключи равно пригодны в качестве первичного ключа, на практике в качестве первичного обычно выбирается тот из потенциальных ключей, который имеет меньший размер (физического хранения) и/или включает меньшее количество атрибутов.

Если первичный ключ состоит из единственного атрибута, его называют простым ключом. Если первичный ключ состоит из двух и более атрибутов, его называют составным ключом.

Отношения имеют следующие свойства: 1) отсутствие кортежей-дубликатов (из этого свойства вытекает наличие у каждого кортежа первичного ключа); 2) отсутствие упорядоченности кортежей; 3) отсутствие упорядоченности атрибутов; 4) атомарность значений атрибутов, т.е. среди значений домена не могут содержаться множества значений (отношения).

Упрощенным представлением отношения является таблица, заголовком которой является схема отношения, а строками - кортежи отношения-экземпляра; в этом случае имена атрибутов именуют столбцы этой таблицы (см. рисунок 1). Для упрощения работы с реляционными базами данных данной терминологии придерживаются в большинстве коммерческих реляционных СУБД. Однако необходимо помнить, что некорректное и нестрогое использование указанных неформальных синонимов терминов вместо терминов «отношение», «кортеж», «домен», «атрибут» недопустимо.

	целое	строка		целое	Типы данных	
	номер	имя	должность	деньги	Домены	
Отношение	Табельный номер	Имя	Должность	Оклад	Премия	Атрибуты
	2934	Иванов	инженер	112	40	Кортежи
	2935	Петров	вед. инженер	144	50	
	2936	Сидоров	бухгалтер	92	35	
						Ключ

Рисунок 1. Основные компоненты реляционного отношения

Для отражения ассоциаций (связей) между кортежами разных отношений в реляционной модели данных используется дублирование их ключей. Атрибуты, представляющие собой копии ключей других отношений, называются внешними ключами.

### Даталогическая модель реляционной модели данных

Для реляционной модели данных даталогическая модель – набор схем отношений, обычно с указанием первичных ключей, а также «связей» между отношениями, представляющих собой внешние ключи.

**Преобразование инфологической модели в даталогическую модель,** как правило, осуществляется по формальным правилам:

1. Каждая простая сущность превращается в отношение (таблицу). Имя сущности становится именем отношения (таблицы). Каждый простой атрибут становится атрибутом (столбцом) отношения (таблицы) с тем же именем. Для каждого атрибута определяется тип данных. Наиболее востребованные типы данных приведены в таблице 1.

Таблица 1

Тип	Применение	Размер
<b>TINYINT</b>	целые числа самой маленькой разрядности	Диапазон чисел со знаком: –128 ... 127, без знака(unsigned) – 0 ... 255
<b>SMALLINT</b>	целые числа маленькой разрядности	Диапазон чисел со знаком: –32768 ... 32767, без знака(unsigned) – 0 ... 65535
<b>MEDIUMINT</b>	целые числа средней разрядности	Диапазон чисел со знаком: –8388608 ... 8388607, без знака(unsigned) – 0 ... 16777215
<b>INT</b> или <b>INTEGER</b>	целые числа обыкновенной разрядности	Диапазон чисел со знаком: -2147483648 ... 2147483647, без знака(unsigned) – 0 ... 4294967295
<b>BIGINT</b>	целые числа большой разрядности	Диапазон чисел со знаком: -9223372036854775808 ... 9223372036854775807, без знака(unsigned) – 0 ... 18446744073709551615
<b>FLOAT</b>	числа с плавающей запятой (одинарной точности)	Диапазон от –3.402823466E+38 до –1.175494351E-38 и от 1.175494351E-38 до 3.402823466E+38
<b>DOUBLE, REAL</b>	числа с плавающей запятой (двойной точности)	Диапазон от -1.7976931348623157E+308 до -2.2250738585072014E-308 и от 2.2250738585072014E-308 до 1.7976931348623157E+308
<b>DATE</b>	дата	Диапазон от ‘1000-01-01’ до ‘9999-12-31’, отображается в виде ‘YYYY-MM-DD’
<b>DATETIME</b>	комбинация даты и времени	Диапазон от ‘1000-01-01 00:00:00’ до ‘9999-12-31 23:59:59’, отображается в виде ‘YYYY-MM-DD HH:MM:SS’

