**Лабораторная работа №7. WinRDBI.** Запросы в реляционной модели, в исчислениях на кортежах и доменах.

**Реляционная модель данных**

**A**

 |

[версия для печати](https://www.intuit.ru/intuit?destination=studies%2Fprofessional_skill_improvements%2F12128%2Fcourses%2F929%2Fprint_lecture%2F19322)

[< Лекция 3](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19320) || **Лекция 4**: **1**[2](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=2)[3](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=3)[4](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=4)[5](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=5)[6](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=6)[7](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=7) || [Лекция 5 >](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19324)

**Аннотация:**В этой лекции рассмотрим реляционную модель данных, в которой единственным источником данных являются отношения, может быть связанные между собой.

Для того, чтобы изучение реляционной алгебры не превратилось в скучный перечень операций, мы их вводим порциями, достаточными для получения какого-нибудь интересного результата.

Сначала определим операцию декомпозиции — разбиения заполненного отношения на части. Удивительно, но не всегда воссоединение компонент дает исходное отношение. В этом случае декомпозиция называется неполной. На языке алгебры декомпозиция и воссоединение компонент определяются, соответственно, операциями проекции и естественного соединения.

Оказывается, функциональные зависимости, определенные на отношениях, дают естественный язык для задания идентификаторов кортежей отношений (ключей) и для формулирования теоремы Хиса, позволяющей выделить допустимые декомпозиции.

Выделим набор независимых операций. Рассмотрим самую необычную для алгебры операцию переименования атрибутов.

Разберёмся с отображением модели "сущность-связь" в реляционную модель.

В самом конце главы перейдём к реализациям и рассмотрим соотношения между реляционной и табличными моделями.

К 70-м годам направление баз данных представляло мощную индустрию, в которой доминировали иерархические и сетевые модели.

К 1973-му году был принят стандарт CODASYL, определивший общепринятые особенности сетевой модели. Он определил понятия простого и составного элементов данных — аналогов простого поля записи и агрегата. Запись —это именованный агрегат, не входящий ни в какой другой агрегат. В записях можно использовать неопределённое значение. Определялось два типа агрегатов — "вектор", состоящий из простых элементов данных, и "повторяющаяся группа элементов". При этом повторяющаяся группа может образовываться элементами данных, векторами, агрегатами и другими повторяющимися группами.

Наборы представляют связи между типами записей. Как упоминалось в разделе 3.3.1, термин "набор" в модели CODASYL не совпадает с употребляемым нами термином "набор записей". Можно определить любое количество типов наборов между двумя типами записей. Тип набора — это двучленная конструкция, объединяющая тип записи владельца набора и типы записей-членов набора.

В это же время в 1970 г. появилась основополагающая работа Э.Ф.Кодда, в которой он применил к описанию баз данных алгебру отношений — реляционную алгебру. Появилась реляционная модель данных, представляющая базу данных как набор отношений, может быть связанных. Началась эра реляционных баз данных.

Можно спросить, чего же не хватало в существовавших моделях данных? Что обеспечило победное шествие реляционной модели, которому не помешало низкое быстродействие первых её реализаций? Иерархической и сетевой модели не хватало математической убедительности, то есть отсутствовала простая и красивая математическая модель. Это бы ещё полбеды. Плохо то, что в старых моделях было много обременительно лишнего. Язык манипулирования данными и язык запросов носили навигационный характер и требовали знания деталей структуры базы.

Принятие реляционной модели широким кругом пользователей и вендоров обеспечило развитие математической теории, и создание стандартов для её реализаций. Вот с реализациями получилась весьма интересная история. С некоторой стадии их развития математические идеи иссякли и теперь развитие языков манипулирования данными и запросов направляют другие движущие силы. Но об этом мы поговорим в главах, начиная с 8-й. Изменения вызванные практическими потребностями, настолько велики, что в настоящее время реляционные модели практически заменены тем, что в дальнейшее мы будем называть табличными моделями или моделями реляционного типа. Поэтому, кроме отношений в конце этой главы будем рассматривать таблицы.

Давайте разберёмся с основами реляционной модели данных.

**4.1 Отношения и их свойства. Схема и состояния отношения**

Основные структуры данных в реляционной модели — отношения и атрибуты.

Схему отношения можно определить через задание предиката. Спецификациям отношений соответствуют предикаты. При этом типу записи соответствует предикатный символ, а полям соответствуют аргументные места предикатного символа.

Пример предиката, соответствующего спецификации отдела приведен на [рисунке 4.1](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=1#image.4.1).

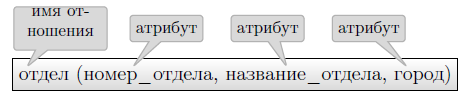


**Рис. 4.1.**Представление отношения предикатом

Каждой записи соответствует высказывание (на языке Пролог — факт), например, отдел(001, "Рога и копыта", "Нью-Москва")?

Принадлежность кортежа к отношению определяется истинностью описывающего его предиката.

Однако, в реляционной модели принято говорить об отношениях, которые имеют имя и набор свойств, называемых атрибутами ([рисунок 4.2](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=1#image.4.2)).



**Рис. 4.2.**Отношения и атрибуты

Под сигнатурой отношения будем понимать набор, состоящий из имени отношения, имён и типов его атрибутов. Отношения без атрибутов не рассматриваются.

*Свойство*. Данные в реляционной модели хранятся только в отношениях с конечных числом атрибутов. Других источников данных не существует.

Будем рассматривать отношения как наборы однотипных строк-кортежей. Пример: Отношение "Студент" с атрибутами "Фамилия", "Имя", "Отчество", "Дата\_рождения", "Студенческая\_группа", "Телефон".

Схема отношения:

Студент(Фамилия, Имя, Отчество, Дата\_рождения, Студенческая\_груп-па, Телефон)

Каждый студент представляется строкой значений своих атрибутов — кортежем, например,

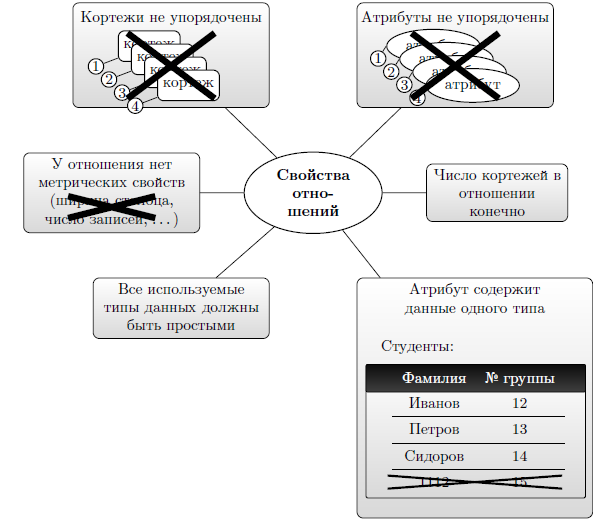
("Иванов", "Петр", "Сидорович", "22.05.82", 32, "111-0000, 222-0000").