Тип	Применение	Размер
<b>TIME</b>	время	Диапазон от '-838:59:59' до '838:59:59', отображается в виде 'HH:MM:SS'
<b>YEAR</b>	год (2-х или 4-х цифирное представление)	Диапазон от 1901 до 2155 (для 4-х цифирного представления) и 1970-2069 (70-69, для 2-х цифирного представление), отображается в виде 'YYYY'
<b>CHAR</b>	строка фиксированной длины	Диапазон от 1 до 255 символов, пробелы по краям обрезаются при получении значения, заполняется пробелом до заданной длины (если символов меньше, чем заданная длина), сортировка и сравнение значений регистро-независимое
<b>VARCHAR</b>	строка изменяющийся длины	Диапазон от 1 до 255 символов, пробелы по краям обрезаются при сохранении, сортировка и сравнение значений регистро-независимое
<b>TEXT</b>		максимальная длина 65535 символов

2. Компоненты уникального идентификатора сущности превращаются в первичный ключ (допускается использование в качестве уникального идентификатора искусственный код). Если имеется несколько возможных уникальных идентификаторов, выбирается наиболее используемый. Учитываются также следующие факторы: длина ключа – в качестве первичного ключа выбирается, как правило, самый короткий из вероятных ключей; стабильность – желательно выбирать в качестве первичного ключа атрибуты, которые не изменяются; мнемоничность – при прочих равных условиях следует отдавать предпочтение тем из вероятных ключей, которые легче запомнить.

Если в состав уникального идентификатора входят связи, то к числу столбцов первичного ключа добавляется копия уникального идентификатора сущности, находящейся на дальнем конце связи (процесс может продолжаться рекурсивно).

3. Каждому из многозначных атрибутов ставится в соответствие отношение, полями которого будут идентификатор, выбранный в качестве первичного ключа, и многозначный атрибут. Ключ этого отношения будет составным, включающим оба эти атрибута.

4. Если сущность имеет необязательный атрибут, возможны два варианта: 1) если таким свойством обладают многие экземпляры объекта, его можно хранить как обычный атрибут в той же таблице (столбец может содержать неопределенные значения); 2) если свойством обладает малое

число экземпляров, то можно выделить отношение, включающее идентификатор и соответствующий атрибут. Отношение будет содержать столько строк, сколько объектов имеет свойство.

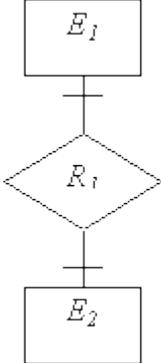
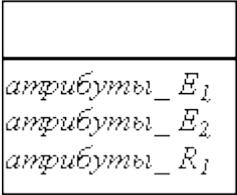
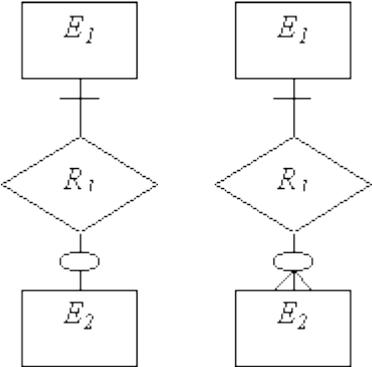
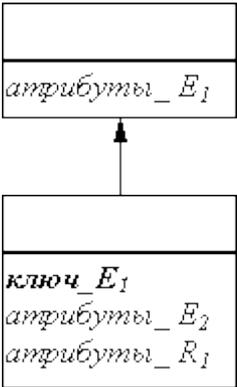
5. Если сущность имеет составной атрибут, то возможны два варианта: 1) составному свойству ставится в соответствие отдельное поле; 2) каждому из составляющих элементов составного свойства ставится в соответствие отдельное поле.

Выбор варианта зависит от характера обработки данных. При реализации запросов проще объединить поля, чем выделить часть поля. Если предполагается использование компонентов атрибута, лучше вариант 2, иначе – вариант 1.

### 6. Правила построения бинарных связей

Организация типовых связей при преобразовании инфологической модели в датологическую приведена в таблице 2.

Таблица 2

Тип связи	Пример связи	Правило построения отношений	Отношения
Тип 1 (1,1):(1,1)		Требуются только одно отношение. Первичным ключом данного отношения может быть ключ любой из сущностей.	
Тип 2 (1,1):(0,1) (1,1):(0,n)		Для каждой сущности создается свое отношение, при этом ключи сущностей служат ключами соответствующих отношений. Кроме того, ключ сущности с обязательным классом принадлежности добавляется в качестве внешнего	

		ключа в отношении, созданное для сущности с необязательным классом принадлежности .	
Тип 3 (0,1):(0,1)		Необходимо использовать три отношения: по одному для каждой сущности (ключи сущностей служат первичными ключами отношений) и одно отношение для связи. Отношение, выделенное для связи, имеет два атрибута - внешних ключа - по одному от каждой сущности.	
Тип 4 (0,1):(0,n) (0,1):(1,n)		Формируются три отношения: по одному для каждой сущности, причем ключ каждой сущности служит первичным ключом соответствующего отношения, и одно отношение для связи. Отношение, выделенное для связи, имеет два атрибута - внешних ключа - по одному от каждой сущности.	
Тип 5 n : m		В этом случае всегда используются три отношения: по одному для каждой сущности, причем ключ каждой сущности служит первичным ключом соответствующего отношения, и одно отношение для связи. Последнее отношение должно иметь среди своих атрибутов внешние ключи, по одному от каждой сущности.	

**Связь один-к-одному** между сущностями встречается редко. Если класс принадлежности обеих сущностей является обязательным, то для отображения обеих связанных сущностей можно использовать одну таблицу (тип 1).

Однако таким решением злоупотреблять не следует. Если для каждого объекта потребуются свои связи или в запросах потребуются информация по каждой сущности, то выбранное решение усложнит или замедлит работу с базой данных.

Если для каждой сущности создаются отдельные отношения, то информацию о связях можно отразить, включив в одно из отношений идентификатор из другого отношения (причем это можно сделать в любом из отношений)

Если класс принадлежности одной из сущностей является необязательным, то идентификатор сущности с необязательным классом добавляется в отношение, соответствующее сущности с обязательным классом принадлежности (тип 2).

Если класс принадлежности обеих сущностей является необязательным, то, чтобы избежать наличия пустых полей, следует использовать три отношения: по одному для каждой сущности и одно – для отображения связи между ними (тип 3).

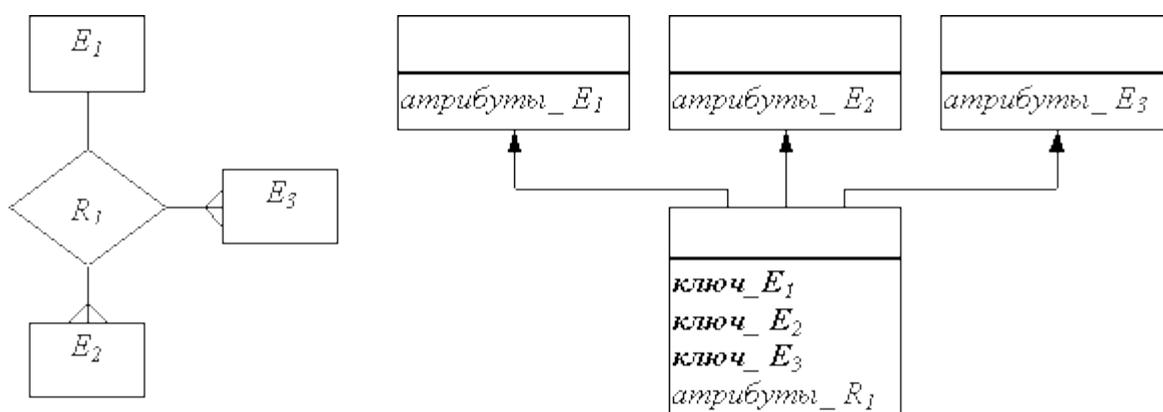
**Преобразование бинарной связи один-ко-многим (1:N)** зависит только от класса принадлежности N-связной сущности. Если он является обязательным, то можно использовать два отношения (по одному для каждой сущности). В отношение для N-связной сущности добавляется идентификатор 1-связной сущности (тип 2).