Обратите внимание, на то, что последний атрибут это всего лишь строка "111-0000, 222-0000", и нельзя будет определить, есть ли у Иванова телефон 111-0000. С подстроками модель не работает.

Основные свойства отношений:

* кортежи не упорядочены;
* в отношении не может быть двух одинаковых кортежей;
* атрибуты не упорядочены;
* число кортежей в отношении конечно;
* любой атрибут отношения должен содержать данные одного типа (домена);
* все используемые типы данных простые;
* отношение не обладает метрическими свойствами, такими как ширина столбцов, число записей и т.д.

Свойства отношений для удобства запоминания представлены на [рисунке 4.3](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=1#image.4.3).



**Рис. 4.3.**Свойства

Обратите внимание на то, что простота типов данных вовсе не означает, что хранящиеся значения не имеют какой-нибудь структуры. Однако в рамках реляционной модели мы не можем эту структуру воспринять.

*Замечание.* В реализациях отсутствие упорядоченности может быть не удобным или не реализуемым, а метрические свойства всегда важны, поскольку они определяют быстродействие.

*Замечание.* Связей между отношениями, которые мы изучали в модели "сущность-связь", в реляционной модели нет. Связи определяются через ключевые атрибуты. В реализациях они задаются ограничениями целостности первичного ключа и внешнего ключа.

Состоянием отношения называется набор входящих в него кортежей.

Лекция 4:

**Реляционная модель данных**

**A**

 |

[версия для печати](https://www.intuit.ru/intuit?destination=studies%2Fprofessional_skill_improvements%2F12128%2Fcourses%2F929%2Fprint_lecture%2F19322)

[< Лекция 3](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19320) || **Лекция 4**: [1](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=1)**2**[3](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=3)[4](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=4)[5](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=5)[6](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=6)[7](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=7) || [Лекция 5 >](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19324)

**4.2 Операции проекции и соединения. Декомпозиция отношений.**

Изучим две операции над отношениями — проекцию и естественное соединение. Это позволит рассмотреть разбиение (декомпозицию) отношения на его проекции и восстановление исходного отношения путём соединения этих проекций. Обнаружатся неожиданные для начинающих неполные декомпозиции.

Описания всех остальных операций даны в разделе 4.6.

*Определение.* Проекция —это набор унарных операций выбора подмножества атрибутов отношений proj_X(r)projx(r), где R — схема отношения r и X\subseteq R — набор атрибутов.

Пример проекции отношения с атрибутами C_1, C_2, C_3 на подмножество \{C_1, C_2\} приведён ниже:

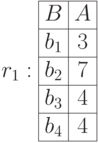
r: \begin{array}{|c|c|c|} \hline C_1 & C_2 & C_3 \\ \hline \end{array}

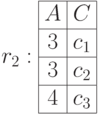
proj_{\{c1,c2\}}r: \begin{array}{|c|c|} \hline C_1 & C_2 \\ \hline \end{array}

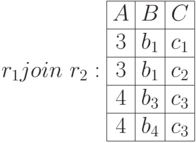
*Свойство.*Если Y\subseteq X\subseteq R, то proj_y(proj_x(r))=proj_y(r).

*Определение*(Естественное соединение). Пусть отношения r_1 и r_2 имеют схемыR_1(A_1,\dots,A_k,B_1,\dots,B_n) и R_2(A_1,\dots,A_k,C_1,\dots,C_m). Тогда естественное соединение отношений r_1 и r_2 есть отношение r_3 со схемой R_3(A_1,\dots,A_k,B_1,\dots,B_n,C_1,\dots,C_m), в котором каждая запись(экземпляр) получена конкатенацией каждой записи из r_1 с теми записями из r_2, у которых совпадают значения в общих атрибутах A_1,\dots,A_k.

Естественное соединение отношений r_1, r_2 с общим атрибутом A приведено ниже:



.



Обозначение естественного соединения: join(r_1,r_2) или join_{=A}(r_1,r_2) или r_1join r_2.

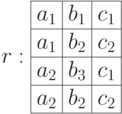
Рассмотрим декомпозицию отношения подробнее.

*Определение.* Полная декомпозиция отношения — это набор его проекций, соединение которых идентично отношению.

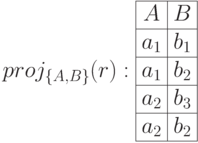
*Существуют неполные декомпозиции!*

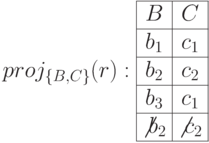
Пример полной декомпозиции приведен ниже.

Декомпозируемое отношение:



Проекции, дающие полную декомпозицию (вычеркиваем повторяющиеся записи):

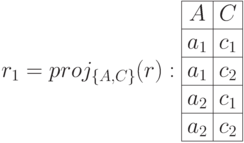


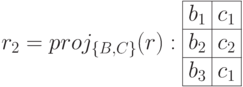


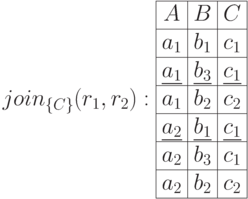
Для исходного отношения r_1 получены проекции на атрибуты \{A,B\} и \{B,C\}. Соединение их даст исходное отношение. Соедините проекции сами.

Для некоторых проекций декомпозиция оказывается неполной. Это означает появление так называемых присоединённых записей, которых не было в исходном отношении.

Соединение проекций r_1,r_2 того же отношения r создает присоединенные записи:







В восстановленном исходном отношении появились выделенные (подчерктутые) две записи, которых в исходном отношении r не было.

**4.3 Первичный ключ. Роль функциональных зависимостей**

*Уточнение понятия ключа*(Первичный ключ отношения). Атрибут или набор атрибутов, значения которых позволяют однозначно выбрать кортеж отношения, называется первичным ключом.

*Определение*. Ключ, состоящий из одного атрибута, называют простым.

*Определение*. Ключ, состоящий из двух или более атрибутов, называют составным или конкатенированным.

*Определение*(Суррогатный ключ). Суррогатным называется ключ, не имеющий прототипа в предметной области. Обычно он генерируется приложением.

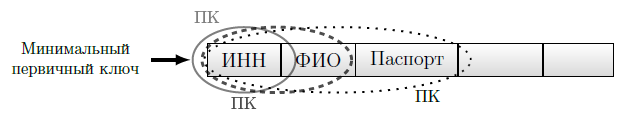
*Утверждение*. Из того, что кортежи отношений не повторяются, следует, что любое отношение имеет ключ.

Заметим, что в ключ могут войти все атрибуты отношения.

*Утверждение*. Пополнение ключа еще одним (не ключевым) атрибутом есть ключ.

*Определение*(Минимальный ключ). Если удаление одного атрибута лишает ключевой набор атрибутов статуса ключа, то такой ключ называют минимальным.

*Замечание*. Первичным ключом называют в действительности минимальный первичный ключ ([рисунок 4.4](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=2#image.4.4)).

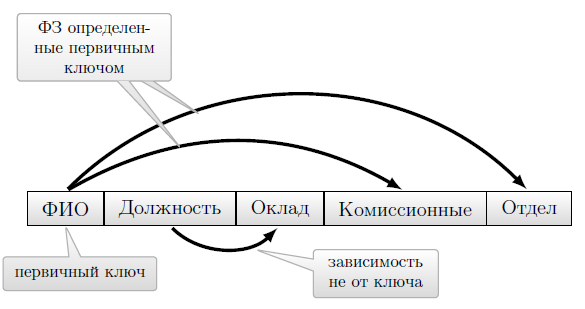


**Рис. 4.4.**Говорим "первичный ключ", а подразумеваем "минимальный первичный ключ"

*Замечание*. Изучая базы данных, следует понимать, что все приводимые примеры по необходимости сделаны простыми. Всегда следует ограничиваться контекстом примера и не пытаться рассуждать "как в жизни" или "как может быть". Так в отношении на [рисунке 4.4](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=2#image.4.5). можно было спросить: "А как быть с теми, у кого нет ИНН?". Так вот, в примере предполагается, что ИНН есть у всех и другие варианты не рассматриваются.

При проектировании реальных баз данных необходимые ориентиры дает спецификация модели бизнеса.

Функциональные зависимости (ФЗ) на отношениях и зависимости связанные с обобщениями функций играют важную роль в теории баз данных. В частности, ключи определяются через функции ([рисунок 4.5](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=2#image.4.5)).



**Рис. 4.5.**Функциональные зависимости на отношении

*Утверждение*. Любой атрибут находится в функциональной зависимости от ключа. Могут существовать и другие функциональные зависимости.

Лекция 4:

**Реляционная модель данных**

**A**

 |

[версия для печати](https://www.intuit.ru/intuit?destination=studies%2Fprofessional_skill_improvements%2F12128%2Fcourses%2F929%2Fprint_lecture%2F19322)

[< Лекция 3](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19320) || **Лекция 4**: [1](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=1)[2](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=2)**3**[4](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=4)[5](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=5)[6](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=6)[7](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=7) || [Лекция 5 >](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19324)

**4.4 Теорема Хиса**

Докажем теорему Хиса (Heath), устанавливающую связь между функциональными зависимостями в схеме отношения и способом его полной декомпозиции. Она определяет условия при которых декомпозиция будет полной.

*Теорема Хиса*. Пусть в отношении r со схемой R(S), где S — полный набор атрибутов отношения, выделены три набора атрибутов A,B,C таких что A \cap B=\not 0, A \cap C=\not 0, B \cap C=\not 0, A \cup B \cup C=S. Тогда, если набор С функционально зависит от В, то проекцииproj_{\{A,B\}}(r), proj_{\{B,C\}}(r) образуют полную декомпозицию отношения r.

*Доказательство*. Введем вспомогательное отношение

|  |  |
| --- | --- |
| r_1=proj_{\{A,B\}}(r) {join} {proj_{\{B,C\}}(r)} | ( 4.1) |

как соединение двух проекций отношения r. Покажем, что r_1\equiv r.

1. Выберем произвольную запись (кортеж) \langle{a,b,c}\rangle из r. Так как

\langle{a,b}\rangle \in proj_{\{A,B\}}(r), a

\langle{b,c}\rangle \in proj_{\{B,C\}}(r)

то по свойству операции соединения \langle{a,b,c}\rangle \in r_1.

Следовательно,

|  |  |
| --- | --- |
| r\subseteq r_1 | ( 4.2) |

1. Выберем произвольный кортеж \langle {a',b',c'}\rangle \in r_1. Из (4.1) следует, что

|  |  |  |  |
| --- | --- | --- | --- |
| \langle {a',b'}\rangle \in proj_{\{A,B\}}(r) | | ( 4.3) | |
| \langle {b',c'}\rangle \in proj_{\{B,C\}}(r) | ( 4.4) | |

1. Значит в r существуют записи

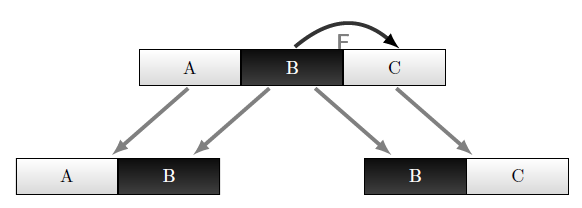
|  |  |  |
| --- | --- | --- |
| (4.3)\Rightarrow \langle {a',b',c}\rangle \in r | ( 4.5) | |
| (4.4)\Rightarrow \langle {a',b',c'}\rangle \in r | | ( 4.6) | |

1. Но С функционально зависит от В. Поэтому в кортежах (4.5) и (4.6) c=c', а значит запись \langle {a',b',c'}\rangle входит в отношение r, иначе говоря

|  |  |
| --- | --- |
| r_1\subseteq r | ( 4.7) |

1. Сопоставляя (4.2) и (4.7), получаем r_1\equiv r. Иначе говоря, декомпозиция (4.1) полная.

Мнемоническое изображение декомпозиции по теореме Хиса даёт [рисунок 4.6](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=3#image.4.6)



**Рис. 4.6.**Мнемоника для теоремы Хиса

**4.5 Структура и особенности реляционной алгебры**

Отношение r определяется своей схемой R. Набор записей в отношении определяет его состояние. При этом повторяющиеся кортежи в отношении отсутствуют.

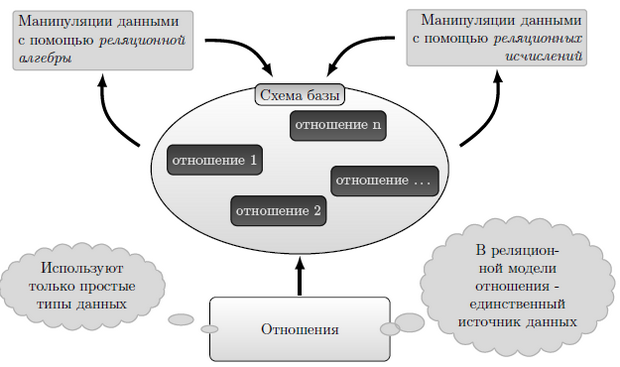
Как вы помните, операция в любой алгебре над некоторым множеством даёт результат, принадлежащий этому же множеству Реляционная алгебра строится над множеством отношений. Поэтому в реляционной модели при выполнении операций создаются новые, может быть временные, отношения на основе набора уже имеющихся отношений. Средств для создания отношений, не выводимых из имеющихся нет. Изменять состояния отношений, то есть заполнять их кортежами, в реляционной модели невозможно. Иначе говоря, реляционная алгебра —это модель запросов к базе.

Реляционная алгебра определяется на конечном множестве отношений с фиксированной сигнатурой и конечным числом кортежей. Поскольку сигнатуры отношений могут не совпадать, реляционная алгебра многосортна. Сами отношения и кортежи разных отношений могут быть не сравнимы.