Если класс принадлежности N-связной сущности является необязательным, то для отображения связи создается третье отношение, которое будет содержать ключи каждой из связанных сущностей (тип 3).

**Для бинарной связи многие-ко-многим (M:N)** потребуются три отношения: по одному для каждой сущности и одно дополнительное – для

отображения связи между ними. Последнее отношение будет содержать идентификаторы связанных объектов. Ключ этого отношения будет составным (тип 4).

7. В случае N-арной связи необходимо использовать (n+1) отношение – по одному для каждой сущности, и одно для связи. Идентификатор каждой сущности станет первичным ключом соответствующего отношения. Отношение, порождаемое связью, будет иметь среди своих атрибутов ключи каждой сущности. Если связь имеет атрибуты, то они становятся атрибутами отношения связи.

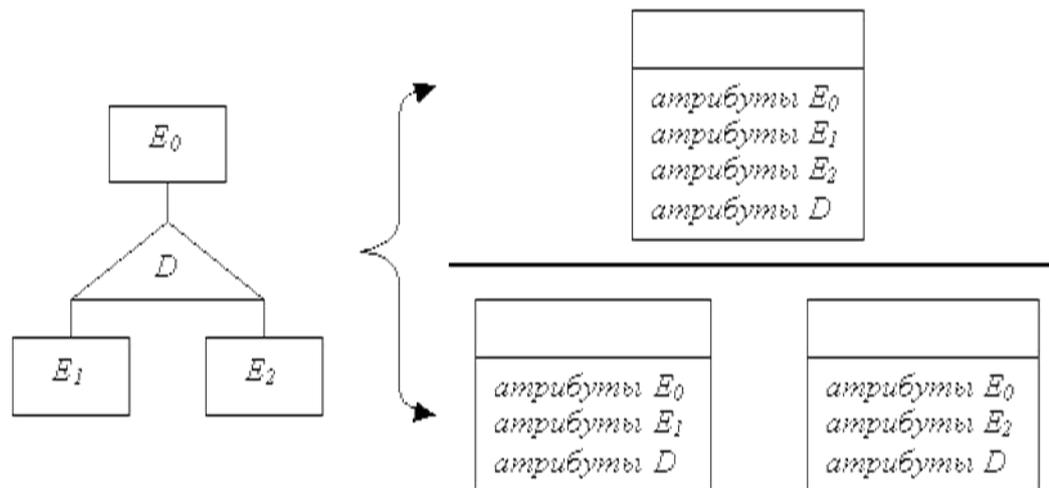


8. В случае иерархической связи необходимо возможны два варианта построения реляционных отношений.