Ещё раз обращаем внимание на то, что набор схем отношений предполагается заданным заранее. Реляционная алгебра не изменяет его и не может изменять состояние отношений, то есть вводить, удалять и изменять записи. Манипуляции данными создают врeменные, не сохраняемые отношения.

Обратите внимание на то, что в реляционной модели кроме алгебры могут использоваться исчисления, которыми мы займёмся в ["Языки реляционных баз данных"](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19328).

Сводка некоторых особенностей реляционной модели приведена на [рисунке 4.7](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=3#image.4.7).



**Рис. 4.7.**Некоторые особенности реляционной модели

**4.6 Операции реляционной алгебры**

Полный перечень операций реляционной алгебры:

* Проекция.
* Естественное соединение.
* \vartheta — соединение.
* Декартово произведение.
* Селекция.
* Булевы операции.
* Частное.
* Переименование атрибутов.

Все операции за исключением переименования работают с содержимым отношений. Результат операции есть новое отношение. Над ним так же можно выполнять операции.

Первые две операции, проекция и естественное соединение, позволили нам разобраться с декомпозицией отношений, сформулировать и доказать теорему Хиса. Изучим все остальные операции, не затрагивая пока переименования атрибутов.

**4.6.1 υ-соединение**

*Определение*. Пусть даны отношения r_1,r_2 со схемами R_1(A_1,\dots,A_k,B_1,\dots,B_l) и R_2(C_1,\dots,C_m,D_1,\dots,D_n) соответственно; \vartheta—оператор сравнения на группах атрибутов Аи С. Тогда \vartheta -соединение отношений r_1 и r_2 есть отношение r со схемой R_3(A_1,\dots,A_k,B_1,\dots,B_l,C_1,\dots,C_m,D_1,\dots,D_n), полученной объединением атрибутов схем R_1и R_2 без повторения. Записи r_3 получаются конкатенацией тех записей из r_1 и r_2, у которых значения группы столбцов А в r_1 и группы столбцов С в r_2 находятся в отношении \vartheta (удовлетворяют \vartheta).

Обозначение \vartheta -соединения: join_{A\vartheta C}(r_1,r_2).

Замечание. Очевидно, если в качестве \vartheta взять равенство "=" и A\equiv C, получим естественное соединение со схемой

R_3(A_1,\dots,A_k,B_1,\dots,B_l,D_1,\dots,D_n).

Рассмотрим пример \vartheta-соединения для отношений employee и salgrade, то есть "работник" и "категория оплаты":

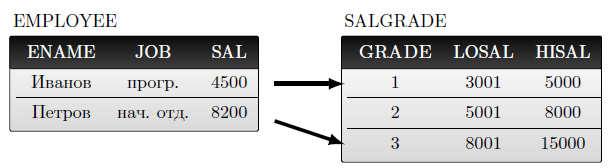
Отношение employee:

EMPLOYEE: \begin{array}{|c|c|c|} \hline ENAME & JOB & SAL \\ \hline \end{array}

Отношение salgrade:

SALGARDE: \begin{array}{|c|c|c|} \hline GRADE & LOSAL & HISAL \\ \hline \end{array}

По условию \vartheta=\mathbf{losal} \le \mathbf{sal} \le \mathbf{hisal} с вариантом заполнения (т.е. состоянием), изображённым на [рисунке 4.8](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=3#image.4.8), даст отношение, приведенное ниже. Здесь grade — категория оплаты, losal и hisal, соответственно, минимальная и максимальная зарплаты данной категории.



**Рис. 4.8.**Пример заполнения исходных отношений

Результат \vartheta-соединения:



Лекция 4:

**Реляционная модель данных**

**A**

 |

[версия для печати](http://www.intuit.ru/intuit?destination=studies%2Fprofessional_skill_improvements%2F12128%2Fcourses%2F929%2Fprint_lecture%2F19322)

[< Лекция 3](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19320) || **Лекция 4**: [1](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=1)[2](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=2)[3](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=3)[4](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=4)**5**[6](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=6)[7](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=7) || [Лекция 5 >](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19324)

**4.8 Независимые операции реляционной алгебры**

Для выделения независимых операций реляционной алгебры выделим базисный набор свойств операций, так чтобы все особенности операций отражались наиболее полно. Перечислим базисные свойства понятия "реляционная операция":

* увеличение количества атрибутов отношений;
* уменьшение количества атрибутов отношений;
* сохранение количества атрибутов отношений;
* сравнение атрибутов;
* работа с данными однотипных отношений;
* изменение имён.

Заметим, что независимость свойств и полнота этой системы свойств не доказаны.

*Утверждение*. Объединение, вычитание, декартово произведение, выборка и проекция независимые (примитивные) операции. Их нельзя выразить друг через друга.

Покажем это.

В самом деле, увеличивают число атрибутов декартово произведение и соединения.

Уменьшает число атрибутов только проекция. Поэтому она не выразима через остальные операции, не обладающие этим свойством.

Сравнивают атрибуты только селекция, выбирающая строки, и частное, основанное на выделении группы строк.

С данными нескольких однотипных отношений могут работать только булевы операции. Поэтому выделяем их в отдельную группу. Выбор группы строк основан на выборе строки по условию. Можно считать независимой более простую операцию селекции.

Изменение имён выполняет только операция переименования.

Разберёмся с булевыми операциями: объединением, пересечением и разностью. Кроме них сохраняют количество атрибутов отношения ещё селекция и пересечение. Но они не имеют теоретико-множественной природы. Так что выделение булевых операций обоснованно.

Используя известные свойства булевой алгебры, можно утверждать, что операции объединения и разности независимы.

В самом деле, пересечение определяется через вычитание A\cap B=A-(A-B), а частное через вычитание, декартово произведение и проекцию:

A\div B=proj_xA-proj_x((proj_xA\times B)-A)

Итак, заведомо независимыми будут проекция, селекция, частное и переименование.

Разберёмся с тройкой — декартово произведение, естественное и соединения. Операции соединения определяются через операции декартового произведения и выборки.

join_F(r_1,r_2)=sel_F(r_1\timesr_2)

Так что естественно считать независимой, в некотором смысле более простую, операцию декартово произведение.

**4.9 Определение реляционной алгебры**

Обозначим:

И — множество атрибутов (универсум),

D — множество доменов,

dom — полная функция dom : И \to D; назначает домен каждому атрибуту,

R — множество всех возможных схем отношений на И, \tau=\tau_1,\dots,\tau_p —есть множество отношений \tau_i со схемами R_ij, соответственно,

\theta —множество бинарных отношений, определенных на доменах из D, содержащее, по крайней мере, отношение равенства и неравенства для каждого домена.