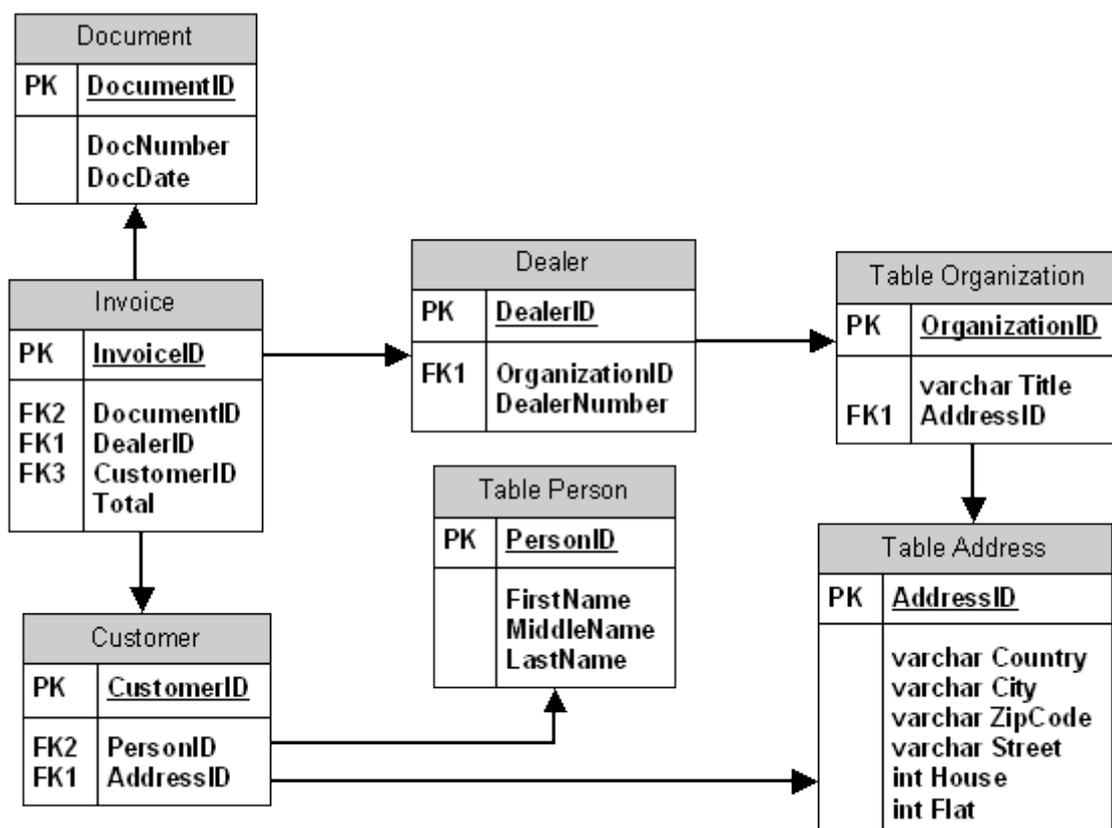
Согласно первому для иерархической структуры создается одно отношение, которое содержит атрибуты связи и всех сущностей. Недостаток такого способа - для каждого кортежа часть атрибутов всегда будет неопределенна.

По второму способу генерируется по одному отношению для каждой дочерней сущности. Каждое из этих отношений включает атрибуты родительской сущности и связи кроме атрибутов – дискриминантов. Недостатком данного способа является невозможность получить в одном запросе избыточное количество кортежей.

Оба описанных способа представлены на рисунке:



### Графическое изображение модели



Построенные таким образом реляционные отношения, не являются окончательной схемой базы данных. Их необходимо проверить на избыточные функциональные зависимости и привести к NFBK или нормальной форме более высокого порядка.

### **Задание для самостоятельной работы**

На основе спроектированной в лабораторной работе №2 инфологической модели предметной области необходимо спроектировать датологическую модель предметной области с использованием модели «сущность-связь».

### **Контрольные вопросы**

- 1) Что понимается под термином «отношение»?
- 2) В чем отличие отношения и таблицы
- 3) Что понимается под терминами «кортеж», «схема данных», «домен»?
- 4) Перечислите основные типы данных?
- 5) Что представляет собой реляционная модель данных?
- 6) Что представляет собой Логическое (датологическое) проектирование?
- 7) Перечислите правила преобразования инфологической модели в датологическую модель