*Определение*. Реляционной алгеброй над И, D,dom, K, \tau, \theta называется семиместный кортеж

B=(И,D,dom,K,\tau,\theta,O)

где O — это множество, содержащее операции селекции, проекции, объединения, пересечения, разности, частного, естественного и \vartheta-соединения, а также операцию переименования атрибутов из И.

Перечислим основные особенности реляционной алгебры:

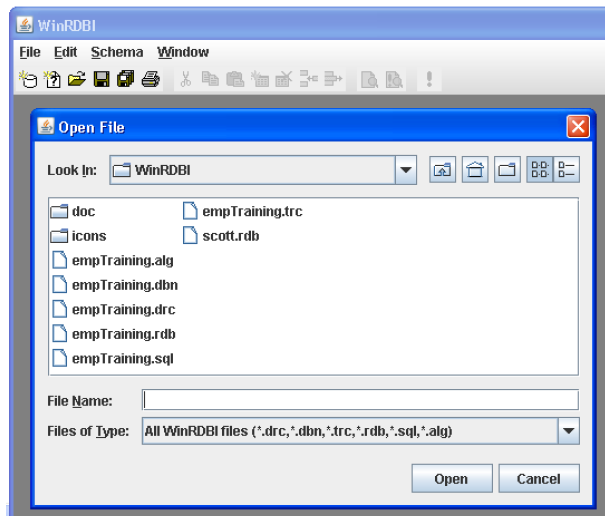
* Алгебра строится на множестве отношений с атрибутами.
* Элементы данных (атрибуты) типизированы (относятся к одному домену).
* Отношения можно соединять. Связей между отношениями, подобных рассмотренным в ["Модель сущность-связь"](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19318), не существует, так как по природе своей связи требуют активности системы и потому не являются алгебраической конструкцией.
* Используются операции над множествами (объединение, пересечение, вычитание, декартово произведение, частное) и реляционные операции (выборка (селекция), проекция, соединения, деление).
* Для выполнения некоторых операций необходимо обеспечить совместимость отношений по сигнатуре.
* В некоторых случаях операции требуют переименования атрибутов.
* Выражения реляционной алгебры строятся на отношениях и возвращают отношения же. Отношения-результаты можно использовать как аргументы в других выражениях.
* Независимость операций. Операции соединение, пересечение и деление можно выразить через другие реляционные операции. Операции объединение, вычитание, декартово произведение, выборка, проекция нельзя выразить друг через друга.
* Реляционная алгебра — это язык запросов. Выразить создание исходных отношений, заполнить их, изменить или удалить кортежи в этой алгебре нельзя.

**4.10 Примеры запросов**

Как вы помните, мы договорились, по всем разделам решать примеры, которые помогут в освоении материала. Для изучения запросов в реляционной алгебре будет использовано свободно доступное инструментальное средство WinRDBI, созданное в университете штата Аризона. В последующих главах мы будем использовать его для работы с запросами в исчислениях на кортежах и доменах. Можно поработать и в первой версии языка SQL, которая в существующих СУБД не используется.

**4.10.1 Устанавливаем WinRDBI**

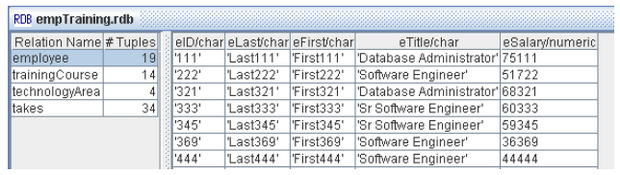
Скачиваем файл дистрибутива WinRDBISetup-3.1.0.130 с сайта <http://winrdbi.asu.edu/> и, щёлкнув по нему левой кнопкой мыши, инсталлируем WinRDBI. Скачиваем с сайта книги дополнительную схему базы данных, находящуюся в текстовом файле scott.rdb. Помещаем этот файл либо непосредственно в область баз данных показанную на [рисунке 4.9](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=5#image.4.9), либо в корневую папку WinRDBI, хранящую эти данные. На моей машине это папка C:\Program Files\WinRDBI. Чтобы загрузить схему scott.rdb, наберите File > Open и выберите путь к файлу scott.rdb. Файлы с расширением .rdb хранят базы данных. Изначально имеется файл empTraming.rdb, и мы только что добавили scott.rdb. Файл empTraining.alg хранит набор запросов в реляционной алгебре. В остальных файлах с именем erpTraining хранятся запросы в исчислениях и SQL.



**Рис. 4.9.**Файлы данных и запросов в WinRDBI

**4.10.2 Схема scott**

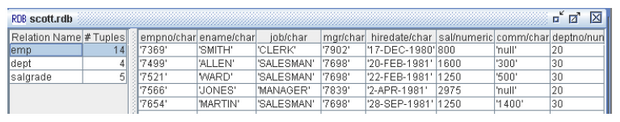
Рассмотрим отношение employee ([рисунок 4.10](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=5#image.4.10)), входящее в схему erpTraining. Атрибуты обозначены именами, начинающимися со строчной буквы, за которыми следует имя типа, отделённое знаком "/". Эти особенности связаны с тем, что WinRDBI написан на языке Пролог.



**Рис. 4.10.**Отношение employee

Формат текстового файла .rdb несложен. Вы можете разобраться в нём самостоятельно, прочитав файлы empTraining.rdb и scott.rdb с помощью текстового редактора. Строчные значения заключаются в одинарные кавычки.

В схеме scott ([рисунок 4.11](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=5#image.4.11)) заданы отношения emp, dept, salgrade, описанные на сайте книги.

[](http://www.intuit.ru/EDI/06_03_17_3/1488752433-23564/tutorial/959/objects/4/files/04_1.png)

[увеличить изображение](http://www.intuit.ru/EDI/06_03_17_3/1488752433-23564/tutorial/959/objects/4/files/04_1.png)  
**Рис. 4.11.**Отношение emp

Обратите внимание на то, что в отношении emp появились текстовые значения 'null'. Дело в том, что неопределённых значений в реляционной лгебре не существует. Поэтому NULL был заменён текстовой константой 'null', не эквивалентной ему по смыслу, а числовые значения атрибута comm пришлось заменить текстовыми, заключив их в кавычки.

**4.10.3 Обозначения операций**

Обозначения операций реляционной алгебры сведены в [таблицу 4.1](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=5#table.4.1)

|  |  |
| --- | --- |
| Таблица 4.1. Соответствие терминов реляционной и табличных моделей | |
| Операция реляционной алгебры | Операция в WinRDBI |
| Проекция proj_x(r) | project X (r) |
| Селекция sel_F(r) | select F (r) |
| Естественное соединение join(r,s), или join_{=A}(r,s), или r join s | r njoin s |
| \vartheta-соединение | не поддерживается |
| Декартово произведение t=r\times s | r product s |
| Частное r division s , или division(r,s), или r\div s | не поддерживается |
| Объединение r\cup s | r union s |
| Пересечение r\cap s | r intersect s |
| Разность r-s | r difference s |

\vartheta -соединение и частное не поддерживаются. Их необходимо выражать через другие операции. Выполните следующую последовательность File > Open > empTraining.alg. В появившейся панели изучите образцы запросов реляционной алгебры. Обратите внимание на то, что результат запроса —это временное отношение, и на нём, в свою очередь, можно выполнять запросы. Если результат запроса поименован, то после выполнения запроса, образованное отношение-ответ сохраняется и доступно для дальнейшей работы.

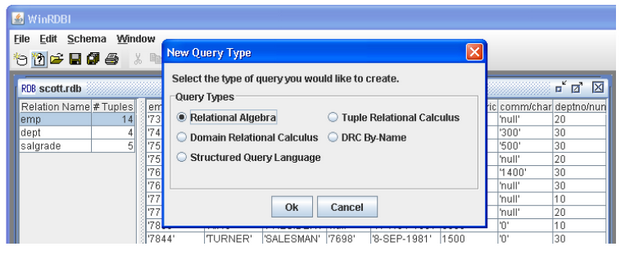
**4.10.4 Примеры запросов**

Для создания запроса необходимо выполнить следующую последовательность действий ([рисунок 4.13](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=5#image.4.13)):

1. Нажать кнопку

http://www.intuit.ru/EDI/06_03_17_3/1488752433-23564/tutorial/959/objects/4/files/04_12.png

1. Выбрать Relational Algebra.
2. Нажать Ok.

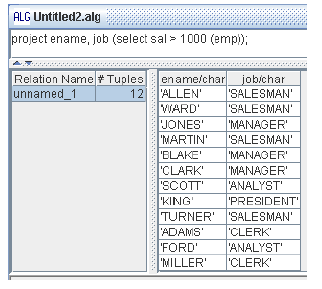
[](http://www.intuit.ru/EDI/06_03_17_3/1488752433-23564/tutorial/959/objects/4/files/04_13.png)

[увеличить изображение](http://www.intuit.ru/EDI/06_03_17_3/1488752433-23564/tutorial/959/objects/4/files/04_13.png)  
**Рис. 4.13.**Переход к выполнению запросов в реляционной алгебре

В появившейся панели можно набирать текст запроса.

Примеры запросов

1. Выдать фамилии и должности лиц, получающих зарплату больше 1000. [Рисунок 4.14](http://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=5#image.4.14).



**Рис. 4.14.**Результат запроса в WinRDBI

1. Выдать список сотрудников в виде отношения с атрибутами: empno, ename, job, dname. Запрос: project empno, ename, job, dname (emp njoin dept);

**4.11 Реляционная модель и модель сущность-связь**

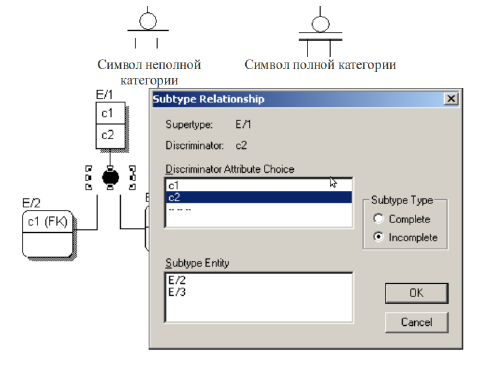
Мы уже освоили основы двух моделей данных — реляционной и сущность-связь. Начиная с этого раздела, каждый раз, изучив новую модель, будем устанавливать особенности её отображения в уже известные модели. Эти морфизмы позволят выработать единый взгляд на модели баз данных. Опыт преподавания показал, что только при таком подходе можно за ограниченное время освоить три и более моделей.

Займёмся теперь сравнением реляционной модели и модели сущность-связь. Прежде всего, обратим внимание на то, что обе модели работают на логическом уровне. На физическом уровне может работать только модель сущность-связь. На уровне аппаратной реализации обе модели бесполезны.

Основные отличия рассматриваемых моделей возникают из-за определения сильных и слабых сущностей и неотделимых от них идентифицирующей и неидентифицирующей связей.

При этом в ER-модели могут содержаться смыслы концептуального уровня, предназначенные только для человека и не реализуемые непосредственно в логической модели.

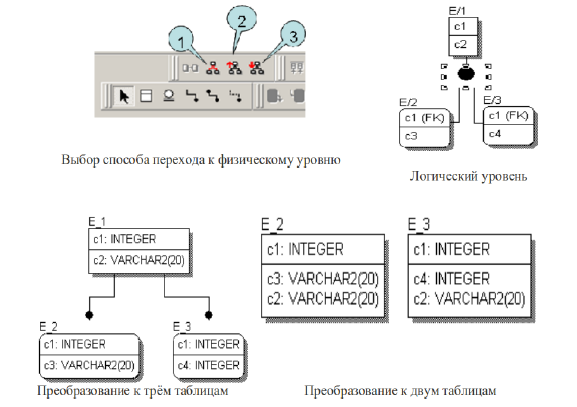
В расширенной ER-модели (EER-модель) определено ещё понятие категории, представляющей наследование в той мере, как его можно передать, не переходя к объектной системе. На логическом уровне принято изображать полную и неполную категории знаками, приведенными на [рисунке 4.15](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=6#image.4.15). Вверху изображена родительская сущность, а внизу дочерние. В полной категории перечислены все потомки, а в неполной только часть. В некоторых категориях могут существовать экземпляры сущностей предка, которые не являются экземплярами сущностей потомка.



**Рис. 4.15.**Категория

Для перехода на физический уровень существует три варианта. Рассмотрим их на примере с двумя потомками ([рисунок 4.16](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=6#image.4.16)):

1. Все три сущности преобразуются в одну таблицу, содержащую столбцы, позволяющие отметить, к какой из исходных сущностей относится строка таблицы.
2. Все сущности преобразуются каждая в свою таблицу
3. Промежуточный вариант, в котором создаются две таблицы, соответствующие двум потомкам.



**Рис. 4.16.**Сущности, связанные категорией на физическом уровне

Выбор варианта преобразования зависит от запросов, которые собираются выполнять.

Сильные и слабые сущности можно представить в реляционной модели через внешний ключ, входящий в состав первичного ключа или отнесённый к неключевым атрибутам. Атрибуты внешнего ключа должны совпадать с атрибутами первичного ключа другого отношения, по крайней мере, по типам, а возможно, и по именам.

Сильная сущность отличается тем, что внешний ключ не может входить в состав первичного ключа. Слабая сущность должна содержать внешний ключ в своём первичном ключе. Это обстоятельство и определяет зависимость слабой сущности от другой, связываемой с ней, сущности.

В ER-модели связь называют неидентифицирующей, если она переносит внешний ключ в область неключевых атрибутов. Идентифицирующая связь перенесёт внешний ключ в область ключевых атрибутов.

Неразрывность понятий сильной и слабой сущностей с неидентифици-рующей и идентифицирующей связями в реляционной модели выразить невозможно, так как отсутствует понятие агрегации, объединения отношений в структуру высшего порядка.

В следующей главе мы используем найденные соответствия для того, чтобы теорию нормализации, первоначально созданную в рамках реляционной модели, перенести в диаграммы "сущность-связь".

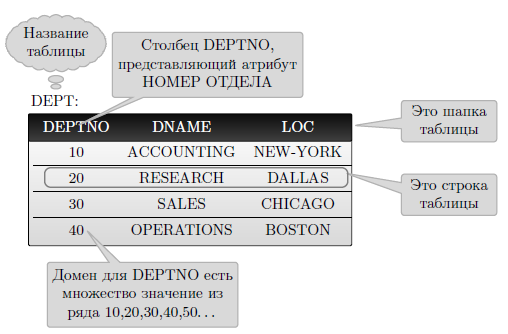
**4.12 Отношения и таблицы**

В настоящее время реляционная модель играет роль эталона и не используется в реализациях баз данных. Одна из причин в том, что построение запросов очень часто требует выполнения декартова произведения в качестве промежуточной операции. Для больших таблиц это приводит к созданию громадных промежуточных отношений и сильно замедляет запросы.

В реализациях баз данных на физическом уровне отношениям соответствуют плоские (реляционные) таблицы, которые имеют одну одноуровневую шапку и атомарные значения в ячейках таблицы. Кортежам отношения соответствуют строки таблицы.

Существует соответствие кортежей отношения строкам представляющей его таблицы. Поскольку строки реляционных таблиц могут повторяться, то взаимно однозначного соответствия из таблицы в отношение не существует.

Пример реляционной таблицы приведен на [рисунке 4.17](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=6#image.4.17).

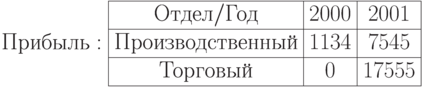


**Рис. 4.17.**Реляционная таблица

Мы уже установили, что состояние отношения определяется набором входящих в него кортежей. Ещё раз отметим, что состояние отношения в реляционной теории не рассматривается. Пример состояния отношения "Сотрудники" :



Ниже приведен пример нереляционной таблицы с двумя шапками, верхней и боковой:



Самостоятельно преобразуйте её в реляционную таблицу.

Основные отличия таблиц от отношений:

* В отношении нет одинаковых кортежей. Таблицы без первичного ключа могут содержать одинаковые строки. Если, например, в таблице "сотрудник" выбирается единственный столбец "но-мер\_отдела", то в таблице-результате часть строк будет повторена. Устранение повторов может изменить смысл полученного результата.
* Тело отношения есть множество, и потому кортежи не упорядочены. Строки таблиц могут быть упорядочены. В этом случае отношение можно реализовать таблицами, отличающимися порядком строк.
* Атрибуты отношения определяются с уникальными в пределах отношения именами и потому не нуждаются в упорядочении. Столбцы таблиц могут быть упорядочены. Одно отношение можно реализовать таблицами со столбцами, записанными в разном порядке. В реализациях порядок столбцов может влиять на быстродействие, хотя и незначительно.
* Отношения не имеют метрических свойств. Для таблиц они важны. Размеры столбцов и число строк определяют быстродействие запросов и инструкций манипуляции данными.

[Таблица 4.2](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=6#table.4.2) — это краткий словарь перевода с языка реляционной модели на язык табличной модели.

|  |  |
| --- | --- |
| Таблица 4.2. Соответствие терминов реляционной и табличных моделей | |
| Термин реляционной модели | Термин табличной модели |
| Схема базы | Схема базы |
|  | Создание схемы базы |
|  | Ввод строки |
|  | Обновление строки |
|  | Удаление строки |
| Отношение | Таблица (реляционного типа) |
| Заголовок отношения | Заголовок таблицы |
| Тело отношения | Тело таблицы |
| Атрибут отношения | Столбец таблицы |
| Кортеж отношения | Строка таблицы |
| Арность отношения | Количество столбцов |
| Типы данных и домены | Типы данных и домены |
|  | Ширина столбца |
|  | Количество строк |
|  | Размер таблицы |
|  | Время исполнения |

Обратите внимание на то, что в реализациях появляются команды определения и манипулирования данными "создание схемы", "ввод строки" и т.д. Вводится несколько терминов, позволяющих охарактеризовать размеры базы и быстродействие запросов "ширина столбца", "количество строк" и т.д. Это требует изучения и учёта ещё одного уровня модели, который мы в разделе 1.3 условно назвали уровнем аппаратной реализации.

Лекция 4:

**Реляционная модель данных**

**A**

 |

[версия для печати](https://www.intuit.ru/intuit?destination=studies%2Fprofessional_skill_improvements%2F12128%2Fcourses%2F929%2Fprint_lecture%2F19322)

[< Лекция 3](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19320) || **Лекция 4**: [1](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=1)[2](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=2)[3](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=3)[4](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=4)[5](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=5)[6](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19322?page=6)**7** || [Лекция 5 >](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19324)

**4.13 Об алгебре Дейта и Дарвена и перспективах развития реляционной модели**

Реляционная алгебра Кодда, один из вариантов которой мы изучили, удобна тем, что семантика языка SQL, который мы рассмотрим в ["Язык SQL"](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19330)основывается именно на этой алгебре. Существует ряд расширений алгебры Кодда.

Дейтом и Дарвеном был предложен новый вариант реляционной алгебры, с минимальным набором базисных операций. В него входят реляционное дополнение, реляционная конъюнкция (или дизъюнкция) и проекция (удаление атрибута). Использование реляционного дополнения возможно, поскольку предполагается, что домен имеет конечную мощность. Реляционная конъюнкция понимается как некоторое обобщение естественного соединения определённого в алгебре Кодда. Операция переименования, понимаемая как замена имени ровно одного из атрибутов отношения, оказывается зависимой. С помощью константного отношения, представляющего таблицу перехода значений при замене имён атрибута, переименование представляется через операции реляционной конъюнкции и удаления атрибута.

Можно показать, что новая алгебра полна, то есть все операции алгебры Кодда выражаются через операции алгебры Дейта и Дарвена.

Важным преимуществом нового варианта реляционной алгебры является более точная связь с логикой. Однако, для наших целей достаточно результата Джекобса, показавшего, что и иерархическая, и сетевая, и реляционная модели основываются на многосортной логике. Изучить основы алгебры Дейта и Дарвена лучше всего по книгам Кузнецова С.Д., например, [Кузнецов С.Д. Основы баз данных: курс лекций: учебное пособие. - М.: Интернет-Ун-т Информ. Технологий, 2005. - 488 с.].

Исходя из множественности математических моделей понятия "реляционная модель данных" можно предположить, что и другие понятия в базах данных и вообще в информатике могут иметь несколько экспликаций (так называется методологический прием замещения неточного понятия или представления точным понятием). Важно выбрать в некотором смысле "минимальный", либо наиболее удобный, как-то "согласованный" вариант. Это обстоятельство следует иметь в виду, рассматривая в дальнейшем отображения моделей данных.

Как известно, в последние двадцать лет в компьютерных науках наметился разрыв между работой теоретиков и практическими реализациями [Парнас Д.Л. Реальное переосмысление "формальных методов". Оригинал: David Lorge Parnas. Really Rethinking "Formal Methods". Computer (IEEE Computer Society), V. 43, No 1, January 2010]. Есть основания полагать, что главная причина такого положения дел в том, что для правильного понимания перспектив использования теоретических построений в практике, следует учитывать следующие пять аспектов:

* свойства (вид, тип) моделей, возможность их представления математическими структурами;
* лингвистическая компонента, в частности, учёт того, что языки моделей, языки программирования — это ещё и искусственные языки, распространённые в определённых группах людей;
* особенности аппаратных реализаций моделей;
* стоимость работ, связанных с переходом на новые модели;
* влияние маркетинговых соображений.

Как отмечено Парнасом, ". . . многие исследовательские работы пишутся так, будто только математика и имеет значение". Многие практически важные модели слишком сложны и, главное, не позволяют эффективно обрабатывать информацию. Это ограничивает их возможности. Как это ни тривиально, но модель должна быть адекватной с учётом более одного из упомянутых пяти аспектов.

Учёт лингвистической компоненты, точнее когнитивных аспектов, связанных с особенностями восприятия искусственного языка человеком, позволяет оптимизировать деятельность самого слабого звена в системах разработки программного обеспечения, то есть человека.

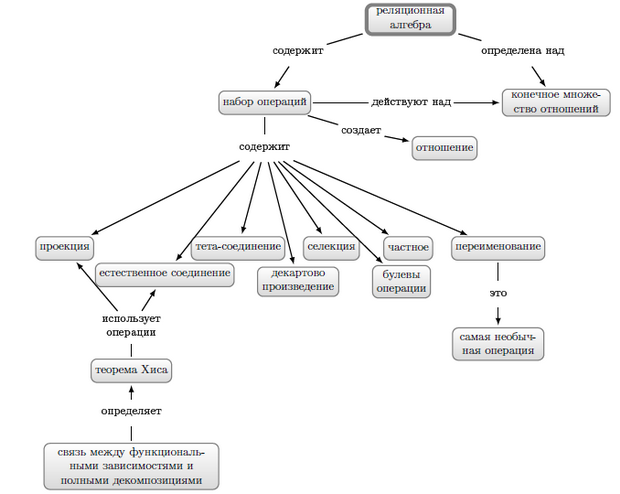
Особенности реализации, в том числе аппаратные, могут быть критическими. Так, в реализации алгебры Кодда широкое использование декартова произведения приводило к созданию промежуточных таблиц таких размеров, что от этого подхода давно отказались.

И, наконец, если мы говорим о реализациях, то необходимо понимать, что они выполняются конкретными организациями, работающими в конкурентной среде и требуют вполне определённых материальных затрат. Трудно представить вендора, который заменяет старую отлаженную СУБД на принципиально новую и тем самым обрекает себя на повторение пути, который уже был пройден и позволил решить массу частных проблем.

Так что шансы модели Дейта-Дарвена на реализацию следует оценить как незначительные. Что касается методов выработанных в маркетинге, то видимо подход, основанный на выделении сегментов рынка и установлении соответствия продукта некоторым сегментам, по-видимому, можно перенести на разработки моделей. Конечно, в этой ситуации следует рассматривать не рынок в собственном смысле слова, а предметную область, а сегменты рынка заменить набором парадигм, действующих и возможных, задав их определяющие свойства.

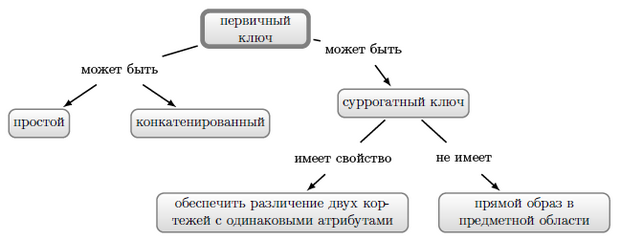
**4.14 Основные понятия главы**

К определению реляционной алгебры:

[](https://www.intuit.ru/EDI/06_03_17_3/1488752433-23564/tutorial/959/objects/4/files/04_18.png)

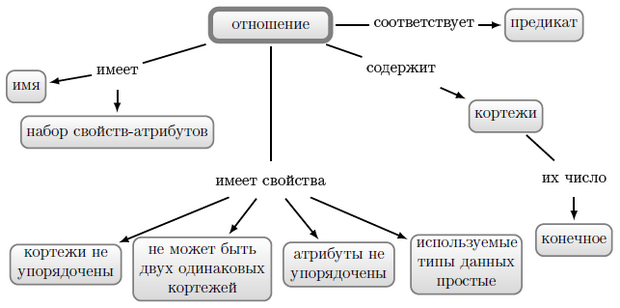
[увеличить изображение](https://www.intuit.ru/EDI/06_03_17_3/1488752433-23564/tutorial/959/objects/4/files/04_18.png)  
**Рис. 4.18.**Определение реляционной алгебры

Понятие ключа было достаточно подробно исследован в ["Модель сущность-связь"](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19318). Семантическая сеть для этого понятия приведена в разделе 2.3. Напомним, что, говоря "первичный ключ", мы на самом деле имеем в виду "минимальный первичный ключ".



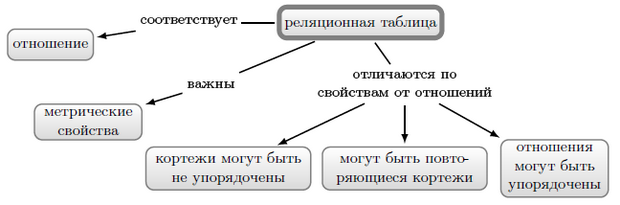
**Рис. 4.19.**Первичный ключ

Отношения и их связи с таблицами, которые представляют отношения в реализации баз данных.



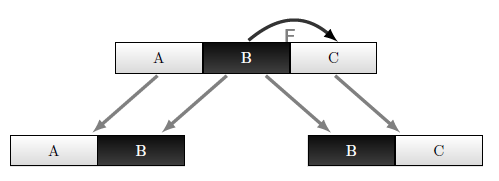
**Рис. 4.20.**Отношения и их связи с таблицами базы данных

Число кортежей и атрибутов предполагается конечным. Важно помнить, что отношение не обладает метрическими свойствами, такими как число кортежей и размер значения атрибутов. Для таблиц, представляющих отношения, возможны упорядочения, а метрические свойства исключительно важны. Они определяют объём памяти, время выполнения команд.



**Рис. 4.21.**Свойства таблиц в базе данных

Мнемоника для теоремы Хиса в виде таких "шортиков":



**Рис. 4.22.**Мнемоника для теоремы Хиса

**[Дальше >>](https://www.intuit.ru/studies/professional_skill_improvements/12128/courses/929/lecture/19324)**